Sergei N. Artemov
Anil Nerode (Eds.)

LNCS 4514

Logical Foundations of Computer Science

International Symposium, LFCS 2007
New York, NY, USA, June 2007
Proceedings

Springer

Sergei N. Artemov   Anil Nerode (Eds.)

Logical Foundations of Computer Science

International Symposium, LFCS 2007
New York, NY, USA, June 4-7, 2007
Proceedings

Springer

Volume Editors

Sergei N. Artemov
CUNY Graduate Center
Computer Science
365 Fifth Ave., New York City, NY 10016, USA
E-mail: Sartemov@gc.cuny.edu

Anil Nerode
Cornell University
Department of Mathematics
545 Malott Hall, Ithaca NY 14853, USA
E-mail: anil@math.cornell.edu

Library of Congress Control Number: 2007927056

CR Subject Classification (1998): F.4.1, F.4, F.3, I.2.2-4

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-540-72732-9 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-72732-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12067930 06/3180 5 4 3 2 1 0

Preface

The Symposium on Logical Foundations of Computer Science series provides a forum for the fast-growing body of work in the logical foundations of computer science, e.g., those areas of fundamental theoretical logic related to computer science. The LFCS series began with "Logic at Botik," Pereslavl-Zalessky, 1989, which was co-organized by Albert R. Meyer (MIT) and Michael Taitslin (Tver). After that, organization passed to Anil Nerode.

Currently, LFCS is governed by a Steering Committee consisting of Anil Nerode, Cornell (General Chair); Stephen Cook, Toronto; Dirk van Dalen, Utrecht; Yuri Matiyasevich, St. Petersburg; John McCarthy, Stanford; J. Alan Robinson, Syracuse; Gerald Sacks, Harvard; and Dana Scott, Carnegie-Mellon.

The 2007 Symposium on Logical Foundations of Computer Science took place in New York, USA at the Graduate Center of the City University of New York during June 4 - 7. This volume contains the extended abstracts of talks selected by the Program Committee for presentation at LFCS 2007.

The scope of the symposium is broad and contains constructive mathematics and type theory; logical foundations of programming; logical aspects of computational complexity; logic programming and constraints; automated deduction and interactive theorem proving; logical methods in protocol and program verification; logical methods in program specification and extraction; domain theory logics; logical foundations of database theory; equational logic and term rewriting; lambda and combinatory calculi; categorical logic and topological semantics; linear logic; epistemic and temporal logics; intelligent and multiple agent system logics; logics of proof and justification; nonmonotonic reasoning; logic in game theory and social software; logic of hybrid systems; distributed system logics; system design logics; other logics in computer science.

We thank the authors and reviewers for their contributions. We acknowledge the support of the Graduate Center of the City University of New York, the Mid-Atlantic Mathematical Logic Seminar, the New York Logic Colloquium, and the CUNY Computer Science Colloquium.

We are grateful to Evan Goris, Bryan Renne, and Roman Kuznets for preparing this volume for Springer.

March 2007

Anil Nerode
Sergei Artemov

Organization

Steering Committee

Stephen Cook (Toronto)
Dirk van Dalen (Utrecht)
Yuri Matiyasevich (St. Petersburg)
John McCarthy (Stanford)
Anil Nerode (Cornell) - General Chair
J. Alan Robinson (Syracuse)
Gerald Sacks (Harvard)
Dana Scott (Carnegie-Mellon)

Program Committee

Samson Abramsky (Oxford)
Sergei Artemov (New York) - PC Chair
Matthias Baaz (Vienna)
Lev Beklemishev (Moscow)
Andreas Blass (Ann Arbor)
Lenore Blum (Carnegie-Mellon)
Samuel Buss (San Diego)
Thierry Coquand (Göteborg)
Ruy de Queiroz (Recife, Brazil)
Denis Hirschfeldt (Chicago)
Bakhadyr Khoussainov (Auckland)
Yves Lafont (Marseille)
Joachim Lambek (McGill)
Daniel Leivant (Indiana)
Victor Marek (Kentucky)
Anil Nerode (Cornell) - General LFCS Chair
Philip Scott (Ottawa)
Anatol Slissenko (Paris)
Alex Simpson (Edinburgh)
V.S. Subrahmanian (Maryland)
Michael Rathjen (Leeds)
Alasdair Urquhart (Toronto)

Additional Reviewers

Alexandru Baltag
Jan Broersen

Table of Contents

Justified and Common Knowledge: Limited Conservativity

Evangelia Antonakos

CUNY Graduate Center
Ph.D. Program in Mathematics
365 Fifth Avenue
New York, NY 10016, USA
Eva@Antonakos.net

Abstract. We consider the relative strengths of three formal approaches to public knowledge: "any fool" knowledge by McCarthy (1970), Common Knowledge by Halpern and Moses (1990), and Justified Knowledge by Artemov (2004). Specifically, we show that epistemic systems with the Common Knowledge modality C are conservative with respect to Justified Knowledge systems on formulas $\chi \wedge C\varphi \rightarrow \psi$, where χ, φ, and ψ are C-free.

Keywords: justified knowledge, common knowledge, Artemov, conservative.

1 Multi-agent Logics

The logics T_n, $\mathsf{S4}_n$, and $\mathsf{S5}_n$ are logics in which each of the finitely many (n) agents has a knowledge operator K_i which is T, or $\mathsf{S4}$, or $\mathsf{S5}$ respectively. We only consider cases where all agents' modalities are of the same logical strength.

Definition 1. *The formal systems for* T_n, $\mathsf{S4}_n$, *and* $\mathsf{S5}_n$ *are as follows:*
Propositional Logic *plus for* K_i, $i = 1, 2, \ldots, n$ *we have*
Axioms for $\mathsf{S4}_n$**:**

$\mathsf{K} : K_i(\varphi \rightarrow \psi) \rightarrow (K_i\varphi \rightarrow K_i\psi)$	each agent can do *modus ponens*
$\mathsf{T} : K_i\varphi \rightarrow \varphi$	agents can know only true propositions
$4 : K_i\varphi \rightarrow K_iK_i\varphi$	agents have positive introspection

Rules:

Necessitation: $\vdash \varphi \Rightarrow \vdash K_i\varphi$, *for* $i = 1, 2, \ldots, n$

For T_n, *omit the final axiom.*
For $\mathsf{S5}_n$, *add negative introspection:* $\neg K_i\varphi \rightarrow K_i\neg K_i\varphi$.

Definition 2. *Kripke models for* $\mathsf{S4}_n$*:* $M = \langle W,\ R_1,\ R_2, \ldots,\ R_n,\ \Vdash\rangle$ *where*

- W *is a non-empty set of worlds*
- $R_i \subseteq W \times W$ *is agent* i*'s accessability relation.* R_i *is reflexive and transitive.*

• *$\Vdash \subseteq W \times Var$ where Var is the set of propositional variables. The forcing relation $\Vdash$ is naturally extended to all formulas so that R_i corresponds to K_i:*

$$M, u \Vdash K_i\varphi \Leftrightarrow (\forall v \in M)[uR_iv \to M, v \Vdash \varphi] \ .$$

For T_n-models, each R_i is reflexive while for $\mathsf{S5}_n$-models, each R_i is an equivalence relation.

Theorem 1. *T_n, $\mathsf{S4}_n$, and $\mathsf{S5}_n$ are sound and complete with respect to their models (cf.* [10]*).*

Multi-agent systems are enhanced by the addition of modalities which take into account shared or public knowledge of agents. Three such modalities C, J, and O will be discussed, all of which model variations of public information. We will compare their logical strengths, semantics, and complexity and will see why Justified Knowledge (J) systems are sufficient to solve classical epistemic scenarios, a role usually designated for Common Knowledge (C).

2 Common Knowledge

The most recognized concept of public knowledge is common knowledge, and the literature addressing it, both philosophical and mathematical, is vast. The initial investigation was philosophical: Lewis's book [15] on convention. The intuition behind the informal definition of common knowledge below derives from Aumann's oft-cited [5], where it was used in the context of agents having common priors. McCarthy's 'any fool' operator of 1970 ([10], p. 13) is closely related to common knowledge and his systems in [16] (see section 4 of this paper) may be the first to address it axiomatically . Rigorous work on common knowledge in the context of multi-agent systems was done by Halpern and Moses in [13] (an expansion of a 1984 work of the same title) and Lehmann [14]. Much of the work by Halpern and Moses appears in [10]. Common knowledge continues to be actively investigated.

Informally, the epistemic operator $C\varphi$, to be read 'φ is common knowledge,' can be given as infinite conjunction:

$$C\varphi \leftrightarrow \varphi \wedge E\varphi \wedge EE\varphi \wedge EEE\varphi \wedge E^4\varphi \wedge \cdots \wedge E^n\varphi \wedge \cdots \tag{1}$$

where $E\varphi = K_1\varphi \wedge K_2\varphi \wedge \cdots \wedge K_n\varphi$ ('everyone knows φ') and K_i is an individual agent's knowledge operator corresponding to T, $\mathsf{S4}$ or $\mathsf{S5}$ as appropriate. One formal characterization which [10] and [7] take is via the Fixed Point Axiom

$$C\varphi \leftrightarrow E\,(\varphi \wedge C\varphi) \tag{2}$$

and the Induction Rule

$$\frac{\varphi \to E\,(\varphi \wedge \psi)}{\varphi \to C\psi} \tag{3}$$

yielding Common Knowledge to be the greatest fixed point solution to $X \leftrightarrow E(\varphi \wedge X)$ [10]. Common Knowledge does not take into account the means by which the knowledge is acquired. As we will see, this is in contrast to Justified Knowledge. The distinction between the infinite conjunction, the fixed point axiom, and how common knowledge is achieved is addressed in [6]. [12] too, provides a survey with examples but does not include a distinct formalism. There is also an equivalent axiomatic formulation of common knowledge which replaces the induction rule with the induction axiom in [17], which, for technical convenience, we will use.

Definition 3. T_n^C*,* $\mathsf{S4}_n^C$*, and* $\mathsf{S5}_n^C$ *axiom systems:*
Propositional Logic *plus*
Axioms:

T*,* $\mathsf{S4}$*, or* $\mathsf{S5}$ *axioms for* K_i*,* $i = 1, 2, \dots, n$*, respectively;*
K: $C(\varphi \to \psi) \to (C\varphi \to C\psi)$;
T: $C\varphi \to \varphi$;
$C\varphi \to E(C\varphi)$*, where* $E\varphi = K_1\varphi \wedge K_2\varphi \wedge \dots \wedge K_n\varphi$;
Induction Axiom: $\varphi \wedge C(\varphi \to E\varphi) \to C\varphi$.

Rules:

Necessitation: $\vdash \varphi \Rightarrow \vdash K_i\varphi$ *, for* $i = 1, 2, \dots, n$
Necessitation: $\vdash \varphi \Rightarrow \vdash C\varphi$.

Definition 4. *Models for* T_n^C*,* $\mathsf{S4}_n^C$*, and* $\mathsf{S5}_n^C$*:* $M = \langle W, R_1, R_2, \dots, R_n, R_C, \Vdash\rangle$ *where*

- $M = \langle W, R_1, R_2, \dots, R_n, \Vdash\rangle$ *is a* T_n*,* $\mathsf{S4}_n$*, or* $\mathsf{S5}_n$ *model, respectively*
- $R_C = (\bigcup_{i=1}^{n} R_i)^*$*, that is the transitive closure of all the agents' relations*
- *The forcing relation* $\Vdash$ *is extended to all formulas so* R_C *corresponds to* C*:*

$$M, u \Vdash C\varphi \Leftrightarrow (\forall v \in M)[uR_Cv \to M, v \Vdash \varphi] \ .$$

Theorem 2. T_n^C*,* $\mathsf{S4}_n^C$*, and* $\mathsf{S5}_n^C$ *are sound and complete with respect to their models (cf.* [10], p. 70ff, [17], p. 47ff*).*

The agents' logic plays a role in determining the strength of the common knowledge operator C. In the systems defined above, C is always at least as strong as K_i. Showing that in T_n^C, $\mathsf{S4}_n^C$, and $\mathsf{S5}_n^C$, C satisfies the T, $\mathsf{S4}$, and $\mathsf{S5}$ axioms, respectively, is given as an exercise in [10], p. 93.

3 Justified Knowledge

Justified Knowledge was introduced by Artemov in [3,4] as the forgetful projection of the *evidence-based* knowledge represented by an appropriate adaptation of LP (Logic of Proofs). In LP systems ($\mathsf{T}_n\mathsf{LP}$, $\mathsf{S4}_n\mathsf{LP}$, $\mathsf{S5}_n\mathsf{LP}$), each formula / subformula carries with it a proof term representing a particular proof of the formula / subformula from the axioms. Justified knowledge systems are ones in which all proofs are identified as one. Whereas $C\varphi$ asserts that φ is common knowledge, $J\varphi$ asserts that φ is common knowledge arising from a proof of φ or some other agreed-upon acceptable set of evidences. Though the proof of φ is not explicitly presented with the assertion $J\varphi$, it is reproducible. This is the important Realization Theorem which provides an algorithm to reconstruct LP proof terms. For more details on this, the reader should consult [4].

As with the common knowledge logics, the construction of the justified knowledge logics T_n^J, $\mathsf{S4}_n^J$, and $\mathsf{S5}_n^J$ builds on the multi-agent logics. In C systems the agents' logic determines the strength of C while in J systems the strength of J is chosen independently to be weaker, stronger, or the same as that of the agents'. In the aforementioned logics, the modality J will be assumed to be $\mathsf{S4}$ unless otherwise specified.

Definition 5. T_n^J, $\mathsf{S4}_n^J$, *and* $\mathsf{S5}_n^J$ *axiom systems:*
Propositional Logic *plus*
Axioms:

T, $\mathsf{S4}$, *or* $\mathsf{S5}$ *axioms for* K_i, $i = 1, 2, \ldots, n$;
$\mathsf{S4}$ *axioms for* J ;
Connection Principle: $J\varphi \rightarrow K_i\varphi$.

Rules:

Necessitation for all K_i: $\vdash \varphi \Rightarrow \vdash K_i\varphi$;
Necessitation for J: $\vdash \varphi \Rightarrow \vdash J\varphi$.

Definition 6. *Models for* T_n^J, $\mathsf{S4}_n^J$, *and* $\mathsf{S5}_n^J$: $M = \langle W, R_1, R_2, \ldots, R_n, R_J, \Vdash\rangle$ *where*

- $M = \langle W, R_1, R_2, \ldots, R_n, \Vdash\rangle$ *is a* T_n, $\mathsf{S4}_n$, *or* $\mathsf{S5}_n$ *model, respectively*
- $R_J \subseteq W \times W$ *is reflexive and transitive relation such that* $R_J \supseteq (\bigcup_{i=1}^{n} R_i)^*$ *(where* * *is transitive closure)*
- *The forcing relation* $\Vdash$ *is extended to all formulas so* R_J *corresponds to* J:

$$M, u \Vdash J\varphi \Leftrightarrow (\forall v \in M)[uR_Jv \rightarrow M, v \Vdash \varphi] \ .$$

Theorem 3. T_n^J, $\mathsf{S4}_n^J$, *and* $\mathsf{S5}_n^J$ *are sound and complete with respect to their models, as shown in* [4].

Recall that in common knowledge models, $R_C = (\bigcup_{i=1}^{n} R_i)^*$ and so $R_C \subseteq R_J$. Thus in a context where we can compare the two, i.e. a hybrid model with both R_C and R_J, it seems (if φ contains no Js) $J\varphi \Rightarrow C\varphi$ but not vice versa. More formally, we have the following proposition.

Definition 7. *Let* φ^* *be* φ *with each instance of a* J *replaced by a* C.

Proposition 1. $(\mathsf{S4}_n^J)^* \subset \mathsf{S4}_n^C$ *but* $(\mathsf{S4}_n^J)^* \neq \mathsf{S4}_n^C$.

Proof. It needs to be shown that the $*$-translation of each each rule and axiom of $\mathsf{S4}_n^J$ is provable in $\mathsf{S4}_n^C$. Artemov shows this in [4] using the equivalent axiomatization of $\mathsf{S4}_n^C$ from [10]. It is only the Induction Axiom of $\mathsf{S4}_n^C$ which is not provable in $(\mathsf{S4}_n^J)^*$, yielding the strict inclusion. □

The case in which the J of $\mathsf{S5}_n^J$ is an $\mathsf{S5}$ modality, that is $\mathsf{S5}_n^{J(\mathsf{S5})}$, is considered in [18] (where she names it $\mathsf{S5}_n\mathsf{S5}$). An $\mathsf{S5}_n^{J(\mathsf{S5})}$-model is like an $\mathsf{S5}_n^J$ model except that R_J will now be an equivalence relation.

Corollary 1. *Let* I.A. *be the induction axiom. Then*

$$\begin{aligned}&\mathsf{S4}_n^C \equiv (\mathsf{S4}_n^J)^* + \mathsf{I.A.}\ ,\\ &\mathsf{T}_n^C \equiv (\mathsf{T}_n^J)^* + \mathsf{I.A.}\ ,\\ &\mathsf{S5}_n^C \equiv (\mathsf{S5}_n^{J(\mathsf{S5})})^* + \mathsf{I.A.}\ .\end{aligned}$$

Proof. The strict inclusion of the J systems follows from Proposition 1 and noticing that C satisfies the $\mathsf{4}$ axiom in T_n^C and the $\mathsf{5}$ axiom in $\mathsf{S5}_n^C$. When the induction axiom is added, the equivalence is clear. □

Indeed, from Corollary 1, in any of the justified knowledge systems mentioned, $J\varphi \Rightarrow C\varphi^*$.

The evidence-based common knowledge semantics for J systems are further enriched by Artemov's Realization Theorem mentioned at the start of the section. This gives a constructive approach to recovering or realizing the full proof terms of the evidence-based knowledge systems.

Theorem 4 (Realization Theorem). *There is an algorithm that, given an* $\mathsf{S4}_n^J$*-derivation of a formula* φ*, retrieves an* $\mathsf{S4}_n\mathsf{LP}$*-formula* ψ*, a* realization of φ*, such that* φ *is* ψ°*, where* $^\circ$ *replaces all proof terms with* J*, and* $\mathsf{S4}_n\mathsf{LP}$ *proves* ψ*.*

This theorem and a realization theorem for $\mathsf{S5}_n^J$ (where J is an $\mathsf{S4}$-modality) is established in [4] while a realization theorem for $\mathsf{S5}_n^{J(\mathsf{S5})}$ is given in [18].

Other major advantages to justified knowledge are

- proofs in $\mathsf{S4}_n^J$ are normalizable ([4]), but those in $\mathsf{S4}_n^C$ are not
- $\mathsf{S4}_2^J$ is *PSPACE*-complete [9], whereas for $n \geq 2$, $\mathsf{S4}_n^C$ is *EXPTIME*-complete [10].

These features have been exploited by Bryukhov in [8] to develop an automated theorem prover for $\mathsf{S4}_n^J$. Justified Knowledge offers simpler, more constructive, and more automation-friendly approach to common knowledge.

4 Any Fool's Knowledge

McCarthy's model of common knowledge via "any fool knows" apparently goes back to roughly 1970 ([10], p. 13), though its first published appearance is in [16]. In this epistemic multi-agent system, the modality for each agent is denoted by S, with an additional virtual agent, "any fool" denoted by O. In [16] p. 2, whatever any fool knows, "everyone knows that everyone else knows," and so someone knows. Thus we may add an additional axiom linking the fool to the other people: $O\varphi \to S\varphi$. Call this the linking axiom. This corresponds exactly to Artemov's connection principle: $J\varphi \to K_i\varphi$. When McCarthy et al. use subscripted modals, S_i, to specify individual agents, S_0 is the distinguished any fool operator O. Thus we see that the "fool" is a particular agent, hence in any axiom, we may replace all S modals by Os, though not vice versa.

Definition 8. *The McCarthy et al. axioms are based on propositional logic plus:*

linking axiom: $O\varphi \to S\varphi$
K0: $S\varphi \to \varphi$
K1: $O(S\varphi \to \varphi)$
K2: $O(O\varphi \to OS\varphi)$
K3: $O(S\varphi \wedge S(\varphi \to \psi) \to S\psi)$
K4: $O(S\varphi \to SS\varphi)$
K5: $O(\neg S\varphi \to S\neg S\varphi)$.

We will look at three systems identified in [16] given by axioms K0-K3, K0-K4, and K0-K5.[1] These will be referred to as MT, $\mathsf{M4}$, and $\mathsf{M5}$ respectively. Model semantics and completeness results for a variant of $\mathsf{M5}$ is stated in [16]. From this and Lemma 3 below, it follows that $\mathsf{S5}_n^{J(\mathsf{S5})}$ is also sound and complete.

These logics immediately lend themselves to epsitemic scenarios, of which Wise Men and Unfaithful Wives are addressed in [16]. These particular axioms do not seem to be built on standard formulations of modal logics yet we can see that Artemov's justified knowledge operator J plays a role equivalent to McCarthy's any fool operator O. In particular, we have the following theorem.

Definition 9. *Let $\varphi^\star$ be φ with each instance of a J replaced by an O and each K_i replaced by an S_i.*

Theorem 5. $(\mathsf{T}_n^J)^\star \equiv \mathsf{MT}$, $(\mathsf{S4}_n^J)^\star \equiv \mathsf{M4}$, *and* $(\mathsf{S5}_n^{J(\mathsf{S5})})^\star \equiv \mathsf{M5}$.

Proof. Immediate from the following three lemmas. □

Lemma 1. $(\mathsf{T}_n^J)^\star \equiv \mathsf{MT}$.

Proof. Recall that J is an $\mathsf{S4}$ modality while the K_i are T modalities.

($\Leftarrow$) To show $(\mathsf{T}_n^J)^\star \supset \mathsf{MT}$, T_n^J must satisfy MT axioms (K0-K3 and the linking axiom), where Os are Js and S_is are K_is.

Linking axiom: $\mathsf{T}_n^J \vdash J\varphi \to K_i\varphi$;	the connection principle
K0: $\mathsf{T}_n^J \vdash K_i\varphi \to \varphi$;	T axiom for K_i
K1: $\mathsf{T}_n^J \vdash J(K_i\varphi \to \varphi)$;	J necessitation of T axiom of K_i
K2: $\mathsf{T}_n^J \vdash J(J\varphi \to JK_i\varphi)$;	

[1] In [16], K0 is omitted from these lists. Given other statements in the paper, this clearly is just an oversight.

1. $\mathsf{T}_n^J \vdash J\varphi \to JJ\varphi$ — $\mathsf{4}$ axiom for J
2. $\mathsf{T}_n^J \vdash J\varphi \to K_i\varphi$ — the connection principle
3. $\mathsf{T}_n^J \vdash J(J\varphi \to K_i\varphi)$ — from 2. by J necessitation
4. $\mathsf{T}_n^J \vdash JJ\varphi \to JK_i\varphi$ — from 3. by K axiom for J
5. $\mathsf{T}_n^J \vdash J\varphi \to JK_i\varphi$ — from 1. and 4.
6. $\mathsf{T}_n^J \vdash J(J\varphi \to JK_i\varphi)$ — from 5. by J necesitation

K3: $\mathsf{T}_n^J \vdash J(K_i\varphi \wedge K_i(\varphi \to \psi) \to K_i\psi)$; — J necessitation of K axiom for K_i.

As mentioned above, "any fool" is a particular agent and so in any axiom all the Ss may be replaced by Os. Consider K0′-K3′ and the linking axiom′ where we do just that:

Linking axiom′: $\mathsf{T}_n^J \vdash J\varphi \to J\varphi$; — propositional tautology
K0′: $\mathsf{T}_n^J \vdash J\varphi \to \varphi$; — T axiom for J
K1′: $\mathsf{T}_n^J \vdash J(J\varphi \to \varphi)$; — J necessitation of T axiom of J
K2′: $\mathsf{T}_n^J \vdash J(J\varphi \to JJ\varphi)$; — J necessitaton of $\mathsf{4}$ axiom for J
K3′: $\mathsf{T}_n^J \vdash J(J\varphi \wedge J(\varphi \to \psi) \to J\psi)$; — J necessitation of K axiom for J.

($\Rightarrow$) $(\mathsf{T}_n^J)^\star \subset \mathsf{MT}$. We must show that MT satisfies the $(\mathsf{T}_n^J)^\star$ axioms and rules. Remember that "any fool" O is a particular S agent.

S axioms:
K: $\mathsf{MT} \vdash S\varphi \wedge S(\varphi \to \psi) \to S\psi$; — by K3, linking axiom, K0
T: $\mathsf{MT} \vdash S\varphi \to \varphi$; — K0

O axioms:
T: $\mathsf{MT} \vdash O\varphi \to \varphi$; — by K0, O is a particular S
K: $\mathsf{MT} \vdash O\varphi \wedge O(\varphi \to \psi) \to O\psi$; — by K axiom for S, O is a an S
$\mathsf{4}$: $\mathsf{MT} \vdash O\varphi \to OO\varphi$; — by K2, T axiom for O, O is an S

Connection axiom: $\mathsf{MT} \vdash O\varphi \to S\varphi$; — the linking axiom

O necessitation: This follows from the fact that each S and O axiom is nesessitated.
K axiom for S and O is necessitated by K3.
T axiom for S and O is necessitated by K1.
$\mathsf{4}$ axiom for O is necessitated by K2.
S necessitation: This follows from O necessitation and the linking axiom. □

Lemma 2. $(\mathsf{S4}_n^J)^\star \equiv \mathsf{M4}$.

Proof. Recall that J and K_i are $\mathsf{S4}$ modalities.
($\Leftarrow$) $(\mathsf{S4}_n^J)^\star \supset \mathsf{M4}$ follows from Lemma 1 and
K4: $\mathsf{S4}_n^J \vdash J(K_i\varphi \to K_iK_i\varphi)$; — by J necessitation of $\mathsf{4}$ axiom for K_i
K4′ = K2′

($\Rightarrow$) $(\mathsf{S4}_n^J)^\star \subset \mathsf{M4}$ follows from Lemma 1 and
S axioms:
$\mathsf{4}$: $\mathsf{M4} \vdash S\varphi \to SS\varphi$; — by K4, T axiom for O.
O necessitation: $\mathsf{4}$ axiom for S is necessitated by K4. □

Lemma 3. $(\mathsf{S5}_n^{J(S5)})^\star \equiv \mathsf{M5}$.

Proof. Recall that J and K_i are $\mathsf{S5}$ modalities.

($\Leftarrow$) $(\mathsf{S5}_n^{J(S5)})^\star \supset \mathsf{M5}$ follows from Lemma 2 and
K5: $\mathsf{S5}_n^{J(S5)} \vdash J(\neg K_i\varphi \to K_i\neg K_i\varphi)$; by J necessitation of $\mathsf{5}$ axiom for K_i.
K5′: $\mathsf{S5}_n^{J(S5)} \vdash J(\neg J\varphi \to J\neg J\varphi)$; by J necessitation of $\mathsf{5}$ axiom for J.

($\Rightarrow$) $(\mathsf{S5}_n^{J(S5)})^\star \subset \mathsf{M5}$ follows from Lemma 2 and
S axioms:
$\mathsf{5}$: $\mathsf{M5} \vdash \neg S\varphi \to S\neg S\varphi$; by K5, T axiom for O
O axioms:
$\mathsf{5}$: $\mathsf{M5} \vdash \neg O\varphi \to O\neg O\varphi$; by K5, T axiom for O, O is an S.
O necessitation: $\mathsf{5}$ axiom for O and S necessitated by K5. □

Lemma 3 completes the proof of Theorem 5. Despite quite different motivations and technical backgrounds, McCarthy's "any fool" and Artemov's justified knowledge approaches lead to the same multi-modal logics.

Corollary 2. *There is a Realization Theorem for* MT*,* $\mathsf{M4}$*, and* $\mathsf{M5}$ *providing evidence-based semantics for McCarthy's "any fool" knowledge operator* O*.*

5 Limited Conservativity

A logic T with language $\mathcal{L}$ is a conservative extension of a logic T' with language $\mathcal{L}' \subseteq \mathcal{L}$ if for sentences φ of $\mathcal{L}'$, T proves φ if and only if T' proves φ. Recall the definition for * from Section 3 which renames J to C. As the logics $(\mathsf{S4}_n^J)^*$ and $\mathsf{S4}_n^C$ have the same language and yet are not equal, it is clear that $\mathsf{S4}_n^C$ can not be a conservative extension of $(\mathsf{S4}_n^J)^*$, it is however a conservative extension over all formulas in which C occurs only negatively. We say that a symbol or subformula X occurs negatively in a formula F if, when F is rewritten to have no implication symbols, X is in the scope of a negation symbol (or an odd number of negations). For example, X occurs only negatively in these first two formulas and both positively and negatively in the last: $X \to Y$, $(\neg(A \wedge X) \to B) \to Y$, $A \wedge X \to B \vee X$.

Theorem 6. *If φ is a formula of* $\mathsf{S4}_n^J$ *such that all occurrences of J in φ are negative, then* $\mathsf{S4}_n^J \vdash \varphi \quad \Leftrightarrow \quad \mathsf{S4}_n^C \vdash (\varphi)^*$.

In some sense this result is tight as the induction axiom $(\varphi \wedge J(\varphi \to E\varphi) \to J\varphi)$ which distinguishes $(\mathsf{S4}_n^J)^*$ from $\mathsf{S4}_n^C$ has, along with a negative occurrence of J, a single positive occurrence of J.

Proof. ($\Rightarrow$) is secured by the inclusion $(\mathsf{S4}_n^J)^* \subset \mathsf{S4}_n^C$ of Proposition 1.

($\Leftarrow$) This direction is a consequence of the Main Lemma which follows. We show this direction by proving the contrapositive. Suppose φ is a formula of $\mathsf{S4}_n^J$ such that all occurrences of J in φ are negative and $\mathsf{S4}_n^J \nvdash \varphi$. By completeness, there is a model M and a world x such that $M, x \Vdash \neg\varphi$. By the Main Lemma, $M^C, x \Vdash^C \neg(\varphi)^*$, hence $\mathsf{S4}_n^C \nvdash (\varphi)^*$, since M^C (with R_J ignored) is a model for $\mathsf{S4}_n^C$. □

Lemma 4 (Main Lemma). *Let M be a $\mathsf{S4}_n^J$-model. Add the relation R_C of reachability along $R_1, \ldots, R_n$ to M and get the augmented model M^C, where $\Vdash^C$ coincides with $\Vdash$ on variables, and the modality C corresponds to R_C. Let φ be a formula of $\mathsf{S4}_n^J$. Then*

if all occurrences of J in φ are positive, then $x \Vdash \varphi \quad \Rightarrow \quad x \Vdash^C (\varphi)^*$;

if all occurrences of J in φ are negative, then $x \nVdash \varphi \quad \Rightarrow \quad x \nVdash^C (\varphi)^*$.

Proof. By induction on φ.

Base case is secured by the definition of $\Vdash^C$.

Boolean case: $\varphi \equiv \psi \to \theta$.

Subcase: all occurrences of J in φ are positive and $x \Vdash \varphi$. Then $x \nVdash \psi$ or $x \Vdash \theta$. In the former case all occurrences of J in ψ are negative and, by the induction hypothesis, $x \nVdash^C (\psi)^*$. In the latter case all occurrences of J in θ are positive and, by the induction hypothesis, $x \Vdash^C (\theta)^*$. In either case, $x \Vdash^C (\varphi)^*$.

Subcase: all occurrences of J in φ are negative and $x \nVdash \varphi$. Then $x \Vdash \psi$ and $x \nVdash \theta$. Since all occurrences of J in ψ are positive and all occurrences of J in θ are negative, by the induction hypothesis, $x \Vdash^C (\psi)^*$ and $x \nVdash^C (\theta)^*$, hence $x \nVdash^C (\varphi)^*$.

Case: $\varphi \equiv K_i\psi$.

Subcase: all occurrences of J in φ are positive and $x \Vdash \varphi$. Then all occurrences of J in ψ are positive and $y \Vdash \psi$, for all y such that xR_iy. By the induction hypothesis, $y \Vdash^C (\psi)^*$, for all y such that xR_iy, hence $x \Vdash^C (K_i\psi)^*$, i.e., $x \Vdash^C (\varphi)^*$.

Subcase: all occurrences of J in φ are negative and $x \nVdash \varphi$. Then for some y such that xR_iy, $y \nVdash \psi$. Since all occurrences of J in ψ are also negative, by the induction hypothesis, $y \nVdash^C (\psi)^*$, hence $x \nVdash^C (K_i\psi)^*$, i.e., $x \nVdash^C (\varphi)^*$.

Case: $\varphi \equiv J\psi$.

Subcase: all occurrences of J in φ are positive and $x \Vdash \varphi$. Then all occurrences of J in ψ are also positive and $y \Vdash \psi$, for all y such that xR_Jy. Since $R_C \subseteq R_J$, $y \Vdash \psi$, for all y such that xR_Cy. By the induction hypothesis, $y \Vdash^C (\psi)^*$, for all y such that xR_Cy. Hence $x \Vdash^C C(\psi)^*$, i.e., $x \Vdash^C (J\psi)^*$, i.e., $x \Vdash^C (\varphi)^*$.

Subcase: 'all occurrences of J in φ are negative and $x \nVdash \varphi$' is impossible, since $\varphi \equiv J\psi$ and the displayed occurrence of J is positive in $J\psi$. □

Corollary 3. *If χ, φ and ψ are formulas in the language of $\mathsf{S4}_n$, then* $\mathsf{S4}_n^C \vdash \chi \wedge C\varphi \to \psi \Leftrightarrow \mathsf{S4}_n^J \vdash \chi \wedge J\varphi \to \psi$.

Proof. As per Theorem 6, $\chi \wedge J\varphi \to \psi$ has J only in negative position. □

Corollary 4. *If χ, φ and ψ are formulas in the language of T_n, then* $\mathsf{T}_n^C \vdash \chi \wedge C\varphi \to \psi \Leftrightarrow \mathsf{T}_n^J \vdash \chi \wedge J\varphi \to \psi$.

Proof. Analogous to the proof of Theorem 6 if the Main Lemma starts with T_n^J-models (J is an $\mathsf{S4}$ modality) and completeness for T_n^J. $\square$

Corollary 5. *If χ, φ and ψ are formulas in the language of $\mathsf{S5}_n$, then* $\mathsf{S5}_n^C \vdash \chi \wedge C\varphi \to \psi \Leftrightarrow \mathsf{S5}_n^{J(\mathsf{S5})} \vdash \chi \wedge J\varphi \to \psi$.

Proof. Analogous to the proof of Theorem 6. if the Main Lemma starts with $\mathsf{S5}_n^{J(\mathsf{S5})}$-models ($J$ is an $\mathsf{S5}$ modality) and completeness for $\mathsf{S5}_n^{J(\mathsf{S5})}$. $\square$

In fact, in Corollary 5, the justified knowledge system can be weakened to $\mathsf{S5}_n^J$, where J is an $\mathsf{S4}$ modality. Note that in the proof of Theorem 6, the case of $\varphi \equiv J\psi$ requires only that $R_C \subseteq R_J$ and not that R_J be of the same logical strength. The proof will actually go through with any modality whose accessibility relation contains R_C. However, if J is to be knowledge, R_J is semantically required to be reflexive and transitive. Thus for all formulas with only negative occurences of C, $\mathsf{S5}_n^C$ is a conservative extension of $(\mathsf{S5}_n^J)^*$.

6 Conclusions

This conservativity of C over J limited to formulas with C in negative position would seem to lend itself to uses of J in place of C in situations in which common knowledge is applied or assumed, rather than derived or concluded.

We have also seen that Artemov's evidence-based approach to common knowledge leads to the same multi-modal logic systems as McCarthy's 'any fool' axiomatic approach. This points towards applications of J and endows the O systems with a constructive, evidence-based semantics via the Realization Theorem.

We may also care to consider whether conservativity holds for a larger class of formulas and what benefits there may be to considering a logic which contains both J and C modalities.

Acknowledgements

Gratitude is due to Sergei Artemov for his guidance and for pointing me towards this problem. Thanks are due to Melvin Fitting for providing a preprint of his article referenced and to those who offered suggestions to drafts of this paper, including the reviewers of this volume.

References

1. L. Alberucci, G. Jaeger. About cut elimination for logics of common knowledge. *Annals of Pure and Applied Logic*, 133:73-99, 2005.
2. S. Artemov. Explicit provability and constructive semantics. *Bulletin of Symbolic Logic,* 7(1): 1-36, 2001.
3. S. Artemov. Evidence-Based Common Knowledge. Technical Report TR-2004018, CUNY Ph.D. Program in Computer Science, 2004.
4. S. Artemov. Justified Common Knowledge. *Theoretical Computer Science*, 357 (1-3): 4-22, 2006.
5. R. J. Aumann. Agreeing to Disagree. *Annals of Statistics*, 4(6): 1236-1239, 1976.
6. J. Barwise. Three Views of Common Knowledge. TARK 1988: 365-379.

7. J. van Benthem, D Sarenac. The Geometry of Knowledge. ILLC Report PP-2004-21, University of Amsterdam, 2004.
8. Y. Bryukhov. *Integration of decision procedures into high-order interactive provers.* Ph.D. thesis, CUNY Graduate School, 2005.
9. S. Demri. Complexity of Simple Dependent Bimodal Logics. Laboratoire LEIBNIZ-CNRS, U.M.R. 5522, manuscript, 2005.
10. R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning About Knowledge.* MIT Press, 1995.
11. M. Fitting. Modal Proof Theory. In *Handbook of Modal Logic*, Patrick Blackburn and Johan van Benthem and Frank Wolter, editors, Elsevier, 2006.
12. J. Geanakoplos. Common Knowledge. TARK 1992: 254-315.
13. J. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37(3): 549-587, 1990. (A preliminary version appeared in 1984.)
14. D. Lehmann. Knowledge, common knowledge, and related puzzles. *Proc.3rd ACM Symposium on Principles of Distributed Computing*, 62-67, 1984.
15. D. Lewis. *Convention, A Philosophical Study.* Harvard University Press, 1969.
16. J. McCarthy, M. Sato, T. Hayashi, and S. Igarishi. On the Model Theory of Knowledge. Technical Report STAN-CS-78-657, Stanford University, 1978.
17. J.-J. Ch. Meyer and W. van der Hoek. *Epistemic Logic of AI and Computer Science.* Cambridge Tracts in Theoretical Computer Science, 41. Cambridge University Press, 1995.
18. N. Rubtsova. On realization of $\mathsf{S5}$ modalitiy by evidence terms. *Journal of Logic and Computation*, 16: 671-684, 2006.

The Intensional Lambda Calculus

Sergei Artemov[1] and Eduardo Bonelli[2]

[1] Graduate Center CUNY, PhD Program in Computer Science, 365 Fifth Ave., New York, NY 10016, U.S.A.
sartemov@gc.cuny.edu
[2] LIFIA, Fac. de Informática, UNLP, Argentina and CONICET
eduardo@lifia.info.unlp.edu.ar

Abstract. We introduce a natural deduction formulation for the Logic of Proofs, a refinement of modal logic **S4** in which the assertion $\Box A$ is replaced by $[\![s]\!]A$ whose intended reading is "*s is a proof of A*". A term calculus for this formulation yields a typed lambda calculus $\lambda^{\mathbf{I}}$ that internalises *intensional* information on *how* a term is computed. In the same way that the Logic of Proofs internalises its own *derivations*, $\lambda^{\mathbf{I}}$ internalises its own *computations*. Confluence and strong normalisation of $\lambda^{\mathbf{I}}$ is proved. This system serves as the basis for the study of type theories that internalise intensional aspects of computation.

1 Introduction

This paper introduces a typed lambda calculus that internalises its own computations. Such a system is obtained by a propositions-as-types [GLT89] interpretation of a logical system for provability which internalises its own proofs, namely the Logic of Proofs **LP** [Art95, Art01]. Proofs are represented as combinatory terms (*proof polynomials*). In the minimal propositional logic fragment of **LP** proof polynomials are constructed from proof variables and constants using two operations: application "$\cdot$" and proof-checker "!". The usual propositional connectives are augmented by a new one: given a proof polynomial s and a proposition A build $[\![s]\!]A$. The intended reading is: "*s is a proof of A*". The axioms and inference schemes of **LP** are:

A0. Axiom schemes of minimal logic in the language of **LP**	
A1. $[\![s]\!]A \supset A$	*"verification"*
A2. $[\![s]\!](A \supset B) \supset ([\![t]\!]A \supset [\![s \cdot t]\!]B)$	*"application"*
A3. $[\![s]\!]A \supset [\![!s]\!][\![s]\!]A$	*"proof checker"*
R1. $\Gamma \vdash A \supset B$ and $\Gamma \vdash A$ implies $\Gamma \vdash B$	*"modus ponens"*
R2. If **A** is an axiom **A0**-**A3**, and c is a proof constant, then $\vdash [\![c]\!]\mathbf{A}$	*"necessitation"*

For verification one reads: "*if s is a proof of A, then A holds*". As regards the proof polynomials the standard interpretation is as follows. For application one reads: "*if s is a proof of $A \supset B$ and t is a proof of A, then $s \cdot t$ is a proof of B*". Thus "$\cdot$" represents composition of proofs. For proof checking one reads: "*if s is* *a proof of A, then $!s$ is a proof of the sentence* 's is a proof of A' ". Thus $!s$ is seen as a computation that verifies $[\![s]\!]A$.

First we introduce a natural deduction (ND) formulation $\mathbf{LP}^{-}_{nd}$ for $\mathbf{LP}$. Following recent work on *judgemental reconstruction* [ML83] of intuitionistic **S4** [DP96, DP01b, DP01a], judgements are introduced in which a distinction is made between propositions whose *truth* is assumed from those whose *validity* is assumed. Judgements in $\mathbf{LP}^{-}_{nd}$ are of the form:

$$v_1 : A_1 \ valid, \ldots, v_n : A_n \ valid; a_1 : B_1 \ true, \ldots, a_m : B_m \ true \vdash A \ true \mid s$$

which expresses "*s is evidence that A is true, assuming that for each $i \in 1..n$, v_i is evidence that A_i is valid and assuming that for each $j \in 1..m$, a_j is evidence that B_j is true*". Such judgements are called hypothetical judgements [ML83]. Evidence s is a constituent part of the judgement without which the proposed reading is no longer possible. Its importance is reflected in the following introduction rule for the $[\![s]\!]$ connective:

$$\frac{\Delta; \cdot \vdash A \mid s}{\Delta; \Gamma \vdash [\![s]\!]A \mid !s} \ \Box\mathsf{I}$$

This scheme internalises proofs of validity: If s is evidence that A is unconditionally true ("$\cdot$" indicates an empty set of hypothesis of truth), then it is true that s is a proof of A. The new witness to this fact is registered as the evidence $!s$. The "!" operator is reminiscent of that of proof polynomials. However, in $\mathbf{LP}^{-}_{nd}$, proof terms such as s encode ND derivations and thus are no longer the proof polynomials of $\mathbf{LP}$.

At the basis of the meaning of hypothetical judgements (provided by the axioms and inference schemes presented in Sec. 2) is the notion of substitution. The following two principles, the Substitution Principle for Truth with Evidence and the Substitution Principle for Validity with Evidence, reflect the true hypothetical nature of hypothesis.

- If $\Delta; \Gamma \vdash A \mid s$ and $\Delta; \Gamma, a : A, \Gamma' \vdash B \mid t$, then $\Delta; \Gamma, \Gamma' \vdash B \mid t^{a}_{s}$
- If $\Delta; \cdot \vdash A \mid s$ and $\Delta, v : A, \Delta'; \Gamma \vdash B \mid t$, then $\Delta, \Delta'; \Gamma \vdash B^{v}_{s} \mid t^{v}_{s}$

These principles allow derivations to be composed, a fundamental operation on which the process of normalisation of derivations relies on. In fact, composition of derivations suffices, in general, to formulate rules for eliminating redundancy in derivations. However, the fact that $\mathbf{LP}^{-}_{nd}$ internalises its own proofs presents a complication in this respect. For example, the naïve simplification step depicted in Fig. 1 which relies on the Substitution Principle for Truth with Evidence fails given that it modifies the judgement that was originally justified. On a more pragmatical level, such a normalisation process may produce invalid derivations [AB06]. The problem stems in that the normalisation step is attempting to identify, at the meta-level, the two derivations and $\mathbf{LP}^{-}_{nd}$ happens to internalise its own derivations. As a consequence, the normalisation step must be reflected in the logic too. More precisely, a new judgement expressing the equality on evidence must be introduced. Accordingly, in Sec. 2.2 we extend

$$\dfrac{\dfrac{\Delta; \Gamma, a : A \vdash B \mid s}{\Delta; \Gamma \vdash A \supset B \mid \lambda a : A.s} \supset \mathsf{I} \qquad \Delta; \Gamma \vdash A \mid t}{\Delta; \Gamma \vdash B \mid (\lambda a : A.s) \cdot t} \supset \mathsf{E} \qquad \rightsquigarrow \qquad \Delta; \Gamma \vdash B \mid s^a_t$$

Fig. 1. Naïve simplification

our ND presentation $\mathbf{LP}^-_{nd}$ with *hypothetical judgements for evidence equality.* The normalisation process is thus internalised into the logic. For this amended system, $\mathbf{LP}_{nd}$, the set of derivations is seen to be closed under normalisation.

In Sec. 4 we study a term assignment for $\mathbf{LP}_{nd}$, namely the *intensional lambda calculus* ($\lambda^{\mathbf{I}}$). $\lambda^{\mathbf{I}}$ results from extending the propositions-as-types correspondence to $\mathbf{LP}_{nd}$. The normalisation process of derivations in $\mathbf{LP}_{nd}$ yields a notion of *reduction* on the typed lambda calculus terms. Just as $\mathbf{LP}_{nd}$ internalises its own derivations, the operational counterpart of this logic is seen to internalise the reduction of derivations. We show that $\lambda^{\mathbf{I}}$ is strongly normalising and confluent by applying properties of higher-order rewrite systems.

Related work. S. Artemov introduced the Logic of Proofs in [Art95, Art01]. A ND presentation for **LP** is provided in [Art01]. This presentation relies on combinatory terms as proof terms (proof polynomials). It is a ND system for a logic that internalises Hilbert style proofs. As a consequence, the presence of normalisation is not felt at the level of proof terms. Since we use proof terms that encode ND proofs, the internalisation scheme implemented by $\Box\mathsf{I}$ together with the normalisation process on derivations has a visible impact in the design of the inference schemes for our system $\mathbf{LP}_{nd}$.

V. Brezhnev [Bre01] formulates a system of labeled sequents. Roughly, a refinement of the sequent presentation of **LP** [Art01] is presented in which labeled sequents are derived rather than the sequents themselves. It has been proved [Art95, Art01] that **LP** is a refinement of **S4** in the sense that any cut-free derivation of **S4** can be *realized* by one of **LP**. A realization of an **S4** derivation is the process of appropriately filling in all occurrences of boxes $\Box$ with proof polynomials such that a valid **LP** derivation is obtained. The aim of the work of Brezhnev is to make this correspondence explicit. Also, he extends the correspondence to other modal logics such as **K**, **K4**, **D**, **D4** and **T**.

From a type theoretic perspective we should mention the theory of dependent types [Bar92]. Dependent type theory is the type-theoretic counterpart of first-order logic via the propositions-as-types correspondence. Types may depend on terms, in much the same way that a type $[\![s]\!]A$ depends on the proof term s. In contrast to $\lambda^{\mathbf{I}}$, dependent type theory lacks a notion of internalisation of derivations.

More closely related to $\lambda^{\mathbf{I}}$ is the reflective λ-calculus (λ^∞) [AA01]. λ^∞ is a rigidly typed (all variables and subterms carry a fixed type) lambda calculus which essentially results from a term assignment of the aforementioned ND presentation of [Art01]. The difference with the approach of this paper is that in the reflective λ-calculus $[\![s]\!]A$ is read as "*s has type A*". Accordingly, hypothesis are not labeled with variables, rather they are part of the formula. For example, $x : A \vdash x : A$ becomes $[\![x]\!]A \vdash [\![x]\!]A$. An unwanted complication is that the desired internalisation property (namely, $A_1, A_2, \ldots, A_n \vdash B$ implies that for fresh variables $x_1, x_2, \ldots, x_n$ there exists a term $t(x_1, x_2, \ldots, x_n)$ such that we can prove $[\![x_1]\!]A_1, [\![x_2]\!]A_2, \ldots, [\![x_n]\!]A_n \vdash [\![t(x_1, x_2, \ldots, x_n)]\!]B$) changes the types of the assumptions. As a consequence, operations on types having nested copies of proof terms are required for typing. This also complicates the definition of reduction on terms.

Note: For further details and full proofs see [AB06].

2 Natural Deduction for LP

Following [DP01b] we distinguish the following judgements: "*A is a proposition*" ("*A proposition*" for short), "*A true*" and "*A valid*". In the case of the second and third judgements we assume that it is already known that "*A proposition*". The inference schemes defining the meaning of "*A proposition*" are the usual well-formedness conditions and hence are omitted. Our interest lies in providing meaning to the following *hypothetical judgements with explicit evidence*:

$$v_1 : A_1 \ \mathit{valid}, \ldots, v_n : A_n \ \mathit{valid}; a_1 : B_1 \ \mathit{true}, \ldots, a_m : B_m \ \mathit{true} \vdash A \ \mathit{true} \mid s$$

by a set of axiom schemes and inference schemes, where v_i, $i \in 1..n$, and a_j, $j \in 1..m$, range over some given some set of *evidence (of proof) variables* $\{x_1, x_2, \ldots\}$. To the left of the semi-colon we place the assumptions of validity and to the right the assumptions of truth. For the sake of readability, we drop the qualifiers "*valid*" and "*true*". Consequently, these judgements take the form:

$$v_1 : A_1, \ldots, v_n : A_n; a_1 : B_1, \ldots, a_m : B_m \vdash A \mid s$$

In addition to the usual requirement that the v_i and a_i be distinct, we must also require that they be fresh (i.e. that they do not occur in the A_i and B_i). Note also that since we assume J_1 through J_n, in a hypothetical proof of a hypothetical judgement with explicit evidence, we may use the J_i as if we knew them. As a consequence we can substitute an arbitrary derivation of J_i for all its uses by means of the two aforementioned substitution principles.

2.1 Axiom and Inference Schemes

It is convenient to introduce first a preliminary ND system ($\mathbf{LP}^{-}_{nd}$), point out its weaknesses and then introduce the final ND system $\mathbf{LP}_{nd}$. We begin by defining the set of Proof Terms, Propositions, Truth Contexts and Validity Contexts.

$$\begin{array}{ll}
\textit{Proof Terms} & s ::= x \mid s \cdot s \mid \lambda a : A.s \mid !s \mid \textsc{Xtrt}\, s \,\textsc{as}\, v : A \,\textsc{in}\, s \\
\textit{Propositions} & A ::= P \mid A \supset A \mid [\![s]\!]A \\
\textit{Truth Contexts} & \Gamma ::= \cdot \mid \Gamma, a : A \\
\textit{Validity Contexts} & \Delta ::= \cdot \mid \Delta, v : A
\end{array}$$

Minimal Propositional Logic Fragment

$$\frac{}{\Delta; \Gamma, a : A, \Gamma' \vdash A \mid a}\,\mathsf{oVar}$$

$$\frac{\Delta; \Gamma, a : A \vdash B \mid s}{\Delta; \Gamma \vdash A \supset B \mid \lambda a : A.s}\supset\mathsf{I} \qquad \frac{\Delta; \Gamma \vdash A \supset B \mid s \quad \Delta; \Gamma \vdash A \mid t}{\Delta; \Gamma \vdash B \mid s \cdot t}\supset\mathsf{E}$$

Provability Fragment

$$\frac{}{\Delta, v : A, \Delta'; \Gamma \vdash A \mid v}\,\mathsf{mVar}$$

$$\frac{\Delta; \cdot \vdash A \mid s}{\Delta; \Gamma \vdash [\![s]\!]A \mid !s}\Box\mathsf{I} \qquad \frac{\Delta; \Gamma \vdash [\![r]\!]A \mid s \quad \Delta, v : A; \Gamma \vdash C \mid t}{\Delta; \Gamma \vdash C^v_r \mid \textsc{Xtrt}\, s\, \textsc{as}\, v : A\, \textsc{in}\, t}\Box\mathsf{E}$$

Fig. 2. Explanation for Hypothetical Judgements with Explicit Evidence

We write $\mathsf{fv}(s)$ for the set of free variables of a proof term. All free occurrences of a (resp. v) in s are bound in $\lambda a : A.s$ (resp. $\textsc{Xtrt}\, t\, \textsc{as}\, v : A\, \textsc{in}\, s$). A proposition is either a propositional variable P, an implication $A \supset B$ or a validity proposition $[\![s]\!]A$. Truth and validity contexts are sequences of labeled propositions; "$\cdot$" denotes the empty context. We write s^x_t for the result of substituting all free occurrences of x in s by t and assume that bound variables are renamed whenever necessary; likewise for A^x_t.

Definition 1. $\mathbf{LP}^-_{nd}$ *is defined by the schemes of Fig. 2.*

An informal explanation of some of these schemes follows. The axiom scheme oVar states that the judgement "$\Delta; \Gamma, a : A, \Gamma' \vdash A \mid a$" is evident in itself. Indeed, if we assume that a is evidence that proposition A is true, then we may immediately conclude that A is true with evidence a. The introduction scheme for the $[\![s]\!]$ modality internalises metalevel evidence into the object logic. It states that if s is unconditional evidence that A is true, then A is in fact valid with witness s (i.e. $[\![s]\!]A$ is true). Evidence for the truth of $[\![s]\!]A$ is constructed from the (verified) evidence that A is unconditionally true by prefixing it with a bang constructor. Finally, $\Box\mathsf{E}$ allows the discharging of validity hypothesis. In order to discharge the validity hypothesis $v : A$, a proof of the validity of A is required. In our system, this requires proving that $[\![r]\!]A$ is true with evidence s, for some evidence of proof r and s. Note that r is evidence that A is unconditionally true (i.e. valid) whereas s is evidence that $[\![r]\!]A$ is true. The former is then substituted in the place of all free occurrences of v in the proposition C. This construction is recorded with evidence $\textsc{Xtrt}\, s\, \textsc{as}\, v : A\, \textsc{in}\, t$ in the conclusion. The mnemonic symbols "$\textsc{Xtrt}$" stand for "extract" since, intuitively, evidence of the validity of A may be seen to be extracted from evidence of the truth of $[\![r]\!]A$. A sample derivation in $\mathbf{LP}^-_{nd}$ of $[\![s]\!]A \supset [\![!s]\!][\![s]\!]A$ follows.

$$\dfrac{\dfrac{\dfrac{}{\cdot; a : [\![s]\!]A \vdash [\![s]\!]A \mid a}\,\mathsf{oVar} \qquad \dfrac{\dfrac{\dfrac{}{w : A; \cdot \vdash A \mid w}\,\mathsf{mVar}}{w : A; \cdot \vdash [\![w]\!]A \mid !w}\,\Box\mathsf{I}}{w : A; a : [\![s]\!]A \vdash [\![!w]\!][\![w]\!]A \mid !!w}\,\Box\mathsf{I}}{\cdot; a : [\![s]\!]A \vdash [\![!s]\!][\![s]\!]A \mid \textsc{Xtrt}\, a\, \textsc{as}\, w : A\, \textsc{in}\, !!w}\,\Box\mathsf{E}}{\cdot; \cdot \vdash [\![s]\!]A \supset [\![!s]\!][\![s]\!]A \mid \lambda a : [\![s]\!]A.\textsc{Xtrt}\, a\, \textsc{as}\, w : A\, \textsc{in}\, !!w}\,\supset\mathsf{I}$$

The standard structural properties of judgements (weakening, contraction and exchange) hold. Also, the substitution principles for truth with evidence and validity with evidence may be proved by induction on the derivation. A more interesting property is that $\mathbf{LP}^{-}_{nd}$ internalises its own proofs of unconditional truth.

Lemma 1 (Lifting [Art95]). *Let $\Delta = u_1 : A_1, \ldots, u_n : A_n$ and $\Gamma = b_1 : B_1, \ldots, b_m : B_m$. If $\Delta; \Gamma \vdash A \mid r$, then $\Delta, v_1 : B_1, \ldots, v_m : B_m; \cdot \vdash [\![s(\boldsymbol{u}, \boldsymbol{v})]\!]A \mid t(\boldsymbol{u}, \boldsymbol{v})$ where $s(\boldsymbol{u}, \boldsymbol{v}) = (\lambda \boldsymbol{b} : \boldsymbol{B}.r) \cdot v_1 \cdot v_2 \cdot \ldots \cdot v_m$ and $t(\boldsymbol{u}, \boldsymbol{v}) = \textsc{Xtrt}\, !\lambda \boldsymbol{b} : \boldsymbol{B}.r\, \textsc{as}\, u : (\boldsymbol{B} \supset A)\, \textsc{in}\, !(u \cdot v_1 \cdot v_2 \ldots \cdot v_m)$.*

2.2 Normalisation and Evidence Equality

As mentioned above a naïve approach to normalisation is doomed to fail unless our attempt to simplify (hence equate) derivations is *reflected* in the object logic. Indeed, a new judgement must be considered, namely *hypothetical judgements for evidence equality*:

$$\Delta; \Gamma \vdash s \equiv t : A$$

Read: "*s and t are provably equal evidence of the truth of A under the validity assumptions of Δ and the truth assumptions of Γ*". This judgement internalises at the object level the equality of derivations induced by the normalisation steps. Note that evidence for provable equality is not considered in hypothetical judgements for evidence equality. Although this could be an interesting route for exploration, in our setting we would then be forced to define a notion of equality on this new kind of evidence, thus leading to an infinite regression.

In addition to defining the meaning of this new judgement by means of new axiom and inference schemes, we must indicate how it affects the meaning of hypothetical judgements with explicit evidence.

$$\dfrac{\Delta; \Gamma \vdash A \mid s \quad \Delta; \Gamma \vdash s \equiv t : A}{\Delta; \Gamma \vdash A \mid t}\,\mathsf{EqEvid}$$

The upper left judgement of EqEvid is called the minor premise and the one on the right the major premise. Fig. 3 defines the meaning of hypothetical judgement for evidence equality[1].

[1] We omit the standard inference schemes for symmetry, transitivity and congruence of evidence equality [AB06].

$$\frac{\Delta;\Gamma \vdash A \mid s}{\Delta;\Gamma \vdash s \equiv s : A}\ \mathsf{EqRefl} \qquad \frac{\Delta;\Gamma, a : A \vdash B \mid s \quad \Delta;\Gamma \vdash A \mid t}{\Delta;\Gamma \vdash s^a_t \equiv (\lambda a : A.s) \cdot t : B}\ \mathsf{EqBeta}$$

$$\frac{\Delta;\cdot \vdash A \mid s \quad \Delta, v : A;\Gamma \vdash C \mid t}{\Delta;\Gamma \vdash t^v_s \equiv \textsc{Xtrt}\,!s\,\textsc{as}\,v : A\,\textsc{in}\,t : C^v_s}\ \mathsf{Eq\Box Beta}$$

$$\frac{\Delta;\Gamma \vdash A \supset B \mid s \quad a \notin \mathsf{fv}(s)}{\Delta;\Gamma \vdash \lambda a : A.(s \cdot a) \equiv s : A \supset B}\ \mathsf{EqEta} \qquad \frac{\Delta;\Gamma \vdash [\![s]\!]A \mid t \quad u \notin \mathsf{fv}(t)}{\Delta;\Gamma \vdash \textsc{Xtrt}\,t\,\textsc{as}\,u : A\,\textsc{in}\,!u \equiv t : [\![s]\!]A}\ \mathsf{Eq\Box Eta}$$

Fig. 3. Axioms for evidence equality

Definition 2. $\mathbf{LP}_{nd}$ *is obtained by augmenting the schemes of Fig. 2 with* EqEvid *and the schemes of Fig. 3.*

In the sequel we study hypothetical judgements derivable in $\mathbf{LP}_{nd}$. Note that the structural properties of $\mathbf{LP}^{-}_{nd}$ extend to $\mathbf{LP}_{nd}$.

We now return to normalisation of derivations. Two groups[2] of contractions of derivations are defined: principal contractions and silent permutative contractions. The first is internalised by the inference schemes defining provable equality of evidence. Permutative conversions need not be internalised since, in contrast to principal contractions, they do not alter the end judgement. They are thus dubbed *silent* permutative conversions. By defining an appropriate notion of cut segment one can show that contraction is weakly normalising: there is a sequence of contractions to normal form [AB06].

Lemma 2. *Contraction in* $\mathbf{LP}_{nd}$ *is weakly normalising.*

More importantly, we shall see shortly that contraction is in fact strongly normalising. The proof of this is established via weak normalisation.

3 Provability Semantics

Rules of $\mathbf{LP}_{nd}$ can be interpreted as admissible rules of $\mathbf{LP}$, hence supplied with a natural provability semantics. Interpretation of all rules other then $\supset$ I and $\Box$E are straightforward. The rule $\supset$ I corresponds to the Abstraction Rule which is admissible in $\mathbf{LP}$ [Art96]. There are two $\mathbf{LP}$-compliant interpretations of the rule $\Box$E, cf. Fig. 4. The left one, which we suggest calling *internalized reading* is self-explanatory. The right one, which we call *leveled* requires that a proof constant d is specified as $d : (r : A \supset A)$. We leave a more detailed investigation of the provability semantics of $\mathbf{LP}_{nd}$ to further studies.

[2] We ignore principal expansions in this extended abstract (see [AB06]).

$$\frac{\Gamma \vdash s : r : A \quad \Gamma, v : A \vdash t : C}{\Gamma \vdash t^v_r : C^v_r} \qquad \frac{\Gamma \vdash s : r : A \quad \Gamma, v : A \vdash t : C}{\Gamma \vdash t^v_{d \cdot s} : C^v_{d \cdot s}}$$

Fig. 4. Interpretations of $\Box\mathsf{E}$

4 The Intensional Lambda Calculus

This section introduces the *intensional lambda calculus* ($\lambda^{\mathbf{I}}$) and studies confluence and strong normalisation. We begin by defining the set of *raw* terms of $\lambda^{\mathbf{I}}$:

$$\begin{array}{lrl}
\textit{Proper Terms} & M ::= & x \mid M \cdot M \mid \lambda a : A.M \\
 & \mid & !M \mid \textsc{Xtrt}\, M \,\textsc{as}\, v : A \,\textsc{in}\, M \mid e \blacktriangleright M \\
\textit{Reduction Evidence} & e ::= & \beta([a : A]M, N) \mid \beta_{\Box}([v : A]M, N) \\
 & \mid & \textsc{Refl}(M) \mid \textsc{Sym}(e) \mid e; e \\
 & \mid & \textsc{Abs}([a : A]e) \mid \textsc{App}(e, e) \\
 & \mid & \textsc{BoxL}(e) \mid \textsc{BoxR}(e) \mid \textsc{xtrt}(e, [v : A]e)
\end{array}$$

A raw term of the form $M \cdot N$ is an *application*, $\lambda a : A.M$ is an *abstraction*, $!M$ is a *bang* term, $\textsc{Xtrt}\, M \,\textsc{as}\, v : A \,\textsc{in}\, N$ is an *extraction* and $e \blacktriangleright M$ is a *registered* term. Reduction evidence $\beta([a : A]M, N)$ is used to register that a principal $\supset$ contraction was applied together with the actual parameters ($\lambda a : A.M$ and N) and $\beta_{\Box}([v : A]M, N)$ is for principal $\Box$ contractions. The remaining reduction evidence terms are for the congruence inference schemes of evidence equality.

Let P range over an enumerable set of type variables. The set of *raw types* is the set of propositions of $\mathbf{LP}_{nd}$. In $\lambda^{\mathbf{I}}$ proper terms are assigned *pointed types* $\langle A, s\rangle$ and reduction evidence is assigned *equality types* $s \equiv t : A$. Since the typing schemes follow the axiom and inference schemes of $\mathbf{LP}_{nd}$, there are two *typing judgements*:

1. $\Delta; \Gamma \vdash M \rhd \langle A, s\rangle$, read: "*Proper term M has pointed type $\langle A, s\rangle$ under type assumptions Δ and Γ*" and
2. $\Delta; \Gamma \vdash e \rhd s \equiv t : A$, read: "*Reduction evidence e has equality type $s \equiv t : A$ under type assumptions Δ and Γ*".

Definition 3. *A proper term M is typable if there exist type assumptions Δ and Γ and a pointed type $\langle A, s\rangle$ such that $\Delta; \Gamma \vdash M \rhd \langle A, s\rangle$ is derivable using the typing schemes presented in Fig. 5. Typability of reduction evidence ($\Delta; \Gamma \vdash e \rhd s \equiv t : A$) is defined similarly [AB06]. A $\lambda^{\mathbf{I}}$-term is a raw term that is typable.*

The *contractions* defining normalisation on derivations of $\mathbf{LP}_{nd}$ induce a corresponding *reduction* relation on the $\lambda^{\mathbf{I}}$-terms that encode the derivations.

Definition 4 ($\lambda^{\mathbf{I}}$-reduction). *The $\lambda^{\mathbf{I}}$-reduction relation ($\to$) is obtained by taking the contextual closure of the reduction axioms:*

Minimal Propositional Logic Fragment

$$\frac{}{\Delta;\Gamma,a:A,\Gamma'\vdash a\rhd\langle A,a\rangle}\,\textsf{oVar}$$

$$\frac{\Delta;\Gamma,a:A\vdash M\rhd\langle B,s\rangle}{\Delta;\Gamma\vdash\lambda a:A.M\rhd\langle A\supset B,\lambda a:A.s\rangle}\supset\textsf{I}\qquad\frac{\Delta;\Gamma\vdash M\rhd\langle A\supset B,s\rangle\quad\Delta;\Gamma\vdash N\rhd\langle A,t\rangle}{\Delta;\Gamma\vdash M\cdot N\rhd\langle B,s\cdot t\rangle}\supset\textsf{E}$$

Provability Fragment

$$\frac{}{\Delta,v:A,\Delta';\Gamma\vdash v\rhd\langle A,v\rangle}\,\textsf{mVar}\qquad\frac{\Delta;\cdot\vdash M\rhd\langle A,s\rangle}{\Delta;\Gamma\vdash !M\rhd\langle[\![s]\!]A,!s\rangle}\,\Box\textsf{I}$$

$$\frac{\Delta;\Gamma\vdash M\rhd\langle[\![s]\!]A,s'\rangle\quad\Delta,v:A;\Gamma\vdash N\rhd\langle C,t\rangle}{\Delta;\Gamma\vdash \textsc{Xtrt}\,M\,\textsc{as}\,v:A\,\textsc{in}\,N\rhd\langle C^v_s,\textsc{Xtrt}\,s'\,\textsc{as}\,v:A\,\textsc{in}\,t\rangle}\,\Box\textsf{E}$$

$$\frac{\Delta;\Gamma\vdash M\rhd\langle A,s\rangle\quad\Delta;\Gamma\vdash e\rhd s\equiv t:A}{\Delta;\Gamma\vdash e\blacktriangleright M\rhd\langle A,t\rangle}\,\textsf{EqEvid}$$

Fig. 5. Typing schemes for proper terms

$$\begin{array}{lll}
(\lambda a:A.M)\cdot N & \to_\beta & \beta([a:A]M,N)\blacktriangleright M^a_N\\
\textsc{Xtrt}\,!N\,\textsc{as}\,v:A\,\textsc{in}\,M & \to_{\beta_\Box} & \beta_\Box([v:A]M,N)\blacktriangleright M^v_N\\
\\
(e\blacktriangleright M)\cdot N & \to_{\blacktriangleright L} & \textsc{App}(e,\textsc{Refl}(N))\blacktriangleright M\cdot N\\
\textsc{Xtrt}\,e\blacktriangleright N\,\textsc{as}\,v:A\,\textsc{in}\,M & \to_{\blacktriangleright xtr} & \textsc{xtrt}(e,[v:A]\textsc{Refl}(M))\blacktriangleright \textsc{Xtrt}\,N\,\textsc{as}\,v:A\,\textsc{in}\,M
\end{array}$$

Note that, just as proof terms are internalised as part of the process of proving a formula in **LP**, so the process of reducing a $\lambda^{\mathbf{I}}$-term internalises evidence of reduction. Indeed, an application of the β reduction rule results in a $\lambda^{\mathbf{I}}$-term that incorporates a witness to the fact that such a reduction step was applied. This reduction evidence provides *intensional* information on *how* the result was computed.

Consider the term from the ordinary typed lambda calculus $I\cdot(I\cdot b)$ (which is also a term in $\lambda^{\mathbf{I}}$) where I abbreviates $\lambda a:A.a$. In the typed lambda calculus it reduces in two different ways to $I\cdot b$ (we underline the contracted redex):

1. $I\cdot(\underline{I\cdot b})\to I\cdot b$ 2. $\underline{I\cdot(I\cdot b)}\to I\cdot b$

The fact that both these reductions reach the same term is known as a "syntactic coincidence" [HL91] in the rewriting/lambda calculus community. Although the same term is reached they are computed in rather different ways in the sense that unrelated redexes are contracted. Note, however, that in $\lambda^{\mathbf{I}}$ these two derivations now end in different terms:

1. $I\cdot(I\cdot b)\to I\cdot(\beta([a:A]a,b)\blacktriangleright b)$
2. $I\cdot(I\cdot b)\to\beta([a:A]a,(I\cdot b))\blacktriangleright I\cdot b$

Since reduction is obtained as a straightforward mapping of contraction of derivations, the following type-soundness result holds.

Lemma 3 (Subject Reduction). *If $M \to_{\lambda^{\mathbf{I}}} N$ and $\Delta; \Gamma \vdash M \rhd \langle A, s \rangle$, then $\Delta; \Gamma \vdash N \rhd \langle A, s \rangle$.*

4.1 Confluence and Strong Normalisation for $\boldsymbol{\lambda}^{\mathbf{I}}$

Higher-order term rewrite systems (HORS) [Klo80, Nip91, TER03] extend first-order term rewrite systems by allowing terms with binders. The λ-calculus is the prototypical example of a HORS. $\lambda^{\mathbf{I}}$ can also be presented as a HORS - we'll present it as an HRS [Nip91]. In HRS the simply typed lambda calculus is used as a meta-language for writing the left and right-hand side of rewrite rules. Boldface is used for constants, $x, y, \ldots$ for variables, $x.M$ for abstraction and $M(N)$ for application. The rewrite rules for $\lambda^{\mathbf{I}}$ are (the signature of the symbols involved is straightforward and hence omitted):

$$\begin{array}{lll}
\mathbf{app}(\mathbf{abs}(x.z(x)), y) & \to_{\beta} & \mathbf{evid}(\mathbf{betaE}(x.z(x), y), z(y)) \\
\mathbf{xtrt}(\mathbf{bang}(y), x.z(x)) & \to_{\beta_{\Box}} & \mathbf{evid}(\mathbf{betaBoxE}(x.z(x), y), z(y)) \\
\mathbf{app}(\mathbf{evid}(x, y), z) & \to_{\blacktriangleright L} & \mathbf{evid}(\mathbf{appE}(x, \mathbf{reflE}(z)), \mathbf{app}(y, z)) \\
\mathbf{xtrt}(\mathbf{evid}(w, y), x.z(x)) & \to_{\blacktriangleright xtr} & \mathbf{evid}(\mathbf{xtrtE}(w, x.\mathbf{reflE}(z(x))), \mathbf{xtrt}(y, x.z(x)))
\end{array}$$

The interest in HOR is that general results on combinatorial properties of rewriting can be established. Two such results are of use to us. The first states that orthogonal, pattern HRS are confluent. *Orthogonal* means that rewrite steps are independent: If two redexes in a term may be reduced, the reduction of one of them does not "interfere" with the other one except possibly by duplicating or erasing it. *Pattern* means that in the left-hand sides of rewrite rules free variables can only be applied to distinct bound variables (modulo η-equivalence). This guarantees that higher-order pattern matching behaves similar to the first-order case: unification of higher-order patterns is decidable and most general unifiers can be computed. We write PRS for pattern HRS.

Proposition 1 ([Nip91]). *Orthogonal PRS are confluent.*

The $\lambda^{\mathbf{I}}$-calculus is easily seen to be an orthogonal PRS: it is left-linear and non-overlapping. We may thus immediately conclude, from Prop. 1, that it is confluent.

Proposition 2. *$\lambda^{\mathbf{I}}$ is confluent.*

The other interesting property is that of *uniform normalisation.* First we introduce some terminology. A rewrite step $M \to N$ is *perpetual* if whenever M has an infinite reduction, N has one too. A rewrite system is *uniformly normalising* if all its steps are perpetual. An example is the λI-calculus [CR36] which is the standard λ-calculus in which the set of terms is restricted to those M such that $\lambda x.N \subseteq M$ implies $x \in \mathsf{fv}(N)$. The proof of this fact for λI relies on two properties: (1) all reduction steps are *non-erasing* and (2) it is orthogonal. It turns out that this result can be extended to arbitrary higher-order rewrite systems.

Proposition 3 ([KOvO01]). *Non-erasing, orthogonal and fully-extended*[3] *second-order*[4] *PRS are uniformly normalising.*

A close look at the HRS presentation of $\lambda^{\mathbf{I}}$ reveals that it is in fact a non-erasing, fully-extended, second-order PRS. Furthermore, we have already mentioned that it is orthogonal. As a consequence, we conclude the following from Prop. 3.

Proposition 4. $\lambda^{\mathbf{I}}$ *is uniformly normalising.*

The interesting thing about uniformly normalisable rewrite systems is that weak normalisation is equivalent to strong normalisation. Therefore, since we have proved that $\lambda^{\mathbf{I}}$ is weakly normalising, we conclude that:

Proposition 5. $\lambda^{\mathbf{I}}$ *is strongly normalising.*

5 Conclusions

A study of the computational interpretation of the Logic of Proofs via the propositions-as-types correspondence requires an appropriate ND presentation. This paper presents one such system, $\mathbf{LP}_{nd}$, resulting from a judgemental analysis [ML83, DP01a] of **LP**. The term assignment yields a typed lambda calculus, called the intensional lambda calculus ($\lambda^{\mathbf{I}}$), that is capable of internalising computation evidence, in much the same way that **LP** is capable of internalising derivability evidence. Computations in $\lambda^{\mathbf{I}}$ yield terms that include information on how this computation is performed.

As mentioned, the fact that $I \cdot (\underline{I \cdot b}) \to I \cdot b$ and $\underline{I \cdot (I \cdot b)} \to I \cdot b$ reduce to the same term in the standard lambda calculus is known as a "syntactic coincidence" [HL91] since these terms are computed in different ways. In $\lambda^{\mathbf{I}}$ the corresponding reductions are no longer cofinal given that intensional information on how the term was computed is part of the result. Further investigation on the relation with equivalence of reductions as defined by Lévy [Lév78, TER03] is left to future work.

Other interesting directions are the formulation of intensional calculi for linear and classical logic given their tight connections with resource conscious computing and control operators and the analysis of the explicit modality and how it relates to staged computation and run-time code generation [DP96, WLPD98].

[3] A rewrite system is said to be fully-extended if each of its rewrite rules (l, r) verifies the following: for every occurrence $x(P_1, \ldots, P_n)$ in l of a free variable x, $P_1, \ldots, P_n$ is the list of *all* bound variables above it.

[4] Define the *order* of a type A of the simply typed lambda calculus, written $ord(A)$, to be 1 if the type is a base type and $max(ord(A_1) + 1, A_2)$ if $A = A_1 \to A_2$. The order of rewrite system is the maximum order of the types of the variables that occur in its rewrite rules.

References

[AA01] Jesse Alt and Sergei Artemov. Reflective λ-calculus. In *Proceedings of the Dagstuhl-Seminar on Proof Theory in Computer Science*, volume 2183 of *LNCS*, 2001.

[AB06] Sergei Artemov and Eduardo Bonelli. The intensional lambda calculus. Technical report, December 2006. http://www.lifia.info.unlp.edu.ar/~eduardo/lamIFull.pdf.

[Art95] Sergei Artemov. Operational modal logic. Technical Report MSI 95-29, Cornell University, 1995.

[Art96] Sergei Artemov. Proof realization of intuitionistic and modal logics. Technical Report MSI 96-06, Cornell University, 1996.

[Art01] Sergei Artemov. Unified semantics of modality and λ-terms via proof polynomials. *Algebras, Diagrams and Decisions in Language, Logic and Computation*, pages 89–118, 2001.

[Bar92] Henk P. Barendregt. Lambda Calculi with Types. In S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2. Oxford University Press, 1992.

[Bre01] Vladimir Brezhnev. On the logic of proofs. In Kristina Striegnitz, editor, *Proceedings of the Sixth ESSLLI Student Session*, pages 35–45, 2001.

[CR36] Alonzo Church and John B. Rosser. Some properties of conversion. *Transactions of the American Mathematical Society*, 39:472–482, 1936.

[DP96] Rowan Davies and Frank Pfenning. A modal analysis of staged computation. In Jr. Guy Steele, editor, *Proceedings of the 23rd Annual Symposium on Principles of Programming Languages*, pages 258–270, St. Petersburg Beach, Florida, January 1996. ACM Press.

[DP01a] Rowan Davies and Frank Pfenning. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11:511–540, 2001.

[DP01b] Rowan Davies and Frank Pfenning. A modal analysis of staged computation. *Journal of the ACM*, 48(3):555–604, May 2001.

[GLT89] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1989.

[HL91] Gérard Huet and Jean-Jacques Lévy. Computations in orthogonal rewriting systems. In J.L. Lassez and G.D. Plotkin, editors, *Computational Logic; Essays in honor of Alan Robinson*, pages 394–443. MIT Press, 1991.

[Klo80] Jan W. Klop. *Combinatory Reduction Systems*. PhD thesis, CWI, Amsterdam, 1980. Mathematical Centre Tracts n.127.

[KOvO01] Zurab Khasidashvili, Mizuhito Ogawa, and Vincent van Oostrom. Perpetuality and uniform normalization in orthogonal rewrite systems. *Information and Computation*, 164:118–151, 2001.

[Lév78] Jean-Jacques Lévy. *Réductions correctes et optimales dans le lambda-calcul*. PhD thesis, Université Paris VII, 1978.

[ML83] Per Martin-Löf. On the meaning of the logical constants and the justifications of the logical laws. Lectures given at the meeting Teoria della Dimostrazione e Filosofia della Logica, in Siena, 6-9 April 1983, by the Scuola di Specializzazione in Logica Matematica of the Università degli Studi di Siena., 1983.

[Nip91] Tobias Nipkow. Higher-order critical pairs. In *Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, July 1991.
[TER03] TERESE. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, March 2003.
[WLPD98] Philip Wickline, Peter Lee, Frank Pfenning, and Rowan Davies. Modal types as staging specifications for run-time code generation. *ACM Computing Surveys*, 30(3es), September 1998.

A Contractions for $\mathbf{LP}_{nd}$

1. Principal Contractions

$$\dfrac{\dfrac{\Delta;\Gamma,a:A\vdash B\mid s}{\Delta;\Gamma\vdash A\supset B\mid \lambda a:A.s}\supset\mathsf{I}\qquad \Delta;\Gamma\vdash A\mid t}{\Delta;\Gamma\vdash B\mid(\lambda a:A.s)\cdot t}\supset\mathsf{E}\qquad\rightsquigarrow$$

$$\dfrac{\dfrac{\pi}{\Delta;\Gamma\vdash B\mid s^a_t}\qquad \dfrac{\Delta;\Gamma,a:A\vdash B\mid s\quad \Delta;\Gamma\vdash A\mid t}{\Delta;\Gamma\vdash s^a_t\equiv(\lambda a:A.s)\cdot t:B}\mathsf{EqBeta}}{\Delta;\Gamma\vdash B\mid(\lambda a:A.s)\cdot t}\mathsf{EqEvid}$$

$$\dfrac{\dfrac{\Delta;\cdot\vdash A\mid s}{\Delta;\Gamma\vdash [\![s]\!]A\mid !s}\Box\mathsf{I}\qquad \Delta,v:A;\Gamma\vdash C\mid t}{\Delta;\Gamma\vdash C^v_s\mid \textsc{Xtrt}\,!s\,\textsc{as}\,v:A\,\textsc{in}\,t}\Box\mathsf{E}\qquad\rightsquigarrow$$

$$\dfrac{\dfrac{\pi}{\Delta;\Gamma\vdash C^v_s\mid t^v_s}\qquad \dfrac{\Delta;\cdot\vdash A\mid s\quad \Delta,v:A;\Gamma\vdash C\mid t}{\Delta;\Gamma\vdash t^v_s\equiv\textsc{Xtrt}\,!s\,\textsc{as}\,v:A\,\textsc{in}\,t:C^v_s}\mathsf{Eq}\Box\mathsf{Beta}}{\Delta;\Gamma\vdash C^v_s\mid \textsc{Xtrt}\,!s\,\textsc{as}\,v:A\,\textsc{in}\,t}\mathsf{EqEvid}$$

where π results from the Substitution Principle for Validity with Evidence.
2. Silent Permutative Contractions

$$\dfrac{\dfrac{\Delta;\Gamma\vdash A_1\supset A_2\mid s\quad \Delta;\Gamma\vdash s\equiv t:A_1\supset A_2}{\Delta;\Gamma\vdash A_1\supset A_2\mid t}\mathsf{EqEvid}\qquad \Delta;\Gamma\vdash A_1\mid r}{\Delta;\Gamma\vdash A_2\mid t\cdot r}\supset\mathsf{E}\qquad\rightsquigarrow$$

$$\dfrac{\dfrac{\Delta;\Gamma\vdash A_1\supset A_2\mid s\quad \Delta;\Gamma\vdash A_1\mid r}{\Delta;\Gamma\vdash A_2\mid s\cdot r}\supset\mathsf{E}\qquad \dfrac{\Delta;\Gamma\vdash s\equiv t:A_1\supset A_2\quad \dfrac{\Delta;\Gamma\vdash A_1\mid r}{\Delta;\Gamma\vdash r\equiv r:A_1}\mathsf{EqRefl}}{\Delta;\Gamma\vdash s\cdot r\equiv t\cdot r:A_2}\mathsf{Eq}\supset\mathsf{E}}{\Delta;\Gamma\vdash A_2\mid t\cdot r}\mathsf{EqEvid}$$

$$\dfrac{\dfrac{\Delta;\Gamma\vdash [\![s_1]\!]A \mid s_2 \quad \Delta;\Gamma\vdash s_2\equiv r : [\![s_1]\!]A}{\Delta;\Gamma\vdash [\![s_1]\!]A \mid r}\,\mathsf{EqEvid} \qquad \Delta,v:A;\Gamma\vdash C\mid t}{\Delta;\Gamma\vdash C^v_{s_1} \mid \textsc{Xtrt}\, r \,\textsc{as}\, v : A \,\textsc{in}\, t}\,\Box\mathsf{E} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\Delta;\Gamma\vdash [\![s_1]\!]A \mid s_2 \quad \Delta,v:A;\Gamma\vdash C\mid t}{\Delta;\Gamma\vdash C^v_{s_1} \mid \textsc{Xtrt}\, s_2 \,\textsc{as}\, v : A \,\textsc{in}\, t}\,\Box\mathsf{E}}{\Delta;\Gamma\vdash C^v_{s_1} \mid q}\,\Box\mathsf{E} \qquad \dfrac{\Delta;\Gamma\vdash s_2\equiv r : [\![s_1]\!]A \quad \dfrac{\Delta,v:A;\Gamma\vdash C\mid t}{\Delta,v:A;\Gamma\vdash t\equiv t : C}\,\mathsf{EqRefl}}{\Delta;\Gamma\vdash q\equiv \textsc{Xtrt}\, r \,\textsc{as}\, v : A \,\textsc{in}\, t : C^v_{s_1}}\,\mathsf{Eq}\Box\mathsf{E}}{\Delta;\Gamma\vdash C^v_{s_1} \mid \textsc{Xtrt}\, r \,\textsc{as}\, v : A \,\textsc{in}\, t}\,\mathsf{EqEvid}$$

where q is the proof term $\textsc{Xtrt}\, s_2 \,\textsc{as}\, v : A \,\textsc{in}\, t$

Generalized Non-deterministic Matrices and (n,k)-ary Quantifiers

Arnon Avron and Anna Zamansky

Tel Aviv University, Ramat Aviv, Israel
{aa,annaz}@post.tau.ac.il

Abstract. An (n, k)-ary quantifier is a generalized logical connective, binding k variables and connecting n formulas. Canonical Gentzen-type systems with (n, k)-ary quantifiers are systems which in addition to the standard axioms and structural rules have only logical rules in which exactly one occurrence of an (n, k)-ary quantifier is introduced. The semantics of such systems for the case of $k \in \{0, 1\}$ are provided in [16] using two-valued non-deterministic matrices (2Nmatrices). A constructive syntactic coherence criterion for the existence of a 2Nmatrix for which a canonical system is strongly sound and complete, is formulated there. In this paper we extend these results from the case of $k \in \{0, 1\}$ to the general case of $k \geq 0$. We show that the interpretation of quantifiers in the framework of Nmatrices is not sufficient for the case of $k > 1$ and introduce *generalized Nmatrices* which allow for a more complex treatment of quantifiers. Then we show that (i) a canonical calculus G is coherent iff there is a 2GNmatrix, for which G is strongly sound and complete, and (ii) any coherent canonical calculus admits cut-elimination.

1 Introduction

Propositional canonical Gentzen-type systems, introduced in [2,3], are systems which in addition to the standard axioms and structural rules have only logical rules in which exactly one occurrence of a connective is introduced and no other connective is mentioned. Intuitively, the term "canonical systems" refers to systems in which the introduction rules of a logical connective determine the semantic meaning of that connective[1]. A natural constructive *coherence* criterion can be defined for the non-triviality of such systems, and it can be shown that a canonical system admits cut-elimination iff it is coherent. The semantics of such systems are provided by two-valued non-deterministic matrices (2Nmatrices), which form a natural generalization of the classical matrix. A characteristic 2Nmatrix can be constructed for every coherent canonical propositional system.

In [16] the notion of a canonical system is extended to languages with (n, k)-ary quantifiers. An (n, k)-*ary quantifier* (for $n > 0$, $k \geq 0$) is a generalized logical connective, which binds k variables and connects n formulas. Any n-ary propositional connective can be thought of as an $(n, 0)$-ary quantifier: for instance,

[1] This is according to a long tradition in the philosophy of logic, established by Gentzen in his classical paper *"Investigations Into Logical Deduction"* ([11]).

the standard $\wedge$ connective is an $(2,0)$-ary quantifier, as it binds no variables and connects two formulas: $\wedge(\psi_1, \psi_2)$. The standard first-order quantifiers $\exists$ and $\forall$ are $(1,1)$-quantifiers, while the simplest Henkin quantifier $\mathcal{Q}^H$ ([13]) is a (4,1)-quantifier, as it binds 4 variables and connects one formula[2]:

$$\mathcal{Q}^H x_1 x_2 y_1 y_2 \psi(x_1, x_2, y_1, y_2) := \begin{matrix} \forall x_1 \ \exists y_1 \\ \forall x_2 \ \exists y_2 \end{matrix} \psi(x_1, x_2, y_1, y_2)$$

Non-deterministic matrices (Nmatrices) are a natural generalization of the standard multi-valued matrix introduced in [2,3] and extended in [4,16]. In these structures the truth-value assigned to a complex formula is chosen non-deterministically out of a given non-empty set of options. [16] use two-valued Nmatrices (2Nmatrices) extended to languages with (n,k)-ary quantifiers to provide non-deterministic semantics for canonical systems for the case of $k \in \{0,1\}$. It is shown that there is a strong connection between the coherence of a canonical calculus G and the existence of a 2Nmatrix, for which G is strongly sound and complete.

In this paper we extend these results from the case of $k \in \{0,1\}$ to the general case of $k \geq 0$. We show that the interpretation of quantifiers used in [16] is not sufficient for the case of $k > 1$ and conclude that a more general interpretation of quantifiers is needed. Then we introduce *generalized Nmatrices* (GNmatrices), a generalization of Nmatrices, in which the approach to quantifiers used in Church's type theory ([10]) is adapted. Then it is shown that the following statements concerning a canonical calculus G with (n,k)-ary quantifiers for $k \geq 0$ and $n > 0$ are equivalent: (i) G is coherent, and (ii) there exists a 2GNmatrix, for which G is strongly sound and complete. Finally, we show that any coherent canonical calculus with (n,k)-ary quantifiers admits cut-elimination.

2 Preliminaries

In what follows, L is a language with (n,k)-ary quantifiers, that is with quantifiers $\mathcal{Q}_1, ..., \mathcal{Q}_m$ with arities (n_1, k_1), ..., (n_m, k_m) respectively. For any $n > 0$ and $k \geq 0$, if a quantifier $\mathcal{Q}$ in a language L is of arity (n,k), then $\mathcal{Q}x_1...x_k(\psi_1, ..., \psi_n)$ is an L-formula whenever $x_1, ..., x_k$ are distinct variables and $\psi_1, ..., \psi_n$ are formulas of L. Denote by Frm_L (Frm_L^{cl}) the set of L-formulas (closed L-formulas). Denote by Trm_L (Trm_L^{cl}) the set of L-terms (closed L-terms). $Var = \{v_1, v_2, ..., \}$ is the set of variables of L. We use the metavariables x, y, z to range over elements of Var. Given an L-formula A, $Fv[A]$ is the set of variables occurring free in A. $\equiv_\alpha$ is the α-equivalence relation between formulas, i.e identity up to the renaming of bound variables. We use [ ] for application of functions in the meta-language, leaving the use of ( ) to the object language. We write $\mathcal{Q}\overrightarrow{x}A$ instead of $\mathcal{Q}x_1...x_kA$, and $\psi\{\overrightarrow{\mathbf{t}}/\overrightarrow{z}\}$ instead of $\psi\{\mathbf{t}_1/z_1, ..., \mathbf{t}_k/z_k\}$.

[2] In this way of recording combinations of quantifiers, dependency relations between variables are expressed as follows: an existentially quantified variable depends on those universally quantified variables which are on the left of it in the same row.

In the following two subsections, we briefly reproduce the relevant definitions from [16] of canonical systems with (n,k)-ary quantifiers and of the semantic framework of Nmatrices.

2.1 Canonical Systems with (n,k)-ary Quantifiers

We use a simplified representation language from [16] for a schematic representation of canonical rules.

Definition 1. *For $k \geq 0$, $n \geq 1$ and a set of constants Con, $L_k^n(Con)$ is the language with n k-ary predicate symbols $p_1, ..., p_n$ and the set of constants Con (and no quantifiers or connectives). The set of variables of $L_k^n(Con)$ is $Var = \{v_1, v_2, ..., \}$.*

Note that $L_k^n(Con)$ and L share the same set of variables. Henceforth we also assume[3] that for every (n,k)-ary quantifier $\mathcal{Q}$ of L, $L_k^n(Con)$ is a subset of L.

Definition 2. *Let Con be some set of constants. A* canonical quantificational rule of arity (n,k) *is an expression of the form $\{\Pi_i \Rightarrow \Sigma_i\}_{1\leq i\leq m}/C$, where $m \geq 0$, C is either $\Rightarrow \mathcal{Q}v_1...v_k(p_1(v_1,...,v_k),...,p_n(v_1,...,v_k))$ or $\mathcal{Q}v_1...v_k(p_1(v_1,...,v_k),..., p_n(v_1,...,v_k)) \Rightarrow$ for some (n,k)-ary quantifier $\mathcal{Q}$ of L and for every $1 \leq i \leq m$: $\Pi_i \Rightarrow \Sigma_i$ is a clause[4] over $L_k^n(Con)$.*

Henceforth, in cases where the set of constants Con is clear from the context (it is the set of all constants occurring in a canonical rule), we will write L_k^n instead of $L_k^n(Con)$.

A canonical rule is a schematic representation of the actual rule, while for a specific application of the rule we need to instantiate the schematic variables by the terms and formulas of L. This is done using a mapping function:

Definition 3. *Let $R = \Theta/C$ be an (n,k)-ary canonical rule, where C is of one of the forms $(\mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}),...,p_n(\overrightarrow{v})) \Rightarrow)$ or $(\Rightarrow \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}),...,p_n(\overrightarrow{v})))$. Let Γ be a set of L-formulas and $z_1,...,z_k$ - distinct variables of L. An $\langle R,\Gamma,z_1,...,z_k\rangle$-* mapping *is any function χ from the predicate symbols, terms and formulas of L_k^n to formulas and terms of L, satisfying the following conditions:*

- *For every $1 \leq i \leq n$, $\chi[p_i]$ is an L-formula. $\chi[y]$ is a variable of L, and $\chi[x] \neq \chi[y]$ for every two variables $x \neq y$. $\chi[c]$ is an L-term, such that $\chi[x]$ does not occur in $\chi[c]$ for any variable x occurring in Θ.*
- *For every $1 \leq i \leq n$, whenever $p_i(\boldsymbol{t}_1,...,\boldsymbol{t}_k)$ occurs in Θ, for every $1 \leq j \leq k$: $\chi[\boldsymbol{t}_j]$ is a term free for z_j in $\chi[p_i]$, and if $\boldsymbol{t}_j$ is a variable, then $\chi[\boldsymbol{t}_j]$ does not occur free in $\Gamma \cup \{\mathcal{Q}z_1...z_k(\chi[p_1],...,\chi[p_n])\}$.*
- *$\chi[p_i(\boldsymbol{t}_1,...,\boldsymbol{t}_k)] = \chi[p_i]\{\chi[\boldsymbol{t}_1]/z_1,...,\chi[\boldsymbol{t}_k]/z_k\}$.*

χ is extended to sets of L_k^n-formulas as follows: $\chi[\Delta] = \{\chi[\psi] \mid \psi \in \Delta\}$.

[3] This assumption is not necessary, but it makes the presentation easier, as will be explained in the sequel.

[4] By a clause we mean a sequent containing only atomic formulas.

Definition 4. *An* application *of a canonical rule of arity* (n, k) $R = \{\Pi_i \Rightarrow \Sigma_i\}_{1 \leq i \leq m} / \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v})) \Rightarrow$ *is any inference step of the form:*

$$\frac{\{\Gamma, \chi[\Pi_i] \Rightarrow \Delta, \chi[\Sigma_i]\}_{1 \leq i \leq m}}{\Gamma, \mathcal{Q}z_1...z_k\ (\chi[p_1], ..., \chi[p_n]) \Rightarrow \Delta}$$

where $z_1, ..., z_k$ *are variables,* Γ, Δ *are any sets of* L*-formulas and* χ *is some* $\langle R, \Gamma \cup \Delta, z_1, ..., z_k \rangle$*-mapping.*
An application of a canonical quantificational rule of the form $\{\Pi_i \Rightarrow \Sigma_i\}_{1 \leq i \leq m} / \Rightarrow \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))$ *is defined similarly.*

For example, the two standard introduction rules for the $(1, 1)$-ary quantifier $\forall$ can be formulated as follows: $\{p(c) \Rightarrow\} / \forall v_1\, p(v_1) \Rightarrow$ and $\{\Rightarrow p(v_1)\} / \Rightarrow \forall v_1\, p(v_1)$. Applications of these rules have the forms:

$$\frac{\Gamma, A\{\mathbf{t}/w\} \Rightarrow \Delta}{\Gamma, \forall w\, A \Rightarrow \Delta}\ (\forall \Rightarrow) \quad \frac{\Gamma \Rightarrow A\{z/w\}, \Delta}{\Gamma \Rightarrow \forall w\, A, \Delta}\ (\Rightarrow \forall)$$

where z is free for w in A, z is not free in $\Gamma \cup \Delta \cup \{\forall w A\}$, and $\mathbf{t}$ is any term free for w in A.

Notation: (Following [3,16]). Let $-t = f, -f = t$. Let $ite(t, A, B) = A$ and $ite(f, A, B) = B$. Let Φ, A^s (where Φ may be empty) denote $ite(s, \Phi \cup \{A\}, \Phi)$. For instance, the sequents $A \Rightarrow$ and $\Rightarrow A$ are denoted by $A^{-a} \Rightarrow A^a$ for $a = f$ and $a = t$ respectively. With this notation, an (n, k)-ary canonical rule has the form $\{\Sigma_j \Rightarrow \Pi_j\}_{1 \leq j \leq m} / \mathcal{Q}\overrightarrow{z}(p_1(\overrightarrow{z}), ..., p_n(\overrightarrow{z}))^{-s} \Rightarrow \mathcal{Q}\overrightarrow{z}(p_1(\overrightarrow{z}), ..., p_n(\overrightarrow{z}))^s$ for some $s \in \{t, f\}$. For further abbreviation, we denote such rule by $\{\Sigma_j \Rightarrow \Pi_j\}_{1 \leq j \leq m} / \mathcal{Q}(s)$.

Definition 5. *A Gentzen-type calculus G is* canonical *if in addition to the* α*-axiom* $A \Rightarrow A'$ *for* $A \equiv_\alpha A'$ *and the standard structural rules, G has only canonical rules.*

Definition 6. *Two* (n, k)*-ary canonical introduction rules* Θ_1/C_1 *and* Θ_2/C_2 *for* $\mathcal{Q}$ *are* dual *if for some* $s \in \{t, f\}$*:* $C_1 = A^{-s} \Rightarrow A^s$ *and* $C_2 = A^s \Rightarrow A^{-s}$*, where* $A = \mathcal{Q}v_1...v_k(p_1(v_1, ..., v_k), ..., p_n(v_1, ..., v_k))$*.*

Definition 7. *For two sets of clauses* Θ_1, Θ_2 *over* L_k^n*,* $\mathsf{Rnm}(\Theta_1 \cup \Theta_2)$ *is a set* $\Theta_1 \cup \Theta_2'$*, where* Θ_2' *is obtained from* Θ_2 *by a fresh renaming of constants and variables which occur in* Θ_1*.*

Definition 8 (Coherence)[5]**.** *A canonical calculus G is* coherent *if for every two dual canonical rules* $\Theta_1 / \Rightarrow A$ *and* $\Theta_2 / A \Rightarrow$*, the set of clauses* $\mathsf{Rnm}(\Theta_1 \cup \Theta_2)$ *is classically inconsistent.*

[5] The coherence criterion for the propositional case was first introduced in [2,3] and then extended to the first-order case in [16]. A strongly related coherence criterion was also used in [14], where linear logic is used to reason about various sequent systems. Also, the coherence criterion defined in this paper can be shown to be equivalent in the context of canonical calculi to the reductivity condition of [9] (defined for Gentzen-type systems with (n, k)-ary quantifiers which are more general than the canonical calculi), as will be explained in the sequel.

Proposition 9 (Decidability of coherence). *([16]) The coherence of a canonical calculus G is decidable.*

2.2 Non-deterministic Matrices

Non-deterministic matrices[6] (Nmatrices), were first introduced in [2,3] and extended to the first-order case in [4,17]. These structures are a generalization of the standard concept of a many-valued matrix, in which the truth-value of a formula is chosen non-deterministically from a given non-empty set of truth-values. For interpretation of quantifiers, generalized *distribution quantifiers*[7] are used.

Definition 10 *([16])* **(Non-deterministic matrix).** *A non-deterministic matrix (Nmatrix) for L is a tuple $\mathcal{M} =< \mathcal{V}, \mathcal{D}, \mathcal{O} >$, where: (i) $\mathcal{V}$ is a non-empty set of truth values, (ii) $\mathcal{D}$ (designated truth values) is a non-empty proper subset of $\mathcal{V}$, and (iii) $\mathcal{O}$ is a set of interpretation functions: for every (n,k)-ary quantifier $\mathcal{Q}$ of L, $\mathcal{O}$ includes the corresponding distribution function $\tilde{\mathcal{Q}}_{\mathcal{M}} : P^+(\mathcal{V}^n) \to P^+(\mathcal{V})$. A 2Nmatrix is any Nmatrix with $\mathcal{V} = \{t, f\}$ and $\mathcal{D} = \{t\}$.*

The notion of an L-structure is defined standardly (see, e.g. [16,4]). In order to interpret quantifiers, the substitutional approach is used, which assumes that every element of the domain has a term referring to it. Thus given a structure $S = \langle D, I \rangle$, the language L is extended with *individual constants*: $\{\overline{a} \mid a \in D\}$. Call the extended language $L(D)$. The interpretation function I is extended as follows: $I[\overline{a}] = a$.

An L-substitution σ is any function from variables to $Trm^{cl}_{L(D)}$. For an L-substitution σ and a term $\mathbf{t}$ (a formula ψ), the closed term $\sigma[\mathbf{t}]$ (the sentence $\sigma[\psi]$) is obtained from $\mathbf{t}$ (ψ) by substituting every variable x for $\sigma[x]$.

Definition 11 (Congruence of terms and formulas)[8]. *Let S be an L-structure for an Nmatrix $\mathcal{M}$. The relation $\sim^S$ between terms of $L(D)$ is defined inductively as follows: (i) $x \sim^S x$, (ii) For closed terms $\boldsymbol{t}, \boldsymbol{t}'$ of $L(D)$: $\boldsymbol{t} \sim^S \boldsymbol{t}'$ when $I[\boldsymbol{t}] = I[\boldsymbol{t}']$, (iii) If $\boldsymbol{t}_1 \sim^S \boldsymbol{t}'_1, ..., \boldsymbol{t}_n \sim^S \boldsymbol{t}'_n$, then $f(\boldsymbol{t}_1, ..., \boldsymbol{t}_n) \sim^S f(\boldsymbol{t}'_1, ..., \boldsymbol{t}'_n)$. The relation $\sim^S$ between formulas of $L(D)$ is defined as follows:*

- *If $\boldsymbol{t}_1 \sim^S \boldsymbol{t}'_1, \boldsymbol{t}_2 \sim^S \boldsymbol{t}'_2, ..., \boldsymbol{t}_n \sim^S \boldsymbol{t}'_n$, then $p(\boldsymbol{t}_1, ..., \boldsymbol{t}_n) \sim^S p(\boldsymbol{t}'_1, ..., \boldsymbol{t}'_n)$.*
- *If $\psi_1\{\overrightarrow{z}/\overrightarrow{x}\} \sim^S \varphi_1\{\overrightarrow{z}/\overrightarrow{y}\}, ..., \psi_n\{\overrightarrow{z}/\overrightarrow{x}\} \sim^S \varphi_n\{\overrightarrow{z}/\overrightarrow{y}\}$, where $\overrightarrow{x} = x_1...x_k$ and $\overrightarrow{y} = y_1...y_k$ are distinct variables and $\overrightarrow{z} = z_1...z_k$ are new distinct variables, then $\mathcal{Q}\overrightarrow{x}(\psi_1, ..., \psi_n) \sim^S \mathcal{Q}\overrightarrow{y}(\varphi_1, ..., \varphi_n)$ for any (n,k)-ary quantifier $\mathcal{Q}$ of L.*

[6] For the connection between Nmatrices and other abstract semantics, see e.g. [7].

[7] Distribution quantifiers were introduced in [6], with the intention to generalize Mostowski's proposal.

[8] The motivation for this definition is purely technical and is related to extending the language with the set of individual constants $\{\overline{a} \mid a \in D\}$. Suppose we have a closed term $\mathbf{t}$, such that $I[\mathbf{t}] = a \in D$. But a also has an individual constant $\overline{a}$ referring to it. We would like to be able to substitute $\mathbf{t}$ for $\overline{a}$ in every context.

The following is a straightforward generalization of Lemma 3.6 from [16].

Lemma 12. *Let S be an L-structure for an Nmatrix $\mathcal{M}$. Let ψ, ψ' be formulas of $L(D)$. Let $\boldsymbol{t}_1, ..., \boldsymbol{t}_n, \boldsymbol{t}'_1, ..., \boldsymbol{t}'_n$ be closed terms of $L(D)$, such that $\boldsymbol{t}_i \sim^S \boldsymbol{t}'_i$ for every $1 \leq i \leq n$. Then (1) If $\psi \equiv_\alpha \psi'$, then $\psi \sim^S \psi'$, and (2) If $\psi \sim^S \psi'$, then $\psi\{\overrightarrow{\boldsymbol{t}}/\overrightarrow{x}\} \sim^S \psi'\{\overrightarrow{\boldsymbol{t}'}/\overrightarrow{x}\}$.*

Definition 13 (Legal valuation). *Let $S = \langle D, I \rangle$ be an L-structure for a Nmatrix $\mathcal{M}$. An S-valuation $v : Frm^{cl}_{L(D)} \rightarrow \mathcal{V}$ is* legal in $\mathcal{M}$ *if it satisfies the following conditions:*

- *$v[\psi] = v[\psi']$ for every two sentences ψ, ψ' of $L(D)$, such that $\psi \sim^S \psi'$.*
- *$v[p(\boldsymbol{t}_1, ..., \boldsymbol{t}_n)] = I[p][I[\boldsymbol{t}_1], ..., I[\boldsymbol{t}_n]]$.*
- *$v[\mathcal{Q}x_1, ..., x_k(\psi_1, ..., \psi_n)$ is in the set $\tilde{\mathcal{Q}}_\mathcal{M}[\{\langle v[\psi_1\{\overline{a}_1/x_1, ..., \overline{a}_k/x_k\}], ..., v[\psi_n\{\overline{a}_1/x_1, ..., \overline{a}_k/x_k\}]\rangle \mid a_1, ..., a_k \in D\}]$ for every (n, k)-ary quantifier $\mathcal{Q}$ of L.*

Definition 14. *Let $S = \langle D, I \rangle$ be an L-structure for an Nmatrix $\mathcal{M}$.*

1. *An $\mathcal{M}$-legal S-valuation v is* a model of a sentence ψ in $\mathcal{M}$, *denoted by $S, v \models_\mathcal{M} \psi$, if $v[\psi] \in \mathcal{D}$.*
2. *Let v be an $\mathcal{M}$-legal S-valuation. A sequent $\Gamma \Rightarrow \Delta$ is* $\mathcal{M}$-valid in $\langle S, v\rangle$ *if for every S-substitution σ: if $S, v \models_\mathcal{M} \sigma[\psi]$ for every $\psi \in \Gamma$, then there is some $\varphi \in \Delta$, such that $S, v \models_\mathcal{M} \sigma[\varphi]$. A sequent $\Gamma \Rightarrow \Delta$ is $\mathcal{M}$-valid if for every L-structure S and every $\mathcal{M}$-legal S-valuation v, $\Gamma \Rightarrow \Delta$ is $\mathcal{M}$-valid in $\langle S, v\rangle$.*
3. *The consequence relation $\vdash_\mathcal{M}$ between sets of L-formulas is defined as follows: $\Gamma \vdash_\mathcal{M} \Delta$ if $\Gamma \Rightarrow \Delta$ is $\mathcal{M}$-valid.*

Definition 15. *A system G is* sound *for an Nmatrix $\mathcal{M}$ if $\vdash_G \subseteq \vdash_\mathcal{M}$. A system G is* complete *for an Nmatrix $\mathcal{M}$ if $\vdash_\mathcal{M} \subseteq \vdash_G$. An Nmatrix $\mathcal{M}$ is* characteristic *for G if G is sound and complete for $\mathcal{M}$.*

A system G is strongly sound *for $\mathcal{M}$ if for every set of sequents $\mathcal{S}$: if $\Gamma \Rightarrow \Delta$ is derivable in G from $\mathcal{S}$, then for every L-structure S and every S-substitution v, whenever $\mathcal{S}$ is $\mathcal{M}$-valid in $\langle S, v\rangle$, $\Gamma \Rightarrow \Delta$ is $\mathcal{M}$-valid in $\langle S, v\rangle$.*

Note that strong soundness implies (weak) soundness.

In addition to L-structures for languages with (n, k)-ary quantifiers, we will also use L^n_k-structures for the simplified languages L^n_k, using which the canonical rules are formulated. To make the distinction clearer, we shall use the metavariable S for the former and $\mathcal{N}$ for the latter. Since the formulas of L^n_k are always atomic, the specific 2Nmatrix for which $\mathcal{N}$ is defined is immaterial, and can be omitted. Henceforth we may speak simply of validity of sets of sequents over L^n_k.

Definition 16. *([16]) Let $\mathcal{N} = \langle D, I \rangle$ be an L^n_k-structure. $Dist_\mathcal{N}$ (the distribution of $\mathcal{N}$) is the set $\{\langle I[p_1][a_1, ..., a_k], ..., I[p_n][a_1, ..., a_k]\rangle \mid a_1, ..., a_k \in D\}$.*

3 Generalizing the Framework of Nmatrices

It is shown in [16] for the case of $k \in \{0,1\}$ that a canonical calculus has a strongly characteristic 2Nmatrix iff it is coherent. Moreover, if a 2Nmatrix $\mathcal{M}$ is suitable for a calculus G, then G is strongly sound for $\mathcal{M}$:

Definition 17 *([16]). Let G be a canonical calculus over L. A 2Nmatrix $\mathcal{M}$ is* suitable *for G if for every (n,k)-ary canonical rule $\Theta/\mathcal{Q}(s)$ of G, it holds that for every L_k^n-structure $\mathcal{N}$ in which Θ is valid: $\tilde{\mathcal{Q}}_{\mathcal{M}}[Dist_{\mathcal{N}}] = \{s\}$.*

We will now show that the above property does not hold for the case of $k > 1$. We first prove that the suitability of $\mathcal{M}$ for G is not only a sufficient, but also a *necessary* condition for the strong soundness of G for $\mathcal{M}$ for any $k \geq 0$. Then we will construct a coherent calculus with a (2,1)-ary quantifier, for which there is no suitable 2Nmatrix. This immediately implies that G has no strongly characteristic 2Nmatrix.

Proposition 18. *If a canonical calculus G is strongly sound for a 2Nmatrix $\mathcal{M}$, then $\mathcal{M}$ is suitable for G.*

Proof: Let G be a canonical calculus which is strongly sound for $\mathcal{M}$ and suppose for contradiction that $\mathcal{M}$ is not suitable for G. Then there is some (n,k)-ary canonical rule $R = \Theta/\mathcal{Q}(s)$ of G, such that there is some L_k^n-structure $\mathcal{N}$ in which Θ is valid, but $\tilde{\mathcal{Q}}_{\mathcal{M}}[Dist_{\mathcal{N}}] \neq \{s\}$. Suppose that $s = t$. Then $R = \Theta/ \Rightarrow \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))$ and $(*)$ $f \in \tilde{\mathcal{Q}}_{\mathcal{M}}[Dist_{\mathcal{N}}]$. Let S be any extension of $\mathcal{N}$ to L (recall that we assume for simplicity that L_k^n is a subset of L). It is easy to see that Θ is also $\mathcal{M}$-valid in $\langle S, v\rangle$ for every S-valuation v (note that Θ only contains atomic formulas). Obviously, $\Rightarrow \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))$ is derivable from Θ in G. Now since G is strongly sound for $\mathcal{M}$, $(**)$ $\mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))$ should also be $\mathcal{M}$-valid in $\langle S, v\rangle$ for every S-valuation v. Let v_0 be any $\mathcal{M}$-legal S-valuation, such that $v_0[\mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))] = f$ (the existence of such a valuation follows from $(*)$ and the fact that $\{\langle v_0[p_1\{\overrightarrow{a}/\overrightarrow{v}\}], ..., v[p_n\{\overrightarrow{a}/\overrightarrow{v}\}]\rangle \mid a_1, ..., a_k \in D\} = Dist_{\mathcal{N}}$). Obviously $\Rightarrow \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))$ is not $\mathcal{M}$-valid in $\langle S, v_0\rangle$, in contradiction to $(**)$. □

Next, consider the calculus G, which consists of the following two dual introduction rules $\Theta_1/ \Rightarrow \mathcal{Q}v_1v_2p(v_1,v_2)$ and $\Theta_2/\mathcal{Q}v_1v_2p(v_1,v_2) \Rightarrow$, where $\Theta_1 = \{p(c,v_1) \Rightarrow\}$ and $\Theta_2 = \{\Rightarrow p(v_1,c)\}$. The set of clauses $\mathsf{Rnm}(\Theta_1 \cup \Theta_2) = \{p(c,v_1) \Rightarrow, \Rightarrow p(v_2,d)\}$ is classically inconsistent, and so G is coherent. Suppose by contradiction that there is a 2Nmatrix $\mathcal{M}$ suitable for G. Consider the L_k^n-structures $\mathcal{N}_1 = \langle D_1, I_1\rangle$ and $\mathcal{N}_2 = \langle D_2, I_2\rangle$, defined as follows. $D_1 = D_2 = \{a_1, a_2\}$, $I_1[p][a_1,a_1] = I_1[a_1,a_2] = f$, $I_1[p][a_2,a_1] = I_1[p][a_2,a_2] = t$, $I_1[c] = a_1$. $I_2[p][a_1,a_1] = I_2[a_1,a_2] = t$, $I_2[p][a_2,a_1] = I_2[p][a_2,a_2] = f$, $I_2[c] = a_1$. Obviously, Θ_1 is valid in $\mathcal{N}_1$, and so by suitability of $\mathcal{M}$, $\tilde{\mathcal{Q}}_{\mathcal{M}}[Dist_{\mathcal{N}_1}] = t$. Θ_2 is valid in $\mathcal{N}_2$, and so $\tilde{\mathcal{Q}}_{\mathcal{M}}[Dist_{\mathcal{N}_2}] = f$. But this is impossible, since $Dist_{\mathcal{N}_1} = Dist_{\mathcal{N}_2} = \{t, f\}$. Thus G has no suitable 2Nmatrix, although it is coherent. By Prop. 18 above, G has no strongly characteristic 2Nmatrix.

We conclude that the interpretation of (n,k)-ary quantifiers using distributions is not sufficient for the case of $k > 1$. Using them, we cannot capture any kind of dependencies between elements of the domain. For instance, there is no way we can express the fact that there exists an element b in the domain, such that for *every* element a, $p(a,b)$ holds. It is clear that a more general interpretation of a quantifier is needed.

We will generalize the interpretation of quantifiers as follows. Given an L-structure $S = \langle D, I\rangle$, an interpretation of an (n,k)-ary quantifier $\mathcal{Q}$ in S is an operation $\tilde{\mathcal{Q}}_S : (D^k \to \mathcal{V}^n) \to P^+(\mathcal{V})$, which for every function (from k-ary vectors of the domain elements to n-ary vectors of truth-values) returns a non-empty set of truth-values.

Definition 19. *A* generalized non-deterministic matrix *(henceforth GNmatrix) for L is a tuple $\mathcal{M} =< \mathcal{V}, \mathcal{D}, \mathcal{O} >$, where:*

- *$\mathcal{V}$ is a non-empty set of truth values.*
- *$\mathcal{D}$ is a non-empty proper subset of $\mathcal{V}$.*
- *For every (n,k)-ary quantifier $\mathcal{Q}$ of L, $\mathcal{O}$[9] includes a corresponding operation $\tilde{\mathcal{Q}}_S : (D^k \to \mathcal{V}^n) \to P^+(\mathcal{V})$ for every L-structure $S = \langle D, I\rangle$. A 2GNmatrix is any GNmatrix with $\mathcal{V} = \{t, f\}$ and $\mathcal{D} = \{t\}$.*

Examples:

1. Given an L-structure $S = \langle D, I\rangle$, the standard $(1,1)$-ary quantifier $\forall$ is interpreted as follows for any $g \in D \to \{t, f\}$: $\tilde{\forall}_S[g] = \{t\}$ if for every $a \in D$, $g[a] = t$, and $\tilde{\forall}_S[g] = \{f\}$ otherwise. The standard $(1,1)$-ary quantifier $\exists$ is interpreted as follows for any $g \in D \to \{t, f\}$: $\tilde{\exists}_S[g] = \{t\}$ if there exists some $a \in D$, such that $g[a] = t$, and $\tilde{\exists}_S[g] = \{f\}$ otherwise.
2. Given an L-structure $S = \langle D, I\rangle$, the $(1,2)$-ary bounded universal[10] quantifier $\overline{\forall}$ is interpreted as follows: for any $g \in D \to \{t, f\}^2$, $\tilde{\overline{\forall}}_S[g] = \{t\}$ if for every $a \in D$, $g[a] \neq \langle t, t\rangle$, and $\tilde{\overline{\forall}}_S[g] = \{f\}$ otherwise. The $(1,2)$-ary bounded existential[11] quantifier $\overline{\exists}$ is interpreted as follows: for any $g \in D \to \{t, f\}^2$, $\tilde{\overline{\forall}}_S[g] = \{t\}$ if there exists some $a \in D$, such that $g[a] = \langle t, t\rangle$, and $\tilde{\overline{\forall}}_S[g] = \{f\}$ otherwise.
3. Consider the $(2,2)$-ary quantifier $\mathcal{Q}$, with the intended meaning of $Qxy(\psi_1, \psi_2)$ as $\exists y \forall x(\psi_1(x,y) \wedge \neg\psi_2(x,y))$. Its interpretation for every L-structure $S = \langle D, I\rangle$, every $g \in D^2 \to \{t, f\}^2$ is as follows: $\tilde{Q}_S[g] = t$ iff there exists some $a \in D$, such that for every $b \in D$: $g[a,b] = \langle t, f\rangle$.
4. Consider the $(4,1)$-ary Henkin quantifier[12] $\mathcal{Q}_H$ discussed in section 1. Its interpretation for for every L-structure $S = \langle D, I\rangle$ and every $g \in D^4 \to \{t, f\}$

[9] In the current definition, $\mathcal{O}$ is a class and the tuple $\langle \mathcal{V}, \mathcal{D}, \mathcal{O}\rangle$ is not well-defined. We can overcome this technical problem by assuming that the domains of all the structures are prefixes of the set of natural numbers. A more general solution to this problem is a question for further research.

[10] The intended meaning of $\overline{\forall}x(p_1(x), p_2(x))$ is $\forall x(p_1(x) \to p_2(x))$.

[11] The intended meaning of $\overline{\exists}x(p_1(x), p_2(x))$ is $\exists x(p_1(x) \wedge p_2(x))$.

[12] We note that the current framework of canonical systems is not adequate to handle such quantifiers.

is as follows: $\tilde{\mathcal{Q}}^H_S[g] = \{t\}$ if for every $a \in D$ there exists some $b \in D$ and for every $c \in D$ there exists some $d \in D$, such that $g[a,b,c,d] = t$. $\tilde{\mathcal{Q}}^H_S[g] = \{f\}$ otherwise.

Definition 20 (Legal valuation). *Let $S = \langle D, I\rangle$ be an L-structure for a GNmatrix $\mathcal{M}$. An S-valuation $v : Frm^{\mathsf{cl}}_{L(D)} \to \mathcal{V}$ is* legal in $\mathcal{M}$ *if it satisfies the following conditions: $v[\psi] = v[\psi']$ for every two sentences ψ, ψ' of $L(D)$, such that $\psi \sim^S \psi'$, $v[p(\boldsymbol{t}_1, ..., \boldsymbol{t}_n)] = I[p][I[\boldsymbol{t}_1], ..., I[\boldsymbol{t}_n]]$, and $v[\mathcal{Q}x_1, ..., x_k(\psi_1, ..., \psi_n)$ is in the set $\tilde{\mathcal{Q}}_S[\lambda a_1, ..., a_k \in D.\langle v[\psi_1\{\overline{a}_1/x_1, ..., \overline{a}_k/x_k\}], ..., v[\psi_n\{\overline{a}_1/x_1, ..., \overline{a}_k/x_k\}]\rangle]$ for every (n,k)-ary quantifier $\mathcal{Q}$ of L.*

The semantic notions from Defn. 14 and 15 are defined similarly for the case of GNmatrices.

Next we generalize the notion of a *distribution* of L^n_k-structures (see Defn. 16).

Definition 21. *Let $\mathcal{N} = \langle D, I\rangle$ be a structure for L^n_k. The* functional distribution *of $\mathcal{N}$ is a function $FDist_{\mathcal{N}} \in D^k \to \{t,f\}^n$, such that: $FDist_{\mathcal{N}} = \lambda a_1, ..., a_k \in D.\langle I[p_1][a_1, ..., a_k], ..., I[p_n][a_1, ..., a_k]\rangle$.*

4 Semantics for Canonical Calculi

In this section we show that a canonical calculus G with (n,k)-ary quantifiers is coherent iff it has a strongly characteristic 2GNmatrix.
First we construct a strongly characteristic 2GNmatrix for every coherent canonical calculus.

Definition 22. *Let G be a coherent canonical calculus. For every L-structure $S = \langle D, I\rangle$, the GNmatrix $\mathcal{M}_G$ contains the operation $\tilde{\mathcal{Q}}_S$ defined as follows. For every (n,k)-ary quantifier $\mathcal{Q}$ of L, every $r \in \{t,f\}$ and every $g \in D^k \to \{t,f\}^n$:*

$$\tilde{\mathcal{Q}}_S[g] = \begin{cases} \{r\} & \Theta/\mathcal{Q}(r) \in G \text{ and there is an } L^n_k - \text{structure } \mathcal{N} = \langle D_{\mathcal{N}}, I_{\mathcal{N}}\rangle \\ & \text{such that } D_{\mathcal{N}} = D,\ FDist_{\mathcal{N}} = g \text{ and } \Theta \text{ is valid in } \mathcal{N}. \\ \{t,f\} & \text{otherwise} \end{cases}$$

It should be noted that as opposed to the definition of the Nmatrix $\mathcal{M}_G$ in [16] (see Defn. 4.2 there), the above definition is not constructive. This is because the question whether Θ is valid in some L^n_k-structure with a given functional distribution is not generally decidable. Next, let us show that $\mathcal{M}_G$ is well-defined. Assume by contradiction that there are two dual rules $\Theta_1/ \Rightarrow A$ and $\Theta_2/A \Rightarrow$, such that there exist two L^n_k-structures $\mathcal{N}_1 = \langle D, I_1\rangle$ and $\mathcal{N}_2 = \langle D, I_2\rangle$, which satisfy: $FDist_{\mathcal{N}_1} = FDist_{\mathcal{N}_2}$ and Θ_i is valid in $\mathcal{N}_i$ for $i \in \{1,2\}$. But then $\mathcal{N}_1$ and $\mathcal{N}_2$ only differ in their interpretations of constants from Θ_1 and Θ_2. Then we can easily construct an L^n_k-structure $\mathcal{N}_3 = \langle D, I_3\rangle$, such that $\mathsf{Rnm}(\Theta_1 \cup \Theta_2)$ is valid in $\mathcal{N}_3$ (the renaming is essential since it may be the case that the same constant occurs both in Θ_1 and Θ_2). And so $\mathsf{Rnm}(\Theta_1 \cup \Theta_2)$ is classically consistent, in contradiction to the coherence of G.

Theorem 23. *Any coherent canonical calculus G is strongly sound for $\mathcal{M}_G$.*

Proof: Let $S = \langle D, I\rangle$ be some L-structure and v - an $\mathcal{M}$-legal S-valuation. Let $\mathcal{S}$ be any set of sequents closed under substitution. We will show that if the sequents of $\mathcal{S}$ are $\mathcal{M}$-valid in $\langle S, v\rangle$, then any sequent provable from $\mathcal{S}$ in G is $\mathcal{M}$-valid in $\langle S, v\rangle$. Obviously, the axioms of G are $\mathcal{M}$-valid, and the structural rules, including cut, are strongly sound. It remains to show that for every application of a canonical rule R of G: if the premises of R are $\mathcal{M}$-valid in $\langle S, v\rangle$, then its conclusion is $\mathcal{M}$-valid in $\langle S, v\rangle$. Let R be an (n, k)-ary rule of G of the form: $R = \Theta_R / \Rightarrow \mathcal{Q}\overrightarrow{v}(p_1(\overrightarrow{v}), ..., p_n(\overrightarrow{v}))$, where $\Theta_R = \{\Sigma_j \Rightarrow \Pi_j\}_{1\leq j\leq m}$. An application of R is of the form:

$$\frac{\{\Gamma, \chi[\Sigma_j] \Rightarrow \chi[\Pi_j], \Delta\}_{1\leq j\leq m}}{\Gamma \Rightarrow \Delta, \mathcal{Q}\overrightarrow{z}(\chi[p_1], ..., \chi[p_n])}$$

where χ is some $\langle R, \Gamma\cup\Delta, \overrightarrow{z}\rangle$-mapping. Let $\{\Gamma, \chi[\Sigma_j] \Rightarrow \chi[\Pi_j], \Delta\}_{1\leq j\leq m}$ be $\mathcal{M}$-valid in $\langle S, v\rangle$. Let σ be an S-substitution, such that $S, v \models_{\mathcal{M}} \sigma[\Gamma]$ and for every $\psi \in \Delta$: $S, v \not\models_{\mathcal{M}} \sigma[\psi]$. Denote by $\widetilde{\psi}$ the L-formula obtained from a formula ψ by substituting every free occurrence of $w \in Fv[\psi] - \{z\}$ for $\sigma[w]$. Construct the L^n_k-structure $\mathcal{N} = \langle D_{\mathcal{N}}, I_{\mathcal{N}}\rangle$ as follows: $D_{\mathcal{N}} = D$, for every $a_1, ..., a_k \in D$: $I_{\mathcal{N}}[p_i][a_1, ..., a_k] = v[\widetilde{\chi[p_i]}\{\overrightarrow{a}/\overrightarrow{z}\}]$, and for every constant c occurring in Θ_R, $I_{\mathcal{N}}[c] = I[\sigma[\chi[c]]]$. It is not difficult to show that Θ_R is valid in $\mathcal{N}$. Thus by definition of $\mathcal{M}_G$, $\tilde{\mathcal{Q}}_S[FDist_{\mathcal{N}}] = \{t\}$. Finally, by definition of $\mathcal{N}$, $FDist_{\mathcal{N}} = \lambda a_1, ..., a_k \in D.\{\langle v[\widetilde{\chi[p_1]}\{\overrightarrow{a}/\overrightarrow{z}\}], ..., v[\widetilde{\chi[p_n]}\{\overrightarrow{a}/\overrightarrow{z}\}]\rangle\}$. Since v is $\mathcal{M}$-legal, $v[\sigma[\mathcal{Q}\overrightarrow{z}(\chi[p_1], ..., \chi[p_n])]] = v[\mathcal{Q}\overrightarrow{z}(\widetilde{\chi[p_1]}, ..., \widetilde{\chi[p_n]})] \in FDist_{\mathcal{N}} = \{t\}$. And so $\Gamma \Rightarrow \Delta, \mathcal{Q}\overrightarrow{z}(\chi[p_1], ..., \chi[p_n])$ is $\mathcal{M}$-valid in $\langle S, v\rangle$. □

Example 1. The canonical calculus G_1 consists of (1,1)-ary rule $\Rightarrow p(v_1)/ \Rightarrow \forall v_1 p(v_1)$. Clearly, G_1 is coherent. For every L-structure $S = \langle D, I\rangle$, $\mathcal{M}_{G_1}$ contains the operation $\tilde{\forall}_S$ defined as follows for every $g \in D \to \mathcal{V}$:

$$\tilde{\forall}_S[g] = \begin{cases} \{t\} & \textit{if for all } a \in D: \ g[a] = t \\ \{t, f\} & \textit{otherwise} \end{cases}$$

Example 2. The canonical calculus G_2 consists of the following rules: (i) $\{p_1(v_1) \Rightarrow p_2(v_1)\}/ \Rightarrow \overline{\forall} v_1\ (p_1(v_1), p_2(v_1))$, (ii) $\{p_2(c) \Rightarrow\ , \ \Rightarrow p_1(c)\}/\overline{\forall} v_1(p_1(v_1), p_2(v_1)) \Rightarrow$ and (iii) $\{\Rightarrow p_1(c)\ ,\ \Rightarrow p_2(c)\}/ \Rightarrow \overline{\exists} v_1(p_1(v_1), p_2(v_1))$. G' is obviously coherent. The operations $\tilde{\overline{\forall}}_S$ and $\tilde{\overline{\exists}}_S$ in $\mathcal{M}_{G_2}$ are defined as follows for every $g \in D \to \{t, f\}^2$:

$$\tilde{\overline{\forall}}_S[g] = \begin{cases} \{t\} & \textit{if there are no such } a, b \in D, \textit{ that } g[a, b] = \langle t, f\rangle \\ \{f\} & \textit{otherwise} \end{cases}$$

$$\tilde{\overline{\exists}}_S[g] = \begin{cases} \{t\} & \textit{if there are } a, b \in D, s.t.\ g[a, b] = \langle t, t\rangle \\ \{t, f\} & \textit{otherwise} \end{cases}$$

The rule (i) dictates the condition that $\overline{\forall}_S[g] = \{t\}$ for the case that there are no $a, b \in D$, s.t. $g[a, b] = \langle t, f\rangle$. The rule (ii) dictates the condition that $\overline{\forall}_S[g] = \{f\}$ for the case that there are such $a, b \in D$. Since G_2 is coherent, the dictated conditions are non-contradictory. The rule (iii) dictates the condition that $\overline{\exists}_S[g] = \{t\}$ in the case that there are $a, b \in D$, s.t. $g[a, b] = \langle t, t\rangle$. There is no rule which dictates conditions for the case of $\langle t, t\rangle \notin H$, and so the interpretation in this case is non-deterministic.

Example 3. Consider the canonical calculus G_3 consisting of the following $(2,2)$-ary rule: $\{p_1(v_1, v_2) \Rightarrow\ ;\ \Rightarrow p_2(c, v_1)\}/ \Rightarrow \mathcal{Q}v_1v_2(p_1(v_1, v_2), p_2(v_1, v_2))$. G_3 is (trivially) coherent. For a tuple $v = \langle a_1, ..., a_n\rangle$, denote by $(v)_i$ the i-th element of v. For every L-structure $S = \langle D, I\rangle$, $\mathcal{M}_{G_3}$ contains the operation $\tilde{\mathcal{Q}}_S$ defined as follows for every $g \in D^2 \to \{t, f\}^2$:

$$\tilde{\mathcal{Q}}_S[g] = \begin{cases} \{t\} & \textit{if there is some } a \in D, \textit{ s.t. for every } b, c \in D \\ & (g[b, c])_1 = f \textit{ and } (g[a, b])_2 = t \\ \{t, f\} & \textit{otherwise} \end{cases}$$

Next we show that for every canonical calculus G: (i) $\mathcal{M}_G$ is a characteristic 2Nmatrix for G, and (ii) G admits cut-elimination. For this we first prove the following proposition.

Proposition 24. *Let G be a coherent calculus. Let $\Gamma \Rightarrow \Delta$ be a sequent which satisfies the free-variable condition*[13]*. If $\Gamma \Rightarrow \Delta$ has no cut-free proof in G, then $\Gamma \not\vdash_{\mathcal{M}_G} \Delta$.*

Proof: Let $\Gamma \Rightarrow \Delta$ be a sequent which satisfies the free-variable condition. Suppose that $\Gamma \Rightarrow \Delta$ has no cut-free proof from in G. To show that $\Gamma \Rightarrow \Delta$ is not $\mathcal{M}_G$-valid, we will construct an L-structure S, an S-substitution σ^* and an $\mathcal{M}_G$-legal valuation v, such that $v[\sigma^*[\psi]] = t$ for every $\psi \in \Gamma$, while $v[\sigma^*[\varphi]] = f$ for every $\varphi \in \Delta$.

It is easy to see that we can limit ourselves to the language L^*, which is a subset of L, consisting of all the constants and predicate and function symbols, occurring in $\Gamma \Rightarrow \Delta$.

Let $\mathbf{T}$ be the set of all the terms in L^* which do not contain variables occurring bound in $\Gamma \Rightarrow \Delta$. It is a standard matter to show that Γ, Δ can be extended to two (possibly infinite) sets Γ', Δ' (where $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$), satisfying the following properties:

1. For every finite $\Gamma_1 \subseteq \Gamma'$ and $\Delta_1 \subseteq \Delta'$, $\Gamma_1 \Rightarrow \Delta_1$ has no cut-free proof in G.
2. There are no $\psi \in \Gamma'$ and $\varphi \in \Delta'$, such that $\psi \equiv_\alpha \varphi$. n
3. If $\{\Pi_j \Rightarrow \Sigma_j\}_{1 \le j \le m}/\mathcal{Q}(r)$ is an (n, k)-ary rule of G and $\mathcal{Q}z_1...z_k\ (A_1, ..., A_n) \in ite(r, \Delta', \Gamma')$, then there is some $1 \le j \le m$ satisfying the following condition. Let $\mathbf{t}_1, ..., \mathbf{t}_m$ be the L^n_k-terms occurring in $\Pi_j \cup \Sigma_j$, where $\mathbf{t}_{j_1}, ..., \mathbf{t}_{j_l}$ are constants and $\mathbf{t}_{j_{l+1}}, ..., \mathbf{t}_{j_m}$ are variables. Then for every $\mathbf{s}_1, ..., \mathbf{s}_l \in \mathbf{T}$ there are

[13] A sequent $\mathcal{S}$ satisfies the free-variable condition if the set of variables occurring free in $\mathcal{S}$ and the set of variables occurring bound in $\mathcal{S}$ are disjoint.

some[14] $\mathbf{s}_{l+1},...,\mathbf{s}_m \in \mathbf{T}$, such that whenever $p_i(\mathbf{t}_{n_1},...,\mathbf{t}_{n_k}) \in ite(r, \Pi_j, \Sigma_j)$ for some $1 \leq n_1,...,n_k \leq m$: $A_i\{\mathbf{s}_{n_1}/z_1,...,\mathbf{s}_{n_k}/z_k\} \in ite(r, \Gamma', \Delta')$.

Let $S = \langle D, I\rangle$ be the L^*-structure defined as follows: $D = \mathbf{T}$, $I[c] = c$ for every constant c of L^*; $I[f][\mathbf{t}_1,...,\mathbf{t}_n] = f(\mathbf{t}_1,...,\mathbf{t}_n)$ for every n-ary function symbol f; $I[p][\mathbf{t}_1,...,\mathbf{t}_n] = t$ iff $p(\mathbf{t}_1,...,\mathbf{t}_n) \in \Gamma'$ for every n-ary predicate symbol p. It is easy to show by induction on $\mathbf{t}$ that: $(*)$ For every $\mathbf{t} \in \mathbf{T}$: $I[\sigma^*[\mathbf{t}]] = \mathbf{t}$.

Let σ^* be any S-substitution satisfying $\sigma^*[x] = \overline{x}$ for every $x \in \mathbf{T}$. (Note that every $x \in \mathbf{T}$ is also a member of the domain and thus has an individual constant referring to it in $L^*(D)$).

For an $L(D)$-formula ψ (an $L(D)$-term $\mathbf{t}$), we will denote by $\widehat{\psi}$ ($\widehat{\mathbf{t}}$) the L-formula (L-term) obtained from ψ ($\mathbf{t}$) by replacing every individual constant of the form $\overline{\mathbf{s}}$ for some $\mathbf{s} \in \mathbf{T}$ by the term $\mathbf{s}$. Then the following property can be proved by an induction on ψ: $(**)$ For every $\psi \in \Gamma' \cup \Delta'$: $\widehat{\sigma^*[\psi]} = \psi$.

Define the S-valuation v as follows: (i) $v[p(\mathbf{t}_1,...,\mathbf{t}_n)] = I[p][I[\mathbf{t}_1],...,I[\mathbf{t}_n]]$, (ii) If there is some $C \in \Gamma' \cup \Delta'$, s.t. $C \equiv_\alpha \widehat{\mathcal{Q}\overrightarrow{z}(\psi_1,...,\psi_n)}$, then $v[\mathcal{Q}\overrightarrow{z}(\psi_1,...,\psi_n)] = t$ iff $C \in \Gamma'$. Otherwise $v[\mathcal{Q}\overrightarrow{z}(\psi_1,...,\psi_n)] = t$ iff $\tilde{\mathcal{Q}}_S[\lambda a_1...a_k \in D.\{\langle v[\psi_1\{\overrightarrow{\overline{a}}/\overrightarrow{z}\}],..., v[\psi_n\{\overrightarrow{\overline{a}}/\overrightarrow{z}\}]\rangle\}] = \{t\}$.

It is not difficult to show that v is legal in $\mathcal{M}_G$.

Next we show that for every $\psi \in \Gamma' \cup \Delta'$: $v[\sigma^*[\psi]] = t$ iff $\psi \in \Gamma'$. If $\psi = p(\mathbf{t}_1,...,\mathbf{t}_n)$, then $v[\sigma^*[\psi]] = I[p][I[\sigma^*[\mathbf{t}_1]],...,I[\sigma^*[\mathbf{t}_n]]]$. Note[15] that for every $1 \leq i \leq n$, $\mathbf{t}_i \in \mathbf{T}$. By $(*)$, $I[\sigma^*[\mathbf{t}_i]] = \mathbf{t}_i$, and by the definition of I, $v[\sigma^*[\psi]] = t$ iff $p(\mathbf{t}_1,...,\mathbf{t}_n) \in \Gamma'$. Otherwise $\psi = Q\overrightarrow{z}(\psi_1,...,\psi_n)$. If $\psi \in \Gamma'$, then by $(**)$: $\widehat{\sigma^*[\psi]} = \psi \in \Gamma'$ and so $v[\sigma^*[\psi]] = t$. If $\psi \in \Delta'$ then by property 2 of $\Gamma' \cup \Delta'$ it cannot be the case that there is some $C \in \Gamma'$, such that $C \equiv_\alpha \widehat{\sigma^*[\psi]} = \psi$ and so $v[\sigma^*[\psi]] = f$.

We have constructed an L-structure S, an S-substitution σ^* and an $\mathcal{M}_G$-legal valuation v, such that $v[\sigma^*[\psi]] = t$ for every $\psi \in \Gamma'$, while $v[\sigma^*[\varphi]] = f$ for every $\varphi \in \Delta'$. Since $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$, $\Gamma \Rightarrow \Delta$ is not $\mathcal{M}_G$-valid. □

Theorem 25. *Let G be a canonical calculus. Then the following statements concerning G are equivalent:*

1. *G is coherent.*
2. *There exists a 2GNmatrix $\mathcal{M}$, such that G is strongly sound and complete for $\mathcal{M}$.*

Proof: (1) $\Rightarrow$ (2):
Suppose that G is coherent. By theorem 23, G is strongly sound for $\mathcal{M}_G$. For completeness, let $\Gamma \Rightarrow \Delta$ be a sequent which has no proof in G. If it does not satisfy the free-variable condition, obtain a sequent $\Gamma' \Rightarrow \Delta'$ which does satisfy this condition by renaming the bound variables. (Otherwise, set $\Gamma' = \Gamma$ and

[14] Note that in contrast to $\mathbf{t}_1,...,\mathbf{t}_m$, $\mathbf{s}_1,...,\mathbf{s}_m$ are L-terms and not L^n_k-terms.

[15] This is obvious if $\mathbf{t}_i$ does not occur in $\Gamma \Rightarrow \Delta$. If it occurs in $\Gamma \Rightarrow \Delta$, then since $\Gamma \Rightarrow \Delta$ satisfies the free-variable condition, $\mathbf{t}_i$ does not contain variables bound in this set and so $\mathbf{t}_i \in \mathbf{T}$ by definition of $\mathbf{T}$.

$\Delta' = \Delta$). Then also $\Gamma' \Rightarrow \Delta'$ has no proof in G (otherwise we could obtain a proof of $\Gamma \Rightarrow \Delta$ from a proof of $\Gamma' \Rightarrow \Delta'$ by using cuts on logical axioms). By proposition 24, $\Gamma' \not\vdash_{\mathcal{M}_G} \Delta'$. That is, there is an L-structure S, an S-substitution σ and an $\mathcal{M}_G$-legal valuation v, such that $v[\sigma[\psi]] = t$ for every $\psi \in \Gamma'$, while $v[\sigma[\varphi]] = f$ for every $\varphi \in \Delta'$. By lemma 12-1, v respects the $\equiv_\alpha$-relation, and so $v[\sigma[\psi]] = t$ for every $\psi \in \Gamma$, while $v[\sigma[\varphi]] = f$ for every $\varphi \in \Delta$. Hence, $\Gamma \not\vdash_{\mathcal{M}_G} \Delta$, and G is complete (and strongly sound) for $\mathcal{M}_G$.
(2) $\Rightarrow$ (1):

Suppose that G is strongly sound and complete for some 2GNmatrix $\mathcal{M}$. Assume by contradiction that G is not coherent. Then there exist two dual (n, k)-ary rules $R_1 = \Theta_1 / \Rightarrow A$ and $R_2 = \Theta_2 / A \Rightarrow$ in G, such that $\mathsf{Rnm}(\Theta_1 \cup \Theta_2)$ is classically consistent. Recall that $\mathsf{Rnm}(\Theta_1 \cup \Theta_2) = \Theta_1 \cup \Theta_2'$, where Θ_2' is obtained from Θ_2 by renaming constants and variables that occur also in Θ_1 (see defn. 7). For simplicity[16] we assume that the fresh constants used for renaming are all in L. Since $\Theta_1 \cup \Theta_2'$ is classically consistent, there exists an L_k^n-structure $\mathcal{N} = \langle D, I \rangle$, in which both Θ_1 and Θ_2' are valid. Recall that we also assume that L_k^n is a subset of L[17] and so $\dfrac{\Theta_1}{\Rightarrow A}$ and $\dfrac{\Theta_2'}{A \Rightarrow}$ are applications of R_1 and R_2 respectively. Let S be any extension of $\mathcal{N}$ to L and v - any $\mathcal{M}$-legal S-valuation. It is easy to see that Θ_1 and Θ_2' are $\mathcal{M}$-valid in $\langle S, v \rangle$ (since they only contain atomic formulas). Since G is strongly sound for $\mathcal{M}$, both $\Rightarrow A$ and $A \Rightarrow$ should also be $\mathcal{M}$-valid in $\langle S, v \rangle$, which is of course impossible. □

Corollary 26. *The existence of a strongly characteristic 2GNmatrix for a canonical calculus G is decidable.*

Proof: By Theorem 25, the question whether G has a strongly characteristic 2Nmatrix is equivalent to the question whether G is coherent, and this, by Proposition 9, is decidable.

Corollary 27. *If G is a coherent canonical calculus then it admits cut-elimination.*

As was shown in [16], the opposite does not hold: a canonical calculus which is not coherent can still admit cut-elimination.

Remark: The above results are related to the results in [9], where a general class of sequent calculi with (n, k)-ary quantifiers, called *standard* calculi is defined. Standard calculi may include any set of structural rules, and so canonical calculi are a particular instance of standard calculi which include all of the standard structural rules. [9] formulate syntactic sufficient and (under some limitations) necessary conditions for modular cut-elimination, a particular version of cut-elimination with non-logical axioms consisting only of atomic formulas. The

[16] This assumption is not necessary and is used only for simplification of presentation, since we can instantiate the constants by any L-terms.
[17] This assumption is again not essential for the proof, but it simplifies the presentation.

reductivity condition of [9] can be shown to be equivalent to our coherence criterion in the context of canonical systems. Thus from the results of [9] it follows that coherence is a necessary condition for modular cut-elimination in canonical calculi.

5 Summary and Further Research

In this paper we have extended the results of [16] for canonical systems with (n, k)-ary quantifiers from the case of $k \in \{0, 1\}$ to the general case of $k \geq 0$ (while preserving the decidability of coherence). We have demonstrated that the framework of Nmatrices is not sufficient to provide semantics for canonical systems for the case of $k > 1$, and generalized the framework of Nmatrices by introducing more general interpretations of quantifiers. Then we have shown that a canonical calculus G is coherent iff there is a 2GNmatrix $\mathcal{M}$ for which G is strongly sound and complete. Furthermore, any coherent calculus admits cut-elimination. However, the opposite direction does not hold: we have seen that coherence is not a necessary condition for (standard) cut-elimination in canonical calculi. From the results of [9] it follows that coherence *is* a necessary condition for modular cut-elimination. Whether it is possible to extend these results to more general forms of cut-elimination, is a question for further research.

Other research directions include extending the results of this paper to more general systems, such as the standard calculi of [9], which use non-standard sets of structural rules, and treating more complex quantifier extensions, such as Henkin quantifiers. Although the syntactic formulation of canonical systems given in this paper is not expressible enough to deal with Henkin quantifiers, the proposed semantic framework of GNmatrices provides an adequate interpretation of such quantifiers. This might be a promising starting point for a proof-theoretical investigation of canonical systems with Henkin quantifiers.

Yet another research direction is gaining an insight into the connection between non-determinism and axiom expansion in canonical systems. In [5] it is shown (on the propositional level) that any many-sided calculus which satisfies: (i) a condition similar to coherence and (ii) axiom expansion[18] (i.e axioms can be reduced to atomic axioms), has a deterministic characteristic matrix. We conjecture that there is a direct connection between axiom expansion in a coherent canonical system, and the degree of non-determinism in its characteristic 2Nmatrix.

Acknowledgement

This research was supported by the *Israel Science Foundation* founded by the Israel Academy of Sciences and Humanities (grant No 809/06).

[18] This is also closely connected to the criterion of rigidity of [8] for propositional simple calculi.

References

1. Avron, A., ‘Gentzen-Type Systems, Resolution and Tableaux’, *Journal of Automated Reasoning*, vol. 10, 265–281, 1993.
2. Avron, A. and I. Lev, ‘Canonical Propositional Gentzen-type Systems’, *Proceedings of the 1st International Joint Conference on Automated Reasoning (IJCAR 2001)*, R. Gore, A. Leitsch, T. Nipkow, eds., Springer Verlag, LNAI 2083, 529-544, Springer Verlag, 2001.
3. Avron, A. and I. Lev, ‘Non-deterministic Multi-valued Structures’, *Journal of Logic and Computation*, vol. 15, 241–261, 2005.
4. Avron, A. and A. Zamansky, ‘Quantification in Non-deterministic Multi-valued Structures’, *Proceedings of the 35th IEEE International Symposium on Multiple-Valued Logics*, 296–301, IEEE Computer Society Press, 2005.
5. Baaz M., C.G. Fermüller, G. Salzer and R.Zach, ‘Labeled Calculi and Finite-valued Logics’, *Studia Logica*, vol. 61, 7–33, 1998.
6. Carnielli W.A., ‘Systematization of Finite Many-valued Logics through the method of Tableaux’, *Journal of Symbolic Logic*, vol. 52 (2), 473–493, 1987.
7. Carnielli W.A. and M.E. Conglio,‘Splitting Logics’, in *We Will Show Them!, Essays in Honour of Dov Gabbay*, Artemov, Barringer, Garcez and Lamb eds., Woods College Publications, vol. 1, 389–414, 2005.
8. Ciabattoni A. and Terui K., ‘Towards a semantic characterization of cut elimination’, *Studia Logica*, vol. 82(1), 95–119, 2006.
9. Ciabattoni A. and Terui K., ‘Modular cut-elimination: finding proofs or counterexamples’, *Proceedings of the 13-th International Conference of Logic for Programming AI and Reasoning (LPAR06)*, LNAI 4246, 135–149, 2006.
10. Church A., ‘A formulation of the simple theory of types’, *Journal of Symbolic Logic*, vol. 5, 56–68, 1940.
11. Gentzen, G., ‘Investigations into Logical Deduction’, in *The collected works of Gerhard Gentzen* (M.E. Szabo, ed.), 68–131, North Holland, Amsterdam , 1969.
12. Hähnle, R., ‘Commodious Axiomatization of Quantifiers in Many-valued Logic’, *Studia Logica*, vol. 61, 101–121, 1998.
13. Krynicki, M. and M.Mostowski, ‘Henkin Quantifiers’, *Quantifiers: logics, models and computation, M. Krynicki, M. Mostowski and L. Szcerba eds.*, vol. 1, 193–263, Kluwer Academic Publishers, 1995.
14. Miller, D. and E. Pimentel, ‘Using Linear logic to reason about sequent systems’, *Tableaux’02, LNAI*, 2–23, 2002.
15. Avron, A. and A. Zamansky, ‘Quantification in Non-deterministic Multi-valued Structures’, *Proceedings of the 35th IEEE International Symposium on Multiple-Valued Logics*, 296–301, 2005.
16. Zamansky, A. and A. Avron, ‘Canonical Gentzen-type calculi with (n,k)-ary quantifiers ’, *Proceedings of the Third International Joint Conference in Automated Reasoning (IJCAR06)*, 251 – 265, Furbach and Shankar eds, LNAI 4130, Springer, 2006.
17. Zamansky, A. and A. Avron, ‘Cut Elimination and Quantification in Canonical Systems’, *Studia Logica (special issue on Cut Elimination)*, vol. 82(1), 157–176, 2006.

Elementary Differential Calculus on Discrete and Hybrid Structures

Howard A. Blair[1], David W. Jakel[1], Robert J. Irwin[1], and Angel Rivera[2]

[1] Syracuse University, Syracuse NY 13244-4100 USA
{blair,dwjakel,rjirwin}@ecs.syr.edu
http://www.cis.syr.edu/~blair
[2] Utica College, Utica, NY 13502 USA
arivera@utica.edu

Abstract. We set up differential calculi in the Cartesian-closed category CONV of *convergence spaces*. The central idea is to uniformly define the 3-place relation __ **is a differential of** __ **at** __ for each pair of convergence spaces X, Y in the category, where the first and second arguments are elements of $\mathrm{Hom}(X, Y)$ and the third argument is an element of X, in such a way as to (1) obtain the chain rule, (2) have the relation be in agreement with standard definitions from real and complex analysis, and (3) depend only on the convergence structures native to the spaces X and Y. All topological spaces and all reflexive directed graphs (i.e. discrete structures) are included in CONV. Accordingly, ramified hybridizations of discrete and continuous spaces occur in CONV. Moreover, the convergence structure within each space local to each point, individually, can be discrete, continuous, or hybrid.

Keywords: Differential, convergence space, discrete structure, hybrid structure.

1 Introduction

With topology, continuity of functions generalizes from the contexts of classical analysis to a huge collection of structures, the topological spaces. The purpose here is to do the same for differentiability and also to allow for differentiation of such functions as, for example, functions between discrete structures (represented as reflexive directed graphs) as well as functions between discrete structures and topological spaces, particularly continua commonly occurring in elementary analysis.

Just as continuity itself neither presupposes any separation strength nor any notion of linearity, neither does differentiability. The familiar differential calculus on Euclidean spaces is of course intrinsically dependent on the vector space structure, but this is due to the choice of functions used to serve as differentials, and the consequent determination of the conditions under which functions are differentiable. What matters is the differentiability relation "differential g **is a differential of** f **at** x". Unless we demand of g that it satisfy some kind of linearity property, linearity does not intrinsically enter into the relation.

A word about derivatives: The derivative of a function at a point is a differential. For example, the derivative of $\lambda x \,.\, x^2$ at 1 is $\lambda x \,.\, 2x$, the linear function with slope 2. The derivative of a function f on a subset of the function's domain is another function that maps each point x of the subset to the derivative of f at x. The point is that derivatives are differential-valued. In the case of $\mathbf{E}^1$, the real numbers with the standard Euclidean topology, the space of linear functions, i.e. the space of differentials, is taken with a topology making it homeomorphic to $\mathbf{E}^1$. For situations where no such homeomorphism is available, we expect the codomain of a derivative of f to be different from the codomain of f. This is evident already with 2-dimensional vector spaces over the reals.

We will set up differential calculi in the Cartesian-closed category CONV of *convergence spaces*. The central idea is to uniformly define the 3-place relation

__ is a differential of __ at __

for each pair of convergence spaces X, Y in the category, where the first and second arguments are elements of $\mathrm{Hom}(X, Y)$ and the third argument is an element of X, in such a way as to (1) obtain the chain rule, (2) have the relation be in agreement with standard definitions in real and complex analysis, and (3) depend only on the convergence structure native to the spaces X and Y.

Plan of the papers: In section 2 we define convergence spaces and the notion of continuity of functions at a point and discuss some of relevant properties of the resulting category CONV. The representation of reflexive digraphs and topological spaces as convergence structures is discussed in section 3. Section 4 presents the algebraic ideas that constitute the extraction of *linear structure* from the symmetries of a convergence space's convergence structure. Section 5 gives the definition of a differential calculus involving homogeneous spaces and the definition of the 3-place differentiability relation. Section 5 includes the statement and proof of the chain rule and identifies the differential calculi associated with CONV. Section 6 presents examples of differential calculi. Section 7 concludes the paper by extending the ideas to differential calculi that include nonhomogeneous spaces.

It is important to note that the spaces and functions of interest are naturally organized into categories and to note the nature of the containments and embeddings that are involved. In particular, any method for constructing a differential calculus for mappings between arbitrary convergence spaces gives such a method for all reflexive digraphs and all topological spaces. The results of this paper should thus be seen as constituing a tool-kit for setting up mathematical structures that import techniques from continuous mathematics into discrete contexts.

2 Convergence Spaces, CONV and Prior Work

There is a beautiful paper including a brief but powerful tutorial on convergence spaces due to R. Heckmann [2003]. We present a few of the fundamental ideas necessary for our work.

A *filter* on a set X is a collection of subsets of X closed under finite intersection and reverse inclusion. $\mathcal{F}$ is a *proper* filter if the empty set is not a member of $\mathcal{F}$. Let $\Phi(X)$ denote the set of all filters on X. For a subset A of X, $\{ B \mid A \subseteq B \subseteq X \}$ is a member of $\Phi(X)$. We denote this filter by $[A]$. In the special case where A is a singleton $\{x\}$ we denote $[A]$ by $[x]$ and call this the *point filter* at x.

Definition 1. *[1964, 2003] A* convergence structure *on X is a relation $\downarrow$ (read as "converges to") between members of $\Phi(X)$ and members of X such that for each $x \in X$: (1) $[x]$ converges to x, and (2) the set of filters converging to x is closed under reverse inclusion. A pair $(X, \downarrow)$ consisting of a set X and a convergence structure $\downarrow$ on X is called a* convergence space.

A function $f : X \longrightarrow Y$ where X and Y are sets, induces functions $\hat{f} : 2^X \longrightarrow 2^Y$ and $\hat{\hat{f}} : \Phi(X) \longrightarrow \Phi(Y)$. $\hat{f}$ is defined by $\hat{f}(A) = \{f(a) \mid a \in A\}$, which we call the *$f$-image* of A. For $\mathcal{F} \in \Phi(X)$ note that the collection of all supersets of f-images of members of $\mathcal{F}$ forms a filter which we call $\hat{\hat{f}}(\mathcal{F})$. Hereafter we overload notation and drop the $\hat{}$ and $\hat{\hat{}}$ annotations.

When convenient, we will refer to a convergence space $(X, \downarrow)$ by its carrier, X.

Definition 2. *[1964, 2003] Let $f : X \longrightarrow Y$ where X and Y are convergence spaces, and let $x_0 \in X$. f is* continuous at x_0 *iff for each $\mathcal{F} \in \Phi(X)$, if $\mathcal{F} \downarrow x_0$ in X, then $f(\mathcal{F}) \downarrow f(x_0)$ in Y. f is* continuous *iff f is continuous at every point of X.*

Continuity can be characterized in terms of filter members, which play a role analogous to the role played by neighborhoods, as supersets of open sets, in topological spaces.

Proposition 1. *Let $f : X \longrightarrow Y$ where X and Y are convergence spaces, and let x_0 be a point of X. f is continuous at x_0 iff for every filter $\mathcal{F}$ converging to x_0 in X, there is a filter $\mathcal{G}$ converging to $f(x_0)$ in Y such that $(\forall V \in \mathcal{G})(\exists U \in \mathcal{F})[f(U) \subseteq V]$.*

Definition 3. *[1964] A* homeomorphism *between two convergence spaces is a continuous bijection whose inverse is continuous.*

The objects of the category of convergence spaces CONV are the convergence spaces. For convergence spaces X and Y, $\mathrm{HOM}(X, Y)$ is the set of continuous functions from X to Y.

The category CONV includes all topological spaces but enjoys several substantial advantages over the category TOP of topological spaces. Importantly for computation, CONV contains multiple representations of all reflexive directed graphs (finite and infinite). Among digraphs, continuity is the property of being *edge-preserving*, i.e. a digraph homomorphism. But, powerfully, and unlike TOP, CONV is a Cartesian-closed category. Several immediate consequences of Cartesian-closure and the relationship between TOP and CONV are: (1) convergence spaces preserve the notion of continuity on topological spaces;

(2) convergence spaces allow fine control over continuity, and in various circumstances allow for strengthening the conditions for continuity; (3) at one's option, there is a uniform way of regarding all spaces of continuous functions as convergence spaces, but other topological structures, (for example, a structure derived from a norm) are available, and (4) function composition and application are continuous.

Over time, a number of researchers have sought to generalize differentiability to spaces where the generalization is non-obvious. Some of the more serious and sophisticated results in this direction have employed one or another restriction of the notion of convergence space, often near to pre-topological spaces, or else stayed within TOP [1946, 1968, 1966, 1966, 1945, 1966, 1974, 1983, 1963, 1940]. These explorations assumed the existence of additional structure characterizing linearity. [1983] recognized the importance of Cartesian-closure for obtaining a robust chain-rule.

3 Reflexive Digraphs and Topological Spaces as Convergence Spaces

Definition 4. *Let x be a point of a convergence space X, and let U be a subset of X. U is said to be a* neighborhood *of x iff U belongs to every filter converging to x.*

Definition 5. *[1947] A convergence space $(X, \downarrow)$ is said to be a* pretopological space *if and only if $\downarrow$ is a* pretopology*, i.e. for each $x \in X$, the collection of all neighborhoods of x converges to x.*

Proposition 2. *Let $f : X \longrightarrow Y$ where X and Y are pretopological spaces, and let $x_0 \in X$. f is continuous at x_0 iff for every neighborhood V of $f(x_0)$, there is a neighborhood U of x_0 such that $f(U) \subseteq V$.*

It is evident that every topological space is a pretopological space (cf. [1947, 1940, 1955]), and that the convergence space notion of continuity and the topological space notion of continuity coincide for topological spaces. As indicated in the introduction, the spaces and functions of interest to us are naturally organized as categories. The main categories of interest in this paper are:

CONV	the category of convergence spaces and continuous functions
PreTOP	the category of pretopological spaces and continuous functions
TOP	the category of topological spaces and continuous functions
ReRe	the category of reflexive digraphs (i.e. directed graphs with a loop at each vertex) and edge-preserving functions
PostD	the full subcategory of CONV whose objects are the postdiscrete (see below) convergence spaces

TOP is a full subcategory of **PreTOP**, which, in turn, is a full subcategory of **CONV**. Both of these full inclusions are *reflective*, via induced pretopology and induced topology operations, respectively. **ReRe** is isomorphic to **PostD**, which, in turn, embeds into **PreTOP**.[1]

The reflection functor PreT from **CONV** to **PreTOP** can be realized by letting the carrier of $\mathrm{PreT}(X)$ be the carrier of X, and and letting a filter $\mathcal{F}$ converge to a point x in $\mathrm{PreT}(X)$ iff the collection of all neighborhoods of x in X is a subcollection of $\mathcal{F}$.

Similarly, the reflection functor T from **PreTOP** to **TOP** can be realized by letting the carrier of $\mathrm{T}(X)$ be the carrier of X, and defining the topology on $\mathrm{T}(X)$ as $\{U \subseteq X \mid U \text{ is a neighborhood in } X \text{ of each point of } U\}$.

ReRe can be embedded, in more than one way, as a full subcategory of **CONV**.

Definition 6. *A convergence space X will be said to be* postdiscrete *if and only if every convergent proper filter is a point filter.*

Proposition 3. *The postdiscrete pretopological spaces are precisely the discrete topological spaces.*

Definition 7. *Let (V, E) be a reflexive digraph. Induce a convergence structure on V by letting a proper filter $\mathcal{F}$ converge to a vertex x iff $\mathcal{F} = [y]$ for some vertex y with an edge in E from x to y.*

It is readily verified that if (V_1, E_1) and (V_2, E_2) are reflexive digraphs, then a function $f : V_1 \longrightarrow V_2$ is continuous (with respect to the induced convergence structures on V_1 and V_2) iff, for all edges (x, y) in E_1, the edge $(f(x), f(y))$ is present in E_2.

Proposition 4. *The construction in Definition 7 embeds **ReRe** as a full subcategory of **CONV**, namely the full subcategory whose objects are the postdiscrete spaces where this embedding is coreflective. The coreflection functor* ReR : ***CONV*** $\longrightarrow$ ***ReRe*** *can be obtained letting the vertices of* $\mathrm{ReR}(X)$ *be the members of X, and letting an ordered pair (x, y) be an edge of* $\mathrm{ReR}(X)$ *iff $[y] \downarrow x$ in X.*

Alternatively, **ReRe** can be embedded as a full subcategory of **PreTOP**, and thence as a full subcategory of **CONV** [2001, 2003], by letting a filter $\mathcal{F}$ converge to a vertex x iff $\{\, y \mid (x, y) \in E \,\}$ is a member of $\mathcal{F}$.

In general, this embedding of **ReRe** into **PreTOP** imposes a weaker convergence structure on reflexive digraphs than the embedding in Definition 7.

Proposition 5. *The embedding of **ReRe** into **PreTOP** [2001, 2003] is the composite of the embedding in Definition 7 of **ReRe** into **CONV** with the reflection functor from **CONV** to **PreTOP**, and embeds **ReRe** as a full, coreflective subcategory of **PreTOP**, and thence as a full, coreflective subcategory of **CONV**. The coreflection functor from **PreTOP** to **ReRe** (and the coreflection functor from **CONV** to **ReRe** via **PreTOP**) can be obtained in precisely the same way as in Proposition 4.*

[1] The embedding is the restriction of the induced pretopology reflection from **CONV** to **PostD**.

The reflexive digraphs whose induced pretopologies are topological are precisely those in which the underlying binary relation is transitive as well as reflexive. [2001, 2003] Unlike **TOP** and **PreTOP**, **CONV** is a Cartesian closed category ([1971, 1975, 1990, 2001, 1965]):

Definition 8. *[1965]*
Let X and Y be convergence spaces. The function space Y^X *is the set of all continuous functions from X to Y, equipped with the convergence structure $\downarrow$ defined as follows: For each $\mathcal{H} \in \Phi(Y^X)$ and each $f_0 \in Y^X$, let $\mathcal{H} \downarrow f_0$ if, and only if, for each $x_0 \in X$ and each $\mathcal{F} \downarrow x_0$, $\{\,\{\,f(x) \mid f \in H, x \in F\,\} \mid H \in \mathcal{H}, F \in \mathcal{F}\,\}$ is a base for a filter which converges to $f(x_0)$ in Y.*

4 Translation Groups and Homogeneous Convergence Spaces

Definition 9. *An* automorphism *of a convergence space X is a homeomorphism $f : X \longrightarrow X$.*

Definition 10. *A* translation group *on a convergence space X is a group T of automorphisms of X such that, for each pair of points p and q of X, there is at most one member of T which maps p to q. In general, we will denote this unique member of T (if it exists) by $(q - p)$.*

Notation: *The group operation of a translation group T on a convergence space X will be written additively, whether or not T is Abelian. Furthermore, for all $\tau \in T$ and all $x \in X$, we will write $\tau(x)$ as $x + \tau$.*

In this notation, the requirement that the translation $(q-p)$ (if it exists) maps p to q becomes the familiar requirement that if $(q-p)$ exists, then $p+(q-p) = q$.

A full translation group *on a convergence space X is a translation group on X which contains a translation $(q - p)$ for each pair of points p and q.*

Proposition 6

i. *Every convergence space X can be embedded as a subspace of a convergence space HX which has a full translation group.*
ii. *X and HX have the same cardinality if and only if the cardinality of X is either zero or infinite.*
iii. *The embedding of X into HX is onto HX if and only if X is empty.*
iv. *If X and Y are arbitrary convergence spaces then every continuous function $f : X \longrightarrow Y$ can be be extended to a continuous function $Hf : HX \longrightarrow HY$.*
v. *If f is a homeomorphism, then so is Hf.*

An immediate consequence of (ii) in Proposition 6 is that, if X is a non-empty finite space with (or without) a full translation group, then HX cannot be homomeomorphic to X. It should also be noted that, given a particular f, the continuous extention Hf need not be unique.

Definition 11. *A convergence space X is* homogeneous *iff for each pair of points x_1 and x_2 of X, there is an automorphism of X which maps x_1 to x_2*

Observation 1. *A convergence space which has a full translation group must be homogeneous. Furthermore, a full translation group on a nonempty convergence space X must have the same cardinality as X.*

5 Differential Calculi

Definition 12. *A* differential calculus *is a category $\mathcal{D}$ in which*

i. *every object of $\mathcal{D}$ is a triple $\mathcal{X} = (X, 0, T)$ such that X is a convergence space, 0 is a point of X (called the* origin *of $\mathcal{X}$), and T is a full translation group on X.*
ii. *every arrow in $\mathcal{D}$ from an object $(X, 0_X, T_X)$ to an object $(Y, 0_Y, T_Y)$ is a continuous function from X to Y which maps 0_X to 0_Y*
iii. *composition of arrows in $\mathcal{D}$ is function composition.*
iv. *for every object $\mathcal{X} = (X, 0_X, T_X)$, the identity function on X is an arrow in $\mathcal{D}$ from $\mathcal{X}$ to $\mathcal{X}$*
v. *for each pair of objects $\mathcal{X} = (X, 0_X, T_X)$ and $\mathcal{Y} = (Y, 0_Y, T_Y)$, the constant function mapping every point of X to 0_Y is an arrow in $\mathcal{D}$ from $\mathcal{X}$ to $\mathcal{Y}$.*

In view of Proposition 6, the requirement that each object have a full translation group is not unduly restrictive. We are now in a position to define the differentiability relation. Let $a \in A \subseteq X$ and let $B \subseteq Y$, where $\mathcal{X} = (X, 0_X, T_X)$ and $\mathcal{Y} = (Y, 0_Y, T_Y)$ are objects of a differential calculus $\mathcal{D}$. Let $f : A \longrightarrow B$ be an arbitrary function.

Let $L \in \mathcal{D}(\mathcal{X}, \mathcal{Y})$, where $\mathcal{D}(\mathcal{X}, \mathcal{Y})$ is the set of all arrows in $\mathcal{D}$ from $\mathcal{X}$ to $\mathcal{Y}$, equipped with the subspace convergence structure inherited from the function space Y^X in **CONV**.

Definition 13. *L is a* differential *of f at a iff*
for every $\mathcal{F} \downarrow a$ in A, there is some $\mathcal{H} \downarrow L$ in $\mathcal{D}(\mathcal{X}, \mathcal{Y})$ such that

i. *$\mathcal{H} \subseteq [L]$, and*
ii. *for every $H \in \mathcal{H}$, there is some $F \in \mathcal{F}$ such that*
for every point $x \in F$, there is at least one function $t \in H$ such that

$$t(x - a) = f(x) - f(a))$$

In Definition 13, $(f(a) - 0_Y) \circ t \circ (0_X - a)$ is called an *extrapolant* of f through $(a, f(a))$ and $(x, f(x))$.

Definition 14. *A function from A to B is* differentiable *(respectively,* uniquely differentiable*) at a point a iff it has* at least *one (respectively,* precisely *one) differential at a. A function from A to B is* differentiable *(respectively,* uniquely differentiable*) iff it is differentiable (respectively, uniquely differentiable) at each point of A.*

We next obtain the chain rule. As we indicated in the introduction, the chain rule plays a central role in differential calculi. In elementary real and complex analysis, for example, the product rule follows from the chain rule after obtaining the differential of the multiplication operation.

Example 1. Expressed in terms of differentials, the product rule for real-valued functions of a real variable reduces to matrix multiplication (i.e. composition of linear functions).

$$\begin{aligned} D_x(\text{mult} \circ (f,g)) &= D_{(f,g)(x)}\text{mult} \circ D_x(f,g) \\ &= D_{(f(x),g(x))}\text{mult} \circ (D_x f, D_x g) \\ &= [g(x) f(x)] \begin{bmatrix} D_x f \\ D_x g \end{bmatrix} \\ &= g(x) D_x f + f(x) D_x g \end{aligned}$$

Returning to our more general setting, let $a \in A \subseteq X$, let $B \subseteq Y$, and let $C \subseteq Z$, where $\mathcal{X} = (X, 0_X, T_X)$, $\mathcal{Y} = (Y, 0_Y, T_Y)$, and $\mathcal{Z} = (Z, 0_Z, T_Z)$ are objects of a differential calculus $\mathcal{D}$. Let $f : A \longrightarrow B$ and $g : B \longrightarrow C$ be arbitrary functions. Let $K : \mathcal{X} \longrightarrow \mathcal{Y}$ and $L : \mathcal{Y} \longrightarrow \mathcal{Z}$ be arrows of $\mathcal{D}$.

Theorem 2. ***(Chain Rule)*** *Suppose that f is continuous at a. Also suppose that K is a differential of f at a, and L is a differential of g at $f(a)$. Then $L \circ K$ is a differential of $g \circ f$ at a.*

Proof: Let $\mathcal{F}$ be a filter converging to a in X. Since K is a differential of f at a, there is some $\mathcal{G} \downarrow K$ in $\mathcal{D}(\mathcal{X}, \mathcal{Y})$ such that $\mathcal{G} \subseteq [K]$ and, for every $G \in \mathcal{G}$, there is some $F_{1,G} \in \mathcal{F}$ such that for each point $x \in F_{1,G}$ there is some function $s_{G,x} \in G$ such that

$$s_{G,x}(x - a) = f(x) - f(a) \tag{1}$$

On the other hand, since f is continuous at a, we have $f(\mathcal{F}) \downarrow f(a)$ in B. Since L is a differential of g at $f(a)$, there is some filter $\mathcal{H} \downarrow L$ in $\mathcal{D}(\mathcal{Y}, \mathcal{Z})$ such that $\mathcal{H} \subseteq [L]$ and, for every $H \in \mathcal{H}$, there is some $N_H \in f(\mathcal{F})$ such that for each point $y \in N_H$, there is some function $t_{H,y} \in H$ such that

$$t_{H,y}(y - f(a)) = g(f(x)) - g(f(a)) \tag{2}$$

Consider such a set N_H. By definition, $N_H \in f(\mathcal{F})$, i.e. there is some $F_{2,H} \in \mathcal{F}$ such that

$$f(F_{2,H}) \subseteq N_H$$

By (2), for each point $x \in F_{2,H}$, we have

$$t_{H,f(x)}(f(x) + (0_Y - f(a))) = g(f(x)) + (0_Z - g(f(a))) \tag{3}$$

Next, note that $\{ \{ h_2 \circ h_1 \mid h_1 \in G, h_2 \in H \} \mid G \in \mathcal{G}, H \in \mathcal{H} \}$ is a basis for a filter $\mathcal{J}$ on $\mathcal{D}(\mathcal{X}, \mathcal{Z})$, and that $\mathcal{J} \subseteq [L \circ K]$.

By joint continuity of composition, $\mathcal{J} \downarrow L \circ L$ in $\mathcal{D}(\mathcal{X}, \mathcal{Z})$.

Let J be an arbitrary member of $\mathcal{J}$. There exist $G \in \mathcal{G}$ and $H \in \mathcal{H}$ such that

$$\{ h_2 \circ h_1 \mid h_1 \in G, h_2 \in H \} \subseteq J$$

Let $F = F_{1,G} \cap F_{2,H}$. Then $F \in \mathcal{F}$. For each point $x \in F$, we have $s_{G,x} \in G$ and $t_{H,f(x)} \in H$, and therefore $t_{H,f(x)} \circ s_{G,x} \in J$. Furthermore, by (1) and (3),

$$t_{H,f(x)}(s_{G,x}(x+(0_X-a))) = t_{H,f(x)}(f(x)+(0_Y-f(a))) = g(f(x))+(0_Z-g(f(a))) \tag{4}$$

Thus, $(g(f(a))-0_Z) \circ t_{H,f(x)} \circ s_{G,x} \circ (0_X - a)$ is the required extrapolant of $g \circ f$ through $(a, g(f(a))$ and $(x, g(f(x))$.

Since the selection of x is arbitrary (once G and H have been chosen), $L \circ K$ is indeed a differential of $g \circ f$ at a.

6 Examples

Throughout, let $\mathbf{R}$ be the real line (equipped with its Euclidean topology), and let $\mathbf{N}$ be the set of all natural numbers.

Example 2. **The classical differential calculus of real variables:** The objects of this differential calculus are the spaces $\mathbf{R}^n$ ($n \in \mathbf{N}$), equipped with their respective Euclidean topologies, with their respective zero vectors as origins, and with their usual translation groups. The arrows of this calculus are the $\mathbf{R}$-linear functions. In this calculus, differentiability and unique differentiability are equivalent, and a function f has a differential at a point p iff f is differentiable (in the usual sense) at p.

Example 3. **The directional calculus of real variables:** Again, the objects are the sets $\mathbf{R}^n$ ($n \in \mathbf{N}$), with their respective zero vectors as origins, and with their usual translation groups. Again, $\mathbf{R}$ is equipped with its Euclidean topology.

However, for $n > 1$, the convergence structure imposed on $\mathbf{R}^n$ is stronger than the (Euclidean) product structure. In the directional calculus of real variables, a filter $\mathcal{F}$ will be said to converge to a point p iff there is some unit vector q such that $\{ p + \alpha q \mid \alpha \in \mathbf{R}, |\alpha| < \epsilon \} \in \mathcal{F}$ for every real number $\epsilon > 0$.

The arrows of this calculus are the $\mathbf{R}$-homogeneous functions of degree one.

In this calculus, differentiabilty and unique differentiabilty are equivalent, but a function f has a differential at a point p iff f has directional derivatives in all directions at p.

Example 4. **Boolean differential calculus** (cf. Boolean derivatives [1954, 1959, 1990]): Let $\mathbf{B}$ be a complete digraph with two vertices, F and T, and equipped with the induced postdiscrete convergence structure (cf. Definition 7).

Both of the point filters on $\mathbf{B}$ converge *both* to F and to T. (This convergence structure differs from the induced pretopological structure (namely, the indiscrete structure) in that the filter $\{F, T\}$ does *not* converge in $\mathbf{B}$, but converges to both points in the indiscrete structure.)

In this calculus, the carriers of objects are the spaces $\mathbf{B}^n$ ($n \in \mathbf{N}$). Each carrier is equipped with the product convergence structure (which coincides with the postdiscrete convergence structure induced by the complete digraph on the carrier).

For each n, the bit vector $(F, F, \ldots, F)$ of length n is taken as the origin of $\mathbf{B}^n$.

The group generated by the flips $h_1, h_2, \ldots, h_n$ is taken as the translation group of $\mathbf{B}^n$, where (as one would expect) $h_k(\boldsymbol{b})$ is obtained from bit vector $\boldsymbol{b}$ by changing the k^{th} bit of $\boldsymbol{b}$ (and leaving every other bit unchanged).

For each m and n, the arrows from $\mathbf{B}^m$ to $\mathbf{B}^n$ are defined to be *all* origin-preserving functions from $\mathbf{B}^m$ to $\mathbf{B}^n$. (This is compatible with our definition of a differential calculus, since every function between complete digraphs preserves edges.)

In particular, there are precisely two arrows between $\mathbf{B}$ and itself, namely, the identity function, and the constant function which returns F.

In the Boolean differential calculus, every function between $\mathbf{B}$ and itself is uniquely differentiable.

Example 5. **Differentiating a function from 3R to $\mathbf{K}_3^-$:** $\mathbf{K}_3^-$ is the complete directed graph on 3 vertices, but with one edge from one vertex to another removed. It is universal for the all pretopological convergence spaces in the sense that every pretopological space embed in some Cartesian power of it, (Bourdaud [1976]). The space **3R** is our designation for the set of real numbers equipped with Euclidean filter structure at each real number r, and in addition at r, all filters containing the filters generated by the open intervals whose right end point is r, and all filters containing the filters generated by the open intervals whose left end point is r. Take all functions from **3R** to $\mathbf{K}_3^-$ that are piecewise constant at 0 for differentials. Then $g : \mathbf{3R} \longrightarrow \mathbf{K}_3^-$ is a differential of $f : \mathbf{3R} \longrightarrow \mathbf{K}_3^-$ at x_0 iff f is constant on an open interval whose right end point is r and constant on an open interval whose left end point is r.

7 Differential Calculi with Nonhomogeneous Objects

In the preceding development, convergence spaces without full translation groups are "second class citizens" in the sense that they cannot be the carriers of objects of a differential calculus. A somewhat more general (and slightly more complicated) concept of "differential calculus" permits all convergence spaces to be carriers of objects.

Observation 3. *Let T be a translation group on a convergence space X. For each point x of X, let $[x]_T$ be the T-*orbit *of x, i.e. $[x]_T = \{\, x + \tau \mid \tau \in T \,\}$.*

If X is nonempty, then the set of all T orbits partitions X into homogeneous subspaces. For each T-orbit $[x]_T$, the restrictions of the members of T to $[x]_T$ form a full translation group $T_{[x]}$ on $[x]_T$.

Each $T_{[x]}$ is a quotient group of T.

Definition 15. *A* system of origins *for a convergence space X with respect to a translation group T is a set of representatives of the T-orbits, i.e., a subset $\mathcal{O}$ of X containing precisely one member of each T-orbit.*

For each point x of X, let 0_x be the unique member of $\mathcal{O}$ belonging to the same T-orbit as x.

Definition 16. *Let $f : X \longrightarrow Y$ be a function between convergence spaces. Let $\mathcal{O}_X$ ($\mathcal{O}_Y$, respectively) be a system of origins for X with respect to a translation group S (for Y with respect to a translation group T, respectively).*

i. f will be said to respect orbits *iff, for each pair of points p and q of X, if p and q lie in the same S-orbit, then $f(p)$ and $f(q)$ lie in the same T-orbit.*
ii. f will be said to be preserve origins *iff $f(\mathcal{O}_X) \subseteq \mathcal{O}_Y$.*

Definition 17. *A* generalized differential calculus *is a category $\mathcal{D}$ in which*

i. every object of $\mathcal{D}$ is a triple $\mathcal{X} = (X, T, \mathcal{O})$ such that X is a convergence space, T is a translation group on X, and $\mathcal{O}$ is a system of origins for X with respect to T.
ii. every arrow in $\mathcal{D}$ from an object $(X, S, \mathcal{O}_X)$ to an object $(Y, T, \mathcal{O}_Y)$ is a continuous, orbit-respecting, origin-preserving function from X to Y.
iii. composition of arrows in $\mathcal{D}$ is function composition.
iv. for every object $\mathcal{X} = (X, T, \mathcal{O})$, the identity function on X is an arrow in $\mathcal{D}$ from $\mathcal{X}$ to $\mathcal{X}$
v. for each pair of objects $\mathcal{X} = (X, S, \mathcal{O}_X)$ and $\mathcal{Y} = (Y, T, \mathcal{O}_Y)$ and each ζ in $\mathcal{O}_Y$, the constant function mapping every point of X to ζ is an arrow in $\mathcal{D}$ from $\mathcal{X}$ to $\mathcal{Y}$

A differential calculus (in the sense of Definition 12) is essentially the same notion as a generalized differential calculus in which the translation group of every object is a full translation group.

At the opposite extreme, there are generalized differential calculii in which the translation group of every object is trivial (and hence all orbits are singletons).

Example 6. **CONV as a generalized differential calculus**
The objects of the *trivial generalized differential calculus* are *all* convergence spaces, equipped with trivial translation groups. The arrows from an object X to an object Y are *all* continuous functions from X to Y. (Since all orbits are singletons, *every* function is orbit-respecting and origin-preserving.)

Let $a \in X$ and let $b \in Y$, where $\mathcal{X} = (X, S, \mathcal{O}_X)$ and $\mathcal{Y} = (Y, T, \mathcal{O}_Y)$ are objects of a generalized differential calculus $\mathcal{D}$. Let $f : A \longrightarrow B$ be an arbitrary function.

Let $L \in \mathcal{D}(\mathcal{X}, \mathcal{Y})$, where, again, $\mathcal{D}(\mathcal{X}, \mathcal{Y})$ is the set of all arrows in $\mathcal{D}$ from $\mathcal{X}$ to $\mathcal{Y}$, equipped with the subspace convergence structure inherited from the function space Y^X in **CONV**.

Definition 18. *L is a* differential *of f at a iff for every $\mathcal{F} \downarrow a$ in X, there is some $\mathcal{H} \downarrow L$ in $\mathcal{D}(\mathcal{X}, \mathcal{Y})$ such that*

i. *$\mathcal{H} \subseteq [\{L\}]$, and*
ii. *for every $H \in \mathcal{H}$, there is some $F \in \mathcal{F}$ such that for every point $x \in F$, there is at least one function $t \in H$ such that*

$$t(x + (0_x - a)) = f(x) + (0_{f(x)} - f(a))$$

Differentiability and unique differentiability are defined exactly as before.

Example 7. **The classical affine differential calculus of real variables** The objects of this generalized differential calculus are the Euclidean spaces, equipped with trivial translation groups. The arrows from $\mathbf{R}^m$ to $\mathbf{R}^n$ are all affine functions from $\mathbf{R}^m$ to $\mathbf{R}^n$

As in the classical linear differential calculus, a function f has a differential at a point p iff f is differentiable (in the usual sense) at p.

Let $E_p f$ be the differential of f at p in the classical affine differential calculus of real variables. That is, $E_p(f)$ is the affine function which best approximates f in arbitrarily small neighhborhoods of p.

Then the differential of f at p in the classical linear differential calculus is the unique linear function which can be obtained from $E_p f$ by composing it on both sides with translations (in the usual sense), i.e. the function which maps each point q to $f(p) + (E_p f)(q - p)$.

Next, we obtain the chain rule for generalized differential calculi. Let $a \in X$, where $\mathcal{X} = (X, R, \mathcal{O}_X)$, $\mathcal{Y} = (Y, S, \mathcal{O}_Y)$, and $\mathcal{Z} = (Z, T, \mathcal{O}_Z)$ are objects of a generalized differential calculus $\mathcal{D}$. Let $f : X \longrightarrow Y$ and $g : Y \longrightarrow Z$ be arbitrary functions. Let $K : \mathcal{X} \longrightarrow \mathcal{Y}$ and $L : \mathcal{Y} \longrightarrow \mathcal{Z}$ be arrows of $\mathcal{D}$.

Theorem 4. ***(Chain Rule)***
Suppose that f is continuous at a. Also suppose that K is a differential of f at a, and L is a differential of g at $f(a)$.

Then $L \circ K$ is a differential of $g \circ f$ at a.

Proof: Similar to the proof of Theorem 2, but with the obvious modifications.

References

[1990] J. Adámek, H. Herrlich, and G. E. Strecker, *Abstract and Concrete Categories*, Wiley Interscience, 1990.

[1959] S. B. Akers Jr., On a theory of Boolean functions, *J. SIAM* **7**, no. 4 (1959), pp. 487-498.

[1946] R. F. Arens, A Topology for Spaces of Transformations. Annals of Mathematics (2) 47 (1946), 480-495.

[1975] M. A. Arbib and E. Manes, *Arrows, Structures, and Functors: The categorical imperative*, Academic Press, 1975.

[1968] W. I. Averbukh and O. G. Smolyanov, The various definitions of the derivative in linear topological spaces, *Russian Math. Surveys* **23** (1968), no. 4, pp. 67-113.

[1966] E. Binz, Ein Differenzierbarkeitsbegriff limitieren Vektorraäume, *Comment. Math. Helv.* **41** (1966), pp. 137-156.

[1966] E. Binz and E. Keller, Functionenräume in der Kategorie der Limesräume, *Ann. Acad. Sci. Fenn.*, Ser. A.I., **1966**, pp. 1-21.

[1940] N. Bourbaki, *Topologie Générale*, Actualités Sci. Ind. **858** (1940), **916** (1942), **1029** (1947), **1045** (1948), **1084** (1949).

[1976] Bourdaud, G., Some cartesian closed categories of convergence spaces, in: Categorical Topology (Proc. Conf. Mannheim 1975), Lecture Notes in Mathematics 540 (1976), pp. 93108.

[1947] C. Choquet, Convergences, *Ann. Univ. Grenoble* **23** (1947), pp. 55-112.

[1945] R. H. Fox, On topologies for function spaces, *Bull. Amer. Math. Soc.* **51** (1945), pp. 429-432.

[1966] A. Frölicher and W. Bucher, *Calculus in Vector Spaces without Norm*, Lecture Notes in Math. **30**, Springer-Verlag, 1966.

[2003] R. Heckmann, A non-topological view of dcpo's as convergence spaces, *Theoretical Computer Science* **305** (2003), pp. 159 - 186.

[1965] M. Katětov, On continuity structures and spaces of mappings, *Comm. Math. Univ. Carol.* **6**, no. 2 (1965), pp. 257-279.

[1974] E. Keller, *Differential Calculus in Locally Convex Spaces*, Lecture Notes in Math. **417**, Springer-Verlag, 1974.

[1955] J. L. Kelley, *General Topology*, Van Nostrand Reinhold, 1955.

[1964] D. C. Kent, Convergence functions and their related topologies, *Fund. Math.* **54** (1964), pp. 125-133.

[1983] A. Kriegl, Eine kartesische abgeschlossene Kategorie glatter Abbildungen Zwischen beleibigen lokalkonvexen Vektoräumen, *Monatsh. Math.* **95** (1983), pp. 287-309.

[1971] S. MacLane, *Categories for the Working Mathematician*, Graduate Texts in Mathematics **5**, Springer Verlag (1971).

[1963] G. Marinescu, *Espaces Vectoriels Pseudo Topologique et le Théorie de Distributions*, Deutcshe Verlag d. Wiss., 1963.

[1940] A. D. Michal, Differential calculus in linear topological spaces, *Proc. Nat. Acad. Sci.* **24** (1938), no. 8, pp. 340-342.

[1954] I. S. Reed, A class of multiple-error-correcting codes and the decoding scheme. *IRE Trans. Inform. Theory* **IT-4**, no. 9 (1954), pp. 38-49.

[2003] C. M. Reidys and P. F. Stadler, Combinatorial landscapes,
http://www.santafe.edu/sfi/publications/Working−Papers/01−03−014.pdf
http://www.santafe.edu/sfi/publications/Working−Papers/01−03−014.ps

[2001] L. Schröder, Categories: a free tour, in *Categorical Perspectives* (J. Kozlowski and A. Melton, ed.), Birkhäuser, 2001, pp. 1-27.

[2001] B. M. R. Stadler, P. F. Stadler, G. P. Wagner, and W. Fontana, The topology of the possible: Formal spaces underlying patterns of evolutionary change, *Journal of Theoretical Biology* **213** (2001) no. 2, pp. 241-274.

[1990] G. Y. Vichniac, Boolean derivatives on cellular automata, in *Cellular automata: theory and experiment*, (H. Gutowitz, ed.), MIT Press, 1991, (*Physica D* **45** (1990) no. 1-3, pp. 63-74).

Weighted Distributed Systems and Their Logics

Benedikt Bollig[1] and Ingmar Meinecke[2]

[1] LSV, CNRS UMR 8643 & ENS de Cachan
61 Av. du Président Wilson, F-94235 Cachan Cedex, France
bollig@lsv.ens-cachan.fr
[2] Institut für Informatik, Universität Leipzig
Johannisgasse 26, D-04103 Leipzig, Germany
meinecke@informatik.uni-leipzig.de

Abstract. We provide a model of weighted distributed systems and give a logical characterization thereof. Distributed systems are represented as weighted asynchronous cellular automata. Running over directed acyclic graphs, Mazurkiewicz traces, or (lossy) message sequence charts, they allow for modeling several communication paradigms in a unifying framework, among them probabilistic shared-variable and probabilistic lossy-channel systems. We show that any such system can be described by a weighted existential MSO formula and, vice versa, any formula gives rise to a weighted asynchronous cellular automaton.

1 Introduction

Classical automata theory has become an indispensable tool in many modern areas of computer science, supporting, for example, programming languages and specification and verification techniques. In some applications, automata need to cope with quantitative phenomena. Then, taking a transition in an automaton is accompanied by measuring its cost or weight. For example, a system might provide a counter tracking the number of occurrences of a given pattern; or its behavior might depend on probability laws so that the outcome of a transition is generally uncertain and depends on a probability distribution. Actually, automata with weights enjoy manifold applications in numerous areas such as speech recognition [18], probabilistic systems [12,1], and image compression [4].

Formally, the behavior of a weighted automaton is no longer characterized by the pure existence of an accepting run. Rather, a weighted automaton comes up with a *formal power series* assigning to any possible execution sequence a value from a semiring. More precisely, the values collected along an automaton execution are multiplied, whereas nondeterminism is resolved by summation therewith generalizing the two operations of the two-valued Boolean algebra, cf. [13].

For a long time, the correspondence of automata and logic has been a captivating research direction in computer science. The probably most famous result goes back to Büchi and Elgot, who discovered a precise correspondence between finite automata and the logical formalism of monadic second-order (MSO) formulas [3,11]. In particular, any system description formalized in the MSO language comes up with an implementation in terms of a finite automaton. Concerning weighted automata, most results establish Kleene-like theorems stating that a formal power series is described by a weighted automaton iff it is rational [21,6,15]. A logical characterization of weighted automata has been achieved only recently: Droste and Gastin opened a new research direction by providing a weighted MSO logic to define formal power series over words [7]. Their achievements have been extended, among others, to automata on infinite words [9], trees [10], pictures [16], and traces [17].

In this paper, we deal with a model for weighted distributed systems that unifies many communication paradigms such as shared-memory systems, (lossy) channel systems, etc. It is constituted by asynchronous cellular automata (ACAs) [8] running on directed acyclic graphs (dags) without auto-concurrency. Unlike finite automata, which process their input words in a sequential fashion, ACAs are appropriate to concurrent executions. Accordingly, the assignment of weights does not depend on the order in which independent events are executed. ACAs have already been equipped with weights by Kuske to recognize formal power series over traces [15]. Generalizing results by Ochmański [19] and Droste and Gastin [6], he showed that a series is regular iff it is recognized by some weighted ACA. Actually, we provide an even more general model subsuming Kuske's automata. Running over dags rather than traces, our weighted ACA can cope with many common domains for concurrency, not only traces but also message sequence charts which play a prominent role in telecommunication. As we will discuss in the course of this paper, the latter domain allows an embedding of probabilistic lossy-channel systems. Our main result states that weighted ACAs recognize precisely the formal power series that are definable in an existential fragment of a weighted MSO logic over dags. This result cannot be obtained by a translation of the word setting as it was done for traces [17]. On the other hand, a lot of technical difficulties arise in our setting compared to this of words. Especially, we have to prove an unambiguity result for first-order definable languages before establishing the main theorem. For words such an unambiguity result is for free since deterministic devices suffice to recognize all regular languages. Moreover, the construction of a weighted formula from a given weighted asynchronous cellular automaton is much more tricky than for words.

The paper is structured as follows: in Section 2, we introduce our notion of a dag over a distributed alphabet. Hereby, a distributed alphabet constitutes the system architecture by assigning to any process its supply of actions. Section 3 introduces ACAs first in their classical, then in their weighted form. The behavior of a weighted ACA will be described in terms of a formal power series over (a subset of) the class of dags. Having introduced weighted MSO logic over dags in Section 4, Sections 5 and 6 derive our main result, the precise correspondence between weighted ACAs and the existential fragment of weighted MSO logic.

2 Dags over Distributed Alphabets

We fix a nonempty finite set Ag of *agents*, a *distributed alphabet* $\widetilde{\Sigma}$, which is a tuple $(\Sigma_i)_{i \in Ag}$ of (not necessarily disjoint) alphabets Σ_i, and an alphabet C. Elements from Σ_i are understood to be *actions* that are performed by agent i. Let $\Sigma = \bigcup_{i \in Ag} \Sigma_i$ denote the set of all the actions. The actions will label the nodes of a graph, which we will later refer to as *events*. Elements from C label edges of a graph to provide a kind of control information. For example, they might reflect the type of a message represented by an edge between communicating events. A (directed) *graph* over (Σ, C) is a structure $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ where V is its finite set of *nodes*, $\lhd_\ell \subseteq V \times V$ are disjoint binary relations on V, and $\lambda : V \to \Sigma$ is the *labeling function*. We call $\lhd := \bigcup_{\ell \in C} \lhd_\ell$ the *edge relation* and set $\leq = \lhd^*$ and $< = \lhd^+$. For $u, v \in V$, we define the *cover relation* $u \lessdot v$ of $\leq$ by $u < v$ and, for any $w \in V$, $u < w \leq v$ implies $w = v$. A *directed acyclic graph* (dag) over (Σ, C) is a graph $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ over (Σ, C) such that $\lhd$ is irreflexive and $\leq$ is a partial order. The set of all those dags is denoted by $\mathbb{DAG}(\Sigma, C)$. For $a \in \Sigma$, we put $loc(a) := \{i \in Ag \mid a \in \Sigma_i\}$. Then, a and b are *independent*, writing $a\ I_{\widetilde{\Sigma}}\ b$, if $loc(a) \cap loc(b) = \emptyset$. Otherwise, we say a and b are *dependent*, writing $a\ D_{\widetilde{\Sigma}}\ b$.

We now introduce the models representing the behavior of a system of communicating agents. In doing so, we combine and extend the models from [8,14,2].

Definition 2.1. *A* $(\widetilde{\Sigma}, C)$-dag *is a dag* $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda) \in \mathbb{DAG}(\Sigma, C)$ *where*

- *for any* $i \in Ag$, $\lambda^{-1}(\Sigma_i)$ *is totally ordered by* $\leq$ *and*
- *for any* $\ell \in C$ *and* $(u, v), (u', v') \in \lhd_\ell$ *with* $\lambda(u) D_{\widetilde{\Sigma}} \lambda(u')$ *and* $\lambda(v) D_{\widetilde{\Sigma}} \lambda(v')$, *we have* $u \leq u'$ *iff* $v \leq v'$.

The set of all $(\widetilde{\Sigma}, C)$*-dags is denoted by* $\mathbb{DAG}(\widetilde{\Sigma}, C)$.

The first condition reflects that a single agent is considered to operate sequentially. Especially, there is no auto-concurrency. The second condition ensures a FIFO architecture of communicating systems. Messages (u, v) and (u', v') of the same type between the same agents are received in the same order as they have been sent. Because of the FIFO-architecture and the absence of auto-concurrency, we conclude that, in a $(\widetilde{\Sigma}, C)$-dag $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$, for any $u \in V$, $\ell \in C$, and $a \in \Sigma$, there is at most one vertex $v \in V$ such that both $u \lhd_\ell v$ (or $v \lhd_\ell u$) and $\lambda(v) = a$.[1] If C is a singleton, we actually deal with structures $(V, \lhd, \lambda)$ and we speak of $\widetilde{\Sigma}$-dags.

The automaton model as introduced in the next section monitors for every node $u \in V$ of a $(\widetilde{\Sigma}, C)$-dag $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ the direct neighborhood of u. Therefore, we introduce the following abbreviations: For $u \in V$, we denote by $\mathrm{Read}(u) := \{(a, \ell) \in \Sigma \times C \mid \exists v \in V : v \lhd_\ell u \wedge \lambda(v) = a\}$ the *read domain* of u and, given $(a, \ell) \in \mathrm{Read}(u)$, let (a, ℓ)-$\mathrm{pred}(u)$ be the unique vertex v such that both $v \lhd_\ell u$ and $\lambda(v) = a$. Similarly, let $\mathrm{Write}(u) := \{(a, \ell) \in \Sigma \times C \mid \exists v \in V : u \lhd_\ell v \wedge \lambda(v) = a\}$ be the *write domain* of u and, for $(a, \ell) \in \mathrm{Write}(u)$, (a, ℓ)-$\mathrm{succ}(u)$ denote the unique vertex v such that both $u \lhd_\ell v$ and $\lambda(v) = a$. For $i \in Ag$ and $V_i = \{u \in V \mid \lambda(u) \in \Sigma_i\}$, sequential progress of an agent $i \in Ag$ is reflected by $\lhd_i := \lhd \cap (V_i \times V_i)$ and the total order $\leq_i := \leq \cap (V_i \times V_i)$ (do not mistake relation $\lhd_i$ of agent i for edge relation $\lhd_\ell$ for $\ell \in C$). For $u \in V$ and $i \in Ag$, u is Σ_i*-maximal* if $u \in V_i$ and there is no $v \in V_i$ such that $u < v$. Obviously, there is at most one Σ_i-maximal vertex.

Dags over distributed alphabets subsume popular domains of concurrency:

Example 2.1 (Mazurkiewicz Traces [5]). We consider distributed systems where an action $a \in \Sigma$ is executed simultaneously by any component $i \in loc(a)$. The behavior of

[1] As a consequence, the underlying graph has bounded degree. This property is essential in establishing the coincidence between recognizability and logical definability [2].

such a "shared-memory" system is described naturally by a set of traces. Commonly, traces are defined as congruence classes of words or as dependence graphs. In our setting, we model a trace as the union of the Hasse diagrams of the total orders of the different agents. Moreover, the labeling of an edge between two nodes u and v provides information about which agents execute u and v consecutively. In detail, a *trace* over $\widetilde{\Sigma}$ is a dag $(V, \{\lhd_\ell\}_{\ell \in 2^{Ag}}, \lambda)$ from $\mathbb{DAG}(\widetilde{\Sigma}, 2^{Ag})$ such that both $\lhd = \bigcup_{i \in Ag} \lessdot_i$ and, for any $(u, v) \in \lhd$ and $\ell \in 2^{Ag}$, $u \lhd_\ell v$ iff $\ell = \{i \in Ag \mid u \lessdot_i v\}$ (recall that $\lessdot_i$ is the cover relation of $\leq_i$). This modeling of a trace will turn out to be tremendously helpful when simulating shared-memory systems in terms of asynchronous cellular automata, as the edge relation will be used to access, for any event u and any agent $i \in loc(\lambda(u))$, the current state of $i \in Ag$ right before executing u. A sample trace over $\widetilde{\Sigma} = (\{a, b, c\}, \{a, b, d\}, \{a, b\})$ (with $Ag = \{1, 2, 3\}$) is depicted in Fig. 1(a).

Example 2.2 ((Lossy) Message Sequence Charts). Another communication paradigm is that of channel systems: several components $i \in Ag$ communicate by sending and receiving messages through channels. So let $Ch = (Ag \times Ag) \setminus id_{Ag}$ be the set of channels. To model the behavior of such a system, we need to fix supplies of send and receive actions: for $i \in Ag$, let Γ_i denote $\{i!j \mid (i, j) \in Ch\} \cup \{i?j \mid (i, j) \in Ch\}$, the set of *(communication) actions* of agent i. Action $i!j$ reads as "i sends a message to j". Accordingly, $j?i$ is the complementary receive action. Let $\widetilde{\Gamma}$ be the distributed alphabet $(\Gamma_i)_{i \in Ag}$. A *message sequence chart* (MSC) over Ag is a $\widetilde{\Gamma}$-dag $(V, \lhd, \lambda)$ such that, for any $i \in Ag$, $\lhd_i$ is the cover relation of $\leq_i$, for any $(u, v) \in \lhd$ with $\lambda(u)\ I_{\widetilde{\Gamma}}\ \lambda(v)$, $\lambda(u) = i!j$ and $\lambda(v) = j?i$ for some i, j, and, for any $u \in V$, there is $v \in V$ satisfying both $\lambda(u)\ I_{\widetilde{\Gamma}}\ \lambda(v)$ and either $u \lhd v$ or $v \lhd u$. Observe that, due to the general definition of a $\widetilde{\Gamma}$-dag, we deal with a model for FIFO communication. If we do not require a send event to be followed by a corresponding receive event, we deal with a *lossy* MSC. More precisely, the last condition in the definition of an MSC is weakened as follows: for any $v \in V$ with $\lambda(v)$ a receive action, there is $u \in V$ satisfying $\lambda(u)\ I_{\widetilde{\Gamma}}\ \lambda(v)$ and $u \lhd v$. Figure 1(b) depicts an MSC over $\{1, 2\}$, whereas the structure from Fig. 1(c) is not an MSC but a lossy MSC.

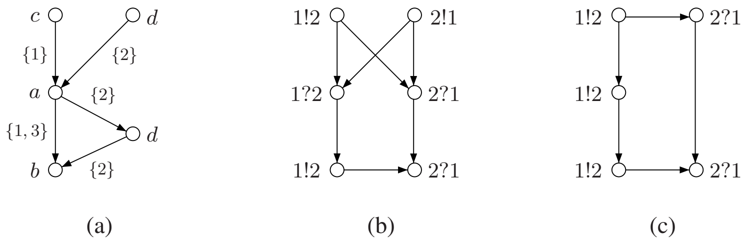


Fig. 1. A trace over $(\{a, b, c\}, \{a, b, d\}, \{a, b\})$, an MSC over $\{1, 2\}$, and a lossy MSC over $\{1, 2\}$ that is not an MSC

3 Weighted Asynchronous Cellular Automata

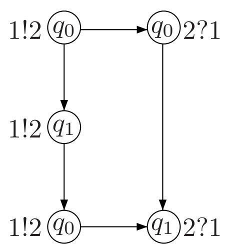
First we provide the unweighted model of an asynchronous cellular automaton, similar to the one proposed in [2]. Actually, we deal with asynchronous cellular automata *with types* (ACATs) over $(\widetilde{\Sigma}, C)$-dags, which have limited access to the future. To express "communication requests", a type function associates with any action a and any state q the set of actions that henceforth "communicate" with a, provided executing a results in state q. Regarding lossy MSCs, for example, we might require an event labeled with a send action 1!2 to be followed by the suitable receive event, which is then labeled with the communication action 2?1. For some classes the expressive power of ACAs with and without types coincide. But in general, omitting the type function severely restricts the expressive power of ACATs [2].

Definition 3.1. *An* asynchronous cellular automaton with types *(ACAT) over* $(\widetilde{\Sigma}, C)$ *is a structure* $\mathcal{A} = (Q, \Delta, T, F)$ *where*

- Q *is the nonempty finite set of* states*,*
- $\Delta \subseteq \mathit{Trans}_{(\widetilde{\Sigma},C)}(Q) := (Q \cup \{-\})^{\Sigma \times C} \times \Sigma \times Q$ *is the set of* transitions*,*
- $T : (\Sigma \times Q) \to 2^{\Sigma \times C}$ *is the* type function*, and*
- $F \subseteq (Q \cup \{\imath\})^{Ag}$ *is the set of global* final states*.*

We often write $(\overline{q}, a, q) \in \Delta$ with $\overline{q} \in (Q \cup \{-\})^{\Sigma \times C}$ as $\overline{q} \longrightarrow (a, q)$. Note that $\overline{q}[(b,\ell)] = -$ means that there is no (b,ℓ)-predecessor. Hence, we will sometimes write $\overline{q}$ as an element from $\mathfrak{P}(\Sigma \times C \times Q)$. The idea of a run of an asynchronous cellular automaton $\mathcal{A}$ on a $(\widetilde{\Sigma}, C)$-dag $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ is an additional labeling of the nodes $u \in V$ with states $q \in Q$ such that the local neighborhoods match the transitions, after executing $\mathcal{D}$ the system is in a final state, and the requests of the type function are satisfied.

First, let us consider the following example: $\mathcal{A} = (Q, \Delta, T, F)$ running on lossy MSCs over agents $\{1, 2\}$, cf. Example 2.2. We put $Q = \{q_0, q_1\}$. Now, the following transitions are in Δ: $\emptyset \to (1!2, q_0)$, $(1!2, q_0) \to (1!2, q_1)$, $(1!2, q_1) \to (1!2, q_0)$, $(1!2, q_0) \to (2?1, q_0)$, $\{(1!2, q_0), (2?1, q_0)\} \to (2?1, q_1)$, and $\{(1!2, q_1), (2?1, q_0)\} \to (2?1, q_1)$. Moreover, we put $T(1!2, q_0) = \{2?1\}$ and $F = \{(1, q_0), (2, q_1)\}$. Then the picture on the left hand side depicts a successful run of $\mathcal{A}$ on the lossy MSC from Figure 1(c). For every node, the node itself together with its read domain is covered by a transition. Furthermore, agent 1 stops in q_0 and agent 2 in q_1. Last but not least, every send event 1!2 in state q_0 is followed by a receive event 2?1 as imposed by the type function.



To be precise, let $\rho : V \to Q$. We write $(\mathcal{D}, \rho)$ to denote the dag $(V, \{\lhd_\ell\}_{\ell \in C}, (\lambda, \rho))$ over $(\Sigma \times Q, C)$. For $(\mathcal{D}, \rho)$, let $\mathit{trans}_{(\mathcal{D},\rho)} : V \to \mathit{Trans}_{(\widetilde{\Sigma},C)}(Q)$ describe the downward local neighborhood, i.e., for any $u \in V$ let $\mathit{trans}_{(\mathcal{D},\rho)}(u) = (\overline{q}, \lambda(u), \rho(u))$ where, for any $(b, \ell) \in \Sigma \times C$,

$$\overline{q}[(b,\ell)] = \begin{cases} - & \text{if } (b,\ell) \notin \mathrm{Read}(u), \\ \rho((b,\ell)\text{-pred}(u)) & \text{if } (b,\ell) \in \mathrm{Read}(u). \end{cases}$$

Moreover, we define $\mathit{final}_{(\mathcal{D},\rho)} \in (Q \uplus \{\imath\})^{Ag}$ by $\mathit{final}_{(\mathcal{D},\rho)}[i] = \imath$ for any agent $i \in Ag$ with $V_i = \emptyset$. Otherwise, $\mathit{final}_{(\mathcal{D},\rho)}[i] = \rho(u)$ where u is Σ_i-maximal in V. Thus, if the system starts in the global state $(\imath)_{i\in Ag}$ and executes $\mathcal{D}$, then it ends up in the global state $\mathit{final}_{(\mathcal{D},\rho)}$. Now a *run* of $\mathcal{A}$ on $\mathcal{D}$ is a mapping $\rho : V \to Q$ such that, for any $u \in V$, $\mathit{trans}_{(\mathcal{D},\rho)}(u) \in \Delta$. Moreover, ρ is *accepting* if both $\mathit{final}_{(\mathcal{D},\rho)} \in F$ and, for any $u \in V$, we have $T(\lambda(u), \rho(u)) \subseteq \mathrm{Write}(u)$. The intuition behind the latter condition is that we require $\mathrm{Write}(u)$ to contain at least the communication requests imposed by the type function of the automaton. The language $L(\mathcal{A})$ is the set of all $\mathcal{D}$ such that there is at least one accepting run of $\mathcal{A}$ on $\mathcal{D}$. We call $\mathcal{A}$ *unambiguous* if, for any $(\widetilde{\Sigma}, C)$-dag $\mathcal{D}$ and any two accepting runs ρ, ρ' of $\mathcal{A}$ on $\mathcal{D}$, we have $\rho = \rho'$.

A set $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$ is called *recognizable* if $L(\mathcal{A}) = L$ for some ACAT $\mathcal{A}$ over $(\widetilde{\Sigma}, C)$. Similarly, we say that L is *unambiguously recognizable* if $L(\mathcal{A}) = L$ for some unambiguous ACAT $\mathcal{A}$ over $(\widetilde{\Sigma}, C)$.

A weighted automaton is no longer characterized by the set of accepted executions. Rather, it assigns to any possible execution a value from a semiring. A semiring is a structure $\mathbb{K} = (K, \oplus, \circ, \mathbb{0}, \mathbb{1})$ with two binary operations, addition and multiplication, and constants $\mathbb{0}$ and $\mathbb{1}$, such that $(K, \oplus, \mathbb{0})$ is a commutative monoid, $(K, \circ, \mathbb{1})$ is a monoid, multiplication distributes over addition, and $\mathbb{0} \circ k = k \circ \mathbb{0}$ for any $k \in K$. We say $\mathbb{K}$ is *commutative* if the multiplication $\circ$ is commutative. Sample semirings are $(\mathbb{N}, +, \cdot, 0, 1)$, the 2-valued Boolean algebra $\mathbb{B} = (\{\mathbb{0}, \mathbb{1}\}, \vee, \wedge, \mathbb{0}, \mathbb{1})$, and the probabilistic semiring $\mathbb{P} = ([0, 1], \max, \cdot, 0, 1)$. Throughout this paper, we fix a commutative semiring $\mathbb{K} = (K, \oplus, \circ, \mathbb{0}, \mathbb{1})$. Commutativity is needed for a proper definition of automata behavior and several closure properties.

Definition 3.2. *A* **weighted** asynchronous cellular automaton with types *(wACAT) over* $\mathbb{K}$ *and* $(\widetilde{\Sigma}, C)$ *is a structure* (Q, μ, T, γ) *where*

- Q *is the nonempty finite set of* states*,*
- $\mu : \mathit{Trans}_{(\widetilde{\Sigma},C)}(Q) \to \mathbb{K}$ *is the* transition weight function*,*
- $T : (\Sigma \times Q) \to 2^{\Sigma\times C}$ *is the* type function*, and*
- $\gamma : (Q \uplus \{\imath\})^{Ag} \to \mathbb{K}$ *is the* final weight function.

In a wACAT, the values of a semiring that are collected along an execution of the automaton are multiplied, whereas nondeterminism is resolved by summation. The behavior of such an automaton will be a function $S : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{K}$, also called a *formal power series*. The collection of all these functions is denoted by $\mathbb{K}\langle\langle\mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$.

More precisely: let $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell\in C}, \lambda)$ be a $(\widetilde{\Sigma}, C)$-dag. In the weighted setting, every mapping $\rho : V \to Q$ is referred to as a *run*. The *weight* of ρ is the product

$$\mathit{weight}(\mathcal{D}, \rho) := \Big(\prod_{u\in V} \mu(\mathit{trans}_{(\mathcal{D},\rho)}(u))\Big) \circ \gamma(\mathit{final}_{(\mathcal{D},\rho)}) .$$

We call ρ *successful* if $T(\lambda(u), \rho(u)) \subseteq \mathrm{Write}(u)$ for any $u \in V$. We thus can assign to $\mathcal{A}$ a formal power series $\|\mathcal{A}\| \in \mathbb{K}\langle\langle\mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$ by

$$(\|\mathcal{A}\|, \mathcal{D}) := \bigoplus_{\substack{\rho:V\to Q\\ \rho \text{ successful}}} \mathit{weight}(\mathcal{D}, \rho)$$

for any $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, \lambda) \in \mathbb{DAG}(\widetilde{\Sigma}, C)$. Note that, in the context of formal power series, $(\|\mathcal{A}\|, \mathcal{D})$ is a common notation for $\|\mathcal{A}\|(\mathcal{D})$.

For $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$, the *characteristic series* $\mathbb{1}_L : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{K}$ is given by $(\mathbb{1}_L, \mathcal{D}) = \mathbb{1}$ if $\mathcal{D} \in L$ and $(\mathbb{1}_L, \mathcal{D}) = \mathbb{0}$ if $\mathcal{D} \notin L$. We say that $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$ is *recognizable* if there is a wACAT $\mathcal{A}$ with $\|\mathcal{A}\| = S$.

Example 3.1 (Probabilistic Lossy-Channel Systems [20]). A probabilistic lossy-channel system is a tuple $\mathcal{P} = ((Q_i, \delta_i)_{i \in Ag}, \overline{q}^{in}, (r_{ij}(q))_{(i,j) \in Ch, q \in Q_i})$: with any agent i, we associate a sequential process, which is composed of a finite state space Q_i and a transition relation $\delta_i \subseteq Q_i \times \Gamma_i \times Q_i$. Recall that Γ_i comprises the set of communication actions executed by agent i, i.e., actions of the form $i!j$ or $i?j$ with $i \neq j$. We shall assume δ_i to be deterministic, i.e., for any $q \in Q_i$ and $\sigma \in \Gamma_i$, there is at most one $q' \in Q_i$ such that $(q, \sigma, q') \in \delta_i$. Moreover, the system is equipped with a global initial state $\overline{q}^{in} \in \prod_{i \in Ag} Q_i$. There is an unreliable channel in between any two agents i and j with $i \neq j$, i.e., depending on a state $q \in Q_i$ in which a message is sent, a channel (i, j) has a reliability $r_{ij}(q) \in [0, 1]$. Thus, the message arrives at agent j with probability $r_{ij}(q)$ and is lost with probability $1 - r_{ij}(q)$.

We will give the probabilistic lossy-channel system $\mathcal{P}$ a semantics in terms of a wACAT $\mathcal{A}_\mathcal{P} = (Q, \mu, T, \gamma)$ over $\mathbb{P} = ([0, 1], \max, \cdot, 0, 1)$ and $\widetilde{\Gamma}$ reading lossy MSCs where $Q = (\bigcup_{i \in Ag} Q_i) \times \{\mathsf{success}, \mathsf{failure}, \mathsf{rec}\}$. Here, we give just the idea of the construction. Roughly speaking, we shift the reliabilities of the channels to the sequential processes. Then a state with second component $\mathsf{success}$ is assigned to a send event that succeeds in delivering a message, which is guaranteed by the type function, i.e., T maps a pair of the form $(i!j, (q, \mathsf{success}))$ to $\{j?i\}$ and any other pair to the empty set. Such a $\mathsf{success}$-state is entered with the probability that the transmission succeeds. In contrast, a send event that is equipped with a state that carries the attribute $\mathsf{failure}$ is entered with the probability that the transmission fails. Thus, it cannot be followed by a corresponding receive. Any other event will carry rec to indicate that we deal with a receive event. As we do not explicitly deal with final states, γ maps any possible final configuration to 1. For a lossy MSC $\mathcal{M}$, $(\| \mathcal{A}_\mathcal{P} \|, \mathcal{M}) \in [0, 1]$ might now be interpreted to be the probability of acceptance of $\mathcal{M}$ by $\mathcal{P}$.

Example 3.2 (Probabilistic Asynchronous Automata [12]). The model of asynchronous automata [22] over Mazurkiewicz traces represents shared-memory systems rather than channel systems. In an asynchronous automaton running on traces, any action a has to be executed simultaneously by any component $i \in loc(a)$. Probabilistic asynchronous automata have been introduced by Jesi, Pighizzini, and Sabadini [12]. In a probabilistic asynchronous automaton, the outcome of a transition depends on a probability distribution on the set of global states of the system. Formally, a *probabilistic asynchronous automaton* over $\widetilde{\Sigma}$ is a structure $\mathcal{B} = ((S_i)_{i \in Ag}, (\mathbf{P}_a)_{a \in \Sigma}, \overline{q}_0, \eta)$ where

- for each $i \in Ag$, S_i is a nonempty finite set of *(i-)local states*,
- for each $a \in \Sigma$, $\mathbf{P}_a$ is a mapping $S_a \times S_a \to [0, 1]$ such that, for any $\overline{s} \in S_a$, $\mathbf{P}_a(\overline{s}, .)$ is a probability distribution on S_a where $S_a := \{\overline{s} \in \prod_{i \in Ag}(S_i \cup \{*\}) \mid$ for any $i \in Ag$, $\overline{s}[i] = *$ iff $i \notin loc(a)\}$,
- $\overline{q}_0 \in \prod_{i \in Ag} S_i$ is the *global initial state*, and
- $\eta : \prod_{i \in Ag} S_i \to \{0, 1\}$ assigns a weight to any possible final configuration.

A probability distribution $\mathbf{P}_a(\overline{s})$ reflects that, in a global configuration from $\prod_{i\in Ag} S_i$ that coincides with $\overline{s}$ with respect to the locations from $loc(a)$, executing an a will alter at most the local states $\overline{s}$ of agents from $loc(a)$.

We provide the reader with a rather intuitive semantics of $\mathcal{B}$ and refer to [12] for details. Roughly speaking, $\mathcal{B}$ assigns to any trace a probability of acceptance. To determine the acceptance probability of a trace $\mathcal{T} = (V, \{\lhd_\ell\}_{\ell\in 2^{Ag}}, \lambda)$ over $\widetilde{\Sigma}$ (see Example 2.1), $\mathcal{B}$ will fix an arbitrary linear extension $w = (V, \leq', \lambda)$ of $\mathcal{T}$, i.e., $\leq'$ is a total-order relation containing $\leq$. As usual, w can be seen as a word $a_1 \ldots a_n \in \Sigma^*$ with $n = |V|$. Then, starting in the global initial state $\overline{q}_0$, $\mathcal{B}$ reads w letter by letter and assigns to any position $k = 1, \ldots, n$ a global state $\overline{q}_k \in \prod_{i\in Ag} S_i$ such that going from $\overline{q}_{k-1}$ to $\overline{q}_k$ changes at most the components from $loc(a_k)$, i.e., $\overline{q}_{k-1}[i] = \overline{q}_k[i]$ for any $i \notin loc(a_k)$. A step from $\overline{q}_{k-1}$ to $\overline{q}_k$ uniquely determines a pair $(\overline{s}_{k-1}, \overline{s}_k) \in S_{a_k} \times S_{a_k}$ with $\overline{s}_{k-1}[i] = \overline{s}_k[i] = *$ for any $i \notin loc(a_k)$ and $\overline{s}_{k-1}[i] = \overline{q}_{k-1}[i]$ and $\overline{s}_k[i] = \overline{q}_k[i]$ for any other i. The sequence $\overline{q}_0, \ldots, \overline{q}_n$ might be called a run of $\mathcal{B}$ on w. The weight of this particular run is the product $\prod_{k=1,\ldots,n} \mathbf{P}_{a_k}(\overline{s}_{k-1}, \overline{s}'_k) \cdot \eta(\overline{q}_n)$ (if $n = 0$, then we set its weight to be $\eta(\overline{q}_0)$). Summing up the weights of all possible runs of $\mathcal{B}$ on w determines the value $\mathbf{P}_{\mathcal{B}}(\mathcal{T}) \in [0, 1]$, the probability that $\mathcal{T}$ is accepted by $\mathcal{B}$.

Lemma 3.1. *There is a wACAT $\mathcal{A} = (Q, \mu, T, \gamma)$ over $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$ and $(\widetilde{\Sigma}, 2^{Ag})$ such that $|Q| \leq |\Sigma| \times |\prod_{i\in Ag} S_i|$ and $(\|\mathcal{A}\|, \mathcal{T}) = \mathbf{P}_{\mathcal{B}}(\mathcal{T})$ for any trace $\mathcal{T}$.*[2]

Proof. Let $Q = \bigcup_{a\in\Sigma} S_a$ and $T(a, \overline{s}) = \emptyset$ for any $(a, \overline{s}) \in \Sigma \times Q$.

- Suppose $t = \{((a_1, \overline{s}_1), \ell_1), \ldots, ((a_n, \overline{s}_n), \ell_n)\} \longrightarrow (a, \overline{s}) \in \mathit{Trans}_{(\widetilde{\Sigma}, 2^{Ag})}(Q)$. If $\overline{s}_k \in S_{a_k}$, $k = 1, \ldots, n$, $\overline{s} \in S_a$, and the sets $\ell_k \in 2^{Ag}$ are pairwise disjoint, then $\mu(t)$ is set to be $\mathbf{P}_a(\overline{s}', \overline{s})$ where $\overline{s}'$ is determined as follows: for any $i \in loc(a)$, $\overline{s}'[i] = \overline{q}_0[i]$ if $i \notin \bigcup_{k=1,\ldots,n} \ell_k$, and, otherwise, $\overline{s}'[i] = \overline{s}_k[i]$ for the unique $k \in \{1, \ldots, n\}$ with $i \in \ell_k$. Any other transition is mapped to 0.
- Suppose $\overline{q} \in (Q \uplus \{\imath\})^{Ag}$. If there is $\overline{q}' \in \prod_{i\in Ag} S_i$ such that, for any $i \in Ag$, $\overline{q}[i] = \imath$ implies $\overline{q}'[i] = \overline{q}_0[i]$ and $\overline{q}[i] \in Q$ implies $\overline{q}'[i] = \overline{q}[i][i]$, then set $\gamma(\overline{q})$ to be $\eta(\overline{q}')$. Otherwise, set $\gamma(\overline{q})$ to be 0. □

Note that (weighted) ACATs relative to traces can actually do without types. By Lemma 3.1 and Theorem 4.2, we will give, as a byproduct, a weighted formula defining the behavior of a probabilistic asynchronous automaton $\mathcal{B}$.

We collect some closure properties of recognizable series needed to show that definable series are recognizable. Let $S, S' \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$. Then, we define $k \circ S$ for $k \in \mathbb{K}$, $S + S'$, and $S \odot S'$ by $(k \circ S, \mathcal{D}) = k \circ (S, \mathcal{D})$, $(S + S', \mathcal{D}) = (S, \mathcal{D}) \oplus (S', \mathcal{D})$, and $(S \odot S', \mathcal{D}) = (S, \mathcal{D}) \circ (S', \mathcal{D})$ for any $\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}, C)$.

Proposition 3.1. *Let $S, S' : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{K}$ be recognizable and $k \in \mathbb{K}$. Then, $k \circ S$, $S + S'$, and $S \odot S'$ are recognizable.*

[2] Note that we calculate values in the interval $[0, 1]$ only. But unfortunately, $([0, 1], +, \cdot, 0, 1)$ is not a semiring. Therefore, we turn to $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$.

Now let Σ_i, Γ_i be arbitrary alphabets for $i \in Ag$ with $\Sigma = \bigcup_{i \in Ag} \Sigma_i$ and $\Gamma = \bigcup_{i \in Ag} \Gamma_i$. Moreover, let $\pi_v : \Sigma \to \Gamma$ such that $\pi_v(\Sigma_i) \subseteq \Gamma_i$ for all $i \in Ag$ and $(a,b) \in D_{\widetilde{\Sigma}}$ iff $(\pi_v(a), \pi_v(b)) \in D_{\widetilde{\Gamma}}$. Then, we call $\pi : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{DAG}(\widetilde{\Gamma}, C)$ with $\pi(\mathcal{D}) = (V, \{\lhd_l\}_{l \in C}, \pi_v \circ \lambda)$ for $\mathcal{D} = (V, \{\lhd_l\}_{l \in C}, \lambda) \in \mathbb{DAG}(\widetilde{\Sigma}, C)$ a *projection* from $\mathbb{DAG}(\widetilde{\Sigma}, C)$ to $\mathbb{DAG}(\widetilde{\Gamma}, C)$. Note that $\pi(\mathcal{D})$ is indeed a $(\widetilde{\Gamma}, C)$-dag because of the properties of π_v. For $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$, let $\pi(S)$ be the series defined for every $\mathcal{D}' \in \mathbb{DAG}(\widetilde{\Gamma}, C)$ by $(\pi(S), \mathcal{D}') = \bigoplus_{\mathcal{D} \in \pi^{-1}(\mathcal{D}')} (S, \mathcal{D})$.

Proposition 3.2. *Let $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$ and $\pi : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{DAG}(\widetilde{\Gamma}, C)$ be a projection. If S is recognizable, then $\pi(S) \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Gamma}, C) \rangle\rangle$ is recognizable.*

Proposition 3.3. *Let $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$ be an unambiguously recognizable language. Then, the characteristic series $\mathbb{1}_L$ over $\mathbb{K}$ is recognizable.*

4 Weighted Monadic Second-Order Logic

We fix sets $Var = \{x, y, \ldots\}$ of *first-order* and $VAR = \{X, Y, \ldots\}$ of *second-order variables*. Still, we assume the semiring $\mathbb{K}$ being commutative.

Definition 4.1. *The set* wMSO($\mathbb{K}, (\widetilde{\Sigma}, C)$) *of* weighted monadic second-order (wMSO) *formulas over $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$ is given by (let $k \in K$, $a \in \Sigma$, and $\ell \in C$):*

$$\begin{aligned}\varphi ::= & k \mid \lambda(x) = a \mid \neg(\lambda(x) = a) \mid x \lhd_\ell y \mid \neg(x \lhd_\ell y) \mid x = y \mid \neg(x = y) \mid \\ & x \in X \mid \neg(x \in X) \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x.\varphi \mid \exists X.\varphi \mid \forall x.\varphi \mid \forall X.\varphi\end{aligned}$$

The formulas k, $\lambda(x) = a$, $x \lhd_\ell y$, $x = y$ and $x \in X$ are called *atomic*. Negation of k has no reasonable semantics for general semirings. Thus, to obtain an intuitive interpretation of negation in terms of $\mathbb{0}$ and $\mathbb{1}$, it is pushed to the atomic level, omitting k. In exchange, we have to enrich the syntax by conjunction and universal quantification, cf. [7]. Let $Free(\varphi)$ be the set of free variables of φ and $\mathcal{V} \in 2^{Var \cup VAR}$ a finite set of variables. We say that $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, (\lambda, \rho))$ with $\rho : V \to \{0,1\}^{\mathcal{V}}$ is *valid* if, for any first-order variable $x \in \mathcal{V}$, there is a unique node $u \in V$ such that $\rho(u)[x] = 1$. In that case, $\rho(x)$ shall refer to u. Given $x \in \mathcal{V}$ and $u \in V$, we define the *update* $\rho[x/u] = \rho' : V \to \{0,1\}^{\mathcal{V}}$ such that $\rho'(u)[x] = 1$, $\rho'(v)[x] = 0$ for any $v \in V \setminus \{u\}$, and $\rho'(v)[\boldsymbol{x}] = \rho(v)[\boldsymbol{x}]$ for any $v \in V$ and $\boldsymbol{x} \in \mathcal{V} \setminus \{x\}$. Similarly, $\rho[X/V']$ is defined for $X \in \mathcal{V}$ and $V' \subseteq V$.

Note that, given a set $\mathcal{V}$ of variables, (weighted) ACATs can be extended to run on dags over $(\Sigma \times \{0,1\}^{\mathcal{V}}, C)$: We define a distributed alphabet $\widetilde{\Sigma}^{\mathcal{V}} = (\widetilde{\Sigma}_i^{\mathcal{V}})_{i \in Ag}$ by $\widetilde{\Sigma}_i^{\mathcal{V}} = \Sigma_i \times \{0,1\}^{\mathcal{V}}$.

Definition 4.2. *Suppose $\varphi \in$ wMSO($\mathbb{K}, (\widetilde{\Sigma}, C)$) and suppose $\mathcal{V} \in 2^{Var \cup VAR}$ is finite with $Free(\varphi) \subseteq \mathcal{V}$. The semantics of φ wrt. $\mathcal{V}$ is a series $[\![\varphi]\!]_{\mathcal{V}} \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C) \rangle\rangle$, given as follows: if $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, (\lambda, \rho)) \in \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)$ is not valid, we set $[\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \mathbb{0}$. Otherwise, $[\![\varphi]\!]_{\mathcal{V}}(\mathcal{D})$ is determined inductively as shown in Table 1.*

Table 1. The semantics of wMSO-formulas

$$[\![k]\!]_{\mathcal{V}}(\mathcal{D}) = k$$

$$[\![\lambda(x) = a]\!]_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \lambda(\rho(x)) = a \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$[\![x \lhd_\ell y]\!]_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \rho(x) \lhd_\ell \rho(y) \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$[\![x = y]\!]_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \rho(x) = \rho(y) \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$[\![x \in X]\!]_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \rho(x) \in \rho(X) \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$[\![\neg\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } [\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \mathbb{0} \\ \mathbb{0} & \text{if } [\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \mathbb{1} \end{cases}$$

$$[\![\varphi_1 \vee \varphi_2]\!]_{\mathcal{V}}(\mathcal{D}) = [\![\varphi_1]\!]_{\mathcal{V}}(\mathcal{D}) \oplus [\![\varphi_2]\!]_{\mathcal{V}}(\mathcal{D})$$

$$[\![\varphi_1 \wedge \varphi_2]\!]_{\mathcal{V}}(\mathcal{D}) = [\![\varphi_1]\!]_{\mathcal{V}}(\mathcal{D}) \circ [\![\varphi_2]\!]_{\mathcal{V}}(\mathcal{D})$$

$$[\![\exists x.\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \bigoplus_{u \in V} [\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}[x/u]])$$

$$[\![\forall x.\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \prod_{u \in V} [\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}[x/u])$$

$$[\![\exists X.\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \bigoplus_{V' \subseteq V} [\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}[X/V'])$$

$$[\![\forall X.\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \prod_{V' \subseteq V} [\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}[X/V'])$$

We abbreviate $[\![\varphi]\!]_{Free(\varphi)}$ by $[\![\varphi]\!]$. For $\mathbb{K}$ being the 2-valued Boolean algebra $\mathbb{B} = \{\mathbb{0}, \mathbb{1}\}$, wMSO$(\mathbb{B}, (\widetilde{\Sigma}, C))$ reduces to the usual MSO logic. Accordingly, $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$ is *FO-definable* if its support is definable in FO$(\mathbb{B}, (\widetilde{\Sigma}, C))$, i.e., in the fragment of wMSO$(\mathbb{B}, (\widetilde{\Sigma}, C))$ in which no second-order quantifier occurs. We say that the series $S \in \mathbb{K}\langle\langle\mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$ is an *FO-definable step function* if $S = \bigoplus_{i=1}^{n} k_i \circ \mathbb{1}_{L_i}$ for some $n \in \mathbb{N}$, $k_i \in K$, and FO-definable languages L_i. We call $\varphi \in$ wMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ *restricted* if it contains no universal second-order quantification and, for any subformula $\forall x.\psi$ of φ, $[\![\psi]\!]$ is an FO-definable step function. We denote the set of restricted wMSO-formulas over $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$ by wRMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$. Finally, let wREMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ be the *existential* fragment of wRMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$, which contains the formulas of the form $\exists X_1 \ldots \exists X_n.\varphi$ where the kernel formula $\varphi \in$ wRMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ contains no second-order quantifier.[3]

Even for words, wMSO has to be restricted because, otherwise, definability exceeds recognizability. While, in their logic, Droste and Gastin [7] deal with *recognizable* step functions exploiting the notion of determinism for finite automata, we have to cope with *FO-definable* functions in the context of dags. Fortunately, we can show unambiguity of $\mathbb{1}_L$ for FO-definable L, which is a cornerstone in establishing a logical characterization of wACATs.

Theorem 4.1. *Any FO-definable set of $(\widetilde{\Sigma}, C)$-dags is unambiguously recognizable.*

Proof (Sketch). It is well-known that any first-order formula can be written as the Boolean combination of statements "the pattern P occurs at least n times". Here, P is meant to be the (isomorphism type of the) environment of a node bounded by some radius $R \in \mathbb{N}$, also called an R-sphere. In [2], an ACAT $\mathcal{A}_R$ over dags detects the R-environment of any node. To transform the formula into an equivalent ACAT, we need to equip $\mathcal{A}_R$ with a (deterministic) threshold counting procedure to count how often a sphere is used in a run. However, $\mathcal{A}_R$ from [2] is not unambiguous due to some coloring

[3] It is not trivial to rewrite every wRMSO-formula into a wREMSO-formula. The problem is that it is not clear if for an FO-definable language L (as used in an FO-definable step function) the characteristic series $\mathbb{1}_L$ is again wFO-definable (see also the discussion in Section 6).

of spheres that is not unique. Such a coloring can be performed unambiguously so that any first-order formula can be simulated by an unambiguous ACAT. □

Corollary 4.1. $\{\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C) \mid \mathcal{D}$ *valid*$\}$ *is unambiguously recognizable.*

Proof. It suffices to show FO-definability. In fact, it is easy to provide an FO formula requiring that, for any first-order variable x, there is exactly one node whose labeling is 1 in the component that corresponds to x. □

Example 4.1. Consider the ring $\mathbb{Z} = (\mathbb{Z}, +, \cdot, 0, 1)$ and the class of lossy message sequence charts with $Ag = \{1, 2\}$, cf. Example 2.2. Then the formula

$$\varphi = (\exists x.\lambda(x) = 1!2) \vee (\exists y. -1 \wedge \lambda(y) = 2?1)$$

defines a series $[\![\varphi]\!]$ which maps every lossy MSC $\mathcal{M}$ to the number of messages from process 1 to 2 that are lost.

The remainder of this paper is dedicated to the proof of our main theorem:

Theorem 4.2. *Let* $\mathbb{K}$ *be a commutative semiring and* $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$. *Then, the following are equivalent:*

1. *S is recognizable,*
2. *S is wRMSO-definable, and*
3. *S is wREMSO-definable.*

5 Definable Series Are Recognizable

In this section we show that series defined by restricted formulas are recognizable. Due to Corollary 4.1 and Propositions 3.1 and 3.3, we can restrict to valid $(\widetilde{\Sigma}, C)$-dags.

By the closure properties of wACATs as stated in Propositions 3.1 and 3.2, we get:

Proposition 5.1. *Let* $\varphi, \psi \in \mathrm{wMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$.

(a) If φ *is atomic or the negation of an atomic formula, then* $[\![\varphi]\!]$ *is recognizable.*
(b) If $[\![\varphi]\!]$ *and* $[\![\psi]\!]$ *are recognizable, then* $[\![\varphi \vee \psi]\!]$ *and* $[\![\varphi \wedge \psi]\!]$ *are recognizable.*
(c) If $[\![\varphi]\!]$ *is recognizable, then* $[\![\exists x.\varphi]\!]$ *and* $[\![\exists X.\varphi]\!]$ *are recognizable.*

Proposition 5.2. *Let* $\varphi \in \mathrm{wMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ *with* $[\![\varphi]\!] = \sum_{i=1}^{n} k_i \circ \mathbb{1}_{L_i}$ *an FO-definable step function. Then,* $[\![\forall x.\varphi]\!]$ *is recognizable.*

Proof (Sketch). Let $\mathcal{W} = \mathrm{free}(\varphi)$ and $\mathcal{V} = \mathrm{free}(\forall x.\varphi) = \mathcal{W} \setminus \{x\}$. Furthermore, $[\![\varphi]\!] = \sum_{i=1}^{n} k_i \mathbb{1}_{L_i}$ with $k_i \in \mathbb{K}$ and $L_i \subseteq \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{W}}, C)$ FO-definable languages for $i = 1, \ldots, n$. By Theorem 4.1, every L_i is recognized by an unambiguous ACAT. FO-definable languages are closed under union, complement and intersection. Therefore, $\{L_i \mid i = 1, \ldots, n\}$ can be assumed being a partition of $\mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{W}}, C)$ with $k_i \neq k_j$ for $i \neq j$. First, let $x \in \mathcal{W}$. We put $\Gamma = \Sigma \times \{1, \ldots, n\}$ and consider $(\widetilde{\Gamma}^{\mathcal{V}}, C)$-dags $\widetilde{\mathcal{D}} = (V, \{\lhd_l\}_{l \in C}, (\lambda, \sigma, \rho))$ with $\lambda : V \to \Sigma$, $\sigma : V \to \{1, \ldots, n\}$, and ρ :

$V \to \{0,1\}^{\mathcal{V}}$. Let $\widetilde{L} \subseteq \mathbb{DAG}(\widetilde{\Gamma}^{\mathcal{V}}, C)$ such that, for any $\widetilde{\mathcal{D}} \in \widetilde{L}$, any $v \in V$, and any $i \in \{1,\ldots,n\}$, we have $(\sigma(v) = i) \iff (V, \{\lhd_l\}_{l\in C}, (\lambda, \rho[x/v])) \in L_i$. Note that $\rho[x/v] : V \to \{0,1\}^{\mathcal{W}}$.

Since the L_i are FO-definable, one can build an FO-formula $\widetilde{\varphi}$ defining $\widetilde{L}$ (we omit the details). As $\widetilde{\varphi}$ is an FO-formula, the language $\widetilde{L} = L(\widetilde{\varphi})$ is recognizable by an unambiguous ACAT $\widetilde{\mathcal{A}} = (Q, \Delta, T, F)$ by Theorem 4.1. Let $t = (\overline{q}, (a, \sigma_t, \rho_t), q) \in \mathit{Trans}_{(\widetilde{\Gamma}^{\mathcal{V}},C)}(Q)$ with $a \in \Sigma$, $\sigma_t \in \{1,\ldots,n\}$, and $\rho_t \in \{0,1\}^{\mathcal{V}}$. We transform $\widetilde{\mathcal{A}} = (Q, \Delta, T, F)$ into a wACAT $\mathcal{A} = (Q, \mu, T, \gamma)$ by adding weights as follows: set $\mu(t)$ to be k_i if $t \in \Delta$ and $\sigma_t = i$. Otherwise, $\mu(t) = \mathbb{0}$. Moreover, $\gamma(q_1,\ldots,q_{|Ag|}) = \mathbb{1}$ if $(q_1,\ldots,q_{|Ag|}) \in F$, and $\gamma(q_1,\ldots,q_{|Ag|}) = \mathbb{0}$ otherwise. Since $\widetilde{\mathcal{A}}$ is unambiguous and recognizes $\widetilde{L}$, the weight of an extended dag $\widetilde{\mathcal{D}} \in \widetilde{L}$ in $\mathcal{A}$ is $\prod_{1\le i\le n} k_i^{|\sigma^{-1}(i)|}$ and for $\widetilde{\mathcal{D}} \notin \widetilde{L}$ we have $(\|\mathcal{A}\|, \widetilde{\mathcal{D}}) = \mathbb{0}$. Now we consider the projection $h : \mathbb{DAG}(\widetilde{\Gamma}^{\mathcal{V}}, C) \to \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)$ mapping $\widetilde{\mathcal{D}} = (V, \{\lhd_l\}_{l\in C}, (\lambda, \sigma, \rho))$ to $\mathcal{D} = (V, \{\lhd_l\}_{l\in C}, (\lambda, \rho))$. Note that for $\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)$ there is a unique $\widetilde{\mathcal{D}} \in \widetilde{L}$ with $h(\widetilde{\mathcal{D}}) = \mathcal{D}$. Hence, we have

$$\begin{aligned}(h(\|\mathcal{A}\|), \mathcal{D}) &= \bigoplus_{\widetilde{\mathcal{D}} \in h^{-1}(\mathcal{D}) \cap \widetilde{L}} (\|\mathcal{A}\|, \widetilde{\mathcal{D}}) = (\|\mathcal{A}\|, \widetilde{\mathcal{D}}) = \prod_{1\le i\le n} k_i^{|\sigma^{-1}(i)|}\\ &= \prod_{v\in V} ([\![\varphi]\!], (\mathcal{D}, \rho[x/v])) = ([\![\forall x.\varphi]\!], \mathcal{D}) .\end{aligned}$$

By Proposition 3.2, $[\![\forall x.\varphi]\!]$ is recognizable. The case $x \notin \mathcal{W}$ is derived easily. □

By Propositions 5.1 and 5.2, we have immediately:

Theorem 5.1. *Let $\mathbb{K}$ be a commutative semiring and let $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$. If S is wRMSO-definable, then S is also recognizable.*

6 Recognizable Series Are Definable

For $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$, let $\mathrm{Supp}(S) = \{\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}, C) \mid (S, \mathcal{D}) \neq \mathbb{0}\}$. We adopt the notion of an *unambiguous* FO($\mathbb{K}, (\widetilde{\Sigma}, C)$)-formula [7]:

– All atomic formulas apart from k and their negations are unambiguous.
– If φ and ψ are unambiguous, then so are $\varphi \wedge \psi$, $\forall x.\varphi$, and $\forall X.\varphi$.
– If φ and ψ are unambiguous and $\mathrm{Supp}([\![\varphi]\!]) \cap \mathrm{Supp}([\![\psi]\!]) = \emptyset$, then $\varphi \vee \psi$ is unambiguous.
– If, finally, φ is unambiguous and, for any $(\mathcal{D}, \rho)$ with $\rho : V \to \{0,1\}^{\mathit{Free}(\varphi)}$, there is at most one vertex u of $\mathcal{D}$ such that $[\![\varphi]\!]_{\mathit{Free}(\varphi)\cup\{x\}}(\mathcal{D}, \rho[x/u]) \neq \mathbb{0}$, then $\exists x.\varphi$ is unambiguous.

Observe that, though, syntactically, we deal with ordinary MSO formulas, an unambiguous formula is primarily a weighted formula, as unambiguousness is defined in terms of its series. Let $\varphi \in \mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$. If φ is unambiguous, then $[\![\varphi]\!]$ is an FO-definable step function. We know from [7] that certain simple formulas can be made unambiguous:

Proposition 6.1 ([7]). *Let $\varphi \in \mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ be a (positive) Boolean combination of atomic formulas apart from k and their negations. Then, there is an unambiguous formula $\varphi^+ \in \mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ such that $[\![\varphi^+]\!] = [\![\varphi]\!]$.*

Proof. We proceed by induction and simultaneously define formulas φ^+ and φ^-. If $\varphi \in \mathrm{FO}(\mathbb{K}, (\Sigma, C))$ is atomic or the negation of an atomic formula, we set $\varphi^+ = \varphi$ and $\varphi^- = \neg\varphi$ (where $\neg\neg\psi$ is reduced to ψ). Moreover, we let

– $(\varphi \vee \psi)^+ = \varphi^+ \vee (\varphi^- \wedge \psi^+)$,
– $(\varphi \vee \psi)^- = \varphi^- \wedge \psi^-$,
– $(\varphi \wedge \psi)^- = \varphi^- \vee (\varphi^+ \wedge \psi^-)$, and
– $(\varphi \wedge \psi)^+ = \varphi^+ \wedge \psi^+$. □

In the context of words and an (E)MSO logic that employs the predicate $\leq$ instead of the direct successor relation, Droste and Gastin need to transform an ordinary MSO formula φ into an unambiguous weighted MSO formula φ' such that $[\![\varphi']\!]$ is the characteristic series of the language of φ [7]. To this aim, they identify the unique least position of a word (wrt. $\leq$) that satisfies a given property. In our logic, such an identification is no longer feasible. Nevertheless, we can transform any wACAT into an equivalent weighted formula.

Theorem 6.1. *Let $\mathcal{A}$ be a wACAT over commutative $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$. There is a sentence ψ from* wREMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ *such that $[\![\psi]\!] = \|\mathcal{A}\|$.*

Proof. Let $\mathcal{A} = (Q, \mu, T, \gamma)$ be a wACAT over $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$. In the following, t and t' will range over $\mathit{Trans}_{(\widetilde{\Sigma},C)}(Q)$. Set $\overline{X}$ to be a collection (X_t) of second-order variables and suppose $Ag = \{1, \ldots, N\}$ for some $N \in \mathbb{N}$. The construction of a wREMSO sentence from $\mathcal{A}$ follows the route of transforming a finite automaton into a formula where an interpretation of second-order variables reflects an assignment of vertices to transitions. We first provide some building blocks of the desired wREMSO formula.

The unambiguous formula

$$\mathrm{Partition}(\overline{X}) := \forall x. \bigvee_{t} (x \in X_t \wedge \bigwedge_{t' \neq t} \neg(x \in X_{t'}))$$

claims that $\overline{X}$ actually represents a run, i.e., an assignment of vertices to transitions.

Given $a \in \Sigma$ and $q \in Q$, $\varphi^+_{(a,q)}(x)$ $(\varphi^-_{(a,q)}(x))$ shall denote the disjunction (conjunction) of formulas $x \in X_t$ $(\neg(x \in X_t))$ such that $t = (\overline{q}, a, q)$ for some $\overline{q}$, respectively.

Now let $t = \{((a_1, q_1), \ell_1), \ldots, ((a_m, q_m), \ell_m)\} \longrightarrow (a, q)$. To ensure that x is contained in X_t only if the transition taken at x corresponds to t, we use

$$\begin{aligned}\mathrm{Trans}_t(x, \overline{X}) := \quad & x \in X_t \wedge \lambda(x) = a \wedge \bigwedge_{k \in \{1,\ldots,m\}} \exists y. \Big[y \lhd_{\ell_k} x \wedge \varphi^+_{(a_k,q_k)}(y) \Big]^+ \\ & \wedge \forall y. \Big[\bigwedge_{\ell \in C} \neg(y \lhd_\ell x) \vee \bigvee_{k \in \{1,\ldots,m\}} \Big(y \lhd_{\ell_k} x \wedge \lambda(y) = a_k \Big) \Big]^+ .\end{aligned}$$

Another difficulty is to determine the weight of a global final state $\overline{q}$ with respect to an extended dag. We would like to identify, for any agent $i \in Ag$ with $\overline{q}[i] \neq \imath$, the Σ_i-maximal node. For this purpose, we demand the unique upwards-closed set of nodes Y that contains a single minimal element x such that x is the only node executed by agent i. Then, x is Σ_i-maximal and shall be contained in X_t for some transition $t = \overline{q} \longrightarrow (a, \overline{q}[i])$. Therefore, we define $\max_i(x, Y)$

$$\begin{aligned} \max_i(x,Y) := \ & \Big[\bigvee_{a\in\Sigma_i} \lambda(x) = a\Big]^+ \wedge x \in Y \\ & \wedge \forall y.\forall z.\Big[\neg(y \in Y) \vee \neg(y \lhd z) \vee z \in Y\Big]^+ \\ & \wedge \forall y.\Big[\neg(y \in Y) \vee \bigwedge_{a\in\Sigma_i} \neg(\lambda(y) = a) \vee y = x\Big]^+ \\ & \wedge \forall y.(\neg\varphi_1 \vee (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \neg\varphi_2 \wedge \varphi_3)) \end{aligned}$$

where $\varphi_1 = (y \in Y)$, $\varphi_2 = (y = x)$, and $\varphi_3 = \exists z.(z \in Y \wedge [\bigvee_{(a,\ell)\in\Sigma\times C} (z \lhd_\ell y \wedge \lambda(z) = a)]^+)$. Hereby, the last conjunct ensures unambiguity of $\max_i(x, Y)$ by requiring that x is the *only* minimal event in Y.

To collect the weights of global final states, we require, for any $\overline{q} \in (Q \cup \{\imath\})^{Ag}$, a formula $\mathrm{Final}_{\overline{q}}(\overline{X}, Y_1, \ldots, Y_N) :=$

$$\bigwedge_{\substack{i\in Ag\\ \overline{q}[i]\in Q}} \exists x.\Big[\max_i(x,Y_i) \wedge \Big(\bigvee_{a\in\Sigma} \varphi^+_{(a,\overline{q}[i])}(x)\Big)^+\Big] \wedge \bigwedge_{\substack{i\in Ag\\ \overline{q}[i]=\imath}} \forall x. \bigwedge_{a\in\Sigma_i} \neg(\lambda(x) = a)\,.$$

To simulate the type function T, we make use of the unambiguous formula $\mathrm{Type}(\overline{X}) :=$

$$\forall x. \bigwedge_{(a,q)\in\Sigma\times Q} \Big[\varphi^-_{(a,q)}(x) \vee \Big([\varphi^+_{(a,q)}(x)]^+ \wedge \bigwedge_{(b,\ell)\in T(a,q)} \exists y.(x \lhd_\ell y \wedge \lambda(y) = b)\Big)\Big]^+\,.$$

We are now prepared to specify the desired formula ψ. Namely, setting

$$\begin{aligned} \psi'(\overline{X}) = & \exists Y_1 \ldots \exists Y_N. \\ & \mathrm{Partition}(\overline{X}) \wedge \bigwedge_t \forall x.(\neg(x \in X_t) \vee \mathrm{Trans}_t(x, \overline{X})) \\ & \wedge \bigwedge_{i\in Ag} \Big(\exists x.\max_i(x,Y_i)\Big) \vee \forall x.\Big(\neg(x \in Y_i) \wedge \bigwedge_{a\in\Sigma_i} \neg(\lambda(x) = a)\Big) \\ & \wedge \bigwedge_t \forall x.(\neg(x \in X_t) \vee ((x \in X_t) \wedge \mu(t))) \\ & \wedge \mathrm{Type}(\overline{X}) \wedge \bigvee_{\overline{q}\in F} \Big(\mathrm{Final}_{\overline{q}}(\overline{X}, Y_1, \ldots, Y_N) \wedge \gamma(\overline{q})\Big)\,, \end{aligned}$$

we finally let $\psi = \exists\overline{X}.\psi' \in \mathrm{wREMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$. Observe that the subformula $\neg(x \in X_t) \vee ((x \in X_t) \wedge \mu(t))$ of ψ is an FO-definable step function.

In fact, for any $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell\in C}, \lambda) \in \mathbb{DAG}(\widetilde{\Sigma}, C)$, we have $[\![\psi]\!](\mathcal{D}) = (\|\mathcal{A}\|, \mathcal{D})$. Thus, we obtain $[\![\psi]\!] = \|\mathcal{A}\|$. $\square$

References

1. C. Baier and M. Größer. Recognizing omega-regular languages with probabilistic automata. In *Proceedings of LICS 2005*. IEEE Computer Society Press, 2005.
2. B. Bollig. On the expressiveness of asynchronous cellular automata. In *Proceedings of FCT 2005*, volume 3623 of *Lecture Notes in Comp. Sc.*, pages 528–539. Springer, 2005.
3. J. Büchi. Weak second order arithmetic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.
4. K. Culik and J. Kari. Image compression using weighted finite automata. *Computer and Graphics*, 17(3):305–313, 1993.
5. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, Singapore, 1995.
6. M. Droste and P. Gastin. The Kleene-Schützenberger theorem for formal power series in partially commuting variables. *Inform. and Comp.*, 153:47–80, 1999.
7. M. Droste and P. Gastin. Weighted automata and weighted logics. In *Proceedings of ICALP 2005*, volume 3580 of *Lecture Notes in Comp. Sc.*, pages 513–525. Springer, 2005.
8. M. Droste, P. Gastin, and D. Kuske. Asynchronous cellular automata for pomsets. *Theoret. Comp. Sc.*, 247(1-2):1–38, 2000.
9. M. Droste and G. Rahonis. Weighted automata and weighted logics on infinite words. In *10th Int. Conf. on Developments in Language Theory (DLT)*, volume 4036 of *Lecture Notes in Comp. Sc.*, pages 49–58. Springer, 2006.
10. M. Droste and H. Vogler. Weighted tree automata and weighted logics. *Theoret. Comp. Sc.*, 366:228–247, 2006.
11. C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.*, 98:21–52, 1961.
12. S. Jesi, G. Pighizzini, and N. Sabadini. Probabilistic asynchronous automata. *Mathematical Systems Theory*, 29(1):5–31, 1996.
13. W. Kuich. Semirings and Formal Power Series. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, chapter 9, pages 609–677. Springer, 1997.
14. D. Kuske. Emptiness is decidable for asynchronous cellular machines. In *Proceedings of CONCUR 2000*, volume 1877 of *Lecture Notes in Comp. Sc.*, pages 536–551. Springer, 2000.
15. D. Kuske. Weighted asynchronous cellular automata. In *Proceedings of STACS 2006*, volume 3884 of *Lecture Notes in Comp. Sc.*, pages 685–696. Springer, 2006.
16. I. Mäurer. Weighted picture automata and weighted logics. In *Proceedings of STACS 2006*, volume 3884 of *Lecture Notes in Comp. Sc.*, pages 313–324. Springer, 2006.
17. I. Meinecke. Weighted logics for traces. In *Proceedings of CSR 2006*, volume 3967 of *Lecture Notes in Comp. Sc.*, pages 235–246. Springer, 2006.
18. M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.
19. E. Ochmański. Regular behaviour of concurrent systems. *Bulletin of the EATCS*, 27:56–67, 1985.
20. Ph. Schnoebelen. The verification of probabilistic lossy channel systems. In *Valid. of Stochastic Systems*, volume 2925 of *Lecture Notes in Comp. Sc.*, pages 445–465. Springer, 2004.
21. M.P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4:245–270, 1961.
22. W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21, 1987.

Weighted O-Minimal Hybrid Systems Are More Decidable Than Weighted Timed Automata!*

Patricia Bouyer, Thomas Brihaye, and Fabrice Chevalier

LSV - CNRS & ENS de Cachan
61, avenue du Président Wilson, 94230 Cachan, France
{bouyer,brihaye,chevalie}@lsv.ens-cachan.fr

Abstract. We consider weighted o-minimal hybrid systems, which extend classical o-minimal hybrid systems with cost functions. These cost functions are "observer variables" which increase while the system evolves but do not constrain the behaviour of the system. In this paper, we prove two main results: (*i*) optimal o-minimal hybrid games are decidable; (*ii*) the model-checking of WCTL, an extension of CTL which can constrain the cost variables, is decidable over that model. This has to be compared with the same problems in the framework of timed automata where both problems are undecidable in general, while they are decidable for the restricted class of one-clock timed automata.

1 Introduction

O-minimal hybrid systems. Hybrid systems are finite-state machines where each state is equipped with a continuous dynamics. In the last thirty years, formal verification of such systems has become a very active field of research in computer science. In this context, hybrid automata, an extension of timed automata [AD94], have been intensively studied [Hen95, Hen96], and decidable subclasses of hybrid systems have been identified like initialized rectangular hybrid automata [Hen96] or o-minimal hybrid automata. This latter model has been pointed out in [LPS00] as an interesting class of systems with very rich continuous dynamics, but limited discrete steps (at each discrete step, all variables have to be reset, independently from their initial values). Behaviours of such a system can be decoupled into continuous and discrete parts, properties of a global o-minimal system can thus be deduced directly from properties of the continuous parts of the system. This property and properties of o-minimal structures (see [vdD98] for an overview) are exploited in the word encoding techniques, which have been developed in [BMRT04] for (finitely) abstracting behaviours of the system. Using techniques based on this abstraction, reachability properties [BM05], reachability control properties [BBC06] have been proved decidable for o-minimal hybrid systems. This technique was also used in order to compute a (tight) exponential bound on the size of the coarsest finite bisimulation of *Pfaffian hybrid systems* (see [KV06]).

* Work partly supported by ACI Sécurité Informatique CORTOS.

Models for resource consumption. A research direction which has recently received substantial attention is the twist or extension of (decidable) models for representing more fairly interesting properties of embedded systems, for instance resource consumption. In that context, timed automata [AD94] have been extended with cost information bringing the model of weighted (or priced) timed automata [ALP01, BFH$^+$01]. A timed automaton is a finite automaton with clock variables (*i.e.* variables which increase as global time) that can be tested towards constants or reset. In the model of weighted timed automata, an extra cost variable is added, which is used as an *observer* variable (it does not constrain the behaviour of the system), evolving linearly while time elapses, and subject to discrete jumps when discrete transitions are taken. This model was appealing for expressing quantitative properties of real-time systems, which was concretized by the decidability of the optimal reachability problem (find the best way — in terms of cost — of reaching a given state) [ALP01, BFH$^+$01, BBBR06] together with the development of the tool Uppaal Cora [cor06], and then by the computability of the optimal mean-cost (find the best way for the system to have a "cost per time unit" as low as possible) [BBL04]. However, more involved properties like cost-optimal reachability control (find the minimum cost that can be ensured for reaching a given state, whatever does the environment in which the system is embedded) or WCTL model-checking (WCTL extends the branching-time temporal logic CTL with cost constraints on modalities [BBR04, BBR06]) have been proved undecidable for weighted timed automata with three clocks or more, see [BBR04, BBR05, BBM06]. Though both problems have recently been proved decidable for one-clock weighted timed automata [BLMR06, BLM07] these undecidability results are nevertheless disappointing, because the one-clock assumption is rather restrictive.

Our contributions. In this paper, we propose a natural extension of o-minimal hybrid systems with (definable) positive cost functions which increase while time progresses and which can be used in an optimization criterion, as in the case of weighted timed automata. It is worth noticing here that though the underlying system is o-minimal, this extended model, called *weighted o-minimal hybrid automaton*, is not o-minimal as we do not require that the cost is reset when a discrete transition is taken. However, we prove in this paper that the cost-optimal reachability control problem and the WCTL model-checking problem are both decidable for this class of systems. Because of the existing results on weighted timed automata, this is really a surprise, and makes o-minimal hybrid systems an analyzable, though powerful model. The decidability results of course partly rely on the word encoding techniques that we mentioned earlier, but also require refinements and involved techniques, specific to each of the two problems.

2 General Background

Let $\mathcal{M}$ be a structure. In this paper when we say that some relation, subset or function is *definable*, we mean it is first-order definable in the sense of the structure $\mathcal{M}$. A general reference for first-order logic is [Hod97]. We denote by $\mathsf{Th}(\mathcal{M})$ the theory of $\mathcal{M}$. In the sequel we only consider structures $\mathcal{M}$ that are expansions of ordered groups containing two constant symbols, *i.e.* $\mathcal{M} = \langle M, +, 0, 1, <, \dots\rangle$.

2.1 O-Minimality

Let us recall the definition of o-minimal structures [PS86] and give some examples of such structures. The reader interested in o-minimality should refer to [vdD98] for further results and an extensive bibliography on this subject. In the sequel of the paper we focus on o-minimal structures with a decidable theory in order to obtain decidability and computability results.

Definition 1. *A totally ordered structure $\mathcal{M} = \langle M, <, \dots\rangle$ is* o-minimal *if every definable subset of M is a finite union of points and open intervals (possibly unbounded).*

Example 1. Examples of o-minimal structures are the ordered group of rationals $\langle \mathbb{Q}, <, +, 0, 1\rangle$, the ordered field of reals $\langle \mathbb{R}, <, +, \cdot, 0, 1\rangle$, the field of reals with restricted pfaffian functions and the exponential function [Wil96].

2.2 O-Minimal Dynamical Systems

In this subsection we define the notion of *o-minimal dynamical systems.*

Definition 2 (O-minimal dynamical system). *An* o-minimal dynamical system *is a pair $(\mathcal{M}, \gamma)$ where:*

- *$\mathcal{M} = \langle M, +, 0, 1, <, \dots\rangle$ is an o-minimal expansion of an ordered group,*
- *$\gamma : V_1 \times V \to V_2$ is a function definable in $\mathcal{M}$ (where $V_1 \subseteq M^{k_1}$, $V \subseteq M$, and $V_2 \subseteq M^{k_2}$).[1] The function γ is called the dynamics of the system.*

Classically, when M is the set of the real numbers, we see V as the time, $V_1 \times V$ as the space-time, V_2 as the (output) space and V_1 as the input space. We keep this terminology in the more general context of a structure $\mathcal{M}$.

Example 2. We can view the continuous dynamics of *timed automata* [AD94] as an o-minimal dynamical system. In this case, we have that $\mathcal{M} = \langle \mathbb{R}, <, +, 0, 1\rangle$ and the dynamics $\gamma : (\mathbb{R}^+)^n \times \mathbb{R}^+ \to (\mathbb{R}^+)^n$ is defined by $\gamma(x_1, ..., x_n, t) = (x_1 + t, ..., x_n + t)$.

We define a transition system associated with the dynamical system, this definition is an adaptation to our context of the classical *continuous transition system* in the case of hybrid systems (see [LPS00] for example).

Definition 3. *Given $(\mathcal{M}, \gamma)$ a dynamical system, we define a* transition system $T_\gamma = (Q, \Sigma, \to_\gamma)$ associated with $(\mathcal{M}, \gamma)$ *by: the set Q of states is V_2; the set Σ of*

[1] We use these notations in the rest of the paper.

events is $M^+ = \{m \in M \mid m \geq 0\}$*; the transition relation* $y_1 \xrightarrow{\tau}_\gamma y_2$ *is defined by:* $\exists x \in V_1,\ \exists t_1, t_2, \in V$ *s.t.* $t_1 \leq t_2,\ \gamma(x, t_1) = y_1,\ \gamma(x, t_2) = y_2$ *and* $\tau = t_2 - t_1$.

2.3 O-Minimal Hybrid Systems and Games

In this subsection, we define o-minimal hybrid systems and games

Definition 4 (O-minimal hybrid systems). *Let* $\mathcal{M} = (M, +, 0, 1, <, \cdots)$ *be an o-minimal structure. An* $\mathcal{M}$-hybrid system *(or simply* o-minimal hybrid system*)* $\mathcal{H}$ *is a tuple* $(Q, \Sigma, \delta, \gamma)$ *where* Q *is a finite set of locations,* Σ *is a finite set of actions,* δ *consists in a finite number of transitions* $(q, g, a, R, q') \in Q \times 2^{V_2} \times \Sigma \times 2^{V_2} \times Q$ *where the* guard g *and the* reset R *are definable in* $\mathcal{M}$*, and* γ *maps every location* $q \in Q$ *to a dynamic* $\gamma_q : V_1 \times V \to V_2$ *definable in* $\mathcal{M}$*.*

An $\mathcal{M}$-hybrid system $\mathcal{H} = (Q, \Sigma, \delta, \gamma)$ defines a *mixed transition system* $T_{\mathcal{H}} = (S, \Gamma, \rightarrow)$ where:

- the set S of states is $Q \times V_2$;
- the set Γ of labels is $M^+ \cup \Sigma$;
- the transition relation $(q, y) \xrightarrow{e} (q', y')$ is defined when:
  - $e \in \Sigma$ and there exists $(q, g, e, R, q') \in \delta$ with $y \in g$ and $y' \in R(y)$, or
  - $e \in M^+$, $q = q'$, and $y \xrightarrow{e}_{\gamma_q} y'$ where γ_q is the dynamic in location q.

We will also need more precise notions of transitions. When $(q, y) \xrightarrow{\tau} (q, y')$ with $\tau \in M^+$, this is due to some choice of $(x, t) \in V_1 \times V$ such that $\gamma_q(x, t) = y$. We say that $(q, x, t, y) \xrightarrow{\tau} (q', x', t', y')$ if $q = q'$, $x = x'$, $t' = t + \tau$, $\gamma_q(x, t) = y$ and $\gamma'_q(x', t') = y'$. We say that an action $(\tau, a) \in M^+ \times \Sigma$ is enabled in a state (q, x, t, y) if there exists (q', x', t', y') and (q'', x'', t'', y'') such that $(q, x, t, y) \xrightarrow{\tau} (q', x', t', y') \xrightarrow{a} (q'', x'', t'', y'')$. We then write $(q, x, t, y) \xrightarrow{\tau, a} (q'', x'', t'', y'')$. We note $\mathsf{Enb}(q, x, t, y)$ the set of actions enabled in (q, x, t, y).

A *run* of $\mathcal{H}$ is a(n) (in)finite sequence $\varrho = (q_0, x_0, t_0, y_0) \xrightarrow{\tau_1, a_1} (q_1, x_1, t_1, y_1) \cdots$ A *position* π along ϱ is a pair $(i, \tau) \in \mathbb{N} \times M^+$ such that $\tau \leq \tau_{i+1}$. We define $\varrho[(i, \tau)] = (q_i, \gamma_{q_i}(x_i, t_i + \tau))$. Let us notice that the positions of a given run are totally ordered in a natural way. If ϱ is finite we define $last(\varrho) = (q_n, x_n, t_n, y_n)$. We note $\mathsf{Runs}_f(\mathcal{H})$ the set of finite runs in $\mathcal{H}$.

An interesting tool to study hybrid systems is the *(time-abstract) bisimulation*[2] (see [Hen95]). One of the main results concerning o-minimal hybrid systems is that they admit finite bisimulation quotient. This result has been first proved in [LPS00], it was reproved in [Dav99] in a more topological way, amongst a lot of other interesting results. In [Bri06a], the existence of finite bisimulations for o-minimal hybrid systems is proved by means of the *suffix partition*, a technique initiated in [BMRT04, BM05, Bri06b].

[2] Two systems are time-abstract bisimilar whenever they can both do the same actions and wait some delay.

Theorem 5. ([Bri06a, Theorem 12.6.14])
Let $\mathcal{H}$ be an o-minimal hybrid system and $\mathcal{P}_{\mathcal{H}}$ be a finite and definable partition of $Q \times V_2$ respecting the guards and the resets of $\mathcal{H}$. There is a finite and definable partition, noted $\mathsf{Suf}(\mathcal{P}_{\mathcal{H}})$, inducing a bisimulation on $\mathcal{H}$.

O-minimal hybrid systems are models for *closed systems*, where all transitions are controlled. If we want to consider *open systems* where we distinguish between the actions of a *controller* and of an *environment*, we need to consider games on o-minimal hybrid systems. We are now going to define this notion.

Definition 6 (O-minimal game). *Let $\mathcal{M} = (M, +, 0, 1, <, \cdots)$ be an o-minimal structure. An $\mathcal{M}$-*game *(or simply an* o-minimal game*) is a tuple $(Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ where $\mathsf{Goal} \subseteq Q$ is a subset of winning locations, $(Q, \Sigma, \delta, \gamma)$ is an $\mathcal{M}$-hybrid system and Σ is partitioned into two subsets Σ_c and Σ_u corresponding to* controllable *and* uncontrollable *actions.*

Let $\mathcal{H}$ be an o-minimal game. The game is played by two players, the *controller* and the *environment*; the goal of the controller is to reach a winning state whatever the environment does. In every state s, the controller picks a delay τ and an action $a \in \Sigma_c$ such that there is a transition $s \xrightarrow{\tau,a} s'$. The environment has two choices:

– either it waits τ and executes a transition $s \xrightarrow{\tau,a} s'$ proposed by the controller,
– or it waits τ', $0 \le \tau' \le \tau$, and executes a transition $s \xrightarrow{\tau',u} s''$ with $u \in \Sigma_u$.

The game then evolves to a new state (according to the choice of the environment) and the two players proceed to play as before.

We will now formalize the semantic through the concept of *strategy*.

Definition 7 (Strategy). *A (controller)* strategy[3] *is a partial function λ from $\mathsf{Runs}_f(\mathcal{H})$ to $M^+ \times \Sigma_c$ such that for all runs ϱ in $\mathsf{Runs}_f(\mathcal{H})$, if $\lambda(\varrho)$ is defined, then $\lambda(\varrho)$ is enabled in $last(\varrho)$.*

Intuitively, the strategy tells what needs to be done for controlling the system: at each instant it tells how much time we need to wait and which controllable action needs to be done after this delay. Note that even when the environment follows the controller's choice, it has to choose between several edges, each one labeled by the action given by the strategy (because the original game is not supposed to be deterministic).

Let $\varrho = (q_0, x_0, t'_0, y_0) \xrightarrow{\tau_1,a_1} \ldots$ be a run, and set for every i, ϱ_i the prefix of ϱ ending at position $(i, 0)$. The run ϱ is said to be *consistent with a strategy* λ when for all i, if $\lambda(\varrho_i) = (\tau, a)$ then either $\tau_{i+1} = \tau$ and $a_{i+1} = a$, or $\tau_{i+1} \le \tau$ and $a_{i+1} \in \Sigma_u$. We denote by $\mathsf{Outcome}(s, \lambda)$ the set of runs starting from a state s of the (output) space consistent with the strategy λ. A run $\varrho = (q_0, x_0, t_0, y_0) \xrightarrow{\tau_1,a_1} \ldots \xrightarrow{\tau_p,a_p} (q_p, x_p, t_p, y_p)$ is said *winning* if $q_i \in \mathsf{Goal}$ for some i. A run ϱ is said *maximal* with respect to a strategy λ if it is infinite or if $\lambda(\varrho)$ is not defined. A strategy λ is *winning from a state* (q, y) if for all (x, t) such that $\gamma(x, t) = y$, all maximal runs starting in (q, x, t, y) compatible with λ are winning.

[3] In the context of control problems, a strategy is also called a *controller*.

3 Weighted O-Minimal Hybrid Systems and Games

In this section, we define the weighted o-minimal hybrid systems and games, which extend the two models of the previous section with cost functions. These cost functions give a quantitative information on the behaviours of the systems, which allows to give a measure of the performance of the system. These models are respectively inspired by the model of weighted (resp. priced) timed automata [ALP01, BFH$^+$01] and the model of weighted (resp. priced) timed games [ABM04, BCFL04].

3.1 Definitions

We consider cost functions which are **non-negative** and **time-non-decreasing**. Note that cost functions in weighted timed automata [ALP01, BFH$^+$01] satisfy these hypotheses.

A non-negative and time-non-decreasing cost function is a definable function $\mathsf{Cost} : Q \times V_1 \times V \times M^+ \to M^+$ such that for all $q \in Q$, $x \in V_1$, $t \in V$ and $\tau_1,\ \tau_2 \in M^+$ with $\tau_1 \leq \tau_2$ we have that $\mathsf{Cost}(q, x, t, \tau_1) \leq \mathsf{Cost}(q, x, t, \tau_2)$.

Definition 8 (Weighted o-minimal hybrid system and game). *An $\mathcal{M}$-*weighted hybrid system (resp. game) *is an $\mathcal{M}$-hybrid system (resp. game) $\mathcal{H} = (Q, \Sigma, \delta, \gamma)$ with a definable non-negative and time-non-decreasing cost function* Cost.

The semantics of an o-minimal weighted hybrid system (resp. game) is the one of the underlying o-minimal hybrid system (resp. game). Hence, the cost function does not affect the behaviours of the system; it gives for every single step of the system a non-negative value, which represents the cost of evolving following that step. It naturally extends to a run in the system: let $\varrho = (q_0, x_0, t_0, y_0) \xrightarrow{\tau_1, a_1} \dots \xrightarrow{\tau_p, a_p} (q_p, x_p, t_p, y_p)$ be a finite run in $\mathcal{H}$. The *cost of ϱ* is $\mathsf{Cost}(\varrho) = \sum_{i=1}^{p} \mathsf{Cost}(q_{i-1}, x_{i-1}, t_{i-1}, \tau_i)$.

Let us give an example of weighted o-minimal hybrid system.

Example 3 ([BLM07]). The weighted o-minimal hybrid system of Figure 1 models a never-ending process of repairing problems. The repair of a problem has a certain cost, captured in the model by the cost function Cost. As soon as a problem occurs (modeled by the *pb* transition) the value of the cost grows with rate 1, until actual repair is taking place by one of the transitions rp_1 (cheap but long repair) or rp_2 (expensive but quick repair). At most 24 time units after the occurrence of a problem it will have been repaired one way or another.

3.2 Related Problems and Results

In this subsection we define the two problems we are interested in: the *cost-optimal control problem* and the WCTL *model-checking problem.*

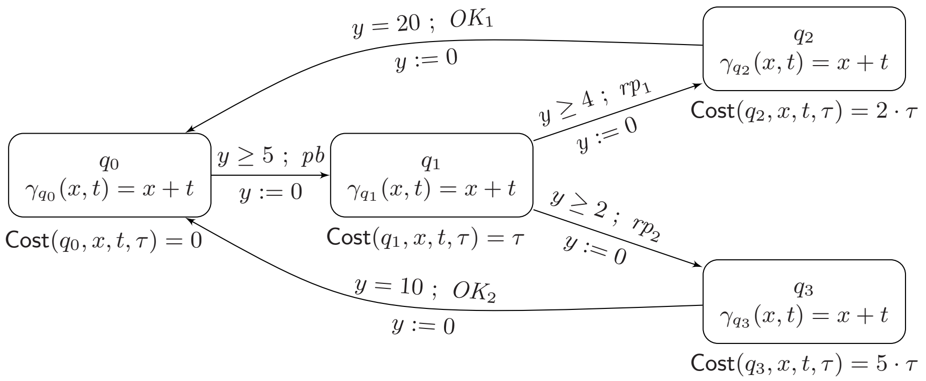

$y = 20$; OK_1
$y := 0$
q_2
$\gamma_{q_2}(x,t) = x + t$
$\mathsf{Cost}(q_2, x, t, \tau) = 2 \cdot \tau$
$y \geq 4$; rp_1
$y := 0$
q_0
$\gamma_{q_0}(x,t) = x + t$
$\mathsf{Cost}(q_0, x, t, \tau) = 0$
$y \geq 5$; pb
$y := 0$
q_1
$\gamma_{q_1}(x,t) = x + t$
$\mathsf{Cost}(q_1, x, t, \tau) = \tau$
$y \geq 2$; rp_2
$y := 0$
$y = 10$; OK_2
$y := 0$
q_3
$\gamma_{q_3}(x,t) = x + t$
$\mathsf{Cost}(q_3, x, t, \tau) = 5 \cdot \tau$

Fig. 1. A repair problem example

The Cost-Optimal Control Problem. The cost-optimal reachability control problem was first considered on weighted timed automata in [ABM04, BCFL04]. However it has been shown in [BBR05, BBM06] that the cost-optimal reachability control problem for weighted timed automata is undecidable.

In our context of weighted o-minimal games, the *cost-optimal control problem* asks what is the optimal cost for the controller to reach Goal whatever the environement does. In order to take the cost function into account, we now need to define the *cost of a strategy from a state* and the *optimal cost from a state.*

Definition 9 (Cost of a strategy from a state). *Let* $(\mathcal{H}, \mathsf{Cost})$ *be a weighted o-minimal game,* Goal *be a subset of winning locations,* s *be a state of the (output) space* V_2 *and* λ *be a strategy. The* cost $\mathsf{Cost}(s,\lambda)$ of λ from s is defined by:

$$\mathsf{Cost}(s,\lambda) = \sup\{\mathsf{Cost}(\varrho) \mid \varrho \in \mathsf{Outcome}(s,\lambda)\}.$$

Intuitively the presence of the supremum is explained by the fact that the environment tries to maximize the cost.

Definition 10 (Optimal cost from a state). *Let* $(\mathcal{H}, \mathsf{Cost})$ *be a weighted o-minimal game,* Goal *be a subset of winning locations and* s *be a state of the (output) space* V_2*. The* optimal cost $\mathsf{OptCost}(q)$ associated with s is defined by:

$$\mathsf{OptCost}(s) = \inf\{\mathsf{Cost}(s,\lambda) \mid \lambda \textit{ is a winning strategy}\}.$$

A winning strategy from s is called *optimal* whenever $\mathsf{Cost}(s,\lambda) = \mathsf{OptCost}(s)$.

Problem 1 (Cost-optimal control problem). Given a weighted o-minimal game $\mathcal{H}$, a definable state s and a definable constant c, decide if there exists a winning strategy λ from q such that $\mathsf{Cost}(s,\lambda) \leq c$.

Problem 2 (Computation of the optimal cost). Given a weighted o-minimal game $\mathcal{H}$ and a definable state s, compute the optimal cost $\mathsf{OptCost}(s)$.

Remark 1. There is an optimal winning strategy from state s iff the infimum can be replaced by a minimum in the definition of $\mathsf{OptCost}(s)$. If we solve problem 1 and 2, we can also determine if there is an optimal winning strategy by asking if there is a strategy λ with $\mathsf{Cost}(s, \lambda) \le \mathsf{OptCost}(s)$. In [BBR05, BBM06], it has been shown that the variant of Problem 1 for weighted timed automata is undecidable.

The WCTL Model-Checking Problem. The logic WCTL [4] has been proposed in the context of (weighted) timed systems as an extension of CTL with cost constraints on modalities [BBR04, BBM06, BLM07]. In our context, we define for every structure $\mathcal{M}$ the logic $\mathsf{WCTL}_{\mathcal{M}}$ over Σ inductively as follows:

$$\mathsf{WCTL}_{\mathcal{M}} \ni \varphi \ ::= \ a \ | \ \varphi \vee \varphi \ | \ \neg\varphi \ | \ \mathsf{E}\,\varphi \mathsf{U}_{\sim c}\varphi \ | \ \mathsf{A}\,\varphi \mathsf{U}_{\sim c}\varphi$$

where $a \in \Sigma$, $\sim \in \{<, \le, =, \ge, >\}$ and c is an $\mathcal{M}$-definable constant.

Let $(\mathcal{H}, \mathsf{Cost})$ be an $\mathcal{M}$-hybrid system. The semantics of $\mathsf{WCTL}_{\mathcal{M}}$ is defined for every state $(q, y) \in Q \times V_2$ of $(\mathcal{H}, \mathsf{Cost})$ as follows:

$$\begin{array}{rcl}
(q,y) \models a & \Leftrightarrow & (q,y) \xrightarrow{a} (q',y') \text{ for some } (q',y') \in Q \times V_2 \ ^{5} \\
(q,y) \models \neg\varphi & \Leftrightarrow & (q,y) \not\models \varphi \\
(q,y) \models \varphi_1 \vee \varphi_2 & \Leftrightarrow & (q,y) \models \varphi_1 \text{ or } (q,y) \models \varphi_2 \\
(q,y) \models \mathsf{E}\,\varphi_1 \mathsf{U}_{\sim c}\varphi_2 & \Leftrightarrow & \text{there is an infinite run } \varrho \text{ in } \mathcal{H} \text{ from } (q,y) \\
 & & \text{s.t. } \varrho \models \varphi_1 \mathsf{U}_{\sim c}\varphi_2 \\
(q,y) \models \mathsf{A}\,\varphi_1 \mathsf{U}_{\sim c}\varphi_2 & \Leftrightarrow & \text{every infinite run } \varrho \text{ in } \mathcal{H} \text{ from } (q,y) \\
 & & \text{satisfies } \varrho \models \varphi_1 \mathsf{U}_{\sim c}\varphi_2 \\
\varrho \models \varphi_1 \mathsf{U}_{\sim c}\varphi_2 & \Leftrightarrow & \text{there exists } \pi \ge 0 \text{ position along } \varrho \text{ s.t.} \\
 & & \varrho[\pi] \models \varphi_2, \text{ for all positions } 0 \le \pi' < \pi \text{ on } \varrho, \\
 & & \varrho[\pi'] \models \varphi_1 \vee \varphi_2, \text{ and } \mathsf{Cost}(\varrho_{\le\pi}) \sim c^{6}
\end{array}$$

We use `true` for $a \vee \neg a$, and classical "eventually" and "always" operators: $\mathsf{E}\,\mathsf{F}_{\sim c}\phi$ (resp. $\mathsf{A}\,\mathsf{F}_{\sim c}\phi$) stands for $\mathsf{E}\,\mathtt{true}\mathsf{U}_{\sim c}\phi$ (resp. $\mathsf{A}\,\mathtt{true}\mathsf{U}_{\sim c}\phi$) and $\mathsf{A}\,\mathsf{G}_{\sim c}\phi$ (resp. $\mathsf{E}\,\mathsf{G}_{\sim c}\phi$) stands for $\neg\mathsf{E}\,\mathsf{F}_{\sim c}\neg\phi$ (resp. $\neg\mathsf{A}\,\mathsf{F}_{\sim c}\neg\phi$).

Let us give an example of WCTL formulae on the repair problem of Example 3.

Example 4. [BLM07] An example of a property that can be expressed with WCTL is "*Whenever a problem occurs it can be repaired within a total cost of* 55". It can be expressed with the following formula:

$$\mathsf{A}\,\mathsf{G}\big(pb \implies \mathsf{E}\,\mathsf{F}_{\le 55}(OK_1 \vee OK_2)\big).$$

One can easily check that this formula holds for every state of the weighted o-minimal hybrid system of Figure 1.

Problem 3 (Model-checking of WCTL*).* Given $(\mathcal{H}, \mathsf{Cost})$ an $\mathcal{M}$-weighted hybrid system, φ a $\mathsf{WCTL}_{\mathcal{M}}$-formula and $(q, y) \in Q \times V_2$ a definable state of $\mathcal{H}$, decide whether $(q, y) \models \varphi$.

[4] WCTL stands for "Weighted CTL."
[5] This can be viewed equivalently as atomic propositions.
[6] Following [Ras99] we use $\varphi_1 \vee \varphi_2$ to handle open intervals in timed models.

Remark 2. Note that classical results on o-minimal hybrid systems cannot be used to solve the two problems presented above (for instance, weighted o-minimal hybrid systems do not have a finite bisimulation), and hence, *ad-hoc* proofs have to be developed.

4 Solving the Cost-Optimal Control Problem

In this section we prove the decidability of Problem 1.

Definition 11. *Let $(\mathcal{H}, \mathsf{Cost})$ be a weighted o-minimal game. We say that a run $(q_0, x_0, t_0, y_0) \xrightarrow{\tau_1, a_1} \ldots$ crosses the transition $e = (q, g, a, R, q')$ at step i if $q = q_{i-1}$, $q' = q_i$, $a = a_i$, $\gamma_{q_{i-1}}(x_{i-1}, t_{i-1} + \tau_i) \in g$ and $y_i \in R$.*

We are now going to prove a key proposition: we can restrict to winning strategies that cross each edge at most once. In order to prove this proposition we use the fact that the cost functions have non-negative values.

Proposition 12. *Let $\mathcal{H}$ be a weighted o-minimal game and (q, y) a state of $\mathcal{H}$. If there exists a winning strategy λ from (q, y) then there exists a winning strategy λ_{new} with $\mathsf{Cost}((q, y), \lambda_{new}) \leq \mathsf{Cost}((q, y), \lambda)$ such that every run starting in (q, y) compatible with λ_{new} crosses each transition at most once.*

Proof (Sketch). The idea of the proof is the following: for each transition e, we iteratively ensure that there is a winning strategy (of cost $c' \leq c$) crossing e at most once. See $\mathsf{Outcome}((q, y), \lambda)$ as a(n infinitely branching) tree: every path in this tree is finite and reaches Goal, but it may cross e several times. We construct a new strategy λ_{new} which short-cuts λ in the following sense: it simulates λ until e is crossed for the first time and then switches to a descendant in the tree from which all paths do not cross e anymore (this is possible as λ is winning). Such a strategy crosses e at most once, its cost is smaller than that of λ as its compatible runs are "shorter" than ones of λ and the cost function is non-negative. □

We now give a backward algorithm which computes the optimal cost, based on the formulation of the problem given in [LMM02, ABM04]. The termination of the algorithm relies on Proposition 12. For this, given $(q, y) \in Q \times V_2$ and $n \in \mathbb{N}$ we define $c_n(q, y)$, the optimal cost of reaching Goal from (q, y) in at most n steps. Formally we have that:

- $c_0(q, y) = 0$ if $q \in \mathsf{Goal}$, $+\infty$ if $q \notin \mathsf{Goal}$.
- $c_{n+1}(q, y) =$
$$\sup_{\gamma_q(x,t)=y} \inf_{(\tau,a)\in\mathsf{Enb}(q,x,t,y)} \max\begin{cases} \mathsf{Cost}_{c_n}^{\tau,a}(q,x,t,y) \\ \sup\{\mathsf{Cost}_{c_n}^{\tau',u}(q,x,t,y) \mid {\scriptstyle (\tau',u)\in\mathsf{Enb}(q,x,t,y),\tau'\leq\tau}\} \end{cases}$$
where $(\tau, e) \in \mathsf{Enb}(q, x, t, y)$ iff $e \in \mathsf{Enb}(q, x, t + \tau, \gamma_q(x, t + \tau))$, and $\mathsf{Cost}_c^{\tau,e}$ $(q, x, t, y) = \mathsf{Cost}(q, x, t, y, \tau) + \sup\{c(q', y') \mid (q, x, t, y) \xrightarrow{\tau,a} (q', x', t', y')\}$.

Definition 13. *A strategy λ is said n-bounded from (q, y) if every run compatible with λ starting from (q, y) has length at most n.*

Lemma 14. *For every $\varepsilon > 0$, for every $(q, y) \in Q \times V_2$ s.t. $c_n(q, y) < +\infty$, there exists a definable n-bounded strategy λ from (q, y) such that $\mathsf{Cost}((q, y), \lambda) \leq c_n(q, y) + \varepsilon$.*

Lemma 15. *If λ is an n-bounded strategy from (q, y) then $\mathsf{Cost}((q, y), \lambda) \geq c_n(q, y)$.*

Theorem 16. *Let $\mathcal{M} = \langle M, +, 0, 1, \ldots\rangle$ be an o-minimal structure such that $\mathsf{Th}(\mathcal{M})$ is decidable. The cost-optimal control problem over $\mathcal{M}$-weighted hybrid systems is decidable.*

Proof. Let $\mathcal{H}$ be a weighted o-minimal game and s a state of $\mathcal{H}$. Lemmas 14 and 15 imply that $c_k(s) = \inf\big\{\mathsf{Cost}(s, \lambda) \;\mid\; \lambda \text{ is a } k\text{-bounded strategy strategy}\big\}$.

Let n be the number of transitions of $\mathcal{H}$. Proposition 12 shows that for every winning strategy from s there is an $(n + 1)$-bounded winning strategy from s with smallest cost. Thus $\mathsf{OptCost}(s) = \inf\big\{\mathsf{Cost}(s, \lambda) \;\mid\; \lambda \text{ is an } (n + 1)\text{-bounded strategy}\big\} = c_{n+1}(s)$. Note that $c_{n+1}(s)$ is computable since $\mathsf{Th}(\mathcal{M})$ is decidable.

We can moreover decide if the optimal cost can be achieved by a strategy: by Proposition 12 it is sufficient to enumerate all $(n + 1)$-bounded strategies using τ_i's as parameters and check if the cost of one of them is equal to $c_{n+1}(s)$. $\square$

Remark 3. Let us notice that Theorem 16 encompasses the decidability of the time-bounded reachability problem considered in [Gen05]. Moreover, a consequence of Theorem 16 is the decidability of the (cost-)optimal reachability problem formulated in [ALP01, BFH$^+$01] for weighted timed automata.

5 Solving the WCTL Model-Checking Problem

The aim of this section is to prove the following decidability theorem. The techniques that we will develop for proving this result are partly inspired by the recent decidability proof for WCTL over one-clock weighted timed automata [BLM07].

Theorem 17. *Let $\mathcal{M} = \langle M, +, 0, 1, <, \ldots\rangle$ be an o-minimal structure such that $\mathcal{M}$ is Archimedian*[7] *and $\mathsf{Th}(\mathcal{M})$ is decidable. The model-checking of $\mathsf{WCTL}_{\mathcal{M}}$ over $\mathcal{M}$-weighted hybrid systems is decidable.*

If $\mathcal{P}$ and $\mathcal{P}'$ are two partitions, we write $\mathcal{P} \sqcap \mathcal{P}'$ for the joint partition,*i.e.* the smallest partition that refines both $\mathcal{P}$ and $\mathcal{P}'$. Let us notice that if $\mathcal{P}$ and $\mathcal{P}'$ are both finite and definable the joint partition $\mathcal{P} \sqcap \mathcal{P}'$ is also finite and definable.

Let $\mathcal{H} = (Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ be an $\mathcal{M}$-weighted hybrid system. On each location $q \in Q$, let us denote by $\mathcal{P}_q$ the partition induced by the guards and the resets associated with location q. We denote by $\mathcal{P}_{\mathcal{H}}$ the partition of state space $S = Q \times V_2$ induced by the $\mathcal{P}_q$'s. $\mathcal{P}_{\mathcal{H}}$ is a finite definable partition of S.

[7] A structure is *Archimedian* whenever for every pair $(a, b) \in M^2$ such that $a \neq 0$, there exists some integer n such that $n \cdot a > b$.

Two states of the same piece P of $\mathcal{P}_{\mathcal{H}}$ agree on all atomic formulae $a \in \Sigma$. We will now inductively construct for every $\mathsf{WCTL}_{\mathcal{M}}$-formula φ a refined (finite and definable) partition $\mathcal{P}_\varphi$ of $\mathcal{P}_{\mathcal{H}}$ such that two states of a piece of $\mathcal{P}_\varphi$ agree on formula φ. We will proceed in three steps: from partitions for ϕ and ψ we will successively construct a partition for $\mathsf{E}\,\phi\mathsf{U}\psi$ and $\mathsf{A}\,\phi\mathsf{U}\psi$, then for $\mathsf{E}\,\phi\mathsf{U}_{\sim c}\psi$ and finally for $\mathsf{A}\,\phi\mathsf{U}_{\sim c}\psi$.

The first step is achieved using the following lemma, which applies the construction of a finite and definable partition (the so-called suffix-partition) correct w.r.t. bisimulation (see Theorem 5), and thus for CTL-formulae [AHLP00, HMR05].

Lemma 18. *Let $\mathcal{P}_\phi$ (resp. $\mathcal{P}_\psi$) be a partition for ϕ (resp. ψ). The partition $\mathsf{Suf}(\mathcal{P}_\phi \sqcap \mathcal{P}_\psi)$ is a time-abstract bisimulation and is a partition for formula $\mathsf{E}\,\phi\mathsf{U}\psi$. That is, if P is a piece of $\mathsf{Suf}(\mathcal{P}_\phi \sqcap \mathcal{P}_\psi)$, then all states (q,y) with $y \in P$ either satisfy $\mathsf{E}\,\phi\mathsf{U}\psi$ or do not satisfy this formula. The same holds for $\mathsf{A}\,\phi\mathsf{U}\psi$.*

The second step, the construction of a (finite and definable) partition respecting $\mathsf{E}\,\phi\mathsf{U}_{\sim c}\psi$, is more involved.

Let us denote by $\mathcal{P}$ the partition $\mathsf{Suf}(\mathcal{P}_\phi \sqcap \mathcal{P}_\psi)$. Let q and q' be two locations of $\mathcal{H}$. Let P and P' be two pieces of $\mathcal{P}$. We give formulae which define the set of possible costs of paths from a state (or a piece of $\mathcal{P}$) to another one:

- $\theta^{\phi\vee\psi}_{(q,P)\to(q',P')}(c,y)$ expresses that it is possible to go from some (q,y) with $y \in P$ to some (q',y') with $y' \in P'$ by a continuous step followed by a discrete action, with cost c, and always satisfying $\phi \vee \psi$.
- $\theta^{\phi\vee\psi}_{(q,P\rightsquigarrow P')}(c,y)$ expresses that it is possible to go from (q,y) with $y \in P$ to some (q,y') with $y' \in P'$ by a continuous step (and no discrete action), with cost c, and always satisfying $\phi \vee \psi$.

From the two previous formulae, we define the following definable sets:

$$\kappa^{\phi\vee\psi}_{(q,P)\to(q',P')} = \{c \in M^+ \mid \exists y \in P \text{ s.t. } \theta^{\phi\vee\psi}_{(q,P)\to(q',P')}(c,y)\}\ ;$$

$$\kappa^{\phi\vee\psi}_{(q,P\rightsquigarrow P')} = \{c \in M^+ \mid \exists y \in P \text{ s.t. } \theta^{\phi\vee\psi}_{(q,P\rightsquigarrow P')}(c,y)\}\ ;$$

$$\lambda^{\phi\vee\psi}_{(q,P)\to(q',P')}(y) = \{c \in M^+ \mid \theta^{\phi\vee\psi}_{(q,P)\to(q',P')}(c,y)\}\ ;$$

$$\lambda^{\phi\vee\psi}_{(q,P\rightsquigarrow P')}(y) = \{c \in M^+ \mid \theta^{\phi\vee\psi}_{(q,P\rightsquigarrow P')}(c,y)\}.$$

We then construct a finite graph which abstracts away all dynamical parts of $\mathcal{H}$ and which will be restricted to parts of $\mathcal{H}$ which satisfy formula $\mathsf{E}\,\phi\mathsf{U}\psi$. Each edge of this graph will be labeled with a weight (indeed a definable set) which will represent the set of costs of all paths in $\mathcal{H}$ witnessing formula $\mathsf{E}\,\phi\mathsf{U}\psi$. More formally, we construct a *(definable) weighted finite graph* $\mathcal{G}_{\phi,\psi} = (V,E)$ as follows:

- its set of vertices V is

$$\{(q,P),(q,P,\mathsf{init}) \mid (q,P) \models \phi \vee \psi\}^8 \quad \cup \quad \{(q,P,\mathsf{final}) \mid (q,P) \models \psi\}$$

– its set of edges E is

$$\begin{aligned}&\{(q,P,\mathsf{init}) \xrightarrow{\lambda^{\phi\vee\psi}_{(q,P)\to(q',P')}(y)} (q',P')\} \cup \{(q,P,\mathsf{init}) \xrightarrow{\lambda^{\phi\vee\psi}_{(q,P\leadsto P'')}(y)} (q,P'',\mathsf{final})\}\\ \cup\ &\{(q,P,\mathsf{init}) \xrightarrow{[0]} (q,P,\mathsf{final})\} \cup \{(q,P) \xrightarrow{\kappa^{\phi\vee\psi}_{(q,P)\to(q',P')}} (q',P')\}\\ \cup\ &\{(q,P) \xrightarrow{\kappa^{\phi\vee\psi}_{(q,P)\to(q',P')}} (q',P',\mathsf{final})\} \cup \{(q,P) \xrightarrow{\kappa^{\phi\vee\psi}_{(q,P\leadsto P'')}} (q,P'',\mathsf{final})\}\end{aligned}$$

Let $\varrho = (q_0,P_0,\mathsf{init}) \xrightarrow{\lambda(y)} (q_1,P_1) \xrightarrow{\kappa_1} \ldots \xrightarrow{\kappa_n} (q_n,P_n,\mathsf{final})$ be a finite path of $\mathcal{G}_{\phi,\psi}$. Then we define $K_\varrho(y)$ to be the set of all possible costs of the path ϱ: $K_\varrho(y) = \{\lambda(y) + c_2 + \cdots + c_n \mid c_i \in \kappa_i\}$.

The following proposition shows that the graph $\mathcal{G}_{\phi,\psi}$ can be used to model-check the formula $\mathsf{E}\,\phi\mathsf{U}_{\sim c}\psi$.

Proposition 19. *Let $q_0 \in Q$, $P_0 \in \mathcal{P}$ and $y_0 \in P_0$. Then, $(q_0,y_0) \models \mathsf{E}\,\phi\mathsf{U}_{\sim c}\psi$ iff there exists a path $\varrho = (q_0,P_0,\mathit{init}) \xrightarrow{\lambda(y)} (q_1,P_1) \xrightarrow{\kappa_1} \ldots \xrightarrow{\kappa_n} (q_n,P_n,\mathit{final})$ in $\mathcal{G}_{\phi,\psi}$, there exists $\zeta \in M$ such that $\zeta \in K_\varrho(y_0)$ and $\zeta \sim c$.*

We prove that, under the Archimedian hypothesis, we can bound the length of witnessing paths in the graph $\mathcal{G}_{\phi,\psi}$. This hypothesis has not been used yet, as everything holds without it, but is required by the next proposition. For every $(m,c) \in M^2$ such that $m \neq 0$, we note $\left[\frac{c}{m}\right]$ the smallest integer k such that $k \cdot m > c$.

Proposition 20. *We assume that $(q_0,y_0) \models \mathsf{E}\,\phi\mathsf{U}_{\sim c}\psi$, and we define $N = 2 + (|Q| \cdot |\mathcal{P}| + 1) \cdot \left(\left[\frac{c}{m}\right] + 2\right)$ where m is the smallest positive constant defining an interval of some weight labelling the transitions of $\mathcal{G}_{\phi,\psi}$. Then there exists a path $\varrho = (q_0,P_0,\mathit{init}) \xrightarrow{\lambda(y)} (q_1,P_1) \xrightarrow{\kappa_1} \ldots \xrightarrow{\kappa_n} (q_n,P_n,\mathit{final})$ in $\mathcal{G}_{\phi,\psi}$ such that $y_0 \in P_0$, $n \leq N$, and there exists $\zeta \in M$ such that $\zeta \in K_\varrho(y_0)$ and $\zeta \sim c$.*

Proof (Sketch). By Proposition 19, there exists a path $\varrho = (q_0,P_0,\mathsf{init}) \xrightarrow{\lambda(y)} (q_1,P_1) \xrightarrow{\kappa_1} \ldots \xrightarrow{\kappa_n} (q_n,P_n,\mathsf{final})$ in $\mathcal{G}_{\phi,\psi}$ s.t. $\exists \zeta \in K_\varrho(y_0)$, $\zeta \sim c$. From ϱ, we will construct a run of length smaller than N which witnesses the formula.

By o-minimality, each κ_i is a finite union of intervals. We first choose accurately one such interval per κ_i and obtain a *realisation* of ϱ noted $\varrho[\mathcal{I}]$, so that the accumulated union of all intervals along $\varrho[\mathcal{I}]$ contains ζ such that $\zeta \sim c$. We call [0]-cycle a cycle labeled only by the interval [0]. We can suppose that there is no [0]-cycle in $\varrho[\mathcal{I}]$ otherwise we can remove this cycle and get a shorter path also witnessing the formula.

If the length of $\varrho[\mathcal{I}]$ is shorter than N then we are done. Otherwise ϱ contains at least $\left(\left[\frac{c}{m}\right] + 2\right)$ simple cycles (see Fig. 2), each of which is labeled by an (accumulated) interval whose right bound is greater than m. Thus, $K_{\varrho[\mathcal{I}]}(y_0) = \langle a,b\rangle$ [9] with $b \geq c + 2m$. By hypothesis, there exists $\zeta \in \langle a,b\rangle$ such that $\zeta \sim c$.

[8] This is a notation misuse, but $(q,P) \models \phi \vee \psi$ means that for all $y \in P$, $(q,y) \models \phi \vee \psi$, which is equivalent to the property that $(q,y) \models \phi \vee \psi$ for some $y \in P$.

[9] $\langle a,b\rangle$ stands for either $[a,b]$ or $[a,b)$ or $(a,b]$ or (a,b).

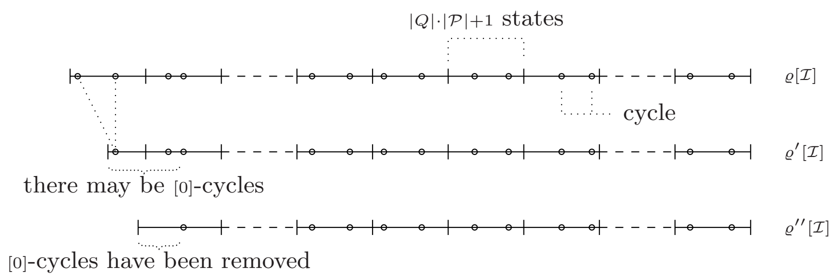


Fig. 2. The construction in the proof of Proposition 20

We then remove the first cycle of the run and remove [0]-cycles that can have been potentially created (see Fig. 2). We obtain $\varrho''[\mathcal{I}]$ a strictly smaller run. Note that $K_{\varrho''[\mathcal{I}]}(y_0)$ might be different from $K_{\varrho[\mathcal{I}]}(y_0)$; nevertheless $\varrho''[\mathcal{I}]$ still has at least $\left[\frac{c}{m}\right]$ simple cycles so $K_{\varrho''[\mathcal{I}]}(y_0) = \langle a', b' \rangle$ with $b' > c$ (as previously). It proves that there exists $\zeta \in K_{\varrho''[\mathcal{I}]}(y_0)$ with $\zeta \sim c$ (since $c < b' \leq b$ and $a' \leq a$). We iterate this process until we find a run smaller than N with this property. □

Applying the previous proposition, we can build a first-order formula which checks if a given state (q, y) satisfies the WCTL formula $\mathsf{E}\,\phi \mathsf{U}_{\sim c}\psi$, and we thus get the following corollary.

Corollary 21. *Let $\mathcal{M} = \langle M, +, 0, 1, <, \dots \rangle$ be an o-minimal structure such that $\mathcal{M}$ is Archimedian and $\mathsf{Th}(\mathcal{M})$ is decidable. Let $\mathcal{H}$ be an $\mathcal{M}$-weighted hybrid system and ϕ and ψ be two WCTL formulae. Assume we have built two finite and definable partitions $\mathcal{P}_\phi$ and $\mathcal{P}_\psi$ for ϕ and ψ, then we can compute a definable finite partition correct for the formula $\mathsf{E}\,\phi \mathsf{U}_{\sim c}\psi$.*

We do not enter into the details of the third step. However, the construction of the partition respecting $\mathsf{A}\,\phi \mathsf{U}_{\sim c}\psi$ (from the partition $\mathcal{P}_\phi$ and $\mathcal{P}_\psi$) shares ideas with the construction of the partition respecting $\mathsf{E}\,\phi \mathsf{U}_{\sim c}\psi$ that we have described. In particular, the idea exploited in the proof of Proposition 20 plays an important role.

References

[ABM04] Rajeev Alur, Mikhail Bernadsky, and Parthasarathy Madhusudan. Optimal reachability for weighted timed games. In *ICALP'04: Automata, Languages, and Programming*, vol. 3142 of *LNCS*, pp. 122–133. Springer, 2004.

[AD94] Rajeev Alur and David Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AHLP00] Rajeev Alur, Thomas A. Henzinger, Gerardo Lafferriere, and George J. Pappas. Discrete abstractions of hybrid systems. *Proc. of the IEEE*, 88:971–984, 2000.

[ALP01] Rajeev Alur, Salvatore La Torre, and George J. Pappas. Optimal paths in weighted timed automata. In *HSCC'01: Hybrid Systems, Computation and Control*, vol. 2034 of *LNCS*, pp. 49–62. Springer, 2001.

[BBBR06] Patricia Bouyer, Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On the optimal reachability problem. Submitted, 2006.

[BBC06] Patricia Bouyer, Thomas Brihaye, and Fabrice Chevalier. Control in o-minimal hybrid systems. In *LICS'06: Logic in Computer Science*, pp. 367–378. IEEE Computer Society Press, 2006.

[BBL04] Patricia Bouyer, Ed Brinksma, and Kim G. Larsen. Staying alive as cheaply as possible. In *HSCC'04: Hybrid Systems, Computation and Control*, vol. 2993 of *LNCS*, pp. 203–218. Springer, 2004.

[BBM06] Patricia Bouyer, Thomas Brihaye, and Nicolas Markey. Improved undecidability results on weighted timed automata. *Information Processing Letters*, 98(5):188–194, 2006.

[BBR04] Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. Model-checking for weighted timed automata. In *FORMATS-FTRTFT'04: Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault-Tolerant Systems*, vol. 3253 of *LNCS*, pp. 277–292. Springer, 2004.

[BBR05] Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On optimal timed strategies. In *FORMATS'05: Formal Modelling and Analysis of Timed Systems*, vol. 3829 of *LNCS*, pp. 49–64. Springer, 2005.

[BBR06] Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On model-checking timed automata with stopwatch observers. *Information and Computation*, 204(3):408–433, 2006.

[BCFL04] Patricia Bouyer, Franck Cassez, Emmanuel Fleury, and Kim G. Larsen. Optimal strategies in priced timed game automata. In *FSTTCS'04: Foundations of Software Technology and Theoretical Computer Science*, vol. 3328 of *LNCS*, pp. 148–160. Springer, 2004.

[BFH$^+$01] Gerd Behrmann, Ansgar Fehnker, Thomas Hune, Kim G. Larsen, Paul Pettersson, Judi Romijn, and Frits Vaandrager. Minimum-cost reachability for priced timed automata. In *HSCC'01: Hybrid Systems, Computation and Control*, vol. 2034 of *LNCS*, pp. 147–161. Springer, 2001.

[BLM07] Patricia Bouyer, Kim G. Larsen, and Nicolas Markey. Model-checking one-clock priced timed automata. In *FoSSaCS'07: Foundations of Software Science and Computation Structures*, vol. 4423 of *LNCS*, pp. 108–122. Springer, 2007.

[BLMR06] Patricia Bouyer, Kim G. Larsen, Nicolas Markey, and Jacob Illum Rasmussen. Almost optimal strategies in one-clock priced timed automata. In *FSTTCS'06: Fundations of Software Technology and Theoretical Computer Science*, vol. 4337 of *LNCS*, pp. 346–357. Springer, 2006.

[BM05] Thomas Brihaye and Christian Michaux. On the expressiveness and decidability of o-minimal hybrid systems. *Journal of Complexity*, 21(4):447–478, 2005.

[BMRT04] Thomas Brihaye, Christian Michaux, Cédric Rivière, and Christophe Troestler. On o-minimal hybrid systems. In *HSCC'04: Hybrid Systems, Computation and Control*, vol. 2993 of *LNCS*, pp. 219–233. Springer, 2004.

[Bri06a] Thomas Brihaye. *Verification and Control of O-Minimal Hybrid Systems and Weighted Timed Automata*. Thèse de doctorat, Université Mons-Hainaut, Belgium, 2006.

[Bri06b] Thomas Brihaye. Words and bisimulation of dynamical systems. *Journal of Automata, Languages and Combinatorics*, 2006. To appear.

[cor06] Uppaal Cora, 2006. `http://www.cs.aau.dk/~behrmann/cora/`.

[Dav99] Jennifer M. Davoren. Topologies, continuity and bisimulations. *Theoretical Informatics and Applications*, 33(4-5):357–382, 1999.

[Gen05] Raffaella Gentilini. Reachability problems on extended o-minimal hybrid automata. In *FORMATS'05: Formal Modeling and Analysis of Timed Systems*, vol. 3829 of *LNCS*, pp. 162–176. Springer, 2005.

[Hen95] Thomas A. Henzinger. Hybrid automata with finite bisimulations. In *ICALP'95: Automata, Languages, and Programming*, vol. 944 of *LNCS*, pp. 324–335. Springer-Verlag, 1995.

[Hen96] Thomas A. Henzinger. The theory of hybrid automata. In *LICS'96: Logic in Computer Science*, pp. 278–292. IEEE Computer Society Press, 1996.

[HMR05] Thomas A. Henzinger, Rupak Majumdar, and Jean-François Raskin. A classification of symbolic transition systems. *ACM Transactions on Computational Logic*, 6(1):1–32, 2005.

[Hod97] Wilfrid Hodges. *A Shorter Model Theory*. Cambridge University Press, 1997.

[KV06] Margarita V. Korovina and Nicolai Vorobjov. Upper and lower bounds on sizes of finite bisimulations of Pfaffian hybrid systems. In *CiE*, vol. 3988 of *Lecture Notes in Computer Science*, pp. 267–276. Springer, 2006.

[LMM02] Salvatore La Torre, Supratik Mukhopadhyay, and Aniello Murano. Optimal-reachability and control for acyclic weighted timed automata. In *TCS'02: Theoretical Computer Science*, vol. 223 of *IFIP Conf. Proc.*, pp. 485–497. Kluwer, 2002.

[LPS00] Gerardo Lafferriere, George J. Pappas, and Shankar Sastry. O-minimal hybrid systems. *Mathematics of Control, Signals, and Systems*, 13(1):1–21, 2000. Appeared as a preprint in 1998.

[PS86] Anand Pillay and Charles Steinhorn. Definable sets in ordered structures. *Transactions of the American Mathematical Society*, 295(2):565–592, 1986.

[Ras99] Jean-François Raskin. *Logics, Automata and Classical Theories for Deciding Real-Time*. Thèse de doctorat, Université Namur, Belgium, 1999.

[vdD98] Lou van den Dries. *Tame Topology and O-Minimal Structures*, vol. 248 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1998.

[Wil96] Alex J. Wilkie. Model completeness results for expansions of the ordered field of real numbers by restricted Pfaffian functions and the exponential function. *Journal of the AMS*, 9(4):1051–1094, 1996.

On Decidability and Expressiveness of Propositional Interval Neighborhood Logics

Davide Bresolin[1,⋆], Valentin Goranko[2],
Angelo Montanari[1], and Guido Sciavicco[3]

[1] Department of Mathematics and Computer Science,
University of Udine, Udine, Italy
[2] School of Mathematics, University of the Witwatersrand,
Johannesburg, South Africa
[3] Department of Information Engineering and Communications,
University of Murcia, Murcia, Spain
{bresolin,montana}@dimi.uniud.it, goranko@maths.wits.ac.za, guido@um.es

Abstract. Interval-based temporal logics are an important research area in computer science and artificial intelligence. In this paper we investigate decidability and expressiveness issues for Propositional Neighborhood Logics (PNLs). We begin by comparing the expressiveness of the different PNLs. Then, we focus on the most expressive one, namely, $\mathrm{PNL}^{\pi+}$, and we show that it is decidable over various classes of linear orders by reducing its satisfiability problem to that of the two-variable fragment of first-order logic with binary relations over linearly ordered domains, due to Otto. Next, we prove that $\mathrm{PNL}^{\pi+}$ is expressively complete with respect to such a fragment. We conclude the paper by comparing $\mathrm{PNL}^{\pi+}$ expressiveness with that of other interval-based temporal logics.

Keywords: neighbourhood interval logics, decidability, expressiveness.

1 Introduction

Interval-based temporal logics over ordered domains are an important research area in various fields of computer science and artificial intelligence. Unfortunately, even when restricted to the case of *propositional languages* and *linear time*, they usually exhibit a bad computational behavior where undecidability rules. The main species of studied propositional interval temporal logics include Moszkowski's Propositional Interval Logic (PITL) [17], Halpern and Shoham's modal logic of time intervals (HS) [12], Venema's CDT logic [22] (extended to branching-time frames with linear intervals by Goranko, Montanari and Sciavicco [9]), Lodaya's Begins/Ends fragment of HS (BE) [15], and Montanari, Goranko and Sciavicco's Propositional Neighborhood Logics [7]. Many expressiveness and (un)decidability results for these logics substantially depend on the assumptions about the class of frames over which they are interpreted. Typical classes of frames are the class of all (resp., dense, discrete, Dedekind complete)

⋆ Current affiliation: Dept. of Computer Science, University of Verona, Verona, Italy.

linear frames, and of specific linear orders such as $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$, and $\mathbb{R}$. Basic results about these logics are the undecidability of HS and CDT over most of classes of frames, of PITL over dense and discrete frames, and of BE over dense frames. A comprehensive survey of the main developments, results, and open problems in the area of propositional interval temporal logics can be found in [8].

In this paper we focus our attention on expressiveness and decidability issues for Propositional Neighborhood Logics (PNLs) over various classes of linear orders. PNLs are fragments of HS which feature two modalities, corresponding to Allen's relations *meets* and *met by*, and (possibly) the interval operator π. Sound and complete axiomatic systems for PNLs and a tableau-based semi-decision procedure for them have been developed in [7]. A tableau-based decision procedure for the future fragments of PNLs, interpreted over $\mathbb{N}$, together with a proof of NEXPTIME-completeness, have been given in [2,4] and later extended to full PNLs over $\mathbb{Z}$ [3]. To the best of our knowledge, these are the first non-trivial decidability results for propositional interval logics interpreted over *fully-instantiated* temporal structures, that is, temporal structures containing all intervals that can be built up from a given linearly ordered set of points, which do not resort to any *projection principle*, such as locality or homogeneity [12].

Here is a summary of the paper. First, we compare the expressive power of three PNLs, namely, $\text{PNL}^{\pi+}$, PNL^{+}, and PNL^{-}, and we show that $\text{PNL}^{\pi+}$ is strictly more expressive than PNL^{+} and PNL^{-}. Then, we prove that the satisfiability problem for $\text{PNL}^{\pi+}$ over the classes of all linear orders, of all well-orders, and of all finite linear orders, can be decided in NEXPTIME by reducing it to the satisfiability problem for the two-variable fragment of first-order logic over the same classes of structures [18]. Next, we focus on expressive completeness, in the spirit of Kamp's theorem [14]. Kamp proved the functional completeness of the *Since* (S) and *Until* (U) temporal logic with respect to first-order definable connectives over Dedekind-complete linear orders. This result has been later re-proved and generalized in several ways (see [13,6]). In particular, Stavi extended Kamp's result to the class of all linear orders by adding the binary operators S' and U' (see [6] for details), while Etessami et al. proved the functional completeness of the *future* (F) and *past* (P) temporal logic (TL[F,P] for short) with respect to the monadic two-variable fragment of first-order logic $\text{MFO}^2[<]$ over $\mathbb{N}$ [5]. As for interval-based logics, Venema showed the functional completeness of CDT with respect to the three-variable (with at most two of them free) fragment of first-order logic $\text{FO}^3_{x,y}[<]$ over all linear orders. In this paper we prove the expressive completeness of $\text{PNL}^{\pi+}$ with respect to the full two-variable fragment of first-order logic over various classes of linear orders. We conclude the paper with a comparison of $\text{PNL}^{\pi+}$ expressive power with that of other HS fragments.

2 Basics

Propositional Neighborhood Logics. The syntax and semantics of propositional neighborhood logics (PNLs for short), interpreted over linear orders, are defined as follows. Let $\mathbb{D} = \langle D, < \rangle$ be a linearly ordered set. An *interval* over $\mathbb{D}$ is an ordered pair $[a, b]$, where $a, b \in D$ and $a \leq b$. An interval $[a, b]$ is a *strict interval* if $a < b$, while it is a *point interval* if $a = b$. We denote the set of all (resp., strict) intervals over $\mathbb{D}$ by $\mathbb{I}(\mathbb{D})^+$ (resp., $\mathbb{I}(\mathbb{D})^-$). The language of *Full Propositional Neighborhood Logic* ($\text{PNL}^{\pi+}$) consists of a set $\mathcal{AP}$ of propositional letters, the propositional connectives $\neg, \vee$, the modal constant π, and the modal operators $\Diamond_r$ and $\Diamond_l$. The other propositional connectives, as well as the logical constants $\top$ (*true*) and $\bot$ (*false*) and the dual modal operators $\Box_r$ and $\Box_l$, are defined as usual. *Formulas* of $\text{PNL}^{\pi+}$, denoted by $\varphi, \psi, \ldots$, are recursively defined by the following grammar: $\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \pi \mid \Diamond_r\varphi \mid \Diamond_l\varphi$. The language of *Non-strict Propositional Neighborhood Logic* (PNL^+) is the fragment of $\text{PNL}^{\pi+}$ devoid of the modal constant π, while the language of *Strict Propositional Neighborhood Logic* (PNL^-) is obtained from that of PNL^+ by replacing the modalities $\Diamond_r$ and $\Diamond_l$ with the modalities $\langle A\rangle$ and $\langle\overline{A}\rangle$ (with dual modalities $[A]$ and $[\overline{A}]$), respectively. We adopt different notations for the modalities of $\text{PNL}^{\pi+}$ /PNL^+ and PNL^- to reflect their historical links and to make it easier to distinguish between their non-strict/strict semantics from the syntax. We will write PNLs when referring to either $\text{PNL}^{\pi+}$, PNL^+, or PNL^-. The semantics of $\text{PNL}^{\pi+}$ /PNL^+ is given in terms of *non-strict interval models* $\mathbf{M}^+ = \langle\mathbb{I}(\mathbb{D})^+, V^+\rangle$, while that of PNL^- is given in terms of *strict interval models* $\mathbf{M}^- = \langle\mathbb{I}(\mathbb{D})^-, V^-\rangle$. The *valuation function* $V^+ : \mathcal{AP} \mapsto 2^{\mathbb{I}(\mathbb{D})^+}$ (resp., $V^- : \mathcal{AP} \mapsto 2^{\mathbb{I}(\mathbb{D})^-}$) assigns to every propositional variable p the set of (all, resp. strict) intervals $V(p)$ over which p holds. Instead of V^+ and V^-, we will write just V whenever there is no risk of confusion; likewise we will write $\mathbb{I}(\mathbb{D})$ for either $\mathbb{I}(\mathbb{D})^+$ or $\mathbb{I}(\mathbb{D})^-$. Note that for every p, $V(p)$ can be viewed as a binary relation on D, and we will use that later on. When referring to either the strict or the non-strict interval model, we will use $\mathbf{M}$. The *truth relation* of a formula at a given interval in a model $\mathbf{M}$ is defined by structural induction on formulas:

- $\mathbf{M}, [a, b] \Vdash p$ iff $[a, b] \in V(p)$, for all $p \in \mathcal{AP}$;
- $\mathbf{M}, [a, b] \Vdash \neg\psi$ iff it is not the case that $\mathbf{M}, [a, b] \Vdash \psi$;
- $\mathbf{M}, [a, b] \Vdash \varphi \vee \psi$ iff $\mathbf{M}, [a, b] \Vdash \varphi$ or $\mathbf{M}, [a, b] \Vdash \psi$;
- $\mathbf{M}, [a, b] \Vdash \Diamond_r\psi$ (resp., $\langle A\rangle\psi$) iff there exists c such that $c \geq b$ (resp., $c > b$) and $\mathbf{M}, [b, c] \Vdash \psi$;
- $\mathbf{M}, [a, b] \Vdash \Diamond_l\psi$ (resp., $\langle\overline{A}\rangle\psi$) iff there exists c such that $c \leq a$ (resp., $c < a$) and $\mathbf{M}, [c, a] \Vdash \psi$;
- $\mathbf{M}^+, [a, b] \Vdash \pi$ iff $a = b$.

A formula is *satisfiable* if it is true over some interval in some interval model (for the respective language) and it is *valid* if it is true over every interval in every interval model. As shown in [7], PNLs are powerful enough to express interesting temporal properties, e.g., they allow one to constrain the structure of the underlying linear ordering. In particular, $\text{PNL}^{\pi+}$ and PNL^- allow one to express the *difference* operator and thus to simulate *nominals.*

The two-variable fragment of first-order logic. In this section we give some basic definitions about fragments of first-order logic. Let us denote by FO^2 (resp., $\text{FO}^2[=]$) the fragment of first-order logic (resp., first-order logic with equality) whose language uses only two distinct (possibly reused) variables. We denote its formulas by $\alpha, \beta, \ldots$. For example, the formula $\forall x(P(x) \rightarrow \forall y \exists x Q(x,y))$ belongs to FO^2, while the formula $\forall x(P(x) \rightarrow \forall y \exists z(Q(z,y) \wedge Q(z,x)))$ does not. We focus our attention on the logic $\mathrm{FO}^2[<]$ over a purely relational vocabulary $\{=, <, P, Q, \ldots\}$ including equality and a distinguished binary relation $<$ interpreted as a linear ordering. Since atoms in the two-variable fragment can involve at most two distinct variables, we may further assume without loss of generality that the arity of every relation is exactly 2.

Let x and y be the two variables of the language. The formulas of $\mathrm{FO}^2[<]$ can be defined recursively as follows:

$$\begin{aligned}
\alpha &::= A_0 \mid A_1 \mid \neg\alpha \mid \alpha \vee \beta \mid \exists x \alpha \mid \exists y \alpha \\
A_0 &::= x = x \mid x = y \mid y = x \mid y = y \mid x < y \mid y < x \\
A_1 &::= P(x,x) \mid P(x,y) \mid P(y,x) \mid P(y,y),
\end{aligned}$$

where A_1 deals with (uninterpreted) binary predicates. For technical convenience, we assume that both variables x and y occur as (possibly vacuous) free variables in every formula $\alpha \in \mathrm{FO}^2[<]$, that is, $\alpha = \alpha(x,y)$.

Formulas of $\mathrm{FO}^2[<]$ are interpreted over *relational models* of the form $\mathcal{A} = \langle \mathbb{D}, V_{\mathcal{A}} \rangle$, where $\mathbb{D}$ is a linear ordering and $V_{\mathcal{A}}$ is a *valuation function* that assigns to every *binary relation* P a subset of $D \times D$. When we evaluate a formula $\alpha(x,y)$ on a pair of elements a, b, we write $\alpha(a,b)$ for $\alpha[x := a, y := b]$.

The satisfiability problem for FO^2 without equality was proved decidable by Scott [19] by a satisfiability preserving reduction of any FO^2-formula to a formula of the form $\forall x \forall y \psi_0 \wedge \bigwedge_{i=1}^{m} \forall x \exists y \psi_i$, which belongs to the Gödel's prefix-defined decidable class of first-order formulas [1]. Later, Mortimer extended this result by including equality in the language [16]. More recently, Grädel, Kolaitis, and Vardi improved Mortimer's result by lowering the complexity bound [11]. Finally, by building on techniques from [11] and taking advantage of an in-depth analysis of the basic 1-types and 2-types in $\mathrm{FO}^2[<]$-models, Otto proved the decidability of $\mathrm{FO}^2[<]$ over the class of all linear orderings as well as on some natural subclasses of it [18].

Theorem 1 ([18]). *The satisfiability problem for formulas in* $\mathrm{FO}^2[<]$ *is decidable in NEXPTIME on each of the classes of structures where* $<$ *is interpreted as (i) any linear ordering, (ii) any well-ordering, (iii) any finite linear ordering, and (iv) the linear ordering on* $\mathbb{N}$.

Comparing the expressive power of interval logics. In the following we will compare the expressive power of $\mathrm{PNL}^{\pi+}$ with that of PNL^{+} and PNL^{-} as well as with that of other classical/temporal logics. There are several ways to compare the expressive power of different modal languages/logics, e.g., they can be compared with respect to frame validity, that is, with respect to the properties of frames that they can express (such a comparison for PNLs has been done in [7]). Here we compare the considered logics with respect to truth at a given element of a model. We distinguish three different cases: the case in which we compare two interval logics over the same class of models, e.g., $\mathrm{PNL}^{\pi+}$ and PNL^{+}, the case in which we compare strict and non-strict interval logics, e.g., PNL^{-} and $\mathrm{PNL}^{\pi+}$, and the case in which we compare an interval logic with a first-order logic, e.g., $\mathrm{PNL}^{\pi+}$ and $\mathrm{FO}^2[<]$.

Given two interval logics L and L' interpreted over the same class of models $\mathcal{C}$, we say that L' is *at least as expressive as* L (with respect to $\mathcal{C}$), denoted by $\mathrm{L} \preceq_{\mathcal{C}} \mathrm{L}'$ ($\mathcal{C}$ is omitted if clear from the context), if there exists an effective translation τ from L to L' (inductively defined on the structure of formulas) such that for every model $\mathbf{M}$ in $\mathcal{C}$, any interval $[a,b]$ in $\mathbf{M}$, and any formula φ of L, $\mathbf{M},[a,b] \Vdash \varphi$ iff $\mathbf{M},[a,b] \Vdash \tau(\varphi)$. Furthermore, we say that L is *as expressive as* L', denoted by $\mathrm{L} \equiv_{\mathcal{C}} \mathrm{L}'$, if both $\mathrm{L} \preceq_{\mathcal{C}} \mathrm{L}'$ and $\mathrm{L}' \preceq_{\mathcal{C}} \mathrm{L}$, while we say that L is *strictly more expressive than* L', denoted by $\mathrm{L}' \prec_{\mathcal{C}} \mathrm{L}$, if $\mathrm{L}' \preceq_{\mathcal{C}} \mathrm{L}$ and $\mathrm{L} \not\preceq_{\mathcal{C}} \mathrm{L}'$.

When comparing an interval logic L^- interpreted over *strict* interval models with an interval logic L^+ interpreted over *non-strict* ones, we need to slightly revise the above definitions. Given a strict interval model $\mathbf{M}^- = \langle \mathbb{I}(\mathbb{D})^-, V^- \rangle$, we say that a non-strict interval model $\mathbf{M}^+ = \langle \mathbb{I}(\mathbb{D})^+, V^+ \rangle$ is *a non-strict extension* of $\mathbf{M}^-$ (and that $\mathbf{M}^-$ is *the strict restriction* of $\mathbf{M}^+$) if V^- and V^+ agree on the valuation of strict intervals, that is, if for every strict interval $[a,b] \in \mathbb{I}(\mathbb{D})^-$ and propositional letter $p \in \mathcal{AP}$, $[a,b] \in V^-(p)$ if and only if $[a,b] \in V^+(p)$. We say that L^+ is *at least as expressive as* L^-, and we denote it by $\mathrm{L}^- \preceq_I \mathrm{L}^+$, if there exists an effective translation τ from L^- to L^+ such that for any strict interval model $\mathbf{M}^-$, any interval $[a,b]$ in $\mathbf{M}^-$, and any formula φ of L^-, $\mathbf{M}^-,[a,b] \Vdash \varphi$ iff $\mathbf{M}^+,[a,b] \Vdash \tau(\varphi)$ for every non-strict extension $\mathbf{M}^+$ of $\mathbf{M}^-$. Conversely, we say that L^- is *at least as expressive as* L^+, and we denote it by $\mathrm{L}^+ \preceq_I \mathrm{L}^-$, if there exists an effective translation τ' from L^+ to L^- such that for any non-strict interval model $\mathbf{M}^+$, any strict interval $[a,b]$ in $\mathbf{M}^+$, and any formula φ of L^+, $\mathbf{M}^+,[a,b] \Vdash \varphi$ iff $\mathbf{M}^-,[a,b] \Vdash \tau'(\varphi)$, where $\mathbf{M}^-$ is the strict restriction of $\mathbf{M}^+$. $\mathrm{L}^- \equiv_I \mathrm{L}^+$, $\mathrm{L}^- \prec_I \mathrm{L}^+$, and $\mathrm{L}^+ \prec_I \mathrm{L}^-$ are defined in the usual way.

Finally, we compare interval logics with first-order logics interpreted over relational models. In this case, the above criteria are no longer adequate, since we need to compare logics which are interpreted over different types of models (interval models and relational models). We deal with this complication by following the approach outlined by Venema in [22]. First, we define suitable model transformations (from interval models to relational models and vice versa); then, we compare the expressiveness of interval and first-order logics modulo these transformations. To define the mapping from interval models to relational models, we associate a binary relation P with every propositional variable $p \in \mathcal{AP}$ of the considered interval logic [22].

Definition 1. *Given an interval model* $\mathbf{M} = \langle \mathbb{I}(\mathbb{D}), V_{\mathbf{M}} \rangle$*, the corresponding relational model* $\eta(\mathbf{M})$ *is a pair* $\langle \mathbb{D}, V_{\eta(\mathbf{M})} \rangle$*, where for all* $p \in \mathcal{AP}$*,* $V_{\eta(\mathbf{M})}(P) = \{(a,b) \in D \times D : [a,b] \in V_{\mathbf{M}}(p)\}$*.*

As a matter of fact, the above relational models can be viewed as 'point' models for logics over $\mathbb{D}^2$ and the above transformation as a mapping of propositional letters of the interval logic, interpreted over $\mathbb{I}(\mathbb{D})$, into propositional letters of the target logic, interpreted over $\mathbb{D}^2$ [21,20].

To define the mapping from relational models to interval ones, we have to solve a technical problem: the truth of formulas in interval models is evaluated only on ordered pairs $[a, b]$, with $a \leq b$, while in relational models there is not such a constraint. To deal with this problem, we associate two propositional letters $p^{\leq}$ and $p^{\geq}$ of the interval logic with every binary relation P.

Definition 2. *Given a relational model $\mathcal{A} = \langle \mathbb{D}, V_{\mathcal{A}} \rangle$, the corresponding non-strict interval model $\zeta(\mathcal{A})$ is a pair $\langle \mathbb{I}(\mathbb{D})^{+}, V_{\zeta(\mathcal{A})} \rangle$ such that for any binary relation P and any interval $[a, b]$, $[a, b] \in V_{\zeta(\mathcal{A})}(p^{\leq})$ iff $(a, b) \in V_{\mathcal{A}}(P)$ and $[a, b] \in V_{\zeta(\mathcal{A})}(p^{\geq})$ iff $(b, a) \in V_{\mathcal{A}}(P)$.*

Given an interval logic L_I and a first-order logic L_{FO}, we say that L_{FO} is *at least as expressive as* L_I, denoted by $\mathrm{L}_I \preceq_R \mathrm{L}_{FO}$, if there exists an effective translation τ from L_I to L_{FO} such that for any interval model $\mathbf{M}$, any interval $[a, b]$, and any formula φ of L_I, $\mathbf{M}, [a, b] \Vdash \varphi$ iff $\eta(\mathbf{M}) \models \tau(\varphi)(a, b)$. Conversely, we say that L_I is *at least as expressive as* L_{FO}, denote by $\mathrm{L}_{FO} \preceq_R \mathrm{L}_I$, if there exists an effective translation τ' from L_{FO} to L_I such that for any relational model $\mathcal{A}$, any pair (a, b) of elements, and any formula φ of L_{FO}, $\mathcal{A} \models \varphi(a, b)$ iff $\zeta(\mathcal{A}), [a, b] \Vdash \tau'(\varphi)$ if $a \leq b$ or $\zeta(\mathcal{A}), [b, a] \Vdash \tau'(\varphi)$ otherwise. We say that L_I is *as expressive as* L_{FO}, denoted by $\mathrm{L}_I \equiv_R \mathrm{L}_{FO}$, if $\mathrm{L}_I \preceq_R \mathrm{L}_{FO}$ and $\mathrm{L}_{FO} \preceq_R \mathrm{L}_I$. $\mathrm{L}_I \prec_R \mathrm{L}_{FO}$ and $\mathrm{L}_{FO} \prec_R \mathrm{L}_I$ are defined in the usual way.

3 PNL$^{\pi+}$, PNL^{+}, and PNL^{-} Expressiveness

In this section we compare the relative expressive power of PNL$^{\pi+}$, PNL^{+}, and PNL^{-}. The comparison of the expressive power of PNL$^{\pi+}$ and PNL^{+} is based on an application of the bisimulation game for modal logics [10]. More precisely, we exploit a game-theoretic argument to show that there exist two models that can be distinguished by a PNL$^{\pi+}$ formula, but not by a PNL^{+} formula. To this end, we define the notion of *k-round* PNL^{+}- *bisimulation game* to be played on a pair of PNL^{+} models $(\mathbf{M_0}^{+}, \mathbf{M_1}^{+})$, with $\mathbf{M_0}^{+} = \langle \mathbb{I}(\mathbb{D}_0)^{+}, V_0 \rangle$ and $\mathbf{M_1}^{+} = \langle \mathbb{I}(\mathbb{D}_1)^{+}, V_1 \rangle$, which starts from a given *initial configuration*, where a configuration is a pair of intervals $([a_0, b_0], [a_1, b_1])$, with $[a_0, b_0] \in \mathbb{I}(\mathbb{D}_0)^{+}$ and $[a_1, b_1] \in \mathbb{I}(\mathbb{D}_1)^{+}$. The game is played by two players, Player I and Player II. If after any given round the current position is not a local isomorphism between the submodels of $\mathbf{M_0}^{+}$ and $\mathbf{M_1}^{+}$ induced by the corresponding configuration, Player I wins the game; otherwise, Player II wins. At every round, given a current configuration $([a_0, b_0], [a_1, b_1])$, Player I plays one of the following two moves:

$\Diamond_r$**-move:** Player I chooses $\mathbf{M_i}^{+}$, with $i \in \{0, 1\}$, and an interval $[b_i, c_i]$;
$\Diamond_l$**-move:** Player I chooses $\mathbf{M_i}^{+}$, with $i \in \{0, 1\}$, and an interval $[c_i, a_i]$.

In the first case, Player II replies by choosing an interval $[b_{1-i}, c_{1-i}]$, which leads to the new configuration $([b_0, c_0], [b_1, c_1])$; in the other case, Player II chooses an interval $[c_{1-i}, a_{1-i}]$, which leads to the new configuration $([c_0, a_0], [c_1, a_1])$.

Roughly speaking, Player II has a *winning strategy* in the k-round PNL$^+$-bisimulation game on the models $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$ with a given initial configuration if she can win regardless of the moves played by Player I; otherwise, Player I has a winning strategy. A formal definition of winning strategy can be found in [10]. The following key property of the k-round PNL$^+$-bisimulation game directly follows from standard results for bisimulation games in modal logics [10].

Proposition 1. *Let $\mathcal{P}$ be a finite set of propositional letters. For all $k \geq 0$, Player II has a winning strategy in the k-round* PNL$^+$*-bisimulation game on $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$, with initial configuration $([a_0, b_0], [a_1, b_1])$, iff $[a_0, b_0]$ and $[a_1, b_1]$ satisfy the same* PNL$^+$*-formulas over $\mathcal{P}$ with operator depth at most k.*

We exploit Proposition 1 to prove that the π operator of PNL$^{\pi+}$ cannot be expressed in PNL$^+$. We choose two models $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$ that can be distinguished with a PNL$^{\pi+}$ formula which makes an essential use of π, but not by a PNL$^+$ formula. The claim is proved by showing that for all k, Player II has a winning strategy in the k-round PNL$^+$-bisimulation game on $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$.

Theorem 2. *The interval operator π cannot be defined in* PNL$^+$*.*

Proof. Let $\mathbf{M}^+ = \langle \mathbb{I}(\mathbb{Z})^+, V \rangle$, where V is such that p holds everywhere, be a non-strict model. Consider the k-round PNL$^+$-bisimulation game on $(\mathbf{M}^+, \mathbf{M}^+)$ with initial configuration $([0, 1], [1, 1])$. The intervals $[0, 1]$ and $[1, 1]$ can be easily distinguished in PNL$^{\pi+}$, since π holds in $[1, 1]$ but not in $[0, 1]$. We show that this pair of intervals cannot be distinguished in PNL$^+$ by providing a simple winning strategy for Player II in the k-round PNL$^+$-bisimulation game on $(\mathbf{M}^+, \mathbf{M}^+)$ with initial configuration $([0, 1], [1, 1])$, as follows: if Player I plays a $\Diamond_r$-move on a given structure, then Player II arbitrarily chooses a right-neighbor of the current interval on the other structure. Likewise, if Player I plays a $\Diamond_l$-move on a given structure, then Player II arbitrarily chooses a left-neighbor of the current interval on the other structure. Since the valuation V is such that p holds everywhere, in any case the new configuration is a local isomorphism. $\square$

The next theorem shows that PNL$^-$ is strictly less expressive than PNL$^{\pi+}$.

Theorem 3. PNL$^- \prec_I$ PNL$^{\pi+}$.

Proof. We prove the claim by showing that PNL$^- \preceq_I$ PNL$^{\pi+}$ and PNL$^{\pi+} \not\preceq_I$ PNL$^-$. To prove the former, we provide a translation τ from PNL$^-$ to PNL$^{\pi+}$. Consider the mapping τ_0 defined as follows:

$$\begin{aligned} \tau_0(p) &= p & \tau_0(\langle A \rangle \varphi) &= \Diamond_r(\neg\pi \wedge \tau_0(\varphi)) \\ \tau_0(\neg\varphi) &= \neg\tau_0(\varphi) & \tau_0(\langle \overline{A} \rangle \varphi) &= \Diamond_l(\neg\pi \wedge \tau_0(\varphi)) \\ \tau_0(\varphi_1 \vee \varphi_2) &= \tau_0(\varphi_1) \vee \tau_0(\varphi_2) \end{aligned}$$

For every PNL$^-$-formula φ, let $\tau(\varphi) = \neg\pi \wedge \tau_0(\varphi)$. Given a strict model $\mathbf{M}^- = \langle \mathbb{I}(\mathbb{D})^-, V^- \rangle$, let $\mathbf{M}^+ = \langle \mathbb{I}(\mathbb{D})^+, V^+ \rangle$ be a non-strict extension of $\mathbf{M}^-$. It is immediate to show that for any interval $[a, b]$ in $\mathbf{M}^-$ and any PNL$^-$-formula φ,

$\mathbf{M}^-, [a,b] \Vdash \varphi$ if and only if $\mathbf{M}^+, [a,b] \Vdash \tau(\varphi)$. The proof is an easy induction on the structure of φ. This proves that $\mathrm{PNL}^- \preceq_I \mathrm{PNL}^{\pi+}$.

To prove that $\mathrm{PNL}^{\pi+} \not\preceq_I \mathrm{PNL}^-$, suppose by contradiction that there exists a translation τ' from $\mathrm{PNL}^{\pi+}$ to PNL^- such that, for any non-strict model $\mathbf{M}^+$, any strict interval $[a,b]$, and any formula φ of $\mathrm{PNL}^{\pi+}$, $\mathbf{M}^+, [a,b] \Vdash \varphi$ iff $\mathbf{M}^-, [a,b] \Vdash \tau'(\varphi)$, where $\mathbf{M}^-$ is the strict restriction of $\mathbf{M}^+$. Consider the non-strict models $\mathbf{M}_0^+ = \langle \mathbb{I}(\mathbb{Z})^+, V_0 \rangle$ and $\mathbf{M}_1^+ = \langle \mathbb{I}(\mathbb{Z})^+, V_1 \rangle$, where $V_0(p) = \{[a,b] \in \mathbb{I}(\mathbb{Z})^+ : a \leq b\}$ and $V_1(p) = \{[a,b] \in \mathbb{I}(\mathbb{Z})^+ : a < b\}$. It is immediate to see that $\mathbf{M}_0^+, [0,1] \Vdash \Box_r p$, while $\mathbf{M}_1^+, [0,1] \not\Vdash \Box_r p$. Let $\mathbf{M}^- = \langle \mathbb{I}(\mathbb{Z})^-, V^- \rangle$ be a strict interval model such that p holds everywhere in $\mathbb{I}(\mathbb{Z})^-$. We have that $\mathbf{M}^-$ is the strict restriction of both $\mathbf{M}_0^+$ and $\mathbf{M}_1^+$. Hence, we may conclude that $\mathbf{M}^-, [0,1] \Vdash \tau'(\Box_r p)$ and $\mathbf{M}^-, [0,1] \not\Vdash \tau'(\Box_r p)$, which is a contradiction. □

Finally, we show that neither $\mathrm{PNL}^+ \preceq_I \mathrm{PNL}^-$ nor $\mathrm{PNL}^- \preceq_I \mathrm{PNL}^+$.

Theorem 4. *The expressive powers of* PNL^+ *and* PNL^- *are incomparable, namely,* $\mathrm{PNL}^- \not\preceq_I \mathrm{PNL}^+$ *and* $\mathrm{PNL}^+ \not\preceq_I \mathrm{PNL}^-$.

Proof. We first prove that $\mathrm{PNL}^- \not\preceq_I \mathrm{PNL}^+$. Let $\mathbf{M_0}^+ = \langle \mathbb{I}(\mathbb{Z})^+, V_0 \rangle$ and $\mathbf{M_1}^+ = \langle \mathbb{I}(\mathbb{Z} \setminus \{2\})^+, V_1 \rangle$, where V_0 is such that $V_0(p) = \{[1,1],[1,2],[2,2]\}$ and V_1 is such that $V_1(p) = \{[1,1]\}$, be two PNL^+-models. For any $k \geq 0$, consider the k-round PNL^+-bisimulation game between $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$, with initial configuration $([0,1],[0,1])$. Player II has the following winning strategy: at any round, if Player I chooses an interval $[a,b] \in \mathbb{I}(\mathbb{Z} \setminus \{2\})^+$ in one of the models, then Player II chooses the same interval on the other model, while if Player I chooses an interval $[a,2]$ (resp., $[2,b]$) in $\mathbf{M_0}^+$, then Player II chooses the interval $[a,1]$ (resp., $[1,b]$) in $\mathbf{M_1}^+$. On the contrary, the strict restrictions $\mathbf{M_0}^-$ of $\mathbf{M_0}^+$ and $\mathbf{M_1}^-$ of $\mathbf{M_1}^+$ can be easily distinguished by PNL^-: we have that $\mathbf{M_0}^-, [0,1] \Vdash \langle A \rangle p$, while $\mathbf{M_1}^-, [0,1] \not\Vdash \langle A \rangle p$. Since $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$ satisfy the same formulas over the interval $[0,1]$, there cannot exist a translation τ' from PNL^- to PNL^+ such that $\mathbf{M_0}^+, [0,1] \Vdash \tau'(\langle A \rangle p)$ and $\mathbf{M_1}^+, [0,1] \not\Vdash \tau'(\langle A \rangle p)$.

As for $\mathrm{PNL}^+ \not\preceq_I \mathrm{PNL}^-$, we can exploit the very same proof we gave to show that $\mathrm{PNL}^{\pi+} \not\preceq_I \mathrm{PNL}^-$ (it suffices to notice that $\Box_r p$ is a PNL^+ formula). □

4 Decidability of PNLs

In this section we prove the decidability of $\mathrm{PNL}^{\pi+}$, and thus that of its proper fragments PNL^+ and PNL^-, by embedding it into the two-variable fragment of first-order logic interpreted over linearly ordered domains. $\mathrm{PNL}^{\pi+}$ can be translated into $\mathrm{FO}^2[<]$ as follows. Let $\mathcal{AP}$ be the set of propositional letters in $\mathrm{PNL}^{\pi+}$. The signature for $\mathrm{FO}^2[<]$ includes a binary relational symbol P for every $p \in \mathcal{AP}$. The translation function $ST_{x,y}$ is defined as follows:

$$ST_{x,y}(\varphi) = x \leq y \wedge ST'_{x,y}(\varphi),$$

where x, y are two first-order variables and

$$\begin{array}{rr} ST'_{x,y}(p) = P(x,y) & ST'_{x,y}(\varphi \vee \psi) = ST'_{x,y}(\varphi) \vee ST'_{x,y}(\psi) \\ ST'_{x,y}(\pi) = (x = y) & ST'_{x,y}(\Diamond_r \varphi) = \exists x(y \leq x \wedge ST'_{y,x}(\varphi)) \\ ST'_{x,y}(\neg\varphi) = \neg ST'_{x,y}(\varphi) & ST'_{x,y}(\Diamond_l \varphi) = \exists y(y \leq x \wedge ST'_{y,x}(\varphi)) \end{array}$$

Two variables are thus sufficient to translate $\mathrm{PNL}^{\pi+}$ into $\mathrm{FO}^2[<]$. As we will show later, this is not the case with other interval temporal logics, such as, for instance, HS and CDT. The next theorem proves that that $\mathrm{FO}^2[<]$ is at least as expressive as $\mathrm{PNL}^{\pi+}$ (η is the model transformation defined in Section 2).

Theorem 5. *For any* $\mathrm{PNL}^{\pi+}$*-formula* φ*, any non-strict interval model* $\mathbf{M}^+ = \langle \mathbb{I}(\mathbb{D})^+, V\rangle$*, and any interval* $[a, b]$ *in* $\mathbf{M}^+$*:*

$$\mathbf{M}^+, [a,b] \Vdash \varphi \textit{ iff } \eta(\mathbf{M}^+) \models ST_{x,y}(\varphi)[x := a, y := b].$$

Proof. The proof is by structural induction on φ. The base case, as well as the cases of Boolean connectives, are straightforward, and thus omitted. Let $\varphi = \Diamond_r \psi$. From $\mathbf{M}^+, [a, b] \Vdash \varphi$, it follows that there exists an element c such that $c \geq b$ and $\mathbf{M}^+, [b, c] \Vdash \psi$. By inductive hypothesis, we have that $\eta(\mathbf{M}^+) \models ST_{y,x}(\psi)[y := b, x := c]$. By definition of $ST_{y,x}(\psi)$, this is equivalent to $\eta(\mathbf{M}^+) \models y \leq x \wedge ST'_{y,x}(\psi)[y := b, x := c]$. This implies that $\eta(\mathbf{M}^+) \models \exists x(y \leq x \wedge ST'_{y,x}(\psi))[y := b]$. Since $a \leq b$ ($[a, b]$ in $\mathbf{M}^+$), we can conclude that $\eta(\mathbf{M}^+) \models ST_{x,y}(\Diamond_r \psi)[x := a, y := b]$. The converse direction can be proved in a similar way. The case $\varphi = \Diamond_l \psi$ is completely analogous and thus omitted. □

Corollary 1. *A* $\mathrm{PNL}^{\pi+}$*-formula* φ *is satisfiable in a class of non-strict interval structures built over a class of linear orderings* $\mathcal{C}$ *iff* $ST_{x,y}(\varphi)$ *is satisfiable in the class of all* $\mathrm{FO}^2[<]$*-models expanding linear orderings from* $\mathcal{C}$*.*

Since the above translation is polynomial in the size of the input formula, decidability of $\mathrm{PNL}^{\pi+}$ follows from Theorem 1.

Corollary 2. *The satisfiability problem for* $\mathrm{PNL}^{\pi+}$ *is decidable in NEXPTIME for each of the classes of non-strict interval structures built over (i) the class of all linear orderings, (ii) the class of all well-orderings, (iii) the class of all finite linear orderings, and (iv) the linear ordering on* $\mathbb{N}$*.*

This result can be extended to decide the satisfiability problem for $\mathrm{PNL}^{\pi+}$ over any class of linear orderings, definable in $\mathrm{FO}^2[<]$ within any of the above, e.g., the class of all (un)bounded (above, below) linear orderings or all (un)bounded above well-orderings, etc. On the contrary, the decidability of the satisfiability problem for $\mathrm{PNL}^{\pi+}$ on any of the classes of all discrete, dense, or Dedekind complete linear orderings is still open.

Since $\mathrm{PNL}^+ \prec \mathrm{PNL}^{\pi+}$ and $\mathrm{PNL}^- \prec_I \mathrm{PNL}^{\pi+}$, both PNL^+ and PNL^- are decidable in NEXPTIME (at least) over the same classes of orderings as $\mathrm{PNL}^{\pi+}$. Moreover, a translation from PNL^+ to $\mathrm{FO}^2[<]$ can be obtained from that for $\mathrm{PNL}^{\pi+}$ by simply removing the rule for π, while a translation from PNL^- to

$\mathrm{FO}^2[<]$ can be obtained from that for $\mathrm{PNL}^{\pi+}$ by removing the rule for π, by substituting $<$ for $\leq$, and by replacing $\Diamond_r$ (resp., $\Diamond_l$) with $\langle A \rangle$ (resp., $\langle \overline{A} \rangle$).

The NEXPTIME-hardness of the satisfiability problem for $\mathrm{PNL}^{\pi+}$, PNL^{+}, and PNL^{-} can be proved by exploiting the very same reduction from the exponential tiling problem given by Bresolin et al. for PNLs future fragments [4].

Theorem 6. *The satisfiability problem for* PNL^{-}*,* PNL^{+}*, and* $\mathrm{PNL}^{\pi+}$ *interpreted in the class of all linear orderings, the class of all well-orderings, the class of all finite linear orderings, and the linear ordering on* $\mathbb{N}$ *is NEXPTIME-complete.*

5 Expressive Completeness

In this section, we show that $\mathrm{PNL}^{\pi+}$ is at least as expressive as $\mathrm{FO}^2[<]$, that is, we show that every formula of $\mathrm{FO}^2[<]$ can be translated into an equivalent formula of $\mathrm{PNL}^{\pi+}$ (see Section 2). This allows us to conclude that $\mathrm{PNL}^{\pi+}$ is as expressive as $\mathrm{FO}^2[<]$. A similar result for CDT was given by Venema in [22], where the expressive completeness of CDT with respect to $\mathrm{FO}^3_{x,y}[<]$ (the fragment of first-order logic interpreted over linear orderings whose language features only three, possibly reused variables and at most two of them, x and y, can be free) was proved. Both results can be viewed as interval-based counterparts of Kamp's theorem for propositional point-based linear time temporal logic [14].

The translation τ from $\mathrm{FO}^2[<]$ to $\mathrm{PNL}^{\pi+}$ is given in the following table:

Basic formulas	Non-basic formulas
$\tau[x,y](x=x) = \tau[x,y](y=y) = \top$	$\tau[x,y](\neg\alpha) = \neg\tau[x,y](\alpha)$
$\tau[x,y](x=y) = \tau[x,y](y=x) = \pi$	$\tau[x,y](\alpha \vee \beta) = \tau[x,y](\alpha) \vee \tau[x,y](\beta)$
$\tau[x,y](y<x) = \bot$	$\tau[x,y](\exists x\beta) =$
$\tau[x,y](x<y) = \neg\pi$	$\Diamond_r(\tau[y,x](\beta)) \vee \Box_r\Diamond_l(\tau[x,y](\beta))$
$\tau[x,y](P(x,x)) = \Diamond_l(\pi \wedge p^{\leq} \wedge p^{\geq})$	$\tau[x,y](\exists y\beta) =$
$\tau[x,y](P(y,y)) = \Diamond_r(\pi \wedge p^{\leq} \wedge p^{\geq})$	$\Diamond_l(\tau[y,x](\beta)) \vee \Box_l\Diamond_r(\tau[x,y](\beta))$
$\tau[x,y](P(x,y)) = p^{\leq}$	
$\tau[x,y](P(y,x)) = p^{\geq}$	

As formally stated by Theorem 7 below, every $\mathrm{FO}^2[<]$-formula $\alpha(x,y)$ is mapped into two distinct $\mathrm{PNL}^{\pi+}$-formulas $\tau[x,y](\alpha)$ and $\tau[y,x](\alpha)$. The first one captures all and only the models of $\alpha(x,y)$ where $x \leq y$ (if any), while the second one captures all and only the models of $\alpha(x,y)$ where $y \leq x$ (if any).

Example 1. Consider the formula $\alpha = \exists x \neg \exists y (x < y)$, which constrains the model to be right-bounded. Let $\beta = \exists y(x<y)$. We have that

$$\begin{aligned}\tau[x,y](\beta) &= \Diamond_l(\tau[y,x](x<y)) \vee \Box_l\Diamond_r(\tau[x,y](x<y)) = \\ &= \Diamond_l\bot \vee \Box_l\Diamond_r\neg\pi \ (\equiv \ \Box_l\Diamond_r\neg\pi)\end{aligned}$$

and that

$$\begin{aligned}\tau[y,x](\beta) &= \Diamond_r(\tau[x,y](x<y)) \vee \Box_r\Diamond_l(\tau[y,x](x<y)) = \\ &= \Diamond_r\neg\pi \vee \Box_r\Diamond_l\bot \ (\equiv \ \Diamond_r\neg\pi)\end{aligned}$$

The resulting translation of α is:

$$\begin{aligned}\tau[x,y](\alpha) &= \Diamond_r(\tau[y,x](\neg\beta)) \vee \Box_r\Diamond_l(\tau[x,y](\neg\beta)) \ = \\ &= \Diamond_r(\neg\tau[y,x](\beta)) \vee \Box_r\Diamond_l(\neg\tau[x,y](\beta)) \ = \\ &= \Diamond_r\neg\Diamond_r\neg\pi \vee \Box_r\Diamond_l\neg\Box_l\Diamond_r\neg\pi \ = \\ &= \Diamond_r\Box_r\pi \vee \Box_r\Diamond_l\Diamond_l\Box_r\pi \ (\equiv \ \Diamond_r\Box_r\pi \vee \Box_r\pi)\end{aligned}$$

which is a PNL$^{\pi+}$-formula which constrains the model to be right-bounded.

Let $\mathcal{A} = \langle \mathbb{D}, V_{\mathcal{A}}\rangle$ be a FO$^2[<]$-model and let $\zeta(\mathcal{A}) = \langle \mathbb{I}(\mathbb{D})^+, V_{\zeta(\mathcal{A})}\rangle$ be the corresponding PNL$^{\pi+}$-model (see Section 2).

Theorem 7. *For every* FO$^2[<]$*-formula* $\alpha(x,y)$*, every* FO$^2[<]$*-model* $\mathcal{A} = \langle \mathbb{D}, V_{\mathcal{A}}\rangle$*, and every pair* $a, b \in D$*, with* $a \leq b$*, (i)* $\mathcal{A} \models \alpha(a,b)$ *if and only if* $\zeta(\mathcal{A}), [a,b] \Vdash \tau[x,y](\alpha)$ *and (ii)* $\mathcal{A} \models \alpha(b,a)$ *if and only if* $\zeta(\mathcal{A}), [a,b] \Vdash \tau[y,x](\alpha)$*.*

Proof. The proof is by simultaneous induction on the complexity of α.

- $\alpha = (x = x)$ or $\alpha = (y = y)$. Both α and $\tau[x,y](\alpha) = \top$ are true.
- $\alpha = (x < y)$. As for claim *(i)*, $\mathcal{A} \models \alpha(a,b)$ iff $a < b$ iff $\zeta(\mathcal{A}), [a,b] \Vdash \neg\pi$. As for claim *(ii)* $\mathcal{A} \not\models \alpha(b,a)$, since $a \leq b$, and $\zeta(\mathcal{A}), [a,b] \nVdash \tau[y,x](x < y)(= \bot)$. Likewise, for $\alpha = (y < x)$.
- $\alpha = P(x,y)$ or $\alpha = P(y,x)$. Both claims follow from the valuation of $p^{\leq}$ and $p^{\geq}$ (given in Section 2).
- $\alpha = P(x,x)$. As for claim *(i)*, $\mathcal{A} \models \alpha(a,b)$ iff $\mathcal{A} \models P(a,a)$ iff $\zeta(\mathcal{A}), [a,a] \Vdash \pi \wedge p^{\leq} \wedge p^{\geq}$ iff $\zeta(\mathcal{A}), [a,b] \Vdash \Diamond_l(\pi \wedge p^{\leq} \wedge p^{\geq})$. A similar argument can be used to prove claim *(ii)*. Likewise for $\alpha = P(y,y)$.
- The Boolean cases are straightforward.
- $\alpha = \exists x\beta$. As for claim *(i)*, suppose that $\mathcal{A} \models \alpha(a,b)$. Then, there is $c \in \mathcal{A}$ such that $\mathcal{A} \models \beta(c,b)$. There are two (non-exclusive) cases: $b \leq c$ and $c \leq b$. If $b \leq c$, by the inductive hypothesis, we have that $\zeta(\mathcal{A}), [b,c] \Vdash \tau[y,x](\beta)$ and thus $\zeta(\mathcal{A}), [a,b] \Vdash \Diamond_r(\tau[y,x](\beta))$. Likewise, if $c \leq b$, by the inductive hypothesis, we have that $\zeta(\mathcal{A}), [c,b] \Vdash \tau[x,y](\beta)$ and thus for every d such that $b \leq d$, $\zeta(\mathcal{A}), [b,d] \Vdash \Diamond_l(\tau[x,y](\beta))$, that is, $\zeta(\mathcal{A}), [a,b] \Vdash \Box_r\Diamond_l(\tau[x,y](\beta))$. Hence $\zeta(\mathcal{A}), [a,b] \Vdash \Diamond_r(\tau[y,x](\beta)) \vee \Box_r\Diamond_l(\tau[x,y](\beta))$, that is, $\zeta(\mathcal{A}), [a,b] \Vdash \tau[x,y](\alpha)$. For the converse direction, it suffices to note that the interval $[a,b]$ has at least one right neighbor, viz. $[b,b]$, and thus the above argument can be reversed. Claim *(ii)* can be proved in a similar way.
- $\alpha = \exists y\beta$. Analogous to the previous case. □

Corollary 3. *For every formula* $\alpha(x,y)$ *and every* FO$^2[<]$*-model* $\mathcal{A} = \langle \mathbb{D}, V_{\mathcal{A}}\rangle$*,* $\mathcal{A} \models \forall x\forall y\alpha(x,y)$ *if and only if* $\zeta(\mathcal{A}) \Vdash \tau[x,y](\alpha) \wedge \tau[y,x](\alpha)$*.*

Definition 3. *We say that a* PNL$^{\pi+}$*-model* **M** *of the considered language is* synchronized on a pair of variables $(p^{\leq}, p^{\geq})$ *if these variables are equally true at any point interval* $[a,a]$ *in* **M***;* **M** *is* synchronized for a FO$^2[<]$-formula α *if it is synchronized on every pair of variables* $(p^{\leq}, p^{\geq})$ *corresponding to a predicate* p *occurring in* α*;* **M** *is* synchronized *if it is synchronized on every pair* $(p^{\leq}, p^{\geq})$*.*

It is immediate to see that every model $\zeta(\mathcal{A})$, where $\mathcal{A}$ is a $\mathrm{FO}^2[<]$-model, is synchronized. Conversely, every synchronized $\mathrm{PNL}^{\pi+}$-model $\mathbf{M}$ can be represented as $\zeta(\mathcal{A})$ for some model $\mathcal{A}$ for $\mathrm{FO}^2[<]$: the linear ordering of $\mathcal{A}$ is inherited from $\mathbf{M}$ and the interpretation of every binary predicate P is defined in accordance with Theorem 7, that is, for any $a, b \in \mathcal{A}$ we set $P(a, b)$ to be true precisely when $a \leq b$ and $\mathbf{M}, [a, b] \Vdash p^{\leq}$ or $b \leq a$ and $\mathbf{M}, [b, a] \Vdash p^{\geq}$. Due to the synchronization, these two conditions agree when $a = b$. Furthermore, the condition that a $\mathrm{PNL}^{\pi+}$-model $\mathbf{M}$ is synchronized on a pair of variables $p^{\leq}$ and $p^{\geq}$ can be expressed by the validity in $\mathbf{M}$ of the formula $[U](\pi \rightarrow (p^{\leq} \leftrightarrow p^{\geq}))$, where $[U]$ is the *universal modality*, which is definable in $\mathrm{PNL}^{\pi+}$ as follows [7]:

$$[U]\varphi ::= \Box_r\Box_r\Box_l\varphi \wedge \Box_r\Box_l\Box_l\varphi \wedge \Box_l\Box_l\Box_r\varphi \wedge \Box_l\Box_r\Box_r\varphi.$$

Building on this observation, we associate with every $\mathrm{FO}^2[<]$-formula α the formulas

$$\sigma_v(\alpha) = \left(\bigwedge_{p^{\leq}, p^{\geq}} [U](\pi \rightarrow (p^{\leq} \leftrightarrow p^{\geq})) \right) \rightarrow (\tau[x, y](\alpha) \wedge \tau[y, x](\alpha))$$

and

$$\sigma_s(\alpha) = \left(\bigwedge_{p^{\leq}, p^{\geq}} [U](\pi \rightarrow (p^{\leq} \leftrightarrow p^{\geq})) \right) \wedge (\tau[x, y](\alpha) \vee \tau[y, x](\alpha)),$$

where the conjunctions range over all pairs $p^{\leq}, p^{\geq}$ corresponding to predicates occurring in α.

Corollary 4. *For any* $\mathrm{FO}^2[<]$*-formula* α*, (i)* α *is valid in all* $\mathrm{FO}^2[<]$*-models iff* $\sigma_v(\alpha)$ *is a valid* $\mathrm{PNL}^{\pi+}$*-formula, and (ii)* α *is satisfiable in some* $\mathrm{FO}^2[<]$*-model iff* $\sigma_s(\alpha)$ *is a satisfiable* $\mathrm{PNL}^{\pi+}$*-formula.*

Notice that the proposed translation from $\mathrm{FO}^2[<]$ to $\mathrm{PNL}^{\pi+}$ is exponential, due to the clause for the existential quantifier. We do not know whether there exists a polynomial translation or not.

$$\begin{tikzcd}
\mathrm{CDT} \arrow[rr, dash, "\equiv_R"] \arrow[dr, dash, "\prec"] & & \mathrm{FO}^3_{x,y}[<] \arrow[dr, dash, "\prec"] & \\
& \mathrm{PNL}^{\pi+} \arrow[rr, dash, "\equiv_R"] & & \mathrm{FO}^2[<]
\end{tikzcd}$$

Fig. 1. Expressive completeness results for interval logics

In Figure 1 we put together the expressive completeness results for CDT and $\mathrm{PNL}^{\pi+}$, using the notation introduced in Section 2. Since $\mathrm{FO}^2[<]$ is a proper fragment of $\mathrm{FO}^3_{x,y}[<]$, from the equivalences between CDT and $\mathrm{FO}^3_{x,y}[<]$ and between $\mathrm{PNL}^{\pi+}$ and $\mathrm{FO}^2[<]$ it immediately follows that CDT is strictly more expressive than $\mathrm{PNL}^{\pi+}$.

6 PNL$^{\pi+}$ and Other HS Fragments

In this section we explore the relationships between PNL$^{\pi+}$ and other fragments of HS. More precisely, we describe the fragments of HS which are fragments of PNL$^{\pi+}$ as well. To this end, we consider all other interval modalities of HS, namely, $\langle B\rangle$, $\langle E\rangle$, $\langle O\rangle$, $\langle D\rangle$, $\langle L\rangle$, and their transposes, which correspond to Allen's relations *begins, ends, overlaps, during*, and *after*, and their inverse relations. The semantics of such modalities can be given by their *standard translations* into first-order logic:

$$\begin{aligned}
ST_{x,y}(\langle B\rangle\varphi) &= x \leq y \wedge \exists z(z < y \wedge ST_{x,z}(\varphi))\\
ST_{x,y}(\langle E\rangle\varphi) &= x \leq y \wedge \exists z(x < z \wedge ST_{z,y}(\varphi))\\
ST_{x,y}(\langle O\rangle\varphi) &= x \leq y \wedge \exists z(x < z < y \wedge \exists y(y < x \wedge ST_{y,z}(\varphi)))\\
ST_{x,y}(\langle D\rangle\varphi) &= x \leq y \wedge \exists z(x < z < y \wedge \exists y(x < y \wedge ST_{y,z}(\varphi)))\\
ST_{x,y}(\langle L\rangle\varphi) &= x \leq y \wedge \exists x(y < x \wedge \exists y ST_{x,y}(\varphi))
\end{aligned}$$

The standard translation of $\langle L\rangle$ is a two-variable formula, while the standard translations of the other modalities are three-variable formulas. By taking advantage of the translation from FO$^2[<]$ to PNL$^{\pi+}$, $\langle L\rangle$ can be defined in PNL$^{\pi+}$ as follows: $\langle L\rangle\varphi = \diamond_r(\neg\pi \wedge \diamond_r\varphi)$. We show that the other interval modalities cannot be defined in PNL$^{\pi+}$ by a game-theoretic argument similar to the one of Theorem 2. To this end, we define the *k-round* PNL$^{\pi+}$*-bisimulation game* played on a pair of PNL$^{\pi+}$ models $(\mathbf{M_0}^+, \mathbf{M_1}^+)$ starting from a given initial configuration as follows: the rules of the game are the same of the k-round PNL$^+$-bisimulation game described in Section 3; the only difference is that a configuration $([a_0, b_0], [a_1, b_1])$ constitutes a local isomorphism between $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$ if and only if *(i)* $[a_0, b_0]$ and $[a_1, b_1]$ share the same valuation of propositional variables, and *(ii)* $a_0 = b_0$ iff $a_1 = b_1$, that is, $\mathbf{M_0}^+, [a_0, b_0] \Vdash \pi$ iff $\mathbf{M_1}^+, [a_1, b_1] \Vdash \pi$. The following proposition is analogous to Proposition 1.

Proposition 2. *Let $\mathcal{P}$ be a finite set of propositional letters. For all $k \geq 0$, Player II has a winning strategy in the k-round* PNL$^{\pi+}$*-bisimulation game on* $\mathbf{M_0}^+$ *and* $\mathbf{M_1}^+$ *with initial configuration* $([a_0, b_0], [a_1, b_1])$ *iff* $[a_0, b_0]$ *and* $[a_1, b_1]$ *satisfy the same formulas of* PNL$^{\pi+}$ *over $\mathcal{P}$ with operator depth at most k.*

We exploit Proposition 2 to prove that none of the interval modalities $\langle B\rangle$, $\langle E\rangle$, $\langle O\rangle$, and $\langle D\rangle$ is expressible in PNL$^{\pi+}$. The proof structure is always the same: for every operator $\langle X\rangle$, we choose two models $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$ that can be distinguished with a formula containing $\langle X\rangle$ and we prove that Player II has a winning strategy in the k-rounds PNL$^{\pi+}$-bisimulation game.

Theorem 8. *Neither of $\langle B\rangle$, $\langle E\rangle$, $\langle O\rangle$, and $\langle D\rangle$ can be defined in* PNL$^{\pi+}$.

Proof. We prove the claim for $\langle B\rangle$ and $\langle D\rangle$; the other cases are analogous. Consider the PNL$^{\pi+}$-models $\mathbf{M_0}^+ = \langle \mathbb{I}(\mathbb{Z} \setminus \{1, 2\})^+, V_0\rangle$ and $\mathbf{M_1}^+ = \langle \mathbb{I}(\mathbb{Z})^+, V_1\rangle$, where V_1 is such that p holds for all intervals $[a, b]$ such that $a < b$ and V_0 is the restriction of V_1 to $\mathbb{I}(\mathbb{Z} \setminus \{1, 2\})^+$. Note that $\mathbf{M_1}^+, [0, 3] \Vdash \langle B \rangle p$, while $\mathbf{M_0}^+, [0, 3] \not\Vdash \langle B \rangle p$; likewise for $\langle D \rangle p$. Thus, to prove the claims it suffices to show that Player II has a winning strategy for the k-round PNL$^{\pi+}$-bisimulation game between $\mathbf{M_0}^+$ and $\mathbf{M_1}^+$, with initial configuration $([0, 3], [0, 3])$. In fact, Player II has a *uniform* strategy to play forever that game: at any position, assuming that Player I has not won yet, if he chooses a $\Diamond_r$-move then Player II arbitrarily chooses a right-neighbor of the current interval on the other structure, with the only constraint to take a point-interval if and only if Player I has taken a point-interval as well. If Player I chooses a $\Diamond_l$-move, Player II acts likewise. $\square$

7 Conclusions

In this paper we explored expressiveness and decidability issues for PNLs. First, we compared PNL$^{\pi+}$ with PNL$^+$ and PNL$^-$, and we showed that the former is strictly more expressive than the other two. Then, we proved that PNL$^{\pi+}$ is decidable by embedding it into FO$^2[<]$. Next, we proved that PNL$^{\pi+}$ is as expressive as FO$^2[<]$. Finally, we compared PNL$^{\pi+}$ with other interval logics.

A number of open questions remain. To mention just two: Is the satisfiability problem for PNL$^{\pi+}$ over the classes of all discrete, dense, or Dedekind complete linear orders decidable? Can we extend PNL$^{\pi+}$ with any modality in the set $\{\langle B \rangle, \langle E \rangle, \langle O \rangle, \langle D \rangle\}$ to preserve decidability? We can foresee various natural further developments stemming from the present work. In particular, the tableau systems that have been developed in [2,3,4] for PNLs over specific structures, such as $\mathbb{N}$ and $\mathbb{Z}$, can be considered for adaptation to deal with FO$^2[<]$ over these and related classes of linear orders. As for expressiveness, here we have only partially explored the relationships between PNLs and other fragments of HS. A comparison with instant-based temporal logics can be of interest as well. For example, there is an obvious embedding of the standard instant-based temporal logic TL[F,P] into PNL$^{\pi+}$. The (non-)existence of the opposite embedding is more interesting, but also more difficult to state in a precise way.

Acknowledgements

This work has been funded by the bilateral project "Temporal logics in computer and information sciences", supported by the Italian Ministero degli Affari Esteri and the National Research Foundation of South Africa, under the Joint Italy/South Africa Science and Technology Agreement. In addition, Davide Bresolin and Angelo Montanari have been supported by the European INTAS project on "Algebraic and Deduction Methods in Non Classical Logics and their Applications to Computer Science" and the Italian PRIN project on "Constraints and preferences as a unifying formalism for system analysis and solution of real-life problems", while Guido Sciavicco has been funded by the Spanish

Ministry of Education and Science (MEC) and the European Regional Development Fund of the European Commission (FEDER), project IDEATIO, Ref. No. TIN2006-15460-C04-01.

References

1. E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem.* Perspectives of Mathematical Logic. Springer, 1997.
2. D. Bresolin and A. Montanari. A tableau-based decision procedure for Right Propositional Neighborhood Logic. In *Proceedings of TABLEAUX 2005*, volume 3702 of *LNAI*, pages 63–77. Springer, 2005.
3. D. Bresolin, A. Montanari, and P. Sala. An optimal tableau-based decision algorithm for Propositional Neighborhood Logic. In *Proceedings of STACS 2007*, volume 4393 of *LNCS*, pages 549–560. Springer, 2007.
4. D. Bresolin, A. Montanari, and G. Sciavicco. An optimal decision procedure for Right Propositional Neighborhood Logic. *Journal of Automated Reasoning*, 2006. DOI10.1007/s10817-006-9051-0.
5. K. Etessami, M. Y. Vardi, and T. Wilke. First-order logic with two variables and unary temporal logic. *Information and Computation*, 179(2):279–295, 2002.
6. D. M. Gabbay, I. M. Hodkinson, and M. Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects.* Oxford University Press, 1994.
7. V. Goranko, A. Montanari, and G. Sciavicco. Propositional interval neighborhood temporal logics. *Journal of Universal Computer Science*, 9(9):1137–1167, 2003.
8. V. Goranko, A. Montanari, and G. Sciavicco. A road map of interval temporal logics and duration calculi. *J. of Applied Non-Classical Logics*, 14(1–2):9–54, 2004.
9. V. Goranko, A. Montanari, G. Sciavicco, and P. Sala. A general tableau method for propositional interval temporal logics: Theory and implementation. *Journal of Applied Logic*, 4(3):305–330, 2006.
10. V. Goranko and M. Otto. Model theory of modal logic. In P. Blackburn et al., editor, *Handbook of Modal Logic*, pages 249–329. Elsevier, 2007.
11. E. Grädel, P. G. Kolaitis, and M. Y. Vardi. On the decision problem for two-variable first-order logic. *Bulletin of Symbolic Logic*, 3(1):53–69, 1997.
12. J. Halpern and Y. Shoham. A propositional modal logic of time intervals. *Journal of the ACM*, 38(4):935–962, 1991.
13. N. Immerman and D. Kozen. Definability with bounded number of bound variables. *Information and Computation*, 83(2):121–139, 1989.
14. H. Kamp. Events, instants and temporal reference. In R. Bäuerle, U. Egli, and C. Schwarze, editors, *Semantics from different points of view*, pages 376–417. de Gruyter, 1979.
15. K. Lodaya. Sharpening the undecidability of interval temporal logic. In *Proc. of 6th Asian Computing Science Conference*, volume 1961 of *LNCS*, pages 290–298. Springer, 2000.
16. M. Mortimer. On languages with two variables. *Zeitschr. f. math. Logik u. Grundlagen d. Math*, 21:135–140, 1975.
17. B. Moszkowski. *Reasoning about digital circuits.* Tech. rep. stan-cs-83-970, Dept. of Computer Science, Stanford University, Stanford, CA, 1983.
18. M. Otto. Two variable first-order logic over ordered domains. *Journal of Symbolic Logic*, 66(2):685–702, 2001.

19. D. Scott. A decision method for validity of sentences in two variables. *Journal of Symbolic Logic*, 27:377, 1962.
20. I. Shapirovsky and V. Shehtman. Chronological future modality in Minkowski spacetime. In P. Balbiani, N. Y. Suzuki, F. Wolter, and M. Zakharyaschev, editors, *Advances in Modal Logic*, volume 4, pages 437–459. King's College Publications, London, 2003.
21. Y. Venema. Expressiveness and completeness of an interval tense logic. *Notre Dame Journal of Formal Logic*, 31(4):529–547, 1990.
22. Y. Venema. A modal logic for chopping intervals. *Journal of Logic and Computation*, 1(4):453–476, 1991.

Reasoning About Sequences of Memory States[⋆]

Rémi Brochenin, Stéphane Demri, and Etienne Lozes

LSV, ENS Cachan, CNRS, INRIA
`{brocheni,demri,lozes}@lsv.ens-cachan.fr`

Abstract. In order to verify programs with pointer variables, we introduce a temporal logic $\mathrm{LTL}^{\mathrm{mem}}$ whose underlying assertion language is the quantifier-free fragment of separation logic and the temporal logic on the top of it is the standard linear-time temporal logic LTL. We analyze the complexity of various model-checking and satisfiability problems for $\mathrm{LTL}^{\mathrm{mem}}$, considering various fragments of separation logic (including pointer arithmetic), various classes of models (with or without constant heap), and the influence of fixing the initial memory state. We provide a complete picture based on these criteria. Our main decidability result is PSPACE-completeness of the satisfiability problems on the record fragment and on a classical fragment allowing pointer arithmetic. Σ^0_1-completeness or Σ^1_1-completeness results are established for various problems by reducing standard problems for Minsky machines, and underline the tightness of our decidability results.

1 Introduction

Verification of programs with pointers. Model-checking of infinite-state systems is a very active area of formal verification [BCMS01] even though in full generality, simple reachability questions are undecidable. Nevertheless, many classes of infinite-state systems can be analyzed, such as Petri nets, timed automata, etc. Programs with pointer variables suffer the same drawback since reachability problems are also undecidable, see e.g. [BFN04, BBH+06]. It is worth noting that specific properties need to be verified for such programs, such as the existence of memory leaks, memory violation, or shape analysis. Prominent logics for analyzing such programs are Separation Logic [Rey02], pointer assertion logic PAL [JJKS97], TVLA [LAS00] and alias logic [BIL04], to quote a few examples.

Temporal Separation Logic: what for? Since [Pnu77], temporal logics are also used as languages for formal specification of programs. General and powerful automata-based techniques for verification have been developed, see e.g. [VW94]. On the other hand, Separation Logic is a static logic for program annotation [Rey02], and more recently for symbolic computation [BCO05]. Extending the scope of application of Separation Logic to standard temporal logic-based verification techniques has many potential interests. First, it provides a rich underlying assertion language where properties more complex than accessibility

⋆ Work supported by the RNTL project "AVERILES". The first author is supported by a fellowship from CNRS/DGA.

can be stated. Second, this probably yields a significant feedback for the purely static Separation Logic extended with general recursion, which has not been very studied up to now. For instance, if we write $\mathsf{X}\mathtt{x}$ to denote the next value of $\mathtt{x}$ (also sometimes written $\mathtt{x}'$), the formula $(\mathtt{x} \hookrightarrow \mathsf{X}\mathtt{x})\mathsf{U}(\mathtt{x} \hookrightarrow \mathtt{null})$, understood on a model with constant heap, characterises the existence of a simple flat list, which is usually written $\mu L(\mathtt{x}).\ \mathtt{x} \hookrightarrow \mathtt{null} \vee \exists \mathtt{x}'.\mathtt{x} \hookrightarrow \mathtt{x}' \wedge L(\mathtt{x}')$. Third, temporal logics allow to work in the very convenient framework of "programs-as-formulae" and decision procedures for logical problems can be directly used for program verification. For instance, the previous formula can be seen as a program walking on a list, and more generally programs without destructive updates can be expressed as formulae. Some programs with destructive updates that perform a simple pass on the heap, have an input-output relation that may be described by a formula. For instance, the formula $(\mathtt{x} \hookrightarrow_0 \mathsf{X}\mathtt{x} \wedge \mathsf{X}\mathtt{x} \hookrightarrow_1 \mathtt{x})\mathsf{U}\mathtt{x} \hookrightarrow_0 \mathtt{null}$ expresses broadly that the list in initial heap h_0 is reversed in final heap h_1. Fourth, pointer arithmetic has been poorly studied until now, whereas arithmetical constraints in temporal logics are known to lead to undecidability, see e.g. [CC00]. Actually, there is a growing interest in understanding the interplay of pointer arithmetic, temporal reasoning, and non aliasing properties.

Our contribution. We introduce a linear-time temporal logic $\mathrm{LTL}^{\mathrm{mem}}$ to specify sequences of memory states with underlying assertion language based on quantifier-free Separation Logic [Rey02]. From a logical perspective, the logic $\mathrm{LTL}^{\mathrm{mem}}$ can be viewed as a many-dimensional logic [GKWZ03] since $\mathrm{LTL}^{\mathrm{mem}}$ contains a temporal dimension and the spatial dimension for memory states. Our logic addresses a very general notion of models, including the aspects of pointer arithmetic and recursive structures with records. We distinguish the satisfiability problems from the model-checking problems, as well as distinct subclasses of interesting programs, like for instance the programs without destructive update. The result that is the most promising for future implementation is the PSPACE-completeness of the satisfiability problems SAT(CL) and SAT(RF) where CL is the classical fragment without separation connectives and RF is the record fragment with no pointer arithmetic but with separation connectives. This result is very tight, as both propositional LTL and static Separation Logic are already PSPACE-complete [SC85, CYO01]. These results are obtained by reduction to the nonemptiness problem for Büchi automata on an alphabet of symbolic memory states obtained by an abstraction that we show sound and complete, see e.g. [Loz04, CGH05]. Such abstractions are similar to resource graphs from [GM05]. This is a variant of the automata-based approach introduced in [VW94] for plain LTL and further developed with concrete domains of interpretation in [DD07]. Surprisingly, the abstraction method used to establish these results does not scale to the whole logic, due to a subtle interplay between separation connectives and pointer arithmetic. Moreover, we provide new undecidability results for several problems, for instance $\mathrm{SAT}^{ct}(\mathrm{LF})$ (satisfiability with constant heap on the list fragment).

Related work. Previous temporal logics designed for pointer verification include Evolution Temporal Logic [YRSW03], based on the three-valued logic abstraction method that made the success of TVLA [LAS00], and Navigation temporal logic [DKR04], based on a tableau method quite similar to our automaton-based reduction. In these works, the assertion language for states is quite rich, as it includes for instance list predicate, quantification over adresses, and a freshness predicate. Because of this high expressive power, only incomplete abstractions are proposed, whereas we stick to exact methods. More importantly, our work addresses models with constant heaps and pointer arithmetic, which has not been done so far, and leads to a quite different perspective.

Omitted proofs can be found in the report [BDL07].

2 Memory Model and Specification Language

In this section, we introduce a separation logic dealing with pointer arithmetic and record values, and a temporal logic $\mathrm{LTL}^{\mathrm{mem}}$. Unlike BI's pointer logic from [IO01], we allow pointer arithmetic. Model-checking programs with pointer variables over $\mathrm{LTL}^{\mathrm{mem}}$ specifications is our main problem of interest.

2.1 A Separation Logic with Pointer Arithmetic

Memory states. Let us introduce our model of memory. It captures features of programs with pointer variables that use pointer arithmetic and records. We assume a countably infinite set $\mathtt{Var}$ of variables (as usual, for a fixed formula we need only a finite amount), and an infinite set $\mathtt{Val}$ of values containing the set $\mathbb{N}$ of naturals, thought as address indexes, and a special value *nil*. For simplicity, we assume that $\mathtt{Val} = \mathbb{N} \uplus \{nil\}$. In order to model field selectors, we consider some infinite set $\mathtt{Lab}$ of labels. We will usually range over values with u, v, over naturals with i, j, over labels with $l, r, next, prev$, and over variables with $\mathtt{x}, \mathtt{y}$. In the remainder, we will assume some fixed injection $(\mathtt{x}, i) \in \mathtt{Var} \times \mathbb{N} \mapsto \langle \mathtt{x}, i \rangle \in \mathtt{Var}$.

We use the notation $E \rightharpoonup_{fin} F$ for the set of partial functions from E to F of finite domain; and $E \rightharpoonup_{fin+} F$ for the set of partial functions from E to F of finite and nonempty domain. The sets $\mathcal{S}$ of stores and $\mathcal{H}$ of heaps are then defined as follows: $\mathcal{S} \overset{\text{def}}{\equiv} \mathtt{Var} \rightarrow \mathtt{Val}$ and $\mathcal{H} \overset{\text{def}}{\equiv} \mathbb{N} \rightharpoonup_{fin} (\mathtt{Lab} \rightharpoonup_{fin+} \mathtt{Val})$. We will range over a store with s, s' and over a heap with h, h', h_1, h_2. We call *memory state* a couple $(s, h) \in \mathcal{S} \times \mathcal{H}$.

We will refer to the domain of a heap h by $\mathtt{dom}(h) \subseteq \mathbb{N}$. Intuitively, in our memory model, each index is thought as an entry point on some record cell containing several fields. Cells are either not allocated, or allocated with some record stored in. In a memory state (s, h), the memory cell at index i is *allocated* if $i \in \mathtt{dom}(h)$; in this case the stored record is $h(i) = \{l_1 \mapsto v_1, .., l_n \mapsto v_n\}$.

Note that the size of the information held in a memory cell is not fixed, nor bounded. Our models could be more concrete considering labels as offsets and relying on pointer arithmetic. But for our purpose, it will be convenient to consider pointer arithmetic independently.

Table 1. The syntax and semantics of SL with pointer arithmetic and records

Expressions
$e ::= \mathtt{x} \mid \mathtt{null}$

Atomic formulae
$\pi ::= e = e' \mid e + i \overset{l}{\hookrightarrow} e$

State Formulae
$\mathcal{A} ::= \pi$
$\mid \mathcal{A} * \mathcal{B} \mid \mathcal{A} -\!\!* \mathcal{B} \mid \mathtt{emp}$ (spatial fragment)
$\mid \mathcal{A} \wedge \mathcal{B} \mid \mathcal{A} \rightarrow \mathcal{A} \mid \top \mid \bot$ (classical fragment)

Satisfaction

$(s,h) \models_{\mathrm{SL}} e = e'$	iff $[\![e]\!]_s = [\![e']\!]_s$, with $[\![\mathtt{x}]\!]_s = s(\mathtt{x})$ and $[\![\mathtt{null}]\!]_s = nil$
$(s,h) \models_{\mathrm{SL}} e + i \overset{l}{\hookrightarrow} e$	iff $[\![e]\!]_s \in \mathbb{N}$ and $[\![e]\!] + i \in \mathtt{dom}(h)$ and $h(s(\mathtt{x}) + i)(l) = [\![e]\!]_s$
$(s,h) \models_{\mathrm{SL}} \mathtt{emp}$	iff $\mathtt{dom}(h) = \emptyset$
$(s,h) \models_{\mathrm{SL}} \mathcal{A}_1 * \mathcal{A}_2$	iff $\exists\ h_1, h_2$ s.t. $h = h_1 * h_2$, $(s,h_1) \models_{\mathrm{SL}} \mathcal{A}_1$ and $(s,h_2) \models_{\mathrm{SL}} \mathcal{A}_2$
$(s,h) \models_{\mathrm{SL}} \mathcal{A}' -\!\!* \mathcal{A}$	iff for all h', if $h \perp h'$ and $(s,h') \models_{\mathrm{SL}} \mathcal{A}'$ then $(s, h * h') \models_{\mathrm{SL}} \mathcal{A}$
$(s,h) \models_{\mathrm{SL}} \mathcal{A}_1 \wedge \mathcal{A}_2$	iff $(s,h) \models_{\mathrm{SL}} \mathcal{A}_1$ and $(s,h) \models_{\mathrm{SL}} \mathcal{A}_2$
$(s,h) \models_{\mathrm{SL}} \mathcal{A}' \rightarrow \mathcal{A}$	iff $(s,h) \models_{\mathrm{SL}} \mathcal{A}'$ implies $(s,h) \models_{\mathrm{SL}} \mathcal{A}$
$(s,h) \models_{\mathrm{SL}} \bot$	never and $(s,h) \models_{\mathrm{SL}} \top$ always

Separation Logic. We now introduce the separation logic (SL) on top of which we will define our temporal logic. The syntax of the logic is given in Table 1.

In short, Separation logic is about reasoning on disjoint heaps, and we need to define what we mean by "disjoint heaps" in our model. We choose to allow to reason at the granularity of record cells, so that a record cell cannot be decomposed in disjoint parts. Let h_1 and h_2 be two heaps; we say that h_1 and h_2 are disjoint, noted $h_1 \perp h_2$, if $\mathtt{dom}(h_1) \cap \mathtt{dom}(h_2) = \emptyset$. The operation $h_1 * h_2$ is defined for disjoint heaps as the *disjoint union* of the two partial functions. Semantics of formulae is defined by the satisfaction relation $\models_{\mathrm{SL}}$ (see Table 1).

Formulae $\mathcal{A}$ of SL are called *state formulae.* The size of the state formula $\mathcal{A}$, written $|\mathcal{A}|$, is the length of the string $\mathcal{A}$ for some reasonably succinct encoding of variables and integers (binary representation). We will use the map $|\cdot|$ for other syntactic objects such as $\mathrm{LTL}^{\mathrm{mem}}$ formulae. A formula $\mathcal{A} * \mathcal{B}$ with the *separation conjunction* states that $\mathcal{A}$ holds on some portion of the memory heap and $\mathcal{B}$ holds on a disjoint portion. A formula $\mathcal{A} -\!\!* \mathcal{B}$ states that the current heap, when extended with any disjoint heap verifying $\mathcal{A}$, will verify $\mathcal{B}$. Boolean operators are understood as usual. In the remainder, we focus on several specific fragments of this separation logic. We say that a formula is in the *record fragment* (RF) if all subformulae $\mathtt{x} + i \overset{l}{\hookrightarrow} e$ use $i = 0$. In that case, we write $\mathtt{x} \overset{l}{\hookrightarrow} e$. We say that a formula is in the *classical fragment* (CL) if it does not use the connectives $*, -\!\!*$. Finally, we say that a formula is in the *list fragment* (LF) if it is in the classical fragment and all subformulae $\mathtt{x} + i \overset{l}{\hookrightarrow} e$ use $i = 0$ and $l = next$, and we may simply write $\mathtt{x} \hookrightarrow e$. Clearly, the classical and record fragments are incomparable, while the list fragment is included in both of them.

Let us illustrate the expressive power of SL on examples. The formula $\neg\mathtt{emp} * \neg\mathtt{emp}$ means that at least two memory cells are allocated. The formula $\mathtt{x} \overset{l}{\mapsto} e$, defined as $\neg(\neg\mathtt{emp} * \neg\mathtt{emp}) \wedge \mathtt{x} \overset{l}{\hookrightarrow} e$, is the local version of $\mathtt{x} \overset{l}{\hookrightarrow} e$: $s, h \models_{\mathrm{SL}} \mathtt{x} \overset{l}{\mapsto} e$ iff $\mathtt{dom}(h) = \{s(\mathtt{x})\}$ and $h(s(\mathtt{x}))(l) = [\![e]\!]_s$. The formula $(\mathtt{x} \overset{l}{\hookrightarrow} \mathtt{null}) -\!\!* \bot$ is satisfied at (s_0, h_0) whenever there is no heap h_1 with $h_1 \perp h_0$ that allocates the variable $\mathtt{x}$ to nil on l field, that is $\mathtt{x}$ is allocated in h_0.

$\mathcal{A}$ is valid iff for every memory state (s, h), we have $(s, h) \models_{\mathrm{SL}} \mathcal{A}$ (written $\models_{\mathrm{SL}} \mathcal{A}$). Satisfiability is defined dually.

Proposition 1. *The model-checking, satisfiability and validity problems for* SL *are* PSPACE-*complete.*

PSPACE-hardness results are consequences of [CYO01, Sect. 5.2]. The PSPACE upper bound for model-checking for SL is obtained by reduction to model-checking for RF that is shown in PSPACE thanks to forthcoming Lemma 2. Satisfiability for SL is reduced to model-checking for SL thanks to a small memory state property: every satisfiable state formulae $\mathcal{A}$ can be satisfied by a memory state that can be encoded in polynomial size in $\mathcal{A}$.

2.2 Temporal Extension

Memory states sequences. Models of the logic $\mathrm{LTL}^{\mathrm{mem}}$ are ω-sequences of memory states, that is elements in $(\mathcal{S} \times \mathcal{H})^{\omega}$ and they are understood as infinite computations of programs with pointer variables. In order to analyze computations from programs without destructive update, we shall also consider models with constant heap, that is elements in $\mathcal{S}^{\omega} \times \mathcal{H}$.

The logic $\mathrm{LTL}^{\mathrm{mem}}$. Formulae of $\mathrm{LTL}^{\mathrm{mem}}$ are defined in Table 2. Atomic formulae of $\mathrm{LTL}^{\mathrm{mem}}$ are state formulae from SL except that variables can be prefixed by the symbol "X". For instance, $\mathsf{X}\mathtt{x}$ is interpreted by the value of $\mathtt{x}$ at the next memory state. We use the notation $\mathsf{X}^i\mathtt{x}$ for $\overbrace{\mathsf{X}\ldots\mathsf{X}}^{i \text{ times}}\mathtt{x}$ (but keep in mind that encoding $\mathsf{X}^i\mathtt{x}$ requires memory space in $\mathcal{O}(i)$). The temporal operators are the standard next-time operator X and until operator U present in LTL, see e.g. [SC85]. The satisfaction relation $\rho, t \models \phi$ where ρ is a model of $\mathrm{LTL}^{\mathrm{mem}}$, $t \in \mathbb{N}$ and ϕ is a formula is also defined in Table 2. We use standard abbreviations such as $\mathsf{F}\phi$, $\mathsf{G}\phi$ ...

We freely use propositional variables p, q, having in mind that the propositional variable p should be understood as $\mathtt{x}_p = \mathtt{x}_{\top}$ for some fixed extra variables $\mathtt{x}_p, \mathtt{x}_q, \ldots, \mathtt{x}_{\top}$.

Given a fragment Frag of SL, $\mathrm{LTL}^{\mathrm{mem}}$(Frag) is the restriction of $\mathrm{LTL}^{\mathrm{mem}}$ to formulae in which occur only state formulae built over Frag (with extended variables $\mathsf{X}^i\mathtt{x}$), and we write SAT(Frag) to denote the satisfiability problem for $\mathrm{LTL}^{\mathrm{mem}}$(Frag): given a temporal formula ϕ in $\mathrm{LTL}^{\mathrm{mem}}$(Frag), is there a model ρ such that $\rho, 0 \models \phi$? The variant problem in which we require that the model has a constant heap [resp. that the initial memory state is fixed, say (s, h)] is denoted by SAT^{ct}(Frag) [resp. SAT_{init}(Frag)]. The problem SAT^{ct}_{init}(Frag) is defined analogously.

Table 2. The syntax and semantics of LTL$^{\mathrm{mem}}$

Enriched expressions	$\eta ::= \ \mathtt{x} \mid \mathsf{X}\eta \mid \ \mathtt{null}$
Atomic formulae	$\pi ::= \ \eta = \eta' \mid \ \eta + i \overset{l}{\hookrightarrow} \eta'$
State formulae	$\mathcal{A} ::= \ \pi \mid \ \mathtt{emp} \mid \ \mathcal{A} * \mathcal{B} \mid \ \mathcal{A} \mathrel{-\!\!*} \mathcal{B} \mid \ \mathcal{A} \wedge \mathcal{B} \mid \ \mathcal{A} \rightarrow \mathcal{B} \mid \ \bot$
Temporal formulae	$\phi ::= \ \mathcal{A} \mid \ \mathsf{X}\phi \mid \ \phi \mathsf{U} \phi' \mid \ \phi \wedge \phi' \mid \ \neg\phi$

Semantics

$\rho, t \models \mathsf{X}\phi$	iff $\rho, t+1 \models \phi$.
$\rho, t \models \phi \mathsf{U} \phi'$	iff there is $t_1 \geq t$ s.t. $\rho, t_1 \models \phi'$ and $\rho, t' \models \phi$ for all $t' \in \{t, .., t_1 - 1\}$.
$\rho, t \models \phi \wedge \psi$	iff $\rho, t \models \phi$ and $\rho, t \models \psi$.
$\rho, t \models \neg\phi$	iff $\rho, t \not\models \phi$.
$\rho, t \models \mathcal{A}$	iff $s'_t, h_t \models_{\mathrm{SL}} \mathcal{A}[\mathsf{X}^k\mathtt{x} \leftarrow (\mathtt{x}, k)]$ where $\rho = (s_t, h_t)_{t \geq 0}$ and s'_t is defined by $s'_t(\langle \mathtt{x}, k \rangle) = s_{t+k}(\mathtt{x})$.

2.3 Programs with Pointer Variables

In this section, we define the model-checking problems for programs with pointer variables over LTL$^{\mathrm{mem}}$ specifications. The set $\mathtt{I}$ of *instructions* used in the programs is defined by the grammar below:

$$\begin{aligned}\mathtt{instr} ::= \ & \mathtt{x} := \mathtt{y} \mid \mathtt{skip} \\ & \mid \mathtt{x} := \mathtt{y} \rightarrow l \mid \mathtt{x} \rightarrow l := \mathtt{y} \mid \mathtt{x} := \mathtt{cons}(l_1 : x_1, .., l_k : x_k) \mid \mathtt{free}\ \mathtt{x} \\ & \mid \mathtt{x} := \mathtt{y}[i] \mid \mathtt{x}[i] := \mathtt{y} \mid \mathtt{x} = \mathtt{malloc}(i) \mid \mathtt{free}\ \mathtt{x}, i\end{aligned}$$

The denotational semantics of an instruction $\mathtt{instr}$ is defined as a partial function $[\![\ \mathtt{instr}\ ]\!] : \mathcal{S} \times \mathcal{H} \rightarrow \mathcal{S} \times \mathcal{H}$, undefined when the instruction would cause a memory violation. We list in Table 3 the formal denotational semantics of our instruction set. Boolean combinations of equalities between expressions are called guards and its set is denoted by G. A program is defined as a triple (Q, δ, q_I) such that Q is a finite set of control states, q_I is the initial state and δ is the transition relation, a subset of $Q \times G \times \mathtt{I} \times Q$. We use $q \xrightarrow{g,\mathtt{instr}} q'$ to denote a transition. We say that a program is *without destructive update* if transitions are labeled only with instructions of the form $\mathtt{x} := \mathtt{y}$, $\mathtt{x} := \mathtt{y} \rightarrow l$, and $\mathtt{x} := \mathtt{y}[i]$. We write $\mathtt{P}$ to denote the set of programs and $\mathtt{P}^{ct}$ to denote the set of programs without destructive update.

A program is a finite object whose interpretation can be viewed as an infinite-state system. More precisely, given a program $\mathtt{p} = (Q, \delta, q_I)$, the transition system $\mathcal{S}_{\mathtt{p}} = (S, \rightarrow)$ is defined as follows: $S = Q \times (\mathcal{S} \times \mathcal{H})$ (set of configurations) and $(q, (s, h)) \rightarrow (q', (s', h'))$ iff there is a transition $q \xrightarrow{g,\mathtt{instr}} q' \in \delta$ such that $(s, h) \models g$ and $(s', h') = [\![\ \mathtt{instr}\ ]\!](s, h)$. Note that $\mathcal{S}_{\mathtt{p}}$ is not necessarily linear. A computation (or execution) of $\mathtt{p}$ is defined as an infinite path in $\mathcal{S}_{\mathtt{p}}$ starting with control state q_I. Computations of $\mathtt{p}$ can be viewed as LTL$^{\mathrm{mem}}$ models, using propositional variables to encode the extra information about the control states (details are omitted herein).

Table 3. Semantics for instructions

$[\![\ \mathtt{x} := \mathtt{y} \ ]\!] \ (s,h)$	$\overset{\text{def}}{\equiv} \ (s[\mathtt{x} \mapsto s(\mathtt{y})], h).$
$[\![\ \mathtt{x} := \mathtt{y} \rightarrow l \ ]\!] \ (s, h * \{i \mapsto \{l \mapsto v, \dots\}\})$	$\overset{\text{def}}{\equiv} \ (s[\mathtt{x} \mapsto v], h * \{i \mapsto \{l \mapsto v, \dots\}\})$ with $s(\mathtt{y}) = i$
$[\![\ \mathtt{x} \rightarrow l := \mathtt{y} \ ]\!] \ (s, h * \{i \mapsto \{l \mapsto v, \dots\}\})$	$\overset{\text{def}}{\equiv} \ (s, h * \{i \mapsto \{l \mapsto s(\mathtt{y}), \dots\}\})$ with $s(\mathtt{x}) = i$
$[\![\ \mathtt{x} := \mathtt{cons}(l_1 : \mathtt{x}_1, .., l_k : \mathtt{x}_k) \ ]\!] \ \ (s,h)$	$\overset{\text{def}}{\equiv} \ \big(s[\mathtt{x} \mapsto i], h * \{i \mapsto \{l_1 \mapsto s(\mathtt{x}_1), \dots, l_k \mapsto s(\mathtt{x}_k)\}\}\big)$ with $i \notin \mathtt{dom}(h)$
$[\![\ \mathtt{free} \ \mathtt{x}, l \ ]\!] \ (s, h * \{i \mapsto \{l \mapsto v, \dots\}\})$	$\overset{\text{def}}{\equiv} \ (s, h * \{i \mapsto \{\dots\}\})$ with $s(\mathtt{x}) = i$
$[\![\ \mathtt{skip} \ ]\!] \ (s,h)$	$\overset{\text{def}}{\equiv} \ (s,h)$
$[\![\ \mathtt{x} := \mathtt{y}[i] \ ]\!] \ (s, h * \{i + i' \mapsto \{next \mapsto v\}\})$	$\overset{\text{def}}{\equiv} \ (s[\mathtt{x} \mapsto v], h * \{i \mapsto \{next \mapsto v\}\}))$ with $s(\mathtt{y}) = i'$
$[\![\ \mathtt{x}[i] := \mathtt{y} \ ]\!] \ (s, h * \{i' + i \mapsto \{next \mapsto v\}\})$	$\overset{\text{def}}{\equiv} \ (s, h * \{i + i' \mapsto \{next \mapsto s(\mathtt{y})\}\})$ with $s(\mathtt{x}) = i'$
$[\![\ \mathtt{x} := \mathtt{malloc}(i) \ ]\!](s,h)$	$\overset{\text{def}}{\equiv} \ \big(s[\mathtt{x} \mapsto i'], h * \{i' \mapsto \{next \mapsto nil\}\} * \dots * \{i' + i \mapsto \{next \mapsto nil\}\big)$ with $i', .., i' + i \notin \mathtt{dom}(h)$
$[\![\ \mathtt{free} \ \mathtt{x}, i \ ]\!] \ (s, h * \{i' + i \mapsto f\})$	$\overset{\text{def}}{\equiv} \ (s,h)$ with $s(\mathtt{x}) = i'$

Model-checking aims at checking properties expressible in LTL$^{\text{mem}}$ along computations of programs. To a logical fragment (SL, CL, RF, or LF), we associate a set of programs : all programs for SL and CL, programs with instructions having $i = 0$ for RF, and moreover with only the label *next* for LF. Given one of these fragments Frag of SL, we write MC(Frag) to denote the model-checking problem for Frag: given a temporal formula ϕ in LTL$^{\text{mem}}$ with state formulae built over Frag and a program $\mathtt{p}$ of the associated fragment, is there an infinite computation ρ of $\mathtt{p}$ such that $\rho, 0 \models \phi$ (which we write $\mathtt{p} \models \phi$)? The variant problem in which we require that the program is without destructive update [resp. that the initial memory state is fixed, say (s,h)] is denoted by MC^{ct}(Frag) [resp. MC$_{init}$(Frag)]. The problem MC$^{ct}_{init}$(Frag) is defined analogously. We may write $\mathtt{p}, (s,h) \models \phi$ to emphasize what is the initial memory state.

All the model-checking and satisfiability problems defined above can be placed in Σ^1_1 in the analytical hierarchy. Additionnally, all the above problems can easily be shown PSPACE-hard since they all generalize LTL satisfiability and model-checking [SC85].

Using extended variables $\mathsf{X}\mathtt{x}$, we may express some programs as formulae. This actually holds only for programs without update, for the semantics with constant heap. Intuitively, we express the control of the program with propositional variables, and define a formula that encode the transitions. To do so, we translate instructions of the form $\mathtt{x} := \mathtt{y}$ into $\mathsf{X}\mathtt{x} = \mathtt{y}$, $\mathtt{x} := \mathtt{y} \rightarrow l$ into

$\mathtt{y} \overset{l}{\hookrightarrow} \mathsf{X}\mathtt{x}$, and $\mathtt{x} := \mathtt{y}[i]$ into $\mathtt{y} + i \hookrightarrow \mathsf{X}\mathtt{x}$. Guards are translated accordingly. As a consequence, the following result can be derived:

Lemma 1. *Let* Frag *be a fragment among* SL, CL, RF, *or* LF. *There is a logspace reduction from* $\mathrm{MC}^{ct}(\mathrm{Frag})$ *to* $\mathrm{SAT}^{ct}(\mathrm{Frag})$ *(resp. from* $\mathrm{MC}^{ct}_{init}(\mathrm{Frag})$ *to* $\mathrm{SAT}^{ct}_{init}(\mathrm{Frag})$*).*

3 Decidable Satisfiability Problems by Abstracting Computations

In this section we establish the PSPACE-completeness of the problems SAT(CL) and SAT(RF). To do so, we abstract memory states whose size is a priori unbounded by finite symbolic memory states. As usual, temporal infinity in models is handled by Büchi automata recognizing ω-sequences. We propose below an abstraction that is correct for CL (allowing pointer arithmetic) and for RF (allowing all operators from Separation Logic) taken separately but that is not exact for the full language SL.

3.1 Syntactic Measures

The main approach to get decision procedures to verify infinite-state systems consists in introducing a symbolic representation for infinite sets of configurations. The symbolic representation defined below plays a similar role and has similarities with symbolic heaps for Separation Logic in Smallfoot [BCO05]. Let us start by some useful definitions. Following [Loz04], we introduce the set of *test formulae* that are formulae from SL of the forms below:

– $\mathtt{alloc}\ \mathtt{x} \overset{\text{def}}{\equiv} (\mathtt{x} \overset{next}{\mapsto} \mathtt{null}) {-\!\!*} \bot$ (x is allocated).

– $\mathtt{size} \geq k \overset{\text{def}}{\equiv} \overbrace{\neg\mathtt{emp} * \ldots * \neg\mathtt{emp}}^{k\ \text{times}}$ (at least k indices are allocated).

– $e + i \overset{l}{\hookrightarrow} e$, $e = e'$.

Given a formula ϕ of $\mathrm{LTL}^{\mathrm{mem}}$, we define its measure μ_ϕ understood as some pieces of information about the syntactic resources involved in ϕ. Indeed, forthcoming symbolic states are finite objects parameterized by such syntactic measures.

For a state formula $\mathcal{A}$ of $\mathrm{LTL}^{\mathrm{mem}}$, the size of memory examined by $\mathcal{A}$, written $w_{\mathcal{A}}$, is inductively defined as follows: $w_{\mathcal{A}}$ is 1 for atomic formulae, $max\{w_{\mathcal{A}_1}, w_{\mathcal{A}_2}\}$ for $\mathcal{A}_1 \wedge \mathcal{A}_2$ or $\mathcal{A}_1 \rightarrow \mathcal{A}_2$ or $\mathcal{A}_1 {-\!\!*} \mathcal{A}_2$, and $w_{\mathcal{A}_1} + w_{\mathcal{A}_2}$ for $\mathcal{A}_1 * \mathcal{A}_2$. Observe that $w_{\mathcal{A}} \leq |\mathcal{A}|$. Other simple sets about the syntactic resources of $\mathcal{A}$ need to be defined: $\mathtt{Lab}_{\mathcal{A}}$ is the set of labels from $\mathtt{Lab}$ occurring in $\mathcal{A}$, $\mathtt{Var}_{\mathcal{A}}$ is the set of variables from $\mathtt{Var}$ occurring in $\mathcal{A}$, $\epsilon_{\mathcal{A}}$ is the set of natural numbers i such that $e + i \overset{l}{\hookrightarrow} e'$ occurs in $\mathcal{A}$ and $m_{\mathcal{A}}$ is the maximal k such that $\mathsf{X}^k\mathtt{x}$ occurs in $\mathcal{A}$ for some variable $\mathtt{x}$. A measure is defined as an element of $\mathbb{N} \times \mathcal{P}_f(\mathbb{N}) \times \mathbb{N} \times \mathcal{P}_f(\mathtt{Lab}) \times \mathcal{P}_f(\mathtt{Var})$ where $\mathcal{P}_f(X)$ denotes the set of finite subsets of some set X. The set of measures has a natural lattice structure for the pointwise order, noted below $\mu \leq \mu'$. We also write $\mu[w \leftarrow 0]$ to denote the measure μ except that $w = 0$.

The measure for $\mathcal{A}$, written $\mu_{\mathcal{A}}$, is the tuple $(m_{\mathcal{A}}, \epsilon_{\mathcal{A}}, w_{\mathcal{A}}, \texttt{Lab}_{\mathcal{A}}, \texttt{Var}_{\mathcal{A}})$. The measure of some formula ϕ of LTL$^{\text{mem}}$, written μ_{ϕ}, is $\sup\{\mu_{\mathcal{A}} : \mathcal{A} \text{ occurs in } \phi\}$.

Definition 1. *Given a measure $\mu = (m, \epsilon, w, X, Y)$, we write $\mathcal{T}_{\mu}$ to denote the finite set of test formulae ψ of the grammar:*

$$e ::= \langle \mathtt{x}, u \rangle \mid \mathtt{null} \qquad f ::= e + i$$
$$\psi ::= f \overset{l}{\hookrightarrow} e \mid \mathtt{alloc}\ f \mid e = e' \mid \mathtt{size} \geq k$$

with $u \leq m$, $i \in \epsilon$, $l \in X$, $k < w$ and $\mathtt{x} \in Y$.

Observe that the cardinal of $\mathcal{T}_{\mu_{\phi}}$ is polynomial in $|\phi|$. Given a measure $\mu = (m, \epsilon, w, X, Y)$ and a memory state (s, h), we write $Abs_{\mu}(s, h) = \{\mathcal{A} \in \mathcal{T}_{\mu} : (s, h) \models_{\text{SL}} \mathcal{A}\}$ to denote the abstraction of (s, h) wrt μ. Given a measure μ and two memory states (s, h) and (s', h'), we write $(s, h) \simeq_{\mu} (s', h')$ iff $Abs_{\mu}(s, h) = Abs_{\mu}(s', h')$, that is formulae in $\mathcal{T}_{\mu}$ cannot distinguish the two memory states. Lemma 2 below states that our abstraction is correct for CL and RF.

Lemma 2. *Let (s, h) and (s', h') be two memory states such that $(s, h) \simeq_{\mu} (s', h')$ [resp. $(s, h) \simeq_{\mu[w \leftarrow 0]} (s', h')$]. For any state formula $\mathcal{A}$ such that $\mu_{\mathcal{A}} \leqslant \mu$ and $\mathcal{A}$ belongs to* RF *[resp.* CL*], we have $(s, h) \models_{SL} \mathcal{A}$ iff $(s', h') \models_{SL} \mathcal{A}$.*

Note that we can extend this result to the whole SL by considering test formulae of the form $e + i = e' + j$.

3.2 Symbolic Models

We write Σ_{μ} to denote the powerset of $\mathcal{T}_{\mu}$; Σ_{μ} is thought as an alphabet, and elements $a \in \Sigma_{\mu}$ are called *letters*. A symbolic model wrt μ is defined as an infinite sequence $\sigma \in \Sigma_{\mu}^{\omega}$. Symbolic models are abstractions of models from LTL$^{\text{mem}}$: given a model $\rho : \mathbb{N} \to \mathcal{S} \times \mathcal{H}$ and a measure μ, we write $Abs_{\mu}(\rho) : \mathbb{N} \to \Sigma_{\mu}$ to denote the symbolic model wrt μ such that for any t, $Abs_{\mu}(\rho)(t) \overset{\text{def}}{=} \{\mathcal{A} \in \mathcal{T}_{\mu} : \rho, t \models \mathcal{A}[\langle \mathtt{x}, u \rangle \leftarrow \mathsf{X}^{u}\mathtt{x}]\}$.

To a letter a, we associate the formula $\mathcal{A}_a = \bigwedge_{\mathcal{A} \in a} \mathcal{A} \wedge \bigwedge_{\mathcal{A} \notin a} \neg \mathcal{A}$. For σ a symbolic model, and ϕ a formula such that $\mu_{\phi} \leq \mu$, we define the symbolic satisfaction relation $\sigma, t \models_{\mu} \phi$ as satisfaction for models except for the clause about atomic subformulae that becomes: $\sigma, t \models_{\mu} \mathcal{A}$ iff $\models_{\text{SL}} \mathcal{A}_{\sigma(t)} \Rightarrow \mathcal{A}[\mathsf{X}^{u}\mathtt{x} \leftarrow \langle \mathtt{x}, u \rangle]$. We write $\text{L}^{\mu}(\phi)$ to denote the set of symbolic models σ wrt μ such that $\sigma, 0 \models_{\mu} \phi$. As a corollary of Lemma 2, we get a soundness result for our abstraction:

Proposition 2. *Let ϕ be a formula of* LTL$^{\text{mem}}$(RF) *[resp. of* LTL$^{\text{mem}}$(CL)*] and $\mu_{\phi} \leq \mu$. For any model ρ, we have that $\rho \models \phi$ iff $Abs_{\mu}(\rho) \models \phi$ [resp. $Abs_{\mu[w \leftarrow 0]}(\rho) \models \phi$].*

Note that Abs_{μ} is not surjective; we note L^{μ}_{sat} the set of symbolic models wrt μ that are abstractions of some model of LTL$^{\text{mem}}$. Consequently, ϕ in LTL$^{\text{mem}}$(RF) is satisfiable iff $\text{L}^{\mu_{\phi}}(\phi) \cap \text{L}^{\mu_{\phi}}_{sat}$ is nonempty.

3.3 ω-Regularity and PSPACE Upper Bound

In order to show that SAT(RF) and SAT(CL) are in PSPACE we shall explain why testing the nonemptiness of $\mathrm{L}^{\mu_\phi}(\phi) \cap \mathrm{L}^{\mu_\phi}_{sat}$ can be done in PSPACE. Below we treat explicitly the case for RF. For CL, replace every occurrence of μ_ϕ by $\mu_\phi[w \leftarrow 0]$. To do so, we show that each language can be recognized by an exponential-size Büchi automaton satisfying the good properties to establish the PSPACE upper bound. If $\mathbb{A}$ is a Büchi automaton, we note $\mathrm{L}(\mathbb{A})$ the language recognized by $\mathbb{A}$. Following [VW94, DD07], let $\mathbb{A}$ be the generalized Büchi automaton defined by the structure $(\Sigma, Q, \delta, I, \mathcal{F})$ s.t.:

– Q is the set of so-called atoms of ϕ, that are sets of temporal formulae included in the so-called closure set $cl(\phi)$ (see [VW94]), $I = \{X \in Q : \phi \in X\}$, and $\Sigma = \Sigma_\mu$.
– $X \xrightarrow{a} Y$ iff 1. for every atomic formula $\mathcal{A}$ of X, $\models_{\mathrm{SL}} \mathcal{A}_a \Rightarrow \mathcal{A}[\mathsf{X}^u\mathtt{x} \leftarrow \langle \mathtt{x}, u\rangle]$.
  2. for every $\mathsf{X}\phi' \in cl(\phi)$, $\mathsf{X}\phi' \in X$ iff $\phi' \in Y$.
– Let $\{\phi_1 \mathsf{U} \phi'_1, \ldots, \phi_n \mathsf{U} \phi'_n\}$ be the set of until formulae in $cl(\phi)$. We pose $\mathcal{F} = \{F_1, \ldots, F_n\}$ where $F_i = \{X \in Q : \phi_i \mathsf{U} \phi'_i \notin X \text{ or } \phi'_i \in X\}$ for $i \in \{1, \ldots, n\}$.

Let $\mathbb{A}^\mu_\phi$ be the Büchi automaton equivalent to the generalized Büchi automaton $\mathbb{A}$. It is easy to observe that $\mathbb{A}^{\mu_\phi}_\phi$ has an exponential amount of states in the size of ϕ and its transition relation can be checked in polynomial space in the size of ϕ. Moreover,

Lemma 3. *Let ϕ in* $\mathrm{LTL}^{\mathrm{mem}}(\mathrm{RF})$ *[resp.* $\mathrm{LTL}^{\mathrm{mem}}(\mathrm{CL})$*] and $\mu \geq \mu_\phi$ [resp. $\mu \geq \mu_\phi[w \leftarrow 0]$]. Then,* $\mathrm{L}(\mathbb{A}^\mu_\phi) = \mathrm{L}^\mu(\phi)$.

We can also build a Büchi automaton $\mathbb{A}^\mu_{sat}$ such that $\mathrm{L}(\mathbb{A}^\mu_{sat}) = \mathrm{L}^\mu_{sat}$. $\mathbb{A}^\mu_{sat}$ is defined as $(\Sigma, Q, \delta, I, F)$, where $\Sigma = \Sigma_\mu$, $Q = \Sigma_\mu$, $F = I = Q$ and $a \xrightarrow{a'} a''$ iff:

1. $\mathcal{A}_a, \mathcal{A}_{a''}$ are satisfiable, and $a = a'$,
2. for every formula $\langle \mathtt{x}, u\rangle = \langle \mathtt{x}', u'\rangle \in \mathcal{T}_\mu$ with $u, u' \geq 1$, $\langle \mathtt{x}, u\rangle = \langle \mathtt{x}', u'\rangle \in a$ iff $\langle \mathtt{x}, u-1\rangle = \langle \mathtt{x}', u'-1\rangle \in a''$.

If $\mu = \mu_\phi$, then $\mathbb{A}^\mu_{sat}$ is of exponential-size in $|\phi|$ and the transition relation can be checked in polynomial space in $|\phi|$. More importantly, this automaton recognizes satisfiable symbolic models.

Lemma 4. *Let ϕ in* $\mathrm{LTL}^{\mathrm{mem}}(\mathrm{RF})$ *[resp.* $\mathrm{LTL}^{\mathrm{mem}}(\mathrm{CL})$*] and $\mu = \mu_\phi$ [resp. $\mu = \mu_\phi[w \leftarrow 0]$]. Then,* $\mathrm{L}(\mathbb{A}^\mu_{sat}) = \mathrm{L}^\mu_{sat}$.

This lemma is essential and it is not possible to extend it to the whole logic $\mathrm{LTL}^{\mathrm{mem}}$ even by allowing test formulae of the form $\mathtt{x} + i = \mathtt{y} + j$ since we would need automata with counters. Now, we can state our main complexity result.

Theorem 1. SAT(RF) *and* SAT(CL) *are* PSPACE*-complete.*

Proof. (sketch) The lower bound is from LTL [SC85]. Let ϕ be an instance formula of SAT(RF) (for SAT(CL) replace below μ_ϕ by $\mu_\phi[w \leftarrow 0]$). As seen earlier, ϕ is satisfiable iff $\mathrm{L}^{\mu_\phi}(\phi) \cap \mathrm{L}^{\mu_\phi}_{sat}$ is nonempty. Hence, ϕ is satisfiable iff $\mathrm{L}(\mathbb{A}^{\mu_\phi}_{\phi}) \cap \mathrm{L}(\mathbb{A}^{\mu_\phi}_{sat}) \neq \emptyset$. The intersection automaton is of exponential size in the size of ϕ and can be checked nonempty by a nondeterministic on-the-fly algorithm. Since nonemptiness problem for Büchi automata is NLOGSPACE-complete and the transition relation in the intersection automaton can be checked in polynomial space in $|\phi|$, we obtain a nondeterministic polynomial space algorithm for testing satisfiability of ϕ. By Savitch's theorem, we get the PSPACE upper bound. □

3.4 Other Problems in PSPACE

Let Frag be either the classical fragment or the record fragment. Lemma 1 provides a reduction from $\mathrm{MC}^{ct}_{init}(\mathrm{Frag})$ to $\mathrm{SAT}^{ct}_{init}(\mathrm{Frag})$ based on a program-as-formula encoding. As we will see now, we may also reduce $\mathrm{SAT}^{ct}_{init}(\mathrm{Frag})$ to $\mathrm{SAT}(\mathrm{Frag})$ internalizing an approximation of the initial memory state whose logical language cannot distinguish from the initial memory state. As a consequence, the PSPACE upper bound for $\mathrm{SAT}(\mathrm{Frag})$ entails the PSPACE upper bound for both $\mathrm{SAT}^{ct}_{init}(\mathrm{Frag})$ and $\mathrm{MC}^{ct}_{init}(\mathrm{Frag})$.

Proposition 3. *The problems* $\mathrm{SAT}^{ct}_{init}(\mathrm{RF})$, $\mathrm{MC}^{ct}_{init}(\mathrm{RF})$, $\mathrm{SAT}^{ct}_{init}(\mathrm{CL})$ *and* $\mathrm{MC}^{ct}_{init}(\mathrm{CL})$ *are* PSPACE*-complete.*

Proof. By Lemma 1 and since $\mathrm{SAT}^{ct}_{init}(\mathrm{RF})$ is known to be PSPACE-hard, it remains to establish the PSPACE upper bound for $\mathrm{SAT}^{ct}_{init}(\mathrm{RF})$.

Given a formula ϕ and an initial memory state (s, h), we shall build in polynomial-time a formula $\phi^{ct}_{s,h}$ in $\mathrm{SAT}(\mathrm{RF})$ such that ϕ is satisfiable in a model with initial memory state (s, h) and constant heap iff $\phi^{ct}_{s,h}$ is satisfiable by a general model. Since we have shown that $\mathrm{SAT}(\mathrm{RF})$ is in PSPACE, this guarantees that $\mathrm{SAT}^{ct}_{init}(\mathrm{RF})$ is in PSPACE. The idea of the proof is to internalize the initial memory state and the fact that the heap is constant in the logic $\mathrm{SAT}(\mathrm{RF})$. Actually, one cannot exactly express that the heap is constant (see details below) but the approximation we use will be sufficient for our purpose.

Apart from the variables of ϕ, the formula $\phi^{ct}_{s,h}$ is built over additional variables in $V = \{\mathtt{x}_i : i \in \mathtt{dom}(h) \cup \mathtt{Im}(s)\} \cup \{\mathtt{x}_{i,l} : i \in \mathtt{dom}(h), l \in \mathtt{dom}(h(i))\}$. The formula $\phi^{ct}_{s,h}$ is of the form $\mathsf{G}(\psi_1 \wedge \psi_2 \wedge \psi_3) \wedge \psi_s \wedge \psi'$, where the subformulae are defined as follows.

- ψ_1 states that the heap is almost equal to h since we cannot forbid in the logical language additional labels ($\mathtt{dom}(h) = \{i_1, \ldots, i_k\}$):
  $\psi_1 \overset{\text{def}}{=} (\bigwedge_{l \in \mathtt{dom}(h(i_1))} \mathtt{x}_{i_1} \overset{l}{\mapsto} \mathtt{x}_{i_1,l}) * \ldots * (\bigwedge_{l \in \mathtt{dom}(h(i_k))} \mathtt{x}_{i_k} \overset{l}{\mapsto} \mathtt{x}_{i_k,l})$.
- ψ_2 states which variables are equal and which ones are not, depending on the initial memory state. By way of example, for $i \neq j \in \mathtt{dom}(h)$, a conjunct of ψ_2 is $\mathtt{x}_i \neq \mathtt{x}_j$. Similarly, if $h(i)(l) = j$ and $j \in \mathtt{dom}(h)$ then $\mathtt{x}_{i,l} = \mathtt{x}_j$ is a conjunct of ψ_2. Details are omitted.
- ψ_3 states that the auxiliary variables remain constant: $\bigwedge_{\mathtt{x} \in V} \mathtt{x} = \mathsf{X}\mathtt{x}$.

– The formula ψ' is obtained from ϕ by replacing each occurrence of $\mathtt{x} \overset{l}{\hookrightarrow} e$ by

$$\mathtt{x} \overset{l}{\hookrightarrow} e \wedge \bigwedge_{i \in \mathtt{dom}(h), l \notin \mathtt{dom}(h(i))} \mathtt{x} \neq \mathtt{x}_i.$$

The additional conjunct is useful because our logical language cannot state that a label is not in the domain of some allocated address.

– ψ_s states constraints about the initial store s: $\psi_s \overset{\text{def}}{=} \bigwedge_{\mathtt{x} \in \phi} \mathtt{x} = \mathtt{x}_{s(\mathtt{x})}$.

It is then easy to check that ϕ is satisfiable in a model with initial memory state (s, h) with constant heap iff $\phi^{ct}_{s,h}$ is satisfiable by a general model.

As far as the results for the classical fragment are concerned, by Lemma 1, there is a logspace reduction from $\mathrm{MC}^{ct}_{init}(\mathrm{CL})$ to $\mathrm{SAT}^{ct}_{init}(\mathrm{CL})$ and as done above one can reduce $\mathrm{SAT}^{ct}_{init}(\mathrm{CL})$ to $\mathrm{SAT}(\mathrm{CL})$. □

4 Undecidability Results

In this section, we show several undecidability results by using reduction from problems for Minsky machines. So, first, we recall that a Minsky machine M consists of two counters C_1 and C_2, and a sequence of $n \geq 1$ instructions, each of which may increment or decrement one of the counters, or jump conditionally upon of the counters being zero. The l^{th} instruction (l is its location counter) has one of the following forms either "l: $\mathsf{C}_i := \mathsf{C}_i + 1$; goto l'" or "l: if $\mathsf{C}_i = 0$ then goto l' else $\mathsf{C}_i := \mathsf{C}_i - 1$; goto l''". In a nondeterministic machine, after an incrementation or a decrementation a nondeterministic choice of the form "goto l_1 or goto l_2" is performed.

The configurations of M are triples (l, c_1, c_2), where $1 \leq l \leq n$, $c_1 \geq 0$, and $c_2 \geq 0$ are the current values of the location counter and the two counters C_1 and C_2, respectively. The consecution relation on configurations is defined in the obvious way. A computation of M is a sequence of related configurations, starting with the initial configuration $(1, 0, 0)$.

Different encodings of counters are used here. For instance, in [BFLS06], a counter C with value n is represented by a list of length n pointed to by a $\mathtt{x}$ dedicated to C. The same idea is used in the proof of Proposition 4 below. In order to show undecidability of SAT(SL) we alternatively encode counters by relying on pointer arithmetic and properties of heaps. Programs without destructive updates can simulate finite computations of Minsky machines by guessing at the start of the computation the maximal value of counters (encoded by a list of the length of the maximal value). As a consequence,

Proposition 4. $\mathrm{SAT}^{ct}(\mathrm{LF})$ *and* $\mathrm{MC}^{ct}(\mathrm{LF})$ *are* Σ^0_1*-complete.*

By constrast, programs with destructive update can work with unbounded heaps, and by using the representation of counters as above, they can faithfully simulate a Minsky machine even if an empty heap is the initial heap. Because LTL can express repeated accessibility, Σ^1_1-hardness can be obtained.

Proposition 5. *The problems* MC(LF) *and* MC_{init}(LF) *are* Σ^1_1*-complete.*

Let us briefly explain how to encode incrementation and decrementation with separating connectives and pointer arithmetic. Observe that expressions of the form $\mathtt{x} = \mathtt{y} + 1$ are not allowed in the logical language. We solve this point in two different ways: using non-aliasing expressed by the separating conjunction, and using the precise pointing assertion $\mathtt{x} \overset{next}{\mapsto} \eta$ stating that the heap contains only one cell, in conjunction with the $-\!\!*$ operator.

$$\begin{aligned}
\phi^{*}_{\mathtt{x}++} &= (\mathsf{X}\mathtt{x} \overset{next}{\hookrightarrow} \mathtt{null} \wedge \mathtt{x}+1 \overset{next}{\hookrightarrow} \mathtt{null}) \wedge \neg(\mathsf{X}\mathtt{x} \overset{next}{\hookrightarrow} \mathtt{null} * \mathtt{x}+1 \overset{next}{\hookrightarrow} \mathtt{null})\\
\phi^{*}_{\mathtt{x}--} &= (\mathsf{X}\mathtt{x}+1 \overset{next}{\hookrightarrow} \mathtt{null} \wedge \mathtt{x} \overset{next}{\hookrightarrow} \mathtt{null}) \wedge \neg(\mathsf{X}\mathtt{x}+1 \overset{next}{\hookrightarrow} \mathtt{null} * \mathtt{x} \overset{next}{\hookrightarrow} \mathtt{null})\\
\phi^{-\!*}_{\mathtt{x}++} &= \mathtt{emp} \;\;\wedge\;\; \big((\mathsf{X}\mathtt{x} \overset{next}{\mapsto} \mathtt{null}) -\!\!* \mathtt{x}+1 \overset{next}{\mapsto} \mathtt{null}\big)\\
\phi^{-\!*}_{\mathtt{x}--} &= \mathtt{emp} \;\;\wedge\;\; \big((\mathtt{x} \overset{next}{\mapsto} \mathtt{null}) -\!\!* \mathsf{X}\mathtt{x}+1 \overset{next}{\mapsto} \mathtt{null}\big)
\end{aligned}$$

The formulae based on the separating conjunction correctly express incrementation and decrementation when the cells at index $\mathtt{x}, \mathtt{x}+1, \mathtt{x}-1$ are allocated, whereas formulae based on the operator $-\!\!*$ do not need the same assumption.

Let $\mathrm{SAT}^?_?$(SL) be any satisfiability problem among the four variants.

Proposition 6. $\mathrm{SAT}^?_?$(SL) *is* Σ^1_1*-complete.*

Proof. We reduce the recurrence problem for nondeterministic Minsky machines [AH94] to $\mathrm{SAT}^?_?$(SL). Let ϕ_0 be the formula $\mathsf{G}(\mathtt{emp} \wedge \bigwedge_{i=1}^{2} (\mathtt{x}_i \neq \mathtt{null}))$. Incrementation and decrementation are performed thanks to $\phi^{-\!*}_{\mathtt{x}++}$ and $\phi^{-\!*}_{\mathtt{x}--}$, respectively. For any model ρ such that $\rho, 0 \models \phi_0$, and for any t, we have $\rho, t \models \phi^{-\!*}_{\mathtt{x}_i++}$ iff $s_t(\mathtt{x}_i) + 1 = s_{t+1}(\mathtt{x}_i)$. Hence, we have a means to encode incrementation. Similarly, $\rho, t \models \phi^{-\!*}_{\mathtt{x}_i--}$ and $s_t(\mathtt{x}_i) > 0$ iff $s_t(\mathtt{x}_i) - 1 = s_{t+1}(\mathtt{x}_i)$. The fact that a counter does not change is encoded by $\mathtt{x}_i = \mathsf{X}\mathtt{x}_i$. Given that $\phi_1 = \mathsf{G}(\mathtt{x}_{zero} = \mathsf{X}\mathtt{x}_{zero} \wedge \mathtt{x}_{zero} \neq \mathtt{null})$ holds, zero tests are encoded by $\mathtt{x}_i = \mathtt{x}_{zero}$.

Given a nondeterministic Minsky machine M, we write ψ_l to denote the formula encoding instruction l. For intance for the instruction "l: if $\mathsf{C}_1 = 0$ then goto l' else $\mathsf{C}_1 := \mathsf{C}_1 - 1$; goto l'_1 or goto l'_2" ψ_l is equal to $\mathsf{G}((l \wedge \mathtt{x}_1 \neq \mathtt{x}_{zero}) \Rightarrow (\mathtt{x}_2 = \mathsf{X}\mathtt{x}_2 \wedge (\mathsf{X}l'_1 \vee \mathsf{X}l'_2) \wedge \phi^{-\!*}_{\mathtt{x}_1--})) \wedge \mathsf{G}((l \wedge \mathtt{x}_1 = \mathtt{x}_{zero}) \Rightarrow (\mathtt{x}_1 = \mathsf{X}\mathtt{x}_1 \wedge \mathtt{x}_2 = \mathsf{X}\mathtt{x}_2 \wedge \mathsf{X}l'))$. Hence, $(\mathtt{x}_1 = \mathtt{x}_2 = \mathtt{x}_{zero}) \wedge \phi_0 \wedge \phi_1 \wedge \bigwedge_l \psi_l \wedge \mathsf{GF}n$ is satisfiable iff M has a computation with location counter n repeated infinitely often. □

Proposition 7. *The problem* $\mathrm{SAT}(\mathrm{SL} \setminus \{-\!\!*\})$ *is* Σ^1_1*-complete.*

The proof of Proposition 7 is similar to the proof of Theorem 6 except that incrementation and decrementation are performed with the formulae $\phi^{*}_{\mathtt{x}++}$ and $\phi^{*}_{\mathtt{x}--}$, respectively.

5 Conclusion

In the paper, we have introduced a temporal logic $\mathrm{LTL}^{\mathrm{mem}}$ for which assertion language is quantifier-free separation logic. Figure 1 contains a summary of the

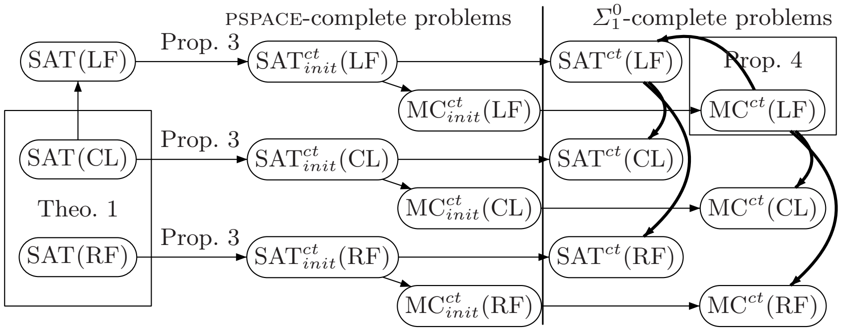


Fig. 1. Complexity of reasoning tasks with $\mathrm{LTL}^{\mathrm{mem}}$

complexity results about satisfiability and model-checking problems for the fragments LF, CL and RF. Σ^1_1-completeness results for $\mathrm{SAT}^?_?(\mathrm{SL})$, $\mathrm{SAT}(\mathrm{SL} \setminus \{-\!*\})$ and MC(LF) can be found in Propositions 6, 7, and 5, respectively. A thin and straight [resp. bold and curved] arrow between a source problem and a target problem means that the upper [resp. lower] bound for the target problem is shown thanks to the upper [resp. lower] bound for the source problem.

Finally, extending $\mathrm{LTL}^{\mathrm{mem}}$ with a special propositional variable $\mathtt{heap}^{=}$ stating that the current heap is equal to the next one, can lead to undecidability (look at the problems of the form $\mathrm{SAT}^{ct}_?(\mathrm{Frag})$). However, it is open whether satisfiability becomes decidable if we restrict the interplay between the "until" operator U and $\mathtt{heap}^{=}$, for instance to forbid subformulae of the form $\mathsf{G}\ \mathtt{heap}^{=}$ with positive polarity.

References

[AH94] R. Alur and T.A. Henzinger. A really temporal logic. *JACM*, 41:181–204, 1994.

[BBH+06] A. Bouajjani, M. Bozga, P. Habermehl, R. Iosif, P. Moro, and T. Vojnar. Programs with lists are counter automata. In *CAV'06*, volume 4144 of *LNCS*, pages 517–531. Springer, 2006.

[BCMS01] O. Burkart, D. Caucal, F. Moller, and B. Steffen. Verification of infinite structures. In *Handbook of Process Algebra*, pages 545–623. Elsevier, 2001.

[BCO05] J. Berdine, C. Calcagno, and P. W. O'Hearn. Symbolic execution with separation logic. *APLAS'05*, 3780:52–68, 2005.

[BDL07] R. Brochenin, S. Demri, and E. Lozes. Reasoning about sequences of memory states. Technical report, LSV, ENS de Cachan, 2007.

[BFLS06] S. Bardin, A. Finkel, E. Lozes, and A. Sangnier. From pointer systems to counter systems using shape analysis. *5th International Workshop on Automated Verification of Infinite-State Systems (AVIS'06)*, 2006.

[BFN04] S. Bardin, A. Finkel, and D. Nowak. Toward symbolic verification of programs handling pointers. In *3rd International Workshop on Automated Verification of Infinite-State Systems (AVIS'04)*, 2004.

[BIL04] M. Bozga, R. Iosif, and Y. Lakhnech. On logics of aliasing. In *SAS'04*, volume 3148 of *LNCS*, pages 344–360. Springer, 2004.

[CC00] H. Comon and V. Cortier. Flatness is not a weakness. *CSL'00*, 1862:262–276, 2000.

[CGH05] C. Calcagno, Ph. Gardner, and M. Hague. From separation logic to first-order logic. In *FOSSACS'05*, volume 3441 of *LNCS*, pages 395–409. Springer, 2005.

[CYO01] C. Calcagno, H. Yang, and P. O'Hearn. Computability and complexity results for a spatial assertion language for data structures. In *FST&TCS'01*, volume 2245 of *LNCS*, pages 108–119. Springer, 2001.

[DD07] S. Demri and D. D'Souza. An automata-theoretic approach to constraint LTL. *Information and Computation*, 205(3):380–415, 2007.

[DKR04] D. Distefano, J.-P. Katoen, and A. Rensink. Who is pointing when to whom? on the automated verification of linked list structures. In *FST&TCS'04*, volume 3328 of *LNCS*, pages 250–262. Springer, 2004.

[GKWZ03] D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-dimensional modal logics: theory and applications*. CUP, 2003.

[GM05] D. Galmiche and D. Mery. Characterizing provability in BI's pointer logic through resource graphs. In *LPAR'05*, volume 3835 of *LNCS*, pages 459–473. Springer, 2005.

[IO01] S. Ishtiaq and P. O'Hearn. BI as an assertion language for mutable data structures. In *POPL'01*, pages 14–26, 2001.

[JJKS97] J. Jensen, M. Jorgensen, N. Klarlund, and M. Schwartzbach. Automatic verification of pointer programs using monadic second-order logic. In *PLDI'97*, pages 226–236. ACM, 1997.

[LAS00] T. Lev-Ami and M. Sagiv. TVLA: A system for implementing static analyses. In *SAS'00*, pages 280–301, 2000.

[Loz04] E. Lozes. Separation logic preserves the expressive power of classical logic. In *2nd Workshop on Semantics, Program Analysis, and Computing Environments for Memory Management (SPACE'04)*, 2004.

[Pnu77] A. Pnueli. The temporal logic of programs. In *FOCS'77*, pages 46–57. IEEE, 1977.

[Rey02] J.C. Reynolds. Separation logic: a logic for shared mutable data structures. In *LICS'02*, pages 55–74. IEEE, 2002.

[SC85] A. Sistla and E. Clarke. The complexity of propositional linear temporal logic. *JACM*, 32(3):733–749, 1985.

[VW94] M. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115:1–37, 1994.

[YRSW03] E. Yahav, Th. Reps, M. Sagiv, and R. Wilhelm. Verifying temporal heap properties specified via evolution logic. In *ESOP'03*, volume 2618 of *LNCS*, pages 204–22. Springer, 2003.

Cut Elimination in Deduction Modulo by Abstract Completion

Guillaume Burel[1] and Claude Kirchner[2]

[1] Université Henri Poincaré & LORIA[3]
guillaume.burel@ens-lyon.org
[2] INRIA & LORIA[3]
Claude.Kirchner@loria.fr
[3] UMR 7503 CNRS-INPL-INRIA-Nancy2-UHP

Abstract. Deduction Modulo implements Poincaré's principle by identifying deduction and computation as different paradigms and making their interaction possible. This leads to logical systems like the sequent calculus or natural deduction modulo. Even if deduction modulo is logically equivalent to first-order logic, proofs in such systems are quite different and dramatically simpler with one cost: cut elimination may not hold anymore. We prove first that it is even undecidable to know, given a congruence over propositions, if cuts can be eliminated in the sequent calculus modulo this congruence.

Second, to recover the cut admissibility, we show how computation rules can be added following the classical idea of completion *a la* Knuth and Bendix. Because in deduction modulo, rewriting acts on terms as well as on propositions, the objects are much more elaborated than for standard completion. Under appropriate hypothesis, we prove that the sequent calculus modulo is an instance of the powerful framework of *abstract canonical systems* and that therefore, cuts correspond to critical proofs that abstract completion allows us to eliminate.

In addition to an original and deep understanding of the interactions between deduction and computation and of the expressivity of abstract canonical systems, this provides a mechanical way to transform a sequent calculus modulo into an equivalent one admitting the cut rule, therefore extending in a significant way the applicability of mechanized proof search in deduction modulo.

Keywords: Knuth-Bendix completion, automated deduction and interactive theorem proving, cut elimination, deduction modulo, proof ordering, abstract canonical system.

1 Introduction

The complementarity and interaction between computation and deduction is known since at least Henri Poincaré, and deduction modulo [16] is a way to present first-order logic taking advantage from this complementarity. Deduction modulo is at the heart of proof assistants and proof search methods, either implicitly or explicitly (see for instance [24,3,16,5]) and getting a deep understanding of its logical behavior is of prime interest either for theoretical or practical purposes.

In deduction modulo, computations are modeled by a congruence relation between terms and between propositions. The logical deductions are done modulo this congruence that is often better represented by a rewrite relation over first-order terms and propositions, leading to the asymmetric sequent calculus [14].

In the sequent calculus modulo, the Hauptsatz, *i.e.* the fact that cuts are not needed to build proofs, is no longer true as one can see from an example derived from Crabbé's proof of the non-normalization of Zermelo's theory [8] (see for instance [16]). But we know that the admissibility of the cut rule is fundamental for at least two related reasons: first, if a system admits the cut rule, then the formulæ needed to build a sequent calculus proof of some sequent are subformulæ[1] of the ones appearing in it, so that the search space is, in a sense, limited. Such proofs are sometimes called *analytic* [14]. The tableaux method is based on this fact, and for instance TaMeD [5], a tableaux method based on deduction modulo, is shown to be complete only for cut-free systems. On the other hand, it has been shown [21] that a proof search method for deduction modulo like ENAR [16]—which generalizes resolution and narrowing—is equivalent to the cut-free fragment of deduction modulo. ENAR is therefore complete if and only if the cut rule is admissible.

So on the one hand, we like to have a powerful congruence but this may be at the price of loosing cut admissibility. How can we get both? Gilles Dowek has shown [14] that cut admissibility is equivalent to the confluence of the rewrite system, provided only first-order *terms* are rewritten. It is no longer true when *propositions* are also rewritten, and the cut admissibility is in that case a stronger notion than confluence. Therefore he wanted to build a generalized completion procedure whose input is a rewrite system over first-order terms and propositions and that computes a rewrite system such that the associated sequent calculus modulo admits the cut rule. Such a completion procedure was proposed for the quantifier free case in [13], based on the construction of a model for the theory associated with the rewrite system.

To fully solve this question, including *unlimited* use of quantifiers, we propose here a quite different approach based on the notion of abstract canonical system and inference introduced in [12,4]. This abstract framework is based on a proof ordering whose goal is to apprehend the notion of proof quality from which the notions of canonicity, completeness and redundancy follow up. It is shown to be adapted to existing completion procedures such as ground completion [10] and standard (a.k.a. Knuth-Bendix [23]) completion [6].

To present the general idea of our approach, let us consider the simple example of Crabbé's axiom [8] $A \Leftrightarrow B \wedge \neg A$.[2] Can we find, for the sequent calculus

[1] In the case of deduction modulo, the intuitive notion of subformula must of course take into account the equivalence relation.

[2] In [8], A represents $r_s \in r_s$ and B is $r_s \in s$ where $r_s \overset{!}{=} \{x \in s : x \notin x\}$. Then, there is a proof of $r_s \notin s$ in Zermelo's set theory that is not normalizing.

modulo the rewrite system $A \to B \wedge \neg A$, a provable sequent without any cut-free proof? Indeed, let us try to build a minimal example. We will show in Prop. 4 that such a proof, in its simplest form, is necessarily of the shape:

$$\dfrac{\dfrac{\overset{\vdots}{A, B \wedge \neg A \vdash}}{A \vdash}\ \uparrow\text{-l} \qquad \dfrac{\overset{\vdots}{\vdash B \wedge \neg A, A}}{\vdash A}\ \uparrow\text{-r}}{\vdash}\ \mathsf{Cut}(A)$$

where the rules labeled "↑-r" and "↑-l" allow to apply the oriented axioms respectively on the right or on the left. In order to validate this proof pattern, we have to check if it is possible to close both sides of the proof tree, possibly adding informations in the initial sequent.

First, we can trivially close the left part as follows:

$$\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\ \mathsf{Axiom}}{A, B, \neg A \vdash}\ \neg\text{-l}}{A, B \wedge \neg A \vdash}\ \wedge\text{-l} \quad .$$

Second, to close the right part, we must have a proof in the form:

$$\dfrac{\vdash B, A \qquad \dfrac{\dfrac{}{A \vdash A}\ \mathsf{Axiom}}{\vdash \neg A, A}\ \neg\text{-r}}{\vdash B \wedge \neg A, A}\ \wedge\text{-r} \quad .$$

To enforce the proof of $\vdash B, A$, we must add either A or B to the left of the sequent, and we only have to consider B, since we have cut around A. We obtain the critical proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\ \mathsf{Axiom}}{A, B, \neg A \vdash}\ \neg\text{-l}}{B, A, B \wedge \neg A \vdash}\ \wedge\text{-l}}{B, A \vdash}\ \uparrow\text{-l} \qquad \dfrac{\dfrac{\dfrac{}{B \vdash B, A}\ \mathsf{Axiom} \qquad \dfrac{\dfrac{}{B, A \vdash A}\ \mathsf{Axiom}}{B \vdash \neg A, A}\ \neg\text{-r}}{B \vdash B \wedge \neg A, A}\ \wedge\text{-r}}{B \vdash A}\ \uparrow\text{-r}}{B \vdash}\ \mathsf{Cut}(A) \quad .$$

We can also easily show that there is no cut-free proof of $B \vdash$, simply because no inference rule is applicable to it except Cut. If we want to have a cut-free proof, we need to make B reducible by the congruence, hence the idea to complete the initial system with a new rule which is a logical consequence of the current system. In our case, we must therefore add the rule $B \to \bot$.

With this new rule, we will show that there is no more critical proof and that therefore the sequent calculus modulo the proposition rewrite system

$$\begin{cases} A \to B \wedge \neg A \\ B \to \bot \end{cases}$$

admits the cut rule and has the same expressive power as the initial one.

The study of this question indeed reveals general properties of the sequent calculus modulo and our contributions are the following:

– We provide an appropriate Noetherian ordering on the proofs of the sequent calculus modulo a rewrite system; This ordering allows us to set on the proof space of sequent calculus modulo a structure of abstract canonical system;
– We characterize the critical proofs in deduction modulo as simple cuts;
– By an appropriate correspondence between sequents and rewrite systems, we establish a precise correspondence between the limit of a completion process and a cut free sequent calculus;
– We show the applicability of the general results, in particular on sequent calculus modulo rewrite systems involving quantifiers, therefore generalizing all previously known results;
– We establish the limits of our approach by proving the undecidability of cut admissibility and of the search for critical proofs.

As an important by-product of these results, we demonstrate the expressive power of abstract canonical systems (ACS for short).

The next section will present the minimal knowledge needed on deduction modulo and abstract canonical systems to make the paper self-contained, and states the undecidability of the admissibility of the cut rule in deduction modulo. In Sect. 3, we show how to set, on the proof space of sequent calculus modulo, a structure of abstract canonical system. In particular we make precise why the postulates of ACS are fulfilled. This allows us in Sect. 4 to characterize the critical proofs of deduction modulo and to set-up the completion process as the appropriate (and indeed non-trivial) instance of the abstract completion process. We also provide an algorithm to systematically transform a set of sequents into an appropriate set of proposition rewrite rules, therefore making the whole framework operational. We conclude after presenting in more details Crabbé's example as well as several examples involving quantifiers. All proofs can be found in the full version of this paper [7].

2 Prerequisites

2.1 Rewritings

We define here how propositions are rewritten in deduction modulo.

We use standard definitions for terms, predicates, propositions (with connectors $\neg, \Rightarrow, \wedge, \vee$ and quantifiers $\forall, \exists$), substitutions, term rewrite rules and term rewriting, as can be found in [2,19]. The set of terms built from a signature Σ and a set of variables V is denoted by $\mathcal{T}(\Sigma, V)$, the replacement of a variable x by a term t in a proposition P by $\{t/x\}P$, the application of a substitution σ in a proposition P by σP.

An atomic proposition $A(s_1, \dots, s_i, \dots, s_n)$ can be rewritten to the atomic proposition $A(s_1, \dots, t_i, \dots, s_n)$ by a term rewrite rule $l \to r$ if s_i can be rewritten to t_i by $l \to r$.

A *proposition rewrite rule* is the pair of an atomic proposition A and a proposition P, such that all free variables of P appear in A. It is denoted $A \to P$.

A *proposition rewrite system* is a set of proposition rewrite rules. The set of all proposition rewrite systems is denoted $\mathcal{PRS}$.

An atomic proposition A can be rewritten to a proposition P by a proposition rewrite rule $B \rightarrow Q$ if there exists some substitution σ such that $\sigma B = A$ and $\sigma Q = P$. Semantically, this proposition rewrite relation must be seen as a logical equivalence between propositions.

Note that we do not define how to rewrite non-atomic propositions by proposition rewrite rules, as in [16], because this can be simulated in the sequent calculus modulo we present in the next section.

In the following, the term rewrite system used in addition to all the proposition rewrite systems we will consider is fixed. It is supposed to be *terminating and confluent*. It will be denoted $R_{\mathcal{T}(\Sigma,V)}$.

The *subformula relation* $\succ$ is the least transitive relation such that:

- $P \succ P_i$ if $P = P_1 \wedge P_2$, $P = P_1 \vee P_2$ or $P = \neg P_1$;
- $P \succ \{t/x\}Q$ if $P = \forall x.\ Q$ or $P = \exists x.\ Q$;
- $P \succ Q$ if P can be rewritten to Q by $R_{\mathcal{T}(\Sigma,V)}$

for all terms t, variables x and propositions P, Q, P_1, P_2. It is well-founded because of the termination of $R_{\mathcal{T}(\Sigma,V)}$.

2.2 Sequent Calculus Modulo

Sequent calculus modulo can be seen as an extension of the sequent calculus of Gentzen [20]. We will use the denominations of [19].

A *sequent* is a pair of multisets of propositions Γ, Δ. It is denoted by $\Gamma \vdash \Delta$. The sets of all sequents will be denoted $\mathcal{S}$. For a sequent $\Gamma \vdash \Delta$, if $x_1, \ldots, x_n$ are the free variables of Γ, Δ, we will denote by $\mathcal{P}(\Gamma \vdash \Delta)$ the proposition $\forall x_1, \ldots, x_n.\ (\bigwedge \Gamma \Rightarrow \bigvee \Delta)$.

In Fig. 1 we present some inference rules of our *sequent calculus modulo*. They differ from the ones of [14] because the congruence is externalized through specific inference rules $\uparrow$-l and $\uparrow$-r (as can be found in [21]), but there is no contraction or weakening rules. The other logical rules are the one of the standard sequent calculus. For $\forall$-l and $\exists$-r, the quantified formula that is decomposed is kept. *Proofs* are trees labeled by sequents built using these rules, and where all leaves are Axioms. The root sequent is called the *conclusion.* A proof is said to be built in the proposition rewrite system R if all $\uparrow$-l and $\uparrow$-r use only rules that appear in $R \cup R_{\mathcal{T}(\Sigma,V)}$. The set of all proofs will be denoted by $\mathcal{SQM}$.

Cut(P) permits essentially to extend the proof search space with the proposition P. Logical Rules decompose some proposition which is called *principal.* Rewrite Rules, that do not appear in Gentzen's sequent calculus, introduce proposition rewriting into the proof system. Note that only atomic propositions are rewritten, and that we keep the original formula in the sequent.

A proposition rewrite system R is said to *admit* Cut if for all sequents $s \in \mathcal{S}$, s has a proof in R if and only if s has a proof in R without using Cut. It is well-known (Gentzen's Hauptsatz [20], or more accurately [14] because of $R_{\mathcal{T}(\Sigma,V)}$) that $\emptyset$ admits Cut.

Identity Group: $\dfrac{}{\Gamma, P \vdash P, \Delta}\ \mathsf{Axiom}$ $\qquad \dfrac{\Gamma, P \vdash \Delta \quad \Gamma \vdash P, \Delta}{\Gamma \vdash \Delta}\ \mathsf{Cut}(P)$

Logical Rules: $\dfrac{\Gamma \vdash P, \Delta}{\Gamma, \neg P \vdash \Delta}\ \neg\text{-}\mathsf{l}$ $\qquad \dfrac{\Gamma \vdash P, \Delta \quad \Gamma, Q \vdash \Delta}{\Gamma, P \Rightarrow Q \vdash \Delta}\ {\Rightarrow}\text{-}\mathsf{l}$

$\dfrac{\Gamma \vdash \{c/x\}P, \Delta}{\Gamma \vdash \forall x.\ P, \Delta}\ \forall\text{-}\mathsf{r}$ c free in Γ, Δ $\qquad \dfrac{\Gamma \vdash \exists x.\ P, \{t/x\}P, \Delta}{\Gamma \vdash \exists x.\ P, \Delta}\ \exists\text{-}\mathsf{r}$ $t \in \mathcal{T}(\Sigma, V)$

Rewrite Rules: if A can be rewritten to P, either by a term or a proposition rewrite rule (in one step),

$$\dfrac{\Gamma, A, P \vdash \Delta}{\Gamma, A \vdash \Delta}\ \uparrow\text{-}\mathsf{l} \qquad \dfrac{\Gamma \vdash A, P, \Delta}{\Gamma \vdash A, \Delta}\ \uparrow\text{-}\mathsf{r}$$

Fig. 1. Some inference rules of the sequent calculus modulo

It is important to be aware that free variables appearing in a sequent play the same role as fresh constants, because no inference rules can modify them. As a consequence, one can restrict oneself to closed sequents, as indicated in [16, Proposition 1.5].

Proposition 1 (Equivalence). *The sequent calculus modulo (partly) presented in Fig. 1 is equivalent to the Asymmetric Sequent Calculus Modulo of [14].*

In particular, our system has the weakening and the contraction properties:

- if there exists a proof of $\Gamma \vdash \Delta$, then for all propositions P there exist proofs of $\Gamma, P \vdash \Delta$ and $\Gamma \vdash P, \Delta$;
- there exists a proof of $\Gamma, P \vdash \Delta$ if and only if there exists a proof of $\Gamma, P, P \vdash \Delta$, and there exists a proof of $\Gamma \vdash P, \Delta$ if and only if there exists a proof of $\Gamma \vdash P, P, \Delta$.

Our sequent calculus also satisfies Kleene's Lemma:

Lemma 1 (Kleene Lemma [21, Lemme 3.3]). *If a sequent, containing the non-atomic formula P, has a proof (resp. cut-free proof) in R, then it has a proof (resp. cut-free proof) in R whose first rule is a logical rule with principal proposition P.*

We prove also the following new result:

Theorem 1 (Undecidability of the Cut Admissibility). *Given a propositional rewrite system $\mathcal{R}$, it is undecidable to know if $\mathcal{R}$ admits* Cut.

2.3 Abstract Canonical Systems and Inference

The results in this section are extracted from [11,12,4], which should be consulted for motivations, details and proofs.

Let $\mathbb{A}$ be the set of all formulæ over some fixed vocabulary. Let $\mathbb{P}$ be the set of all proofs. These sets are linked by two functions: $[\cdot]^{Pm} : \mathbb{P} \to 2^{\mathbb{A}}$ gives the *premises* in a proof, and $[\cdot]_{Cl} : \mathbb{P} \to \mathbb{A}$ gives its *conclusion*. Both are extended to sets of proofs in the usual fashion. The set of proofs built using assumptions in $A \subseteq \mathbb{A}$ is denoted by

$$Pf(A) \quad \overset{!}{=} \quad \{p \in \mathbb{P} : [p]^{Pm} \subseteq A\} \ .$$

The framework described here is predicated on two *well-founded* partial orderings over $\mathbb{P}$: a *proof ordering* $>$ and a *subproof relation* $\rhd$. They are related by a monotonicity requirement (postulate E). We assume for convenience that the proof ordering only compares proofs with the same conclusion ($p > q \Rightarrow [p]_{Cl} = [q]_{Cl}$), rather than mention this condition each time we have cause to compare proofs.

We will use the term *presentation* to mean a set of formulæ, and *justification* to mean a set of proofs. We reserve the term *theory* for deductively closed presentations:

$$Th\, A \quad \overset{!}{=} \quad [Pf(A)]_{Cl} \quad = \quad \{[p]_{Cl} : p \in \mathbb{P},\ [p]^{Pm} \subseteq A\}.$$

Presentations A and B are *equivalent* ($A \equiv B$) if their theories are identical: $Th\, A = Th\, B$. In addition to this, we assume the two following postulates:

Postulate A (Reflexivity) *For all presentations A:*

$$A \subseteq Th\, A$$

Postulate B (Closure) *For all presentations A:*

$$Th\; Th\, A \subseteq Th\, A$$

We call a proof *trivial* when it proves only its unique assumption and has no subproofs other than itself, that is, if $[p]^{Pm} = \{[p]_{Cl}\}$ and $p \unrhd q \Rightarrow p = q$, where $\unrhd$ is the reflexive closure of the subproof ordering $\rhd$. We denote by $\widehat{a}$ such a trivial proof of $a \in \mathbb{A}$ and by $\widehat{A}$ the set of trivial proofs of each $a \in A$.

We assume that proofs use their assumptions (postulate C), that subproofs don't use non-existent assumptions (postulate D), and that proof orderings are monotonic with respect to subproofs (postulate E):

Postulate C (Trivia) *For all proofs p and formulæ a:*

$$a \in [p]^{Pm} \Rightarrow p \unrhd \widehat{a}$$

Postulate D (Subproofs Premises Monotonicity) *For all proofs p and q:*

$$p \unrhd q \Rightarrow [p]^{Pm} \supseteq [q]^{Pm}$$

Postulate E (Replacement) *For all proofs p, q and r:*

$$p \rhd q > r \Rightarrow \exists v \in Pf([p]^{Pm} \cup [r]^{Pm}).\ p > v \rhd r$$

Postulate E essentially says that replacing one of its subproof by a smaller proof makes a proof smaller. However, the proof v is not necessarily obtained by syntactically replacing q by r in p.

We make no other assumptions regarding proofs or their structure and the proof ordering $>$ is lifted to a quasi-ordering $\gtrsim$ over presentations:

$$A \gtrsim B \text{ if } A \equiv B \text{ and } \forall p \in \mathit{Pf}(A).\ \exists q \in \mathit{Pf}(B).\ p \geq q \ .$$

We define what a *normal-form proof* is, i.e. one of the minimal proofs of $\mathit{Pf}(\mathit{Th}\,A)$:

$$\begin{aligned} \mathit{Nf}(A) &\overset{!}{=} \mu\mathit{Pf}(\mathit{Th}\,A) \\ &\overset{!}{=} \{p \in \mathit{Pf}(\mathit{Th}\,A) \ :\ \neg\exists q \in \mathit{Pf}(\mathit{Th}\,A).\ p > q\} \end{aligned}$$

The *canonical presentation* contains those formulæ that appear as assumptions of normal-form proofs:

$$A^\sharp \overset{!}{=} [\mathit{Nf}(A)]^{Pm} \ .$$

So, we will say that A is *canonical* if $A = A^\sharp$.

A presentation A is *complete* if every theorem has a normal-form proof:

$$\mathit{Th}\,A = [\mathit{Pf}(A) \cap \mathit{Nf}(A)]_{Cl}$$

Canonicity implies completeness, but the converse is not true.

We now consider inference and deduction mechanisms. A *deduction mechanism* $\rightsquigarrow$ is a function from presentations to presentations and we call the relation $A \rightsquigarrow B$ a *deduction step*. A sequence of presentations $A_0 \rightsquigarrow A_1 \rightsquigarrow \cdots$ is called a *derivation*. The *result* of the derivation is, as usual, its *persisting* formulæ:

$$A_\infty \overset{!}{=} \liminf_{j\to\infty} A_j = \bigcup_{j>0} \bigcap_{i>j} A_i \ .$$

A deduction mechanism is *completing* if for each step $A \rightsquigarrow B$, $A \gtrsim B$ and the limit A_∞ is complete.

A completing mechanism can be used to build normal-form proofs of theorems of the initial presentation:

Theorem 2 ([4, Lemma 5.13]). *A deduction mechanism is completing if and only if for all derivations* $A_0 \rightsquigarrow A_1 \rightsquigarrow \cdots$*,*

$$\mathit{Th}\,A_0 \subseteq [\mathit{Pf}(A_\infty) \cap \mathit{Nf}(A_0)]_{Cl} \ .$$

A *critical proof* is a minimal proof which is not in normal form, but whose strict subproofs are:

$$\mathit{Crit}(A) \overset{!}{=} \{p \in \mu\mathit{Pf}(A) \setminus \mathit{Nf}(A) : \forall q \in \mathit{Pf}(A).\ p \rhd q \Rightarrow q \in \mathit{Nf}(A)\}$$

Completing formulæ are conclusions of proofs smaller than critical proofs:

$$\mathit{Comp}(A) \overset{!}{=} \bigcup_{p \in \mathit{Crit}(A)\ \wedge\ p' \text{ is some proof such that } p > p'} [p']^{Pm}$$

In this paper, we use a completing deduction mechanism in the following way:

$$A \rightsquigarrow A \cup \mathsf{C}(A)$$

where $Comp(A) \subseteq \mathsf{C}(A) \subseteq Th\,A$.

Proposition 2 ([11, Lemma 10]). *This deduction mechanism is completing.*

3 Deduction Modulo Is an Instance of ACS

We want to show that the sequent calculus modulo can be seen as an instance of ACS. For this purpose, we have to define what the formulæ, the proofs, the premises and conclusions are, and to give the appropriate orderings. After this, we need to check that the postulates are verified by the defined instance.

3.1 Proofs and Formulae

We aim to obtain cut-free proofs, so that the natural candidate for ACS proofs are sequent calculus proofs. Because of the weakening and contraction properties, we can restrict ourselves to proofs using minimal sets of propositions in their conclusions. More precisely, we can consider only proofs where all the propositions appearing in the conclusion are used as principal propositions somewhere in the proof, or in one of the Axioms.

The completion procedure we want to establish deals with rewrite rules over atomic propositions. Nevertheless, the conclusions of the proofs, from which we want to generate the rewrite rules added by the completion mechanism, are sequents. In other words, sequents must be identified with proposition rewrite systems.

Therefore we suppose that there exists a function between sequents and proposition rewrite systems $Rew : \mathcal{S} \rightarrow \mathcal{PRS}$ such that:

Property 1. For all sequents $\Gamma \vdash \Delta$, $R = Rew(\Gamma \vdash \Delta)$ and $\mathcal{P}(\Gamma \vdash \Delta)$ are strongly compatible:

(a) for all propositions P, Q, $P \overset{*}{\underset{R}{\longleftrightarrow}} Q$ implies that there exists a proof of $\mathcal{P}(\Gamma \vdash \Delta) \vdash P \Leftrightarrow Q$ in $\emptyset$ (i.e. without rewrite rules);
(b) there exists a cut-free proof of $\vdash \mathcal{P}(\Gamma \vdash \Delta)$ in R.

Property 2. For all proposition rewrite systems R, for all sequents s and s', if $Rew(s) = Rew(s')$, then s has a proof (resp. cut-free proof) in R iff s' has a proof (resp. cut-free proof) in R.

Property 1 implies compatibility in the sense of Definition 1.4 of [16], which is the same except that we need here a *cut-free* proof in b).

Section 4.3 provides an instance of such a function Rew.

With respect to the definitions of ACSs (see Sect. 2.3) deduction modulo can be seen as an ACS, in the following way:

— $\mathbb{P}$: *proofs* are sequent calculus proofs using minimal sets of propositions in their conclusion:

$$\mathbb{P} \quad \overset{!}{=} \quad \{p \in \mathcal{SQM} \colon \neg (\exists q \in \mathcal{SQM}.\ \mathrm{Weak}(q, p))\}$$

where Weak(q, p) says that the proof p can be obtained from q by weakening.

— $\mathbb{A}$: *formulæ* are proposition rewrite systems corresponding to some sequent:

$$\mathbb{A} \quad \overset{!}{=} \quad \mathit{Rew}(\mathcal{S}) \ \subseteq \ \mathcal{PRS} \ .$$

— The *conclusion* of an ACS proof is the rewrite system associated by *Rew* to the conclusion of the sequent calculus proof: for all proofs p,

$$[p]_{Cl} \quad \overset{!}{=} \quad \mathit{Rew}(\Gamma \vdash \Delta) \text{ when } p = \cfrac{\vdots \quad \vdots}{\Gamma \vdash \Delta} \ .$$

— The *premises* of a proof are the rewrite systems consisting of the proposition rewrite rules appearing in the proof or its subproofs: for all proofs p,

$$[p]^{Pm} \quad \overset{!}{=} \quad \left\{ \begin{array}{l} \left\{ A \to P \colon \begin{array}{c} \text{there exists a } \uparrow\text{-l or} \\ \uparrow\text{-r using } A \to P \text{ in } q \end{array} \right\} \colon \\ q \text{ is a subproof of } p \end{array} \right\}$$

This definition implies that we consider only proofs using proposition rewrite systems corresponding to some sequent.

3.2 Orderings on Proofs

We define the following (infinite, but Noetherian) precedence $>$: for all formulæ P, Q, if P is greater than Q for the subformula relation, then $\mathsf{Cut}(P) > \mathsf{Cut}(Q)$, and for all other inference rules r of Fig. 1, $\mathsf{Cut}(P) > \mathsf{r}$.

We order proofs using the RPO [9] based on this precedence. Since the precedence is well-founded, so is the RPO [9]. We restrict this ordering to proofs which have the same *sequent* as conclusion, modulo weakening.

Because we work modulo weakening and contraction, it is important to note that a proof and its weakened and contracted versions are equivalent with respect to the ordering we have just defined, because they have the same cuts and the same labeled tree structure.

Notice also that with this ordering, a cut-free proof is always strictly smaller than a proof with at least one cut at root.

Subproofs of a proof p are defined as the subproofs of p for the sequent calculus, modulo weakening and contraction (subproofs may use less propositions than their parents).

Unfortunately, this definition is not sufficient to define trivial proofs, because if we use a premise through a $\uparrow$-l or $\uparrow$-r rule, there will always be a strict subproof, so that there is no proofs using premises without strict subproofs.

To solve this problem, we can add manually the trivial proofs, i.e. $\mathbb{P}$ is in fact $\mathbb{P} \cup \widehat{\mathbb{A}}$, where formulæ are identified with their trivial proof.

We have to extend the ordering $>$ to trivial proofs: it can be simply done by saying that they cannot be compared with other proofs. ($>$ over $\mathbb{P} \cup \widehat{\mathbb{A}}$ is the same relation as $>$ over the original $\mathbb{P}$.)

For Postulate C to be verified, we have to extend the subproof relation:

$$p \trianglerighteq q \quad \text{if} \begin{array}{l} -\ q \text{ is a subproof modulo weakening of } p \text{ in } \mathcal{SQM}, \text{ or} \\ -\ \text{if } q = \widehat{a} \text{ with } a \in [p]^{Pm}. \end{array}$$

This relation is well-founded because of the wellfoundedness of the subproof relation in sequent calculus, and because trivial proofs cannot have strict subproofs.

With these definitions we can prove the main theorem of this section:

Theorem 3 (Instance of ACS). *The sequent calculus modulo is an instance of ACS, with the definitions of $\mathbb{A}$, $\mathbb{P}$, $[\cdot]^{Pm}$, $[\cdot]_{Cl}$, $>$ and $\triangleright$ given above.*

4 A Generalized Completion Procedure

We want to define a completion procedure through critical proofs. For this, we first need some characterizations of the normal-form proofs and the critical proofs. The limit of this completion procedure will be an equivalent rewrite system which admits Cut.

4.1 Normal-Form Proofs and Critical Proofs in Deduction Modulo

Proposition 3 (Characterization of Normal-Form Proofs). *A proof in deduction modulo is in normal form iff it is either a trivial proof or a cut-free proof with no useless logical rules.*

We give now a characterization of the critical proofs in deduction modulo.

Proposition 4 (Critical Proofs in Deduction Modulo). *Critical proofs in deduction modulo are of the form*

$$\dfrac{\dfrac{\overset{\pi}{\vdots}}{\Gamma, A, P \vdash \Delta}\,\dfrac{}{}\!\!\!\!\quad}{}$$

$$\cfrac{\cfrac{\begin{array}{c}\pi\\ \vdots\\ \Gamma, A, P \vdash \Delta\end{array}}{\Gamma, A \vdash \Delta}\uparrow\text{-l} \qquad \cfrac{\begin{array}{c}\pi'\\ \vdots\\ \Gamma \vdash Q, A, \Delta\end{array}}{\Gamma \vdash A, \Delta}\uparrow\text{-r}}{\Gamma \vdash \Delta}\ \mathsf{Cut}(A)$$

where π and π' are cut-free and do not use unneeded logical rules, and at least one of $A \to P$ or $A \to Q$ is not a term *rewriting.*

Note 1. If we suppose, as in the order condition of [22], that the proposition rewrite system is confluent, and that it is included in a well-founded ordering compatible with the subformula relation, then we can take this ordering instead of the subformula relation to compare cuts in the precedence. Doing this, we can prove that there are no minimal proofs of this form, and consequently *no critical proofs.* Therefore the admissibility of Cut is verified.

The main difference with [22] is that Hermant gives a semantic proof of the cut admissibility, whereas we have here a cut-elimination algorithm, i.e. a terminating syntactical process that transforms a proof into a cut-free one. It remains to be investigated how this process is related with normalization, i.e. β-reduction. (The last case corresponds in fact to an η-expansion.) It is proved in [17] that such an order condition provides normalization in the quantifier-free case.

This result was also independently found by [1], with the same kind of ordering over proofs.

4.2 The Completion Procedure

As we wrote in Sect. 2.3, we want to define a completing deduction mechanism by adding to a presentation A a presentation $\mathsf{C}(A)$ such that $Comp(A) \subseteq \mathsf{C}(A) \subseteq Th\,A$. So we have to find proofs smaller than critical proofs. Here, using Property 1(b) and Lemma 1, we can find for all sequents $\Gamma \vdash \Delta$ a cut-free proof in $Rew(\Gamma \vdash \Delta)$ with conclusion $\Gamma \vdash \Delta$, which will be smaller than any proof containing a cut proving the same sequent, in particular any critical proof. The premises of this proof are in $Rew(\Gamma \vdash \Delta) = [p]_{Cl}$. The best procedure is thus to add only the conclusions of critical proofs. Nevertheless, this is not possible:

Theorem 4 (Undecidability of Critical Proof Search). *Given a propositional rewrite system R and a sequent $\Gamma \vdash \Delta$, it is undecidable to know if $\Gamma \vdash \Delta$ is the conclusion of a critical proof in R.*

We must therefore add a superset of these conclusions. Here we will add the conclusion of the proofs in the form of Proposition 4, except the one that we know for sure that they are not minimal (for instance if $A \in \Gamma \cup \Delta$).

We must consider proofs of the form of Proposition 4. As π and π' are cut-free and do not use unneeded logical rules, they could be found using for instance a tableaux method modulo, like TaMeD [5], which is complete with respect to cut-free proofs, if we knew Γ and Δ, hence the idea to apply the tableaux method to $A, P \vdash$ and $\vdash Q, A$, and to complete Γ and Δ in order to close the remaining tableaux. Because we work modulo weakening, we can restrict ourselves to the minimal Γ and Δ closing the tableaux. We can then sort the obtained $\Gamma \vdash \Delta$ to remove sequents where $A \in \Gamma \cup \Delta$. The resulting rewrite system is obtained by adding all $Rew(\Gamma \vdash \Delta)$ to our rewrite system.

Theorem 5 (Cut Admissibility of the Limit). *For all sequents $\Gamma \vdash \Delta$, for all proposition rewrite systems R_0, $\Gamma \vdash \Delta$ has a proof in R_0 if and only if it has a* cut-free *proof in R_∞.*

4.3 Sequents and Rewrite Systems

For deduction modulo to be an instance of ACS, we have to define some function *Rew* having Properties 1 and 2. We also want to know how to build proofs that use the rewrite system associated with some sequent, and therefore this function has to be effective.

If we consider only propositional logic (i.e. without quantifiers), we can use the following (non-deterministic) algorithm to transform a set of sequents $\Gamma \vdash \Delta$ into a set of rewrite rules:

Step 1. Choose a sequent. Push all negated formulæ on the other side of the sequent. For instance, $A, \neg B \vdash \neg C, D$ becomes $A, C \vdash B, D$. If the new Γ is empty, go to step 2. If the new Δ is empty, go to step 3. If neither is empty, go to either Step 2 or Step 3.

Step 2. Decompose the last proposition iteratively:

$$\begin{array}{ccc}
P_1, \dots, P_n \vdash Q_1, \dots, Q_m & \text{becomes} & P_1, \dots, P_n, \neg Q_1, \dots, \neg Q_{m-1} \vdash Q_m \\
P_1, \dots, P_n \vdash Q_1 \wedge Q_2 & \text{''} & P_1, \dots, P_n \vdash Q_1 \ ; \ P_1, \dots, P_n \vdash Q_2 \\
P_1, \dots, P_n \vdash Q_1 \vee Q_2 & \text{''} & P_1, \dots, P_n, \neg Q_1 \vdash Q_2 \\
P_1, \dots, P_n \vdash Q_1 \Rightarrow Q_2 & \text{''} & P_1, \dots, P_n, Q_1 \vdash Q_2 \\
P_1, \dots, P_n \vdash A & \text{''} & A \rightarrow A \vee \exists x_1, \dots, x_p. \ (P_1 \wedge \dots \wedge P_n)
\end{array}$$

(A atomic, and the x_i are the free variables appearing in $P_1, \dots, P_n$ but not in A)

for $P_1, \dots, P_n \vdash \neg Q$, return to Step 1

Step 3. Decompose the first proposition iteratively, dually from step 2. For instance,

$$\begin{array}{ccc}
P_1 \Rightarrow P_2 \vdash Q_1, \dots, Q_m & \text{becomes} & P_2 \vdash Q_1, \dots, Q_m \ ; \ \neg P_1 \vdash Q_1, \dots, Q_m \\
A \vdash Q_1, \dots, Q_m & \text{''} & A \rightarrow A \wedge \forall x_1, \dots, x_p. \ (Q_1 \vee \dots \vee Q_m)
\end{array}$$

(A atomic, and the x_i are the free variables appearing in $Q_1, \dots, Q_m$ but not in A)

for $\neg P \vdash Q_1, \dots, Q_m$, return to Step 1.

This algorithm clearly terminates, because each time a step 2 or 3 begins, either the rewrite rule is generated, or a formula is decomposed into subformulæ, so that the number of connectors different from $\neg$ strictly diminishes. Of course, we do not pretend that this algorithm is the most optimized for our purpose.

$Rew(\Gamma \vdash \Delta)$ will be the function returning the rewrite system obtained by applying the algorithm to $\{\Gamma \vdash \Delta\}$.

This algorithm can be extended to the case with quantifiers. In the case of a $\forall$ on the left of the sequent, or a $\exists$ on the right, we will keep the formula in the sequent, but will not decompose it further. We will therefore denote by $\underline{P}$ the fact that P was already decomposed. Then we do not consider underlined formulæ in Step 1 to choose between Step 2 or Step 3, and at the beginning of Step 2 (resp. Step 3), one keep a non-underlined formula to the right (resp. left) side.

We also have to add the following decomposition steps:

– (2) $P_1, \dots, P_n \vdash \forall x. \ Q$ becomes $P_1, \dots, P_n \vdash \{y/x\}Q$ where y does not appear in $P_1, \dots, P_n$;

– (2) $P_1, \ldots, P_n \vdash \exists x.\ Q$ becomes $P_1, \ldots, P_n, \neg\underline{\exists x.\ Q} \vdash \{t/x\}Q$ where t can be any ground term;
– (3) $\exists x.\ P \vdash Q_1, \ldots, Q_m$ becomes $\{y/x\}P \vdash Q_1, \ldots, Q_m$ where y does not appear in $Q_1, \ldots, Q_m$.
– (3) $\forall x.\ QP_1, \ldots, P_n \vdash$ becomes $\{t/x\}P \vdash \neg\underline{\forall x.\ Q}, Q_1, \ldots, Q_m$ where t can be any ground term.

Of course, at the end, the underlines are removed.

Proposition 5. *The function Rew has the Properties 1 and 2.*

This algorithm does not allow all rewrite systems to be considered as formulæ. Nevertheless, one can transform all rewrite systems to equivalent rewrite systems that are images of sequents by *Rew*, by splitting the rules: $A \rightarrow P$ becomes $A \rightarrow A \vee P$ and $A \rightarrow A \wedge P$. This is equivalent to the polarized rewrite systems of [13].

This algorithm can be seen as the attempt to build a cut-free proof of the conclusion of a critical proof, adding rewrite rules to close the branches were an atomic formula appears.

4.4 Examples

In the case of Crabbé's example presented in the introduction, the input is the rewrite system $A \rightarrow B \wedge \neg A$ and the completion procedure generates $B \rightarrow B \wedge \bot$ which is equivalent to $B \rightarrow \bot$.

With this new rule, we can show that there is no more critical proofs. Therefore, the proposition rewrite system

$$\begin{cases} A \rightarrow B \wedge \neg A \\ B \rightarrow \bot \end{cases}$$

admits Cut.

The next example deals with quantifiers and is extracted from [22]:

$$R \in R \ \rightarrow\ \forall y.\ y \simeq R \Rightarrow y \in R \Rightarrow C$$

where $y \simeq z \quad \overset{!}{=} \quad \forall x.\ (y \in x \Rightarrow z \in x)$. It is terminating and confluent, but does not admits Cut.

The critical proofs have the form

$$\dfrac{\dfrac{\overset{\vdots}{R \in R, \forall y.\ y \simeq R \Rightarrow y \in R \Rightarrow C \vdash}}{R \in R \vdash}\ \uparrow\text{-l} \qquad \dfrac{\overset{\vdots}{\vdash R \in R, \forall y.\ y \simeq R \Rightarrow y \in R \Rightarrow C}}{\vdash R \in R}\ \uparrow\text{-l}}{\vdash}\ \mathsf{Cut}(R \in R)$$

The left part can be developed as

$$\dfrac{\dfrac{\dfrac{R \in R, C \vdash \quad R \in R \vdash t_1 \in R}{R \in R, t_1 \in R \Rightarrow C \vdash}\ \Rightarrow\text{-l} \qquad \dfrac{\dfrac{R \in R, t_1 \in c_1 \vdash R \in c_1}{R \in R \vdash t_1 \in c_1 \Rightarrow R \in c_1}\ \Rightarrow\text{-r}}{R \in R \vdash t_1 \simeq R}\ \forall\text{-r}}{R \in R, t_1 \simeq R \Rightarrow t_1 \in R \Rightarrow C \vdash}\ \Rightarrow\text{-l}}{R \in R, \forall y.\ y \simeq R \Rightarrow y \in R \Rightarrow C \vdash}\ \forall\text{-l}$$

and the right part as

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{R \in t_0, c_0 \in R \vdash R \in R, C \quad c_0 \in R \vdash c_0 \in t_0, R \in R, C}{c_0 \in t_0 \Rightarrow R \in t_0, c_0 \in R \vdash R \in R, C}\;\Rightarrow\text{-l}}{c_0 \simeq R, c_0 \in R \vdash R \in R, C}\;\forall\text{-l}}{c_0 \simeq R \vdash R \in R, c_0 \in R \Rightarrow C}\;\Rightarrow\text{-r}}{\vdash R \in R, c_0 \simeq R \Rightarrow c_0 \in R \Rightarrow C}\;\Rightarrow\text{-r}}{\vdash R \in R, \forall y.\ y \simeq R \Rightarrow y \in R \Rightarrow C}\;\forall\text{-r} \qquad .$$

To close the proofs, we can for instance have $t_0 = R = t_1$, and C in the right part of the sequent (to close $R \in R, C \vdash$). One can see that other choices will not produce critical proofs. The resulting sequent is therefore $\vdash C$, and the added rule is $C \rightarrow C \vee \top$. This rule does not generate new critical proofs, and consequently, the proposition rewrite system

$$\begin{cases} R \in R \ \rightarrow\ \forall y.\ y \simeq R \Rightarrow y \in R \Rightarrow C \\ C \rightarrow C \vee \top \end{cases}$$

admits Cut.

One can also think of another example, where there remains quantifiers in the conclusion: consider the following rule derived from Crabbé's example $A \rightarrow (\exists x.\ \forall y.\ B \wedge P(x,y)) \wedge \neg A$ where A and B are atomic propositions, and P a predicate of arity 2. For more convenience we will denote by $Q(x)$ the formula $\forall y.\ B \wedge P(x,y)$. We have exactly the same critical proof than in the case of Crabbé, but where B is replaced by $\exists x.\ Q(x)$. The sequent of its conclusion is transformed into a rewrite rule:

$$\begin{array}{lll} \exists x.\ Q(x) \vdash & \text{becomes} & Q(z) \vdash \\ & \text{becomes} & B \wedge P(z,a) \vdash \neg\underline{Q(z)} \\ & \text{becomes} & B \vdash \neg P(z,a), \neg\underline{Q(z)} \\ & \text{becomes} & B \rightarrow B \wedge \forall z.\ \left(\neg P(z,a) \vee \neg Q(z)\right). \end{array}$$

Then, the resulting systems admits Cut, and a cut free proof of $\exists x.\ Q(x) \vdash$ can be:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Q(c), B, P(c,a) \vdash P(c,a)}\;\text{Axiom}}{Q(c), B, \neg P(c,a), P(c,a) \vdash}\;\neg\text{-l} \qquad \dfrac{\dfrac{}{Q(c), B, P(c,a) \vdash Q(c)}\;\text{Axiom}}{Q(c), B, \neg Q(c), P(c,a) \vdash}\;\neg\text{-l}}{Q(c), B, \neg P(c,a) \vee \neg Q(c), P(c,a) \vdash}\;\vee\text{-l}}{Q(c), B, \forall z.\ \left(\neg P(z,a) \vee \neg Q(z)\right), P(c,a) \vdash}\;\forall\text{-l}}{Q(c), B, B \wedge \forall z.\ \left(\neg P(z,a) \vee \neg Q(z)\right), P(c,a) \vdash}\;\wedge\text{-l}}{Q(c), B, P(c,a) \vdash}\;\uparrow\text{-l}}{Q(c), B \wedge P(c,a) \vdash}\;\wedge\text{-l}}{Q(c) \vdash}\;\forall\text{-l}}{\exists x.\ Q(x) \vdash}\;\exists\text{-l}$$

It remains to be investigated, as for Knuth-Bendix completion, for which conditions the completion procedure we have defined is terminating. We conjecture that it is the case if the original proposition rewrite system is confluent.

5 Conclusion and Perspectives

We have shown how, by setting the right abstract canonical system structure on the proof space of a sequent calculus modulo, we can use abstract completion to obtain an equivalent theory modulo which deduction admits Cut. This abstract completion is precise enough to be operational, and it is actually implemented. It also reveals an original and deep logical correspondence between the sequent calculus, proof orderings and rewriting completion. This opens several new challenging questions.

The ordering on proofs we are using is adapted to consider cut admissibility as a normal-form property of an ACS, but produces many critical proofs, in particular when quantifiers are involved. This is because some of the rules produced by the completion procedure subsumes others: for instance $A \rightarrow A \vee \exists x.\ P(x)$ subsumes $A \rightarrow A \vee P(t)$ for a particular $t \in \mathcal{T}(\Sigma, V)$. It is therefore a challenging goal to understand if this ordering could benefit of refinements allowing to target the more relevant critical proofs.

Our saturation procedure only guarantees cut admissibility, not normalization. For instance, with Crabbé's rule, once the system is completed, the initial proof of $B \vdash$ can still be constructed, and it is still not normalizing, i.e. the λ-term that is associated to the proof can be infinitely β-reduced. In other words, we do not have a process that transforms proofs with cuts to cut-free ones. The introduction of simplification rules as in standard completion may allow us to suppress the possibility to build non-normalizing proofs. Moreover, with such simplification rules, the canonical presentation of the system may be obtained.

Let us finally remark that as an interesting consequence of our results, our procedure can be used to determine if a system admits Cut. Indeed, if a proposition rewrite system is a fixpoint of this procedure, then we know that it admits Cut. The converse is not true, essentially because the procedure uses a superset of the critical proofs. It will be interesting to check what results this procedure will give on systems that are proved to admit Cut, like Higher Order Logic [15] or arithmetic [18].

References

1. Aiguier, M., Boin, C., Longuet, D.: On generalized theorems for normalization of proofs. Technical report, LaMI - CNRS and Université d'Evry Val d'Essonne (2005)
2. Baader, F., Nipkow, T.: Term *Re*writing and *all That*. Cambridge University Press (1998)
3. Barendregt, H., Barendsen, E.: Autarkic computations in formal proofs. Journal of Automated Reasoning **28** (2002) 321–336
4. Bonacina, M.P., Dershowitz, N.: Abstract canonical inference. ACM Trans. Comput. Logic **8** (2007)
5. Bonichon, R.: TaMeD: A tableau method for deduction modulo. In: Basin, D.A., Rusinowitch, M. (eds.): IJCAR. Lecture Notes in Computer Science, Vol. 3097. Springer-Verlag (2004) 445–459

6. Burel, G., Kirchner, C.: Completion is an instance of abstract canonical system inference. In: Futatsugi, K., et al. (eds.): Algebra, Meaning and Computation. Lecture Notes in Computer Science, Vol. 4060. Springer-Verlag (2006) 497–520
7. Burel, G., Kirchner, C.: Cut elimination in deduction modulo by abstract completion (full version). Research report (2007) `http://hal.inria.fr/inria-00132964`.
8. Crabbé, M.: Non-normalisation de la théorie de Zermelo. Manuscript (1974)
9. Dershowitz, N.: Orderings for term-rewriting systems. Theoretical Computer Science **17** (1982) 279–301
10. Dershowitz, N.: Canonicity. In: Dahn, I., Vigneron, L. (eds.): FTP. Electronic Notes in Theoretical Computer Science, Vol. 86. Elsevier Science Publishers B. V. (North-Holland) (2003)
11. Dershowitz, N., Kirchner, C.: Abstract saturation-based inference. In: LICS. IEEE Computer Society (2003) 65–74
12. Dershowitz, N., Kirchner, C.: Abstract Canonical Presentations. Theoretical Computer Science **357** (2006) 53–69
13. Dowek, G.: What is a theory? In: Alt, H., Ferreira, A. (eds.): STACS. Lecture Notes in Computer Science, Vol. 2285. Springer-Verlag (2002) 50–64
14. Dowek, G.: Confluence as a cut elimination property. In: Nieuwenhuis, R. (ed.): RTA. Lecture Notes in Computer Science, Vol. 2706. Springer-Verlag (2003) 2–13
15. Dowek, G., Hardin, T., Kirchner, C.: HOL-$\lambda\sigma$ an intentional first-order expression of higher-order logic. Mathematical Structures in Computer Science **11** (2001) 1–25
16. Dowek, G., Hardin, T., Kirchner, C.: Theorem proving modulo. Journal of Automated Reasoning **31** (2003) 33–72
17. Dowek, G., Werner, B.: Proof normalization modulo. The Journal of Symbolic Logic **68** (2003) 1289–1316
18. Dowek, G., Werner, B.: Arithmetic as a theory modulo. In: Giesl, J. (ed.): RTA. Lecture Notes in Computer Science, Vol. 3467. Springer-Verlag (2005) 423–437
19. Gallier, J.H.: Logic for Computer Science: Foundations of Automatic Theorem Proving. Computer Science and Technology Series, Vol. 5. Harper & Row, New York (1986) Revised On-Line Version (2003), `http://www.cis.upenn.edu/~jean/gbooks/logic.html`.
20. Gentzen, G.: Untersuchungen über das logische Schliessen. Mathematische Zeitschrift **39** (1934) 176–210, 405–431 Translated in Szabo, editor., *The Collected Papers of Gerhard Gentzen* as "Investigations into Logical Deduction".
21. Hermant, O.: Méthodes Sémantiques en Déduction Modulo. PhD thesis, École Polytechnique (2005)
22. Hermant, O.: Semantic cut elimination in the intuitionistic sequent calculus. In: Urzyczyn, P. (ed.): TLCA. Lecture Notes in Computer Science, Vol. 3461. Springer-Verlag (2005) 221–233
23. Knuth, D.E., Bendix, P.B.: Simple word problems in universal algebras. In: Leech, J. (ed.): Computational Problems in Abstract Algebra. Pergamon Press, Oxford (1970) 263–297
24. Peterson, G., Stickel, M.E.: Complete sets of reductions for some equational theories. Journal of the ACM **28** (1981) 233–264

Density Elimination and Rational Completeness for First-Order Logics*

Agata Ciabattoni[1] and George Metcalfe[2]

[1] Institute of Discrete Mathematics and Geometry, Technical University Vienna, Wiedner Hauptstrasse 8-10, A-1040 Wien, Austria
agata@logic.at

[2] Department of Mathematics, Vanderbilt University
1326 Stevenson Center, Nashville TN 37240, USA
george.metcalfe@vanderbilt.edu

Abstract. Density elimination by substitutions is introduced as a uniform method for removing applications of the Takeuti-Titani density rule from proofs in first-order hypersequent calculi. For a large class of calculi, density elimination by this method is guaranteed by known sufficient conditions for cut-elimination. Moreover, adding the density rule to any axiomatic extension of a simple first-order logic gives a logic that is rational complete; i.e., complete with respect to linearly and densely ordered algebras: a precursor to showing that it is a fuzzy logic (complete for algebras with a real unit interval lattice reduct). Hence the sufficient conditions for cut-elimination guarantee rational completeness for a large class of first-order substructural logics.

1 Introduction

Elimination of the cut-rule is a fundamental topic in proof theory, corresponding to the removal of lemmas in proofs. However, the addition and subsequent elimination of other rules can also be of considerable interest. In this paper we consider one such rule of importance in Fuzzy Logic: the so-called "density rule" of Takeuti and Titani [14]:

$$\frac{T \vdash (A \to p) \lor (p \to B) \lor C}{T \vdash (A \to B) \lor C}\ (density)$$

where p is a propositional variable not occurring in T, A, B, or C. Ignoring T and C, the negation of the conclusion may (roughly) be interpreted as "$A > B$" and the negation of the premise as "for some p: $A > p$ and $p > B$". That is, between every two elements there exists another element. Note that adding this rule to any axiomatization of Classical Logic leads to inconsistency (e.g. take A to be $\top$ and B to be $\bot$).

The density rule was used by Takeuti and Titani to axiomatize Intuitionistic Fuzzy Logic [14], better known as first-order Gödel logic, one of the main formalizations of Fuzzy Logic [8]. Alternative axiomatizations by Horn and Takano [13] show that $(density)$ is not required; giving a kind of "semantic elimination" of the rule. Baaz and Zach [2] have also provided a *syntactic* elimination of $(density)$ using a proof

* Work Supported by FWF Project P18731.

system for first-order Gödel logic in the framework of hypersequents, a generalization of Gentzen sequents introduced by Avron [1]. This elimination procedure follows the spirit of Gentzen's cut-elimination method, proceeding by induction on the height of a proof of the premise and shifting applications of the rule upwards.

In [10], Metcalfe and Montagna recognized that these two steps - adding and eliminating the density rule - provide a general method for establishing the so-called "standard completeness" of t-norm[1] based (and related) fuzzy logics: that is, completeness of axiomatic systems with respect to algebras whose lattice reduct is the real unit interval $[0, 1]$. In particular, it was shown that *any* axiomatic extension of the elementary propositional fuzzy logic $\mathbf{UL}$ extended with $(density)$ is complete with respect to corresponding linearly and densely ordered algebras. This constitutes the so-called "rational completeness" step of the proof. Standard completeness may then be obtained in many cases (but not in general) by means of the Dedekind-MacNeille completion. It was also shown in [10] that for particular propositional logics possessing a suitable hypersequent calculus, density elimination can be established (Gentzen-style, following [2]) thereby giving standard completeness for the original logic without density. This general approach is in contrast to more logic-specific "semantic" techniques for proving standard completeness [8,9,6,11].

The contribution of this paper is in two parts. First, we introduce a new general method for density elimination. In this approach, similar to normalization for natural deduction systems, applications of the density rule are removed by making suitable substitutions for the new propositional variables. This avoids the combinatorial difficulties of the Gentzen-style proofs in [2,10]. Applying this method to first-order single-conclusion hypersequent calculi with weakening rules, we are able to show that the syntactic conditions defined in [3] for cut-elimination also guarantee density elimination. In the second part of the paper we show that adding the density rule to any axiomatic extension of the first-order version of $\mathbf{UL}$, gives a logic that is complete with respect to linearly and densely ordered algebras. Combining the two parts we obtain rational completeness for a wide class of first-order fuzzy logics with weakening (see [5] for details). These include the first-order versions of (the logic of left-continuous t-norms) Monoidal t-norm based logic $\mathbf{MTL}$ [7] (proved standard complete in [11] by a different method) and its extensions $\mathbf{SMTL}$ [6] and $\mathbf{CnMTL}$ (with $n \geq 2$) [4].

2 (Hyper)Sequent Calculi

We consider formulae built over a *vocabulary* $\mathcal{V}$ consisting of (countably many): (term) variables $x, y, z, \ldots$, for each $n \geq 0$, constants $c, d, \ldots$, n-ary predicate symbols, as well as m-ary connectives $\star_1, \star_2, \ldots$ for each $m \geq 0$ and the quantifiers $\forall$ and $\exists$. Terms are defined in the usual way. A *formula* (in the vocabulary $\mathcal{V}$) is either an atomic formula or a compound formula of the form $\star_i(\boldsymbol{A})$ or $\mathcal{Q}xA$ with $\star_i$ an m-ary connective, which connect formulae $\boldsymbol{A} \equiv A_1, \ldots, A_n$ and $\mathcal{Q} \in \{\forall, \exists\}$. For convenience we call nullary predicate symbols, *propositional variables*, denoted by $p, q, \ldots$.

We indicate with $\Gamma, \Delta, \Pi, \Sigma, \ldots$ (possibly empty) multisets of formulae. To specify inference rules we use *meta-variables* X, Y, Z standing for arbitrary formulae and

[1] t-norms are the main tool in Fuzzy Logic for conjunctively combining vague information.

$\Theta, \Xi, \Phi, \Psi, \Upsilon, \ldots$ standing for (possibly empty) multisets of meta-variables. When $\lambda \geq 0$, Γ^λ (resp. Θ^λ) denotes $\Gamma, \ldots, \Gamma$ (resp. $\Theta, \ldots, \Theta$), λ times. A *(meta)sequent* $\Gamma \Rightarrow \Delta$ ($\Theta \Rightarrow \Xi$), where Γ (Θ) is said to be *antecedent* and Δ (Ξ) *consequent*, is *single-conclusion* if Δ (Ξ) contains at most one formula (meta-variable). A sequent calculus is *single-conclusion* if all its sequents are single-conclusion.

Definition 1. *We call any propositional single-conclusion sequent calculus* $\mathbf{L_L}$ simple *whenever* $\mathbf{L_L}$ *consists of the* identity axiom *of the form* $X \Rightarrow X$*, together with: the (multiplicative version of the)* cut rule (CUT)*,* structural rules $\{(r_i)\}_{i \in \Lambda_0}$ *and for each logical connective* $\star$*,* left logical rules $\{(\star, l)_j\}_{j \in \Lambda_1}$ *and* right logical rules $\{(\star, r)_k\}_{k \in \Lambda_2}$ *(Λ_0, Λ_1, Λ_2 can be empty):*

$$\frac{\Theta \Rightarrow X \quad \Theta', X \Rightarrow \Xi}{\Theta, \Theta' \Rightarrow \Xi}\ (CUT) \qquad \frac{\Upsilon_1 \Rightarrow \Psi_1 \quad \cdots \quad \Upsilon_n \Rightarrow \Psi_n}{\Theta \Rightarrow \Xi}\ (r_i)$$

$$\frac{\Upsilon_1 \Rightarrow \Psi_1 \quad \cdots \quad \Upsilon_n \Rightarrow \Psi_n}{\Theta, \star(\boldsymbol{X}) \Rightarrow \Xi}\ (\star, l)_j \qquad \frac{\Upsilon_1 \Rightarrow \Psi_1 \quad \cdots \quad \Upsilon_n \Rightarrow \Psi_n}{\Theta \Rightarrow \star(\boldsymbol{X})}\ (\star, r)_k$$

In the rules (r_i), $(\star, l)_j$, and $(\star, r)_k$, $n \geq 0$ and the meta-variables in Θ (called left context meta-variables*), those in Ξ (called* right context meta-variables*), and the meta-variables in $\boldsymbol{X} \equiv X_1, \ldots, X_m$, $m \geq 0$ (called* active meta-variables*) are mutually disjoint. In addition, the structural (and logical) rules satisfy the following condition: $(*)$ Any meta-variable in $\Upsilon_1, \ldots, \Upsilon_n$ is a left context meta-variable (or an active meta-variable), and any meta-variable in $\Psi_1, \ldots, \Psi_n$ is a right context meta-variable (or an active meta-variable).*

As usual, an *instance* of a logical or structural rule is obtained by substituting arbitrary formulae for meta-variables. In an instance of a logical or structural rule, the formulae replacing context meta-variables (active meta-variables, respectively) are called *context formulae* (*active formulae*, respectively) and formulae of the form $\star(\boldsymbol{A})$ as well as the formulae replacing X in identity axioms are called *principal formulae*. The two occurrences of the formula instantiating X in (CUT) are called *cut formulae*. *Proofs* (or *derivations*) are defined in the usual way.

Definition 2. *A* w-simple *calculus is a simple sequent calculus containing the weakening rules (w, l) and (w, r) of Fig. 1. A* first-order (w-)simple *sequent calculus is a (w-)simple sequent calculus extended with the rules for quantifiers in Gentzen's calculus* **LJ** *for intuitionistic logic.*

Hypersequent calculi arise by extending Gentzen calculi to refer to many (a multiset of) sequents, instead of just one. Introduced by Avron in [1], they are particularly suitable for dealing with logics with the linearity axiom (lin) $(A \to B) \vee (B \to A)$, prominent examples being t-norm based fuzzy logics [8].

Definition 3. *A* hypersequent *is a multiset $S_1 \mid \ldots \mid S_n$ where each S_i for $i = 1 \ldots n$ is a sequent, called a* component *of the hypersequent. A hypersequent is called* single-conclusion *if all its components are single-conclusion.*

We will assume from now on that we deal only with single-conclusion (hyper)sequent calculi. Like sequent calculi, hypersequent calculi consist of initial hypersequents, logical rules, and structural rules, where we write rules using meta-sequents and a variable

$$\frac{}{X \Rightarrow X}\ (ID) \qquad \frac{}{\Theta, \bot \Rightarrow \Xi}\ (\bot) \qquad \frac{\Theta \Rightarrow X \quad \Phi, X \Rightarrow \Xi}{\Theta, \Phi \Rightarrow \Xi}\ (CUT)$$

$$\frac{\Theta \Rightarrow \Xi}{\Theta, X \Rightarrow \Xi}\ (w, l) \qquad \frac{\Theta \Rightarrow}{\Theta \Rightarrow X}\ (w, r)$$

$$\frac{\Theta, X, Y \Rightarrow \Xi}{\Theta, X \odot Y \Rightarrow \Xi}\ (\odot, l) \qquad \frac{\Theta \Rightarrow X \quad \Theta' \Rightarrow Y}{\Theta, \Theta' \Rightarrow X \odot Y}\ (\odot, r)$$

$$\frac{\Theta, X_i \Rightarrow \Xi}{\Theta, X_1 \odot X_2 \Rightarrow \Xi}\ (\wedge, l)_{i=1,2} \qquad \frac{\Theta \Rightarrow X \qquad \Theta \Rightarrow Y}{\Theta \Rightarrow X \wedge Y}\ (\wedge, r)$$

$$\frac{\Theta, Y \Rightarrow \Xi \quad \Theta' \Rightarrow X}{\Theta, \Theta', X \to Y \Rightarrow \Xi}\ (\to, l) \qquad \frac{\Theta, X \Rightarrow Y}{\Theta \Rightarrow X \to Y}\ (\to, r)$$

$$\frac{\Theta, X \Rightarrow \Xi \quad \Theta, Y \Rightarrow \Xi}{\Theta, X \vee Y \Rightarrow \Xi}\ (\vee, l) \qquad \frac{\Theta \Rightarrow X_i}{\Theta \Rightarrow X_1 \vee X_2}\ (\vee, r)_{i=1,2}$$

Fig. 1. The sequent calculus $\mathbf{FL_{ew}}$

$\mathcal{G}$ (with an instance G) standing for an arbitrary hypersequent. Logical rules for connectives are then the same as those in sequent calculi, except that a "side hypersequent" may occur, denoted by $\mathcal{G}$. Structural rules are divided into two categories. Internal rules deal with formulae within sequents as in sequent calculi. External rules manipulate whole sequents. For example, external weakening and contraction rules (EW) and (EC) add and contract components respectively:

$$\frac{\mathcal{G}}{\mathcal{G} \mid \Theta \Rightarrow \Xi}\ (EW) \qquad \frac{\mathcal{G} \mid \Theta \Rightarrow \Xi \mid \Theta \Rightarrow \Xi}{\mathcal{G} \mid \Theta \Rightarrow \Xi}\ (EC)$$

while the key rule to deal with the axiom (lin) is Avron's communication rule (COM) which permits interaction between sequents:

$$\frac{\mathcal{G} \mid \Theta, \Theta' \Rightarrow \Xi \quad \mathcal{G} \mid \Theta_1, \Theta_1' \Rightarrow \Xi'}{\mathcal{G} \mid \Theta, \Theta_1 \Rightarrow \Xi \mid \Theta', \Theta_1' \Rightarrow \Xi'}\ (COM)$$

For a sequent rule with premises $S_1 \dots S_n$ and conclusion S, its *hypersequent version* is the rule with premises $\mathcal{G} \mid S_1 \dots \mathcal{G} \mid S_n$ and conclusion $\mathcal{G} \mid S$. E.g., the hypersequent version of the quantifier rules $(\exists, l)$ and $(\forall, r)$ are:

$$\frac{\mathcal{G} \mid \Theta \Rightarrow Y(a)}{\mathcal{G} \mid \Theta \Rightarrow (\forall x)Y(x)}\ (\forall, r) \qquad \frac{\mathcal{G} \mid Y(a), \Theta \Rightarrow \Xi}{\mathcal{G} \mid (\exists x)Y(x), \Theta \Rightarrow \Xi}\ (\exists, l)$$

where the eigenvariable condition (on the rules' instances) applies to the whole hypersequent conclusions of the rules.

Definition 4. *Let* $\mathbf{L_L}$ *be any (first-order) w-simple sequent calculus.* $\mathbf{HL_L^C}$*, the hypersequent version of* $\mathbf{L_L}$ *extended with* (COM)*, consists of the hypersequent versions of the axioms and rules of* $\mathbf{L_L}$ *plus* (EC)*,* (EW)*, and* (COM)*.*

Example 1. Let $\forall\mathbf{FL_{ew}}$ be the first-order multiset version of the Full Lambek calculus with exchange and weakening [12] (see Fig. 1), roughly speaking a calculus for first-order intuitionistic logic without contraction with an internalized exchange rule. We illustrate the use of (COM) with a proof of (lin) in $\mathbf{H\forall FL^C_{ew}}$, a hypersequent calculus for the first-order version of the logic of left-continuous t-norms $\mathbf{MTL}$ [7,11]:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \Rightarrow A}\,(ID) \quad \dfrac{}{B \Rightarrow B}\,(ID)}{A \Rightarrow B \mid B \Rightarrow A}\,(COM)}{\Rightarrow A \to B \mid \ \Rightarrow B \to A}\,(\to, r) \times 2}{\Rightarrow (A \to B) \vee (B \to A) \mid \ \Rightarrow (A \to B) \vee (B \to A)}\,(\vee, r) \times 2}{\Rightarrow (A \to B) \vee (B \to A)}\,(EC)$$

3 Criteria for Cut-Elimination

Syntactic criteria for the preservation of cut-elimination when a sequent calculus $\mathbf{L_L}$ is “lifted” to the hypersequent calculus $\mathbf{HL^C_L}$ were introduced in [3]. The criteria intuitively say that (a) any application of (CUT) can be shifted upwards over each premise's rule (i.e. rules are *substitutive*) and (b) each (CUT) in which the cut formula is principal in both premises can be replaced by applications of (CUT) over smaller cut-formulae (i.e. logical rules are *reductive*).

Definition 5. *Let $\mathbf{L_L}$ be a simple sequent calculus. We call its logical rules $\{(\star, r)_k\}_{k \in \Lambda}$ and $\{(\star, l)_l\}_{l \in \Lambda'}$ for introducing a logical connective $\star$* reductive *in $\mathbf{L_L}$ if either Λ or Λ' is empty, or for any $k \in \Lambda$, $l \in \Lambda'$, and instances :*

$$\dfrac{\Gamma_1 \Rightarrow \Delta_1 \quad \cdots \quad \Gamma_n \Rightarrow \Delta_n}{\Sigma \Rightarrow \star(A_1, \ldots, A_p)}\,(\star, r)_k \qquad \dfrac{\Gamma'_1 \Rightarrow \Delta'_1 \quad \cdots \quad \Gamma'_m \Rightarrow \Delta'_m}{\Sigma', \star(A_1, \ldots, A_p) \Rightarrow \Pi}\,(\star, l)_l$$

the sequent $\Sigma, \Sigma' \Rightarrow \Pi$ is derivable from $\{\Gamma_i \Rightarrow \Delta_i\}_{1 \le i \le n}$ and $\{\Gamma'_i \Rightarrow \Delta'_i\}_{1 \le i \le m}$ using only (CUT) with cut formulae in $\{A_1, \ldots, A_p\}$ and the structural rules of $\mathbf{L_L}$.

Example 2. The logical rules for the connectives $\odot$ (multiplicative ”and”), $\wedge$ (additive ”and”), $\vee$, and $\to$ in the calculus $\mathbf{FL}_{ew}$ (see Fig. 1) are reductive.

Let S be a sequent and A a formula; we define:

$$[S \hookleftarrow^r_A (\Sigma, A \Rightarrow \Pi)] = \{\Gamma, \Sigma \Rightarrow \Pi \mid S \equiv \Gamma \Rightarrow A\}$$
$$[S \hookleftarrow^l_A (\Sigma \Rightarrow A)] = \{\Gamma, \Sigma^\lambda \Rightarrow \Delta \mid S \equiv \Gamma, A^\lambda \Rightarrow \Delta\}$$

Definition 6. *Let $\mathbf{L_L}$ be a (first-order) simple sequent calculus. A rule (r) is said to be* substitutive *in $\mathbf{L_L}$ if for each instance of (r) with premises $S_1, \ldots, S_n$ and conclusion S_0 the following condition holds:*

(*) for any $c \in \{r, l\}$, context formula A (right or left context formula, depending on c) and single-conclusion sequent S' (which does not contain any eigenvariable of (r)), every $U \in [S_0 \hookleftarrow^c_A S']$ has a derivation from $[S_1 \hookleftarrow^c_A S'] \ldots [S_n \hookleftarrow^c_A S']$ using only the structural rules of $\mathbf{L_L}$ and (r).

Example 3. All the rules of the calculus $\forall\mathbf{FL_{ew}}$ (see Fig. 1 for $\mathbf{FL_{ew}}$) are substitutive in $\forall\mathbf{FL_{ew}}$, as are, e.g. the following forms of weak contraction ($n \geq 2$):

$$\frac{\Theta, X, X \Rightarrow}{\Theta, X \Rightarrow}\ (wc) \qquad\qquad \frac{\Theta, \Psi_1^n \Rightarrow \Xi \quad \dots \quad \Theta, \Psi_{n-1}^n \Rightarrow \Xi}{\Theta, \Psi_1, \dots, \Psi_{n-1} \Rightarrow \Xi}\ (nc)$$

Theorem 1 ([3]). *Let* $\mathbf{L_L}$ *be a (first-order) simple calculus in which (a) logical rules are reductive and (b) rules are substitutive, then* $\mathbf{HL_L^C}$ *admits cut-elimination.*

4 Density Elimination by Substitutions

Instances of the density rule for hypersequent calculi are of the form:

$$\frac{G \mid \Sigma, p \Rightarrow \Delta \mid \Gamma \Rightarrow p}{G \mid \Sigma, \Gamma \Rightarrow \Delta}\ (D)$$

where p is a propositional variable not occurring in Σ, Γ, Δ, or G. We show here that the conditions given above for cut elimination also guarantee density elimination for the hypersequent version of any w-simple sequent calculus extended with (COM) and (D). We introduce for this purpose *density elimination by substitutions*: a density elimination method that, similarly to normalization for natural deduction systems, removes applications of (D) by making suitable substitutions in the proof for the introduced variables. Proceeding "by substitutions" instead of shifting applications of (D) upwards in the proof (as e.g. in [2,10]) avoids the need for more complicated combinatorial ("Gentzen's mix"-style) rules as induction hypotheses.

First some notation. Let **HL** be any (hyper)sequent calculus. $d, S_1, \dots, S_n \vdash_{\mathbf{HL}} S$ stands for a derivation d in **HL** of the (hyper)sequent S from the assumptions $S_1, \dots, S_n$. Let H be a hypersequent. $H[{}^{\Sigma}/_{p^l}, {}^{\Gamma \Rightarrow \Delta}/_{p^r}]$ is the hypersequent obtained by replacing in H all the left occurrences of p with Σ and all the components $\Pi \Rightarrow p$ with $\Pi, \Gamma \Rightarrow \Delta$. $d(s)$ and $H(s)$ denote the results of substituting the term s for all free occurrences of x in the derivation $d(x)$ and in the (hyper)sequent $H(x)$, respectively. The *length* $|d|$ of a derivation d in **HL** is (the maximal number of inference rules occurring on any branch of d) + 1.

We require the following crucial lemma (proved by easy inductions on the lengths of derivations) asserting the "substitutivity" of calculi with substitutive rules:

Lemma 1. *Let* $\mathbf{L_L}$ *be a (first-order) simple sequent calculus with substitutive rules:*

(1) If $d_1(x) \vdash_{\mathbf{HL_L^C}} H(x)$*, then* $d_1(y) \vdash_{\mathbf{HL_L^C}} H(y)$ *where* y *does not occur in* $d_1(x)$*.*
(2) If $\vdash_{\mathbf{HL_L^C}} H$*, then* $\vdash_{\mathbf{HL_L^C}} H[{}^{A}/_{p^l}, {}^{\Rightarrow A}/_{p^r}]$ *for any formula* A *and propositional variable* p*.*

Our density elimination method proceeds by removing applications of (D) which are topmost in the proof. Let, e.g. d be:

$$\frac{\begin{array}{c}\vdots\, d' \\ G \mid \Gamma \Rightarrow p \mid \Sigma, p \Rightarrow \Delta\end{array}}{G \mid \Gamma, \Sigma \Rightarrow \Delta}\ (D)$$

a subderivation ending in such an application of (D). The idea is to replace the occurrences of p in d in an "asymmetric" way, according to whether p occurs in the antecedent or consequent of a sequent (component of the hypersequent). Roughly speaking, in d each sequent $\Pi, \underline{p} \Rightarrow \Pi'$ is replaced by $\Pi, \underline{\Gamma} \Rightarrow \Pi'$ and each $\Pi \Rightarrow \underline{p}$ by $\Pi, \underline{\Sigma} \Rightarrow \underline{\Delta}$. (Note that condition $(*)$ in Definition 1 prevents p's from jumping from one side of a sequent in a rule's premise, to the other in the rule's conclusion.) The resulting tree is then transformed into a density-free derivation by replacing:

(a) the application of (D) above by (EC).
(b) each application of a substitutive rule by suitable inferences.
(c) each subproof ending in an application of (COM) and containing one occurrence of the axiom $p \Rightarrow p$, as e.g. in

$$\dfrac{\overset{\vdots\, d_1}{G \mid \Gamma', \Pi \Rightarrow \Pi'} \qquad G \mid \Gamma, \Sigma \Rightarrow \Delta}{G \mid \Gamma, \Pi \Rightarrow \Pi' \mid \Gamma', \Sigma \Rightarrow \Delta}\ (\mathrm{COM})$$

by a suitable derivation of the form

$$\dfrac{\overset{\vdots\, d_1}{G \mid \Gamma', \Pi \Rightarrow \Pi'}}{\begin{array}{c} \\ \vdots \\ G \mid \Gamma, \Pi \Rightarrow \Pi' \mid \Gamma', \Sigma \Rightarrow \Delta \end{array}}$$

Theorem 2 (Density Elimination). *Let $\mathbf{L_L}$ be a (first-order) w-simple sequent calculus whose rules are reductive and substitutive. If $\mathbf{L_L}$ includes the rules $(\odot, l)$ and $(\odot, r)$ in Fig. 1 then $\mathbf{HL_L^C}$ plus (D) admits density elimination.*

Proof. W.l.o.g. consider the above (sub)derivation d in $\mathbf{HL_L^C}$ plus (D) ending in a topmost application of (D). By Theorem 1 we can assume that d is cut-free. We first show that for each hypersequent H in d' in which no component has the form $\Pi, p^j \Rightarrow p$, with $j \geq 1$, one can find d'_H such that

$$d'_H \vdash_{\mathbf{HL_L^C}} G \mid H[{}^{\Gamma}/_{p^l}, {}^{\Sigma \Rightarrow \Delta}/_{p^r}]$$

The proof proceeds by induction on the length of the cut-free derivation d_H of H in $\mathbf{HL_L^C}$. We distinguish cases according to the last rule (r) applied in d_H.

– If $|d_H| = 0$, i.e. H is $G' \mid B \Rightarrow B$, then the claim holds by applying (EW).
– If (r) is (EC) or (EW), then the claim follows by the i.h. and applying (r).
– Let (r) be a rule other than (EC), (EW), or (COM), w.l.o.g. of the form:

$$\dfrac{G' \mid S_1 \ \ldots \ G' \mid S_m}{G' \mid S}$$

Since $G' \mid S$ does not contain any component of the form $\Pi, p^j \Rightarrow p$, with $j \geq 1$, by condition $(*)$ in Def. 1 and the absence of cuts, no $G' \mid S_i$ (with $i \in \{1, \ldots, m\}$) contains any $\Pi, p^j \Rightarrow p$, with $j \geq 1$. Hence by the i.h.:

$$\vdash_{\mathbf{HL_L^C}} G \mid (G' \mid S_1)[{}^{\Gamma}/_{p^l}, {}^{\Sigma \Rightarrow \Delta}/_{p^r}] \quad \ldots \quad \vdash_{\mathbf{HL_L^C}} G \mid (G' \mid S_m)[{}^{\Gamma}/_{p^l}, {}^{\Sigma \Rightarrow \Delta}/_{p^r}]$$

Since the rules of $\mathbf{L_L}$ are substitutive, using Lemma 1.(1) to take care of renaming variables, there exists a derivation for:

$$S_1[{}^{\Gamma}/_{p^l}, {}^{\Sigma\Rightarrow\Delta}/_{p^r}], \ldots, S_m[{}^{\Gamma}/_{p^l}, {}^{\Sigma\Rightarrow\Delta}/_{p^r}] \vdash_{\mathbf{L_L}} S[{}^{\Gamma}/_{p^l}, {}^{\Sigma\Rightarrow\Delta}/_{p^r}]$$

that uses only the structural rules of $\mathbf{L_L}$ and (possibly) (r). The claim then follows by (EW), lifting the above derivation from $\mathbf{L_L}$ to $\mathbf{HL_L}$.

- If (r) is (COM), two cases can occur: (a) none of its premises contains any $\Pi, p^j \Rightarrow p$, with $j \geq 1$ or (b) one of the premises does. For (a), the claim holds by applying the i.h. followed by an application of (COM). As an example, consider the application of (COM):

$$\dfrac{\overset{\vdots\, d_1}{G' \mid \Gamma', \Pi \Rightarrow p} \qquad \overset{\vdots\, d_2}{G' \mid \Sigma', p^k, \Pi' \Rightarrow B}}{G' \mid \Gamma', \Sigma', p^k \Rightarrow B \mid \Pi, \Pi' \Rightarrow p}\ (\mathrm{COM})$$

Let $G^* = G'[{}^{\Gamma}/_{p^l}, {}^{\Sigma\Rightarrow\Delta}/_{p^r}]$. By the i.h.:

$$\vdash_{\mathbf{HL_L^C}} G \mid G^* \mid \Gamma', \Pi, \Sigma \Rightarrow \Delta \quad \text{and} \vdash_{\mathbf{HL_L^C}} G \mid G^* \mid \Sigma', \Gamma^k, \Pi' \Rightarrow B$$

Hence by (COM), $\vdash_{\mathbf{HL_L^C}} G \mid G^* \mid \Gamma', \Sigma', \Gamma^k \Rightarrow B \mid \Pi, \Pi', \Sigma \Rightarrow \Delta$.
For (b), we have an application of (COM):

$$\dfrac{G' \mid \Gamma', \Pi, p^l \Rightarrow p \qquad G' \mid \Sigma', p^{(k-l)}, \Pi' \Rightarrow B}{G' \mid \Gamma', \Sigma', p^k \Rightarrow B \mid \Pi, \Pi' \Rightarrow p}\ (\mathrm{COM})$$

Letting again $G^* = G'[{}^{\Gamma}/_{p^l}, {}^{\Sigma\Rightarrow\Delta}/_{p^r}]$, by the i.h.:

$$d_1 \vdash_{\mathbf{HL_L^C}} G \mid G^* \mid \Sigma', \Gamma^{(k-l)}, \Pi' \Rightarrow B.$$

Our aim is to show $\vdash_{\mathbf{HL_L^C}} G \mid G^* \mid \Gamma', \Sigma', \Gamma^k \Rightarrow B \mid \Pi, \Pi', \Sigma \Rightarrow \Delta$. If $\Pi' = \emptyset$, then the result follows by (w, l) so assume that $\Pi' = A_1, \ldots, A_m$. Let d_1' be the derivation obtained by m applications of $(\odot, l)$ to the end-hypersequent of d_1, i.e.

$$\dfrac{\overset{\vdots\, d_1}{G \mid G^* \mid \Sigma', \Gamma^{(k-l)}, \Pi' \Rightarrow B}}{G \mid G^* \mid \Sigma', \Gamma^{(k-l)}, A_1 \odot \ldots \odot A_m \Rightarrow B}\ (\odot,\mathrm{l})\times\mathrm{m}$$

Denote $(A_1 \odot \ldots \odot A_m)$ by $\odot\Pi'$. Consider the proof d' above ending in the premise $G \mid \Gamma \Rightarrow p \mid \Sigma, p \Rightarrow \Delta$ of the (D) rule. By Lemma 1.(2), $d_2 \vdash_{\mathbf{HL_L^C}} (G \mid \Gamma \Rightarrow p \mid \Sigma, p \Rightarrow \Delta)[{}^{\odot\Pi'}/_{p^l}, {}^{\odot\Pi'}/_{p^r}]$ for some derivation d_2, that is:

$$d_2 \vdash_{\mathbf{HL_L^C}} G \mid \Gamma \Rightarrow \odot\Pi' \mid \Sigma, \odot\Pi' \Rightarrow \Delta.$$

The desired derivation is then obtained by applying (CUT) to the easily derived sequent $\Pi' \Rightarrow \odot\Pi'$ and the end sequent of the following derivation:

$$\dfrac{\dfrac{\overset{\vdots\, d_2}{G \mid \Gamma \Rightarrow \odot\Pi' \mid \Sigma, \odot\Pi' \Rightarrow \Delta}}{G \mid G^* \mid \Gamma \Rightarrow \odot\Pi' \mid \Sigma, \odot\Pi' \Rightarrow \Delta}\ (\mathrm{EW}) \qquad \overset{\vdots\, d_1'}{G \mid G^* \mid \Sigma', \Gamma^{(k-l)}, \odot\Pi' \Rightarrow B}}{G \mid G^* \mid \Sigma', \Gamma^{(k-l)}, \Gamma \Rightarrow B \mid \Sigma, \odot\Pi' \Rightarrow \Delta}\ (\mathrm{CUT})$$

with subsequent applications of (w, l).

Finally, let H be the premise $G \mid \Gamma \Rightarrow p \mid \Sigma, p \Rightarrow \Delta$ of the (D) rule in d. We have shown that $\vdash_{\mathbf{HL_L^C}} G \mid G \mid \Gamma, \Sigma \Rightarrow \Delta \mid \Gamma, \Sigma \Rightarrow \Delta$. (Note that $G[^{\Gamma}/_{p^l}, ^{\Sigma \Rightarrow \Delta}/_{p^r}] = G$). The desired proof of $G \mid \Gamma, \Sigma \Rightarrow \Delta$ follows by multiple applications of (EC). □

Theorem 3. *Let* $\mathbf{L_L}$ *be a (first-order) w-simple sequent calculus whose rules are reductive and substitutive.* $\mathbf{HL_L^C}$ *plus* (D) *admits cut elimination and density elimination.*

Proof. Let $\star$ be a new 2-ary connective. Let $\mathbf{L'_L}$ be $\mathbf{L_L}$ extended with the following (reductive and substitutive) rules:

$$\frac{\Theta, X, Y \Rightarrow \Xi}{\Theta, X \star Y \Rightarrow \Xi}\ (\star, l) \qquad\qquad \frac{\Theta \Rightarrow Y \quad \Theta' \Rightarrow X}{\Theta, \Theta' \Rightarrow X \star Y}\ (\star, r)$$

$\mathbf{HL'^C_L}$ admits cut elimination by Theorem 1. Also, as $\star$ has the rules of $\odot$, $\mathbf{HL'^C_L}$ plus (D) admits density elimination by Theorem 2. Then since $\mathbf{HL'^C_L}$ has the subformula property, $\mathbf{HL_L^C}$ plus (D) admits cut elimination and density elimination. □

5 Axiomatizations

We use density elimination to establish so-called "rational completeness" for a wide class of first-order substructural logics described below. Our vocabulary is assumed to include the binary connectives $\wedge$, $\vee$, $\odot$, $\to$; the constants f, t, $\top$, $\bot$; and the defined connective $A \leftrightarrow B =_{def} (A \to B) \wedge (B \to A)$. A logic $\mathbf{L}$ is treated as a Hilbert system, where $T \vdash_{\mathbf{L}} A$ if there exists a derivation (in the usual sense) of a formula A from a set of formulae T in $\mathbf{L}$. We begin by recalling the (propositional) Uninorm logic $\mathbf{UL}$ of [10] (an axiomatization for $\mathbf{FL}_e$ plus (L5)), given by the axiom schema:

(L1) $X \to X$
(L2) $(X \to Y) \to ((Y \to Z) \to (X \to Z))$
(L3) $(X \to (Y \to Z)) \to (Y \to (X \to Z))$
(L4) $((X \odot Y) \to Z) \leftrightarrow (X \to (Y \to Z))$
(L5) $((X \to Y) \wedge t) \vee ((Y \to X) \wedge t)$
(L6) $((X \to Z) \wedge (Y \to Z)) \to ((X \vee Y) \to Z)$
(L7) $((X \to Y) \wedge (X \to Z)) \to (X \to (Y \wedge Z))$
(L8) $X \to (X \vee Y)$
(L9) $Y \to (X \vee Y)$
(L10) $(X \wedge Y) \to Y$
(L11) $(X \wedge Y) \to X$
(L12) $X \leftrightarrow (t \to X)$
(L13) $\bot \to X$
(L14) $X \to \top$

together with the inference rules:

$$\frac{X \quad X \to Y}{Y}\ (mp) \qquad\qquad \frac{X \quad Y}{X \wedge Y}\ (adj)$$

Definition 7. *We call any logic* $\mathbf{L}$ *resulting from* $\mathbf{UL}$ *by adding "extra" propositional axioms (possibly with "extra" connectives), and satisfying for all formulae A, B, C:*

$$A \leftrightarrow B \vdash_{\mathbf{L}} C(A) \leftrightarrow C(B)$$

a core $\mathbf{UL}$-expansion.[2]

[2] A related notion of a "core fuzzy logic", restricted to logics with weakening, is used in [5].

Definition 8. *For any core* $\mathbf{UL}$*-expansion* $\mathbf{L}$*,* $\forall\mathbf{L}$ *consists of* $\mathbf{L}$ *plus:*

$$\begin{array}{l}(\forall 1)\ \forall x Y \to Y(x/t) \quad (t \textit{ substitutable for } x \textit{ in } Y)\\ (\forall 2)\ \forall x(Y \to Z) \to (Y \to \forall x Z) \quad (x \textit{ not free in } Y)\\ (\forall 3)\ \forall x(Y \vee Z) \to (Y \vee \forall x Z) \quad (x \textit{ not free in } Y)\\ (\exists 1)\ Y(x/t) \to \exists x Y \quad (t \textit{ substitutable for } x \textit{ in } Y)\\ (\exists 2)\ \forall x(Y \to Z) \to (\exists x Y \to Z) \quad (x \textit{ not free in } Z)\end{array}$$

$$\frac{Y}{\forall x Y}\ (gen)$$

Semantics for these logics are based on *pointed bounded commutative residuated lattices*: algebras $\langle L, \wedge, \vee, \odot, \to, t, f, \bot, \top\rangle$ with binary operations $\wedge$, $\vee$, $\odot$, $\to$, and constants t, f, $\bot$, $\top$ such that $\langle L, \wedge, \vee, \bot, \top\rangle$ is a bounded lattice; $\langle L, \odot, t\rangle$ is a commutative monoid; and $z \leq x \to y$ iff $x \odot z \leq y$ for all x,y,$z \in L$.

A *UL-algebra* is a pointed bounded commutative residuated lattice satisfying:

$$t \leq ((x \to y) \wedge t) \vee ((y \to x) \wedge t)$$

If $\mathcal{A}$ is an algebra such that each connective of a language $\mathcal{L}$ occurs as an operation of $\mathcal{A}$, then ($\mathcal{L}$-)*valuations* for $\mathcal{A}$ with (non-empty) domain $\mathcal{D}$ are defined as partial functions v from sentences of $\mathcal{L}$ with parameters in $\mathcal{D}$ into $\mathcal{A}$, total on atomic sentences, where:

1. If $v(A_i)$ is defined for $i = 1\ldots m$, then $v(\star(A_1, \ldots, A_m)) = \star(v(A_1), \ldots, v(A_m))$ for each m-ary connective $\star$ of $\mathcal{L}$.
2. If for all $c \in \mathcal{D}$, $v(A(x/c))$ is defined and $\inf(\{v(A(x/c)) : c \in \mathcal{D}\})$ exists, then $v(\forall x A) = \inf(\{v(A(x/c)) : c \in \mathcal{D}\})$.
3. If for all $c \in \mathcal{D}$, $v(A(x/c))$ is defined and $\sup(\{v(A(x/c)) : c \in \mathcal{D}\})$ exists, then $v(\exists x A) = \sup(\{v(A(x/c)) : c \in \mathcal{D}\})$.

A valuation v for $\mathcal{A}$ with domain $\mathcal{D}$ is *safe* if $v(A)$ is defined for every sentence with parameters in $\mathcal{D}$. A formula A is *valid* in $\mathcal{A}$ (assuming that $\mathcal{A}$ contains the constant t) iff for every non-empty set $\mathcal{D}$ and for every safe valuation v with domain $\mathcal{D}$, $v(A) \geq t$.

Definition 9. *For a core* $\mathbf{UL}$*-expansion* $\mathbf{L}$ *with extra connectives* I*, an* L-algebra *is an algebra* $\mathcal{A} = \langle L, \wedge, \vee, \odot, \to, t, f, \bot, \top, (c)_{c\in I}\rangle$ *such that* $\langle L, \wedge, \vee, \odot, \to, t, f, \bot, \top\rangle$ *is a UL-algebra and each additional axiom of* $\mathbf{L}$ *is valid in* $\mathcal{A}$*. An L-algebra is called an* L-chain *if it is linearly ordered, and a* dense L-chain *if it is linearly and densely ordered.*

Recall that a first-order theory T is a set of sentences:

1. T is *linear* if for each pair A, B of sentences, either $A \to B \in T$ or $B \to A \in T$.
2. T is $\forall\mathbf{L}$-*dense* if for each pair A, B of sentences, whenever $T \not\vdash_{\forall\mathbf{L}} A \to B$, then for some sentence C, $T \not\vdash_{\forall\mathbf{L}} A \to C$ and $T \not\vdash_{\forall\mathbf{L}} C \to B$.
3. T is $\forall\mathbf{L}$-*Henkin* if for each sentence of the form $\forall x A(x)$ where $T \not\vdash_{\forall\mathbf{L}} \forall x A(x)$, there is a constant c in the language of T such that $T \not\vdash_{\forall\mathbf{L}} A(c)$.

The Lindenbaum algebra of a theory T is defined as follows, justified by the fact that the connective $\leftrightarrow$ has the property of a "congruence" for any core $\mathbf{UL}$-expansion:

Definition 10. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion* $\mathbf{L}$ *and* T *a theory. The* Lindenbaum algebra $\mathcal{A}_T^{\forall\mathbf{L}}$ *of a theory* T *has universe* $L_T = \{[A]_T : A \text{ a sentence}\}$ *where* $[A]_T = \{B : T \vdash_{\forall\mathbf{L}} A \leftrightarrow B\}$*, and operations* $\star([A_1]_T, \ldots, [A_n]_T) = [\star(A_1, \ldots, A_n)]$ *for each* n*-ary connective* $\star$ *of* $\mathbf{L}$ *.*

The proof of the following lemma then proceeds exactly as for Lemma 5.2.6 in [8]:

Lemma 2. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion* $\mathbf{L}$ *and* T *a* $\forall\mathbf{L}$*-Henkin theory. Then for any formula* $A(x)$ *with one free variable* x*:* $[\forall x A]_T = \inf_c [A(c)]_T$ *and* $[\exists x A]_T = \sup_c [A(c)]_T$ *where c runs over all constants of* T*.*

6 Adding Density

Here we show that adding the density rule to any first-order core $\mathbf{UL}$-expansion $\mathbf{L}$ gives a logic that is complete with respect to dense L-chains. To add the density rule to our axiomatizations, we explicitly define derivations from a set of formulae T to take account of the fact that applications of $(density)$ are sensitive to T and might therefore require new propositional variables.

Definition 11. *For* $\mathbf{L}$ *a core* $\mathbf{UL}$*-expansion, let* $\forall\mathbf{L}^{\mathbf{D}}$ *be* $\forall\mathbf{L}$ *extended with the rule* $(density)$ *(see Section 1). A* $\forall\mathbf{L}^{\mathbf{D}}$-derivation of a formula C from a set of formulae T*, written* $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C$*, consists of a sequence of formulae in the vocabulary of* $\forall\mathbf{L}^{\mathbf{D}}$ *extended with countably many new constants and propositional variables, ending with* C *and such that each member* A *of the sequence satisfies one of the following:*

(1) $A \in T$ *or* A *is an axiom of* $\forall\mathbf{L}$ *for the extended language.*
(2) A *is obtained from previous members of the sequence by* (mp)*,* (adj)*, or* (gen) *i.e. either* B *and* $B \to A$*,* B *and* C *(where* A *is* $B \wedge C$*), or* $B(a)$ *(where* A *is* $\forall x B(x)$ *and the eigenvariable condition for* T *is satisfied) occur earlier in the sequence.*
(3) A *is obtained from a previous member of the sequence by* $(density)$*, i.e.* A *is* $(B \to C) \vee D$ *and* $(B \to p) \vee (p \to C) \vee D$ *occurs earlier in the sequence, where* p *is a new propositional variable not occurring in* T*,* B*,* C*, or* D*.*

Soundness for $\forall\mathbf{L}^{\mathbf{D}}$ with respect to dense L-chains is established as follows.

Lemma 3. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion. If* $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} A$*, then for every dense L-chain and safe valuation* v *with non-empty domain* $\mathcal{D}$*, if* $v(B) \geq t$ *for all* $B \in T$*, then* $v(A) \geq t$*. In particular, if* $\vdash_{\forall\mathbf{L}^{\mathbf{D}}} A$*, then* A *is valid in all dense L-chains.*

Proof. We proceed by induction on the height of a $\forall\mathbf{L}^{\mathbf{D}}$-derivation for $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} A$, checking the validity of axioms and soundness of rules in the usual manner, the only novel case being $(density)$. Suppose contrapositively that there is a dense L-chain and non-empty domain $\mathcal{D}$ with a safe valuation v such that $v(D) \geq t$ for all $D \in T$, and $v((A \to B) \vee C) < t$. It follows that $v(A \to B) < t$ and $v(C) < t$, and hence also that $v(A) > v(B)$. Since the algebra is dense, there exists an element w such that $v(A) > w > v(B)$. Recall that the propositional variable p in $(density)$ does not occur in T, A, B, or C. Hence we can extend the valuation v with $v(p) = w$. It follows that $v((A \to p) \vee (p \to B) \vee C) < t$. □

We require the following properties (established as in the propositional case in [10]):

Lemma 4. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion:*

(a) If $T, A \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C$ *and* $T, B \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C$*, then* $T, A \vee B \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C$.
(b) If $T, A \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C$ *and* $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} A \vee C$*, then* $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C$.

Definition 12. *A* confusion *of a set of formulae* T *is defined inductively as follows:*

1. t, $\top$*, and any element of* T *are confusions of* T.
2. *If* C_1 *and* C_2 *are confusions of* T*, then so are* $C_1 \odot C_2$ *and* $C_1 \wedge C_2$.

Lemma 5. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion:*

(a) If $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} A$*, then* $T_0 \vdash_{\forall\mathbf{L}^{\mathbf{D}}} A$ *for some finite subset* T_0 *of* T.
(b) If T *is finite, then* $T \vdash_{\forall\mathbf{L}^{\mathbf{D}}} A$ *iff* $\vdash_{\forall\mathbf{L}^{\mathbf{D}}} C \to A$ *for some confusion* C *of* T.
(c) If A *is a confusion of* T*, then* $T \vdash_{\forall\mathbf{L}} A$.

We are now in a position to establish our key lemmas.

Lemma 6. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion and* T *a countable theory. If* $T \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} A$*, then there exists a countable linear* $\forall\mathbf{L}$*-dense* $\forall\mathbf{L}$*-Henkin theory* $\hat{T}$ *such that* $T \subseteq \hat{T}$ *and* $\hat{T} \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} A$.

Proof. We construct $\hat{T}$ in countably many steps. First we extend the vocabulary with countably many new propositional variables and constants not occurring in T or A. In the construction of $\hat{T}$ we have to: (1) deal with linearity and $\forall\mathbf{L}$-density for each pair of sentences (in the extended vocabulary), and (2) deal with the $\forall\mathbf{L}$-Henkin property for each sentence of the form $\forall x A$ (in the extended vocabulary). Since these are countably many tasks we can interleave them.

We begin by defining $T_0 = T$ and $C_0 = A$, noting that $T_0 \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} C_0$. Now for the induction step, assume that T_n and C_n have been constructed such that $T_n \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n$. We construct T_{n+1} and C_{n+1} such that $T_n \subseteq T_{n+1}$; $T_{n+1} \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} C_{n+1}$; $T_{n+1} \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n \to C_{n+1}$; and T_{n+1} fulfills the n-th task. We have two cases:

(1) The n-th task is dealing with linearity and $\forall\mathbf{L}$-density for the sentences A, B.

– If $T_n \cup \{A \to B, B \to A\} \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n$, then it is sufficient to define:

$$T_{n+1} = T_n \cup \{A \to B, B \to A\} \text{ and } C_{n+1} = C_n$$

– Suppose that the previous case does not apply. Let q be a propositional variable not occurring in T_n, A, B, or C_n. We claim that one of the following conditions holds:
(a) $T_n \cup \{A \to B\} \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n \vee (B \to q) \vee (q \to A)$.
(b) $T_n \cup \{B \to A\} \nvdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n \vee (A \to q) \vee (q \to B)$.
Suppose that (a) does not hold. Then by the density rule:

$$T_n \cup \{A \to B\} \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n \vee (B \to A)$$

and since $T_n \cup \{A \to B,\ B \to A\} \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n$, by Lemma 4 (b):

$$T_n \cup \{A \to B\} \vdash_{\forall\mathbf{L}^{\mathbf{D}}} C_n$$

If (b) also does not hold, $T_n \cup \{B \to A\} \vdash_{\forall \mathbf{L}^\mathbf{D}} C_n$, so by Lemma 4 (a):

$$T_n \cup \{(A \to B) \vee (B \to A)\} \vdash_{\forall \mathbf{L}^\mathbf{D}} C_n$$

But since $\vdash_{\mathbf{UL}} (A \to B) \vee (B \to A)$ (see e.g. [10]), we get $T_n \vdash_{\mathbf{L}^\mathbf{D}} C_n$ which contradicts the induction hypothesis. Hence, if (a) holds, let:

$$T_{n+1} = T_n \cup \{A \to B\} \text{ and } C_{n+1} = C_n \vee (B \to q) \vee (q \to A)$$

and if (b) holds, let:

$$T_{n+1} = T_n \cup \{B \to A\} \text{ and } C_{n+1} = C_n \vee (A \to q) \vee (q \to B)$$

Clearly T_{n+1} is linear in both cases, $T_{n+1} \not\vdash_{\forall \mathbf{L}^\mathbf{D}} C_{n+1}$, and $T_{n+1} \vdash_{\forall \mathbf{L}^\mathbf{D}} C_n \to C_{n+1}$. Moreover, if $T_{n+1} \not\vdash_{\forall \mathbf{L}} A \to B$, then $T_{n+1} \not\vdash_{\forall \mathbf{L}^\mathbf{D}} C_n \vee (A \to q) \vee (q \to B)$, so $T_{n+1} \not\vdash_{\forall \mathbf{L}} A \to q$ and $T_{n+1} \not\vdash_{\forall \mathbf{L}} q \to B$. Similarly, if $T_{n+1} \not\vdash_{\forall \mathbf{L}} B \to A$, then $T_{n+1} \not\vdash_{\forall \mathbf{L}} B \to q$ and $T_{n+1} \not\vdash_{\forall \mathbf{L}} q \to A$, so $\forall\mathbf{L}$-density holds.

(2) Suppose that the n-th task is dealing with the $\forall\mathbf{L}$-Henkin property for $\forall x A(x)$. Let c be a new constant not occurring in T_n, C_n, or $A(x)$.

- If $T_n \not\vdash_{\forall \mathbf{L}} C_n \vee A(c)$, then $T_n \not\vdash_{\forall \mathbf{L}} \forall x A(x)$, and let $T_{n+1} = T_n$, and $C_{n+1} = C_n \vee A(c)$.
- If $T_n \vdash_{\forall \mathbf{L}} C_n \vee A(c)$, then $T_n \vdash_{\forall \mathbf{L}} C_n \vee A(x)$ (replacing c in the proof by a variable x not occurring in C_n). Hence $T_n \vdash_{\forall \mathbf{L}} \forall x(C_n \vee A(x))$ by (gen), and by $(\forall 3)$, $T_n \vdash_{\forall \mathbf{L}} C_n \vee \forall x A(x)$. So $T_n \cup \{\forall x A(x) \to C_n\} \vdash_{\forall \mathbf{L}} C_n$. It follows that $T_n \cup \{C_n \to \forall x A(x)\} \not\vdash_{\forall \mathbf{L}} C_n$ (since otherwise $T_n \vdash_{\forall \mathbf{L}^\mathbf{D}} C_n$ a contradiction) and $T_n \cup \{C_n \to \forall x A(x)\} \vdash_{\forall \mathbf{L}} \forall x A(x)$ (using $T_n \vdash_{\forall \mathbf{L}} C_n \vee A(c)$). So let $T_{n+1} = T_n \cup \{C_n \to \forall x A(x)\}$ and $C_{n+1} = C_n$.

Now take $\hat{T} = \bigcup_{n \in \mathbb{N}} T_n$. $\hat{T}$ is linear and $\forall\mathbf{L}$-dense by construction. Also $\hat{T} \not\vdash_{\forall \mathbf{L}^\mathbf{D}} A$ since otherwise $T_n \vdash_{\forall \mathbf{L}^\mathbf{D}} A$ for some n, and since $T_n \vdash_{\forall \mathbf{L}^\mathbf{D}} C_i \to C_{i+1}$ for $i = 1 \ldots n$, $T_n \vdash_{\forall \mathbf{L}^\mathbf{D}} C_n$, a contradiction. Finally to see that $\hat{T}$ is $\forall\mathbf{L}$-Henkin, suppose that $\hat{T} \not\vdash_{\forall \mathbf{L}} \forall x A(x)$ where $\forall x A(x)$ is a sentence dealt with in step n. It follows that $T_{n+1} \not\vdash_{\forall \mathbf{L}} \forall x A(x)$. But then for step n in the above construction, we must have the first possibility $T_n \not\vdash_{\forall \mathbf{L}} C_n \vee A(c)$ where $C_{n+1} = C_n \vee A(c)$. Hence also $\hat{T} \not\vdash_{\forall \mathbf{L}} C_n \vee A(c)$. So $\hat{T} \not\vdash_{\forall \mathbf{L}} A(c)$. □

Lemma 7. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion and* T *a countable linear* $\forall\mathbf{L}$*-dense* $\forall\mathbf{L}$*-Henkin theory. If* $T \not\vdash_{\forall \mathbf{L}^\mathbf{D}} A$*, then there exists a countable dense L-chain and safe valuation* v *with non-empty domain* $\mathcal{D}$*, such that* $v(B) \geq t$ *for all* $B \in T$*, and* $v(A) < t$.

Proof. $\mathcal{A}_T^{\forall \mathbf{L}}$ is an L-algebra. Moreover, $\mathcal{A}_T^{\forall \mathbf{L}}$ is *linearly and densely ordered*, since for all sentences A, B: (1) either $A \to B \in T$ or $B \to A \in T$, and (2) if $T \not\vdash_{\forall \mathbf{L}} A \to B$, then $T \not\vdash_{\forall \mathbf{L}} A \to C$ and $T \not\vdash_{\forall \mathbf{L}} C \to B$ for some sentence C. Let $\mathcal{D}$ be the set of all constants of the vocabulary of T (adding one if necessary so that $\mathcal{D}$ is non-empty) and A. Define a valuation v with domain $\mathcal{D}$ such that $v(p(c_1, \ldots, c_m)) = [p(c_1, \ldots, c_m)]_T$ for each m-ary predicate p. We claim that $v(B) = [B]_T$ for all formulae B and hence in particular $v(B) \geq t$ for all $B \in T$, and $v(A) = [A]_T < [t]_T$ as required. We proceed by induction on the complexity of B, the atomic case holding by definition and the case of propositional connectives being easy. The quantifier cases follow using Lemma 2. □

Combining Lemmas 6 and 7 with Lemma 3 we obtain the following result:

Theorem 4. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion and* T *a countable theory. The following are equivalent:*

(1) $T \vdash_{\forall \mathbf{L}^{\mathbf{D}}} A$.
(2) For every dense L-chain and safe valuation v *with non-empty domain* $\mathcal{D}$*, if* $v(B) \geq t$ *for all* $B \in T$*, then* $v(A) \geq t$.

In particular, $\vdash_{\forall \mathbf{L}^{\mathbf{D}}} A$ *iff* A *is valid in all dense L-chains.*

As an interesting aside, note that $C = \forall x(A(x) \odot B) \to (\forall x A(x) \odot B)$ is valid in all dense BL-chains [8], where $\mathbf{BL}$ is Hájek's Basic fuzzy logic, the logic of continuous t-norms. So by the preceding theorem C is derivable in $\forall \mathbf{BL}^{\mathbf{D}}$. However, C is *not* valid in all BL-chains [7] and hence not derivable in $\forall \mathbf{BL}$. The density rule is therefore not admissible for $\forall \mathbf{BL}$, and it remains an intriguing question as to whether $\forall \mathbf{BL}^{\mathbf{D}}$ can be obtained as an *axiomatic* extension of this logic.

7 Rational Completeness

We are ready now to put together the various pieces and establish rational completeness for a wide class of first-order logics; i.e. show completeness with respect to dense chains. First, we need a way of connecting axiomatizations with hypersequent calculi.

Definition 13. *The* standard interpretation function I *is defined as follows:*

1. $I(\Gamma \Rightarrow C) = \odot\Gamma \to C$, $I(\Rightarrow C) = C$, $I(\Gamma \Rightarrow) = \odot\Gamma \to \bot$, *and* $I(\Rightarrow) = \bot$.
2. $I(\Gamma_1 \Rightarrow \Delta_1 \mid \ldots \mid \Gamma_n \Rightarrow \Delta_n) = I(\Gamma_1 \Rightarrow \Delta_1) \vee \ldots \vee I(\Gamma_n \Rightarrow \Delta_n)$.

Lemma 8. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion and* $\mathbf{L_L}$ *a (first-order) w-simple sequent calculus whose rules are reductive and substitutive, such that* $\vdash_{\mathbf{HL^C_L}} G$ *iff* $\vdash_{\forall \mathbf{L}} I(G)$*:*

$$\text{(a) } \vdash_{\mathbf{HL^C_L}+(D)} G \text{ iff } \vdash_{\forall \mathbf{L}^{\mathbf{D}}} I(G). \qquad \text{(b) } \vdash_{\forall \mathbf{L}} A \text{ iff } \vdash_{\forall \mathbf{L}^{\mathbf{D}}} A.$$

Proof. (a) For the left-to-right direction, suppose that $\vdash_{\forall \mathbf{L}} I(G) \vee I(\Gamma_1, p \Rightarrow \Delta) \vee I(\Gamma_2 \Rightarrow p)$ where p does not occur in G, Γ_1, Γ_2, or Δ. It follows easily that $\vdash_{\forall \mathbf{L}} I(G) \vee (p \to I(\Gamma_1 \Rightarrow \Delta)) \vee I(\Gamma_2 \Rightarrow p)$, and hence by $(density)$, that $\vdash_{\forall \mathbf{L}^{\mathbf{D}}} I(G) \vee I(\Gamma_1, \Gamma_2 \Rightarrow \Delta)$ as required. For the right-to-left direction, it is sufficient to show that $(density)$ (with $T = \emptyset$) is admissible for $\mathbf{HL^C_L} + (D)$. If $\vdash_{\mathbf{HL^C_L}+(D)} \Rightarrow (A \to p) \vee (p \to B) \vee C$ where p does not occur in A, B, or C, then (easily) $\vdash_{\mathbf{HL^C_L}+(D)} A \Rightarrow p \mid p \Rightarrow B \mid \Rightarrow C$. Hence by (D), $\vdash_{\mathbf{HL^C_L}+(D)} A \Rightarrow B \mid \Rightarrow C$. Using (EC), $(\vee, r)_1$, $(\vee, r)_2$, and $(\to, r)$, it follows that $\vdash_{\mathbf{HL^C_L}+(D)} \Rightarrow (A \to B) \vee C$ as required.

(b) The left-to-right direction is trivial. For the right-to-left direction, suppose that $\vdash_{\forall \mathbf{L}^{\mathbf{D}}} A$. Then by (a), $\vdash_{\mathbf{HL^C_L}+(D)} \Rightarrow A$, and by Theorem 2, $\vdash_{\mathbf{HL^C_L}} \Rightarrow A$. Hence by hypothesis and Definition 13, $\vdash_{\forall \mathbf{L}} A$. □

Our main theorem now states that the first order version of *any* core $\mathbf{UL}$-expansion which has a suitable hypersequent calculus is rational complete.

Theorem 5. *Let* $\mathbf{L}$ *be a core* $\mathbf{UL}$*-expansion and* $\mathbf{L_L}$ *a (first-order) w-simple sequent calculus whose rules are reductive and substitutive, such that* $\vdash_{\mathbf{HL^C_L}} G$ *iff* $\vdash_{\forall\mathbf{L}} I(G)$*. Then* $\forall\mathbf{L}$ *is rational complete, i.e. the following are equivalent:*

(1) $T \vdash_{\forall\mathbf{L}} A$*.*
(2) For every dense L-chain and safe valuation v *with non-empty domain* $\mathcal{D}$*, if* $v(B) \geq t$ *for all* $B \in T$*, then* $v(A) \geq t$*.*

In particular, $\vdash_{\forall\mathbf{L}} A$ *iff* A *is valid in all dense L-chains.*

Proof. From (1) to (2) is easy. If (2) holds, then by Theorem 4, $T \vdash_{\forall\mathbf{L^D}} A$. By Lemma 5 (a), there is a finite subset T_0 of T such that $T_0 \vdash_{\forall\mathbf{L^D}} A$, and by Lemma 5 (b), $\vdash_{\forall\mathbf{L^D}} C \to A$ for some confusion C of T_0. By Lemma 8 (b), $\vdash_{\forall\mathbf{L}} C \to A$ and, by Lemma 5 (c), $T_0 \vdash_{\forall\mathbf{L}} C$. Hence, by (mp), $T_0 \vdash_{\forall\mathbf{L}} A$ and therefore also $T \vdash_{\forall\mathbf{L}} A$. □

For example, let $\forall\mathbf{SMTL}$ and $\forall\mathbf{CnMTL}$ ($n \geq 2$) be the first-order versions of the logics $\mathbf{SMTL}$ [6] and $\mathbf{CnMTL}$ ($n \geq 2$) [4]. Hypersequent calculi for these logics, $\mathbf{HL^C_{SMTL}}$ and $\mathbf{HL^C_{CnMTL}}$ are obtained by adding hypersequent versions of the rules (wc) and (nc) to the calculus $\mathbf{H\forall FL^C_{ew}}$ for $\forall\mathbf{MTL}$ (see Examples 1 and 3). Hence:

Corollary 1. $\forall\mathbf{MTL}$ *,* $\forall\mathbf{SMTL}$*, and* $\forall\mathbf{CnMTL}$ *(*$n \geq 2$*) are rational complete.*

References

1. A. Avron. A constructive analysis of RM. *Journal of Symbolic Logic*, 52(4):939–951, 1987.
2. M. Baaz and R. Zach. Hypersequents and the proof theory of intuitionistic fuzzy logic. In *Proceedings of CSL 2000*, pages 187–201. LNCS, Springer-Verlag, 2000.
3. A. Ciabattoni. Automated generation of analytic calculi for logics with linearity. *CSL04: Computer Science Logic. LNCS*, 3210:503–517, 2004.
4. A. Ciabattoni, F. Esteva, and L. Godo. T-norm based logics with n-contraction. *Neural Network World*, 12(5):441–453, 2002.
5. P. Cintula and P. Hájek. On theories and models in fuzzy predicate logics. *Journal of Symbolic Logic*, 71(3):832–863, 2006.
6. F. Esteva, J. Gispert, L. Godo, and F. Montagna. On the standard and rational completeness of some axiomatic extensions of the monoidal t-norm logic. *Studia Logica*, 71(2):199–226, 2002.
7. F. Esteva and L. Godo. Monoidal t-norm based logic: towards a logic for left-continuous t-norms. *Fuzzy Sets and Systems*, 124:271–288, 2001.
8. P. Hájek. *Metamathematics of Fuzzy Logic*. Kluwer, Dordrecht, 1998.
9. S. Jenei and F. Montagna. A proof of standard completeness for Esteva and Godo's MTL logic. *Studia Logica*, 70(2):183–192, 2002.
10. G. Metcalfe and F. Montagna. Substructural fuzzy logics. To appear in *Journal of Symbolic Logic*. http://www.math.vanderbilt.edu/people/metcalfe/publications.
11. F. Montagna and H. Ono. Kripke semantics, undecidability and standard completeness for Esteva and Godo's logic MTL∀. *Studia Logica*, 71(2):227–245, 2002.
12. H. Ono and Y. Komori. Logic without the contraction rule. *Journal of Symbolic Logic*, 50:169–201, 1985.
13. M. Takano. Another proof of the strong completeness of the intuitionistic fuzzy logic. *Tsukuba J. Math.*, 11:851–866, 1984.
14. G. Takeuti and T. Titani. Intuitionistic fuzzy logic and intuitionistic fuzzy set theory. *Journal of Symbolic Logic*, 49(3):851–866, 1984.

Extracting the Resolution Algorithm from a Completeness Proof for the Propositional Calculus

Robert Constable and Wojciech Moczydłowski[*]

Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
{rc,wojtek}@cs.cornell.edu

Abstract. We prove constructively that for any propositional formula ϕ in Conjunctive Normal Form, we can either find a satisfying assignment of true and false to its variables, or a refutation of ϕ showing that it is unsatisfiable. This refutation is a resolution proof of $\neg\phi$. From the formalization of our proof in Coq, we extract Robinson's famous resolution algorithm as a Haskell program correct by construction. The account is an example of the genre of highly readable formalized mathematics.

1 Introduction

Recently Jean Gallier [1] gave a simple constructive proof that the resolution method for the propositional calculus is complete – noting that other proofs in the literature [2,3,4,5] are by contradiction, hence weaker. Gallier's method is to provide an explicit resolution algorithm and a correctness proof for it.

We establish a slightly stronger result in a different way, by giving a simple constructive proof, formalized in Coq. The Coq system enforces constructivity and extracts from this proof an efficient resolution program (in Haskell or OCaml). Our proof is the kind of argument one would see in a logic textbook. Coq guarantees that it is constructive – which is not as well established in Gallier's presentation. The result is also stronger because executable code is extracted.

Our presentation is an instance of the new genre of *formalized mathematics* which we have been promoting, in both type theory [6,7] and set theory [8]. A salient feature of this genre is precise clarity. Moreover, the definitions serve a dual purpose, defining data types as well as mathematical concepts. Likewise, the proofs provide justifications as well as algorithms. We strive to make the account more readable than ordinary mathematical texts. If we succeed, then there will be no question that formal text is a more useful way to present certain results in mathematics. The reader can judge these claims directly.

2 The Resolution Method

The resolution method applies to formulas in Constructive Normal Form (CNF), such $(P \vee Q) \wedge (\neg P) \wedge (\neg Q \vee P)$. These are built from variables (atoms) such

[*] The authors have been partly supported by NSF grant CNS-0614790.

as $P, Q, R, \ldots$ or negations of them. These positive or negative variables are called *literals*. Disjunctions of them are called *clauses*, so a CNF formula is a conjunction of clauses built out of literals.

Resolution is an attempt to *refute* a CNF formula ϕ. To refute is to show that the formula cannot be true no matter what truth values, $true$ or $false$, are assigned to the variables. What is so noteworthy is that a formula can be refuted by systematically applying one rule to its clauses over and over. The rules is called *resolution*; it resolves two clauses into a new one. Refutation starts with two clauses of ϕ and produces a new one by resolution and then is applied to either original clauses from ϕ or newly created ones.

Our result is that given a CNF formula ϕ, we can either refute it by producing a refutation r or show that it is satisfiable by producing an assignment of truth values, $true$, $false$, to the variables. We also show that if r refutes ϕ, then ϕ is not satisfiable. Thus $\neg\phi$ is valid, and indeed, the refutation r can be seen as a proof of $\neg\phi$.

2.1 Prerequisites

Booleans, denoted by *bool*, consist of two elements: $false$ and $true$. Natural numbers are denoted by nat. The standard operations on booleans are called *andb*, *orb* and *negb*. The notation $list\ A$ stands for lists with elements coming from A. We denote the empty list by [] and a list consisting of elements $a_1, a_2, ..., a_k$ by $[a_1, \ldots, a_k]$. The concatenation of lists l_1 and l_2 is denoted by $l_1 @ l_2$.

We will use the `typewriter font` for concepts in Coq. Our choice of names mostly coincides with its standard library. Lists are an important exception: in Coq, the list $[a_1, a_2, \ldots, a_k]$ is denoted by `a1::a2::...::ak::nil` and $l_1 @ l_2$ by `l1 ++ l2`. Thus in particular $[a]@l$ is rendered in Coq as `a::l` and the empty list [] by `nil`,

2.2 Literals, Clauses, Formulas and Valuations

We are interested in formulas in Conjunctive Normal Form (CNF). To build their representation in Coq, we represent propositional variables $P, Q, \ldots$ as natural numbers. To fix our attention, in the examples P, Q, R are represented by $1, 2$ and 3, respectively. A *literal* is either an atom or its negation. We represent the former case by labelling the atom with a tag *pos* and the latter by labelling it with a tag *neg*. A *clause* is a list of literals and a (CNF) *formula* is a list of clauses. For example, the formula $\phi \equiv (P \vee R) \wedge \neg P \wedge Q \wedge (P \vee \neg Q \vee \neg R)$ is represented by the list $[[pos\ 1, pos\ 3], [neg\ 1], [pos\ 2], [pos\ 1, neg\ 2, neg\ 3]]$. The function *notlit* for a given literal returns its negation: $notlit(pos\ n) = neg\ n, notlit(neg\ n) = pos\ n$.

The formalization of these definitions in Coq follows. Anything between `(* *)` characters is a comment.

```
(* lit is a disjoint union nat + nat.
   Its components are natural numbers labelled by pos and neg *)
```

```
Inductive lit : Set :=
         pos : nat -> lit
       | neg : nat -> lit.

Definition notLit (l : lit) : lit :=
  match l with
      pos n => neg n
    | neg n => pos n
  end.

Definition clause := list lit.

Definition form := list clause.

Definition P : nat := 1.
Definition Q : nat := 2.
Definition R : nat := 3.
Definition phi : form := (pos  P:: (pos R) :: nil) ::
                         (neg P :: nil) ::
                         (pos Q:: nil) ::
                         (pos P :: (neg Q) ::(neg R) :: nil) ::
                         nil.
```

A *valuation* assigns booleans to atoms in a formula. We represent valuations as lists of natural numbers. The intended meaning, captured in the formal definitions below, is that the valuation v assigns *true* to the atom represented by a number n, if n is an element of v. For example, the valuation $\{P \rightarrow false, Q \rightarrow true, R \rightarrow true\}$ is represented as $[2, 3]$.

```
Definition val := list nat.
```

The value of a literal *pos n* under a valuation v is equal to *true* if n is an element of v. The value of a negated literal, *neg n*, is equal to $false$ if n is not an element of v.

```
(* valLit : lit -> val -> bool *)
Definition valLit (l : lit) (v : val) :=
           match l with
               pos n => elemL nat eqNat n v
             | neg n => negb (elemL nat eqNat n v)
           end.
```

The value of a clause c under the valuation v is an element of *bool*. It is defined by recursion on c. If c is empty, the value is $false$. Otherwise, if c is a list with the first element x and the rest of its elements denoted by xs, it is a disjunction of the value of the literal x and the value of xs. The value of a formula f is defined similarly.

```
(* valClause : clause -> val -> bool
   valClause nil v = false
   valClause (x::xs) v = orb (valLit x v) (valClause xs v)
*)
Fixpoint valClause (c : clause) (v : val) { struct c } : bool :=
         match c with
              nil => false
            | x :: xs => orb (valLit x v) (valClause xs v)
         end.

Fixpoint valForm (f : form) (v : val) { struct f } : bool :=
         match f with
              nil => true
            | x :: xs => andb (valClause x v) (valForm xs v)
         end.
```

Note that the "empty" formula is always true, while the "empty" clause is always false. We will sometimes say that the valuation v *falsifies* a formula f to mean that $valForm(f)(v) = false$.

A formula f is *satisfiable* if there is a valuation which sets it to true. Or, in other words, if the set of valuations v such that $valForm(f)(v) = true$ is not empty. In Coq's type theory, truth of a proposition is equivalent to the non-emptiness of a corresponding type, so the following definition captures satisfiability correctly.

```
Definition satisfiable (f : form) := { v : val| valForm f v = true }.
```

A reader unfamiliar with type theory can think about `{ v : val | valForm f v = true }` as a (witness-providing) existential quantifier:

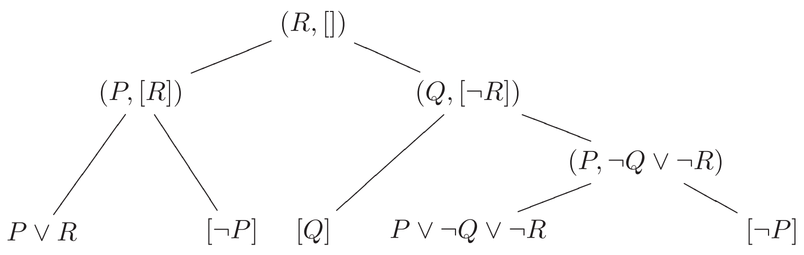
`{ v : val | valForm f v = true }` $\approx \exists v : val.\ valForm(f)(v) = true$

2.3 Resolutions

The definition of the resolution tree we use is taken from [1]. Structurally, it is a binary tree. Its leaves are labelled with clauses. Its nodes are labelled with pairs consisting of a literal and a clause.

```
Inductive resol : Set :=
        leaf : clause -> resol
      | node : lit -> clause -> resol -> resol -> resol.
```

For example, the tree T defined as[1]:



is represented as:

```
(* The prefix ex stands for ''example'' *)
Definition exTree : resol :=
 node (pos R) (nil)
   (node (pos P) (pos R::nil)
      (leaf (pos P::(pos R)::nil))
      (leaf (neg P::nil))
   )
   (node (pos Q) (neg R::nil)
      (leaf (pos Q::nil))
      (node (pos P) (neg Q::(neg R)::nil)
         (leaf (pos P::(neg Q)::(neg R)::nil))
         (leaf (neg P::nil))
      )
  ).
```

Note that each node in a tree is labelled with a clause. Let us denote this clause by `clauseR`:

```
Definition clauseR (r:resol) : clause := match r with
     leaf c => c
   | node l c r1 r2 => c
   end.
```

The *premises* of a tree are the clauses at its nodes.

```
Fixpoint premises (r : resol) {struct r}: list clause:= match r with
       leaf c => c::nil
     | node l c r1 r2 => (premises r1) ++ (premises r2)
   end.
```

So the premises of T are $[P \vee R, [\neg P], [Q], P \vee \neg Q \vee \neg R, [\neg P]]$:

[1] To increase readability, we render clauses with more than one element in a traditional way instead of their list representation.

```
Lemma ex1 :
 premises exTree =
  (pos P::(pos R)::nil)::
  (neg P::nil)::
  (pos Q::nil)::
  (pos P::(neg Q)::(neg R)::nil)::
  (neg P::nil)::
  nil.
```

The correctness restriction makes the trees more meaningful. Intuitively, each node should correspond to a resolution step. A clause c at the node results from the clauses c_1, c_2 at its children by removing l form c_1, the negation of l from c_2 and retaining the rest of c_1, c_2. So $c = (c_1 - \{l\}) \cup (c_2 - \{\neg l\})$.

In the formal definition, `eqCS` denotes equality of clauses treated as finite sets and `elemC l c` checks whether the literal l is an element of the clause c.

```
Fixpoint correctR (r : resol) : bool := match r with
    leaf c => true
  | node l c r1 r2 =>
       andb (andb (correctR r1) (correctR r2))
            (andb
                 (andb (elemC l (clauseR r1))
                       (elemC (notLit l) (clauseR r2)))
                 (eqCS c (removeC l (clauseR r1) ++
                         (removeC (notLit l) (clauseR r2)))))
         end.
```

Resolutions therefore are these trees that satisfy the correctness condition.

```
Definition res : Set := { r : resol | correctR r = true }.
```

We can easily show that $correctR(exTree) = true$ and define an element *exRes* of *res* corresponding to the tree *exTree*.

```
(* p denotes the proof that correctR exTree = true *)
Definition exRes : res := exist _ exTree p.
```

Having defined the resolution trees, we need to relate them to formulas. A tree r *corresponds* to a formula f if the premises of r are a subset of the clauses of f. A tree r *refutes* a formula f if it corresponds to f and its root is labelled with the empty clause. We call a formula *refutable*, if there is a resolution tree which refutes it.

```
(* subL clause eqC f1 f2 checks whether f1 is a subset of f2 *)
(* proj1_sig, given a resolution, returns the underlying tree.
   So proj1_sig exRes = exTree. *)
Definition corresponds (r : res) (f : form) :=
     subL clause eqC (premises (proj1_sig r)) f.
```

```
Definition refutes (r: res) (f : form) :=
 andb (corresponds r f)
      (eqC (clauseR (proj1_sig r)) nil).
```

The proof that the resolution tree *exRes* refutes ϕ is straightforward.

```
Lemma ex2 : refutes exRes phi = true.
```

2.4 Refuted Formulas Are Unsatisfiable

To show that we have chosen our definitions correctly, we prove the following theorem:

Theorem 1. *For all resolutions r and formulas f, if r refutes f, then f is not satisfiable.*

In other words:

```
Theorem resres : forall (r : res) (f : form),
     refutes r f = true ->
          satisfiable f -> False.
```

Proof (Sketch). Take a resolution r, suppose that r refutes f and take a valuation v which sets f to *true*. We know that the root of r is labelled by the empty clause. In this situation we can find a clause c in the premises of r such that v sets c to false. We do this by starting from the root of r and proceeding down, choosing at the node labelled by l the left child if $v(l) = false$ and the right one otherwise. We take c to be the clause labelling the leaf we end at. Furthermore, we collect on the way literals in the following way: if a node is labelled by l and $v(l) = false$, we collect l; otherwise we collect the negation of l. Let d denote the resulting clause. We can easily prove that c is a subclause of d and that v falsifies d. Thus also $v(c) = false$.

Since r refutes f, the clause c is among the clauses of f. It is therefore easy to see that the value of f under v must be false as well. Therefore $valForm(f)(v) = false$, by the assumption about v we also have $valForm(f)(v) = true$ and since $true \neq false$, we get the claim.

We will now show how to make this proof precise. First, we formalize the process of "proceeding down, starting from the root of r", which resulted in the clause c. For the examples, we use our valuation $\{P \to false, Q \to true, R \to true\}$.

```
Fixpoint contraClause (r : resol) (v : val) { struct r} : clause :=
     match r with
         leaf c => c
        |node l c r1 r2 =>
              if valLit l v then contraClause r2 v
                                 else contraClause r1 v
     end.
```

```
Definition exVal : val := Q::R::nil.
```

```
Lemma ex3 : contraClause exTree exVal = pos P::(neg Q)::(neg R)::nil.
```

Lemma 1. *If r corresponds to f, then the clause picked using contraClause is equal to one of the clauses of f.*

```
Lemma cc1 : forall (r : res) (f : form) (v : val),
       corresponds r f = true ->
           elemL _ eqC (contraClause (proj1_sig r) v) f = true.
```

Second, we formalize "collecting literals on the way":

```
Fixpoint lontraClause (r : resol) (v : val) { struct r} :  clause :=
     match r with
         leaf c => nil
        |node l c r1 r2 =>
              if valLit l v then (notLit l)::lontraClause r2 v
                            else l::(lontraClause r1 v)
     end.
```

```
Lemma ex4 : lontraClause exTree exVal = neg R::(neg Q)::(pos P)::nil.
```

Lemma 2. *For any resolution r and valuation v, v falsifies* `lontraClause r v`.

```
Lemma lv1 : forall (r : res) (v : val),
     valClause (lontraClause (proj1_sig r) v) v = false.
```

These definitions enable us to state and prove the main lemma:

```
Lemma lc1 : forall (r : res) (f : form) (v : val),
   let tree := proj1_sig r in
        subL _ eqLit (contraClause tree v)
                ((clauseR tree) ++ (lontraClause tree v)) = true.
```

Proof. Straightforward induction on r.

With Lemma $lc1$ at hand, we show

```
Lemma lv2 : forall (r : res) (f : form) (v : val),
     refutes r f = true ->
          valClause (contraClause (proj1_sig r) v) v = false.
```

Proof. Since r refutes f, the root of r is labelled by the empty clause. Let c denote $contraClause(r)(v)$ and let l denote $lontraClause(r)(v)$. By Lemma $lc1$, $c \subseteq l$. By Lemma $lv1$, $valClause(l)(v) = false$. The claim easily follows.

Now we can show formally what we just sketched at the beginning of this section:

The proof of Theorem 1. Take any f, r refuting f and a valuation v. Suppose the value of f under v is *true*. Let c denote $contraClause(r, v)$. By Lemma $lv2$, the value of c under v is false. By Lemma $cc1$, c is one of the clauses of f, thus also the value of f under v is false, which contradicts our assumption. Therefore f is not satisfiable.

2.5 Graft and Percolate

There are two operations on the trees which we will use in the proof of the final theorem. Both are taken from [1].

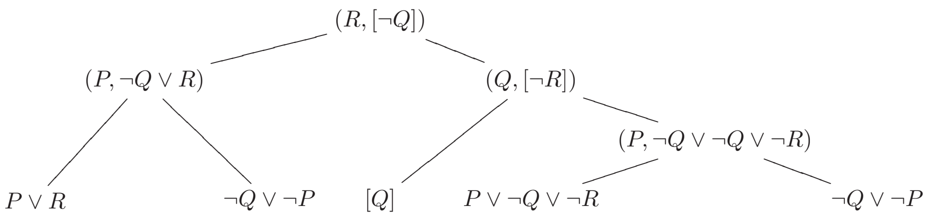
The *percolate* function takes a tree r, a clause c and a literal l as its arguments. It appends l to all clauses in the premises r which are equal (as finite sets) to c. It further "percolates" l up the tree: it travels towards the root and appends l to all clauses labelling nodes on the way. It stops when it either reaches the root or the node which utilizes l in the resolution step. The formal definition follows; the helper function `percolate0` returns additionally a boolean value, set to `false` if the percolating process is to continue and to `true` otherwise.

```
(* (fun x y z => M) is Coq's notation for functional abstraction:
   (fun x y z => M) is a function which given arguments x y z returns M *)
Fixpoint percolate0 (r : resol) (a : clause) (l : lit)
                            { struct r} : resol * bool :=
 match r with
    leaf c => if eqC c a then (leaf (l::c), false) else (r, true)
   |node resL c r1 r2 =>
      (fun p1 p2 cr  => (node resL (fst cr) (fst p1) (fst p2), (snd cr)))
        (percolate0 r1 a l)
        (percolate0 r2 a l)
        (let p1 := percolate0 r1 a l in
         let p2 := percolate0 r2 a l in
         let b1 := snd p1 in
         let b2 := snd p2 in
         let eqll := eqLit resL l in
         let eqlnl := eqLit resL (notLit l) in
           match (b1, b2) with
            (true, true) => (c, true)
           |(true, false) => if eqlnl then (c, true)
                             else ((l::c), false)
           |(false, true) => if eqll  then (c, true)
                                     else ((l::c), false)
           |(false, false) => ((l::c), false)
          end)
    end.
```

```
Definition percolate (r : resol) (a : clause) (l : lit) : resol :=
  fst (percolate0 r a l).
```

```
Definition exPerc := percolate exTree (neg P::nil) (neg Q).
```

For example, `exPerc` is the following tree. The $\neg Q$ attached to the left $[\neg P]$ clause percolated to the top, while the process of percolating the one attached to the right $[\neg P]$ clause was stopped at the node labelled with $(P, \neg Q \vee \neg Q \vee \neg R)$.



The main fact about the percolate operation is that it preserves the correctness condition of resolution trees:

```
Lemma percolateCorrect : forall (r : resol) (a : clause) (l : lit),
       correctR r = true -> correctR (percolate r a l) = true.
```

The *graft* operation takes two trees r, s and attaches s to any leaf of r labelled with the clause equal to the root clause of r.

```
Fixpoint graft (r s : resol) { struct r} : resol :=
 match r with
     leaf c => if eqC (clauseR s) c then s else r
    |node l c r1 r2 => node l c (graft r1 s) (graft r2 s)
 end.
```

Grafting preserves the correctness condition as well.

```
Lemma graftCorrect : forall (r s : resol),
  correctR r = true ->
  correctR s = true ->
    correctR (graft r s) = true.
```

2.6 The Completeness Theorem

We are now ready to present the completeness theorem. It will be proved by measure induction on formulas. The *measure* of a formula $f = [c_1, \ldots, c_k]$ is defined as the number of disjunction symbols in the formula: $measure(f) = \Sigma_{i=1\ldots k} pred(length(c_k))$, where *length* for a given list returns its length and *pred* denotes the total predecessor function on natural numbers: $pred(0) = 0, pred(n+1) = n$.

```
Fixpoint measure (f : form) : nat :=
         match f with
                nil => 0
              | x :: xs => pred (length x) + (measure xs)
         end.
```

We call a formula *one-literal* if all its clauses have at most one literal. Any formula of measure 0 is one-literal:

```
Lemma 10 : forall f : form, measure f = 0 -> onelL lit f = true.
```

For any formula f, we provide a definition of a predicate stating that either f is satisfiable or refutable:

```
Definition fPred (f : form) : Set :=
   sum (satisfiable f) ({ r : res | refutes r f  = true}).
```

We first tackle the base case of the inductive argument:

```
Lemma 13 : forall f : form, onelL lit f = true  -> fPred f.
```

Proof. By induction on f (as a list). If f is empty, then it is trivially satisfiable with any valuation as a witness. For the inductive step, suppose we have the claim for f. We need to show it for f extended with any clause a. So suppose $[a]@f$ is one-literal. Then obviously so is f, so $fPred(f)$ holds. We have two cases to consider:

– f is satisfiable. Let v be the valuation which sets f to true. We know that a has at most one literal. If a is empty, then the resolution tree consisting of a single leaf labelled with the empty clause refutes f (recall that the value of the empty clause is $false$). Otherwise, it consists of a single literal l. Let *notl* denote the negation of l. We have 3 subcases to consider:
  - [*notl*] is one of the clauses of f. Then the resolution tree with two leaves labelled by $[l]$ and [*notl*] and the node labelled with the pair $(l, [])$ refutes f.
  - $[l]$ is one of the clauses of f. Then $[a]@f$ is satisfiable by v as well, as v must set the value of $[l]$ to *true*.
  - Neither [*notl*] nor $[l]$ is a clause of f. Then $[a]@f$ is satisfiable by v extended to set a to true.

– There is a resolution r refuting f. Then it is easy to see that r refutes $[a]@f$ as well.

Finally, we prove the main theorem.

Theorem 2. *Any formula is either satisfiable or refutable.*

```
Theorem t : forall f : form, fPred f.
```

Proof. We proceed by measure induction on f using the function *measure*. The case where $measure(f) = 0$ is handled by Lemmas 10 and 13. Otherwise, $measure(f) > 0$. This implies that there is a clause c in f with at least two literals. Thus we can write c as $[l_1, l_2]@x_0$ for some literals l_1, l_2 and clause x_0. Let Δ denote the formula resulting by removing all occurences of c from f. Let f_1 denote the formula $[l_1]@\Delta$ and let f_2 denote the formula $[l_2, x_0]@\Delta$. It is easy to see that the measures of both f_1 and f_2 are smaller than the measure of f. By the inductive hypothesis, we therefore have several cases to consider.

- f_1 is satisfiable. Then there is a valuation v setting f_1 to *true*. Thus v must set $[l_1]$ and Δ to true, so it also sets $[l_1, l_2]@x_0$ and Δ to true, which implies that the value of f under v is *true*, so f is satisfiable.
- f_2 is satisfiable. Reasoning in the same way, we can derive satisfiability of f.
- There are resolution trees r_1, r_2 refuting f_1 and f_2, respectively. Let r_p denote $percolate(r_2)([l_2]@x_0)(l_1)$. Therefore, r_p results from r_2 by appending l_1 to all leaves labelled by $[l_2]@x_0$ and percolating it up. Since r_2 is a refutation tree, this means that the clause at the root of r_p is either the empty clause or the clause $[l_1]$. In the former case, by Lemma *percolateCorrect*, it is easy to see that r_p refutes f. In the latter, we have two subcases to consider:
  - $[l_1]$ is not among the premises of r_1. In this case, the premises of r_1 are a subset of Δ, so also a subset of f, so r_1 already refutes f.
  - $[l_1]$ is among the premises of r_1. Let r_g denote $graft(r_1)(r_p)$. Therefore, r_g results by replacing every leaf in r_1 labelled with $[l1]$ by r_p. Then r_g refutes f. To show this, we need to show that:
    - r_g corresponds to f. This means that the premises of r_g are a subset of the clauses of f. But note that the premises of r_1 are a subset of $\Delta \cup [l_1]$. By grafting r_p onto r_1, the leaves labelled by $[l_1]$ disappear replaced by r_p. As the premises of r_p are a subset of $\Delta \cup [l_1, l_2]@x_0$, the premises of r_g are a subset of $\Delta \cup [l_1, l_2]@x_0$ and thus also a subset of f.
    - The root of r_g is labelled by the empty clause. This follows trivially by r_1 refuting f_1 and the definition of the grafting function.
    - The tree r_g is a correct resolution tree. This follows easily by Lemmas *graftCorrect* and *percolateCorrect*.

Let us see how the proof works for the formula $f \equiv (P \vee Q) \wedge (\neg P) \wedge (\neg Q)$. Obviously, ϕ is not one-literal. We take $c = P \vee Q$. Then $l_1 = P$, $l_2 = Q$, $x_0 = []$, $\Delta = (\neg P) \wedge (\neg Q)$, $f_1 = P \wedge \neg P \wedge \neg Q$, $f_2 = Q \wedge \neg P \wedge \neg Q$. The application of the inductive hypothesis to f_2 and f_1 results in two refutation tree r_2 and r_1:

$$\begin{tikzcd}
 & r_2 & & & r_1 & \\
 & (Q, []) \arrow[dl, dash] \arrow[dr, dash] & & & (P, []) \arrow[dl, dash] \arrow[dr, dash] & \\
{[Q]} & & {[\neg Q]} & {[P]} & & {[\neg P]}
\end{tikzcd}$$

Then the tree r_p is:

$$\begin{tikzcd}
& (Q,[P]) \arrow[dl, dash] \arrow[dr, dash] & \\
P \vee Q & & [\neg Q]
\end{tikzcd}$$

Since the clause $[P]$ is obviously not empty, we need to perform the graft operation which results in a tree g, which can be easily seen to refute f.

$$\begin{tikzcd}
& & (P,[]) \arrow[dl, dash] \arrow[dr, dash] & \\
& (Q,[P]) \arrow[dl, dash] \arrow[dr, dash] & & [\neg P] \\
P \vee Q & & [\neg Q] &
\end{tikzcd}$$

3 Extraction

Now we can reap the benefits of the constructive proof and formalization in a theorem prover as Underwood did for the tableaux method [9]. Coq provides a program extraction capability, which makes it possible to extract a program from our proof. The program, given a representation of a formula f in a CNF form, returns either a valuation satisfying f or a resolution tree refuting f. The extracted program is correct by construction — there is no need for testing for eventual bugs, as the program is *guaranteed* to compute the refuting resolution or the satisfying valuation.

We have chosen to extract a program in a functional programming language Haskell. Coq also offers choices of other programming languages (Scheme, ML). We now show examples of interaction with our program loaded in a Haskell interpreter. The examples correspond to the lemmas and theorems we have proven.

We can check directly the truth of several lemmas in the interpreter. To make the output more readable, we have added a function which renders lists in the style used in this paper. We use standard notation for natural numbers, rendering $S(S(0))$ as simply 2. The `Main>` string in the examples is the standard prompt of the Haskell interpreter.

```
Main> premises exTree
[[Pos 1, Pos 3], [Neg 1], [Pos 2], [Pos 1, Neg 2, Neg 3], [Neg 1]]
Main> refutes exRes phi
True
Main> contraClause exTree exVal
[Pos 1, Neg 2, Neg 3]
Main> lontraClause exTree exVal
[Neg 3, Neg 2, Pos 1]
Main> exPerc
Node (Pos 3) [Neg 2] (Node (Pos 1) [Neg 2, Pos 3] (Leaf [Pos 1, Pos 3])
(Leaf [Neg 2, Neg 1])) (Node (Pos 2) [Neg 3] (Leaf [Pos 2]) (Node (Pos 1)
[Neg 2, Neg 2, Neg 3] (Leaf [Pos 1, Neg 2, Neg 3]) (Leaf [Neg 2, Neg 1])))
Main>
```

Finally, let us see the application of the completeness theorem:

```
Main> f
[[Pos 1, Pos 2], [Neg 1], [Neg 2]]
Main> t f
Inr (Node (Pos 1) [] (Node (Pos 2) [Pos 1] (Leaf [Pos 1, Pos 2]) (Leaf [Neg 2]))
(Leaf [Neg 1]))
Main> let Inr x = t f in refutes x f
True
```

To give the reader a glimpse of the actual Haskell code, we show the Haskell definition of the function t corresponding to Theorem 2.

```
t :: Form -> FPred
t f =
  induction_ltof1 measure
   (\x h ->
    sumbool_rec
     (\_ -> l3 x)
     (\_ ->
      and_rec (\_ _ ->
        case lm1 x of
          Nil -> false_rec __
          Cons l x0 ->
            (case x0 of
               Nil -> false_rec __
               Cons l2 x1 ->
                let delta = removeL eqC (Cons l (Cons l2 x1)) x in
                 sum_rec
                   (\a -> Inl (sat1 l (Cons l (Cons l2 x1)) x a))
                   (\b0 -> sum_rec
                     (\a -> Inl (sat2 (Cons l2 x1) (Cons l (Cons l2 x1)) x a))
                     (\b1 ->
                       Inr (
                         let d1 = proj1_sig b1 in
                         let d11 = percolate d1 (Cons l2 x1) l in
                          sumbool_rec
                           (\_ -> d11)
                           (\_ ->
                             sumbool_rec
                              (\_ -> graft b0 d11)
                              (\_ -> b0)
                              (btf (elemL eqC (Cons l Nil) (premises b0))))
                           (eq_rec (clauseR d1) (p1 d1 (Cons l2 x1) l) Nil)))
                     (h (Cons (Cons l2 x1) delta) __))
                   (h (Cons (Cons l Nil) delta) __))))
     (case measure x of
        O -> Left
        S n -> Right))
   f
```

4 Conclusion

While proofreading this paper, we have discovered several times mistakes in our informal presentation thanks to the included Coq and Haskell code. This is an example of successful *paper verification* using a proof assistant.

Richard Eaton and the second author have also done this proof in Nuprl [6] and are writing a comparison of the systems in their ability to support readable formalized mathematics. This forthcoming article will discuss several subtle points about extraction and how to guide it to produce code as efficient as what can be written directly. At this point, we only mention that one of the most important differences between Coq and Nuprl, namely the treatment of equality, played no role in our developments. As the proofs use only very basic properties of equality, the extensionality of Nuprl and intensionality of Coq did not influence our development.

References

1. Gallier, J.H.: The completeness of propositional resolution: A simple and constructive proof. Logical Methods in Computer Science **2**(5) (2006) 1–7
2. Chang, C.C., Keisler, H.J.: Model Theory. Volume 73 of Studies in Logic and the Foundations of Mathematics. North-Holland, Netherlands (1973)
3. Robinson, J.: A machine oriented logic based on the resolution principle. Journal of the Association of Computing Machinery **12** (1965) 23–41
4. Lewis, H.R., Papadimitriou, C.H.: Elements of the Theory of Computation. Prentice-Hall, Englewood Cliffs, New Jersey (1994)
5. Gallier, J.H.: Logic for Computer Science, Foundations of Automatic Theorem Proving. Harper and Row, NY (1986)
6. Constable, R.L., Allen, S.F., Bromley, H.M., Cleaveland, W.R., Cremer, J.F., Harper, R.W., Howe, D.J., Knoblock, T.B., Mendler, N.P., Panangaden, P., Sasaki, J.T., Smith, S.F.: Implementing Mathematics with the Nuprl Proof Development System. Prentice-Hall, NJ (1986)
7. Constable, R.L., Howe, D.J.: Implementing metamathematics as an approach to automatic theorem proving. In Banerji, R.B., ed.: Formal Techniques in Artificial Intelligence: A Source Book. Elsevier Science Publishers (North-Holland) (1990) 45–76
8. Constable, R., Moczydłowski, W.: Extracting Programs from Constructive HOL Proofs via IZF Set-Theoretic Semantics. In: Proceedings of 3rd International Joint Conference on Automated Reasoning (IJCAR 2006). Volume 4130 of Lecture Notes in Computer Science., Springer (2006) 162–176
9. Underwood, J.L.: The tableau algorithm for intuitionistic propositional calculus as a constructive completeness proof. In: Proceedings of the Workshop on Theorem Proving with Analytic Tableaux, Marseille, France. (1993) 245–248 Available as Technical Report MPI-I-93-213 Max-Planck-Institut für Informatik, Saarbrücken, Germany.

Topological Semantics and Bisimulations for Intuitionistic Modal Logics and Their Classical Companion Logics[⋆]

J.M. Davoren

Department of Electrical & Electronic Engineering,
The University of Melbourne, VIC 3010 Australia
davoren@unimelb.edu.au

Abstract. We take the well-known intuitionistic modal logic of Fischer Servi with semantics in bi-relational Kripke frames, and give the natural extension to topological Kripke frames. Fischer Servi's two interaction conditions relating the intuitionistic pre-order (or partial-order) with the modal accessibility relation generalise to the requirement that the relation and its inverse be lower semi-continuous with respect to the topology. We then investigate the notion of topological bisimulation relations between topological Kripke frames, as introduced by Aiello and van Benthem, and show that their topology-preserving conditions are equivalent to the properties that the inverse-relation and the relation are lower semi-continuous with respect to the topologies on the two models. Our first main result is that this notion of topological bisimulation yields semantic preservation w.r.t. topological Kripke models for both intuitionistic tense logics, and for their classical companion multi-modal logics in the setting of the Gödel translation. After giving canonical topological Kripke models for the Hilbert-style axiomatizations of the Fischer Servi logic and its classical multi-modal companion logic, we show that the syntactic Gödel translation induces a natural semantic map from the intuitionistic canonical model into the canonical model of the classical companion logic, and this map is itself a topological bisimulation.

1 Introduction

Topological semantics for intuitionistic logic and for the classical modal logic S4 have a long history going back to Tarski and co-workers in the 1930s and 40s, predating the relational Kripke semantics for both [25,31]. A little earlier again is the 1933 Gödel translation GT [21] of intuitionistic logic into classical S4. The translation makes perfect sense within the topological semantics: where $\Box$ is interpreted by topological interior, the translation $\mathrm{GT}(\neg\varphi) = \Box\neg\,\mathrm{GT}(\varphi)$

[⋆] Partially supported by Australian Research Council grants DP0208553 and LX0242359. The author acknowledges valuable discussions with R.P. Goré, B.D. Humberstone, S. Demri, J. Goubault-Larrecq, A. Nerode, T. Moor and V. Coulthard. The workshop paper [12] (available only in a tech report) is a precursor to this paper.

says that intuitionistic negation calls for the *interior* of the complement, and not just the complement. In the topological semantics, a basic semantic object is the *denotation set* $[\![\varphi]\!]^{\mathcal{M}}$ of a formula φ, consisting of the set of all states/worlds of the model $\mathcal{M}$ at which the formula is true, and the semantic clauses of the logic are given in terms of operations on sets of states. The intuitionistic requirement on the semantics is that all formulas must denote open sets: that is, sets that are equal to their own interior. Any formula φ partitions the state space X into three disjoint sets: the two open sets $[\![\varphi]\!]^{\mathcal{M}}$ and $[\![\neg\varphi]\!]^{\mathcal{M}}$, and the closed set $bd([\![\varphi]\!]^{\mathcal{M}})$, with the points in the topological boundary set $bd([\![\varphi]\!]^{\mathcal{M}})$ falsifying the law of excluded middle, since they neither satisfy nor falsify φ.

For the extension from intuitionistic propositional logics to intuitionistic modal logics, Fischer Servi in the 1970s [16,17,18] developed semantics over bi-relational Kripke frames, and this work has generated a good deal of research [15,20,22,29,32,36,37]. In bi-relational frames $(X, \preccurlyeq, R)$ where $\preccurlyeq$ is a pre-order (quasi-order) for the intuitionistic semantics, and R is a binary accessibility relation on X for the modal operators, the two Fischer Servi conditions are equivalent to the following relation inclusions [18,29,32]:

$$(R^{-1} \circ \preccurlyeq) \subseteq (\preccurlyeq \circ R^{-1}) \qquad \text{and} \qquad (R \circ \preccurlyeq) \subseteq (\preccurlyeq \circ R) \tag{1}$$

where $\circ$ is relational/sequential composition, and $(\cdot)^{-1}$ is relational inverse. Axiomatically, the base Fischer Servi modal logic **IK** has normality axioms for both the modal box $\boxdot$ and the diamond $\diamond$, as well as the additional two axiom schemes:

$$\mathbf{FS1}: \ \diamond(\varphi \to \psi) \to (\boxdot\varphi \to \diamond\psi) \ \text{ and } \ \mathbf{FS2}: \ (\diamond\varphi \to \boxdot\psi) \to \boxdot(\varphi \to \psi) \tag{2}$$

A study of various normal extensions of **IK** is given in [32]. Earlier, starting from the 1950s, the intuitionistic S5 logic **MIPC** [30,8] was given algebraic semantics in the form of *monadic Heyting algebras* [4,27,28,34,35][1] and later as bi-relational frames with an equivalence relation for the S5 modality [5,14,28,34]. This line of work has focused on $\mathbf{MIPC} = \mathbf{IK} \oplus \mathbf{T}\boxdot\diamond \oplus \mathbf{5}\boxdot\diamond$ and its normal extensions[2], and translations into intuitionistic and intermediate predicate logics. Within algebraic semantics, topological spaces arise in the context of Stone duality, and in [4,5,14], the focus restricts to Stone spaces (compact, Hausdorff and having as a basis the Boolean algebra of closed-and-open sets).

In this paper, following [12], we give semantics for intuitionistic modal logic over topological Kripke frames $\mathcal{F} = (X, \mathcal{T}, R)$, where $(X, \mathcal{T})$ is a topological space and $R \subseteq X \times X$ is an accesibility relation for the modalities; the Fisher Servi bi-relational semantics are straight-forwardly extended from pre-orders $\preccurlyeq$ on X and their associated *Alexandrov topology* $\mathcal{T}_{\preccurlyeq}$, to arbitrary topological spaces

[1] The additional *monadic* operators are $\forall$ and $\exists$ unary operators behaving as S5 box and diamond modalities, and come from Halmos' work on monadic Boolean algebras.

[2] Here, $\mathbf{T}\boxdot\diamond$ is the conjunction of the separate $\boxdot$ and $\diamond$ characteristic schemes for reflexivity, and likewise $\mathbf{5}\boxdot\diamond$ for Euclideanness, so together they characterize equivalence relations.

$(X, \mathcal{T})$[3]. Over topological Kripke frames, the two Fischer Servi bi-relational conditions on the interaction between modal and intuitionistic semantics ((1) above) generalize to *semi-continuity* properties of the relation R, and of its inverse R^{-1}, with respect to the topology. As for the base logic, Fischer Servi's extension of the Gödel translation reads as a direct transcription of the topological semantics. The translation GT(⊡φ) = □⊡ GT(φ) says that the intuitionistic box requires the interior of the classical box operator, since the latter is defined by an intersection and may fail to preserve open sets. In contrast, the translation clause GT(⟐φ) = ⟐ GT(φ) says that, semantically, the operator ⟐ preserves open sets. This condition is exactly the lower semi-continuity (l.s.c.) condition on the accessibility relation, and corresponds to the first Fischer Servi bi-relational inclusion $R^{-1}\circ \preccurlyeq \subseteq \preccurlyeq \circ R^{-1}$ in (1), and it is this condition that is required to verify topological soundness of the axiom scheme **FS1** in (2)[4]. Similarly, Fischer Servi's second bi-relational inclusion $R \circ \preccurlyeq \subseteq \preccurlyeq \circ R$ generalizes to the l.s.c. property of the R^{-1} relation, where the latter is required to verify topological soundness of the axiom scheme **FS2** in (2).

The symmetry of the interaction conditions on the modal relation R and its inverse R^{-1} means that we can – with no additional semantic assumptions – lift the topological semantics to intuitionistic tense logics extending Fischer Servi's modal logic (introduced by Ewald in [15]), with modalities in pairs ⟐, ⊡, and ◆, ◘, for future and past along the accessibility relation. It soon becomes clear that the resulting semantics and metatheoretic results such as completeness come out *cleaner and simpler* for the tense logic than they do for the modal logic. We can often streamline arguments involving the box modality ⊡ by using its adjoint diamond ◆, which like ⟐, preserves open sets. Furthermore, with regard to applications of interest, the flexibility of having both forwards and backwards modalities is advantageous. For example, if $X \subseteq \mathbb{R}^n$ is equipped with the Euclidean topology, and $R \subseteq X \times X$ is the *reachability relation* of a continuous or hybrid dynamical system [2,3,11], then the formula ◆p denotes the set of states *reachable from* the p states, with p considered as a source or initial state set, while the forward modal diamond formula ⟐p denotes the set of states *from which* p states *can be reached*, here p denoting a target or goal state set. Under some standard regularity assumptions on the differential inclusions or equations, [2,3], the reachability relation R and its inverse will be l.s.c. (as

[3] Other work giving topological semantics for intuitionistic modal logics is [36], further investigated in [23]. This logic is properly weaker than Fischer Servi's as its intuitionistic diamond is not required to distribute over disjunction (hence is sub-normal). Both the bi-relational and topological semantics in [36] and the *relational spaces* in [23] have *no* conditions on the interaction of the intuitionistic and modal semantic structures, and the semantic clauses for both box and diamond require application of the interior operator to guarantee open sets.

[4] In the algebraic setting of Monteiro and Varsavsky's work [27] w.r.t. the logic MIPC, a special case of the l.s.c. property is anticipated: the lattice of open sets of a topological space is a complete Heyting algebra, and the structure yields a monadic Heyting algebra when the space is further equipped with an equivalence relation R with the property that the "R-saturation" or R-expansion of an open set is open.

well as reflexive and transitive), while further assumptions are required for the u.s.c. property (e.g. R is a closed set in the product topology on $X \times X$).

We continue on the theme of semi-continuity properties of relations in our second topic of investigation, namely that of *topological bisimulations* between topological Kripke models. A bisimulation notion for topological spaces $(X, \mathcal{T})$ has recently been developed by Aiello and van Benthem (e.g. [1], Def. 2.1). We show below that their forth and back topology-preserving conditions are equivalent to the lower semi-continuity of the inverse relation and of the relation, respectively. The first main result of the paper is that this notion of topological bisimulation yields the semantic preservation property w.r.t. topological Kripke models for both intuitionistic tense logics, and for their classical companion multi-modal logics in the setting of the Gödel translation.

In the last part of the paper, we give canonical topological Kripke models for the Hilbert-style axiomatizations of the Fischer Servi logics and their classical companions logics – over the set of prime theories of the intuitionistic logic and the set of ultrafilters of the companion classical logic, respectively, with topologies on the spaces that are neither Alexandrov nor Stone. We conclude the paper with the second main result: the syntactic Gödel translation induces a natural semantic map from the intuitionistic canonical model to a sub-model of the canonical model of the classical companion logic, and this map is itself a topological bisimulation.

2 Preliminaries from General Topology

We adopt the notation from set-valued analysis [2] in writing $r : X \rightsquigarrow Y$ to mean both that $r : X \to 2^Y$ is a *set-valued map*, with (possibly empty) set-values $r(x) \subseteq Y$ for each $x \in X$, and equivalently, that $r \subseteq X \times Y$ is a *relation*. The expressions $y \in r(x)$, $(x, y) \in r$ and $x\, r\, y$ are synonymous. For a map $r : X \rightsquigarrow Y$, the *inverse* $r^{-1} : Y \rightsquigarrow X$ given by: $x \in r^{-1}(y)$ iff $y \in r(x)$; the *domain* is $dom(r) := \{x \in X \mid r(x) \neq \varnothing\}$, and the *range* is $ran(r) := dom(r^{-1}) \subseteq Y$. A map $r : X \rightsquigarrow Y$ is *total on* X if $dom(r) = X$, and *surjective on* Y if $ran(r) = Y$. We write (as usual) $r : X \to Y$ to mean r is a *function*, i.e. a single-valued map total on X with values written $r(x) = y$ (rather than $r(x) = \{y\}$). For $r_1 : X \rightsquigarrow Y$ and $r_2 : Y \rightsquigarrow Z$, we write their relational composition as $r_1 \circ r_2 : X \rightsquigarrow Z$ given by $(r_1 \circ r_2)(x) := \{z \in Z \mid (\exists y \in Y)\, [(x, y) \in r_1 \wedge (y, z) \in r_2]\}$. Recall that $(r_1 \circ r_2)^{-1} = r_2^{-1} \circ r_1^{-1}$. A *pre-order* (*quasi-order*) is a reflexive and transitive binary relation, and a *partial-order* is a pre-order that is also anti-symmetric.

A relation $r : X \rightsquigarrow Y$ determines two *pre-image operators* (predicate transformers). The *existential* (or *lower*) pre-image is of type $r^{-\exists} : 2^Y \to 2^X$ and the *universal* (or *upper*) pre-image $r^{-\forall} : 2^Y \to 2^X$ is its dual w.r.t. set-complement:

$$\begin{aligned} r^{-\exists}(W) &:= \{x \in X \mid (\exists y \in Y)[\, (x, y) \in r \quad \wedge \quad y \in W]\} \\ &= \{x \in X \mid W \cap r(x) \neq \varnothing\} \\ r^{-\forall}(W) &:= X - r^{-\exists}(Y - W) \;=\; \{x \in X \mid r(x) \subseteq W\} \end{aligned}$$

for all $W \subseteq Y$. The operator $r^{-\exists}$ distributes over arbitrary unions, while $r^{-\forall}$ distributes over arbitrary intersections: $r^{-\exists}(\varnothing) = \varnothing$, $r^{-\exists}(Y) = dom(r)$, $r^{-\forall}(\varnothing) = X - dom(r)$, and $r^{-\forall}(Y) = X$. Note that when $r : X \to Y$ is a function, the pre-image operators reduce to the standard *inverse-image* operator; i.e. $r^{-\exists}(W) = r^{-\forall}(W) = r^{-1}(W)$. The pre-image operators respect relational inclusions: if $r_1 \subseteq r_2 \subseteq X \times Y$, then for all $W \subseteq Y$, we have $r_1^{-\exists}(W) \subseteq r_2^{-\exists}(W)$, but reversing to $r_2^{-\forall}(W) \subseteq r_1^{-\forall}(W)$. For the case of binary relations $r\colon X \rightsquigarrow X$ on a space X, the pre-images express in operator form the standard relational Kripke semantics for the (future) diamond and box modal operators determined by r. The operators on sets derived from the inverse relation r^{-1} are usually called the *post-image operators* $r^{\exists}$, $r^{\forall} : 2^X \to 2^Y$ defined by $r^{\exists} := (r^{-1})^{-\exists}$ and $r^{\forall} := (r^{-1})^{-\forall}$; these arise in the relational Kripke semantics for the *past* diamond and box operators in tense and temporal logics. The fundamental relationship between pre- and post-images is the *adjoint property*:

$$\forall W_1 \subseteq X,\ \forall W_2 \subseteq Y, \quad W_1 \subseteq r^{-\forall}(W_2) \quad \text{iff} \quad r^{\exists}(W_1) \subseteq W_2\,. \tag{3}$$

A *topology* $\mathcal{T} \subseteq 2^X$ on a set X is a family of subsets of X closed under arbitrary unions and finite intersections. The extreme cases are the *discrete* topology $\mathcal{T}_D = 2^X$, and the *trivial* (or *indiscrete*) topology $\mathcal{T}_\varnothing = \{\varnothing, X\}$. The *interior operator* $int_{\mathcal{T}} : 2^X \to 2^X$ determined by $\mathcal{T}$ is given by $int_{\mathcal{T}}(W) := \bigcup\{U \in \mathcal{T} \mid U \subseteq W\}$. Sets $W \in \mathcal{T}$ are called *open* w.r.t. $\mathcal{T}$, and this is so iff $W = int_{\mathcal{T}}(W)$. Let $-\mathcal{T} := \{W \subseteq X \mid (X - W) \in \mathcal{T}\}$ denote the dual lattice under set-complement. Sets $W \in -\mathcal{T}$ are called *closed* w.r.t. $\mathcal{T}$, and this is so iff $W = cl_{\mathcal{T}}(W)$, where the dual *closure operator* $cl_{\mathcal{T}} : 2^X \to 2^X$ is given by $cl_{\mathcal{T}}(W) := X - cl_{\mathcal{T}}(X - W)$, and the topological *boundary* is $bd_{\mathcal{T}}(W) := cl_{\mathcal{T}}(W) - int_{\mathcal{T}}(W)$. A family of open sets $\mathcal{B} \subseteq \mathcal{T}$ constitutes a *basis* for a topology $\mathcal{T}$ on X if every open set $W \in \mathcal{T}$ is a union of basic opens in $\mathcal{B}$, and for every $x \in X$ and every pair of basic opens $U_1, U_2 \in \mathcal{B}$ such that $x \in U_1 \cap U_2$, there exists $U_3 \in \mathcal{B}$ such that $x \in U_3 \subseteq (U_1 \cap U_2)$.

The purely topological notion of *continuity* for a function $f\colon X \to Y$ is that the inverse image $f^{-1}(U)$ is open whenever U is open. Analogous notions for relations/set-valued maps were introduced by Kuratowski and Bouligand in the 1920s. Given two topological spaces $(X, \mathcal{T})$ and $(Y, \mathcal{S})$, a map $R : X \rightsquigarrow Y$ is called: *lower semi-continuous* (l.s.c.) if for every $\mathcal{S}$-open set U in Y, $R^{-\exists}(U)$ is $\mathcal{T}$-open in X; *upper semi-continuous* (u.s.c.) if for every $\mathcal{S}$-open set U in Y, $R^{-\forall}(U)$ is $\mathcal{T}$-open in X; and (Vietoris) *continuous* if it is both l.s.c. and u.s.c. [2,7,24,33]. The u.s.c. condition is equivalent to $R^{-\exists}(V)$ is $\mathcal{T}$-closed in X whenever V is $\mathcal{S}$-closed in Y. Moreover, we have: $R : X \rightsquigarrow Y$ is l.s.c. iff $R^{-\exists}(int_{\mathcal{S}}(W)) \subseteq int_{\mathcal{T}}(R^{-\exists}(W))$ for all $W \subseteq Y$; and $R : X \rightsquigarrow Y$ is u.s.c. iff $R^{-\forall}(int_{\mathcal{S}}(W)) \subseteq int_{\mathcal{T}}(R^{-\forall}(W))$ for all $W \subseteq Y$ ([24], Vol. I, §18.I, p.173). The two semi-continuity properties reduce to the standard notion of continuity for functions $R : X \to Y$. Both semi-continuity properties are preserved under relational composition, and also under finite unions of relations.

We note the subclass of *Alexandrov topologies* because of their correspondence with Kripke relational semantics for classical S4 and intuitionistic logics. e.g. [1,26]. A topological space $(X, \mathcal{T})$ is called *Alexandrov* if for every $x \in X$, there is a *smallest* open set $U \in \mathcal{T}$ such that $x \in U$. In particular, every *finite* topology (i.e. only finitely many open sets) is Alexandrov. There is a one-to-one correspondence between pre-orders on X and Alexandrov topologies on X. Any pre-order $\preccurlyeq$ on X induces an Alexandrov topology $\mathcal{T}_{\preccurlyeq}$ by taking $int_{\mathcal{T}_{\preccurlyeq}}(W) := (\preccurlyeq)^{-\forall}(W)$, which means $U \in \mathcal{T}_{\preccurlyeq}$ iff U is upwards-$\preccurlyeq$-closed. In particular, $\mathcal{T}_{\preccurlyeq}$ is closed under arbitrary intersections as well as arbitrary unions, and $-\mathcal{T}_{\preccurlyeq} = \mathcal{T}_{\succcurlyeq}$. Conversely, for any topology, define a pre-order $\preccurlyeq_{\mathcal{T}}$ on X, known as the *specialisation pre-order*: $x \preccurlyeq_{\mathcal{T}} y$ iff $(\forall U \in \mathcal{T})\,[x \in U \Rightarrow y \in U]$. For any pre-order, $\preccurlyeq_{\mathcal{T}_{\preccurlyeq}} = \preccurlyeq$, and for any topology, $\mathcal{T}_{\preccurlyeq_{\mathcal{T}}} = \mathcal{T}$ iff $\mathcal{T}$ is Alexandrov (e.g. see [1], pp. 893-894). For further concepts in topology, see, e.g. [33].

3 Intuitionistic Modal and Tense Logics, and Their Classical Companion Logics: Syntax and Topological Semantics

Fix a countably infinite set AP of atomic propositions. The propositional language $\mathcal{L}_0$ is generated from $p \in AP$ using the connectives $\vee$, $\wedge$, $\rightarrow$ and the constant $\perp$. As usual, define further connectives: $\neg\,\varphi := \varphi \rightarrow \perp$ and $\varphi_1 \leftrightarrow \varphi_2 := (\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)$, and $\top := \perp \rightarrow \perp$. Let $\mathcal{L}_{0,\Box}$ be the mono-modal language extending $\mathcal{L}_0$ with the addition of the unary modal operator $\Box$. A further modal operator $\Diamond$ can be defined as the classical dual: $\Diamond\varphi := \neg\Box\neg\varphi$.

For the intuitionistic modal and tense languages, let $\mathcal{L}^{\mathbf{m}}$ ($\mathcal{L}^{\mathbf{t}}$) be the modal (tense) language extending $\mathcal{L}_0$ with the addition of two (four) modal operators $\diamonddot$ and $\boxdot$ (and $\blacklozenge$ and $\blacksquare$) , generated by the grammar:

$$\varphi \;::=\; p \mid \perp \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \diamonddot\varphi \mid \boxdot\varphi \quad (\mid \blacklozenge\varphi \mid \blacksquare\varphi\,)$$

for $p \in AP$. Likewise, for the classical topological modal and tense logics, let $\mathcal{L}^{\mathbf{m}}_{\Box}$ ($\mathcal{L}^{\mathbf{t}}_{\Box}$) be the modal (tense) language extending $\mathcal{L}_{0,\Box}$ with the addition of $\diamonddot$ and $\boxdot$ (and $\blacklozenge$ and $\blacksquare$).

The original Gödel translation [21], as a function $\mathrm{GT} : \mathcal{L}_0 \rightarrow \mathcal{L}_{0,\Box}$, simply prefixes $\Box$ to *every* subformula of a propositional formula. Reading the S4 $\Box$ as topological interior, this means we force every propositional formula to intuitionistically denote an open set. In Fischer Servi's extension of the Gödel translation [18,16], the clauses for the propositional fragment are from a variant translation used by Fitting [19], who shows it to be equivalent to Gödel's original ([19], Ch. 9, # 20). Define the function $\mathrm{GT} : \mathcal{L}^{\mathbf{t}} \rightarrow \mathcal{L}^{\mathbf{t}}_{\Box}$ by induction on formulas as follows:

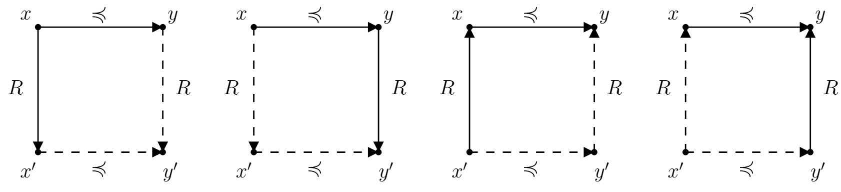
$$\begin{array}{rlrl}
\mathrm{GT}(p) := & \!\!\Box p \;\text{ for } p \in AP & & \\
\mathrm{GT}(\varphi_1 \rightarrow \varphi_2) := & \!\!\Box\,(\mathrm{GT}(\varphi_1) \rightarrow \mathrm{GT}(\varphi_2)) & \mathrm{GT}(\perp) := & \!\!\perp \\
\mathrm{GT}(\varphi_1 \vee \varphi_2) := & \!\!\mathrm{GT}(\varphi_1) \vee \mathrm{GT}(\varphi_2) & \mathrm{GT}(\varphi_1 \wedge \varphi_2) := & \!\!\mathrm{GT}(\varphi_1) \wedge \mathrm{GT}(\varphi_2) \\
\mathrm{GT}(\,\diamonddot\varphi\,) := & \!\!\diamonddot\,\mathrm{GT}(\varphi) & \mathrm{GT}(\,\blacklozenge\varphi\,) := & \!\!\blacklozenge\,\mathrm{GT}(\varphi) \\
\mathrm{GT}(\,\boxdot\varphi\,) := & \!\!\Box\boxdot\,\mathrm{GT}(\varphi) & \mathrm{GT}(\,\blacksquare\varphi\,) := & \!\!\Box\blacksquare\,\mathrm{GT}(\varphi)
\end{array}$$

In topological terms, the only clauses in the translation where it is essential to have an explicit $\Box$ to guarantee openness of denotation sets are for atomic propositions, for implication $\rightarrow$, and for the box modalties $\boxdot$ and $\blacksquare$. There is no such need in the clauses for $\vee$ and $\wedge$ because finite unions and finite intersections of open sets are open. For the diamond modalties, the semi-continuity conditions that R and its inverse R^{-1} are both l.s.c. ensure that the semantic operators $R^{-\exists}$ and $R^{\exists}$ interpretting $\diamondsuit$ and $\blacklozenge$ must preserve open sets. We now explain this generalization, which was first presented in [12].

The bi-relational semantics of Fischer Servi [16,17], and Plotkin and Stirling [29,32] are over Kripke frames $\mathcal{F} = (X, \preccurlyeq, R)$, where $\preccurlyeq$ is a pre-order on X and $R : X \rightsquigarrow X$ is the modal accessibility relation. Using the induced Alexandrov topology $\mathcal{T}_{\preccurlyeq}$, a bi-relational Kripke frame $\mathcal{F}$ is equivalent to the topological frame $(X, \mathcal{T}_{\preccurlyeq}, R)$. A set is open in $\mathcal{T}_{\preccurlyeq}$ exactly when it is $\preccurlyeq$-*persistent* or upward-$\preccurlyeq$-closed. The four bi-relational conditions identified in [29], and also familiar as the forth ("Zig") and back ("Zag") conditions for *bisimulations* (e.g. [6], Ch. 2), can be cleanly transcribed as *semi-continuity conditions* on the relations $R : X \rightsquigarrow X$ and $R^{-1} : X \rightsquigarrow X$ with respect to the topology $\mathcal{T}_{\preccurlyeq}$.

Definition 1. *Let $\mathcal{F} = (X, \preccurlyeq, R)$ be a bi-relational frame. Four conditions expressing interaction between $\preccurlyeq$ and R are identified as follows:*

$$\begin{array}{llllllll}
\mathbf{Zig}(\preccurlyeq, R): & \textit{if} & x \preccurlyeq y & \textit{and} & x\,R\,x' & \textit{then}\ (\exists y' \in X) & [\ y\,R\,y' & \textit{and}\quad x' \preccurlyeq y'\ ] \\
\mathbf{Zag}(\preccurlyeq, R): & \textit{if} & x \preccurlyeq y & \textit{and} & y\,R\,y' & \textit{then}\ (\exists x' \in X) & [\ x\,R\,x' & \textit{and}\quad x' \preccurlyeq y'\ ] \\
\mathbf{Zig}(\preccurlyeq, R^{-1}): & \textit{if} & x \preccurlyeq y & \textit{and} & x'\,R\,x & \textit{then}\ (\exists y' \in X) & [\ y'\,R\,y & \textit{and}\quad x' \preccurlyeq y'\ ] \\
\mathbf{Zag}(\preccurlyeq, R^{-1}): & \textit{if} & x \preccurlyeq y & \textit{and} & y'\,R\,y & \textit{then}\ (\exists x' \in X) & [\ x'\,R\,x & \textit{and}\quad x' \preccurlyeq y'\ ]
\end{array}$$



$$\begin{tikzcd}
x \arrow[r, "\preccurlyeq"] \arrow[d, "R"'] & y \arrow[d, dashed, "R"] \\
x' \arrow[r, dashed, "\preccurlyeq"'] & y'
\end{tikzcd}
\qquad
\begin{tikzcd}
x \arrow[r, "\preccurlyeq"] \arrow[d, dashed, "R"'] & y \arrow[d, "R"] \\
x' \arrow[r, dashed, "\preccurlyeq"'] & y'
\end{tikzcd}
\qquad
\begin{tikzcd}
x \arrow[r, "\preccurlyeq"] & y \\
x' \arrow[u, "R"] \arrow[r, dashed, "\preccurlyeq"'] & y' \arrow[u, dashed, "R"']
\end{tikzcd}
\qquad
\begin{tikzcd}
x \arrow[r, "\preccurlyeq"] & y \\
x' \arrow[u, dashed, "R"] \arrow[r, dashed, "\preccurlyeq"'] & y' \arrow[u, "R"']
\end{tikzcd}$$

$\mathbf{Zig}(\preccurlyeq, R)$ $\quad$ $\mathbf{Zag}(\preccurlyeq, R)$ $\quad$ $\mathbf{Zig}(\preccurlyeq, R^{-1})$ $\quad$ $\mathbf{Zag}(\preccurlyeq, R^{-1})$

From earlier work [9], we know these bi-relational conditions correspond to semi-continuity properties of R with respect to the Alexandrov topology $\mathcal{T}_{\preccurlyeq}$.

Proposition 1. ([9]) *Let $\mathcal{F} = (X, \preccurlyeq, R)$ be a bi-relational frame, with $\mathcal{T}_{\preccurlyeq}$ it induced topology. The conditions in each row below are equivalent.*

1.	$\mathbf{Zig}(\preccurlyeq, R)$	$(R^{-1} \circ \preccurlyeq) \subseteq (\preccurlyeq \circ R^{-1})$	R *is* l.s.c. *in* $\mathcal{T}_{\preccurlyeq}$
2.	$\mathbf{Zag}(\preccurlyeq, R)$	$(\preccurlyeq \circ R) \subseteq (R \circ \preccurlyeq)$	R *is* u.s.c. *in* $\mathcal{T}_{\preccurlyeq}$
3.	$\mathbf{Zig}(\preccurlyeq, R^{-1})$	$(R \circ \preccurlyeq) \subseteq (\preccurlyeq \circ R)$	R^{-1} *is* l.s.c. *in* $\mathcal{T}_{\preccurlyeq}$
4.	$\mathbf{Zag}(\preccurlyeq, R^{-1})$	$(\preccurlyeq \circ R^{-1}) \subseteq (R^{-1} \circ \preccurlyeq)$	R^{-1} *is* u.s.c. *in* $\mathcal{T}_{\preccurlyeq}$

The Fischer Servi interaction conditions between the intuitionistic and modal relations, introduced in [17] and used in [15,18,22,29,32], are the first and third bi-relational conditions $\mathbf{Zig}(\preccurlyeq, R)$ and $\mathbf{Zig}(\preccurlyeq, R^{-1})$. In Kripke frames meeting these conditions, one can give semantic clauses for the diamond and box that are natural under the intuitionistic reading of the restricted $\exists$ and $\forall$ quantification with respect to R-successors. More precisely, the resulting logic is faithfully embedded into intuitionistic first-order logic by the standard modal to first-order translation, and a natural extension of the Gödel translation faithfully embeds it into the classical bi-modal logic combining $\mathbf{S4}\Box$ with $\mathbf{K}$ or extensions.

Since the Fischer Servi interaction conditions for the *forward* or *future* modal operators $\Diamonddot$ and $\boxdot$ for R require the same l.s.c. property of both R and R^{-1}, this means that, *at no extra cost* in semantic assumptions, we can add on the *backward* or *past* modal operators $\blacklozenge$ and $\blacksquare$ for R^{-1}, and obtain the desired interaction condition for R^{-1} *for free.*

Definition 2. *A* topological frame *is a structure* $\mathcal{F} = (X, \mathcal{T}, R)$ *where* $(X, \mathcal{T})$ *is a topological space and* $R : X \rightsquigarrow X$ *is a binary relation.* $\mathcal{F}$ *is an* l.s.c. topological frame *if both* R *and* R^{-1} *are l.s.c. in* $\mathcal{T}$*. A* model *over* $\mathcal{F}$ *is a structure* $\mathcal{M} = (\mathcal{F}, v)$ *where* $v : AP \rightsquigarrow X$ *is an atomic valuation relation. A model* $\mathcal{M}$ *is an* open model *if for each* $p \in AP$*, the denotation set* $v(p)$ *is open in* $\mathcal{T}$*. For open models* $\mathcal{M}$ *over l.s.c. frames* $\mathcal{F}$*, the intuitionistic denotation map* $\llbracket\cdot\rrbracket^{\mathcal{M}}_{\mathbf{I}} : \mathcal{L}^{\mathbf{t}} \rightsquigarrow X$ *(or* $\llbracket\cdot\rrbracket^{\mathcal{M}}_{\mathbf{I}} : \mathcal{L}^{\mathbf{m}} \rightsquigarrow X$*) is defined by:*

$$\begin{array}{rlrl}
\llbracket p\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := v(p) \quad \textit{for } p \in AP & \llbracket\bot\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := \varnothing \\
\llbracket\varphi_1 \to \varphi_2\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := \mathit{int}_{\mathcal{T}}((X - \llbracket\varphi_1\rrbracket^{\mathcal{M}}_{\mathbf{I}}) \cup \llbracket\varphi_2\rrbracket^{\mathcal{M}}_{\mathbf{I}}) & & \\
\llbracket\varphi_1 \vee \varphi_2\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := \llbracket\varphi_1\rrbracket^{\mathcal{M}}_{\mathbf{I}} \cup \llbracket\varphi_2\rrbracket^{\mathcal{M}}_{\mathbf{I}} & \llbracket\varphi_1 \wedge \varphi_2\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := \llbracket\varphi_1\rrbracket^{\mathcal{M}}_{\mathbf{I}} \cap \llbracket\varphi_2\rrbracket^{\mathcal{M}}_{\mathbf{I}} \\
\llbracket\Diamonddot\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := R^{-\exists}(\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}}) & \llbracket\boxdot\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := \mathit{int}_{\mathcal{T}}\big(R^{-\forall}(\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}})\big) \\
\llbracket\blacklozenge\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := R^{\exists}(\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}}) & \llbracket\blacksquare\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} & := \mathit{int}_{\mathcal{T}}\big(R^{\forall}(\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}})\big)
\end{array}$$

A formula $\varphi \in \mathcal{L}^{\mathbf{t}}$ *(or* $\varphi \in \mathcal{L}^{\mathbf{m}}$*) is* int-modal-top valid *in an open model* $\mathcal{M}$*, written* $\mathcal{M} \Vdash \varphi$*, if* $\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} = X$*, and is* int-modal-top valid *in an l.s.c. frame* $\mathcal{F} = (X, \mathcal{T}, R)$*, written* $\mathcal{F} \Vdash \varphi$*, if* $\mathcal{M} \Vdash \varphi$ *for all open models* $\mathcal{M}$ *over* $\mathcal{F}$*. Formula* φ *is* satisfiable *in* $\mathcal{M}$ *if* $\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} \neq \varnothing$*, and* φ *is* falsifiable *in* $\mathcal{M}$ *if* $\llbracket\varphi\rrbracket^{\mathcal{M}}_{\mathbf{I}} \neq X$*. Let* $\mathbb{IK}^{\mathbf{t}}\mathbb{T}$ *(*$\mathbb{IK}^{\mathbf{m}}\mathbb{T}$*) be the set of all* $\varphi \in \mathcal{L}^{\mathbf{t}}$ *(*$\varphi \in \mathcal{L}^{\mathbf{m}}$*) such that* $\mathcal{F} \Vdash \varphi$ *in every l.s.c. topological frame* $\mathcal{F}$*.*

The property that every denotation set $\llbracket\,\varphi\,\rrbracket^{\mathcal{M}}_{\mathbf{I}}$ is open in $\mathcal{T}$ follows immediately from the openness condition on $v(p)$, the l.s.c. properties of $R^{-\exists}$ and $R^{\exists}$, and the extra interior operation in the semantics for $\to$, $\boxdot$ and $\blacksquare$.

Definition 3. *For the tense (modal) language* $\mathcal{L}^{\mathbf{t}}_{\Box}$ *(and* $\mathcal{L}^{\mathbf{m}}_{\Box}$ *), we define the classical denotation map* $\llbracket\cdot\rrbracket^{\mathcal{M}} : \mathcal{L}^{\mathbf{t}}_{\Box} \rightsquigarrow X$ *(* $\llbracket\cdot\rrbracket^{\mathcal{M}} : \mathcal{L}^{\mathbf{m}}_{\Box} \rightsquigarrow X$ *) with respect to arbitrary topological models* $\mathcal{M} = (X, \mathcal{T}, R, v)$*, where* $v : AP \rightsquigarrow X$ *is unrestricted. The map* $\llbracket\cdot\rrbracket^{\mathcal{M}}$ *is defined the same way as* $\llbracket\cdot\rrbracket^{\mathcal{M}}_{\mathbf{I}}$ *for atomic* $p \in AP$*,* $\bot$*,* $\vee$*,* $\wedge$*,* $\Diamonddot$ *and* $\blacklozenge$*, but differs on the following clauses:*

$$\begin{array}{rlrl}
\llbracket\varphi_1 \to \varphi_2\rrbracket^{\mathcal{M}} & := (X - \llbracket\varphi_1\rrbracket^{\mathcal{M}}) \cup \llbracket\varphi_2\rrbracket^{\mathcal{M}} & \llbracket\,\Box\varphi\,\rrbracket^{\mathcal{M}} & := \mathit{int}_{\mathcal{T}}\,(\llbracket\,\varphi\,\rrbracket^{\mathcal{M}}) \\
\llbracket\,\boxdot\varphi\rrbracket^{\mathcal{M}} & := R^{-\forall}\,(\llbracket\varphi\rrbracket^{\mathcal{M}}) & \llbracket\,\blacksquare\varphi\rrbracket^{\mathcal{M}} & := R^{\forall}\,(\llbracket\varphi\rrbracket^{\mathcal{M}})
\end{array}$$

A formula $\varphi \in \mathcal{L}^{\mathbf{t}}_{\Box}$ *(or* $\varphi \in \mathcal{L}^{\mathbf{m}}_{\Box}$*) is* modal-top valid *in* $\mathcal{M}$*, written* $\mathcal{M} \models \varphi$*, if* $\llbracket\varphi\rrbracket^{\mathcal{M}} = X$*, and is* modal-top valid *in a topological frame* $\mathcal{F} = (X, \mathcal{T}, R)$*, written*

$\mathcal{F} \models \varphi$, if $\mathcal{M} \models \varphi$ for all models $\mathcal{M}$ over $\mathcal{F}$. Let $\mathbb{K}^{\mathbf{t}}\mathbb{T}$ ($\mathbb{K}^{\mathbf{m}}\mathbb{T}$) be the set of all $\varphi \in \mathcal{L}^{\mathbf{t}}_{\Box}$ ($\varphi \in \mathcal{L}^{\mathbf{m}}_{\Box}$) such that $\mathcal{F} \models \varphi$ for every topological frame $\mathcal{F}$. Let $\mathbb{K}^{\mathbf{t}}\mathbb{LSC}$ ($\mathbb{K}^{\mathbf{m}}\mathbb{LSC}$) be the set of all $\varphi \in \mathcal{L}^{\mathbf{t}}_{\Box}$ ($\varphi \in \mathcal{L}^{\mathbf{m}}_{\Box}$) such that $\mathcal{F} \models \varphi$ in every l.s.c. topological frame $\mathcal{F}$.

For Fischer Servi's extension of Gödel's translation, Definitions 2 and 3 imply that for any model $\mathcal{M} = (\mathcal{F}, v)$ over an l.s.c. topological frame $\mathcal{F}$, if $\mathcal{M}' = (\mathcal{F}, v')$ is the variant open model with $v'(p) := int_{\mathcal{T}}(v(p))$, then $\forall \varphi \in \mathcal{L}^{\mathbf{t}}$:

$$[\![\varphi]\!]^{\mathcal{M}'}_{\mathbf{I}} = [\![\, \mathrm{GT}(\varphi) \,]\!]^{\mathcal{M}} = [\![\, \Box\mathrm{GT}(\varphi) \,]\!]^{\mathcal{M}}. \tag{4}$$

Consequently, we have semantic faithfulness, as well as the openness property: for all $\varphi \in \mathcal{L}^{\mathbf{t}}$, the formula $\mathrm{GT}(\varphi) \leftrightarrow \Box\mathrm{GT}(\varphi)$ is in $\mathbb{K}^{\mathbf{t}}\mathbb{LSC}$.

Proposition 2. [Extended Gödel translation: semantic faithfulness]
For all $\varphi \in \mathcal{L}^{\mathbf{t}}$, $\varphi \in \mathbb{IK}^{\mathbf{t}}\mathbb{T}$ iff $\mathrm{GT}(\varphi) \in \mathbb{K}^{\mathbf{t}}\mathbb{LSC}$.

The semi-continuity conditions can be cleanly characterized in the companion classical multi-modal logics, as given in [13].

Proposition 3. [[13] Modal characterization of semi-continuity conditions]
Let $\mathcal{F} = (X, \mathcal{T}, R)$ be a topological frame and let $p \in AP$. In the following table, the conditions listed across each row are equivalent.

(1.)	*R is l.s.c. in $\mathcal{T}$*	$\mathcal{F} \models \Diamonddot\Box p \to \Box\Diamonddot p$	$\mathcal{F} \vDash \Diamonddot\Box p \leftrightarrow \Box\Diamonddot\Box p$
(2.)	*R is u.s.c. in $\mathcal{T}$*	$\mathcal{F} \models \boxdot\Box p \to \Box\boxdot p$	
(3.)	*R^{-1} is l.s.c. in $\mathcal{T}$*	$\mathcal{F} \models \Box\boxdot p \to \boxdot\Box p$	$\mathcal{F} \vDash \blacklozenge\Box p \leftrightarrow \Box\blacklozenge\Box p$
(4.)	*R^{-1} is u.s.c. in $\mathcal{T}$*	$\mathcal{F} \models \Box\Diamonddot p \to \Diamonddot\Box p$	

4 Topological Bisimulations

Aiello and van Benthem's notions of topological simulation and bisimulation between classical S4 topological models are as follows.

Definition 4. [[1], Definition 2.1] *Let $(X_1, \mathcal{T}_1)$ and $(X_2, \mathcal{T}_2)$ be two topological spaces, let $v_1 : AP \leadsto X_1$ and $v_2 : AP \leadsto X_2$ be valuations of atomic propositions, and let $\mathcal{M}_1 = (X_1, \mathcal{T}_1, v_1)$ and $\mathcal{M}_2 = (X_2, \mathcal{T}_2, v_2)$ be topological models.*
A relation $B : X_1 \leadsto X_2$ is a topo-bisimulation *between $\mathcal{M}_1$ and $\mathcal{M}_2$ if*
(i.a) $\forall x \in X_1,\ \forall y \in X_2,\ \forall p \in AP,$ if $x\,B\,y$ and $x \in v_1(p)$ then $y \in v_2(p)$;
(i.b) $\forall x \in X_1,\ \forall y \in X_2,\ \forall p \in AP,$ if $x\,B\,y$ and $y \in v_2(p)$ then $x \in v_1(p)$;
(ii.a) $\forall x \in X_1,\ \forall y \in X_2,\ \forall U \in \mathcal{T}_1,$ if $x\,B\,y$ and $x \in U$
then $\exists V \in \mathcal{T}_2$ with $y \in V$ and $\forall y' \in V,\ \exists x' \in U$ such that $x'\,B\,y'$;
(ii.b) $\forall x \in X_1,\ \forall y \in X_2,\ \forall V \in \mathcal{T}_2,$ if $x\,B\,y$ and $y \in V$
then $\exists U \in \mathcal{T}_1$ with $x \in U$ and $\forall x' \in U,\ \exists y' \in V$ such that $x'\,B\,y'$.
If only conditions (i.a) *and* (ii.a) *hold of a relation $B : X_1 \leadsto X_2$, then B is called a* topo-simulation *of $\mathcal{M}_1$ by $\mathcal{M}_2$.*

Proposition 4. *Given a map $B : X_1 \rightsquigarrow X_2$ between $(X_1, \mathcal{T}_1)$ and $(X_2, \mathcal{T}_2)$,*
(1.) *B satisfies condition* (ii.a) *of Definition 4 iff B^{-1} is* l.s.c.*;*
(2.) *B satisfies condition* (ii.b) *of Definition 4 iff B is* l.s.c..

Proof. By rewriting in terms of the pre- and post-image set operators, it is easy to show that conditions (ii.a) and (ii.b) are equivalent to the following:

$$\begin{array}{ll} (\text{ii.a}^\sharp) & \forall U \in \mathcal{T}_1, \quad B^{\exists}(U) \subseteq int_{\mathcal{T}_2}\big(B^{\exists}(U)\big) \\ (\text{ii.b}^\sharp) & \forall V \in \mathcal{T}_2, \ B^{-\exists}(V) \subseteq int_{\mathcal{T}_1}\big(B^{-\exists}(V)\big) \end{array}$$

Clearly, (ii.a$^\sharp$) says that $B^{\exists}(U)$ is open in X_2 whenever U open in X_1, while (ii.b$^\sharp$) says that $B^{-\exists}(V)$ is open in X_1 whenever V open in X_2. ⊣

For the appropriate notion of topological bisimulation between topological Kripke models for the intuitionistic and classical companion modal and tense logics under study here, we need to put together the topology-preserving conditions (ii.a) and (ii.b) above with the standard clauses for preservation of the modal/tense semantic structure.

Definition 5. *Let $\mathcal{M}_1 = (X_1, \mathcal{T}_1, R_1, v_1)$ and $\mathcal{M}_2 = (X_2, \mathcal{T}_2, R_2, v_2)$ be two topological models. A map $B : X_1 \rightsquigarrow X_2$ will be called a* tense topo-bisimulation *between $\mathcal{M}_1$ and $\mathcal{M}_2$ if for all atomic $p \in AP$:*

$$\begin{array}{llll} (\text{i.a}) & B^{\exists}(v_1(p)) \subseteq v_2(p) & (\text{i.b}) & B^{-\exists}(v_2(p)) \subseteq v_1(p) \\ (\text{ii.a}) & B^{-1} : X_2 \rightsquigarrow X_1 \textit{ is l.s.c.} & (\text{ii.b}) & B : X_1 \rightsquigarrow X_2 \textit{ is l.s.c.} \\ (\text{iii.a}) & (B^{-1} \circ R_1) \subseteq (R_2 \circ B^{-1}) & (\text{iii.b}) & (B \circ R_2) \subseteq (R_1 \circ B) \\ (\text{iv.a}) & (B^{-1} \circ R_1^{-1}) \subseteq (R_2^{-1} \circ B^{-1}) & (\text{iv.b}) & (B \circ R_2^{-1}) \subseteq (R_1^{-1} \circ B) \end{array}$$

If only conditions (i.a), (ii.a) *and* (iii.a) *hold of the map $B : X_1 \rightsquigarrow X_2$, then B is called a* modal topo-simulation *of $\mathcal{M}_1$ by $\mathcal{M}_2$; if all but conditions* (iv.a) *and* (iv.b) *hold, then B is a* modal topo-bisimulation *between $\mathcal{M}_1$ and $\mathcal{M}_2$.*

Combining all the conditions (iii) and (iv), one obtains two equalities: $(R_1 \circ B) = (B \circ R_2)$ and $(R_2 \circ B^{-1}) = (B^{-1} \circ R_1)$. The set-operator form of the semantic preservation conditions are:

$$B^{\exists}(\llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_1}) \subseteq \llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_2} \quad \text{and} \quad B^{-\exists}(\llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_2}) \subseteq \llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_1} \tag{5}$$

and likewise for classical denotation maps $\llbracket \varphi \rrbracket^{\mathcal{M}_i}$. We will also use the dual versions under the adjoint equivalence (3). These are:

$$\llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_1} \subseteq B^{-\forall}(\llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_2}) \quad \text{and} \quad \llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_2} \subseteq B^{\forall}(\llbracket \varphi \rrbracket_{\mathbf{I}}^{\mathcal{M}_1}) \tag{6}$$

and likewise for $\llbracket \varphi \rrbracket^{\mathcal{M}_i}$. Note also that $B^{-1} : X_2 \rightsquigarrow X_1$ being l.s.c. has a further equivalent characterization: $int_{\mathcal{T}_1}(B^{-\forall}(W)) \subseteq B^{-\forall}(int_{\mathcal{T}_2}(W))$, for all $W \subseteq X_2$; this is a generalization of the characterization for binary relations on a single space X that is formalized in Proposition 3, Row (*3.*).

What we discover is that *exactly the same* notion of a bisimulation between models yields the same semantic preservation property for *both* the intuitionistic and the classical semantics. Otherwise put, the specifically *topological* requirement that the operators $B^{\exists}$ and $B^{-\exists}$ preserve open sets is enough to push through the result for intuitionistic modal and tense logics.

Theorem 1. [Semantic preservation for tense topo-bisimulations] *Let* $\mathcal{M}_1 = (X_1, \mathcal{T}_1, R_1, v_1)$ *and* $\mathcal{M}_2 = (X_2, \mathcal{T}_2, R_2, v_2)$ *be any two topological models, and let* $B : X_1 \rightsquigarrow X_2$ *be a tense topo-bisimulation between* $\mathcal{M}_1$ *and* $\mathcal{M}_2$*.*

(1.) *If* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *are open and l.s.c., then for all* $x \in X_1$ *and* $y \in X_2$*:*

$$x\,B\,y \qquad \textit{implies} \qquad (\forall \varphi \in \mathcal{L}^{\mathbf{t}})\,[\; x \in [\![\, \varphi \,]\!]^{\mathcal{M}_1}_{\mathbf{I}} \;\Leftrightarrow\; y \in [\![\, \varphi \,]\!]^{\mathcal{M}_2}_{\mathbf{I}} \;]$$

(2.) *For all* $x \in X_1$ *and* $y \in X_2$*:*

$$x\,B\,y \qquad \textit{implies} \qquad (\forall \varphi \in \mathcal{L}^{\mathbf{t}}_{\Box})\,[\; x \in [\![\, \varphi \,]\!]^{\mathcal{M}_1} \;\Leftrightarrow\; y \in [\![\, \varphi \,]\!]^{\mathcal{M}_2} \;]$$

Proof. The proof proceeds as usual, by induction on the structure of formulas, to establish the two inclusions displayed in (5), or their analogs for the classical denotation maps. The base case for atomic propositions is given by conditions (i.a) and (i.b). For the classical semantics in Part (2.), the argument is completely standard for the propositional and modal/tense operators, and the case for topological $\Box$ is made in [1]. For the intuitionistic semantics in Part (1.), we give the cases for implication $\rightarrow$ and for box $\boxdot$. Assume the result holds for φ_1 and φ_2 in $\mathcal{L}^{\mathbf{t}}$. In particular, from Assertions (5) and (6), we have:

$(X_1 - [\![\varphi_1]\!]^{\mathcal{M}_1}_{\mathbf{I}}) \subseteq (X_1 - B^{-\exists}([\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}}))$, and $[\![\varphi_2]\!]^{\mathcal{M}_1}_{\mathbf{I}} \subseteq B^{-\forall}([\![\,\varphi_2\,]\!]^{\mathcal{M}_2}_{\mathbf{I}})$. Now:

$$\begin{array}{ll}
\quad B^{\exists}\,([\![\,\varphi_1 \rightarrow \varphi_2\,]\!]^{\mathcal{M}_1}_{\mathbf{I}}) & \\
= B^{\exists}\,(int_{\mathcal{T}_1}\,(\,(\,X_1 - [\![\varphi_1]\!]^{\mathcal{M}_1}_{\mathbf{I}}\,)\,\cup\,[\![\varphi_2]\!]^{\mathcal{M}_1}_{\mathbf{I}})) & \\
\subseteq B^{\exists}\left(int_{\mathcal{T}_1}\left(\,X_1 - B^{-\exists}([\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\cup\,B^{-\forall}([\![\varphi_2]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\right)\right) & \text{by induction hypothesis} \\
= B^{\exists}\left(int_{\mathcal{T}_1}\left(\,B^{-\forall}(X_2 - [\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\cup\,B^{-\forall}([\![\varphi_2]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\right)\right) & \text{by duality } B^{-\forall}\,/\,B^{-\exists} \\
\subseteq int_{\mathcal{T}_2}\left(B^{\exists}\left(\,B^{-\forall}(X_2 - [\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\cup\,B^{-\forall}([\![\varphi_2]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\right)\right) & \text{by } B^{-1} \text{ being l.s.c.} \\
\subseteq int_{\mathcal{T}_2}\left(B^{\exists}\left(B^{-\forall}\,(\,(X_2 - [\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}})\,\cup\,[\![\varphi_2]\!]^{\mathcal{M}_2}_{\mathbf{I}}\,)\right)\right) & \text{by monotonicity of } B^{-\forall} \\
\subseteq int_{\mathcal{T}_2}\,(\,(\,X_2 - [\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}}\,)\,\cup\,[\![\varphi_2]\!]^{\mathcal{M}_2}_{\mathbf{I}}\,) & \text{by adjoint property} \\
= [\![\,\varphi_1 \rightarrow \varphi_2\,]\!]^{\mathcal{M}_2}_{\mathbf{I}} &
\end{array}$$

Verifying that $B^{-\exists}\,([\![\,\varphi_1 \rightarrow \varphi_2\,]\!]^{\mathcal{M}_2}_{\mathbf{I}}) \subseteq [\![\,\varphi_1 \rightarrow \varphi_2\,]\!]^{\mathcal{M}_1}_{\mathbf{I}}$ proceeds similarly, using from the induction hypothesis: $(X_2 - [\![\varphi_1]\!]^{\mathcal{M}_2}_{\mathbf{I}}) \subseteq (X_2 - B^{\exists}([\![\varphi_1]\!]^{\mathcal{M}_1}_{\mathbf{I}}))$, and $[\![\varphi_2]\!]^{\mathcal{M}_2}_{\mathbf{I}} \subseteq B^{\forall}([\![\,\varphi_2\,]\!]^{\mathcal{M}_1}_{\mathbf{I}})$.
For the $\boxdot$ case:

$$\begin{array}{ll}
\quad [\![\,\boxdot\varphi\,]\!]^{\mathcal{M}_1}_{\mathbf{I}} & \\
= int_{\mathcal{T}_1}\left(R_1^{-\forall}\,([\![\,\varphi\,]\!]^{\mathcal{M}_1}_{\mathbf{I}})\right) & \\
\subseteq int_{\mathcal{T}_1}\left(R_1^{-\forall}\left(B^{-\forall}\,([\![\,\varphi\,]\!]^{\mathcal{M}_2}_{\mathbf{I}})\right)\right) & \text{by induction hypothesis} \\
\subseteq int_{\mathcal{T}_1}\left(B^{-\forall}\left(R_2^{-\forall}\,([\![\,\varphi\,]\!]^{\mathcal{M}_2}_{\mathbf{I}})\right)\right) & \text{since } R_1 \circ B = B \circ R_2 \\
\subseteq B^{-\forall}\left(int_{\mathcal{T}_2}\left(R_2^{-\forall}\,([\![\,\varphi\,]\!]^{\mathcal{M}_2}_{\mathbf{I}})\right)\right) & \text{by } B^{-1} \text{ being l.s.c. (dual } B^{-\forall} \text{ form)} \\
= B^{-\forall}\,([\![\,\varphi\,]\!]^{\mathcal{M}_2}_{\mathbf{I}}) &
\end{array}$$

The argument for $\blacksquare$ symmetrically appeals to B being l.s.c. (dual $B^{\forall}$ form). ⊣

In a sequel paper, [10], we give a partial converse (Hennessy-Milner type result) by proving that a certain class of open l.s.c. models has the property that for any two models $\mathcal{M}_1$ and $\mathcal{M}_2$ in the class, there is a total and surjective tense topo-bisimulation B between them that maximally preserves the intuitionistic semantics, in the sense that for all $x \in X_1$ and $y \in X_2$:

$$x\,B\,y \qquad \text{iff} \qquad (\forall \varphi \in \mathcal{L}^{\mathbf{t}})\,[\;x \in [\![\varphi]\!]_{\mathbf{I}}^{\mathcal{M}_1} \;\Leftrightarrow\; y \in [\![\varphi]\!]_{\mathbf{I}}^{\mathcal{M}_2}\;]\,.$$

5 Axiomatizations and Canonical Models

Let $\mathbf{IPC} \subseteq \mathcal{L}_0$ be the set of intuitionistic propositional theorems, and abusing notation, let **IPC** also denote a standard axiomatisation for that logic. Likewise, let $\mathbf{S4}\Box \subseteq \mathcal{L}_{0,\Box}$ be the set of theorems of classical S4, and let $\mathbf{S4}\Box$ also denote any standard axiomatisation of classical S4. To be concrete, let $\mathbf{S4}\Box$ contain all instances of classical propositional tautologies in the language $\mathcal{L}_{0,\Box}$, and the axiom schemes:

$$\mathbf{N}\Box:\ \Box\top \qquad\qquad \mathbf{T}\Box:\ \Box\varphi \to \varphi$$

$$\mathbf{R}\Box:\ \Box(\varphi_1 \wedge \varphi_2) \leftrightarrow \Box\varphi_1 \wedge \Box\varphi_2 \qquad\qquad \mathbf{4}\Box:\ \Box\varphi \to \Box\Box\varphi$$

and be closed under the inference rules of *modus ponens* (**MP**), *uniform substitution* (**Subst**) (of formulas for atomic propositions), and $\Box$-monotonicity ($\mathbf{Mono}\Box$): from $\varphi_1 \to \varphi_2$ infer $\Box\varphi_1 \to \Box\varphi_2$.

On notation, for any axiomatically presented logic Λ in a language $\mathcal{L}$, set of formulas $\mathcal{A} \subseteq \mathcal{L}$ and formula $\varphi \in \mathcal{L}$, we write $\mathcal{A} \vdash_\Lambda \varphi$ to mean that there exists a finite set $\{\psi_1, \ldots, \psi_n\} \subseteq \mathcal{A}$ of formulas such that $(\psi_1 \wedge \cdots \wedge \psi_n) \to \varphi$ is a theorem of Λ (allowing $n = 0$ and φ is a theorem of Λ). The relation $\vdash_\Lambda \subseteq 2^{\mathcal{L}} \times \mathcal{L}$ is the consequence relation of Λ. We will abuse notation (as we have with **IPC** and $\mathbf{S4}\Box$) and identify Λ with its set of theorems: i.e. $\Lambda = \{\varphi \in \mathcal{L} \mid \varnothing \vdash_\Lambda \varphi\}$.

Let **IK** be the axiomatic system of Fischer Servi [18,15,22], which is equivalent to an alternative axiomatisation given in [29,32]; **IK** also goes by the name **FS** in [22] and [20,37,38]. **IK** has as axioms all instances in the language $\mathcal{L}^{\mathbf{m}}$ of the axiom schemes of **IPC**, and further axiom schemes:

$$\mathbf{R}\diamonddot:\ \diamonddot(\varphi \vee \psi) \leftrightarrow (\diamonddot\varphi \vee \diamonddot\psi) \qquad\qquad \mathbf{N}\diamonddot:\ \neg\diamonddot\bot$$

$$\mathbf{R}\boxdot:\ \boxdot(\varphi \wedge \psi) \leftrightarrow (\boxdot\varphi \wedge \boxdot\psi) \qquad\qquad \mathbf{N}\boxdot:\ \boxdot\top$$

$$\mathbf{F1}\boxdot\diamonddot:\ \diamonddot(\varphi \to \psi) \to (\boxdot\varphi \to \diamonddot\psi) \qquad \mathbf{F2}\boxdot\diamonddot:\ (\diamonddot\varphi \to \boxdot\psi) \to \boxdot(\varphi \to \psi)$$

and is closed under the inference rules (**MP**) and (**Subst**), and the rule ($\mathbf{Mono}\diamonddot$): from $\varphi_1 \to \varphi_2$ infer $\diamonddot\varphi_1 \to \diamonddot\varphi_2$, and likewise ($\mathbf{Mono}\boxdot$).

With regard to notation for combinations of modal logics, we follow that of [20]. If Λ_1 and Λ_2 are axiomatically presented modal logics in languages $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively, then the *fusion* $\Lambda_1 \otimes \Lambda_2$ is the smallest multi-modal logic in the language $\mathcal{L}_1 \otimes \mathcal{L}_2$ containing Λ_1 and Λ_2, and closed under all the inference rules of Λ_1 and Λ_2, where $\mathcal{L}_1 \otimes \mathcal{L}_2$ denotes the least common extension of the languages $\mathcal{L}_1$ and $\mathcal{L}_2$. If Λ is a logic in language $\mathcal{L}$, and Γ is a finite list of schemes in $\mathcal{L}$, then the *extension* $\Lambda \oplus \Gamma$ is the smallest logic in $\mathcal{L}$ extending Λ, containing the schemes in Γ as additional axioms, and closed under the rules of Λ. The basic system in

[37], under the name **IntK**, is such that: $\mathbf{IK} = \mathbf{IntK} \oplus \mathbf{F1}\boxdot\Diamonddot \oplus \mathbf{F2}\boxdot\Diamonddot$. The latter two schemes were identified by Fischer Servi in [18][5].

For the extension to tense logics with forwards and backwards modalities, let $\mathbf{IK^t}$ be Ewald's [15] deductive system, which is the fusion of $\mathbf{IK}\Diamonddot\boxdot := \mathbf{IK}$ with the "mirror" system $\mathbf{IK}\blacklozenge\blacksquare$ having axiom schemes $\mathbf{R}\blacklozenge$, $\mathbf{N}\blacklozenge$, $\mathbf{R}\blacksquare$, $\mathbf{N}\blacksquare$, $\mathbf{F1}\blacksquare\blacklozenge$ and $\mathbf{F2}\blacksquare\blacklozenge$, and inference rules ($\mathbf{Mono}\blacklozenge$) and ($\mathbf{Mono}\blacksquare$), which is then further extended with four axiom schemes expressing the *adjoint property* (Assertion (3)) of the operators interpreting the tense modalities:

$$\mathbf{Ad1} : \varphi \to \boxdot\blacklozenge\varphi \quad \mathbf{Ad2} : \varphi \to \blacksquare\Diamonddot\varphi \quad \mathbf{Ad3} : \Diamonddot\blacksquare\varphi \to \varphi \quad \mathbf{Ad4} : \blacklozenge\boxdot\varphi \to \varphi$$

Thus $\mathbf{IK^t} := (\mathbf{IK}\Diamonddot\boxdot \otimes \mathbf{IK}\blacklozenge\blacksquare) \oplus \mathbf{Ad1} \oplus \mathbf{Ad2} \oplus \mathbf{Ad3} \oplus \mathbf{Ad4}$.

We now identify the companion classical logics. Let $\mathbf{K}\boxdot$ be the minimal normal modal logic (over a classical propositional base), and let $(\mathbf{S4}\Box \otimes \mathbf{K}\boxdot)$ be the bi-modal fusion of $\mathbf{S4}\Box$ and $\mathbf{K}\boxdot$, and let $\mathbf{K^mLSC} := (\mathbf{S4}\Box \otimes \mathbf{K}\boxdot) \oplus (\Diamonddot\Box\varphi \to \Box\Diamonddot\varphi) \oplus (\Box\boxdot\varphi \to \boxdot\Box\varphi)$ be the extension of $(\mathbf{S4}\Box \otimes \mathbf{K}\boxdot)$ with characteristic modal schemes for the R-l.s.c. and R^{-1}-l.s.c. frame conditions, from Proposition 3 (and as identified in [16]). Likewise, $\mathbf{K^t} := (\mathbf{K}\boxdot \otimes \mathbf{K}\blacksquare) \oplus \mathbf{Ad1} \oplus \mathbf{Ad2}$ is the minimal normal tense logic, and $\mathbf{K^tLSC} := (\mathbf{S4}\Box \otimes \mathbf{K^t}) \oplus (\Diamonddot\Box\varphi \to \Box\Diamonddot\varphi) \oplus (\blacklozenge\Box\varphi \to \Box\blacklozenge\varphi)$, here using instead the tense scheme for R^{-1}-l.s.c. from Proposition 3.

In what follows, we will deal generically with extensions $\mathbf{IK} \oplus \Gamma$ or $\mathbf{IK^t} \oplus \Gamma$ for subsets Γ of the five axiom schemes below or their $\blacksquare$-$\blacklozenge$ mirror images:

$$\begin{array}{ll}
\mathbf{T}\boxdot\Diamonddot : (\boxdot\varphi \to \varphi) \wedge (\varphi \to \Diamonddot\varphi) & \mathbf{B}\boxdot\Diamonddot : (\varphi \to \boxdot\Diamonddot\varphi) \wedge (\Diamonddot\boxdot\varphi \to \varphi) \\
\mathbf{D}\Diamonddot : \Diamonddot\top & \\
\mathbf{4}\boxdot\Diamonddot : (\boxdot\varphi \to \boxdot\boxdot\varphi) \wedge (\Diamonddot\Diamonddot\varphi \to \Diamonddot\varphi) & \mathbf{5}\boxdot\Diamonddot : (\Diamonddot\boxdot\varphi \to \boxdot\varphi) \wedge (\Diamonddot\varphi \to \boxdot\Diamonddot\varphi)
\end{array} \tag{7}$$

where the schemes characterize, in turn, the properties of relations $R : X \rightsquigarrow X$ of reflexivity, symmetry, totality (seriality), transitivity and Euclideanness, and the mirror image scheme characterize relations R such that R^{-1} has the property[6]. For a set Γ of schemes, let $\mathbb{C}(\Gamma)$ be the set of all formulas $\varphi \in \mathcal{L}^{\mathbf{t}}$ that are int-modal-top valid in every l.s.c. topological frame whose relation R has the properties corresponding to the schemes in Γ, and let $\mathbb{C}_\Box(\Gamma)$ be the set of all formulas $\varphi \in \mathcal{L}^{\mathbf{t}}_\Box$ that are modal-top valid in every topological frame whose relation R has the properties corresponding to the schemes in Γ.

The topological soundness of $\mathbf{IK^t}$ and of $\mathbf{K^tLSC}$ are easy verifications. For example, the soundness of the Fischer Servi scheme $\mathbf{F1}\boxdot\Diamonddot$ is equivalent to the assertion that, for all open sets $U, V \in \mathcal{T}$:

$$R^{-\exists}\left(int_{\mathcal{T}}(-U \cup V)\right) \;\subseteq\; int_{\mathcal{T}}\big(-int_{\mathcal{T}}(R^{-\forall}(U)) \cup R^{-\exists}(V)\big)\,.$$

[5] The intuitionistic modal logics considered in [36] and [23] are yet weaker sub-systems: they have the normality schemes $\mathbf{R}\boxdot$ and $\mathbf{N}\boxdot$ for $\boxdot$, but $\Diamonddot$ is sub-normal – they include the scheme $\mathbf{N}\Diamonddot$, but $\mathbf{R}\Diamonddot$ is replaced by $(\boxdot\varphi \wedge \Diamonddot\psi) \to \Diamonddot(\varphi \wedge \psi)$.

[6] Note that R has reflexivity, symmetry or transitivity iff R^{-1} has the same property, so the mirrored tense schemes $\mathbf{T}\blacksquare\blacklozenge$, $\mathbf{B}\blacksquare\blacklozenge$ and $\mathbf{4}\blacksquare\blacklozenge$ are semantically equivalent to their un-mirrored modal versions.

The inclusion $R^{-\exists}\left(int_{\mathcal{T}}(-U \cup V)\right) \subseteq int_{\mathcal{T}}\left(R^{-\exists}(-U \cup V)\right)$ follows from R being l.s.c. Applying distribution over unions, duality, and monotonicity, we can get $int_{\mathcal{T}}\left(R^{-\exists}(-U \cup V)\right) \subseteq int_{\mathcal{T}}\left(-int_{\mathcal{T}}(R^{-\forall}(U)) \cup R^{-\exists}(V)\right)$, so we are done. R being l.s.c. is also used for soundness of the adjoint axioms **Ad2** and **Ad3**.

From Proposition 2 and topological completeness in Proposition 6 below, we can derive deductive faithfulness of the extended Gödel translation.

Proposition 5. [Extended Gödel translation: deductive faithfulness] *Let Γ be any finite set of schemes in $\mathcal{L}^{\mathbf{t}}$ from the list in* (7) *above. For all $\varphi \in \mathcal{L}^{\mathbf{t}}$, $\varphi \in \mathbf{IK^t} \oplus \Gamma$ iff* $\mathrm{GT}(\varphi) \in \mathbf{K^tLSC} \oplus \Gamma$.

This result can also be derived from a general result for (an equivalent) Gödel translation given in [38], Theorem 8, on the faithful embedding of modal logics $L = \mathbf{IntK} \oplus \Gamma_1$ (including $\mathbf{IK} \oplus \Gamma = \mathbf{IntK} \oplus \mathbf{F1}\boxdot\diamonddot \oplus \mathbf{F2}\boxdot\diamonddot \oplus \Gamma$) into bimodal logics in the interval between $(\mathbf{S4}\Box \otimes \mathbf{K}\boxdot) \oplus \mathrm{GT}(\Gamma_1)$ and $(\mathbf{Grz}\Box \otimes \mathbf{K}\boxdot) \oplus \mathrm{GT}(\Gamma_1) \oplus \mathbf{mix}$, where $\mathbf{Grz}\Box = \mathbf{S4}\Box \oplus \Box(\Box(\varphi \to \Box\varphi) \to \varphi) \to \varphi$ and $\mathbf{mix} = (\Box\boxdot\varphi \leftrightarrow \boxdot\varphi) \wedge (\boxdot\Box\varphi \leftrightarrow \Box\varphi)$. We have restricted the schemes in Γ to those from a "safe" list of relational properties that *don't* require translating, since the schemes characterize the same relations in the intuitionistic and classical semantics.

Recall that for a logic Λ in a language $\mathcal{L}$ with deductive consequence relation $\vdash_\Lambda$, a set of formulas $x \subseteq \mathcal{L}$ is said to be Λ-*consistent* if $x \nvdash_\Lambda \bot$; x is Λ-*deductively closed* if $x \vdash_\Lambda \varphi$ implies $\varphi \in x$ for all formulas $\varphi \in \mathcal{L}$; and x is *maximal* Λ-*consistent* if x is Λ-consistent, and no proper superset of x is Λ-consistent. A set $x \subseteq \mathcal{L}$ is a *prime theory* of Λ if $\Lambda \subseteq x$, and x has the disjunction property, and is Λ-consistent, and Λ-deductively closed.

Completeness w.r.t. bi-relational frames for **IK** and $\mathbf{IK^t}$ is proved in [18,32] and [15] by building a canonical model over the state space X_{ip} defined to be the set of all sets of formulas $x \subseteq \mathcal{L}^{\mathbf{t}}$ that are prime theories of $\mathbf{IK^t}$. The space X_{ip} is partially ordered by inclusion, so we have available an Alexandrov topology $\mathcal{T}_{\subseteq}$. One then defines the modal accessibility relation R_0 in an "almost classical" way, the only concession to intuitionistic semantics being clauses in the definition for both $\diamonddot$ and $\boxdot$. As verified in [18] and [32] for the modal logic, and [15] for the tense logic, the relations R_0 and R_0^{-1} satisfy the frame conditions $\mathbf{Zig}(\subseteq, R_0)$ and $\mathbf{Zig}(\subseteq, R_0^{-1})$. So we get an l.s.c. topological frame $\mathcal{F}_0 = (X_{\mathrm{ip}}, \mathcal{T}_{\subseteq}, R_0)$, and with the canonical valuation $u : AP \rightsquigarrow X_{\mathrm{ip}}$ given by $u(p) = \{x \in X_{\mathrm{ip}} \mid p \in x\}$; one then proves of the model $\mathcal{M}_0 = (\mathcal{F}_0, u)$ the "Truth Lemma": for all $\varphi \in \mathcal{L}^{\mathbf{t}}$ and $x \in X_{\mathrm{ip}}$, $x \in [\![\varphi]\!]_{\mathbf{I}}^{\mathcal{M}_0}$ iff $\varphi \in x$.

Adapting [1], Sec. 3, on classical **S4**, to the classical companion logics here, we can go beyond pre-orders by equipping the space of maximal consistent sets of formulas with a topology that is neither Alexandrov nor Stone, but rather is the intersection of those two topologies.

Proposition 6. [Topological soundness and completeness] *Let Γ be any finite set of axiom schemes from $\mathcal{L}^{\mathbf{t}}$ from the list in* (7) *above.*
(1.) *For all $\psi \in \mathcal{L}^{\mathbf{t}}_{\Box}$, ψ is a theorem of $\mathbf{K^tLSC} \oplus \Gamma$ iff $\psi \in \mathbb{K}^{\mathbf{t}}\mathbb{LSC} \cap \mathbb{C}_{\Box}(\Gamma)$.*
(2.) *For all $\varphi \in \mathcal{L}^{\mathbf{t}}$, φ is a theorem of $\mathbf{IK^t} \oplus \Gamma$ iff $\varphi \in \mathbb{IK}^{\mathbf{t}}\mathbb{T} \cap \mathbb{C}(\Gamma)$.*

In what follows, we use **IL** and $\mathbf{L}_\Box$, respectively, as abbreviations for the axiomatically presented logics $\mathbf{IK^t} \oplus \Gamma$ and $\mathbf{K^tLSC} \oplus \Gamma$. Taking soundness as established, we sketch completeness by describing the canonical models.

For the classical companion $\mathbf{L}_\Box$, define a model $\mathcal{M}_\Box = (Y_{\Box\mathrm{m}}, \mathcal{S}_\Box, Q_\Box, v_\Box)$:

$Y_{\Box\mathrm{m}} := \{y \subseteq \mathcal{L}^{\mathbf{t}}_\Box \mid y$ is a maximal $\mathbf{L}_\Box$-consistent set of formulas$\}$;

$\mathcal{S}_\Box$ is the topology on $Y_{\Box\mathrm{m}}$ which has as a basis the family
$\{V(\Box\psi) \mid \psi \in \mathcal{L}^{\mathbf{t}}_\Box\}$ where $V(\Box\psi) := \{y \in Y_{\Box\mathrm{m}} \mid \Box\psi \in y\}$;

$Q_\Box : Y_{\Box\mathrm{m}} \rightsquigarrow Y_{\Box\mathrm{m}}$ defined for all $y \in Y_{\Box\mathrm{m}}$ by
$Q_\Box(y) := \{ y' \in Y_{\Box\mathrm{m}} \mid \{\Diamonddot\psi \mid \psi \in y'\} \subseteq y$ and $\{\blacklozenge\psi \mid \psi \in y\} \subseteq y' \}$;

$v_\Box : AP \rightsquigarrow Y_{\Box\mathrm{m}}$ defined for all $p \in AP$ by $v_\Box(p) := \{y \in Y_{\Box\mathrm{m}} \mid p \in y\}$.

As noted in [1], the topology $\mathcal{S}_\Box$ on $Y_{\Box\mathrm{m}}$ is the intersection the "default" Alexandrov topology from the canonical relational Kripke model, and the standard Stone topology on $Y_{\Box\mathrm{m}}$ which has as a basis all sets of the form $V(\psi)$ for all formulas $\psi \in \mathcal{L}^{\mathbf{t}}_\Box$, not just the $V(\Box\psi)$ ones. Moreover, the space $(Y_{\Box\mathrm{m}}, \mathcal{S}_\Box)$ is compact and dense-in-itself (has no isolated points). Verification that $Q_\Box$ and $Q_\Box^{-1}$ are l.s.c. reduces to establishing that for all $\psi \in \mathcal{L}^{\mathbf{t}}_\Box$:

$$Q_\Box^{-\exists}(V(\Box\psi)) = V(\Box\Diamonddot\Box\psi) \quad \text{and} \quad Q_\Box^{\exists}(V(\Box\psi)) = V(\Box\blacklozenge\Box\psi).$$

The "Truth Lemma" is $y \in [\![\varphi]\!]^{\mathcal{M}_\Box}$ iff $\psi \in y$, for all $\psi \in \mathcal{L}^{\mathbf{t}}_\Box$ and $y \in Y_{\Box\mathrm{m}}$.

For the intuitionistic logic **IL**, define an open model $\mathcal{M}_\star = (X_{\mathrm{ip}}, \mathcal{T}_{\mathrm{sp}}, R_\star, u_\star)$:

$X_{\mathrm{ip}} := \{x \subseteq \mathcal{L}^{\mathbf{t}} \mid x$ is a prime **IL**-theory$\}$;

$\mathcal{T}_{\mathrm{sp}}$ is the topology on X_{ip} which has as a basis the family
$\{U(\varphi) \mid \varphi \in \mathcal{L}^{\mathbf{t}}\}$ where $U(\varphi) := \{x \in X_{\mathrm{ip}} \mid \varphi \in x\}$;

$R_\star : X_{\mathrm{ip}} \rightsquigarrow X_{\mathrm{ip}}$ defined for all $x, x' \in X_{\mathrm{ip}}$ by $R_\star := R_0$; i.e. $x\, R_\star\, x'$ iff
$\{\Diamonddot\psi \mid \psi \in x'\} \subseteq x$ and $\{\psi \mid \boxdot\psi \in x\} \subseteq x'$ and
$\{\blacklozenge\psi \mid \psi \in x\} \subseteq x'$ and $\{\psi \mid \blacksquare\psi \in x'\} \subseteq x$;

$u_\star : AP \rightsquigarrow X_{\mathrm{ip}}$ defined for all $p \in AP$ by $u_\star(p) := U(p)$.

Here, the toplogical space $(X_{\mathrm{ip}}, \mathcal{T}_{\mathrm{sp}})$ has a *spectral topology* (e.g. [33], Sec.4), which means it is compact and T_0; the family of compact and open sets in $\mathcal{T}_{\mathrm{sp}}$ gives a basis for the toplogy; and $\mathcal{T}_{\mathrm{sp}}$ is sober, i.e. for every completely prime filter $\mathcal{F}$ of $\mathcal{T}_{\mathrm{sp}}$, there exists a (unique) point $x \in X_{\mathrm{ip}}$ such that $\mathcal{F} = \mathcal{F}_x := \{U \in \mathcal{T}_{\mathrm{sp}} \mid x \in U\}$, the filter of neighbourhoods of x. The hardest parts of the verification are the l.s.c. properties for $R_\star$ and $R_\star^{-1}$, and the task reduces to establishing that for all $\varphi \in \mathcal{L}^{\mathbf{t}}$:

$$R_\star^{-\exists}(U(\varphi)) = U(\Diamonddot\varphi) \qquad \text{and} \qquad R_\star^{\exists}(U(\varphi)) = U(\blacklozenge\varphi)$$

To prove the right-to-left inclusions, a recursive Henkin-style construction can be used to produce a prime **IL**-theory x' such that $x\, R_\star\, x'$ and $\varphi \in x'$, to derive $x \in R_\star^{-\exists}(U(\varphi))$ given $x \in U(\Diamonddot\varphi)$, and symmetrically for the $R_\star^{\exists}(U(\varphi))$ inclusion. The required "Truth Lemma" is $x \in [\![\varphi]\!]_{\mathbf{I}}^{\mathcal{M}_\star}$ iff $\varphi \in x$ for all $\varphi \in \mathcal{L}^{\mathbf{t}}$ and $x \in X_{\mathrm{ip}}$.

6 Topological Bisimulation Between Canonical Models

The Gödel translation is a syntactic function $\mathrm{GT} : \mathcal{L}^{\mathbf{t}} \to \mathcal{L}^{\mathbf{t}}_{\Box}$, which naturally gives rise to a semantic relationship between the canonical model spaces X_{ip} and $Y_{\Box\mathrm{m}}$. Define a set-valued map $G : X_{\mathrm{ip}} \rightsquigarrow Y_{\Box\mathrm{m}}$ by:

$$G(x) \;:=\; \{\, y \in Y_{\Box\mathrm{m}} \mid \mathrm{GT}(x) \subseteq y \,\}$$

Note that the image $\mathrm{GT}(x)$ of an intuitionistic prime theory $x \in X_{\mathrm{ip}}$ will in general have many classical maximal consistent extensions $y \in Y_{\Box\mathrm{m}}$.

Let $\mathcal{M}^*_{\Box} = (Y_{\Box\mathrm{m}}, \mathcal{S}^*_{\Box}, Q_{\Box}, v^*_{\Box})$ be the open and l.s.c. model obtained from $\mathcal{M}_{\Box}$ by taking $\mathcal{S}^*_{\Box}$ to be the proper sub-topology of $\mathcal{S}_{\Box}$ having as a basis the open sets $\{\, V(\Box\mathrm{GT}(\varphi)) \mid \varphi \in \mathcal{L}^{\mathbf{t}} \,\}$[7], with valuation $v^*_{\Box}(p) := \mathit{int}_{\mathcal{S}^*_{\Box}}(v_{\Box}(p)) = V(\Box p)$.

Theorem 2. *The maps $G : X_{\mathrm{ip}} \rightsquigarrow Y_{\Box\mathrm{m}}$ and $G^{-1} : Y_{\Box\mathrm{m}} \rightsquigarrow X_{\mathrm{ip}}$ are such that:*
(1.) *both G and G^{-1} are l.s.c. with respect to $\mathcal{T}_{\mathrm{sp}}$ and $\mathcal{S}^*_{\Box}$;*
(2.) *both G and G^{-1} are total and surjective;*
(3.) *$R_\star \circ G \;=\; G \circ Q_{\Box}$ and $Q_{\Box} \circ G^{-1} \;=\; G^{-1} \circ R_\star$; and*
(4.) *$G^{\exists}(u_\star(p)) \subseteq v_{\Box}(p)$ and $G^{-\exists}(v_{\Box}(p)) \subseteq u_\star(p)$ for all atomic $p \in AP$.*
Hence G is a tense topo-bisimulation between $\mathcal{M}_\star$ and $\mathcal{M}^*_{\Box}$.

Proof. For Part (1.), the l.s.c. properties, we need only look at the basic opens in $\mathcal{T}_{\mathrm{sp}}$ and $\mathcal{S}^*_{\Box}$. Using the openness theorem $\Box\mathrm{GT}(\varphi) \leftrightarrow \mathrm{GT}(\varphi)$, it is readily established that for all $\varphi \in \mathcal{L}^{\mathbf{t}}$:

$$G^{-\exists}(V(\Box\mathrm{GT}(\varphi))) \;=\; U(\varphi) \qquad \text{and} \qquad G^{\exists}(U(\varphi)) \;=\; V(\Box\mathrm{GT}(\varphi)).$$

For Part (2.), the totality of G, note that every prime theory $x \in X_{\mathrm{ip}}$ is **IL**-consistent, hence the image $\mathrm{GT}(x) \subseteq \mathcal{L}_{\Box}$ is $\mathbf{L}_{\Box}$-consistent, and so has a maximal $\mathbf{L}_{\Box}$-consistent superset $y \supseteq \mathrm{GT}(x)$ with $y \in Y_{\Box\mathrm{m}}$, by Lindenbaum's Lemma. For the surjectivity of G (equivalently, the totality of G^{-1}), define as follows the (proper) subset $\mathcal{G}^*$ of formulas $\mathbf{L}_{\Box}$-equivalent to the image under GT of some $\Box$-free formula: $\mathcal{G}^* := \{\psi \in \mathcal{L}^{\mathrm{t}}_{\Box} \mid (\exists\varphi \in \mathcal{L}^{\mathrm{t}}) \;\vdash_{\mathbf{L}_{\Box}}\; \psi \leftrightarrow \mathrm{GT}(\varphi)\,\}$. Now for any maximal $\mathbf{L}_{\Box}$-consistent theory $y \in Y_{\Box\mathrm{m}}$, define the subset $y^* := y \cap \mathcal{G}^*$. Define $X^{\mathrm{m}}_{\mathrm{ip}} := \{x_0 \in X_{\mathrm{ip}} \mid (\forall x \in X_{\mathrm{ip}})\; x_0 \nsubseteq x\}$ to be the (proper) subset of prime **IL** theories that are $\subseteq$-maximal. Then every $x_0 \in X^{\mathrm{m}}_{\mathrm{ip}}$ is a maximal **IL**-consistent theory, and is also a classical $\mathbf{L}_{\Box}$-consistent theory that is maximal within the $\Box$-free language $\mathcal{L}^{\mathbf{t}}$. So by the deductive faithfulness of the Gödel translation, for every $y \in Y_{\Box\mathrm{m}}$, there is a maximal $x_0 \in X^{\mathrm{m}}_{\mathrm{ip}}$ such that $\mathrm{GT}(x_0) = y^*$, and hence $\mathrm{GT}(x_0) \subseteq y$. Hence G is surjective. The verifications for the remaining Parts (3.) and (4.) are somewhat lengthy, but straight-forward. $\dashv$

In a sequel [10], we return to the intuitionistic canonical model $\mathcal{M}_\star$, and use it to give a Hennessy-Milner type result on maximal topological bisimulations preserving the intuitionistic semantics. For both the intuitionistic and classical

[7] $\mathcal{M}^*_{\Box}$ will be an l.s.c. model, as $Q_{\Box}$ and $Q^{-1}_{\Box}$ will still be l.s.c. w.r.t. the sub-topology $\mathcal{S}^*_{\Box}$; using $Q^{-\exists}_{\Box}(V(\Box\psi)) = V(\Box\Diamond\Box\psi)$ and $Q^{\exists}_{\Box}(V(\Box\psi)) = V(\Box\blacklozenge\Box\psi)$ and the openness property $\mathrm{GT}(\varphi) \leftrightarrow \Box\mathrm{GT}(\varphi)$, we have $Q^{-\exists}_{\Box}(V(\Box\mathrm{GT}(\varphi))) = V(\Box\mathrm{GT}(\Diamond\varphi))$ and $Q^{\exists}_{\Box}(V(\Box\mathrm{GT}(\varphi))) = V(\Box\mathrm{GT}(\blacklozenge\varphi))$.

semantics, the classes of models identified have suitable 'saturation' properties w.r.t. the semantics, and the maximal topo-bisimulations are constructed via natural maps into the canonical models. Within these Hennessy-Milner classes, we identify some subclasses of models of continuous, discrete and hybrid dynamical systems.

References

1. M. Aiello, J. van Benthem, and G. Bezhanishvili. Reasoning about space: the modal way. *J. Logic and Computation*, 13:889–920, 2003.
2. J-P. Aubin and H. Frankowska. *Set-Valued Analysis*. Birkhäuser, Boston, 1990.
3. J.-P. Aubin, J. Lygeros, M. Quincampoix, S. Sastry, and N. Seube. Impulse differential inclusions: A viability approach to hybrid systems. *IEEE Trans. Automatic Control*, 47:2–20, 2002.
4. G. Bezhanishvili. Varieties of monadic Heyting algebras. part I. *Studia Logica*, 61:362–402, 1999.
5. G. Bezhanishvili. Varieties of monadic Heyting algebras. part II: Duality theory. *Studia Logica*, 62:21–48, 1999.
6. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001.
7. Marcello M. Bonsangue and Joost N. Kok. Relating multifunctions and predicate transformers through closure operators. In *Theoretical Aspects of Computer Software (TACS '94)*, pages 822–843, 1994.
8. R.A. Bull. MIPC as a formalization of an Intuitionist concept of modality. *J. Symbolic Logic*, 31:609–616, 1966.
9. J.M. Davoren. Topologies, continuity and bisimulations. *Theoretical Informatics and Applications*, 33:357–381, 1999.
10. J.M. Davoren. Topological bisimulations and Hennessy-Milner classes for intuitionistic and classical spatio-temporal modal logics. Technical report, Department of Electrical & Electronic Engineering, University of Melbourne, January 2007.
11. J.M. Davoren, V. Coulthard, N. Markey, and T. Moor. Non-deterministic temporal logics for general flow systems. In R. Alur and G.J. Pappas, editors, *Hybrid Systems: Computation and Control* (HSCC'04), LNCS 2993, pages 280–295. Springer, 2003.
12. J.M. Davoren, V. Coulthard, T. Moor, R.P. Goré, and A. Nerode. Topological semantics for intuitionistic modal logics, and spatial discretisation by A/D maps. In *Workshop on Intuitionistic Modal Logic and Applications (IMLA), Copenhagen, Denmark*, 2002. Proceedings available as Technical Report No. 61, University of Bamberg, Faculty of Information Systems and Applied Computer Sciences.
13. J.M. Davoren and R.P. Goré. Bimodal logics for reasoning about continuous dynamics. In *Advances in Modal Logic 3*, pages 91–110. World Scientific, 2002.
14. L. Esakia. Topological kripke models. *Soviet Mathematics: Doklady*, 15:147–151, 1974. English Translation.
15. W.B. Ewald. Intuitionistic tense and modal logic. *J. of Symbolic Logic*, 51:166–179, 1986.
16. G. Fischer Servi. On modal logic with an Intuitionistic base. *Studia Logica*, 36:141–149, 1977.
17. G. Fischer Servi. Semantics for a class of Intuitionistic modal calculi. In *Italian Studies in the Philosophy of Science*, pages 59–72. D. Reidel, 1981.

18. G. Fischer Servi. Axiomatizations for some Intuitionistic modal logics. *Rend. Sem. Mat. Univers. Politecn. Torino*, 42:179–194, 1984.
19. M.C. Fitting. *Intuitionistic Logic, Model Theory and Forcing.* North Holland, 1968.
20. D.M. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*, volume 148 of *Studies in Logic.* Elsevier, 2003.
21. K. Gödel. An interpretation of the Intuitionistic propositional calculus (1933). In S. Feferman, editor, *Collected Works*, volume 1, pages 301–303. Oxford UP, 1989. Publications 1929-1936.
22. C. Grefe. Fischer Servi's intuitionistic modal logic has the finite model property. In *Advances in Modal Logic*, volume 1. CSLI, Stanford, 1998.
23. B.P. Hilken. Topological duality for intuitionistic modal algebras. *J. of Pure and Applied Algebra*, 148:171–189, 2000.
24. K. Kuratowski. *Topology.* Academic Press, New York, 1966.
25. J.C.C. McKinsey and A. Tarski. The algebra of topology. *Annals of Mathematics*, pages 141–191, 1944.
26. G. Mints. *A Short Introduction to Intuitionistic Logic.* Kluwer, New York, 2000.
27. A. Monteiro and O. Varsavsky. Algebras de Heyting monadicas. *Actas de las X Jornadas de la Union Matematica Argentina*, pages 52–62, 1957.
28. H. Ono. On some Intuitionistic modal logics. *Publications of the Research Institute for Mathematical Science, Kyoto University*, 13:55–67, 1977.
29. G. Plotkin and C. Stirling. A framework for intuitionistic modal logics. In J. Y. Halpern, editor, *Proc. 1986 Conference: Theoretical Aspects of Reasoning About Knowledge*, 1986.
30. A. Prior. *Time and Modality.* Clarendon Press, Oxford, 1957.
31. H. Rasiowa and R. Sikorski. *Mathematics of Metamathematics.* PWN Warsaw, 1963.
32. A.K. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic.* PhD thesis, Department of Computer Science, University of Edinburgh, 1994.
33. M.B. Smyth. Topology. In S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science, Vol 1*, pages 641–751. Oxford Science, 1992.
34. N.-Y. Suzuki. An algebraic approach to Intuitionistic modal logics in connection with Intuitionistic predicate logic. *Studia Logica*, 48:141–155, 1988.
35. O. Varsavsky. Quantifiers and equivalence relations. *Revista Matematica Cuyana*, 2:29–51, 1956.
36. D. Wijesekera. Constructive modal logics I. *Annals of Pure and Applied Logic*, 50:271–301, 1990.
37. F. Wolter and M. Zakharyaschev. Intuitionistic modal logic. In *Logic and Foundations of Mathematics*, pages 227 – 238. Kluwer Academic Publishers, 1995.
38. F. Wolter and M. Zakharyaschev. Intuitionistic modal logics as fragments of classical bimodal logics. In E. Orlowska, editor, *Logic at Work*, pages 168 – 186. Kluwer Academic Publishers, 1998.

A Decidable Temporal Logic of Repeating Values*

Stéphane Demri[1], Deepak D'Souza[2], and Régis Gascon[1]

[1] LSV, ENS Cachan, CNRS, Inria
{demri,gascon}@lsv.ens-cachan.fr
[2] Dept. of Computer Science & Automation,
Indian Institute of Science, Bangalore, India
deepakd@csa.iisc.ernet.in

Abstract. Various logical formalisms with the freeze quantifier have been recently considered to model computer systems even though this is a powerful mechanism that often leads to undecidability. In this paper, we study a linear-time temporal logic with past-time operators such that the freeze operator is only used to express that some value from an infinite set is repeated in the future or in the past. Such a restriction has been inspired by a recent work on spatio-temporal logics. We show decidability of finitary and infinitary satisfiability by reduction into the verification of temporal properties in Petri nets. This is a surprising result since the logic is closed under negation, contains future-time and past-time temporal operators and can express the nonce property and its negation. These ingredients are known to lead to undecidability with a more liberal use of the freeze quantifier.

1 Introduction

Temporal logic with freeze. In logical languages, the freeze mechanism allows to store a value in a register and to test later the value in the register with a current value. This operator is useful to compare values at distinct states of Kripke-like structures. The freeze quantifier has found applications in real-time logics [Hen90], in hybrid logics [Gor96, ABM01], in modal logics with predicate λ-abstraction [Fit02] and for the specification of computations of systems with unboundedly many locations as resources [LP05]. Although it is known that the freeze operator can lead to undecidability (even with only equality on data [LP05, DLN07]), many decidable temporal logics have a freeze mechanism, sometimes implicitly, see e.g. [AH94, LMS02, KV06]. Recent developments have shown the ubiquity of the freeze operator [LP05, tCF05, DLN07, Laz06, Seg06] and its high expressive power as witnessed by the Σ_1^1-completeness results shown in [DLN07].

* Work supported by the Indo-French project "Timed-DISCOVERI" (P2R/RNP scheme).

The need to design decidable fragments of simple linear-time temporal logic LTL with the freeze quantifier stems from [DLN07, Laz06] and most known decidable fragments in [DLN07, Laz06] does not allow unrestricted use of negation. Still, finitary and infinitary satisfiability for Boolean combinations of safety formulae (with a unique register) is decidable [Laz06]. Potential applications range from the verification of infinite-state systems [Hen90, DLN07] to querying XML documents or more modestly data strings [BMS$^+$06, Seg06]. In the paper, we are interested in studying fragments of LTL with the freeze operator that are decidable in the finitary and infinitary cases, that allow unrestricted use of negation (by contrast to the flat fragments in [DLN07]) and that allow all standard past-time operators (by constrast to what is done in [BMS$^+$06, DL06]). Even in terms of expressive power, the fragment newly shown decidable in the paper can express the "nonce property" and its negation (all the values of a variable are different at every position). Moreover, in [WZ00, Sect. 7], the authors advocate the need to consider infinitary disjunction of the form $\bigvee_{i>0} x = \mathsf{X}^i y$ where $\mathsf{X}^i y$ refers to the value of y at the i^{th} next position. This states that a future value of y is equal to the current value of x. Our fragment can express this property, with the formula $x = \Diamond y$, as well as the dual one: $\bigwedge_{i>0} x = \mathsf{X}^i y$ can be expressed by the formula $\neg(x \neq \Diamond y)$. In the paper we introduce the constraint logic CLTL$(\mathbb{N}, =)$ with atomic formulae $x = \Diamond y$ and past-time operators X^{-1} and S. This logic is denoted by CLTL$^{\Diamond}$. Hence, in CLTL$^{\Diamond}$, the freeze quantifier is only used to specify that some values are repeated. Even though CLTL$^{\Diamond}$ does not enjoy first-order completeness, see e.g. [Rab06], it satisfies interesting computational properties as shown below.

Our contribution. We show that finitary and infinitary satisfiability for CLTL$^{\Diamond}$ with temporal operators $\{\mathsf{X}, \mathsf{X}^{-1}, \mathsf{S}, \mathsf{U}\}$ is decidable. We provide a uniform proof for the finite and infinite cases based on some substantial extension of the automaton-based approach for (constraint) LTL from [VW94, DD07]. The possibility to compare two values at unbounded distance requires a special class of counter automata for which finitary and infinitary nonemptiness is shown decidable. To do so, we take advantage of a deep result from [Jan90] establishing that verifying fairness properties based on the temporal operator GF ("always eventually") in Petri nets is decidable. By contrast model-checking for full LTL over Petri nets is undecidable [HR89] (see also [Esp94] with linear-time mu-calculi). Observe that infinitary CLTL$^{\Diamond}$ is the first decidable fragment of CLTL$^{\downarrow}_{1}(\mathbb{N}, =)$ [DLN07] with an unrestricted use of negation and that contains all the temporal operators from $\{\mathsf{X}, \mathsf{X}^{-1}, \mathsf{S}, \mathsf{U}\}$. A nice by-product of our technique is that the extensions with temporal operators definable in Monadic Second Order Logic (MSOL) or with $x = \Diamond^{-1} y$ ("a value of y in the past is equal to the current value of x") are also decidable. Finally, we show that finitary and infinitary satisfiability for CLTL$^{\Diamond}$ restricted to one variable is PSPACE-complete.

Because of lack of space, omitted proofs can be found in [DDG07].

2 Preliminaries

2.1 Temporal Logic with Repeating Values

Let VAR $= \{x_1, x_2, \ldots\}$ be a countably infinite set of variables. The formulae of the logic CLTL$^{\Diamond}$ are defined as follows:

$$\phi ::= x = \mathsf{X}^i y \mid x = \Diamond y \mid \phi \wedge \phi \mid \neg\phi \mid \mathsf{X}\phi \mid \phi \mathsf{U} \phi \mid \mathsf{X}^{-1}\phi \mid \phi \mathsf{S} \phi$$

where $x, y \in$ VAR and $i \in \mathbb{N}$. Formulae of the form either $x = \mathsf{X}^i y$ or $x = \Diamond y$ are said to be atomic and an expression of the form $\mathsf{X}^i x$ (i next symbols followed by a variable) is called a term. Given a set of temporal operators definable from those in $\{\mathsf{X}, \mathsf{X}^{-1}, \mathsf{S}, \mathsf{U}\}$ and $k \geq 0$, we write CLTL$^{\Diamond}_k(\mathcal{O})$ to denote the fragment of CLTL$^{\Diamond}$ restricted to formulae with temporal operators from $\mathcal{O}$ and with at most k variables.

A valuation is a map VAR $\rightarrow \mathbb{N}$ and a model σ is a non-empty sequence of valuations either finite or infinite. All the subsequent developments can be equivalently done with the domain $\mathbb{N}$ replaced by an infinite set D since only equality tests are performed. We write $|\sigma|$ to denote the length of σ. The satisfaction relation is defined inductively as follows where σ is a model and $0 \leq i \leq |\sigma| - 1$:

- $\sigma, i \models x = \mathsf{X}^j y$ iff $i + j \leq |\sigma| - 1$ and $\sigma(i)(x) = \sigma(i + j)(y)$,
- $\sigma, i \models x = \Diamond y$ iff there exists $j > 0$ s.t. $i+j \leq |\sigma|-1$ and $\sigma(i)(x) = \sigma(i+j)(y)$,
- $\sigma, i \models \phi \wedge \phi'$ iff $\sigma, i \models \phi$ and $\sigma, i \models \phi'$, $\quad \sigma, i \models \neg\phi$ iff $\sigma, i \not\models \phi$,
- $\sigma, i \models \mathsf{X}\phi$ iff $i + 1 \leq |\sigma| - 1$ and $\sigma, i + 1 \models \phi$,
- $\sigma, i \models \mathsf{X}^{-1}\phi$ iff $i > 0$ and $\sigma, i - 1 \models \phi$,
- $\sigma, i \models \phi \mathsf{U} \phi'$ iff there is $i \leq j \leq |\sigma| - 1$ s.t. $\sigma, j \models \phi'$ and for every $i \leq l < j$, $\sigma, l \models \phi$.
- $\sigma, i \models \phi \mathsf{S} \phi'$ iff there is $0 \leq j \leq i$ s.t. $\sigma, j \models \phi'$ and for $j \leq l < i$, $\sigma, l \models \phi$.

We write $\sigma \models \phi$ if $\sigma, 0 \models \phi$. We shall use the standard abbreviations about the temporal operators (G, F, F^{-1}, ...) and Boolean operators ($\vee$, $\Rightarrow$, ...). We use the notation $\mathsf{X}^i x = \mathsf{X}^j y$ as an abbreviation for $\mathsf{X}^i(x = \mathsf{X}^{j-i} y)$ (when $i \leq j$).

The *finitary [resp. infinitary] satisfiability problem* consists in checking whether given a formula ϕ, there is a finite [resp. infinite] model such that $\sigma \models \phi$. It is known that finitary satisfiability for LTL can be easily reduced in logspace to infinitary satisfiability by introducing for instance an additional propositional variable p and by requiring that $p\mathsf{UG}\neg p$ holds true. In that way, p holds true at every state of a prefix and p does not hold on the complement suffix. The same principle does not apply to reduce finitary satisfiability for CLTL$^{\Diamond}$ to infinitary satisfiability even by introducing additional variables in order to simulate a propositional variable. This is due to the additional atomic formulae of the form $x = \Diamond y$. That is why we distinguish the two problems in this paper.

We note that a constraint of the form x *diff* $\Diamond y$ ("the value of x differs from some future value of y") can be expressed in CLTL$^{\Diamond}$:

$$x \textit{ diff } \Diamond y \;\Leftrightarrow\; (\neg(x = \mathsf{X}y) \wedge \mathsf{X}\top) \vee ((x = \mathsf{X}y) \wedge \mathsf{X}(y = \mathsf{X}y)\mathsf{U}((y \neq \mathsf{X}y) \wedge \mathsf{X}\top))) \quad (1)$$

With infinite models, the conjunct $\mathsf{X}\top$ can be deleted. We could also introduce constraints of the form $x = \mathsf{X}^{-i}y$ but this can be expressed in the language using the equivalence $x = \mathsf{X}^{-i}y \Leftrightarrow \mathsf{X}^{-i}\top \wedge \mathsf{X}^{-i}(y = \mathsf{X}^{i}x)$. Similarly, CLTL$^{\Diamond}$ can express whether a variable is a nonce by the formula $\mathsf{G}\neg(x = \Diamond x)$. The formula below states a valid property when x and y are nonces:

$$(\mathsf{G}\neg(x = \Diamond x) \wedge \mathsf{G}\neg(y = \Diamond y)) \Rightarrow \mathsf{G}(x = y \Rightarrow \neg(x = \Diamond y)).$$

Other properties witnessing the high expressive power of CLTL$^{\Diamond}$ can be found in [LP05, Sect.3] about systems of pebbles evolving in time.

Apart from the above-mentioned problems, we could also consider standard model-checking problems that can be reduced to the satisfiability problem. Indeed, we can define a suitable class of automata such that the execution of any automaton of this class can be encoded into a CLTL$^{\Diamond}$ formula.

2.2 Known Extensions of CLTL$^{\Diamond}$

In this section, we recall the definition of a few known extensions of CLTL$^{\Diamond}$ which is useful for future comparisons. The logic CLTL$^{\Diamond}$ is clearly a fragment of the logic CLTL$^{\downarrow}(\mathbb{N}, =)$ introduced in [DLN07] and restricted to one register. An equivalent logic of CLTL$^{\downarrow}(\mathbb{N}, =)$ is denoted by CLTL$^{\downarrow}$ in this paper and it is defined as follows. We consider an additional set of registers $\mathrm{REG} = \{r_1, r_2, \ldots\}$ and the formulae of CLTL$^{\downarrow}$ are defined as those of CLTL$^{\Diamond}$ except that we allow only atomic formulae of the form $r = x$ where $r \in \mathrm{REG}$ and $x \in \mathrm{VAR}$, and we add the inductive clause $\downarrow_{r=x} \phi$. The satisfaction relation is parameterized by a register assignment $\rho : \mathrm{REG} \rightarrow \mathbb{N}$ with $\sigma, i \models_{\rho} \downarrow_{r=x} \phi$ iff $\sigma, i \models_{\rho[r \mapsto \sigma(i)(x)]} \phi$ and $\sigma, i \models_{\rho} r = x$ iff $\rho(r) = \sigma(i)(x)$. Consequently, the atomic formula $x = \Diamond y$ in CLTL$^{\Diamond}$ can be naturally encoded in CLTL$^{\downarrow}$ by $\downarrow_{r=x} \mathsf{XF}(r = y)$ and $x = \mathsf{X}^{i}y$ by $\downarrow_{r=x} \mathsf{X}^{i}(r = y)$.

We write CLTL$^{\downarrow}_{(k,k')}(\mathcal{O})$ to denote the fragment of CLTL$^{\downarrow}$ restricted to the temporal operators from $\mathcal{O}$ with at most k variables and k' registers. Following the notation from [DL06], for $\alpha \geq 0$ we write LTL$^{\downarrow}_{\alpha}(\sim, \mathcal{O})$ to denote the fragment CLTL$^{\downarrow}_{(1,\alpha)}(\mathcal{O})$ restricted to atomic formulae of the form $r = x$.

2.3 A Decidable Fragment of Finitary Satisfiability

It is shown in [DLN07] that CLTL$^{\downarrow}$ is strictly more expressive than its freeze-free fragment. The same argument applies to show that CLTL$^{\Diamond}$ is strictly more expressive than its fragment without atomic formulae of the form $x = \Diamond y$. Observe also that CLTL$^{\Diamond}$ is neither a fragment of the pure-future safety fragment in [Laz06] (where occurrences of U formulae are never in the scope of an even number of negations) nor a fragment of the flat fragment of CLTL$^{\downarrow}$. Unlike these fragments, CLTL$^{\Diamond}$ contains past-time operators and negation can be used without any restriction. Infinitary satisfiability for safety LTL$^{\downarrow}_{1}(\sim, \mathsf{X}, \mathsf{U})$ is EXPSPACE-complete [Laz06], for full LTL$^{\downarrow}_{1}(\sim, \mathsf{X}, \mathsf{U})$ is Π^0_1-complete and, finitary and infinitary satisfiability for flat CLTL$^{\downarrow}$ are PSPACE-complete. By contrast, in this paper we show that finitary and infinitary satisfiability for $\mathrm{CLTL}^{\Diamond}$ (with full past-time temporal operators) are decidable problems. By taking advantage of [DLN07, DL06], it is already possible to establish decidability of *finitary* satisfiability for *strict* fragments of $\mathrm{CLTL}^{\Diamond}$.

Theorem 1. (I) *Finitary satisfiability for* $\mathrm{CLTL}^{\Diamond}(\mathsf{X}, \mathsf{U})$ *is decidable.*
(II) *Finitary satisfiability for* $\mathrm{CLTL}^{\Diamond}(\mathsf{X}, \mathsf{X}^{-1}, \mathsf{F}, \mathsf{F}^{-1})$ *is decidable.*

In the paper we shall show much stronger results: finitary and infinitary satisfiability for full $\mathrm{CLTL}^{\Diamond}$ even augmented with MSOL definable temporal operators are decidable. In the rest of the paper, we systematically treat the finitary and infinitary cases simultaneously. However, we provide the full technical details for the infinitary case only and we sketch the main ideas for the finitary case. This latter case cannot be reduced in the obvious way to the infinitary case but its solution is close to the one for the infinitary case.

3 Automata-Based Approach with Symbolic Models

In this section we explain how satisfiability can be solved using symbolic models which are abtractions of concrete $\mathrm{CLTL}^{\Diamond}$ models. We provide the outline of our automata-based approach, and consider the technical details in Sects. 4 and 5.

3.1 Symbolic Models

Let ϕ be a $\mathrm{CLTL}^{\Diamond}$ formula with k variables $\{x_1, \ldots x_k\}$ and we write l (the "X-length" of ϕ) to denote the maximal i such that a term of the form $\mathsf{X}^i x$ occurs in ϕ. In order to define the set of atomic formulae that are helpful to determine the satisfiability status of ϕ, we introduce the set of constraints Ω^l_k that contains constraints of the form either $\mathsf{X}^i x = \mathsf{X}^j y$ or $\mathsf{X}^i(x = \Diamond y)$ and their negation with $x, y \in \{x_1, \ldots x_k\}$ and $i, j \in \{0, \ldots, l\}$. Models are abstracted as sequences of frames that are defined as maximally consistent subsets of Ω^l_k.

An *l-frame* is a set $fr \subseteq \Omega^l_k$ that is maximally consistent in that it satisfies the conditions below:

(F1) For every constraint $\varphi \in \Omega^l_k$, either φ or $\neg\varphi$ belongs to fr but not both.
(F2) For all $i \in \{0, \ldots, l\}$ and $x \in \{x_1, \ldots x_k\}$, $\mathsf{X}^i x = \mathsf{X}^i x \in fr$.
(F3) For all $i, j \in \{0, \ldots, l\}$ and $x, y \in \{x_1, \ldots x_k\}$, $\mathsf{X}^i x = \mathsf{X}^j y \in fr$ iff $\mathsf{X}^j y = \mathsf{X}^i x \in fr$.
(F4) For all $i, j, j' \in \{0, \ldots, l\}$ and $x, y, z \in \{x_1, \ldots, x_k\}$, $\{\mathsf{X}^i x = \mathsf{X}^j y, \mathsf{X}^j y = \mathsf{X}^{j'} z\} \subseteq fr$ implies $\mathsf{X}^i x = \mathsf{X}^{j'} z \in fr$.
(F5) For all $i, j \in \{0, \ldots, l\}$ and $x, y \in \{x_1, \ldots x_k\}$ such that $\mathsf{X}^i x = \mathsf{X}^j y \in fr$:
- if $i = j$, then for every $z \in \{x_1, \ldots, x_k\}$ we have $\mathsf{X}^i(x = \Diamond z) \in fr$ iff $\mathsf{X}^j(y = \Diamond z) \in fr$;
- if $i < j$ then $\mathsf{X}^i(x = \Diamond y) \in fr$, and for $z \in \{x_1, \ldots, x_k\}$, $\mathsf{X}^i(x = \Diamond z) \in fr$ iff either $\mathsf{X}^j(y = \Diamond z) \in fr$ or there exists $i < j' \leq j$ such that $\mathsf{X}^i x = \mathsf{X}^{j'} z \in fr$.

Conditions (F2)–(F4) simply encode that equality is an equivalence relation.

For an l-frame *fr*, $x \in \{x_1, \ldots, x_k\}$ and $i \in \{0, \ldots, l\}$, we define

– the set of future obligations for x at level i in *fr* as $\Diamond_{fr}(x,i) \stackrel{\text{def}}{=} \{y \mid \mathsf{X}^i(x = \Diamond y) \in fr\}$,
– the equivalence class of x at level-i in *fr* as $[(x,i)]_{fr} \stackrel{\text{def}}{=} \{y \mid \mathsf{X}^i x = \mathsf{X}^i y \in fr\}$.

An l-frame *fr* can be represented as an annotated undirected graph G_{fr} which has vertices (x, i) for $x \in \{x_1, \ldots, x_k\}$ and $i \in \{0, \ldots, l\}$, and an edge between (x, i) and (y, j) iff the constraint $\mathsf{X}^i x = \mathsf{X}^j y$ belongs to *fr*. A vertex (x, i) in the graph is annoted with an "open" arc labelled by the set of future obligations $\Diamond_{fr}(x, i)$ for that vertex. Fig. 1 shows an example of a 3-frame over the variables $\{x, y, z\}$. For convenience we avoid showing transitively inferable edges.

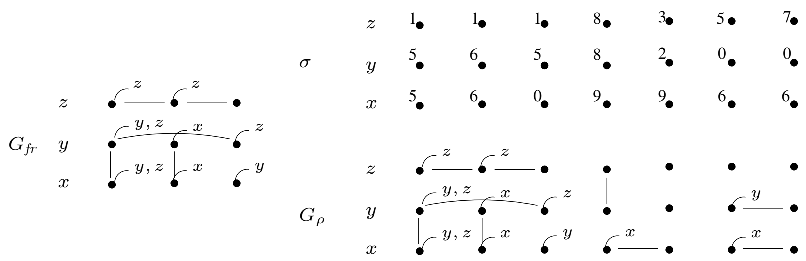


Fig. 1. Example 3-frame graph, concrete model σ, and its induced 3-frame graph G_ρ

We denote by $\texttt{Frame}^l_k$ the set of such frames built w.r.t. k and l. We say that a model σ satisfies a frame *fr* at position i (denoted $\sigma, i \models fr$) iff $\sigma, i \models \varphi$ for every constraint φ in *fr*.

Since a frame can be viewed as a set of constraints about $l + 1$ consecutive positions, for finitary satisfiability we need to add an information about the possibility to end the model before the end of the current window of length $l+1$. This can be done with $\mathcal{O}(l)$ bits and then the conditions (F1)–(F5) need to be updated accordingly in order to take into account this possibility. Note that this method allows to handle the particular case where there exists a model whose size is smaller than the X-length of the formula.

Lemma 1. *For all models σ with k variables and $0 \leq i \leq |\sigma| - 1$, there exists a unique frame $fr \in \texttt{Frame}^l_k$ such that $\sigma, i \models fr$.*

A pair of l-frames $\langle fr, fr' \rangle$ is said to be *one-step consistent* $\stackrel{\text{def}}{\Leftrightarrow}$

– for all $0 < i, j \leq l$, $\mathsf{X}^i x = \mathsf{X}^j y \in fr$ iff $\mathsf{X}^{i-1} x = \mathsf{X}^{j-1} y \in fr'$,
– for all $0 < i \leq l$, $\mathsf{X}^i(x = \Diamond y) \in fr$ iff $\mathsf{X}^{i-1}(x = \Diamond y) \in fr'$.

A *symbolic model* of X-length l is a (finite or infinite) sequence of l-frames ρ such that for $0 \leq i < |\rho| - 1$, $\langle \rho(i), \rho(i + 1) \rangle$ is one-step consistent. We define the symbolic satisfaction relation $\rho, i \models_{\text{symb}} \phi$, for a formula ϕ of X-length l and a symbolic model ρ of X-length l, as done for CLTL$^\Diamond$ except that for atomic formulas φ we have: $\rho, i \models_{\text{symb}} \varphi \overset{\text{def}}{\Leftrightarrow} \varphi \in \rho(i)$. We say a model σ *realizes* a symbolic model ρ (or equivalently that ρ *admits* a model σ) $\overset{\text{def}}{\Leftrightarrow}$ for every $0 \leq i \leq |\sigma| - 1$, we have $\sigma, i \models \rho(i)$.

A symbolic model ρ of X-length l can also be represented as an annotated graph G_ρ in a similar manner to l-frames. Thus the vertices of G_ρ are of the form (x, i) with an edge between (x, i) and (y, j) with $0 \leq j - i \leq l$ iff there was an edge between $(x, 0)$ and $(y, j - i)$ in the frame graph $G_{\rho(i)}$. The annotations for future obligations are added similarly. Fig. 1 shows the graph representation of a symbolic model ρ of X-length 3, and a model it admits. By a *path* p in G_ρ we will mean as usual a (finite or infinite) sequence of vertices $v_0, v_1 \ldots$ in G_ρ such that each v_i, v_{i+1} is connected by an edge in G_ρ. We call p a *forward* path if each v_{i+1} is at a level strictly greater than v_i.

3.2 Automata for Symbolic Models

In order to check whether a CLTL$^\Diamond$ formula is satisfiable we use Lemma 2 below based on the approach developed in [DD07].

Lemma 2. *A* CLTL$^\Diamond$ *formula ϕ of* X*-length l is satisfiable iff there exists a symbolic model ρ of* X*-length l such that $\rho \models_{\text{symb}} \phi$ and ρ admits a model.*

In order to take advantage of Lemma 2, we use the automaton-based approach from [VW94]. We build an automaton $\mathcal{A}_\phi$ as the intersection of two automata $\mathcal{A}_{\text{symb}}$ and $\mathcal{A}_{\text{sat}}$ such that the language recognized by $\mathcal{A}_{\text{symb}}$ is the set of symbolic models satisfying ϕ and the language recognized by $\mathcal{A}_{\text{sat}}$ is the set of symbolic models that are realized by some models.

We define the automaton $\mathcal{A}_{\text{symb}}$ by adapting the construction from [VW94] for LTL. We define $cl(\phi)$ the closure of ϕ as usual, and an atom of ϕ is a maximally consistent subset of $cl(\phi)$. For the infinitary case, $\mathcal{A}_{\text{symb}}$ is the generalized Büchi automaton $(Q, Q_0, \rightarrow, F)$ such that:

- Q is the set of atoms of ϕ and $Q_0 = \{At \in Q \mid \phi \in At,\ \mathsf{X}^{-1}\top \notin At\}$,
- $At \xrightarrow{fr} At'$ iff
  (atomic constraints) for every atomic formula φ in At, $\varphi \in fr$,
  (one step) for every $\mathsf{X}\psi \in cl(\phi)$, $\mathsf{X}\psi \in At$ iff $\psi \in At'$, and for every $\mathsf{X}^{-1}\psi \in cl(\phi)$, $\psi \in At$ iff $\mathsf{X}^{-1}\psi \in At'$
- let $\{\psi_1 \mathsf{U} \phi_1, \ldots, \psi_r \mathsf{U} \phi_r\}$ be the set of until formulae in $cl(\phi)$. We pose $F = \{F_1, \ldots, F_r\}$ where for every $i \in \{1, \ldots, r\}$, $F_i = \{At \in Q : \psi_i \mathsf{U} \phi_i \notin At$ or $\phi_i \in At\}$.

For the finitary case, the finite-state automaton $\mathcal{A}_{\text{symb}}$ accepting finite words is defined as above except that F is a set of states At such that no atomic formula of the form $x = \Diamond y$ occurs in At and no formula of the form either $\mathsf{X}\phi$ or $x = \mathsf{X}^i y$ with $i > 0$ occurs in At. Moreover, such final states can only be reached when the frame labelling the last transition contains proper information about the end of the model.

In the next section, we explain how one can build the automaton $\mathcal{A}_{\text{sat}}$ that recognizes the set of satisfiable symbolic models. Since $\mathcal{A}_\phi$ is the automaton recognizing the intersection of the languages accepted by $\mathcal{A}_{\text{symb}}$ and $\mathcal{A}_{\text{sat}}$, the following result is a direct consequence of Lemma 2.

Theorem 2. *A* CLTL$^\Diamond$ *formula ϕ is satisfiable iff the language recognized by $\mathcal{A}_\phi$ is nonempty.*

Note that we separate the temporal logic part and the constraint part by defining two different automata. This allows to extend the decidability results to any extension of LTL that induces an ω-regular class of models. We only need to change the definition of $\mathcal{A}_{\text{symb}}$.

4 Characterization of Satisfiable Symbolic Models

In order to determine whether a symbolic model ρ is "satisfiable" (i.e. it admits a model), we introduce counters that remember the satisfaction of constraints $x = \Diamond y$. If $x = \Diamond y_1 \wedge \cdots \wedge x = \Diamond y_n$ needs to be satisfied at the current position, then we shall increment a counter indexed by $\{y_1, \ldots, y_n\}$ that remembers this set of obligations. In a finite model, all the obligations need to be fulfilled before the last position whereas in an infinite model either no more unsatisfied obligations arise after a point, or they are essentially fulfilled infinitely often. The exact conditions will be spelt out soon.

4.1 Counting Sequence

For each $X \in \mathcal{P}^+(\{x_1, \ldots, x_k\})$ (set of non-empty subsets of $\{x_1, \ldots, x_k\}$), we introduce a counter that keeps track of the number of obligations that need to be satisfied by X. We identify the counters with elements of $\mathcal{P}^+(\{x_1, \ldots, x_k\})$. A counter valuation $\boldsymbol{c}$ is a map $\boldsymbol{c} : \mathcal{P}^+(\{x_1, \ldots, x_k\}) \to \mathbb{N}$. For instance, we write $\boldsymbol{c}(\{x, y\})$ to denote the value of the counter $\{x, y\}$, which will stand for the number of obligations to repeat a distinct value in x and y.

We will define a canonical sequence of counter valuations along a symbolic model. We introduce some definitions first. For an l-frame *fr* and $X \in \mathcal{P}^+(\{x_1, \ldots, x_k\})$, we define a *point of increment* for X in *fr* to be an equivalence class of the form $[(x, 0)]_{fr}$ such that $\Diamond_{fr}(x, 0) = X$ and $(x, 0)$ is not connected by a forward edge to a node in *fr* (i.e. there is no edge between $(x, 0)$ and (y, j) for any $j \in \{1, \ldots l\}$). A *point of decrement* for X in *fr* is defined to be an equivalence class of the form $[(x, l)]_{fr}$ such that $\Diamond_{fr}(x, l) \cup [(x, l)]_{fr} = X$, and (x, l) is not connected by a backward edge to another node in *fr* (i.e. there is no edge between (x, l) and (y, j) for any $j \in \{0, \ldots l-1\}$). Let u^+_{fr} denote a counter valuation which records the number of points of increment for each counter X, in *fr*. Similarly let u^-_{fr} denote the counter valuation which records the number of points of decrement for each counter X in *fr*.

Now let ρ be a symbolic model of X-length l. We carry over the notations for the set of future obligations and the equivalence class for x at level i to symbolic models as well. Thus $\Diamond_\rho(x,i)$ is equal to $\Diamond_{\rho(i)}(x,0)$ and $[(x,i)]_\rho$ is $[(x,0)]_{\rho(i)}$. For $X \in \mathcal{P}^+(\{x_1,\ldots,x_k\})$, a *point of increment* for X in ρ is an equivalence class of the form $[(x,i)]_\rho$ such that $[(x,0)]_{\rho(i)}$ is a point of increment for X in the frame $\rho(i)$. Similarly, a *point of decrement* for X in ρ is an equivalence class of the form $[(x,i)]_\rho$ such that $i \geq l+1$ and $[(x,l)]$ is a point of decrement for X in the frame $\rho(i-l)$.

We can now define a canonical counter valuation sequence α along ρ, called the *counting sequence* along ρ, which counts the number of "unsatisfied" points of increments for each counter X. Let $\dot{+}$ denote the "proper addition" of integers, defined by $n \dot{+} m = \max(0, n+m)$. We define α inductively for each $X \in \mathcal{P}^+(\{x_1,\ldots,x_k\})$ and $0 \leq i < |\rho|$ as: $\alpha(0)(X) = 0$; and $\alpha(i+1)(X) = \alpha(i)(X) \dot{+} (u^+_{\rho(i)}(X) - u^-_{\rho(i+1)}(X))$.

4.2 Sequences for Satisfiable Symbolic Models

We characterize satisfiable symbolic models using their counting sequences.

Lemma 3. *A finite symbolic model ρ is satisfiable (i.e. admits a model) iff the final value of the counting sequence α along ρ has value 0 for each counter X (i.e. $\alpha(|\rho|-1)(X) = 0$) and in the last frame fr of ρ, there are no "unsatisfied"obligations – i.e. no node (x,i) in G_{fr} and variable $y \in \Diamond_{fr}(x,i)$, with no edge between (x,i) and (y,j) for $j > i$.*

An infinite symbolic model ρ is satisfiable iff the following conditions are satisfied:

(C1) There does not exist an infinite forward path p in ρ and a counter X, such that every node in the path has future obligation X, and there is a variable y in X which is never connected by a forward edge from a node in p (i.e. no node in p is connected by a forward edge to a node of the form (y,i)).

(C2) In the counting sequence along ρ, each counter X satisfies one of the condtions:
   - *(a) there is a point after which the value of counter X is always zero and after which we never see a point of increment for X,*
   - *(b) infinitely often we see a point of decrement for X of the form $[(x,i)]$ with $\Diamond_\rho(x,i) \subset X$, or,*
   - *(c) for each $x \in X$, we infinitely often see a point of decrement for X, which is connected by a forward path to an x node (i.e a node of the form (x,i)).*

In Sect. 5 we show that we can check these conditions on counting sequences using counter automata with a decidable nonemptiness problem.

5 Decidability

We introduce a class of counter automata with a disjunctive variant of generalized Büchi acceptance condition in which, along any run, a zero test is performed at most once for each counter.

5.1 Simple Counter Automata

A simple counter automaton $\mathcal{A}$ is a tuple $\langle \Sigma, C, Q, \mathcal{F}, I, \rightarrow \rangle$ such that

- Σ is a finite alphabet, C is a finite set of counters,
- Q is a finite set of locations, $I \subseteq Q$ is the set of initial locations,
- $\mathcal{F} = \{F_0, F_1, \ldots, F_K\}$ for some $K \geq 0$ where $F_i \subseteq \mathcal{P}(Q)$ for each $i = 1 \ldots K$,
- $\rightarrow$ is a finite subset of $Q \times \mathcal{P}(C) \times \mathbb{Z}^C \times \Sigma \times Q$.

Elements of $\rightarrow$ are also denoted by $q \xrightarrow{Y,up,a} q'$ where Y is interpreted as zero tests on all counters in Y. A configuration $\langle q, \boldsymbol{c} \rangle$ is an element of $Q \times \mathbb{N}^C$ and, $\langle q, \boldsymbol{c} \rangle \rightarrow \langle q', \boldsymbol{c}' \rangle$ iff there is a transition $q \xrightarrow{Y,up,a} q'$ in $\mathcal{A}$ s.t. for $c \in Y$, $\boldsymbol{c}(c) = 0$ and for $c \in C$, $\boldsymbol{c}'(c) = \boldsymbol{c}(c) + up(c)$. As usual, a run is a sequence of configurations ruled by the transitions of $\mathcal{A}$. An infinite run is accepting iff there exists a set $F \in \mathcal{F}$ such that every set $Y \in F$ is visited infinitely often. Elements of Σ^ω labeling accepting runs define the language accepted by $\mathcal{A}$. In order to accept finite words, we suppose that $\mathcal{F}$ defines a single set of final states and a finite run is accepting iff it ends at a final state with all the counters equal to zero.

However, we require additional conditions on the control graph of $\mathcal{A}$ to be declared as simple. We require that there is a partition $\{Q_0, \ldots, Q_K\}$ of Q and corresponding sets of counters $C_0, C_1, \ldots, C_K$ with $C_0 = \emptyset$ such that $I \subseteq Q_0$ and for $i \in \{1, \ldots, K\}$, a transition from a location in Q_0 to a location in Q_i can be fired only if the counters of C_i are equal to zero and all the transitions from a location in Q_i go to another location of Q_i. Moreover every transition from a location of Q_i does not modify the value of the counters in C_i. As a consequence, when we enter in the component made of the locations of Q_i the counters in C_i are equal to zero forever. Finally, for $i \in \{0, \ldots, K\}$, $F_i \subseteq \mathcal{P}(Q_i)$. Let us summarize the conditions:

1. $Q = Q_0 \uplus \cdots \uplus Q_K$ and $I \subseteq Q_0$,
2. $\mathcal{F} = \{F_0, F_1, \ldots, F_K\}$ where each $F_i \subseteq \mathcal{P}(Q_i)$,
3. there exist $K + 1$ sets of counters $C_0, \ldots, C_K \subseteq C$ with $C_0 = \emptyset$ such that the transition relation $\rightarrow \subseteq Q \times \mathcal{P}(C) \times \mathbb{Z}^C \times \Sigma \times Q$ verifies the conditions below: for all $i, i' \in \{0, \ldots, K\}$, $q \in Q_i$ and $q' \in Q_{i'}$, the transitions from q to q' are of the form $q \xrightarrow{Y,up,a} q'$ where
   (a) $i \neq i'$ implies $i = 0$ and $Y = C_{i'}$, (b) $i = i'$ implies $Y = \emptyset$,
   (c) for $c \in C_{i'}$, $up(c) = 0$.

In the sequel we consider simple counter automata with $C = \mathcal{P}^+(\{x_1, \ldots, x_k\})$, $K = 2^k - 1$ and each set F_i contains sets of states reached by decrementing the counters in C_i. Lemma 4 below states that simplicity implies decidability of the nonemptiness problem thanks to [Jan90].

Lemma 4. *The nonemptiness problem for simple counter automata is decidable.*

5.2 Automata Recognizing Satisfiable Counting Sequences

Now, we can build a simple counter automaton $\mathcal{A}_k^l$ recognizing the set of satisfiable symbolic models of X-length l. We describe the construction for the infinite case and the automaton that recognizes finite satisfiable symbolic models can be defined similarly.

The simple counter automaton $\mathcal{A}_k^l$ is defined to be the intersection of the automata $\mathcal{A}^1$ and $\mathcal{A}^2$ which check conditions (C1) and (C2) respectively. Automaton $\mathcal{A}^1$ is a Büchi automaton and is easy to define. We focus on defining the counter automaton $\mathcal{A}^2$. We define $\mathcal{A}^2 = \langle \Sigma, C, Q, F, s, \rightarrow \rangle$, where $\Sigma = \mathtt{Frame}_k^l$, $C = \mathcal{P}^+(\{x_1, \ldots, x_k\})$, $Q = \{s\} \cup \mathtt{Frame}_k^l \cup \bigcup_{Z \subseteq C} Q_{\mathcal{A}_Z}$, where $Q_{\mathcal{A}_Z}$ is the set of states of the automaton $\mathcal{A}_Z$ which we define below, $\rightarrow$ is given by

- $s \xrightarrow{\emptyset, up_0, fr} fr$
- $fr \xrightarrow{\emptyset, up, fr'} fr'$
- $fr \xrightarrow{Z, up, fr'} s_{\mathcal{A}_Z}$

where up_0 is the zero update (i.e. $up_0(X) = 0$ for each $X \in C$), $u_{fr}^+(X) \leq up(X) \leq u_{fr}^+(X) - u_{fr'}^-(X)$ for each $X \in C$, and $s_{\mathcal{A}_Z}$ is the start state of automaton $\mathcal{A}_Z$. Moreover, we require in the last rule that for every counter $X \in Z$ (i.e. every counter that is tested to zero) we have $up(X) = 0$.

The Büchi automaton $\mathcal{A}_Z$ is given by:

$$\mathcal{A}_Z = \mathcal{A}_Z^{2a} \cap \bigcup_{X \in C \setminus Z} (\mathcal{A}_X^{2b} \cup \mathcal{A}_X^{2c}) \qquad \mathcal{A}_X^{2c} = \bigcup_{x \in X} \mathcal{A}_{X,x}$$

where the automaton $\mathcal{A}_Z^{2a}$ accepts symbolic models in which there are no points of increment for any X in Z; the automaton $\mathcal{A}_X^{2b}$ checks that infinitely often there is a point of decrement for X of the form $[(x, i)]$ such that the set of future obligations of (x, i) is a strict subset of X; and the automaton $\mathcal{A}_{X,x}$ checks condition C2(c) for X and a variable $x \in X$. The automaton $\mathcal{A}_{X,x}$ is the complement of the Büchi automaton $\mathcal{B}_{X,x}$ which accepts symbolic models in which there is a point after which we never see an x-node reachable by a forward path from a point of decrement for X. The automaton $\mathcal{B}_{X,x}$ has states of the form (fr, S) where fr is a frame and S is a subset of nodes in fr. We have a transition from (fr, S) to (fr', S') iff S' is the set of nodes in fr' which are either a point of decrement for X in fr' or are connected by a forward edge to a node in S in fr'. The automaton non-deterministically moves to a second copy where it allows the above transitions only if S' does not contain a node of the form (x, i). All states in the second copy are final.

We can easily check that all the properties of simple counter automata are verified by this construction. For the finite case, $\mathcal{A}_k^l$ is similar to $\mathcal{A}^2$ above, except that a word is accepted when the run ends with all the counters equal to zero.

Lemma 5. *Let ρ be a symbolic model of X-length l. Then ρ is accepted by $\mathcal{A}_k^l$ iff ρ is satisfiable.*

We are now in position to state the main result of the paper.

Theorem 3. *Finitary and infinitary satisfiability for* $\mathrm{CLTL}^{\Diamond}$ *is decidable.*

Proof. Let ϕ be a $\mathrm{CLTL}^{\Diamond}$ formula over k variables with X-length l. Let $\mathcal{A}_\phi$ be the simple counter automaton built as the intersection of $\mathcal{A}_{\mathrm{symb}}$, $\mathcal{A}_k^l$ and $\mathcal{A}_{lc}$ that accepts sequences in which consecutive frames are one-step consistent. We can check whether the language accepted by $\mathcal{A}_\phi$ is non-empty (Lemma 4). Thus, by Theorem 2, checking the satisfiability of ϕ is decidable.

According to the acceptance condition of $\mathcal{A}_k^l$ in the finite case, finitary satisfiability reduces to the reachability problem in Petri nets. $\square$

Theorems 2 and 3 entail that this decidability result can be extended to any extension of LTL as soon as the temporal operators as definable in MSOL, see for instance [GK03].

Corollary 1. *Finitary and infinitary satisfiability for* $\mathrm{CLTL}^{\Diamond}$ *augmented with MSOL definable temporal operators is decidable.*

5.3 A PSPACE Fragment of $\mathrm{CLTL}^{\Diamond}$

In this section, we consider the fragment $\mathrm{CLTL}_1^{\Diamond}$ with a unique variable x. The models are sequences of natural numbers and the only counter in counting sequences α is $\{x\}$ (we identify $\alpha(i)(\{x\})$ with $\alpha(i)$). Given a symbolic model ρ over the alphabet $\mathtt{Frame}_1^l$ and the counting sequence α along ρ, for every $0 \leq i < |\rho|$, $\alpha(i+1) = \alpha(i) \dot{+} u_i^+ - u_{i+1}^-$ with $u_i^+, u_{i+1}^- \in \{0,1\}$. By Lemma 3, when ρ is satisfiable, in the counting sequence along ρ either the unique counter remains equal to zero after a finite number of steps or it is decremented infinitely often. Moreover, the value of the unique counter in the counting sequence is nicely bounded unlike in general with strictly more than one variable.

Lemma 6. *Let* ρ *be a symbolic model and* α *be the counting sequence along* ρ. *For* $0 \leq i < |\rho|$, $\alpha(i) \leq l$.

Boundedness entails the possibility to use automata without counters.

Lemma 7. *The set of satisfiable symbolic models over the alphabet* $\mathtt{Frame}_1^l$ *can be recognized by a standard Büchi automaton* $\mathcal{A}_1^l$ *for the infinite case, or by a finite-state automaton for the finite case.*

The automaton $\mathcal{A}_1^l$ has an exponential size and can be built in polynomial space in l. Checking nonemptiness for this automaton can be done in non deterministic logarithmic space which allows to establish Theorem 4 below.

Theorem 4. *Finitary and infinitary satisfiability for* $\mathrm{CLTL}_1^{\Diamond}$ *is* PSPACE-*complete.*

The models for $\mathrm{CLTL}_1^{\Diamond}$ corresponds to models of $\mathrm{LTL}^{\downarrow}(\sim, \mathsf{X}, \mathsf{U})$. Therefore, finitary and infinitary satisfiability for $\mathrm{LTL}_1^{\downarrow}(\sim, \mathsf{X}, \mathsf{X}^{-1}, \mathsf{U}, \mathsf{S})$ restricted to formulae such that the freeze operator is restricted to subformulae of the form $\downarrow_{r=x} \mathsf{XF}(r = x)$ and $\downarrow_{r=x} \mathsf{X}^i(r = x)$ is decidable in polynomial space (r is the unique register and x the unique variable).

5.4 Repeating Values in the Past Is Still Decidable

In this section we explain why we can allow the constraints of the language to state properties about past repetitions of a value without loosing decidability. Let $\mathrm{CLTL}^{\Diamond,\Diamond^{-1}}$ be the extension of $\mathrm{CLTL}^{\Diamond}$ with atomic formulae of the form $x = \Diamond^{-1}y$. The satisfaction relation is extended as follows: $\sigma, i \models x = \Diamond^{-1}y$ iff there is $j > 0$ s.t. $x = \sigma(i-j)(y)$ and $0 \leq i - j$. Similarly to what is done in Section 2.1, x *diff* $\Diamond^{-1}y$ can be omitted since it can be defined from $x = \Diamond^{-1}y$ (a variant of the equivalence (1)).

In order to deal with satisfiability for $\mathrm{CLTL}^{\Diamond,\Diamond^{-1}}$ we need to extend the symbolic representation of models. In addition of the conditions(F1)–(F5) defined in Sect. 3.1, a frame *fr* has to verify the following property (the finitary case admits a similar update):

(F6) for all $i, j \in \{0, \dots, l\}$ and $x, y \in \{x_1, \dots x_k\}$, if $\mathsf{X}^i x = \mathsf{X}^j y$ is in *fr* then
- if $i = j$, then for every $z \in \{x_1, \dots, x_k\}$ we have $\mathsf{X}^i(x = \Diamond^{-1}z) \in \mathit{fr}$ iff $\mathsf{X}^j(y = \Diamond^{-1}z) \in \mathit{fr}$ (we extend the notion of frames);
- if $i > j$ then $\mathsf{X}^i(x = \Diamond^{-1}y) \in \mathit{fr}$, and for every $z \in \{x_1, \dots, x_k\}$, $\mathsf{X}^i(x = \Diamond^{-1}z) \in \mathit{fr}$ iff either $\mathsf{X}^j(y = \Diamond^{-1}z) \in \mathit{fr}$ or there exists $i > j' \geq j$ such that $\mathsf{X}^i x = \mathsf{X}^{j'} z$ is in *fr*.

We pose $\Diamond^{-}_{\mathit{fr}}(\mathsf{X}^i x) \stackrel{\text{def}}{=} \{y \mid \mathsf{X}^i(x = \Diamond^{-}y) \in \mathit{fr}\}$. Since we need to deal with past obligations, a counter is a pair $\langle X_p, X_f \rangle$ in $\mathcal{P}^+(\{x_1, \dots, x_k\}) \times \mathcal{P}^+(\{x_1, \dots, x_k\})$ where X_p is for past obligations and X_f for future obligations. We update the notion of counter valuations accordingly. A value n for $\langle X_p, X_f \rangle$ is the number of values that occurred in a past state of every variable of X_p and that have to be repeated in a future state of every variable in X_f.

We extend some earlier definitions. For an l-frame *fr* and counter $\langle X_p, X_f \rangle$, we define a *point of increment* for $\langle X_p, X_f \rangle$ in *fr* to be an equivalence class of the form $[(x,0)]_{\mathit{fr}}$ such that $\Diamond_{\mathit{fr}}(x,0) = X_f$, $(x,0)$ is not connected by a forward edge to a node in *fr and* $[(x,0)]_{\mathit{fr}} \cup \Diamond^{-1}_{\mathit{fr}}(x,0) = X_p$. A *point of decrement* for $\langle X_p, X_f \rangle$ in *fr* is defined to be an equivalence class of the form $[(x,l)]_{\mathit{fr}}$ such that $\Diamond_{\mathit{fr}}(x,l) \cup [(x,l)]_{\mathit{fr}} = X_f$, (x,l) is not connected by a backward edge to another node in *fr and* $\Diamond^{-1}(x,l) = X_p$. Let u^{+}_{fr} denote a counter valuation which records the number of points of increment for each counter $\langle X_p, X_f \rangle$, in *fr*. Similarly let u^{-}_{fr} denote the counter valuation which records the number of points of decrement for each counter $\langle X_p, X_f \rangle$ in *fr*. We can now define a canonical counter valuation sequence α along ρ, called the *counting sequence* along ρ, which counts the number of "unsatisfied" points of increments for each counter $\langle X_p, X_f \rangle$ with $X_p \neq \emptyset$. We define α inductively: $\alpha(0)(\langle X_p, X_f \rangle) = 0$ and for $0 \leq i < |\rho|$, $\alpha(i+1)(\langle X_p, X_f \rangle) = \alpha(i)(\langle X_p, X_f \rangle) + (u^{+}_{\rho(i)}(\langle X_p, X_f \rangle) - u^{-}_{\rho(i+1)}(\langle X_p, X_f \rangle))$. Note that decrementations are this time compulsory and we allow $\alpha(i)(\langle X_p, X_f \rangle)$ to be negative (but not in the acceptance condition). Though we need more counters, dealing with past repeating values, does not introduce real complications. This is analogous to the passage from LTL to LTL with past-time operators since past is finite and information about past can be accumulated smoothly.

Lemma 8. *A symbolic model ρ for the logic* $\mathrm{CLTL}^{\Diamond,\Diamond^{-1}}$ *is satisfiable iff the counting sequence along ρ satisfies the conditions from Lemma 3 for the future part of the counters and for $0 \leq i < |\rho|$, $\alpha(i)(\langle X_p, X_f\rangle) \geq 0$.*

The proof is similar to the proof of Lemma 3. We just need more registers to store the different values that have to be repeated.

As a consequence, we can easily update the construction of $\mathcal{A}_\phi$ in order to deal with past repeating values. The definition of the automata $\mathcal{A}_{\text{symb}}$ and $\mathcal{A}_k^l$ are just extended by considering the new definition for frames. It is also important to observe that the automaton $\mathcal{A}_\phi$ obtained by synchronization of these automata still belongs to the class of simple counter automata and the decidability result also holds for $\mathrm{CLTL}^{\Diamond,\Diamond^{-1}}$ satisfiability problem.

Theorem 5. *Finitary and infinitary satisfiability for* $\mathrm{CLTL}^{\Diamond,\Diamond^{-1}}$ *[resp.* $\mathrm{CLTL}_1^{\Diamond,\Diamond^{-1}}$*] is decidable [resp.* PSPACE*-complete].*

6 Concluding Remarks

We have shown that satisfiability for $\mathrm{CLTL}^{\Diamond}$ with operators in $\{\mathsf{X}, \mathsf{X}^{-1}, \mathsf{S}, \mathsf{U}\}$ is decidable by reduction into the verification of fairness properties in Petri nets [Jan90]. The proof is uniform for the finitary and infinitary cases and it can be extended to atomic constraints of the form $x = \Diamond^{-1} y$ and to any set of MSOL definable temporal operators. Moreover, satisfiability for $\mathrm{CLTL}^{\Diamond}$ restricted to one variable is shown PSPACE-complete. Hence, we have defined and studied a well-designed decidable fragment of LTL with the freeze quantifier answering some question from [WZ00] and circumventing some undecidability results from [DL06]. Finally, as done also in [DL06, Laz06, BMS$^+$06], we show relationships between fragments of LTL with freeze and counter automata.

The main question left open by our work is the complexity of satisfiability for $\mathrm{CLTL}^{\Diamond}$ and more precisely we do not know whether $\mathrm{CLTL}^{\Diamond}$ satisfiability has elementary complexity. Similarly, are there natural fragments of $\mathrm{CLTL}^{\Diamond}$ that are of lower complexity, for instance the one involved in Theorem 1? Another promising extension consists in considering other concrete domains as $\langle \mathbb{R}, <, = \rangle$ and to allow atomic formulae of the form $x < \Diamond y$. The decidability status of such a variant is still open.

Acknowledgements. We are grateful to Petr Jančar (TU Ostrava) for pointing us to [Jan90] in order to solve the nonemptiness problem for simple counter automata and for suggesting the proof of Lemma 4 and Ranko Lazić (U. of Warwick) for remarks about a preliminary version.

References

[ABM01] C. Areces, P. Blackburn, and M. Marx. Hybrid logics: characterization, interpolation and complexity. *JSL*, 66(3):977–1010, 2001.

[AH94] R. Alur and Th. Henzinger. A really temporal logic. *JACM*, 41(1):181–204, 1994.

[BMS$^+$06] M. Bojańczyk, A. Muscholl, Th. Schwentick, L. Segoufin, and C. David. Two-variable logic on words with data. In *LICS'06*, pages 7–16. IEEE, 2006.

[DD07] S. Demri and D. D'Souza. An automata-theoretic approach to constraint LTL. *I & C*, 205(3):380–415, 2007.

[DDG07] S. Demri, D. D'Souza, and R. Gascon. A decidable temporal logic of repeating values. Technical report, LSV, 2007.

[DL06] S. Demri and R. Lazić. LTL with the freeze quantifier and register automata. In *LICS*, pages 17–26. IEEE, 2006.

[DLN07] S. Demri, R. Lazić, and D. Nowak. On the freeze quantifier in constraint LTL: decidability and complexity. *I & C*, 205(1):2–24, 2007.

[Esp94] J. Esparza. On the decidability of model checking for several μ-calculi and Petri nets. In *CAAP'94*, volume 787 of *LNCS*, pages 115–129. Springer, 1994.

[Fit02] M. Fitting. Modal logic between propositional and first-order. *JLC*, 12(6):1017–1026, 2002.

[GK03] P. Gastin and D. Kuske. Satisfiability and model checking for MSO-definable temporal logics are in PSPACE. In *CONCUR'03*, volume 2761 of *LNCS*, pages 222–236. Springer, 2003.

[Gor96] V. Goranko. Hierarchies of modal and temporal logics with references pointers. *Journal of Logic, Language, and Information*, 5:1–24, 1996.

[Hen90] Th. Henzinger. Half-order modal logic: how to prove real-time properties. In *PODC'90*, pages 281–296. ACM, 1990.

[HR89] R.R. Howell and L.E. Rosier. Problems concerning fairness and temporal logic for conflict-free Petri nets. *TCS*, 64:305–329, 1989.

[Jan90] P. Jančar. Decidability of a temporal logic problem for Petri nets. *TCS*, 74(1):71–93, 1990.

[KV06] O. Kupferman and M. Vardi. Memoryful Branching-Time Logic. In *LICS'06*, pages 265–274. IEEE, 2006.

[Laz06] R. Lazić. Safely freezing LTL. In *FST&TCS'06*, volume 4337, pages 381–392. LNCS, 2006.

[LMS02] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Temporal logic with forgettable past. In *LICS'02*, pages 383–392. IEEE, 2002.

[LP05] A. Lisitsa and I. Potapov. Temporal logic with predicate λ-abstraction. In *TIME'05*, pages 147–155. IEEE, 2005.

[Rab06] A. Rabinovich. Decidability and expressive power of real time logics. In *FORMATS'06*, volume 4202 of *LNCS*, page 32. Springer, 2006. Invited talk.

[Seg06] L. Segoufin. Automata and logics for words and trees over an infinite alphabet. In *CSL'06*, volume 4207 of *LNCS*, pages 41–57. Springer, 2006.

[tCF05] B. ten Cate and M. Franceschet. On the complexity of hybrid logics with binders. In *CSL'05*, volume 3634 of *LNCS*, pages 339–354. Springer, 2005.

[VW94] M. Vardi and P. Wolper. Reasoning about infinite computations. *I & C*, 115:1–37, 1994.

[WZ00] F. Wolter and M. Zakharyaschev. Spatio-temporal representation and reasoning based on RCC-8. In *KR'00*, pages 3–14. Morgan Kaufmann, 2000.

Model Checking Knowledge and Linear Time: PSPACE Cases

Kai Engelhardt[⋆], Peter Gammie, and Ron van der Meyden[⋆]

School of Computer Science and Engineering, The University of New South Wales, and National ICT Australia[⋆⋆], Sydney, NSW 2052, Australia
{kaie|peteg|meyden}@cse.unsw.edu.au

Abstract. We present a general algorithm scheme for model checking logics of knowledge, common knowledge and linear time, based on bisimulations to a class of structures that capture the way that agents update their knowledge. We show that the scheme leads to PSPACE implementations of model checking the logic of knowledge and linear time in several special cases: perfect recall systems with a single agent or in which all communication is by synchronous broadcast, and systems in which knowledge is interpreted using either the agents' current observation only or its current observation and clock value. In all these results, common knowledge operators may be included in the language. Matching lower bounds are provided, and it is shown that although the complexity bound matches the PSPACE complexity of the linear time temporal logic LTL, as a function of the model size the problems considered have a higher complexity than LTL.

1 Introduction

The logic of knowledge [5] has been proposed as a formalism to express information theoretic properties in distributed and multi-agent systems, and has been shown to be useful for the analysis of distributed systems protocols [9], information flow security properties [10, 20, 24], as well as for problems such as diagnosis and recoverability [3, 4].

The semantics for knowledge operators can be defined in a variety of ways, depending on what information agents use when computing what they know. At one extreme (the "observational semantics") agents rely only on their current observation, at the other (the "synchronous perfect recall semantics") agents rely on the log of all their past observations. In between lies a "clock semantics" in which agents rely on their current observation plus a clock value. These semantics have different motivations: the perfect recall semantics is most appropriate for security analyses and derivation of protocols that make optimal use of information; the other semantics are closer to system implementations.

⋆ Work was partially supported by ARC Discovery Grant RM02036.

⋆⋆ National ICT Australia is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council. The third author thanks the Courant Institute, New York University for their hospitality in hosting a sabbatical visit during which this work was conducted.

A number of model checkers for the logic of knowledge have been recently developed, which embody different choices of semantics for the knowledge operators and different types of expressiveness for the temporal dynamics. MCMAS [13] deals with the observational interpretation of knowledge and the branching time logic CTL. DEMO [27] deals with the dynamic logic based "update logic" [1], which handles what is in effect the perfect recall semantics for knowledge. The system MCK [7] covers a broad spectrum of definitions of knowledge (observational, clock, perfect recall), as well as dealing with both linear time and branching time temporal logic.

Where they deal with the perfect recall semantics for knowledge, these systems place severe constraints on the interaction between knowledge and temporal operators, for reasons of inherent complexity. The complexity of model checking the combination of the linear time temporal logic LTL with knowledge operators interpreted according to the perfect recall semantics has been studied by van der Meyden and Shilov [23], who show that this problem is decidable but with a nonelementary lower bound, and undecidable when operators for common knowledge (a type of fixpoint over knowledge operators) are added to the language. (Shilov et al [16, 17, 8] have also studied branching time versions of these results.)

However, as we show in this paper, this general result does not preclude the existence of special cases in which this model checking problem has lower complexity, even when common knowledge operators are included in the language. We identify a number of cases where the problem (including common knowledge) is solvable in PSPACE. These include systems with a single agent (discussed in Section 5.1) and systems in which all communication is by synchronous broadcast (treated in Section 5.2). The result concerning a single agent improves the nonelementary upper bound for the single agent case obtained from the algorithm of van der Meyden and Shilov.

Our approach to the proof of these results is by means of a general algorithm scheme (presented in Section 4) that relies upon the existence of a bisimulation from the (in effect, infinite) systems being checked to a finite structure that represents the way that agents update their knowledge in the system. In addition to the results about the perfect recall semantics, we show that this scheme can be used to obtain PSPACE complexity results for model checking the logic of knowledge and linear time for other interpretations for knowledge: in particular, we show that this complexity bound applies in the case of both the clock semantics and the observational semantics (see Section 6).

As the complexity of model checking the linear time temporal logic LTL alone is already PSPACE-complete, it may seem from these results that the extra expressiveness of the logic of knowledge in these cases comes at no extra cost. In fact, we show that there is a sense in which these model checking problems are harder than model checking LTL alone, by focussing on the complexity of model checking a fixed formula as a function of the size of the model. For LTL, this "model complexity" is linear-time for each formula [11]. We show that the model complexity can be as high as PSPACE-complete once the formula includes knowledge operators.

2 Basic Definitions

In this section we define the semantic framework with respect to which we study the model checking problem. The definitions closely follow [23], which dealt with model checking knowledge and linear time in multi-agent systems for a "perfect recall" interpretation of knowledge. We also define an alternate "clock" interpretation of knowledge, in which agents reason on the basis of their current observation and knowledge of the time.

Let *Prop* be a set of atomic propositional constants, $n > 0$ be a natural number, and let $\mathbb{A} = \{1, \ldots, n\}$ be a set of agents. We will be concerned with model checking a propositional multi-modal language for knowledge and linear time based on the set *Prop* of atomic propositional constants, with formulae generated by the modalities $\bigcirc$ (next), $\mathcal{U}$ (until), a knowledge operator K_i for each agent $i \in \mathbb{A}$, and a common knowledge operator C_G for each group of agents $G \subseteq \mathbb{A}$. Formulae of the language are defined as follows: each atomic propositional constant $p \in \mathit{Prop}$ is a formula, and if φ and ψ are formulae, then so are $\neg\varphi$, $\varphi \wedge \psi$, $\bigcirc\varphi$, $\varphi\,\mathcal{U}\,\psi$, $K_i\varphi$ and $C_G\varphi$ for each $i \in \mathbb{A}$ and group $G \subseteq \mathbb{A}$. We write $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ for the set of formulae. We will refer to sublanguages of this language by a similar expression that lists the operators generating the language. For example, $\mathcal{L}_{\{\bigcirc,\mathcal{U},K\}}$ refers to the sublanguage with just a single agent (in which case we may drop the subscript on the knowledge operator). As usual in temporal logic, we use the abbreviations $\Diamond\varphi$ for $\mathbf{true}\mathcal{U}\varphi$, and $\Box\varphi$ for $\neg\Diamond\neg\varphi$. The *knowledge depth* of a formula φ, denoted $\mathit{depth}(\varphi)$, is defined to be the maximal depth of nesting of K operators in φ. For example, $\mathit{depth}(K(p \wedge \neg Kq)) = 2$.

The semantics of this language is defined with respect to the following class of structures. Define an *interpreted environment (for* $\mathbb{A}$*)* to be a tuple E of the form $(S, I, \rightarrow, (O_i)_{i\in\mathbb{A}}, \pi, \alpha)$ where the components are as follows:

1. S is a set of *states* of the environment,
2. I is a subset of S, representing the possible *initial states*,
3. $\rightarrow \subseteq S^2$ is a *transition relation*,
4. for each $i \in \mathbb{A}$ the component $O_i : S \longrightarrow \mathcal{O}$, where $\mathcal{O}$ is a set of uninterpreted observations, is called the *observation function of agent* i,
5. $\pi : S \longrightarrow \mathcal{P}(\mathit{Prop})$ is an *interpretation*,
6. $\alpha \subseteq S$ is an *acceptance condition.*

Intuitively, an environment is a transition system where states encode values of local variables, messages in transit, failure of components, etc. For states s, s' the relation $s \rightarrow s'$ means that if the system is in state s, then at the next tick of the clock it could be in state s'. We call E finite whenever S is. If s is a state and i an agent then $O_i(s)$ represents the observation agent i makes when the system is in state s, i.e., the information about the state that is accessible to the agent. The interpretation π maps a state s to the set of propositional constants in *Prop* that hold at s. The acceptance conditions are essentially Büchi conditions which model fairness requirements on evolutions of the environment.

A *path* p of E from a state s in S is a finite or infinite sequence of states $s_0 s_1 \ldots$ such that $s_0 = s$ and $s_j \rightarrow s_{j+1}$ for all j. We write $p(m)$ for s_m when m is an index of p. A path p is said to be *initialized* if $p(0) \in I$. We call an initialized finite path a *trace*. A path p is *fair* if it is infinite and $p(i) \in \alpha$ for infinitely many i. Note that we do not assume that S is finite, but when so, this formulation is equivalent to the usual formulation of acceptance for Büchi automata: some $s \in \alpha$ occurs infinitely often. We say that the acceptance condition of E is *trivial* if $\alpha = S$. We assume that environments satisfy the following well-formedness condition: for every state s, there exists a fair path with initial state s. A *run* of E is a fair, initialized path, and we write $r[0..m]$ for the trace that is the prefix of run r up to time m. Let *runs*(E) be the set of all runs of E. A *point* of E is a pair (r, m), where r is a run of E and m a natural number. Intuitively, a point identifies a particular moment in time along the history described by the run.

Individual runs of an environment provide sufficient structure for the interpretation of formulae of linear temporal logic. To interpret formulae involving knowledge, we use the agents' observations to determine the points they consider possible. There are many ways one could do this. The particular approaches used in this paper model a *synchronous perfect-recall*, an *observational*, and a *clock* semantics of knowledge, each defined using a notion of local state. We define the *synchronous perfect recall local state of agent i at a point* (r, m) to be the sequence[1] $\{(r, m)\}_i^{\texttt{pr}} = O_i(r[0..m])$. That is, the synchronous perfect recall local state of an agent at a point in a run consists of a complete record of the observations the agent has made up to that point. The *clock local state of agent i at a point* (r, m) is defined by $\{(r, m)\}_i^{\texttt{clk}} = (m, O_i(r(m)))$. That is, in this definition, the agent's local state is taken to be the current time, together with the agent's current observation. Finally, the *observational local state of agent i at a point* (r, m) is $\{(r, m)\}_i^{\texttt{obs}} = O_i(r(m))$. Effectively, an agent with this view of the world considers any reachable state giving the same observation to be possible. To distinguish these local state assignments, we define a *view* v to be one of the three possibilities `pr`, `clk`, and `obs`.

Given a view v, the corresponding local state assignment may be used to define for each agent i a relation $\overset{v}{\sim}_i$ of *indistinguishability* on points $(r, m), (r', m')$ of E, by $(r, m) \overset{v}{\sim}_i (r', m')$ if $\{(r, m)\}_i^v = \{(r', m')\}_i^v$. Intuitively, when $(r, m) \overset{v}{\sim}_i (r', m')$, agent i's local state according to the view v does not contain enough information for the agent to determine whether it is at one point or the other. Clearly, each $\overset{v}{\sim}_i$ is an equivalence relation. Both the synchronous perfect recall view and the clock view are "synchronous" in the sense that if $(r, m) \overset{v}{\sim}_i (r', m')$, then we must have $m = m'$. Intuitively, this means that the agent "knows the time". The relations $\overset{v}{\sim}_i$ will be used to define the semantics of knowledge for individual agents. By $P_i^v(E, r, m)$ we denote the set $\{\, r'(m') \mid r' \in \textit{runs}(E),\ m' \in \mathbb{N},\ (r', m') \overset{v}{\sim}_i (r, m)\}$ of *possible states for agent i* at point (r, m).

To interpret the common knowledge operators, we use another relation. If $G \subseteq \mathbb{A}$ is a *group* of agents (i.e., two or more) then we define the relation $\overset{v}{\sim}_G$ on

[1] We adopt the convention that functions lift to sequences and sets in a pointwise fashion.

points to be the reflexive transitive closure of the union of all indistinguishability relations $\overset{v}{\sim}_i$ for $i \in G$, i.e., $\overset{v}{\sim}_G = (\bigcup_{i\in G} \overset{v}{\sim}_i)^*$.

The semantics of this language is defined as follows. Suppose we are given an environment E with interpretation π. We define satisfaction of a formula φ at a point (r,m) of a run of E with respect to a view v, denoted $E,(r,m) \models^v \varphi$, inductively on the structure of φ. The cases for the temporal fragment of the language are standard, and independent of v:

$E,(r,m) \models^v p$	if $p \in \pi(r(m))$, where $p \in \mathit{Prop}$,
$E,(r,m) \models^v \varphi_1 \wedge \varphi_2$	if $E,(r,m) \models^v \varphi_1$ and $E,(r,m) \models^v \varphi_2$,
$E,(r,m) \models^v \neg\varphi$	if not $E,(r,m) \models^v \varphi$,
$E,(r,m) \models^v \bigcirc\varphi$	if $E,(r,m+1) \models^v \varphi$,
$E,(r,m) \models^v \varphi_1 \,\mathcal{U}\, \varphi_2$	if there exists $m'' \geq m$ such that $E,(r,m'') \models^v \varphi_2$ and $E,(r,m') \models^v \varphi_1$ for all m' with $m \leq m' < m''$.

The semantics of the knowledge and common knowledge operators is defined by:

$E,(r,m) \models^v K_i\varphi$	if $E,(r',m') \models^v \varphi$ for all points (r',m') of E satisfying $(r',m') \overset{v}{\sim}_i (r,m)$
$E,(r,m) \models^v C_G\varphi$	if $E,(r',m') \models^v \varphi$ for all points (r',m') of E satisfying $(r',m') \overset{v}{\sim}_G (r,m)$

These definitions can be viewed as an instance of the "interpreted systems" framework for the semantics of the logic of knowledge proposed in [9]. Intuitively, an agent knows a formula to be true if this formula holds at all points that the agent is unable to distinguish from the actual point. Common knowledge may be understood as follows. For G a group of agents, define the operator E_G, read "everyone in G knows" by $E_G\varphi \equiv \bigwedge_{i\in G} K_i\varphi$. Then $C_G\varphi$ is equivalent to the infinite conjunction of the formulae $E_G^k\varphi$ for $k \geq 1$. That is, φ is common knowledge if everyone knows φ, everyone knows that everyone knows φ, etc. We refer the reader to [5] for further motivation and background.

3 Main Results

We may now define the model checking problem we consider in this paper and state our main results.

Say that formula φ is *realized* in the environment E with respect to a view v, denoted $E \models^v \varphi$, if, for all runs r of E, we have $E,(r,0) \models^v \varphi$. Our interest is in the following problem, which we call the *realization problem* with respect to a view v: given an an environment E and a formula φ of a language $\mathcal{L}$, determine if φ is realized in E with respect to v.

The realization problem for the logic of knowledge and linear time has been studied by van der Meyden and Shilov [23], who show that for the perfect recall view, the problem is undecidable for the language $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ when $n \geq 2$, and decidable for the language $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n\}}$, but with nonelementary complexity. More specifically, for $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n\}}$ their approach runs in space polynomial in $f(\mathit{depth}(\varphi), O(|E|))$, where the function f is defined by $f(0,m) = m$

and $f(k+1, m) = 2^{f(k,m)}$. It is also shown by van der Meyden and Shilov that there is a similar lower bound on the complexity when there is more than one agent.

Our main contribution in this paper is to develop a general algorithm scheme for model checking the logic of knowledge and time based on a notion of bisimulation of environments, and to show that this scheme yields improved complexity bounds in a number of special cases. The scheme itself is presented in Section 4, and parameterizes a procedure for model checking with respect to the observational view. In particular, this procedure yields the following result for the observational view.[2]

Theorem 1. *The problem of determining if a given formula in the language $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ is realized in a given environment E with respect to the observational view is decidable in PSPACE.*

By showing the existence of bisimulation from an enviroment representing the perfect recall semantics for a single agent to a suitable finite environment, we obtain the following result:

Theorem 2. *The problem of determining if a given formula in $\mathcal{L}_{\{\bigcirc,\mathcal{U},K\}}$ is realized in a given environment E with respect to the perfect recall view is in PSPACE.*

This shows that the complexity of the realization problem for formulae with a single agent with perfect recall is strictly lower than the general case, and significantly improves upon the complexity bound of van der Meyden and Shilov in this case.

By finding other suitable structures we may derive complexity bounds on several other cases of the realization problem, as stated in the following results. First, although with respect to the perfect recall view, the realization problem is non-elementary for $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n\}}$, there exist classes of environments with respect to which the problem has lower complexity, even if we add the common knowledge operators. In particular, this holds for *broadcast environments* [21]. Intuitively, these are environments in which the only communication mechanism available to agents is to broadcast to *all* agents in the system. The formal definition will be given in section 5.2. For broadcast environments we show the following.

Theorem 3. *The problem of determining if a given formula in the language $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ is realized in a given broadcast environment E with respect to the perfect recall view is decidable in PSPACE.*

Realization for the clock view may also handled using the bisimulation technique and again the common knowledge operator may be included in the language.

[2] This result does not appear to have been previously stated in the literature, but we note that results of Vardi [28] on the problem of verifying that a concrete protocol implements a knowledge-based program are very closely related. Lomuscio and Raimondi have studied the complexity of model checking the combination of the logic of knowledge with the braching time logic CTL with respect to the observational semantics [12].

Theorem 4. *The problem of determining if a given formula in the language $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ is realized in a given environment E with respect to the clock view is decidable in PSPACE.*

Note that the complexity of model checking linear time temporal logic (i.e. realization for the language $\mathcal{L}_{\{\bigcirc,\mathcal{U}\}}$) is PSPACE-complete [18]. Since $\mathcal{L}_{\{\bigcirc,\mathcal{U}\}}$ is a sublanguage of the languages in the above results, these results show that the above bounds are tight, in the sense that the problems are in fact PSPACE-complete.

That some of our complexity bounds are no more than the PSPACE complexity of the linear time temporal logic LTL may at first suggest that model checking these cases of the logic and knowledge and time could be as effective in practice as model checking LTL. However, a closer inspection indicates that it is not obvious that this will be case. The time complexity of LTL model checking a *fixed* formula is linear in the size of the model. The time complexity is exponential in the size of the formula. This exponential bound is not an impediment in practice since the formulas of interest tend to be small. The models, on the other hand, may be very large. We show that as a function of model size, the complexity of model checking fixed formulas of the logic of knowledge and time falling within our PSPACE cases can be be as high as PSPACE-hard (for $\mathcal{L}_{\{\bigcirc,\mathcal{U},K\}}$ with respect to perfect recall) and at any level of the polynomial hierarchy for the clock view.[3]

Theorem 5. *There exists a formula φ of $\mathcal{L}_{\{\bigcirc,\mathcal{U},K\}}$ such that the problem of deciding if φ is realized in a given environment E with respect to the perfect recall view is PSPACE-hard.*

In the case of the clock semantics, we may obtain the following lower bound.

Theorem 6. *For each level Π^p_k of the polynomial hierarchy, there exists a formula φ of $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n\}}$ such that the problem of deciding, given an environment E, whether $E \models^{\texttt{clk}} \varphi$, is Π^p_k-hard.*

The proof involves a reduction from Π_k formulas of quantified boolean logic. Note that this implies PSPACE-hardness of the version of the problem in which the formula is given.

4 An Algorithm Scheme

We approach the model checking problem in three stages. First, given a finite environment E and a view v, we construct an infinite environment E^v that reduces the model checking problem with respect to v in E to one of model checking E^v with respect to `obs`. Second, we introduce bisimulations between environments, which, together with the previous step, may enable the problem of model checking E with respect to v to be reduced to model checking with

[3] For reasons of space we refer the reader to a longer version of the paper available at `http//www.cse.unsw.edu.au/∼meyden` for proofs.

respect to obs in finite state environment E' (which is of exponential size in our applications). Finally, we combine alternating Turing machine techniques with standard Büchi automata techniques to obtain the general model checking procedure (which runs in PSPACE in our applications).

Let $E = (S, I, \rightarrow, (O_i)_{i \in \mathbb{A}}, \pi, \alpha)$ be a finite environment and let v be a view. Define E^v, the *v-environment for E*, to be $(S^v, I^v, \rightarrow^v, (O_i^v)_{i \in \mathbb{A}}, \pi^v, \alpha^v)$ where:

- $S^v = \mathit{runs}(E) \times \mathbb{N}$,
- $I^v = \mathit{runs}(E) \times \{0\}$,
- $(r, m) \rightarrow^v (r', m')$ if $r' = r$ and $m' = m + 1$,
- $O_i^v(r, m) = \{(r, m)\}_i^v$,
- $\pi^v(r, m) = \pi(r(m))$, and
- $(r, m) \in \alpha^v$ iff $r(m) \in \alpha$.

The following lemma states that the observational view on this (infinite) environment coincides with the view v on the original (finite) environment. Given a run r of E, write r^v for the run of E^v defined by $r^v(n) = (r, n)$ for all $n \in \mathbb{N}$.

Lemma 7. *Let $\varphi \in \mathcal{L}_{\{\bigcirc, \mathcal{U}, K_1, \ldots, K_n, C\}}$ and let (r, m) be a point of E. Then $E, (r, m) \models^v \varphi$ iff $E^v, (r^v, m) \models^{\mathsf{obs}} \varphi$.*

Since every run of E^v has the form r^v for some run of E, it follows that $E \models^v \varphi$ iff $E^v \models^{\mathsf{obs}} \varphi$.

Let $\mathit{fpaths}(E)$ be the set of all fair paths of E. For $\rho \in \mathit{fpaths}(E)$ and $m \in \mathbb{N}$ let $\rho|_m$ be the fair path with $\rho|_m(j) = \rho(m + j)$, for $j \in \mathbb{N}$.

Observe that the semantics of $E, (r, n) \models^v \varphi$ refers only to the future of the points considered in unfolding the definition. To formalise this, consider the following alternate definition of a relation $E, \rho \models^* \varphi$, defined for all $\rho \in \mathit{fpaths}(E)$, not just the initialized ones:

$E, \rho \models^* p$	if $p \in \pi(\rho(0))$, where $p \in \mathit{Prop}$,
$E, \rho \models^* \varphi_1 \wedge \varphi_2$	if $E, \rho \models^* \varphi_1$ and $E, \rho \models^* \varphi_2$,
$E, \rho \models^* \neg\varphi$	if not $E, \rho \models^* \varphi$,
$E, \rho \models^* \bigcirc\varphi$	if $E, \rho|_1 \models^* \varphi$,
$E, \rho \models^* \varphi_1 \,\mathcal{U}\, \varphi_2$	if there exists $m'' \geq 0$ such that $E, \rho|_{m''} \models^* \varphi_2$ and $E, \rho|_{m'} \models^* \varphi_1$ for all m' with $0 \leq m' < m''$.
$E, \rho \models^* K_i\varphi$	if $E, \rho' \models^* \varphi$ for all $\rho' \in \mathit{fpaths}(E)$ with $O_i(\rho'(0)) = O_i(\rho(0))$
$E, \rho \models^* C_G\varphi$	if for all sequences of states $s_0, s_1, \ldots, s_k$ such that (i) $s_0 = \rho(0)$, (ii) for all $j < k$ there exists an $i \in G$ such that $O_i(s_j) = O_i(s_{j+1})$, and (iii) for all $\rho' \in \mathit{fpaths}(E)$ with $\rho'(0) = s_k$, we have $E, \rho' \models^* \varphi$.

We write $E \models^* \varphi$ if $E, r \models^* \varphi$ for all runs r of E.

For an environment E, define a state to be reachable if it occurs in some run of E. Say that *observations in E preserve reachability* if for all states s, t of E and all agents i, if s is reachable and $O_i(s) = O_i(t)$ then t is reachable.[4]

[4] We remark that it is always possible to ensure this by deleting the unreachable states from E, an operation that preserves satisfaction of formulas. However, this operation is undesirable in our applications since we will deal with exponential size structures, in which observations already preserve reachability.

Lemma 8. *If observations in E preserve reachability then $E, (r, m) \models^{\mathsf{obs}} \varphi$ iff $E, r|_m \models^* \varphi$.*

Next, we introduce a notion of bisimulation on environments (cf. [14]) in order to reduce the infinite state space of E^v to a finite one while preserving validity of formulae with respect to $\mathtt{obs}$. For environments $E = (S, I, \rightarrow, (O_i)_{i\in\mathbb{A}}, \pi, \alpha)$ and $E' = (S', I', \rightarrow', (O'_i)_{i\in\mathbb{A}}, \pi', \alpha')$, a function $\sigma : S \longrightarrow S'$ is said to be a *bisimulation* from E to E' if the following hold:

1. $I' = \sigma(I)$,
2. if $s \rightarrow s'$ then $\sigma(s) \rightarrow' \sigma(s')$,
3. if $\sigma(s) \rightarrow' u$ then there exists $s' \in S$ such that $\sigma(s') = u$ and $s \rightarrow s'$,
4. if $O_i(s) = O_i(t)$ then $O'_i(\sigma(s)) = O'_i(\sigma(t))$,
5. if $O'_i(\sigma(s)) = O'_i(u)$ then there exists a state $t \in S$ such that $O_i(s) = O_i(t)$ and $\sigma(t) = u$.
6. $\pi' \circ \sigma = \pi$, and
7. $\sigma(s) \in \alpha'$ iff $s \in \alpha$.

Lemma 9. *Suppose that σ is a bisimulation from E to E'. Then*

1. *for all (initialised) $\rho \in fpaths(E)$, $\sigma(\rho)$ is a (initialised) fair path of E';*
2. *for all $\rho' \in fpaths(E')$ and (initial) states s of E, if $\sigma(s) = \rho'(0)$, then there exists a (initialised) $\rho \in fpaths(E)$ with $\rho(0) = s$ such that $\sigma(\rho) = \rho'$;*
3. *for all $\rho \in fpaths(E)$ we have $E, \rho \models^* \varphi$ iff $E', \sigma(\rho) \models^* \varphi$.*

Noting that all states of E^v are reachable, we obtain the following:

Corollary 10. *For all environments E and E', if there exists a bisimulation from E^v to E', then $E \models^v \varphi$ iff $E' \models^* \varphi$.*

This result provides the basic reduction that we use to obtain our complexity results. We now show that the relation $E' \models^* \varphi$ is decidable for finite environments E'. However, we will need to deal with the fact that the structure E' will be of size exponential in the size of E in our applications. For this reason, we express our decision procedure for $\models^*$ as an alternating computation [2], in which we guess and verify the components of E'.

We begin with a reduction to well-known techniques for LTL. Say that a formula is a *pure knowledge formula* if it is of the one of the forms $K_i\psi$ or $C_G\psi$, or their negation. Note that for formulas φ that are either atomic propositions or their negation, or pure knowledge formulas, we have that if $\rho(0) = \rho'(0)$, then $E, \rho \models^* \varphi$ iff $E, \rho' \models^* \varphi$. Thus, for such formulas φ, we may define $E, s \models^* \varphi$, where s is a state of E, to hold if $E, \rho \models^* \varphi$ for some (equivalently, every) path ρ with $\rho(0) = s$.

We may use this state-dependence property to transform the $\mathcal{L}_{\{\bigcirc, \mathcal{U}, K_1, \ldots, K_n, C\}}$ model checking problem with respect to $\models^*$ into a problem of model checking $\mathcal{L}_{\{\bigcirc, \mathcal{U}\}}$, by replacing the pure knowledge subformulas by atomic propositions. Introduce a new atomic proposition $q_{K_i\psi}$ for each formula $K_i\psi$ and $q_{C_G\psi}$ for each formula $C_G\psi$. Let $\mathcal{L}^*_{\{\bigcirc, \mathcal{U}\}}$ be the language of temporal logic over the set of atomic propositions *Prop* together with these new atomic propositions. Given a formula φ of $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ and an occurrence of a pure knowledge formula as a subformula of φ, say this occurrence is *maximal* if it does not lie within the scope of a knowledge or common knowledge operator. For example, in $(K_2\bigcirc K_1 p) \vee K_1 p$, the maximal occurrences of knowledge subformulas are the occurrence of $K_2\bigcirc K_1 p$ and the second (but not the first) occurrence of $K_1 p$. Define φ^* to be the formula of $\mathcal{L}^*_{\{\bigcirc,\mathcal{U}\}}$ obtained by replacing each maximal occurrence of a knowledge formula $K_i\psi$ by the proposition $q_{K_i\psi}$ and similarly for the maximal occurrences of $C_G\psi$.

More precisely,

$$p^* = p \qquad (\varphi_1 \wedge \varphi_2)^* = {\varphi_1}^* \wedge {\varphi_2}^* \qquad (\neg\varphi)^* = \neg(\varphi^*) \qquad (\bigcirc\varphi)^* = \bigcirc(\varphi^*)$$
$$(\varphi_1 \,\mathcal{U}\, \varphi_2)^* = {\varphi_1}^* \,\mathcal{U}\, {\varphi_2}^* \qquad K_i\varphi^* = q_{K_i\psi} \qquad C_G\varphi^* = q_{C_G\psi}$$

Thus, $((K_2\bigcirc K_1 p) \vee K_1 p)^* = q_{K_2\bigcirc K_1 p} \vee q_{K_1 p}$. Write $\mathit{Prop}_{\varphi^*}$ for the set of atomic propositions occuring in φ^* and $\mathit{KProp}_{\varphi^*}$ for the set of atomic propositions of the form $q_{K_i\psi}$ and $q_{C_G\psi}$ that occur in φ^*.

Suppose we enrich the structure E by extending the valuation π so that $q_{K_i\psi} \in \pi(s)$ iff $E, s \models^* K_i\psi$ and $q_{C_G\psi} \in \pi(s)$ iff $E, s \models^* C_G\psi$. Call the resulting structure E^*. Then we have $E, \rho \models^* \varphi$ iff $E^*, \rho \models \varphi^*$. This turns the problem of model checking $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ in E into the problem of model checking $\mathcal{L}^*_{\{\bigcirc,\mathcal{U}\}}$ in E^*. Of course, to apply this technique, we need to have the appropriate extension E^* of E. We may deal with this in an NPSPACE computation by *guessing* the extension E^*, iteratively verifying its correctness over larger and larger pure knowledge subformulas of φ (using LTL model checking techniques), and then model checking the formula φ^*. Since NPSPACE = PSPACE, this already yields a proof of Theorem 1.[5]

However, in our applications, we will not be interested in a given structure E, but in a structure E' of size exponential in the size of E. This means that the cost of guessing $(E')^*$ is exponential. We will handle this by guessing the extension not upfront, but on the fly, for each state of E' as it arises during the verification, and using an APTIME computation that incorporates a Büchi automaton emptiness check for the LTL parts of the verification.

Let $\mathcal{M}_{\varphi^*}$ be the nondeterministic Büchi automaton for the $\mathcal{L}^*_{\{\bigcirc,\mathcal{U}\}}$ formula φ^* over propositions $\mathit{Prop}_{\varphi^*}$, with states S_{φ^*}, initial states I_{φ^*}, transitions $\Rightarrow_a$ (where $a \in \mathcal{P}(\mathit{Prop}_{\varphi^*})$) and acceptance condition α_{φ^*}. We make use of the following properties of this automaton [29]: (1) The automaton is of size $O(2^{|\varphi^*|})$, where each state is of size $O(|\varphi|)$. (2) Deciding S_{φ^*}, I_{φ^*}, $\Rightarrow_a$, and α_{φ^*} can be done in ATIME($\log_2 |\varphi|$).

For a finite environment $E = (S, I, \rightarrow, (O_i)_{i\in\mathbb{A}}, \pi, \alpha)$, we define the product $E \times \mathcal{M}_{\varphi^*}$ (a transition system with Büchi acceptance condition) as follows.

- The transition system has states $\langle b, s, v\rangle$, where $b \in \mathcal{P}(\{0,1\})$, $s \in S$ and $v \in S_{\varphi^*}$. Intuitively, $0 \in b$ ($1 \in b$) represents that E (resp., $\mathcal{M}_{\varphi^*}$), has passed through an accepting state since the most recent accepting state of the product.

[5] The guess and verify technique discussed here is essentially that used in Vardi's results on verifying implementations of knowledge-based programs [28].

– The set of initial states consists of all $\langle \emptyset, s, v\rangle$ where $s \in I$ and $v \in I_{\varphi^*}$.
– There is a transition $\langle b, s, v\rangle \Rightarrow_k \langle b', s', v'\rangle$ for a set $k \subseteq \mathit{KProp}_{\varphi^*}$ when:
  - $s \rightarrow s'$,
  - $v \Rightarrow_{\pi(s)\cup k} v'$, and
  - $b' = b_0 \cup b_1 \cup b_2$, where if $b = \{0, 1\}$ then $b_0 = \emptyset$, else $b_0 = b$; if $s \in \alpha$ then $b_1 = \{0\}$, else $b_1 = \emptyset$; and if $v \in \alpha_{\varphi^*}$ then $b_2 = \{1\}$, else $b_2 = \emptyset$;
– the automaton has as accepting states the states $\langle b, s, v\rangle$ with $b = \{0, 1\}$.

Intuitively, this transition system represents running $\mathcal{M}_{\varphi^*}$ as a monitor on runs of E, with the values of the propositions $\mathit{KProp}_{\varphi^*}$ chosen arbitrarily. Thus, there exists a fair path $\rho = s_0 s_1 \ldots$ of E such that $E, \rho \models^* \varphi$ iff there exists an accepting run $\langle b_0, s_0, v_0\rangle \Rightarrow_{k_0} \langle b_1, s_1, v_1\rangle \Rightarrow_{k_1} \langle b_2, s_2, v_2\rangle \Rightarrow_{k_2} \ldots$ of $E \times \mathcal{M}_{\neg\varphi^*}$ such that for all $j \geq 0$, we have $E, s_j \models^* k_j$. Applying the usual emptiness check for Büchi automata, such a path exists iff we can find a finite such sequence with $\langle b_l, s_l, v_l\rangle$ an accepting state and final element $\langle b_{l'}, s_{l'}, v_{l'}\rangle = \langle b_l, s_l, v_l\rangle$ for some $l' > l$, where both l and $l' - l$ are at most $|E \times \mathcal{M}_{\varphi^*}|$. Our decision procedure searches for such paths using a Savitch-style reachability procedure [15] in order to deal with the exponential size of the search-space.

For the verification that $E, s \models^* k$, it suffices to check, for each maximal knowledge subformula $K_i\psi$ of φ, that $q_{K_i\psi} \in k$ iff $O_i(s) = O_i(t)$ implies that for all fair paths $\rho = t_0 t_1 \ldots$ with $t_0 = t$, we have $E, \rho \models \psi^*$. For this, we recursively apply the above ideas on $E \times \mathcal{M}_{\neg\psi^*}$. Since ψ is a strict subformula of φ, the recursion is well founded. A similar check is applied for the common knowledge subformulas.

We are now ready to present our general algorithm scheme as an alternating computation [2]. Suppose that we are given a finite environment E, for which it is known that there exists a bisimulation from E^v to a finite environment $E' = (S', I', \rightarrow', (O'_i)_{i\in\mathbb{A}}, \pi', \alpha')$. We assume that there is a representation of E' such that the states and other components of E' can be represented and verified within known space and alternating time complexity bounds. (That is, given E, the states of E' are representable as strings of length bounded by some known function of $|E|$, in such a way that we can decide whether such a string represents a state of E', whether $s \rightarrow' s'$ etc. with some known complexity bounds.) We define the following alternating procedure that searches for such runs by operating over the states $\langle b, s, v\rangle$ of the automata $E' \times \mathcal{M}_{\psi^*}$ for subformulas ψ of φ and their negations. For clarity, we write expressions referring to the components of E' (such as "choose $s \in I'$ and do X") which need to be expanded to expressions ("choose s and universally (1) verify $s \in I'$ and (2) do X") that use the verification routines assumed to exist.

`VERIFY`(E, φ)**:** Universally choose $s \in I'$ and call $\neg$`FALSIFY`(E, s, φ)
`FALSIFY`(E, s, ψ)**:** Existentially choose $k \subseteq \mathit{KProp}_{\psi^*}$, an initial state v of $\mathcal{M}_{\neg\psi^*}$, an accepting state $\langle b_0, s_0, v_0\rangle$ and a state $\langle b_1, s_1, v_1\rangle$ of $E' \times \mathcal{M}_{\neg\psi^*}$ such that $(b_0, s_0, v_0) \Rightarrow_k (b_1, s_1, v_1)$.
Let $N = \lceil log_2\, |\mathit{states}(E' \times \mathcal{M}_{\neg\psi^*})| \rceil$.

```
    Universally call:
      – REACH($E, (\emptyset, s, v), (b_0, s_0, v_0), N, \neg\psi$),
      – CHECK($E, s_0, k, \psi$), and
      – REACH($E, (b_1, s_1, v_1), (b_0, s_0, v_0), N, \neg\psi$)
CHECK($E, s, k, \psi$): Universally,
      – for each $p_{K_i\psi'}$ in $KProp_{\psi^*}$,
        if $p_{K_i\psi'} \in k$ then call KCHECK($E, s, K_i\psi'$) else call ¬KCHECK($E, s, K_i\psi'$)
      – for each $p_{C_G\psi'}$ in $KProp_{\psi^*}$,
        if $p_{C_G\psi'} \in k$ then call CKCHECK($E, s, C_G\psi'$) else call ¬CKCHECK($E, s, C_G\psi'$)
KCHECK($E, s, K_i\psi$): Universally, for each $s' \in S'$ where $O'_i(s) = O'_i(s')$, call
    ¬FALSIFY($E, s', \psi$)
CKCHECK($E, s, C_G\psi$): Universally, for each $s' \in S'$: (1) verify[6] there is a sequence
    $s = s_0, \ldots, s_k = s'$ with $k \leq |S'|$ and for each $j < k$ there is an $i \in G$ such
    that $O'_i(s_j) = O'_i(s_{j+1})$. (2) call ¬FALSIFY($E, s', \psi$)
REACH($E, (b_0, s_0, v_0), (b_1, s_1, v_1), N, \psi$): Accept if $(b_0, s_0, v_0) = (b_1, s_1, v_1)$.
    Otherwise if $N = 0$, existentially guess $k \subseteq KProp_{\psi^*}$ then
      – universally verify that $(b_0, s_0, v_0) \Rightarrow_k (b_1, s_1, v_1)$ and CHECK($E, s_0, k, \psi$).
    If $N > 0$, existentially guess a state $(b_2, s_2, v_2)$ of $E \times \mathcal{M}_{\psi^*}$, then universally
    call:
      – REACH($E, (b_0, s_0, v_0), (b_2, s_2, v_2), N - 1, \psi$) and
      – REACH($E, (b_2, s_2, v_2), (b_1, s_1, v_1), N - 1, \psi$).
```

An analysis of the complexity of the algorithm scheme yields the following.

Theorem 11. *Let v be a view. Suppose that $\mathcal{C}$ is a class of environments such that for each environment $E \in \mathcal{C}$ there exists an environment E' with states that can be represented in space $f(|E|)$ and components that can be verified in ATIME$(g(|E|))$, such that there is a bisimulation σ from E^v to E'. Then $\{\ (E, \varphi) \in \mathcal{C} \times \mathcal{L}_{\{\bigcirc, \mathcal{U}, K_1, \ldots, K_n, C\}} \mid E \models^v \varphi\ \}$ is in ATIME$(p(f(|E|), g(|E|), |\varphi|))$ for some polynomial p.*

We remark that since the procedure REACH has an alternation before the recursive call, the number of alternations is also polynomial in $|E|$. Theorem 5 can be understood as asserting that this is inherently so.

In the following sections, we apply Theorem 11 to obtain complexity bounds for model checking the logic of knowledge and linear time in a number of cases. In each case, we identify an appropriate environment E' where the states can be represented and verified in polynomial space and time, respectively, hence the complexity of the alternating procedure is APTIME. By [2], this is equivalent to PSPACE. The environments E' and the bisimulations we use are extensions (by the addition of transition relations $\rightarrow'$) of similar structures that have been used elsewhere in the literature [21] for another problem (existence of finite-state implementations of knowledge-based programs).

[6] In general, this may require another Savitch-style search. In fact, in our applications, $k \leq |S|^2$, i.e., the square of the number of states of E, will suffice, so this is not necessary.

5 Model Checking with Respect to Perfect Recall

In this section we consider several special cases of model checking with respect to perfect recall. The first restricts formulas to refer only to the knowledge of a single agent, and the latter concerns model checking the full language $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ in restricted environments.

5.1 Formulas of $\mathcal{L}_{\{\bigcirc,\mathcal{U},K\}}$

We first treat the case of model checking formulas of a single agent (agent 1) using the perfect recall view. (This case may also be applied to model checking formulas that refer only to a single agent's knowledge, simply by dropping the other agents' observation functions from the environment.)

In this setting it suffices to track the set of states the agent considers possible at each point in time. We define the environment $E' = (S', I', \rightarrow', O'_1, \pi', \alpha')$ by:

- $S' = \{ (s, P) \mid s \in S, P \subseteq S, s \in P \}$
- $I' = \{ (s, P_0(s)) \mid s \in I \}$ where $P_0(s) = \{ s' \in I \mid O_1(s) = O_1(s') \}$
- $(s, P) \rightarrow' (s', P')$ iff $s \rightarrow s'$ and $P' = \{ t' \mid t \in P, t \rightarrow t', O_1(t') = O_1(s') \}$, and
- $O'_1(s, P) = P$.

with bisimulation from E^{pr} given by $\sigma(r, m) = (r(m), P_1^{\text{pr}}(E, r, m))$. Observe that a state of E' can be represented in space $O(\log_2 |S| + |S|)$. States of E' can be seen to be a special case (1-trees) of data structures previously used in [22] for model checking $\mathcal{L}_{\{K_1,\ldots,K_n\}}$. That σ is a bisimulation can be seen by arguments in that work. It is easy to check that observations preserve reachability. By Theorem 11 we conclude that this model checking problem can be decided in PSPACE, which completes the proof of Theorem 2.

5.2 Multi-agent Broadcast and $\mathcal{L}_{\{\bigcirc,\mathcal{U},K_1,\ldots,K_n,C\}}$ with Perfect Recall Semantics

Broadcast environments [21, 25] model situations in which agents may maintain private information, but where the only means by which this information can be communicated is by synchronous simultaneous broadcast to all agents.

We give a definition of broadcast environments here that is slightly more abstract than previous formulations, which dealt with a notion of environment in which agents are equipped with actions that they may perform. Formally, we define a broadcast environment to be an environment $E = (S, I, \rightarrow, (O_i)_{i \in \mathbb{A}}, \pi, \alpha)$ in which the states and observation functions and transition relation have a particular structure.

- The set S of states of E is a subset of $S_0 \times S_1 \times \ldots \times S_n$, where S_0 is a finite set of *shared states*, and for each agent i a set S_i of *private states*. If $s = \langle s_0, \ldots, s_n \rangle$ denotes a state, we write $\mathbf{p}_i(s)$ to denote agent i's private state s_i. For each agent i, define the binary function $\bowtie_i$ on S by $\langle s_0, s_1, \ldots s_n \rangle \bowtie_i \langle t_0, \ldots t_n \rangle = \langle s_0, s_1 \ldots s_{i-1}, t_i, s_{i+1}, \ldots s_n \rangle$. We require that S is closed under the functions $\bowtie_i$.

- The observation functions are given by $O_i(s) = (O_c(s_0), \mathbf{p}_i(s))$, where $O_c : S_0 \longrightarrow \mathcal{O}$ is a *common observation function*.
- The transition relation has the property that for each agent $i = 1 \dots n$, if $O_i(s) = O_i(t)$ and $s \to s'$ and $t \to t'$ and $O_c(s') = O_c(t')$, then $s \to s' \bowtie_i t'$.

There is no constraint on the set of initial states. Intuitively, the common observation function models the information that is being broadcast, and the private states model private information that is being maintained by the agents. An agent's observation consists of the broadcast information and its private information. The condition on the transition relation can be understood as saying that an agent's choice of update on its private state (1) may depend only on the current observation and the incoming common observation and (2) does not affect the update on the common state or any of the other agents' updates.

Paradigmatic examples of broadcast systems are card games such as bridge (where both bidding and playing of cards can be viewed as a broadcast) and systems composed of processes attached to bus, with all processes receiving every communication (as in "snoopy cache coherence protocols" [19]).

Note that every single agent system E is isomorphic to a broadcast system E'. For, if we represent a state s of E by a state $\langle s, O_1(s)\rangle$ of E', and view the first component as being the shared state and the second component as the private state of the single agent, and take $O_c(s) = O_1(s)$, then the constraint on the transition relation is trivially satisfied, because if $O_1(s) = O_1(t)$ then $(s, O_1(s)) \bowtie (t, O_1(t)) = (s, O_1(s))$.

For the broadcast, perfect recall case, we use the following environment E':

- S' is the set of elements $(s, f, t) \in I \times (I \longrightarrow \mathcal{P}(S)) \times S$ such that $t \in f(s)$,
- $(s, f, t) \in I'$ iff $s \in I$, f is given by $f(s') = \{\, t \in I \mid O_c(t) = O_c(s') \,\}$ for all $s' \in I$, and $t \in f(s)$,
- $(s, f, t) \to' (s, f', t')$ if $f'(s') = \{\, u' \mid u \in f(s'), u \to u', O_c(u') = O_c(t') \,\}$ and $t \to t'$,
- $O'_i(s, f, t) = (O_i(s), f, O_i(t))$,

and the bisimulation σ given by $\sigma(r, m) = (r(0), f, r(m))$, where $f(s)$ is the set of states t such that there exists a trace $r'[0 \dots m]$ of E with $O_c(r'[0..m]) = O_c(r[0..m])$ and $t = r'(m)$. Observations preserve reachability in this environment. The states of E' can be represented in size $O(\log_2 |S| + |I| \cdot |S| + \log_2 |S|) = O(|S|^2)$, and applying Theorem 11 shows once again that this model checking problem is in PSPACE, completing the proof of Theorem 3. Note also that in this structure, if $(s, f, t) \overset{\mathsf{obs}}{\sim}_G (s', f', t')$, then it does so by means of a sequence of states all of which have second component f, and also $f' = f$. Thus a maximal path length of $|S|^2$ suffices in CKCHECK.

6 Formulas of $\mathcal{L}_{\{\bigcirc, \mathcal{U}, K_1, \dots, K_n, C\}}$ for the Clock and Observational Views

In order to model check formulas with respect to the clock view, the image of a point (r, m) in the simulating environment E' needs to keep track of the set of states that are reachable in exactly m steps. We define E' by

- $S' = \{ (s, P) \mid s \in S, P \subseteq S, s \in P \}$
- $I' = I \times \{I\}$,
- $(s, P) \rightarrow' (s', P')$ if $s \rightarrow s'$ and $P' = \{t' \mid t \in P, t \rightarrow t'\}$.
- $O'_i(s, P) = (O_i(s), P)$

The bisimulation is given by $\sigma(r, m) = (r(m), \{ r'(m) \mid r' \in runs(E) \})$. Observations can be seen to preserve reachability. The states in the constructed environment can be represented in space $O(\log |S| + |S|)$. This problem is again in PSPACE by Theorem 11. This yields a proof of Theorem 4. In this structure, if $(s, P) \overset{\mathsf{obs}}{\sim}_G (s', P')$, then it does so by means of a sequence of states all of which have second component P, and also $P' = P$. Thus a maximal path length of $|S|$ suffices in CKCHECK.

We can already decide realization in a finite environment E with respect to the observational semantics by furnishing a standard LTL model checker with the equivalence classes induced by the observation function. To remain within our framework, it suffices to use an environment identical to E with bisimulation $\sigma(r, m) = r(m)$ from E^{obs} to E. Its states can be represented in size $O(\log |S|)$. However, in order for observations to preserve reachability in this case, we need to first remove unreachable states from the environment. Here also a maximal path length of $|S|$ suffices in CKCHECK.

7 Conclusion

We have shown that our general bisimulation-based scheme for model checking the logic of knowledge and linear time yields PSPACE complexity bounds in a number of interesting cases of the general problem (which has much higher complexity).

Our notion of bisimulation allows reductions on the temporal structure of environments, but we have not exploited this in our applications. It could be worth exploring this observation in practice. Experiments conducted by Fisler and Vardi [6] suggest that bisimulation reduction is of limited utility for temporal logic model checking, but arguments of van der Meyden and Zhang [26] suggest such reductions might be effective for the much larger search spaces produced when dealing with information flow properties.

The techniques are also applicable to show decidability for certain other classes of environments (with higher complexity bounds). We leave the details for elsewhere. We believe that the techniques we have developed can also be adapted to deal with the combination of branching time and the logic of knowledge: we leave this for future work.

Wozna et al have studied model checking a logic of knowledge and branching time in a real time systems modelled using timed automata [30]. Their semantics is close to our clock semantics, but we note that their until operator is bounded to a specific interval, so the closest appropriate comparison is to our language $\mathcal{L}_{\{\bigcirc, K_1, \ldots, K_n, C\}}$. They give decidability but not complexity results, but study bounded model checking techniques for their logic.

References

[1] A. Baltag, L. S. Moss, and S. Solecki. The logic of public announcements, common knowledge, and private suspicions. In *TARK '98: Proc. 7th Conf. Theoretical Aspects of Rationality and Knowledge*, pages 43–56, 1998.

[2] A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981.

[3] A. Cimatti, C. Pecheur, and R. Cavada. Formal verification of diagnosability via symbolic model checking. In *IJCAI*, pages 363–369, 2003.

[4] A. Cimatti, C. Pecheur, and A. Lomuscio. Applications of model checking for multi-agent systems: Verification of diagnosability and recoverability. In *CS&P '05: Proc. Int. Work. Concurrency, Specification, and Programming*, 2005.

[5] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge.* MIT-Press, 1995.

[6] K. Fisler and M. Y. Vardi. Bisimulation and model checking. In *CHARME '99: Conf. Correct Hardware Design and Verification Methods*, pages 338–341, 1999.

[7] P. Gammie and R. van der Meyden. MCK: Model checking the logic of knowledge. In R. Alur and D. Peled, editors, *CAV*, volume 3114 of *LNCS*, pages 479–483. Springer-Verlag, 2004. http://www.cse.unsw.edu.au/ mck/.

[8] N. O. Garanina and N. V. Shilov. Well-structured model checking of multiagent systems. In *Sixth Int. Andrei Ershov Memorial Conf. Perspectives of System Informatics*, LNCS. Springer-Verlag, 2006. to appear.

[9] J. Y. Halpern and Y. Moses. Knowledge and Common Knowledge in a Distributed Environment. *J. ACM*, 37(3), 1990.

[10] J. Y. Halpern and K. R. O'Neill. Anonymity and information hiding in multiagent systems. In *CSFW*, pages 75–88. IEEE Computer Society, 2003.

[11] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *POPL '85: Proc. 12th ACM SIGACT-SIGPLAN Symp. Principles of Programming Languages*, pages 97–107, New York, NY, USA, 1985. ACM Press.

[12] A. Lomuscio and F. Raimondi. The complexity of model checking concurrent programs against CTLK specifications. In H. Nakashima, M. P. Wellman, G. Weiss, and P. Stone, editors, *AAMAS*, pages 548–550. ACM, 2006.

[13] A. Lomuscio and F. Raimondi. MCMAS: A model checker for multi-agent systems. In H. Hermanns and J. Palsberg, editors, *TACAS*, volume 3920 of *LNCS*, pages 450–454. Springer-Verlag, 2006.

[14] D. M. R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Theoretical Computer Science: 5th GI-Conf., Karlsruhe*, volume 104 of *LNCS*, pages 167–183. Springer-Verlag, Mar. 1981.

[15] W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970.

[16] N. V. Shilov and N. O. Garanina. Model checking knowledge and fixpoints. In Z. Ésik and A. Ingólfsdóttir, editors, *FICS*, volume NS-02-2 of *BRICS Notes Series*, pages 25–39. University of Aarhus, 2002.

[17] N. V. Shilov, N. O. Garanina, and K.-M. Choe. Update and abstraction in model checking of knowledge and branching time. *Fundamenta Informaticae*, 72(1–3):347–361, 2006.

[18] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985.

[19] P. Sweazey and A. J. Smith. A class of compatible cache consistency protocols and their support by the IEEE futurebus. In *ISCA '86: Proc. 13th Int. Symp. Computer Architecture*, pages 414–423, Los Alamitos, CA, USA, 1986. IEEE Computer Society Press.

[20] P. F. Syverson. Knowledge, belief, and semantics in the analysis of cryptographic protocols. *J. Computer Security*, 1(3–4):317–334, 1992.

[21] R. van der Meyden. Finite state implementations of knowledge-based programs. In V. Chandru and V. Vinay, editors, *FSTTCS*, volume 1180 of *LNCS*, pages 262–273. Springer-Verlag, 1996.

[22] R. van der Meyden. Common knowledge and update in finite environments. *Information and Computation*, 140(2):115–157, Feb. 1998.

[23] R. van der Meyden and N. Shilov. Model checking knowledge and time in systems with perfect recall (extended abstract). In *FSTTCS '99: Proc. Conf. Foundations of Software Technology and Theoretical Computer Science*, volume 1738 of *LNCS.* Springer-Verlag, Dec. 1999.

[24] R. van der Meyden and K. Su. Symbolic model checking the knowledge of the dining cryptographers. In *CFSW 2004: Proc. 17th IEEE Computer Security Foundations Workshop*. IEEE Computer Society, 2004.

[25] R. van der Meyden and T. Wilke. Synthesis of distributed systems from knowledge-based specifications. In M. Abadi and L. de Alfaro, editors, *CONCUR*, volume 3653 of *LNCS*, pages 562–576. Springer-Verlag, 2005.

[26] R. van der Meyden and C. Zhang. Algorithmic verification of noninterference properties. In *Proceedings Workshop on Views on Designing Complex Systems*, ENTCS, Bertinoro, Italy, Sept. 2006. to appear.

[27] J. van Eijck and S. Orzan. Modelling the epistemics of communication with functional programming. In M. van Eekelen, editor, *TFP '05: 6th Symp. Trends in Functional Programming*, pages 44–59, 2005.

[28] M. Y. Vardi. Implementing knowledge-based programs. In Y. Shoham, editor, *TARK '96: Proc. 6th Conf. Theoretical Aspects of Rationality and Knowledge*, pages 15–30. Morgan Kaufmann, 1996.

[29] M. Y. Vardi and P. Wolper. Automata theoretic techniques for modal logics of programs (extended abstract). In *STOC*, pages 446–456. ACM, 1984.

[30] B. Wozna, A. Lomuscio, and W. Penczek. Bounded model checking for knowledge and real time. In *AAMAS '05: 4th Int. J. Conf. Autonomous Agents and Multiagent Systems*, pages 165–172, Utrecht, 25–29 July 2005. ACM.

Realizations and LP

Melvin Fitting

Lehman College, Cuny
melvin.fitting@lehman.cuny.edu

Abstract. LP can be seen as a logic of knowledge with justifications. Artemov's Realization Theorem says justifications can be extracted from validities in the Hintikka-style logic of knowledge S4, where they are not explicitly present. We provide tools for reasoning about justifications directly. Among other things, we provide machinery for combining two realizations of the same formula, and for replacing subformulas by equivalent subformulas. The results are algorithmic in nature—semantics for LP plays no role. We apply our results to provide a new algorithmic proof of Artemov's Realization Theorem itself.

1 Introduction

The logic LP (logic of proofs) is a propositional modal-like logic introduced by Artemov, [1], to solve a problem originating with Gödel: provide a natural arithmetic foundation for intuitionistic logic, [2]. It also is an interesting logic for its own sake, a logic of justifications that is closely related to a standard logic of knowledge, S4, and generalizable to analogs of the usual variety of logics of knowledge. But it is a difficult logic to work with. Semantics, see [3], is more complex than that of standard modal logics, and the same applies to its proof theory. The LP advantage lies in its representation of explicit justifications, originally intended to be proofs in the mathematical sense. Justifications provide a mechanism for dealing with logical omniscience problems—they supply a measure of how hard it is to know something. It should be possible to reason about justifications themselves in useful ways but it turns out that doing so presents considerable technical difficulties. We provide some tools, algorithms, for manipulating formulas containing justifications. Since this is a short version of my results, only the algorithms are given—proofs that they are correct can be found on my web site.

A central result concerning LP is Artemov's Realization Theorem 3—any theorem of S4 can be *realized*, which means modal operators can be replaced with explicit justifications to produce a theorem of LP, and these justifications can be calculated. It is as if the modal operator of S4 was a quantifier, 'there exists a reason...', and these quantifiers were constructive. At the end of this paper we sketch a new proof of the Realization Theorem. The general plan of this paper is as follows. There is a brief proof-theoretic presentation of LP—semantics plays no role here. An *annotated* version of S4 is given, in which distinct occurrences of modal operators are syntactically distinguished. Then realizations become first-class objects, as *realization functions*. A Realization Modification

Theorem is proved, and a number of consequences are derived from it, all constructive in nature. As remarked earlier, here only the algorithms can be given, while correctness proofs are relegated to my web site.

Since Artemov's original work, LP has been generalized to a family of logics called *Justification Logics*. Thus LP is one among many—indeed various multimodal logics of knowledge have also been brought into the picture. I do not address the whole family of justification logics in detail here—things are complicated enough—but it is clear that my methods do extend beyond LP itself.

2 The Logic LP

This section contains a brief formulation of LP axiomatically. A semantics will not be needed in this paper. The language of LP, denoted L_{LP} here, is built from the following basic machinery: propositional variables, P, Q, P_1, P_2, …; a propositional constant, $\bot$; a logical connective, $\supset$; proof variables, x, y, x_1, x_2, …; proof constants, c, d, c_1, c_2, …; function symbols ! (monadic), $\cdot$, $+$ (binary); and an operator symbol of the type $\langle term\rangle{:}\langle formula\rangle$.

Proof. polynomials are built up from proof variables and proof constants, using the function symbols. *Ground* proof polynomials are those without variables. *Formulas* are built up from propositional variables and the propositional constant $\bot$ using $\supset$ (with other connectives defined in the usual way), and an extra rule of formation: if t is a proof polynomial and X is a formula then $t{:}X$ is a formula.

The formula $t{:}X$ can be read: "t is a proof of X." Proof constants intuitively represent proofs of basic, assumed truths. Proof variables in a formula can be thought of as justifications supplied by the outside—by the world, if you like. If t is a proof of $X \supset Y$ and u is a proof of X, we should think of $t \cdot u$, the application of t to u, as a proof of Y. The operation ! is a proof-checker: if t is a proof of X then $!t$ is a verification that t is such a proof. The operation $+$ combines proofs in the sense that $t + u$ proves all the things that t proves plus all the things that u proves.

The following axiom system for LP is from [1]. Axioms are specified by giving axiom schemas, and these are:

A0. **Classical**	Classical propositional axiom schemes
A1. **Application**	$t{:}(X \supset Y) \supset (s{:}X \supset (t{\cdot}s){:}Y)$
A2. **Reflexivity**	$t{:}X \supset X$
A3. **Proof Checker**	$t{:}X \supset {!t}{:}(t{:}X)$
A4. **Sum**	$s{:}X \supset (s{+}t){:}X$
	$t{:}X \supset (s{+}t){:}X$

Rules of inference are modus ponens, and a version of the necessitation rule for axioms only.

R1. **Modus Ponens**	$\vdash Y$ provided $\vdash X$ and $\vdash X \supset Y$
R2. **Axiom Necessitation**	$\vdash c{:}X$ where X is an axiom *A0 – A4* and c is a proof constant.

As usual, a proof is a finite sequence of formulas each of which is an axiom or comes from earlier terms by one of the rules of inference. A notion of *derivation* can be introduced by defining $\Gamma \vdash X$ to mean that $(G_1 \wedge \ldots \wedge G_n) \supset X$ is a theorem for some finite subset $\{G_1, \ldots, G_n\}$ of Γ.

The specification of which constants are associated with which axioms for rule *R2* applications is called a *constant specification.* A constant specification is *injective* if each proof constant is used for at most one axiom. Injective constant specifications suffice, but are not required. If a proof uses an injective constant specification, I will say the proof is *injective*, and what it proves is *injectively provable.* In [3] constant specifications were assumed to be given beforehand, and their properties were investigated in some detail. Computational complexity is dependent on details of the constant specification. In [1] things were more flexible, and constants were generally assigned during the course of a proof, as will be done here.

Definition 1. *A* substitution *maps proof variables to proof polynomials. If σ is a substitution and X is a formula, $X\sigma$ is the result of replacing each proof variable x in X with the proof polynomial $x\sigma$. Similarly for substitution in proof polynomials.*

Theorem 1 (Substitution Lemma). *If X is a theorem of LP, so is $X\sigma$. Further, if X has an injective proof, so does $X\sigma$.*

This result and the following are from [1]. The Lifting Lemma says that proofs and derivations in LP can be internalized.

Theorem 2 (Lifting Lemma). *Suppose*

$$s_1{:}X_1, \ldots, s_n{:}X_n, Y_1, \ldots, Y_k \vdash Z$$

then there is a proof polynomial $t(s_1, \ldots, s_n, y_1, \ldots, y_k)$ (where the y_i are variables) such that

$$s_1{:}X_1, \ldots, s_n{:}X_n, y_1{:}Y_1, \ldots, y_k{:}Y_k \vdash t(s_1, \ldots, s_n, y_1, \ldots, y_k){:}Z.$$

If the original derivation was injective, the same is the case for the later derivation.

Corollary 1. *If Z has an LP proof, then for some ground proof polynomial t, $t{:}Z$ will have an LP proof, injective if the proof of Z was injective.*

3 Annotations and Realizations

Let $L_\Box$ be the usual language of propositional modal logic, built up from propositional letters using $\bot$, $\supset$, and $\Box$, with other connectives and $\Diamond$ defined as usual.

In the LP Realization Theorem negative $\Box$ occurrences are replaced by proof variables while positives need not be, and different negative occurrences are replaced with distinct variables. To keep track of this we introduce an annotated version, $L_\Box^a$, of the language $L_\Box$, intermediate between $L_\Box$ and L_{LP}.

Definition 2. *The language $L^a_\Box$ and its features are introduced as follows.*

1. *There is an infinite family of* indexed *modal operators,* $\Box_1, \Box_2, \ldots$. *Formulas of $L^a_\Box$ are built up as in $L_\Box$, but using indexed modal operators instead of $\Box$. Formulas of $L^a_\Box$ are called* annotated formulas.
2. *If X is an annotated formula, and X' is the result of replacing all indexed modal operators, $\Box_n$, with $\Box$, then X' is a formula of $L_\Box$. We say X is an* annotated version *of X', and X' is an* unannotated version *of X.*
3. *A* properly *annotated formula is an annotated formula in which no indexed modal operator occurs twice, and if $\Box_n$ occurs in a negative position n is even, and if it occurs in a positive position n is odd.*

Example 1. Here is an example of a properly annotated formula.

$$\Box_2(\Box_1 U \supset \Box_4(\Box_3 P \supset \Box_6 V)) \supset \Box_5 W \tag{1}$$

Annotation is a bookkeeping device to keep track of occurrences of modal operators and their polarities. Properly annotated formulas are fundamental, but formulas that are annotated but not properly so also arise naturally. If X is properly annotated and Y is a negative subformula of X it will not be properly annotated because polarities have been reversed in passing from X to Y. Generally we will fix a properly annotated formula X and work with subformulas of it, all of which are annotated, and properly so when considered as subformulas of X.

In an S4 model $\mathcal{M} = \langle \mathcal{G}, \mathcal{R}, \Vdash \rangle$ we use the following rule of evaluation:

$$\mathcal{M}, \Gamma \Vdash \Box_n X \iff \mathcal{M}, \Delta \Vdash X \text{ for every } \Delta \in \mathcal{G} \text{ with } \Gamma \mathcal{R} \Delta$$

Then in a model, an annotated formula X and its unannotated version X' behave alike at each world. Likewise proof-theoretically annotations play no special role. They are only for syntactic bookkeeping.

Now realizations can be defined functionally, but first, a comment. Call the displayed occurrence of t in the formula $t{:}Z$ *self-referential* if t also has an occurrence in Z. It is shown in [4] that one must admit self-referential proof constants in order to have the Realization Theorem hold. Here we will sometimes need to exercise control over the self-referentiality of proof variables, hence item 3 below.

Definition 3. *Realization functions and related notions are defined as follows.*

1. *A* realization function *is a mapping from positive integers to proof polynomials that maps even integers to* LP *variables. It is assumed that all realization functions behave the same on the even integers: if r is any realization function, $r(2n) = x_n$, where $x_1, x_2, \ldots$ is the list of proof variables arranged in a standardized order.*
2. *If X is a formula of $L^a_\Box$, an annotated formula, and r is a realization function, $r(X)$ is the result of replacing each modal operator $\Box_i$ in X with the proof polynomial $r(i)$. The result, $r(X)$ is formula of* LP.
3. *Realization function r is* non self-referential on variables *over annotated formula X provided, for each subformula $\Box_{2n} Y$ of X the variable $r(2n) = x_n$ does not occur in $r(Y)$.*

Example 2. Let f, g, h, and k be particular proof polynomials, and let r be a realization function such that the following holds.

$$\begin{array}{ll} r(1) = g(x_2, x_3, x_5) & r(4) = x_2 \\ r(2) = x_1 & r(5) = h(x_1, x_2, x_3) \\ r(3) = f(x_3) & r(6) = x_3 \end{array} \qquad (2)$$

Let X be formula (1) from Example 1. Then we have the following. Note that r is non self-referential on variables over X.

$$r(X) = x_1{:}[g(x_2, x_3, x_5){:}U \supset x_2{:}(f(x_3){:}P \supset x_3{:}V)] \supset h(x_1, x_2, x_3){:}W \qquad (3)$$

Definition 4. *If X is a formula of $L_\Box$, a conventional modal formula, a* realization *of X is any formula of* LP *of the form $r(X')$ where r is a realization function and X' is any properly annotated version of X.*

Theorem 3 (Artemov's Realization Theorem). *If Z_0 is a theorem of S4, there is a realization of Z_0 that is an injectively provable theorem of LP. In fact, if Z_0 is a theorem of S4, then for any properly annotated version Z of Z_0 there is a realization function r such that $r(Z)$ is injectively provable in LP.*

4 Realization Modification

Our results concern the combination and modification of realization functions. Full statements are rather complex and technical, but the following may help make it clear why the complexity is arises.

In normal modal logics one can show a Replacement result: If $X(B)$ is like $X(A)$ except that some occurrence of A has been replaced with B, then if $A \equiv B$ is provable so is $X(A) \equiv X(B)$. In the LP Realization Theorem positive and negative occurrences of proof polynomials are treated differently, and the connective $\equiv$ blurs the positive/negative distinction. There is another version of Replacement that respects polarity of subformula occurrence: If $X(B)$ is like $X(A)$ except that some *positive* occurrence of A has been replaced with B, then if $A \supset B$ is provable so is $X(A) \supset X(B)$. This too is correct for normal modal logics. But one would not expect a simple analog of such a result for LP. Proof polynomials represent justifications. If A is replaced with B inside a more complex LP formula, justifications for A must be adjusted to incorporate a justification for the passage from A to B, justifications for subformulas containing justifications for A need adjustment, and so on up. The fact that reasons are *explicit* makes life harder in some ways: we care not only about truth but also about the reasons for that truth. Here is an example to illustrate the complexity.

Example 3. In the modal formula

$$\Box(\Box U \supset \Box(\Box P \supset \Box V)) \qquad (4)$$

we can replace P successively by $\Box R$ and $\Box\Box R$. In S4, $\Box R \supset \Box\Box R$ is provable, and it follows that so is

$$[\Box(\Box U \supset \Box(\Box\Box R \supset \Box V))] \supset [\Box(\Box U \supset \Box(\Box\Box\Box R \supset \Box V))] \qquad (5)$$

In LP, $x_1{:}[g(x_2, x_3, x_5){:}U \supset x_2{:}(f(x_3){:}P \supset x_3{:}V)] \supset h(x_1, x_2, x_3){:}W$ comes from (4) by substituting distinct variables for negative occurrences of $\Box$ and other proof polynomials for positive occurrences—it is a realization of it. Likewise $k(x_3){:}R$ is a realization of $\Box R$ and $!k(x_3){:}k(x_3){:}R$ is a realization of $\Box\Box R$, and $k(x_3){:}R \supset !k(x_3){:}k(x_3){:}R$ is a theorem of LP. But replacing P by $k(x_3){:}R$ and $!k(x_3){:}k(x_3){:}R$ to produce an analog of (5) gives us

$$\begin{aligned}
&\{x_1{:}[g(x_2, x_3, x_5){:}U \supset x_2{:}(f(x_3){:}k(x_3){:}R \supset x_3{:}V)] \\
&\quad \supset h(x_1, x_2, x_3){:}W\} \\
&\supset \\
&\{x_1{:}[g(x_2, x_3, x_5){:}U \supset x_2{:}(f(x_3){:}!k(x_3){:}k(x_3){:}R \supset x_3{:}V)] \\
&\quad \supset h(x_1, x_2, x_3){:}W\}
\end{aligned} \tag{6}$$

and this is *not* a theorem of LP. Reasons need adjustment. Instead the following is an LP theorem,where c, d, and e are particular constants. Note that the form of this is similar to the form of (6), but details differ significantly.

$$\begin{aligned}
&\{(e \cdot x_1){:}[g(d \cdot x_2, x_3, x_5){:}U \supset (d \cdot x_2){:}(f(x_3){:}k(x_3){:}R \supset x_3{:}V)] \\
&\quad \supset h(e \cdot x_1, d \cdot x_2, x_3){:}W\} \\
&\supset \\
&\{x_1{:}[g(d \cdot x_2, x_3, x_5){:}U \supset x_2{:}((c \cdot f(x_3)){:}!k(x_3){:}k(x_3){:}R \supset x_3{:}V)] \\
&\quad \supset h(e \cdot x_1, d \cdot x_2, x_3){:}W\}
\end{aligned} \tag{7}$$

This example will be continued in Example 4.

Our basic construction provides an algorithm for computing formulas like (7). There is also provision for the merging of separate realizations into a single one. Details are complex and in this short paper we only have space to state the algorithms, and not to prove correctness. That can be found on my web page. Terms introduced in the following definition are primarily for use in proving correctness of our algorithm.

Definition 5. *Let X be an annotated formula and let σ be a substitution.*

1. *σ meets the* no new variable *condition provided, for each variable x, the proof polynomial $x\sigma$ contains no variables other than x.*
2. *An indexed modal operator $\Box_{2n}$ that occurs in X is an* input operator *in X. If $\Box_{2n}$ is an input operator, x_n is an* input variable *of $r(X)$, where r is any proper realization function.*
3. *σ* lives on input positions *in X provided the only variables x_n for which $x_n\sigma \neq x_n$ are such that $\Box_{2n}$ occurs in X .*

In what follows, if $\psi(P)$ is an annotated formula and P is a propositional letter, $\psi(A)$ is the result of replacing all occurrences of P in $\psi(P)$ with occurrences of the annotated formula A. We care about *proper* annotation but subformulas of a properly annotated formula need not be properly annotated. To deal with this we fix a formula $X(P)$ that is properly annotated and work with subformulas of $X(P)$.

Definition 6. *Let $X(P)$ be an annotated formula in which the propositional letter P has at most one positive occurrence, let A and B be annotated formulas, let r_0 be a realization function, and let $\varphi(P)$ be a subformula of $X(P)$.*

1. *For a subformula $\varphi(P)$ of $X(P)$, we say a realization/substitution pair $\langle r, \sigma\rangle$* replaces $r_0(A)$ with $r_0(B)$ at P in $\varphi(P)$ within $X(P)$ *provided:*
   (a) *if $\varphi(P)$ is a positive subformula of $X(P)$ then $r_0(\varphi(A))\sigma \supset r(\varphi(B))$ has an injective* LP *proof;*
   (b) *if $\varphi(P)$ is a negative subformula of $X(P)$ then $r(\varphi(B)) \supset r_0(\varphi(A))\sigma$ has an injective* LP *proof.*
2. $\langle r, \sigma\rangle$ hereditarily replaces $r_0(A)$ *with* $r_0(B)$ *at* P *in* $X(P)$ *if, for each subformula $\varphi(P)$ of $X(P)$, $\langle r, \sigma\rangle$ replaces $r_0(A)$ with $r_0(B)$ at P in $\varphi(P)$ within $X(P)$.*

In the definition above one can see the beginnings of applicability to Example 3. A replacement in an LP formula can be specified by giving a replacement to carry out in an S4 formula and a mapping, that is a realization function, to LP. Now, here is our central construction.

Theorem 4 (Realization Modification). *Assume all of the following: $X(P)$ is a properly annotated formula in which the propositional letter P has at most one positive occurrence, A and B are properly annotated formulas, A and $X(P)$ share no indexes, and B and $X(P)$ share no indexes; r_1 and r_2 are realization functions, both non self-referential on variables over $X(A)$; $r_1(A) \supset r_1(B)$ and $r_2(A) \supset r_2(B)$ both have injective proofs; and $r_1(B) = r_2(B)$.*

Then there is a realization/substitution pair $\langle r, \sigma\rangle$ such that: $\langle r, \sigma\rangle$ hereditarily replaces $r_1(A)$ with $r_1(B)$ and $r_2(A)$ with $r_2(B)$ at P in $X(P)$. In addition σ lives on input positions in $X(P)$; σ meets the no new variable condition; and if r_1 and r_2 are non self-referential on variables over $X(B)$ then r is non self-referential on variables over $X(B)$.

The proof is constructive; here is the algorithm involved. A (lengthy) proof of correctness for the algorithm can be found on my web site.

Begin Algorithm. First we construct, for each subformula $\varphi(P)$ of $X(P)$, a pair $\langle r_\varphi, \sigma_\varphi\rangle$ that replaces $r_1(A)$ with $r_1(B)$ and $r_2(A)$ with $r_2(B)$ at P in $\varphi(P)$ within $X(P)$. The algorithm proceeds recursively on the structure of $\varphi(P)$.

Base Case: $\varphi(P)$ is atomic. Set $r_\varphi = r_1$ and σ_φ to be the identity substitution.
Modal Case: $\varphi(P)$ is $\Box_i\theta(P)$, and $\langle r_\theta, \sigma_\theta\rangle$ has been constructed.

$\varphi(P)$ positive: By hypothesis, the following are provable.

$$r_1(\theta(A))\sigma_\theta \supset r_\theta(\theta(B)) \tag{8}$$
$$r_2(\theta(A))\sigma_\theta \supset r_\theta(\theta(B)) \tag{9}$$

Use the Lifting Lemma to produce proof polynomials u and v that "prove" (8) and (9) respectively. Then set $\sigma_\varphi = \sigma_\theta$ and define r_φ as follows.

$$r_\varphi(n) = \begin{cases} u \cdot (r_1(n)\sigma_\theta) + v \cdot (r_2(n)\sigma_\theta) & \text{if } n = i \\ r_\theta(n) & \text{otherwise} \end{cases}$$

$\varphi(P)$ *negative:* In this case i is even. By hypothesis, the following are provable.

$$r_\theta(\theta(B)) \supset r_1(\theta(A))\sigma_\theta \tag{10}$$

$$r_\theta(\theta(B)) \supset r_2(\theta(A))\sigma_\theta \tag{11}$$

Use the Lifting Lemma to produce proof polynomials u and v that "prove" (10) and (11) respectively. Then set $r_\varphi = r_\theta$ and define σ_φ as follows, where $i = 2j$.

$$x_n\sigma_\varphi = \begin{cases} u \cdot x_j + v \cdot x_j & \text{if } n = i \\ x_n\sigma_\theta & \text{otherwise} \end{cases}$$

Implication Case: $\varphi(P)$ is $\theta(P) \supset \eta(P)$ and $\langle r_\theta, \sigma_\theta \rangle$ and $\langle r_\eta, \sigma_\eta \rangle$ have been constructed. It is shown that the two substitutions commute. Set $\sigma_\varphi = \sigma_\theta\sigma_\eta = \sigma_\eta\sigma_\theta$, and define r_φ as follows.

$$r_\varphi(n) = \begin{cases} r_\theta(n)\sigma_\eta & \text{if } \Box_n \text{ in } \theta(B) \\ r_\eta(n)\sigma_\theta & \text{if } \Box_n \text{ in } \eta(B) \\ r_1(n) & \text{otherwise} \end{cases}$$

Now the pair $\langle r_X, \sigma_X \rangle$ can be shown to provide an *hereditary* replacement.
End Algorithm.

5 The Replacement Theorem

In Example 3 the complexity of adapting the Replacement Theorem to LP was illustrated. In Theorem 4, if we take both r_1 and r_2 to be the same, what results is an algorithm to effect a replacement and which gives the results found in the earlier example. The following illustrates how.

Example 4. Continuing Examples 2 and 3, take A to be $\Box_7 R$ and B to be $\Box_9\Box_{11}R$, so $A \supset B$ is $\Box_7 R \supset \Box_9\Box_{11}R$. A and B are properly annotated, as was $X(P)$ from (1), and there is no annotation overlap. Then $X(A)$ is $\Box_2(\Box_1 U \supset \Box_4(\Box_3\Box_7 R \supset \Box_6 V)) \supset \Box_5 W$ and $X(B)$ is $\Box_2(\Box_1 U \supset \Box_4(\Box_3\Box_9\Box_{11}R \supset \Box_6 V)) \supset \Box_5 W$. Specifying more of the realization function r that was partly given in (2): $r(7) = k(x_3)$, $r(9) = !k(x_3)$, and $r(11) = k(x_3)$. Then r is non self-referential on variables over $X(A)$, and $r(A) \supset r(B)$ is $k(x_3){:}R \supset !k(x_3){:}k(x_3){:}R$, which has an injective LP proof. Carrying out a naive replacement, we might expect $r(X(A)) \supset r(X(B))$ to have an injective LP proof, but this is the formula (6), which was earlier seen to be incorrect.

Let c, d, and e be ground proof polynomials such that the following have injective proofs.

$$c{:}[k(x_3){:}R \supset !k(x_3){:}k(x_3){:}R]$$

$$d{:}\{[(c \cdot f(x_3)){:}!k(x_3){:}k(x_3){:}R \supset x_3{:}V] \supset [f(x_3)){:}k(x_3){:}R \supset x_3{:}V]\}$$

$$\begin{aligned} e{:}\{&[g(d \cdot x_2, x_3, x_5){:}U \supset x_2{:}((c \cdot f(x_3)){:}!k(x_3){:}k(x_3){:}R \supset x_3{:}V)] \supset \\ &[g(d \cdot x_2, x_3, x_5){:}U \supset (d \cdot x_2){:}(f(x_3){:}k(x_3){:}R \supset x_3{:}R)]\} \end{aligned}$$

Following the algorithm of Theorem 4, a realization/substitution pair that hereditarily replaces $r(A)$ with $r(B)$ at P in $X(P)$ is $\langle r^*, \sigma^*\rangle$, specified as follows. σ^* is the identity substitution except that $\sigma^*(x_1) = e \cdot x_1$ and $\sigma^*(x_2) = d \cdot x_2$, and

$$\begin{array}{ll} r^*(1) = g(d \cdot x_2, x_3, x_5) & r^*(6) \ = x_3 \\ r^*(2) = x_1 & r^*(7) \ = k(x_3) \\ r^*(3) = c \cdot f(x_3) & r^*(9) \ = !k(x_3) \\ r^*(4) = x_2 & r^*(11) = k(x_3). \\ r^*(5) = h(e \cdot x_1, d \cdot x_2, x_3) & \end{array}$$

The formula $r(X(A))\sigma^*$ is

$$(e\cdot x_1){:}(g(d\cdot x_2, x_3, x_5){:}U \supset (d\cdot x_2){:}(f(x_3){:}k(x_3){:}R \supset x_3{:}V)) \supset h(e\cdot x_1, d\cdot x_2, x_3){:}W$$

and $r^*(X(B))$ is

$$x_1{:}(g(d\cdot x_2, x_3, x_5){:}U \supset x_2{:}((c\cdot f(x_3)){:}!k(x_3){:}k(x_3){:}R \supset x_3{:}V)) \supset h(e\cdot x_1, d\cdot x_2, x_3){:}W$$

and $r^*(X(A))\sigma^* \supset r^*(X(B))\sigma^*$ is the formula (7) seen earlier. This example is worked out in greater detail in [5].

6 Realization Weakening

When replacing $t{:}F$ and $u{:}F$ with the weaker $(t + u){:}F$ it is possible to drop the non self-referentiality condition. This plays an important role in our proof of Artemov's Realization Theorem.

Theorem 5 (Realization Weakening). *Assume the following: $X(P)$ is a properly annotated formula in which the propositional letter P has at most one positive occurrence; $\Box_p K$ and $\Box_q K$ are both are properly annotated, there is no annotation overlap between $X(P)$ and $\Box_p K$, and $X(P)$ and $\Box_q K$, and p and q are different; r_1 and r_2 are realization functions with $r_1(K) = r_2(K)$; and $r_1(q) = r_2(q) = r_1(p) + r_2(p)$.*

Then there is a realization/substitution pair $\langle r, \sigma\rangle$ that hereditarily replaces $r_1(\Box_p K)$ with $r_1(\Box_q K)$ at P in $X(P)$, and hereditarily replaces $r_2(\Box_p K)$ with $r_2(\Box_q K)$ at P in $X(P)$. Also, σ will live on the input positions in $X(P)$ and will meet the no new variable condition.

If we set $r_1(K) = r_2(K) = F$, $r_1(p) = t$, $r_2(p) = u$, and $r_1(q) = r_2(q) = t + u$, this theorem provides a replacement of $t{:}F$ and $u{:}F$ with $(t + u){:}F$, as promised.

Begin Algorithm. Call an index n in a properly annotated formula Z *self-referential* with respect to a realization function r if $\Box_n$ occurs in Z within the scope of $\Box_{2k}$ and x_k occurs in $r(n)$.

Define new realization functions r_1' and r_2' as follows. For each index n_i in $X(\Box_p K)$ outside K that is self-referential with respect to r_1, introduce a new variable u_{n_i} and set $r_1'(n_1) = u_{n_i}$. For each index m_i in $X(\Box_p K)$ outside K

that is self-referential with respect to r_2, introduce a new variable v_{m_i} and set $r_2'(m_i) = v_{m_i}$. For each index p_i in K that is self-referential with respect to r_1 and r_2 (these go together within K), introduce a new variable w_{p_i} and set $r_1'(p_i) = r_2'(p_i) = w_{p_i}$. Set $r_1'(q) = r_2'(q) = r_1'(p) + r_1'(q)$. Otherwise r_1' and r_1 agree, and r_2' and r_2 agree.

r_1' and r_2' are realizaion functions that are non self-referential on variables over $X(\Box_p K)$. Now apply the algorithm of Theorem 4, getting a realization/substitution pair $\langle r^*, \sigma^* \rangle$.

Define an 'undoing' substitution σ' as follows.

$$x\sigma' = \begin{cases} r_1(n_i)\sigma^* & \text{if } x = u_{n_i} \\ r_2(m_i)\sigma^* & \text{if } x = u_{m_i} \\ r_1(p_i)\sigma^* = r_2(p_i)\sigma^* & \text{if } x = u_{p_i} \\ x & \text{otherwise} \end{cases}$$

The realization/substitution pair $\langle r, \sigma \rangle$ we want is given by: $r(n) = r^*(n)\sigma'$ and $\sigma = \sigma^*$.
End Algorithm.

7 The Realization Merging Theorem

We show how two realization functions may be merged into one. Note that again there is no non self-referentiality condition on variables.

Definition 7. *Let X be an annotated formula and r_1 and r_2 be realization functions.*

1. *For a subformula φ of X, we say a realization/substitution pair $\langle r, \sigma \rangle$* merges r_1 and r_2 on φ in X *provided:*
   (a) *if φ is a positive subformula of X then both $r_1(\varphi)\sigma \supset r(\varphi)$ and $r_2(\varphi)\sigma \supset r(\varphi)$ are injective theorems of LP;*
   (b) *if φ is a negative subformula of X then both $r(\varphi) \supset r_1(\varphi)\sigma$ and $r(\varphi) \supset r_2(\varphi)\sigma$ are injective theorems of LP.*
2. *We say $\langle r, \sigma \rangle$* hereditarily merges r_1 *and r_2 on X provided, for each subformula φ of X, $\langle r, \sigma \rangle$ merges r_1 and r_2 on φ in X.*

Theorem 6 (Realization Merging). *Let X be a properly annotated formula, and r_1 and r_2 be realization functions. Then there is a realization/substitution pair $\langle r, \sigma \rangle$ that hereditarily merges r_1 and r_2 on X. Further, σ will live on the input positions in X, and will meet the no new variable condition.*

Begin Algorithm. Let P be a propositional letter not in X, let p and q be distinct odd indexes not in X, and let K be P. Now apply the algorithm of Theorem 5.
End Algorithm.

Example 5. Suppose $(A \vee B) \supset C$ is a properly annotated formula, and suppose we have realization functions r_1 and r_2 such that $r_1(A \supset C)$ and $r_2(B \supset C)$ are both theorems of LP. We explicitly produce a realization function r such that $r((A \vee B) \supset C)$ is a theorem of LP.

By Theorem 6 there is a realization/substitution pair $\langle r, \sigma\rangle$ that hereditarily merges r_1 and r_2 on $(A \vee B) \supset C$. We claim $r((A \vee B) \supset C)$ is a theorem of LP. Note that since A and B are negative subformulas of X, and C is a positive subformula, the following are theorems of LP: $r(A) \supset r_1(A)\sigma$, $r(B) \supset r_2(B)\sigma$, $r_1(C)\sigma \supset r(C)$, $r_2(C)\sigma \supset r(C)$. By assumption, we have as LP theorems $r_1(A) \supset r_1(C)$ and $r_2(B) \supset r_2(C)$ and hence by Theorem 1, we also have $r_1(A)\sigma \supset r_1(C)\sigma$ and $r_2(B)\sigma \supset r_2(C)\sigma$. Putting all this together, we have the following derivation.

$$\begin{aligned} r(A \vee B) &\supset r(A) \vee r(B) \\ &\supset r_1(A)\sigma \vee r_2(B)\sigma \\ &\supset r_1(C)\sigma \vee r_2(C)\sigma \\ &\supset r(C) \vee r(C) \\ &\supset r(C) \end{aligned}$$

8 The Realization Theorem

Here is my construction for the Artemov Realization Theorem, stated earlier as Theorem 3.

Begin Algorithm. We use a standard cut-free sequent calculus for $\mathsf{S4}$, say the one used in [1], except that annotations are carried along from premise sequents to conclusion sequent; an explicit formulation is on my web page. If Z is a properly annotated version of Z_0, and Z_0 is a theorem of $\mathsf{S4}$, Z will have a sequent calculus proof in this system. A realization function is constructed for each sequent in a proof $\mathcal{P}$ of Z, and the function for the final sequent is the desired realization function.

For sequents that are axioms, any realization function will do.

For all rules except $L\supset$ and $R\Box$, if r realizes the premise of a sequent rule, r will also realize the conclusion.

For the $L\supset$ rule, if r_1 realizes $\Gamma, Y \rightarrow \Delta$ and r_2 realizes $\Gamma \rightarrow \Delta, X$, then r will realize $\Gamma, X \supset Y \rightarrow \Delta$, where r is the merging of r_1 and r_2 using the algorithm of Theorem 6.

For the $R\Box$ rule, suppose r_0 realizes $\Box_{2n_1}Y_1, \ldots, \Box_{2n_k}Y_k \longrightarrow X$. We must realize $\Box_{2n_1}Y_1, \ldots, \Box_{2n_k}Y_k \longrightarrow \Box_m X$. By the Lifting Lemma, there is a proof polynomial $t(x_{n_1}, \ldots, x_{n_k})$ such that $(x_{n_1}{:}r_0(Y_1) \wedge \ldots \wedge x_{n_k}{:}r_0(Y_k)) \supset t(x_{n_1}, \ldots, x_{n_k}){:}$ $r_0(X)$ is provable. Let p and q be distinct odd new indexes. Let r_1 and r_2 be the same as r_0 except that $r_1(p) = r_0(m)$, $r_2(p) = t(x_{n_1}, \ldots, x_{n_k})$, and $r_1(q) = r_2(q) = r_1(p) + r_2(p)$. Let P be a new propositional letter, and let $Z(P)$ be Z with $\Box_m X$ replaced with P. Use the algorithm of Theorem 5 to hereditarily replace $r_1(\Box_p X)$ with $r_1(\Box_q X)$ and $r_2(\Box_p X)$ with $r_2(\Box_q X)$ at P in $Z(P)$,

getting a realization/substitution pair $\langle r^*, \sigma^* \rangle$. Finally, let r be like r^* except that $r(m) = r^*(q)$. Then r realizes $\Box_{2n_1} Y_1, \ldots, \Box_{2n_k} Y_k \longrightarrow \Box_m X$.
End Algorithm.

9 Conclusion

The logic LP is one of a family of *justification logics*. It can be weakened by dropping the ! operator and its axiom. It can be strengthened by adding a *negative* proof checker, dual to !, see [6]. The work in this paper extends to these logics. LP can also be extended to incorporate multiple agents, and even common knowledge. It is not yet known if the present work extends to the multiple agent versions. A Realization Theorem for LP with an added negative proof checker, ?, was recently shown in [6] by a semantic argument. Present methods provide a second, proof-theoretic, argument as well.

S4 is a well-known logic of knowledge in the Hintikka tradition—a logic with a single knower having positive introspection. Then LP can be seen as that logic of knowledge made explicit, with "known to be the case" replaced by "known because of such-and-such evidence". The Realization Theorem is a way of extracting the explicit content of a knowledge statement in which justifications are present only implicitly. One can think of the whole LP/S4 package as a kind of labor-saving device. We can reason more easily about knowledge if we don't make things explicit, but when we need explicit justifications, they can be calculated.

Perhaps the major drawback to the present methods, and to all constructive methods in this area, is that computing provable realizations for theorems of logics of knowledge requires cut-free proofs. *Modus Ponens* is the stumbling block. It does not preserve polarities, and thus the methods of this paper do not apply to it. How to deal with this remains a major open question.

References

1. Artemov, S.: Explicit provability and constructive semantics. The Bulletin for Symbolic Logic **7**(1) (2001) 1–36
2. Gödel, K.: Eine Interpretation des intuitionistischen Aussagenkalküls. Ergebnisse eines mathematischen Kolloquiums **4** (1933) 39–40 English translation in [7].
3. Fitting, M.C.: The logic of proofs, semantically. Annals of Pure and Applied Logic **132** (2005) 1–25
4. Kuznets, R.: On self-referentiality in modal logic. The Bulletin of Symbolic Logic **12**(3) (2006) 510
5. Fitting, M.C.: A replacement theorem for LP. Technical report, CUNY Ph.D. Program in Computer Science (2006) `http://www.cs.gc.cuny.edu/tr/`.
6. Rubtsova, N.: Evidence reconstruction of epistemic modal logic S5. In Grigoriev, D., Harrison, J., Hirsch, E.A., eds.: Computer Science — Theory and Applications, Springer-Verlag (2006) 313–321 *LNCS, No. 3967.*
7. Feferman, S., Jr., J.D., Kleene, S.C., Moore, G.H., Solovay, R.M., Heijenoort, J.v., eds.: Kurt Gödel Collected Works. Volume I. Oxford University Press, New York (1986)

Successive Abstractions of Hybrid Automata for Monotonic CTL Model Checking

R. Gentilini[2], K. Schneider[2], and B. Mishra[1,3]

[1] Courant Institute, New York University, New York, NY, U.S.A.
[2] University of Kaiserslautern, Department of Computer Science, Germany
[3] NYU School of Medicine, New York University, New York, NY, U.S.A.
{gentilin,Klaus.Schneider}@informatik.uni-kl.de,
mishra@nyu.edu

Abstract. Current symbolic techniques for the automated reasoning over undecidable hybrid automata, force one to choose between the refinement of either an overapproximation or an underapproximation of the set of reachable states. When the analysis of branching time temporal properties is considered, the literature has developed a number of abstractions techniques based on the simulation preorder, that allow the preservation of only true universally quantified formulæ.

This paper suggests a way to surmount these difficulties by defining a succession of abstractions of hybrid automata, which not only (1) allow the detection and the refinement of both over- and under-approximated reachable sets symmetrically, but also (2) preserves the full set of branching time temporal properties (when interpreted on a dense time domain). Moreover, our approach imposes on the corresponding set of abstractions a desirable monotonicity property with respect to the set of model-checked formulaæ.

1 Introduction

Over the past few years, questions related to the analysis of *hybrid automata* [10] have occupied a considerable amount of attention and interest within the automatic verification research community, since the consequent models provide a high fidelity representation of real world (embedded) systems, and yet the nontrivial computational problems they raise do not yield to the classical techniques either of applied mathematics or of theoretical computer science.

As originally envisioned in [10,9], hybrid automata have aspired to combine the traditional *automata* tools from logic and computer science with differential equation systems, and their long tradition in mathematics. In this respect, the enormous potentials of hybrid automata in challenging applications fields—namely, the analysis of embedded, real time, and biological systems, to cite only a few of them—were immediately recognized. However, the trade-off between the representational fidelity of hybrid automata and the solvability of related decidability problems addressing properties such as *reachability*, was also immediately apparent. Hence, the major effort of the hybrid automata research community, to date, has been devoted to the study of *decidable* classes of hybrid automata, for which at least the reachability problem remains decidable [10,9,13,14,2]. Listed in their chronological order, the (main) decidable families in the literature are the ones corresponding to *timed automata* [1], *singular automata* [10,9], *rectangular automata* [9], and *o-minimal automata* [13]. Unfortunately, for each one of the above families, the sacrifice in the expressiveness of either the discrete or the continuous dynamics [2] that has to be exacted in exchange for the decidability result, strongly casts doubt on the possibility of faithfully capturing complex hybrid dynamics arising, for example, in the system biology area [16,8].

Motivated by the reasons listed above, many authors have recently focused on developing techniques for the symbolic analysis of undecidable—and yet reasonably expressive—hybrid automata [16,8,19,6,17]. However, any method developed so far relies either on the definition of abstractions simulating the underlying hybrid automata [8,19,17] or on symbolic bounded reachability techniques [16,6]. In the first case, only an overapproximation of the reachable state-space is possible. Usually, those techniques target the proof of *safety property*, stating that something undesirable should never happen on any reachable state of the system. In general, the simulation preorder from the abstraction to the hybrid automaton allows for preservation of only *true* formulæ in the *universal fragment* of a branching time temporal logic. In the second case, only an underapproximation of the reachable state-space can be explored and used for generating counterexamples to the reactive system properties of interest (e.g. safety).

In this paper we develop a framework to *both prove and disprove* reactive system properties expressed by means of CTL logic [4,18] on (undecidable) hybrid automata. To the best of authors' knowledge, no other symbolic technique for the analysis of undecidable hybrid automata can be claimed to preserve both true and false reactive systems properties simultaneously. Our framework is based on the design of a succession of abstraction and a corresponding three valued semantics for the logic CTL, allowing for the monotonic preservation of true/false formulæ along the succession of abstractions. Given a structure $\mathcal{A}$ in our succession, we finally show that the three valued CTL model checking problem on $\mathcal{A}$ is linear in the length of the formula and in the size of the abstraction. Because of the space constraints, we omit the proofs of the results shown here, but collect them in [7].

2 Preliminaries

In this section, we introduce the basic definitions and the notations used in the remainder of the paper.

Definition 1 (Hybrid Automata [2]). *A* Hybrid Automaton *is a tuple* $H = (L, E, X, Init, Inv, F, G, R)$ *with the following components:*

- *a finite set of* locations L
- *a finite set of* discrete transitions *(or* jumps*)* $E \subseteq L \times L$
- *a finite set of* continuous variables $X = \{x_1, \dots x_n\}$ *that take values in* $\mathbb{R}$
- *an initial set of conditions: Init* $\subseteq L \times \mathbb{R}^n$
- *Inv:* $L \mapsto 2^{\mathbb{R}^n}$*, the* invariant location labeling
- $F : L \times \mathbb{R}^n \mapsto \mathbb{R}^n$*, assigning to each location* $\ell \in L$ *a vector field* $F(\ell, \cdot)$ *that defines the evolution of continuous variables within* ℓ
- $G : E \mapsto 2^{\mathbb{R}^n}$*, the* guard edge labeling
- $R : E \times \mathbb{R}^n \mapsto 2^{\mathbb{R}^n}$*, the* reset edge labeling.

We write $\mathbf{v}$ to represent a valuation $(v_1, \ldots, v_n) \in \mathbb{R}^n$ of the variables' vector $\mathbf{x} = (x_1, \ldots, x_n)$, whereas $\dot{\mathbf{x}}$ denotes the first derivatives of the variables in $\mathbf{x}$ (they all depend on the time, and are therefore rather functions than variables). A *state* in H is a pair $s = (\ell, \mathbf{v})$, where $\ell \in L$ is called the *discrete component* of s and $\mathbf{v}$ is called the *continuous component* of s. A *run* of $H = (L, E, X, \mathit{Init}, \mathit{Inv}, F, G, R)$, starts at any $(\ell, \mathbf{v}) \in \mathit{Init}$ and consists of continuous evolutions (within a location) and discrete transitions (between two locations). Formally, a run of H is a path with alternating continuous and discrete steps in the *time abstract transition system* of H, defined below:

Definition 2. *The* time abstract transition system *of the hybrid automaton $H = (L, E, X, \mathit{Init}, \mathit{Inv}, F, G, R)$ is the transition system T_H =(Q,Q_0, $\ell_{\rightarrow}$, $\rightarrow$), where:*

- *$Q \subseteq L \times \mathbb{R}^n$ and $(\ell, \mathbf{v}) \in Q$ if and only if $\mathbf{v} \in \mathit{Inv}(\ell)$*
- *$Q_0 \subseteq Q$ and $(\ell, \mathbf{v}) \in Q_0$ if and only if $\mathbf{v} \in \mathit{Init}(\ell) \cap \mathit{Inv}(\ell)$*
- *$E \cup \{\delta\}$ is the set of edge labels, that are determined as follows:*
  - *there is a* continuous transition $(\ell, \mathbf{v}) \xrightarrow{\delta} (\ell, \mathbf{v}')$*, if and only if there is a differentiable function $f : [0, t] \rightarrow \mathbb{R}^n$, with $\dot{f} : [0, t] \rightarrow \mathbb{R}^n$ such that:*
    1. *$f(0) = \mathbf{v}$ and $f(t) = \mathbf{v}'$*
    2. *for all $\varepsilon \in (0, t)$, $f(\varepsilon) \in \mathit{Inv}(\ell)$, and $\dot{f}(\varepsilon) = F(\ell, f(\varepsilon))$.*
  - *there is a* discrete transition $(\ell, \mathbf{v}) \xrightarrow{e} (\ell', \mathbf{v}')$ *if and only if there exists an edge $e = (\ell, \ell') \in E$, $\mathbf{v} \in G(\ell)$ and $\mathbf{v}' \in R((\ell, \ell'), \mathbf{v})$.*

A region is a subset of the states Q of T_H =(Q,Q_0, $\ell_{\rightarrow}$, $\rightarrow$). Given a region B and a transition label $a \in \ell_{\rightarrow}$, the predecessor region $Pre_a(B)$ is defined as the region $\{q \in Q \mid \exists q' \in B. q \xrightarrow{a} q'\}$. The *bisimulation* and the *simulation* relations are two fundamental tools in the context of hybrid automata abstraction.

Definition 3 (Bisimulation). *Let $T^1 = (Q^1, Q^1_0, \ell^1_{\rightarrow}, \rightarrow^1)$, $T^2 = (Q^2, Q^2_0, \ell^2_{\rightarrow}, \rightarrow^2)$ be two edge-labeled transition systems and let $\mathcal{P}$ be a partition on $Q_1 \cup Q_2$. A* bisimulation *for T_1, T_2 is a nonempty relation on $\equiv_B \subseteq Q_1 \times Q_2$ such that, for all $p \equiv_B q$ it holds:*

- *$p \in Q^1_0$ iff $q \in Q^2_0$ and $[p]_{\mathcal{P}} = [q]_{\mathcal{P}}$, where $[p]_{\mathcal{P}}$ denotes the class of q in $\mathcal{P}$.*
- *for each label $a \in \ell_{\rightarrow}$, if there exists p' such that $p \xrightarrow{a} p'$, then there exists q' such that $p' \equiv_B q'$ and $q \xrightarrow{a} q'$.*
- *for each label $a \in \ell_{\rightarrow}$, if there exists q' such that $q \xrightarrow{a} q'$, then there exists p' such that $p' \equiv_B q'$ and $p \xrightarrow{a} p'$.*

If there exists a bisimulation relation for T_1, T_2, then T_1 and T_2 are bisimilation equivalent *(or* bisimilar*), denoted $T_1 \equiv_B T_2$.*

Definition 4 (Simulation). *Let $T^1 = (Q^1, Q^1_0, \ell^1_{\rightarrow}, \rightarrow^1)$, $T^2 = (Q^2, Q^2_0, \ell^2_{\rightarrow}, \rightarrow^2)$ be two edge-labeled transition systems and let $\mathcal{P}$ be a partition on $Q_1 \cup Q_2$. A* simulation *from T_1 to T_2 is a nonempty relation on $\leq_S \subseteq Q_1 \times Q_2$ such that, for all $p \leq_S q$:*

- *$p \in Q^1_0$ iff $q \in Q^2_0$ and $[p]_{\mathcal{P}} = [q]_{\mathcal{P}}$.*
- *for each label $a \in \ell_{\rightarrow}$, if there exists p' such that $p \xrightarrow{a} p'$, then there exists q' such that $p' \leq_S q'$ and $q \xrightarrow{a} q'$.*

If there exists a simulation from T_1 to T_2, then we say that T_2 simulates *T_1, denoted $T_1 \leq_S T_2$. If $T_1 \leq_S T_2$ and $T_2 \leq_S T_1$, then T_1 and T_2 are said to be* similation equivalent *(or* similar*) and we write $T_1 \equiv_S T_2$.*

Definition 6 recapitulates the semantics of the temporal logic CTL (where the neXt temporal operator is omitted because of the density of the underlying time framework) on hybrid automata [1,10].

Definition 5 (CTL for Hybrid Automta). *Let* AP *be a finite set of propositional letters and $p \in \mathsf{AP}$.* CTL *is the set of formulæ defined by the following syntax:*

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \mathsf{E}\phi_1\mathsf{U}\phi_2 \mid \mathsf{A}\phi_1\mathsf{U}\phi_2 \mid \mathsf{E}\phi_1\mathsf{R}\phi_2 \mid \mathsf{A}\phi_1\mathsf{R}\phi_2$$

Definition 6 (CTL Semantics). *Let $H = (L, E, X, Init, Inv, F, G, R)$ be a hybrid automaton, and let* AP *be a set of propositional letters. Consider $\ell_{\mathsf{AP}} : L \times X \mapsto 2^{\mathsf{AP}}$. Given $\phi \in \mathsf{CTL}$ and $s \in Q$, $s \models \phi$ is inductively defined as follows:*

- $s \models \mathsf{p}$ *if and only if* $\mathsf{p} \in \ell_{\mathsf{AP}}(s)$
- $s \models \neg\phi$ *if and only if not* $s \models \phi$
- $s \models \phi_1 \vee \phi_2$ *if and only if* $s \models \phi_1$ *or* $s \models \phi_2$
- $s \models \mathsf{E}\phi_1\mathsf{U}\phi_2$ *if and only if there exists a run ρ and a time t such that:*
  · $\rho(t) \models \phi_2$
  · $\forall t' \leq t\ (\rho(t') \models \phi_1 \vee \phi_2)$
- $s \models \mathsf{A}\phi_1\mathsf{U}\phi_2$ *if and only if for each run ρ there exists a time t such that:*
  · $\rho(t) \models \phi_2$
  · $\forall t' \leq t\ (\rho(t') \models \phi_1 \vee \phi_2)$
- $s \models \mathsf{E}\phi_1\mathsf{R}\phi_2$ *iff* $s \models \neg(\mathsf{A}\neg\phi_1\mathsf{U}\neg\phi_2)$
- $s \models \mathsf{A}\phi_1\mathsf{R}\phi_2$ *iff* $s \models \neg(\mathsf{E}\neg\phi_1\mathsf{U}\neg\phi_2)$

$H \models \phi$ iff for each $s \in Q_0$, $s \models \phi$.

2.1 O-Minimal Theories and O-Minimal Hybrid Automata

In this subsection, we give a brief introduction to *order minimality* (o-minimality) which is used to define o-minimal hybrid automata. We refer to [21,20,22] for a more comprehensive introduction to o-minimality.

Consider a structure over the reals, $\mathcal{M} = \langle \mathbb{R}, <, \ldots \rangle$, where the underlying language includes at least a binary relation interpreted as the usual total order over $\mathbb{R}$. The *theory* $\mathrm{Th}(\mathcal{M})$ associated to $\mathcal{M}$ is the set of first order sentences that hold in $\mathcal{M}$. A set $Y \subseteq \mathbb{R}^n$ is *definable* in $\mathcal{M}$ if and only if there exists a first order formula $\psi(x_1, \ldots, x_n)$ such that $Y = \{(a_1, \ldots, a_n) \mid \mathcal{M} \models \psi(a_1, \ldots, a_n)\}$. A map $f : A \mapsto \mathbb{R}^n$ with $A \subseteq \mathbb{R}^m$ is definable in $\mathcal{M}$ if and only if its graph $\Gamma(f) \subseteq \mathbb{R}^m \times \mathbb{R}^n$ is definable in $\mathcal{M}$.

Definition 7 (O-Minimal Structure). *The structure $\mathcal{M} = \langle \mathbb{R}, <, \ldots \rangle$ is o-minimal if and only if every definable subset of $\mathbb{R}$ is a finite union of points and (possibly unbounded) intervals. In this case, the theory $\mathrm{Th}(\mathcal{M})$ is also said to be o-minimal.*

Given an o-minimal structure $\mathcal{M} = \langle \mathbb{R}, <, \ldots \rangle$, the notion of set definability is closed under each boolean set composition operation, cartesian product, and projection. The notion of map definability is closed under composition, cartesian product, and projection. In the following, we will use the same symbol to denote both a given o-minimal structure and the corresponding theory, omitting the term $\mathrm{Th}(\cdot)$.

The class of o-minimal structures over the reals is quite rich: both the structure $\mathsf{Li}(\mathbb{R}) = (\mathbb{R}, <, +, -, 0, 1)$, used to express linear constraints over the reals, and the ordered real field $\mathsf{OF}(\mathbb{R}) = (\mathbb{R}, <, +, -, *, 0, 1)$ are o-minimal. The extensions of the above structures by the exponential function are also o-minimal. Another important extension is obtained by restricted analytic functions. Further extensions are discussed in [13]. The variety of o-minimal theories over the reals ensures that the family of *o-minimal hybrid automata* as introduced in [13,14] (cf. Definition 9, below) constitutes a large and important family of hybrid automata, admitting powerful continuous evolutions. In the following definitions, we will adopt the notation used in [13].

Definition 8. *Let $F : \mathbb{R}^n \mapsto \mathbb{R}^n$ be a smooth vector field on $\mathbb{R}^n$. For each $\mathbf{v} \in \mathbb{R}^n$, let $\gamma_{\mathbf{v}}(t)$ denote the integral curve of F which passes through $\mathbf{v}$ at $t = 0$, that is $\dot{\gamma}_{\mathbf{v}}(t) = F(\gamma_{\mathbf{v}}(t))$ and $\gamma_{\mathbf{v}}(0) = \mathbf{v}$. We say that F is* complete *if, for each $\mathbf{v} \in \mathbb{R}^n$, $\gamma_{\mathbf{v}}(t)$ is defined for all t. For such an F, the* flow *of F is the function $\phi : \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^n$ given by $\phi(\mathbf{v}, t) = \gamma_{\mathbf{v}}(t)$.*

Definition 9 (O-Minimal Hybrid Automata [13]). *The hybrid automaton $H = (L, E, X,$* Init, Inv, *$F, G, R)$ is* o-minimal *if the following holds:*

a) for each $\ell \in L$ the smooth vector field $F(\ell, \cdot)$ is complete
*b) for each $(\ell, \ell') \in E$, the reset function $R : E \mapsto \mathbb{R}^n$ does not depend on continuous variables (*constant resets*)*
c) for each $\ell \in L$ and $(\ell, \ell') \in E$, the sets $Inv(\ell)$, $R(\ell, \ell')$, $G(\ell)$, $Init(\ell)$, and the flow of $F(\ell, \cdot)$ are definable in the same o-minimal structure.

Given an o-minimal structure $\mathcal{M}$, the o-minimal hybrid automata induced by $\mathcal{M}$ are called o-minimal($\mathcal{M}$) hybrid automata. O-minimal hybrid automata admit a finite bisimulation quotient [13]. Computability of such a bisimulation quotient (and hence decidability) depends on the underlying o-minimal structure: in [14], the class of o-minimal ($\mathsf{OF}(\mathbb{R})$) hybrid automata and various subclasses of o-minimal ($\mathsf{OF}_{\mathsf{exp}}(\mathbb{R})$) automata were proven to be decidable with respect to reachability.

3 A Succession of Abstractions for Monotonic CTL Model Checking on Hybrid Automata

Throughout this section and Section 4, we solve the problem of defining a succession of abstractions for hybrid automata and a corresponding three valued semantics for CTL formulæ with the following property: whenever a CTL formula is true (false) on a given abstraction, its value is preserved on the hybrid automaton. Moreover, we require that the set of formulæ evaluating to $\perp$ (according to the three valued semantics over the abstractions) decreases monotonically in its size along the succession of abstractions.

Such a requirement is reminiscent of the usual *regularity* property for Kleene three valued logics [12] (and its many variants [5]), according to which the behavior of the third value is compatible with any increase of information[1].

The first idea we exploit is based on the classical notion of n*-bounded bisimulation*. In fact, a simulation preorder relates successively finer bounded bisimulations, thus enabling the establishment of a ”monotonic truth-preservation result” for the set of true universally quantified formulæ along this succession of abstractions. Since the n-bounded bisimulation is characterized by the n-bounded modal logic (where at most n neXt operators are admitted in the formulæ), it is possible to recover preservation for non universal CTL formulæ, by evaluating them on n-bounded paths. The preceding ideas can be effectively developed for the analysis of infinite *discrete* transition systems. However, in our context infinite transition systems represent mixed continuous/discrete systems. Hence, first the neXt temporal operator is meaningless in the corresponding logics. Second, a query such as $s \models \mathsf{E}\phi_1\mathsf{U}\phi_2$ can never be checked by considering only a finite path departing from s in the time abstract transition system T_H of a hybrid automaton H. In fact, due to the dense nature of the underlying time framework, each finite path of the kind $z \xrightarrow{\delta} z'$ subsumes an uncountable number of continuous transitions' infinite paths in T_H, on which $\mathsf{E}\phi_1\mathsf{U}\phi_2$ needs to be established. In other words, while for Kripke structures modelling (infinite) discrete dynamical systems, there is a nice correspondence between the index of the bounded bisimulation and the length of path that can be trusted, in the case of hybrid automata this is lost. More precisely, runs that can be trusted in successive finer bounded bisimulations could be never allowed to traverse more than two locations, since they are abstracted by paths containing more and more continuous transitions.

3.1 Discrete Bounded Bisimulation Abstraction

Motivated by the discussion above, we develop here a new succession of hybrid automata abstractions suitable for our purposes, and refer to them as *discrete bounded bisimulation* abstractions. It is well known that the classic bisimulation equivalence can be characterized as a coarsest partition stable with respect to a given transition relation [11]. Bounded bisimulation imposes a bound on the number of times each edge can be used for partition refinement purposes. For discrete bounded bisimulation, the latter bound applies only to discrete edges. Formally, our discrete bounded bisimulation abstractions are inductively defined in Definition 10.

Definition 10 (Discrete Bounded Bisimulation (DBB)). *Consider the time abstract transition system $T_H = (Q, Q_0, \ell_{\rightarrow}, \rightarrow)$ of a hybrid automaton H, and let $\mathcal{P}$ be a partition on Q:*

1. $\equiv_0 \in Q \times Q$ is the maximum relation on Q such that for all $p, q \in Q$:
if $p \equiv_0 q$ then (a) $[p]_{\mathcal{P}} = [q]_{\mathcal{P}}$ and $p \in Q_0$ iff $q \in Q_0$

$$(b)\ \forall p'(p \xrightarrow{\delta} p' \Rightarrow \exists q'(p' \equiv_0 q' \wedge q \xrightarrow{\delta} q')$$

$$(c)\ \forall q'(q \xrightarrow{\delta} q' \Rightarrow \exists p'(p' \equiv_0 q' \wedge p \xrightarrow{\delta} p')$$

[1] Note that, in our framework, the lack of information is inherent in the *abstraction* of the hybrid automaton model, rather than in the (indefinite) value of some propositional letter.

2. *Given $n \in \mathbb{N}^+$, $\equiv_n$ is the maximum relation on Q such that for all $p, q \in Q$:*
   if $p \equiv_n q$ then $(a)\ p \equiv_{n-1} q$
   $(b)\ \forall p'(p \xrightarrow{\delta} p' \Rightarrow \exists q'(p' \equiv_n q' \wedge q \xrightarrow{\delta} q')$
   $(c)\ \forall q'(q \xrightarrow{\delta} q' \Rightarrow \exists p'(p' \equiv_n q' \wedge p \xrightarrow{\delta} p')$
   $(d)\ \forall p'(p \xrightarrow{e} p' \Rightarrow \exists q'(p' \equiv_{n-1} q' \wedge q \xrightarrow{e} q')$
   $(e)\ \forall q'(q \xrightarrow{e} q' \Rightarrow \exists p'(p' \equiv_{n-1} q' \wedge p \xrightarrow{e} p')$

Given $n \in \mathbb{N}$, the relation $\equiv_n$ will be referred to as n-DBB equivalence.

Definition 11 (Succession of DBB Abstractions). *Let $T_H = (Q, Q_0, \ell_\rightarrow, \rightarrow)$ be the time abstract transition system of the hybrid automaton H, let $\mathcal{P}$ be a partition on Q, and consider the n-DBB equivalence $\equiv_n$. The n-DBB abstraction structure $H_{/\equiv_n} = (Q', Q'_0, \ell'_\rightarrow, \rightarrow)$ is defined as:*

– *$Q' = Q_{/\equiv_n}$, $Q'_0 = Q_{0/\equiv_n}$ and $\ell'_\rightarrow = \ell_\rightarrow$.*
– *$\forall \alpha, \beta \in Q'$:*
  • *$\alpha \xrightarrow{e} \beta$ iff $\exists s \in \alpha, \exists q \in \beta(s \xrightarrow{e} q))$*
  • *$\alpha \xrightarrow{\delta} \beta$ iff $\exists s \in \alpha, \exists q \in \beta(s \xrightarrow{\delta} q$ by traversing the only regions α and $\beta)$*

Lemma 1 establishes some folk theorems describing few properties of discrete bounded bisimulation, and can be easily proved using an inductive argument. Among them, we remark the existence of a simulation preorder relating successive elements in our succession of DBB abstractions. The latter property allows one to use the succession of DBB structures to refine an overapproximation of the underlying hybrid automaton reachable set.

Lemma 1. *Let H be a hybrid automaton, and consider the succession of n-DBB abstractions $\langle H_{/\equiv_n} \rangle_{n\in\mathbb{N}}$. For all $n \in \mathbb{N}$:*

– *$T_H \leq_S H_{/\equiv_n}$ and $H_{/\equiv_{n+1}} \leq_S H_{/\equiv_n}$.*
– *If $H_{/\equiv_n}$ coincides with $H_{/\equiv_{n+1}}$, then $T_H \equiv_B H_{/\equiv_n}$.*

As a consequence of Lemma 2, it is also possible to use the succession of DBB abstractions to obtain $\subseteq$-monotonic underapproximations of the underlying hybrid automaton reachable set. More precisely, $H_{/\equiv_n}$ preserves the reachability of a given region of interest (in the initial partition), whenever the latter can be established on H following a path that traverses at most n locations. Given two states in a hybrid automaton H, we use the notation $q \overset{n}{\rightsquigarrow} q'$ to state that q' is reachble from q following a run that contains at most n discrete edges (i.e. traverses at most n locations of H).

Lemma 2. *Let p and q be two states in a hybrid automaton H and let $\equiv_n$ be the n-DBB equivalence on T_H with respect to a partition $\mathcal{P}$. If $p \equiv_n q$, then for all $m \leq n$ it holds that:*

– *For all p' such that $p \overset{m}{\rightsquigarrow} p'$, there exists q' such that $p' \equiv_{n-m} q'$ and $q \overset{m}{\rightsquigarrow} q'$.*
– *For all q' such that $q \overset{m}{\rightsquigarrow} q'$, there exists p' such that $p' \equiv_{n-m} q'$ and $p \overset{m}{\rightsquigarrow} p'$.*

3.2 Finiteness and Computability of DBB Abstractions

Figure 1 presents a semi-decision procedure to obtain the n-DBB equivalence on the time-abstract transition system of a hybrid automaton H. Such a semi-decision procedure takes as input the hybrid automaton H, the bound n, and an initial (finite) partition $\mathcal{P}_0$ over the state-space of H. As stated in Lemma 3, it constitutes an effective algorithm for n-DBB equivalence whenever it is computable and gets to termination. Clearly, while computability depends on disposing of opportune symbolic techniques to represent and manipulate sets of states, termination is related to the n-DBB quotient finiteness.

Lemma 3. *Let $n \in \mathbb{N}$; let H be a hybrid automaton; and let $\mathcal{P}_0$ be a finite partition over the state-space of H. If* $\mathrm{DBB}(n, H, \mathcal{P}_0)$ *terminates, then algorithm* $\mathrm{DBB}(n, H, \mathcal{P}_0)$ *computes the quotient of the n-DBB equivalence with respect to $\mathcal{P}_0$ on T_H.*

Using a number of techniques developed in [13,14], it is rather easy to obtain n-DBB finiteness and computability results for the broad *undecidable* family of *fully o-minimal hybrid automata* (cfr. Definition 12). The latter extends the o-minimal based systems in [13] by admitting arbitrary o-minimal functions as resets, in place of constant functions. Such a relaxation in the formulation of the discrete dynamics allows the family to encompass several classes of hybrid automata for which the reachability problem has been proven undecidable (e.g. the class of uninitialized rectangular automata [10,9], or the undecidable classes studied in [3,15]).

Definition 12 (Fully O-Minimal Hybrid Automata). *The hybrid automaton $H = (L, E, X,$ Init, Inv, $F, G, R)$ is* fully o-minimal *iff:*

a) for each $\ell \in L$ the smooth vector field $F(\ell, \cdot)$ is complete
b) for each $\ell \in L$ and $(\ell, \ell') \in E$, the sets $Inv(\ell)$, $G(\ell)$, $Init(\ell)$, the reset function $R(\ell, \ell') : \mathbb{R}^{|X|} \mapsto \mathbb{R}^{|X|}$ and the flow of $F(\ell, \cdot)$ are definable in the same o-minimal structure.

Theorem 1. *Let H be a fully o-minimal($\mathcal{M}$) hybrid automaton, and let $\mathcal{P}$ be an initial finite partition over the state-space of H definable in $\mathcal{M}$. Then, the algorithm* $\mathrm{DBB}(n, H, \ell_Q)$ *terminates for any $n \in \mathbb{N}$.*

By Theorem 1, the whole family of fully o-minimal automata have for all $n \in \mathbb{N}$ a finite n-DBB abstraction structure. Computability of such a finite abstraction is instead parameterized with respect to the theory underlying fully o-minimal automata. In particular, as stated in Corollary 1, the class of fully o-minimal ($\mathsf{OF}(\mathbb{R})$) hybrid automata has a finite and effectively computable n-DBB abstraction. The result depends on the fact that $\mathsf{OF}(\mathbb{R})$ is a decidable theory admitting quantifier elimination. Therefore, the theory $\mathsf{OF}(\mathbb{R})$ provides the means for representing sets, computing post-images and boolean compositions, as well as checking for set emptiness. Techniques similar to the ones adopted in [13] can be used to obtain further computability results on subclasses of o-minimal($\mathsf{OF}_{\mathsf{exp}}(\mathbb{R})$) hybrid automata.

Corollary 1. *Given $n \in \mathbb{N}$, the n-DBB abstraction on fully o-minimal ($\mathsf{OF}(\mathbb{R})$) hybrid automata is finite and computable.*

```
DBB($n, H, \mathcal{P}_0$)
(1)  Let $\mathcal{P}$ be the coarsest partition refining $\mathcal{P}_0$ compatible with $Q_0$
/*————Compute 0-DBB equivalence quotient————————————*/
(2)  while $(\exists B, B' \in \mathcal{P}$ such that $\emptyset \neq B \cap Pre_\tau(B') \neq B)$
(3)     $B_1 \leftarrow B \cap Pre_\tau(B'); B_2 \leftarrow B \setminus Pre_\tau(B')$;
(4)     $\mathcal{P} \leftarrow (\mathcal{P} \setminus \{B\}) \cup \{B_1, B_2\}$;
/*————Perform $n$ refinement steps to obtain $n$-DBB equivalence quotient—*/
(5)  while $(n > 0)$
(6)     $n \leftarrow n - 1$; $\mathcal{P}_{old} \leftarrow \mathcal{P}$;
(7)     for each $(e = (\ell, \ell') \in E)$
(8)        for each $(B' \in \mathcal{P}_{old}, B \in \mathcal{P}$ such that $\emptyset \neq B \cap Pre_e(B') \neq B)$
(9)           $B_1 \leftarrow B \cap Pre_e(B'); B_2 \leftarrow B \setminus Pre_e(B')$;
(10)          $\mathcal{P} \leftarrow (\mathcal{P} \setminus \{B\}) \cup \{B_1, B_2\}$;
(11)    while $(\exists B, B' \in \mathcal{P}$ such that $\emptyset \neq B \cap Pre_\tau(B') \neq B)$
(12)       $B_1 \leftarrow B \cap Pre_\tau(B'); B_2 \leftarrow B \setminus Pre_\tau(B')$
(13)       $\mathcal{P} \leftarrow (\mathcal{P} \setminus \{B\}) \cup \{B_1, B_2\}$;
(14) return $\mathcal{P}$
```

Fig. 1. Partition refinement algorithm for n-DBB equivalence

Corollary 2. *Let H be a fully o-minimal($\mathsf{OF}_{\mathsf{exp}}(\mathbb{R})$) hybrid automaton in which:*

- *for each $\ell \in L$, the vector field is of the form $F(\ell, \mathbf{x}) = A\mathbf{x}$, where:*
  1. *$A \in \mathbb{Q} \times \mathbb{Q}$ is nilpotent or*
  2. *$A \in \mathbb{Q} \times \mathbb{Q}$ is diagonalizable with rational eigenvalues or*
  3. *$A \in \mathbb{Q} \times \mathbb{Q}$ has purely imaginary eigenvalues of the form ir, $r \in \mathbb{Q}$, with diagonal real Jordan form.*
- *for each $\ell \in L$ and $(\ell, \ell') \in E$, the sets $Inv(\ell)$, $G(\ell)$, $Init(\ell)$, and the reset function $R(\ell, \ell') : X \mapsto X$ are definable inside $\mathsf{OF}(\mathbb{R})$.*

Then, for all $n \in \mathbb{N}$, $\equiv_n$ is finite and computable on H.

4 3-Valued CTL Semantics over DBB Abstractions

In this section we introduce a 3-valued semantics for the logic CTL on n-DBB abstractions. Such a three valued semantics exploits, besides the inductive definition of DBB abstractions, the simulation preorder relating successive DBB abstractions in the succession $\langle H_{/\equiv_n}\rangle_{n\in\mathbb{N}}$. The latter allows us to use unbounded runs in the evaluation of (not purely existential) CTL properties. In other words, each CTL$\setminus$ECTL formula is not constrained to be evaluated by looking exclusively at paths in $H_{/\equiv_n}$ abstracting bounded runs of H, to obtain a value in $\{\mathsf{tt}, \mathsf{ff}, \bot\} \setminus \{\bot\}$. This key point needs to be emphasized, since it endows our framework with the abilities to handle both refutations as well as *proof* of safety or liveness properties over a hybrid automaton H. In fact, such properties intrinsically model some conditions that need to be maintained along the whole evolution of any run in H.

Definition 13. *Let H be a hybrid automaton having state space Q, let AP be a finite set of atomic propositions, and let $\mathcal{P}$ be the partition on Q induced by the labelling function $\ell_{\mathsf{AP}} : Q \mapsto 2^{|\mathsf{AP}|}$. Consider the n-DBB abstraction of H with respect to $\mathcal{P}$, $H_{/\equiv_n}$. Given the node $[s]_{\equiv_n}$ in $H_{/\equiv_n}$, the value $[[s]_{\equiv_n} \models_3 \phi] \in \{\mathsf{tt}, \mathsf{ff}, \bot\}$ is inductively defined on n as follows:*

– *If $\phi = \mathsf{p}$, then $[[s]_{\equiv_n} \models_3 \phi]$ is defined as:*

$$\begin{cases} \mathsf{tt} & \textit{iff} \quad \mathsf{p} \in \ell_{\mathsf{AP}}([s]_{\equiv_n})) \\ \mathsf{ff} & \textit{iff} \quad \mathsf{p} \notin \ell_{\mathsf{AP}}([s]_{\equiv_n})) \end{cases}$$

– *If $\phi = \neg\phi_1$, then $[[s]_{\equiv_n} \models_3 \phi]$ is defined as:*

$$\begin{cases} \mathsf{tt} & \textit{iff} \quad [[s]_{\equiv_n} \models_3 \phi_1] = \mathsf{ff}; \\ \mathsf{ff} & \textit{iff} \quad [[s]_{\equiv_n} \models_3 \phi_1] = \mathsf{tt}; \\ \bot & \textit{otherwise.} \end{cases}$$

– *If $\phi = \phi_1 \wedge \phi_2$, then $[[s]_{\equiv_n} \models_3 \phi]$ is defined as:*

$$\begin{cases} \mathsf{tt} & \textit{iff} \quad [[s]_{\equiv_n} \models_3 \phi_1] = \mathsf{tt} \wedge [[s]_{\equiv_n} \models_3 \phi_2] = \mathsf{tt}; \\ \mathsf{ff} & \textit{iff} \quad [[s]_{\equiv_n} \models_3 \phi_1] = \mathsf{ff} \vee [[s]_{\equiv_n} \models_3 \phi_2] = \mathsf{ff}; \\ \bot & \textit{otherwise} \end{cases}$$

– *If $\phi = \mathsf{E}\phi_1\mathsf{U}\phi_2$, then $[[s]_{\equiv_n} \models_3 \phi]$ is defined as:*

$$\begin{cases} \mathsf{tt} & \textit{iff there exists a path } \langle [s_i]_{\equiv_n} \rangle_{0 \le i \le k} \textit{ such that:} \\ & (1)\ \forall\, i < k\ ([s_i]_{\equiv_n} \xrightarrow{\delta} [s_{i+1}]_{\equiv_n}\ \wedge\ [[s_i]_{\equiv_n} \models_3 \phi_1 \vee \phi_2] = \mathsf{tt}) \\ & (2)\ [[s_k]_{\equiv_n} \models_3 \phi_2] = \mathsf{tt} \vee ([s_k]_{\equiv_n} \xrightarrow{e} [s_{k+1}]_{\equiv_n} \wedge [[s_{k+1}]_{\equiv_{n-1}} \models_3 \phi] = \mathsf{tt}) \\ \mathsf{ff} & \textit{iff for each path } \langle [s_i]_{\equiv_n} \rangle_{i \in \mathbb{N}} \textit{ it holds:} \\ & \forall\, i \in \mathbb{N}\ ([[s_i]_{\equiv_n} \models_3 \phi_2] = \mathsf{tt} \rightarrow (\exists\, j < i([[s_j]_{\equiv_n} \models_3 \neg\phi_1 \wedge \neg\phi_2] = \mathsf{tt}))) \\ \bot & otherwise \end{cases}$$

– *If $\phi = \mathsf{A}\phi_1\mathsf{U}\phi_2$, then $[[s]_{\equiv_n} \models_3 \phi]$ is defined as:*

$$\begin{cases} \mathsf{tt} & \textit{iff for each path } \langle [s_i]_{\equiv_n} \rangle_{i \in \mathbb{N}} \textit{ there exists an index } k \textit{ such that:} \\ & (1)\ \forall\, i < k\ ([[s_i]_{\equiv_n} \models_3 \phi_1 \vee \phi_2] = \mathsf{tt}) \\ & (2)\ [[s_k]_{\equiv_n} \models_3 \phi_2] = \mathsf{tt} \\ \mathsf{ff} & \textit{iff there exists a path } \langle [s_i]_{\equiv_n} \rangle_{i \le k} \textit{ such that:} \\ & (1)\ \forall\, i < k\ ([s_i]_{\equiv_n} \xrightarrow{\delta} [s_{i+1}]_{\equiv_n} \wedge [[s_i]_{\equiv_n} \models_3 \neg\phi_1 \vee \neg\phi_2] = \mathsf{tt}) \\ & (2)\ [[s_k]_{\equiv_n} \models_3 \neg\phi_1] = \mathsf{tt} \vee ([s_k]_{\equiv_n} \xrightarrow{e} [s_{k+1}]_{\equiv_n} \wedge [[s_{k+1}]_{\equiv_{n-1}} \models_3 \phi] = \mathsf{ff}) \\ \bot & otherwise \end{cases}$$

– *If $\phi = \mathsf{E}\phi_1\mathsf{R}\phi_2$, then $[[s]_{\equiv_n} \models_3 \phi] = [[s]_{\equiv_n} \models_3 \neg(\mathsf{A}\neg\phi_1\mathsf{U}\neg\phi_2)]$.*
– *If $\phi = \mathsf{A}\phi_1\mathsf{R}\phi_2$, then $[[s]_{\equiv_n} \models_3 \phi] = [[s]_{\equiv_n} \models_3 \neg(\mathsf{E}\neg\phi_1\mathsf{U}\neg\phi_2)]$.*

Finally, $H_{/\equiv_n} \models_3 \phi$ is defined as:

$$\begin{cases} \mathsf{tt} & \textit{iff} \quad \forall\, [s]_{\equiv_n} \in Q^0_{/\equiv_n}([[s]_{\equiv_n} \models_3 \phi_1] = \mathsf{tt}); \\ \mathsf{ff} & \textit{iff} \quad \exists\, [s]_{\equiv_n} \in Q^0_{/\equiv_n}([[s]_{\equiv_n} \models_3 \phi_1] = \mathsf{ff}); \\ \bot & otherwise \end{cases}$$

Theorem 2 (Preservation). *Let $H_{/\equiv_n}$ be the n-DBB abstraction for a hybrid automaton H, and let ϕ be a* CTL *formula. If* $[H_{/\equiv_n}\models_3\phi] = \mathsf{tt}$, *then* $H \models \phi$*; If* $[H_{/\equiv_n}\models_3\phi] = \mathsf{ff}$, *then* $\neg(H \models \phi)$.

Theorem 3 (Monotonicity). *Let* $\langle H_{/\equiv_n}\rangle_{n\in\mathbb{N}}$ *be the succession of DBB abstractions for a hybrid automaton H. For any* CTL *formula ϕ, for any $n \in \mathbb{N}$ it holds:*

$$([H_{/\equiv_n}\models_3\phi] = b \wedge b \in \{\mathsf{tt}, \mathsf{ff}\}) \rightarrow \forall\, m > n([H_{/\equiv_m}\models_3\phi] = b)$$

5 A Linear Algorithm for 3-Valued CTL Model Checking on Discrete Bounded Bisimulation Abstractions

In this Section we define an efficient algorithm for the three valued CTL model checking on bounded bisimulation abstractions, assuming the latter to be finite. Classical CTL model checking over Kripke structures is known to be linear in the size of the structure and in the length of the formula; analogously the complexity of our three valued model checking procedure is linear in the size of the abstraction and in the length of the formula.

5.1 The Case of 3-Valued ECTL∪ACTL Model Checking

We start solving a simpler problem: Namely, the definition of a procedure for the evaluation of $[H_{/\equiv_n}\models_3\phi]$, where ϕ is either a universal or an existential formula of CTL. Let ϕ be an ACTL formula, and let α be a node in $H_{/\equiv_n}$. According to Definition 13, $[\alpha\models_3\phi] = \mathsf{tt}$ iff $\alpha \models \phi$ (with respect to the classical 2-valued semantics for CTL on the finite transition system $H_{/\equiv_n}$). Hence, it is sufficient to use a classical (2-valued) CTL model checking algorithm on $H_{/\equiv_n}$ to detect those nodes in $H_{/\equiv_n}$ for which $[\alpha\models_3\phi] = \mathsf{tt}$ in time $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$.

The problem of collecting the nodes α in $H_{/\equiv_n}$ for which $[\alpha\models_3\phi] = \mathsf{ff}$ reduces to the problem of (1) rewriting $\neg\phi$ as a formula γ in ECTL (2) determining the set of states α in $H_{/\equiv_n}$ for which $[\alpha\models_3\gamma] = \mathsf{tt}$.

Summarizing the above observations, we can derive an overall $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$ algorithm for computing $H_{/\equiv_n}\models_3\phi$, if we prove that it is possible to recognize all those nodes α for which $[\alpha\models_3\gamma] = \mathsf{tt}$, $\gamma \in\mathsf{ECTL}$, in time $\mathcal{O}(|H_{/\equiv_n}| * |\gamma|)$.

Let γ be an ECTL formula: we solve the above subproblem by employing an efficient ($\mathcal{O}(|H_{/\equiv_n}|*|\gamma|)$) strategy to distribute on the nodes in $H_{/\equiv_n}$ the set of labels $\langle\psi, \mathsf{tt}, m\rangle$, where:

- ψ is a subformula of γ and $m \leq n$.
- $\alpha = [s]_n$ receives the label $\langle\psi, \mathsf{tt}, m\rangle$ if and only if

$$[[s]_m\models_3\psi] = \mathsf{tt} \wedge \forall\, m' < m([[s]_{m'}\models_3\psi] = \bot)$$

Our strategy uses a structural induction on γ. The cases in which γ is either a propositional letter or a boolean composition of subformulæ are easily dealt with (cfr. lines (1.1)–(2.4) of the procedure PROCESSE in Figure 2).

```
ALGO1($H_{/\equiv_n}, n, \phi$)
Input: $n \in \mathbb{N}$, the discrete $n$-bounded bisimulation abstraction $H_{/\equiv_n}$ for a hybrid automaton $H$,
        a formula $\phi \in \mathsf{ECTL} \cup \mathsf{ACTL}$
Output: $[H_{/\equiv_n} \models_3 \phi] \in \{\mathsf{tt}, \mathsf{ff}, \bot\}$

(1) Let $\gamma \equiv \neg\phi$, where $\gamma$ is in negation normal form
(2) if ($\phi \in \mathsf{ACTL}$) then PROCESSA($H_{/\equiv_n}, n, \phi$); PROCESSE($H_{/\equiv_n}, n, \gamma$)
(3)                     else PROCESSE($H_{/\equiv_n}, n, \phi$); PROCESSA($H_{/\equiv_n}, n, \gamma$)
(4) If ($\forall\alpha \in Init(H_{/\equiv_n})$ ($\langle\phi, \mathsf{tt}, -\rangle \in \ell(\alpha)$)) then return tt
(5) If ($\exists\alpha \in Init(H_{/\equiv_n})$ ($\langle\phi, \mathsf{ff}, -\rangle \in \ell(\alpha)$)) then return ff else return $\bot$
```

```
PROCESSE($H_{/\equiv_n}, n, \phi$)
Input: $n \in \mathbb{N}$, the discrete $n$-bounded bisimulation abstraction $H_{/\equiv_n}$ for a hybrid automaton $H$,
        a formula $\phi \in \mathsf{ECTL}$
Output: $\forall\ [s]_n in H_{/\equiv_n}$, $[s]_n$ is labeled $\langle\phi, \mathsf{tt}, m\rangle$ iff $[[s]_m \models_3 \phi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \phi] = \bot$

      case $\phi$ of
(1.1)   p      : for each $\alpha \in Q_{/\equiv_n}$ such that $lab_{\mathsf{p}}(\alpha) = \mathsf{tt}$ do $\ell(\alpha) \leftarrow \ell(\alpha) \cup \langle\mathsf{p}, \mathsf{tt}, 0\rangle$
(1.2)   ¬p     : for each $\alpha \in Q_{/\equiv_n}$ such that $lab_{\mathsf{p}}(\alpha) = \mathsf{ff}$ do $\ell(\alpha) \leftarrow \ell(\alpha) \cup \langle\neg\mathsf{p}, \mathsf{tt}, 0\rangle$

(2.1) $\phi_1 \Diamond \phi_2$ : for each $\alpha \in Q_{/\equiv_n}$ do
(2.2) $\Diamond \in \{\vee\wedge\}$        if $[\langle\phi_1, \mathsf{tt}, m_1\rangle \in \ell(\alpha)] \Diamond [\langle\phi_2, \mathsf{tt}, m_2\rangle \in \ell(\alpha)]$ then
(2.4)                              $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle\phi, \mathsf{tt}, \max(m_1, m_2)\rangle\}$

(3.1) $\mathsf{E}\phi_1\mathsf{U}\phi_2$ : / * Initialization * /
(3.2)         $N \leftarrow |H_{/\equiv_n}|$; $S_0 \leftarrow \emptyset; \ldots; S_n \leftarrow \emptyset$;  for each ($\alpha \in Q_{/\equiv_n}$) $color(\alpha) \leftarrow green$
(3.3)         Let $A_1[0 \ldots N]$, $A_2[0 \ldots N]$ be two array of lists of nodes in $Q_{/\equiv_n}$, such that $\alpha$
              belongs to the list $A_j[i]$ iff $\langle\phi_j, \mathsf{tt}, i\rangle \in \ell(\alpha)$
        / * For $i = 0 \ldots n$, build the set $S_i$ of nodes $\alpha = [s]_{\equiv_n}$ such that $[[s]_{\equiv_i} \models_3 \phi] = \mathsf{tt}$ and * /
        / * $\forall j < i [[s]_{\equiv_j} \models_3 \phi] = \bot$ * /
(3.4)         for each (node $\alpha$ in the list $A_2[0]$) do $S_0 \leftarrow S_0 \cup \{\alpha\}$
(3.5)         Use a breadth-first search like algorithm to discover and color red each node $\alpha$ such that
              $\alpha \overset{0}{\rightsquigarrow} S_0 \wedge \langle\phi_1, \mathsf{tt}, 0\rangle$ and augment $S_0$ with the nodes discovered.
(3.6)         for each ($\alpha \in pre(S_0)$) if ($color(\alpha) = green$) then $color(\alpha) \leftarrow yellow$
(3.7)         for ($i = 1 \ldots n$) do
(3.8)             for each (not red $\alpha$ in the list $A_2[0]$, yellow $\beta$ in the list $A_1[i]$) do $S_i \leftarrow S_i \cup \alpha$;
(3.9)             Use a breadth-first search like algorithm to discover and color red each $\alpha$ such that
                      $\alpha \overset{1}{\rightsquigarrow} (S_i \cup S_{i-1}) \wedge \langle\phi_1, \mathsf{tt}, -\rangle \in \ell(\alpha)$. Augment $S_i$ with the nodes discovered.
(3.10)            for each ($\alpha \in pre(S_i)$) if ($color(\alpha) = green$) then $color(\alpha) \leftarrow yellow$
        / * Assign the labels * /
(3.11)         for ($i = 1 \ldots n$) do
(3.12)             Assign the label $\langle\phi, \mathsf{tt}, i\rangle$ to each node in $S_i$

(4.1) $\mathsf{E}\phi_1\mathsf{R}\phi_2$ : Rewrite $\phi$ as $\mathsf{E}\phi_2\mathsf{U}\phi_1$ and use the rules for case (2)
```

```
PROCESSA($H_{/\equiv_n}, n, \phi$)
Input: $n \in \mathbb{N}$, the discrete $n$-bounded bisimulation abstraction $H_{/\equiv_n}$ for a hybrid automaton $H$,
        a formula $\phi \in \mathsf{ACTL}$
Output: $\forall\ [s]_n in H_{/\equiv_n}$, $[s]_n$ is labeled $\langle\phi, \mathsf{tt}, \bot\rangle$ iff $[[s]_m \models_3 \phi] = \mathsf{tt}$

/ * Use a 2-valued model checking procedure to process the formula $\phi$ on $H_{/\equiv_n}$ * /
(1.1)  for each ($\alpha \in H_{/\equiv_n}$) do
(1.3)      if $\alpha \models \mathsf{A}\phi_1\mathsf{U}\phi_2$ then $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle\phi, \mathsf{tt}, \bot\rangle\}$
```

Fig. 2. The linear algorithm for ACTL∪ECTL 3-valued Model Checking on $H_{/\equiv_n}$

The case for which $\gamma = \mathsf{E}\gamma_1\mathsf{U}\gamma_2$ requires instead more attention. As illustrated in Figure 2 (cfr. lines (3.1)–(3.12) of the procedure PROCESSE), given $N = |H_{/\equiv_n}|$, we first build the two vectors $A_1[1,\dots,N], A_2[1,\dots,N]$ of lists of nodes in $H_{/\equiv_n}$, where $\alpha \in A_j[i]$ iff the label $\langle\gamma_j, \mathsf{tt}, i\rangle$ has been inductively associated to α. Note that, $A_1[1\dots N]$ (resp. $A_2[1\dots N]$) requires space $\mathcal{O}(|H_{/\equiv_n}|)$, since the lists in each slot of the array A are disjoint. Then, by induction on $i = 0\dots n$, we build the sets $S_0,\dots,S_n$, where S_i contains all the nodes of $H_{/\equiv_n}$ that need to be labeled with $\langle\gamma, \mathsf{tt}, i\rangle$. More precisely, such a building process is supported by a coloring marking in which a node is red if it has been already assigned to some $S_{j<i}$; A node is yellow if it admits a transition to a red node; A node is green otherwise. Given the above coloring we let S_i to contain:

- Each not red node in the list $A_2[i]$.
- Each yellow node in the list $A_1[i]$.
- Each node $\beta \in H_{/\equiv_n}$ admitting a path p to S_{i-1} such that:
    1. p contains at most one edge labeled e
    2. the label $\langle\gamma_1 \vee \gamma_2, \mathsf{tt}, i\rangle$ is associated to each node of p.

Finally, if $\gamma = \mathsf{E}\gamma_1\mathsf{R}\gamma_2$, we can reduce to process the formula $\gamma' = \mathsf{E}\gamma_2\mathsf{U}\gamma_1$ according to the three-valued CTL semantics in Definition 13. The pseudocode of the overall algorithm above sketched for determining $[H_{/\equiv_n} \models_3 \phi]$, $\phi \in \mathsf{ECTL} \cup \mathsf{ACTL}$, is reported in Figure 2. Given the formula $\phi \in \mathsf{ECTL} \cup \mathsf{ACTL}$ in negation normal form, Theorem 4 states that our procedure ALGO1$(H_{/\equiv_n}, \phi)$ computes the value $[H_{/\equiv_n} \models_3 \phi]$ in time $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$ and space $\mathcal{O}(|H_{/\equiv_n}|)$.

Theorem 4. *The algorithm* ALGO1$(H_{/\equiv_n}, n, \phi)$ *computes* $[H_{/\equiv_n} \models_3 \phi] \in \{\mathsf{tt}, \mathsf{ff}\bot\}$ *in time* $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$ *and space* $\mathcal{O}(|H_{/\equiv_n}|)$.

5.2 3-Valued CTL Model Checking

In this Subsection we extend the techniques outlined in Subsection 5.1 to design a $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$ algorithm for computing $H_{/\equiv_n} \models_3 \phi$, where ϕ is a general CTL formula. We start with some preliminary observations to illustrate the bottlenecks related to the above extension. Let $\phi_1 = \mathsf{EpUq}$, let $\phi_2 = \mathsf{ApUq}$, and consider the two formulæ belonging to $\mathsf{CTL} \setminus (\mathsf{ECTL} \cup \mathsf{ACTL})$: $\psi_1 = \mathsf{ArU}\phi_1$ and $\psi_2 = \mathsf{ErU}\phi_2$.

Since ϕ_1, ϕ_2, and r belong to $\mathsf{ECTL} \cup \mathsf{ACTL}$, we can assume to have determined with ALGO1 the set of nodes α in $H_{/\equiv_n}$ for which $[\alpha \models_3 \varsigma] = \mathsf{tt}$, $\varsigma \in \{\phi_1, \phi_2, \mathsf{r}\}$. Then, the task of processing the formula $\psi_1 = \mathsf{ArU}\phi_1$ according to Definition 13 does not present any problem. In fact, determining each node β for which $[\beta \models_3 \phi] = \mathsf{tt}$ boils down to applying a *classical model checking* subprocedure targeting the operator AU (where the precomputed labelling is interpreted in a 2-valued fashion and $\bot$ corresponds to ff, according to a 'pessimistic view').

Instead, we face the following problem in processing the formula $\psi_2 = \mathsf{ErU}\phi_2$ above, according to the 3-valued semantics in Definition 13. Namely, for each $\alpha = [s]_{\equiv_n}$ such that $[\alpha \models_3 \phi_2] = \mathsf{tt}$, we *need to know the least* $m \leq n$ *for which* $[[s]_{\equiv_m} \models_3 \phi_2] = \mathsf{tt}$. If ϕ_2 were a formula having an *existential* main path quantifier, say $\phi_2 = \exists\phi_3\mathsf{U}\phi_4$, then such minimum indexes would have been dependent on the number of discrete

```
ALGO2($H_{/\equiv_n}, T_n, n, \phi, \gamma$)
Input: $n \in \mathbb{N}$, the discrete $n$-bounded bisimulation abstraction $H_{/\equiv_n}$, the compact partition tree $T_n$
        associated to $\equiv_n$, $\phi, \gamma \equiv \neg\phi \in$ CTL , where $\phi$ and $\gamma$ are in negation normal form
Output: $[H_{/\equiv_n} \models_3 \phi] \in \{\mathsf{tt}, \mathsf{ff} \bot\}$

(1.1)  if ($\phi \in \{\mathsf{p}, \neg\mathsf{p}\}$ for some $\mathsf{p} \in \mathsf{AP}$) then
(1.2)    for each $\alpha \in Q_{/\equiv_n}$ do
(1.3)      if $\ell_{\mathsf{p}}(\alpha) = \mathsf{tt}$ then $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle \mathsf{p}, \mathsf{tt}, 0\rangle\}$ else $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle \neg\mathsf{p}, \mathsf{tt}, 0\rangle\}$

(2.1)  if ($\phi = \phi_1 \diamondsuit \phi_2, \diamondsuit \in \{\vee \wedge\}$) then
(2.2)    for each $\alpha \in Q_{/\equiv_n}$ do
(2.3)      if $[\langle \phi_1, \mathsf{tt}, m_1\rangle \in \ell(\alpha)] \diamondsuit [\langle \phi_2, \mathsf{tt}, m_2\rangle \in \ell(\alpha)]$ then
(2.4)                        $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle \phi, \mathsf{tt}, \max(m_1, m_2)\rangle\}$

(3.1)  if ($\phi = \mathsf{E}\phi_1\mathsf{U}\phi_2 \wedge \gamma = \mathsf{A}\gamma_1\mathsf{R}\gamma_2$) then
(3.2)    ALGO2($H_{/\equiv_n}, T_n, n, \phi_1, \gamma_1$); ALGO2($H_{/\equiv_n}, T_n, n, \phi_2, \gamma_2$)
(3.3)    PROCESSEU($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$); PROCESSAR($H_{/\equiv_n}, T_n, n, \gamma_1, \gamma_2$)

(4.1)  if ($\phi = \mathsf{A}\phi_1\mathsf{U}\phi_2 \wedge \gamma = \mathsf{E}\gamma_1\mathsf{R}\gamma_2$) then
(4.2)    ALGO2($H_{/\equiv_n}, T_n, n, \phi_1, \gamma_1$); ALGO2($H_{/\equiv_n}, T_n, n, \phi_2, \gamma_2$)
(4.3)    PROCESSAU($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$); PROCESSER($H_{/\equiv_n}, T_n, n, \gamma_1, \gamma_2$)

(5.1)  if ($\phi = \mathsf{E}\phi_1\mathsf{R}\phi_2 \wedge \gamma = \mathsf{A}\gamma_1\mathsf{E}\gamma_2$) then
(5.2)    ALGO2($H_{/\equiv_n}, T_n, n, \phi_1, \gamma_1$); ALGO2($H_{/\equiv_n}, T_n, n, \phi_2, \gamma_2$)
(5.3)    PROCESSER($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$); PROCESSAU($H_{/\equiv_n}, T_n, n, \gamma_1, \gamma_2$)

(6.1)  if ($\phi = \mathsf{A}\phi_1\mathsf{R}\phi_2 \wedge \gamma = \mathsf{E}\gamma_1\mathsf{U}\gamma_2$) then
(6.2)    ALGO2($H_{/\equiv_n}, T_n, n, \phi_1, \gamma_1$); ALGO2($H_{/\equiv_n}, T_n, n, \phi_2, \gamma_2$)
(6.3)    PROCESSAR($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$); PROCESSEU($H_{/\equiv_n}, T_n, n, \gamma_1, \gamma_2$)

(7.1)  If ($\forall \alpha \in Init(Q_{/\equiv_n})$ ($\langle \phi, \mathsf{tt}, -\rangle \in \ell(\alpha)$)) then return tt
(7.2)  If ($\exists \alpha \in Init(Q_{/\equiv_n})$ ($\langle \gamma, \mathsf{tt}, -\rangle \in \ell(\alpha)$)) then return ff else return $\bot$
```

Fig. 3. The linear CTL 3-valued Model Checking on $H_{/\equiv_n}$

edges that one needs to traverse to evaluate ϕ_2 (assuming a previous labeling relative to ϕ_3 and ϕ_4[2]). However, formulæ having a main universal path quantifier are evaluated by considering any path in $H_{/\equiv_n}$, regardless of the number of discrete edges traversed. Our solution to recover the required minimum indexes for formulæ having a main universal path quantifier, is that of disposing of a data structure that we call *compact partition tree*, requiring space $\mathcal{O}(|H_{/\equiv_n}|)$. The formal description of a compact partition tree is given in Definition 14.

Definition 14 (Compact Partition Tree associated to $H_{/\equiv_n}$). *Let H be a hybrid automaton. The compact partition tree T_n associated to $H_{/\equiv_n}$ is inductively defined as:*

- *T_0 is a tree of depth one in which each leaf f is associated to a distinct node $\alpha \in H_{/\equiv_n}$ and is labeled with the triple $\ell(f) = \langle 0, 0, \alpha\rangle$.*
- *T_{n+1} is obtained from T_n processing each leaf f in T_n according to the following procedure:*

[2] In fact, the indices m are determined 'on the fly' in our procedure PROCESSE for ECTL.

- *If f is labeled with the triple $\langle m, n-1, \alpha\rangle$, and α is a class of $H_{/\equiv_n}$, then let $\ell(f) = \langle m, n, \alpha\rangle$.*
- *Otherwise, substitute the label of f with the pair $\ell(f) = \langle m, n-1\rangle$. For each $\alpha_i \subseteq \alpha, \alpha_i \in H_{/\equiv_n}$, create a new successor of f, f_i, and associate to f_i the label $\langle n, n, \alpha_i\rangle$.*

Lemma 4. *Let H be a hybrid automaton. The space complexity of the compact partition tree associated to $H_{/\equiv_n}$ is $\mathcal{O}(|H_{/\equiv_n}|)$.*

Compact partition trees can be easily computed along discrete bounded bisimulations. Let $\phi \in \mathsf{CTL}$, and assume that the main path quantifier in ϕ is universal. Suppose having determined the set of nodes $S = \{[s]_{\equiv_n} \in H_{/\equiv_n} \,|\, [[s]_{\equiv_n} \models_3 \phi] = \mathtt{tt}\}$. Then, the compact partition tree T_n can be used as follows to associate to each node $\alpha = [[s]_{\equiv_n} \in S$ (in time $\mathcal{O}(|H_{/\equiv_n}|)$) the required label $\langle \phi, \mathtt{tt}, m\rangle$, where m is the minimum index such that $[[s]_{\equiv_m} \models_3 \phi] = \mathtt{tt}$. First we use a depth first search-like algorithm to discover each node k of T_n satisfying the two conditions listed below:

1. any leaf of T_n which is a descendant of the node k is associated to a class $\alpha \in H_{/\equiv_n}$ for which $\alpha \models_3 \phi] = \mathtt{tt}$
2. k is a node of minimal depth having property 1.

If $[m, m']$ is the interval labeling k, then each class α associated to a leaf-descendant of k needs to be labeled by the triple $\langle \phi, \mathtt{tt}, m\rangle$.

The ideas sketched above lead to the final algorithm[3] reported in Figure 3, which computes $[H_{/\equiv_n} \models_3 \phi]$, where $\phi \in \mathsf{CTL}$, in time $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$ and space $\mathcal{O}(|H_{/\equiv_n}|)$.

Theorem 5. *The algorithm* ALGO2$(H_{/\equiv_n}, T_n, n, \phi, \gamma)$ *computes* $[H_{/\equiv_n} \models_3 \phi] \in \{\mathtt{tt}, ff\bot\}$ *in time* $\mathcal{O}(|H_{/\equiv_n}| * |\phi|)$ *and space* $\mathcal{O}(|H_{/\equiv_n}|)$.

6 Conclusions

This paper has proposed a novel framework to extend the power of automated reasoning over hybrid automata, even when those automata are undecidable in classical CTL. In this framework, it is possible to *both prove and disprove* reactive system properties expressed by means of CTL logic on (undecidable) hybrid automata. To the best of authors' knowledge, this research is novel and points to a fresh approach, as no other currently available symbolic technique analyzing undecidable hybrid automata can cope with both proofs and refutations of such general reactive systems properties as safety or liveness. The key ingredients of this innovative framework consist of proof systems, built upon a succession of abstractions and a corresponding three valued semantics for the CTL logic, which in turn allows for the monotonic preservation of true and false properties along these successive abstractions. This paper further proves that the new three valued model checking problem is not only decidable on our DBB abstractions, but is as efficient as the classical model checking of discrete Kripke structures, as it is linear in the length of the formula and in the size of the abstraction.

[3] The subprocedures used in the main algorithm ALGO2 are illustrated in the Appendix.

References

1. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
2. R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88:971–984, 2000.
3. T. Brihaye, C. Michaux, C. Rivie, and C. Troestler. On o-minimal hybrid systems. In *Proc. of the 7th Int. Workshop on Hybrid Systems*, pages 219–233, 2004.
4. E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.
5. M. Fitting. Kleene's three valued logics and their children. *Fundam. Inf.*, 20(1-3):113–131, 1994.
6. M. Fränzle. What will be eventually true of polynomial hybrid automata? In *Proc. of Int. Symp. on Theoretical Aspects of Computer Software*, volume 2215 of *LNCS*, pages 340–359. Springer, 2001.
7. R. Gentilini, B. Mishra, and K. Schneider. Successive abstractions of hybrid automata for monotonic CTL model checking. Technical report, Kaiserslautern University, 2007.
8. R. Ghosh, A. Tiwari, and C. Tomlin. Automated symbolic reachability analysis with application to delta-notch signaling automata. In *Hybrid Systems: Computation and Control HSCC*, volume 2623 of *LNCS*, pages 233–248. Springer, 2003.
9. T. Henzinger, P. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? In *Proc. of the 27th Symposium on Theory of Computing*, pages 373–382. ACM Press, 1995.
10. T. A. Henzinger. The theory of hybrid automata. In *Proc. of the 11th IEEE Symp. on Logic in Computer Science*, pages 278–292. IEEE Computer Society, 1996.
11. P. C. Kannellakis and S. A. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86(1):43–68, 1990.
12. S. C. Kleene. *Introduction to Metamathematics*. Wolters-Noordhoff, Groningen, 1971.
13. G. Lafferriere, G. Pappas, and S. Sastry. O-minimal hybrid systems. *Mathematics of Control, Signals, and Systems*, 13:1–21, 2000.
14. G. Lafferriere, J. Pappas, and S. Yovine. A new class of decidable hybrid systems. In *Proc. of the 2nd Int. Workshop on Hybrid Systems*, pages 137–151. Springer, 1999.
15. J. Miller. Decidability and complexity results for timed automata and semi-linear hybrid automata. In *Proc. of the 3th Int. Workshop on Hybrid Systems*, pages 296–309, 2000.
16. C. Piazza, M. Antoniotti, V. Mysore, A. Policriti, F. Winkler, and B. Mishra. Algorithmic algebraic model checking i: Challenges from systems biology. In *Proc. of the 17th Int. Conf. on Computer Aided Verification*, volume 3576 of *LNCS*, pages 5–19. Springer, 2005.
17. S. Ratschan and Z. She. Safety verification of hybrid systems by constraint propagation based abstraction refinement. In *Proc. of the 8th Int. Workshop on Hybrid Systems: Computation and Control*, volume 3414 of *LNCS*, pages 573–589. Springer, 2005.
18. K. Schneider. *Verification of Reactive Systems*. Springer Verlag, 2004.
19. A. Tiwari and G. Khanna. Series of abstractions for hybrid automata. In *Proc. of the 5th International Workshop on Hybrid Systems*, pages 465–478. Springer-Verlag, 2002.
20. L. van den Dries. O-minimal structures. In W. Hodges, editor, *Logic: From Foundations to Applications*, pages 99–108. Clarendon Press, 1996.
21. L. van den Dries. *Tame topology and o-minimal structures*, volume 248 of *London Math. Soc. Lecture Note Ser.* Cambridge University Press, 1998.
22. A. J. Wilkie. Schanuel conjecture and the decidability of the real exponential field. *Algebraic Model Theory*, pages 223–230, 1997.

Appendix

```
PROCESSAU($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$)
Input: For $\psi \in \{\phi_1, \phi_2\}$, each node $\alpha = [s]_n$ in $H_{/\equiv_n}$ is assumed to be labeled with $\langle \psi, \mathsf{tt}, m\rangle$
        if $[[s]_m \models_3 \psi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \psi] = \bot$
Out: $\forall\ [s]_n in H_{/\equiv_n}$, $[s]_n$ is labeled $\langle \phi = \mathsf{A}\phi_1\mathsf{U}\phi_2, \mathsf{tt}, m\rangle$ iff $[[s]_m \models_3 \phi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \phi] = \bot$

/ * Use a 2-valued model checking procedure targeting the AU operator to process the formula $\mathsf{A}\phi_1\mathsf{U}\phi_2$ * /
/ * assuming, for $\psi \in \{\phi_1, \phi_2\}$, that $\alpha \models \psi$ iff $\langle \psi, \mathsf{tt}, -\rangle \in \ell(\alpha)$ * /
(1.1)   for each ($\alpha \in H_{/\equiv_n}$) do
(1.3)           if $\alpha \models \mathsf{A}\phi_1\mathsf{U}\phi_2$ then $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle \mathsf{A}\phi_1\mathsf{U}\phi_2, \mathsf{tt}, -\rangle\}$
/ * Use a depth first search like algorithm on $T_n$ to determine the set of $T_n$ nodes $S$ * /
(1.4)   $S \leftarrow \{k \in T_n \mid$ any $\alpha \in H_{/\equiv_n}$ associated to a leaf descending from $k$ is labeled
                $\langle \mathsf{A}\phi_1\mathsf{U}\phi_2, \mathsf{tt}, -\rangle$, and $k$ is a node of minimal depth having the above property$\}$
(1.5)   for each ($\langle k \in S$, leaf $f$ descending from $k\rangle$) do
(1.6)       if ($\ell(f) = \langle -, -, \alpha\rangle \wedge \ell(k) = \langle m, -\rangle$)
(1.7)           then $\ell(\alpha) \leftarrow (\ell(\alpha) \setminus \{\langle \mathsf{A}\phi_1\mathsf{U}\phi_2, \mathsf{tt}, n\rangle\}) \cup \{\langle \mathsf{A}\phi_1\mathsf{U}\phi_2, \mathsf{tt}, m\rangle\}$
```

```
PROCESSEU($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$)
Input: For $\psi \in \{\phi_1, \phi_2\}$, each node $\alpha = [s]_n$ in $H_{/\equiv_n}$ is assumed to be labeled with $\langle \psi, \mathsf{tt}, m\rangle$
        if $[[s]_m \models_3 \psi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \psi] = \bot$
Out: $\forall\ [s]_n in H_{/\equiv_n}$, $[s]_n$ is labeled $\langle \phi = \mathsf{E}\phi_1\mathsf{U}\phi_2, \mathsf{tt}, m\rangle$ iff $[[s]_m \models_3 \phi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \phi] = \bot$

/ * Initialization * /
(1.1)   $N \leftarrow |H_{/\equiv_n}|; S_0 \leftarrow \emptyset; \ldots; S_n \leftarrow \emptyset$;  for each ($\alpha \in Q_{/\equiv_n}$) $color(\alpha) \leftarrow green$
(1.2)   Let $A_1[0 \ldots N]$, $A_2[0 \ldots N]$ be two array of lists of nodes in $Q_{/\equiv_n}$, such that $\alpha$ belongs to the list
        $A_j[i]$ iff $\langle \phi_j, \mathsf{tt}, i\rangle \in \ell(\alpha)$
/ * For $i = 0 \ldots n$, build the set $S_i$ of nodes $[s]_{\equiv_n}$ such that $[[s]_{\equiv_i} \models_3 \phi] = \mathsf{tt}$ and $\forall j < i [[s]_{\equiv_j} \models_3 \phi] = \bot$ * /
(1.3)   for each ($\alpha$ in the list $A_2[0]$) do $S_0 \leftarrow S_0 \cup \alpha$ endfor
(1.4)   Use a breadth-first search like algorithm to discover and mark each node $\alpha$ such that
        $\alpha \overset{0}{\rightsquigarrow} S_0 \wedge \langle \phi_1, \mathsf{tt}, 0\rangle \in \ell(\alpha)$ and augment $S_0$ with the nodes discovered.
(1.5)   for each ($\alpha \in pre(S_0)$) if ($color(\alpha) = green$) then $color(\alpha) \leftarrow yellow$
(1.6)   for ($i = 1 \ldots n$) do
(1.7)           for each (not red $\alpha$ in $A_2[i]$, yellow $\beta$ in $A_1[i]$) do $S_i \leftarrow S_i \cup \{\alpha, \beta\}$ endfor
(1.8)           Use a breadth-first search like algorithm to discover and color red each $\alpha$ such that
                $\alpha \overset{1}{\rightsquigarrow} (S_i \cup S_{i-1}) \wedge \langle \phi_1, \mathsf{tt}, i\rangle \in \ell(\alpha)$. Augment $S_i$ with the nodes discovered.
(1.9)           for each ($\alpha \in pre(S_i)$) if ($color(\alpha) = green$) then $color(\alpha) \leftarrow yellow$
/ * Assign the labels * /
(1.10)  for ($i = 1 \ldots n$) do Assign the label $\langle \phi, \mathsf{tt}, i\rangle$ to each node in $S_i$ endfor
```

```
PROCESSAR($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$)
Input: For $\psi \in \{\phi_1, \phi_2\}$, each node $\alpha = [s]_n$ in $H_{/\equiv_n}$ is assumed to be labeled with $\langle \psi, \mathsf{tt}, m\rangle$
        if $[[s]_m \models_3 \psi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \psi] = \bot$
Out: $\forall\ [s]_n in H_{/\equiv_n}$, $[s]_n$ is labeled $\langle \phi = \mathsf{A}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, m\rangle$ iff $[[s]_m \models_3 \phi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \phi] = \bot$

/ * Use a 2-valued model checking procedure targeting the AR operator to process the formula $\mathsf{A}\phi_1\mathsf{R}\phi_2$ * /
/ * assuming, for $\psi \in \{\phi_1, \phi_2\}$, that $\alpha \models \psi$ iff $\langle \psi, \mathsf{tt}, -\rangle \in \ell(\alpha)$ * /
(1.1)   for each ($\alpha \in H_{/\equiv_n}$) do
(1.3)           if $\alpha \models \mathsf{A}\phi_1\mathsf{R}\phi_2$ then $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle \mathsf{A}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, n\rangle\}$
/ * Use a depth first search like algorithm on $T_n$ to determine the set of $T_n$ nodes $S$ * /
(1.4)   $S \leftarrow \{k \in T_n \mid$ any $\alpha \in H_{/\equiv_n}$ associated to a leaf descending from $k$ is labeled
                $\langle \mathsf{A}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, -\rangle$, and $k$ is a node of minimal depth having the above property$\}$
(1.5)   for each ($\langle k \in S$, leaf $f$ descending from $k\rangle$) do
(1.6)       if ($\ell(f) = \langle -, -, \alpha\rangle \wedge \ell(k) = \langle m, -\rangle$)
(1.7)           then $\ell(\alpha) \leftarrow (\ell(\alpha) \setminus \{\langle \mathsf{A}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, n\rangle\}) \cup \{\langle \mathsf{A}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, m\rangle\}$
```

```
PROCESSER($H_{/\equiv_n}, T_n, n, \phi_1, \phi_2$)
Input: For $\psi \in \{\phi_1, \phi_2\}$, each node $\alpha = [s]_n$ in $H_{/\equiv_n}$ is assumed to be labeled with $\langle \psi, \mathsf{tt}, m\rangle$
        if $[[s]_m \models_3 \psi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \psi] = \bot$
Out: $\forall\ [s]_n in H_{/\equiv_n}$, $[s]_n$ is labeled $\langle \phi = \mathsf{E}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, m\rangle$ iff $[[s]_m \models_3 \phi] = \mathsf{tt} \wedge \forall k < m\ [[s]_k \models_3 \phi] = \bot$

(1.1)   PROCESSEU($H_{/\equiv_n}, T_n, n, \phi_2, \phi_1$)
(1.2)   for each ($\alpha \in H_{/\equiv_n}$) do
(1.3)           if $\langle \mathsf{E}\phi_2\mathsf{U}\phi_1, \mathsf{tt}, m\rangle \in \ell(\alpha)$ then $\ell(\alpha) \leftarrow \ell(\alpha) \cup \{\langle \mathsf{E}\phi_1\mathsf{R}\phi_2, \mathsf{tt}, m\rangle\}$
```

Fig. 4. The four subrocedures used within the algorithm ALGO2, in Figure 3

Explicit Proofs in Formal Provability Logic

Evan Goris

The Graduate Center of the City University of New York
365 Fifth Avenue, 10016 New York, NY, U.S.A.
evangoris@gmail.com

Abstract. In this paper we answer the question what implicit proof assertions in the provability logic GL can be realized by explicit proof terms. In particular we show that the fragment of GL which can be realized by generalized proof terms of GLA is exactly $\mathsf{S4} \cap \mathsf{GL}$ and equals the fragment that can be realized by proof-terms of LP. In the final sections of this paper we establish the disjunction property for GLA and give an axiomatization for $\mathsf{GL} \cap \mathsf{S4}$.

1 Introduction

One of the most striking applications of classical propositional modal logic to mathematics is without much doubt the interpretation of $\Box$ as 'provable in Peano Arithmetic PA' (for a neat introduction to the purpose of this see [6], we will aim here at a quick technical treatment). The normality axiom (1) below is a a most simple and clear example of a modal formula with an intuitively clear 'provable in PA' interpretation:

$$\Box(A \to B) \to (\Box A \to \Box B). \tag{1}$$

Clearly this scheme expresses the rule of modus ponens. The project of studying provability (and other meta-mathematical notions) in an axiomatic setting using modal logic, originally suggested by Gödel, really came to flourish after the arithmetical completeness theorem of Solovay [15]. This theorem identifies the logic GL as the logic of provability, see also [8]. GL is a remarkable system of modal logic that not only proofs Gödel's second incompleteness theorem, and more generally a formalized version of Löb's Theorem, but even satisfies a fixed-point theorem, very much in the spirit of Gödel's fixed-point lemma but in a purely propositional setting.

However, Gödel originally suggested the modal logic S4 as the logic of provability. This is indeed a most natural candidate for a provability logic but as it turns out incompatible with GL (the least normal modal logic extending both is the inconsistent one). Basically when the $\Box$ is read as *provable* the schemes expressed by S4 are both too strong (reflection) and too weak (no Löb's Theorem).

Artemov's Logic of Proofs LP was invented to tackle this problem [5,1]. In LP the $\Box$'s are replaced by proof-terms and the notion under study switched from *is provable* to *is proved by*. These proof-terms are build up from axiom-constants, proof-variables and function symbols that represent effective operations on proofs. For example there is a binary function symbol $\cdot$ that constructs from a proof x of $A \to B$ and a proof y of A a proof $x{\cdot}y$ of B. Which gives a means to express the rule of modus ponens, just as (1) but now in an explicit way:

$$x{:}(A \to B) \to (y{:}A \to (x{\cdot}y){:}B).$$

There are natural translations between modal formulas and LP-formulas. In one direction we can 'forget' the proof-terms in an LP formula by substituting $\Box$'s for them (forgetful-projection) and in the other direction we can substitute proof-terms for the $\Box$'s in a modal formula (realizations). The link of LP with S4 is as follows. For any theorem F of LP, the forgetful projection of F is a theorem of S4 and for any theorem A of S4 there exists a realization of A that is a theorem of LP. The latter is nicely formulated as LP can *realize* all theorems of S4. This, together with the arithmetical completeness theorem for LP does give a provability reading to S4 for which S4 is complete.

In [16] and [12] (cf. also [4]) the axiomatic study of provability (Provability Logic) and the axiomatic study of proofs (Logic of Proofs) are combined in a single logic that contains both the $\Box$ for formal provability and proof-terms for explicit proofs. The logic LPP from [16] contains a richer language of proof-terms than LP. In [12] this is shown to be not necessary, there an arithmetically complete logic GLA [1] has been recovered that has exactly the same term language as LP.

Recently F. Montagna posed the question whether GLA allows for the realization of more modal formulas than just S4 (given what we know about LP it is immediate that GLA realizes at least S4). A negative answer to this question is the main contribution of this paper.

This paper is organized as follows. In Section 2 we define the Logic of Proofs LP. In Section 3 we define the Logic of Proofs and Formal Provability GLA and formulate the main research question addressed in this paper. In Section 4 we give an answer to these questions. In the final sections of this paper we consider some related issues and give some directions for further research.

2 Logic of Proofs

See [3] for an extensive overview of LP. Here we only state the basic definitions and theorems relevant to this paper. The language of LP is two-sorted. We have *proofs terms* that are build up using

- Countably many proofs variables $x, y, z, \ldots$ and countably many axiom constants $a, b, c, \ldots$

and two binary function symbols $+$, $\cdot$ and a unary one !:

- if t and s are proof terms then so are $t{+}s$, $t{\cdot}s$ and $!t$.

[1] GLA was first introduced (under the name LPGL) supplied with Kripke-style semantics and proved to be arithmetically complete in E. Nogina's part of [4].

And we have LP *formulas*, which are generated by the following clauses.

- 'Propositional Logic',
- If F is a formula and t a proof term then $t{:}F$ is a formula.

We say that an LP formula F is *normal* when all negative occurrences off subformulas $t{:}G$ of F are of the form $x{:}G$, where x is a proof variable.

The logic LP is axiomatized by the following schemata and rules.

A0 'Classical Propositional Logic' (with Modus Ponens),
A1 $t{:}A \to A$,
A2 $s{:}(A \to B) \to t{:}A \to (s{\cdot}t){:}B$,
A3 $s{:}A \to (s+t){:}A$, $s{:}A \to (t+s){:}A$,
A4 $t{:}A \to !t{:}(t{:}A)$,
A5 $c{:}A$, c an axiom constant and A an instance of **A0-A4**.

If F is an LP formula then its *forgetful projection* F° is obtained by replacing all the proof terms by $\Box$'s. More formally:

- $p^\circ \equiv p$ and $\bot^\circ \equiv \bot$,
- $(A \to B)^\circ \equiv (A^\circ \to B^\circ)$,
- $(t{:}A)^\circ \equiv \Box(A^\circ)$.

A *realization* of a modal formula F is an LP formula G for which $G^\circ \equiv F$.

One of the fundamental theorems concerning LP is the following.

Theorem 1 (Artemov[1]). S4 $\vdash G$ *iff there exists a normal F such that* LP $\vdash$ *F and $F^\circ \equiv G$.*

Anther fundamental theorem concerning LP is its arithmetical completeness theorem [1]. See also [2,9]. In the spirit of the arithmetical reading of modal formulas $\Box F$ as 'F is provable' ([15,8]), formulas of the form $t{:}F$ are read as 't is a proof of F'. By Theorem 1 this provides us with a natural provability semantics for modal logic for which S4 is complete. Given this interpretation of LP it is natural to consider a system that includes both expressions of the form $\Box F$ and $t{:}F$. This has been done in detail in [16,12]. However natural liftings of Theorem 1 have not been addressed yet and one of those liftings is the main topic of this paper.

3 The System GLA

In this section we present the logic GLA from [12] and formulate two questions that will be answered in the remainder of the paper.

A joint logic of formal provability and explicit proofs has first been studied in [16]. The logic there however has a richer language of explicit proofs than LP. In [12] (cf. also [4]) a logic GLA, also a joint logic of formal provability and explicit proofs has been recovered in which the language of explicit proofs is exactly that of LP. This is the system we will study here.

The language of GLA is that of LP enriched with the modal operator $\Box$. The formulas of the system GLA are generated by the following rules.

- 'Propositional Logic',
- if A is a formula and t is a proof term then $t{:}A$ is a formula,
- if A is a formula then $\Box A$ is a formula.

The logic GLA is axiomatized by the following axiom schemata and rules.

- 'Classical Propositional Logic',
- Provability Logic GL:
  L1 $\Box(A \to B) \to (\Box A \to \Box B)$,
  L2 $\Box A \to \Box\Box A$,
  L3 $\Box(\Box A \to A) \to \Box A$,
  - $\vdash A$ implies $\vdash \Box A$.
- Logic of Proofs LP:
  A1 $t{:}A \to A$,
  A2 $s{:}(A \to B) \to t{:}A \to (s{\cdot}t){:}B$,
  A3 $s{:}A \to (s+t){:}A$, $s{:}A \to (t+s){:}A$,
  A4 $t{:}A \to !t{:}(t{:}A)$,
  A5 $c{:}A$, c an axiom constant and A an instance of **A0-A4**, **L1-L3** or **C1-C3**.
- Connecting principles:
  C1 $t{:}A \to \Box A$,
  C2 $\neg t{:}A \to \Box\neg t{:}A$,
  C3 $t{:}\Box A \to A$.

Notice that **A5** is richer than its analog in LP. The forgetful projection of an LP formula obviously generalizes to GLA formulas by setting $(\Box A)^\circ \equiv \Box A^\circ$. The following question about GLA will be addressed.

> For which modal formulas A can we find $\Box$-free GLA formulas B with $B^\circ \equiv A$ and GLA $\vdash B$.

As we will argue in the next subsection, the obvious generalization of the forgetful projection to GLA formulas as given above does not give us much to work with for solving this question. But first we finish this section with a few lemmata from [12] that will be of interest later.

In what follows we write

$$\mathsf{GLA}:\ X_1, \ldots, X_n \vdash Y_1, \ldots, Y_k$$

for the assertion that $Y_1 \vee \cdots \vee Y_k$ is provable using modus ponens only from the theorems of GLA and $X_1, \ldots, X_n$.

Lemma 1. *For any formula A there exists a term t such that*

$$\mathsf{GLA} \vdash x{:}A \to t{:}\Box A$$

Proof. We have GLA $\vdash c{:}(x{:}A \to \Box A)$ and GLA $\vdash x{:}A \to !x{:}(x{:}A)$. Thus GLA $\vdash x{:}A \to (c{\cdot}!x){:}\Box A$

Lemma 2 (Constructive necessitation). *If* $\mathsf{GLA} \vdash A$ *then for some ground term* t *we have* $\mathsf{GLA} \vdash t{:}A$.

Proof. Induction on a GLA derivation of A. If A is one of the axioms other than **A5** we can take any axiom constant for t. If A is an instance of **A5**, say $A \equiv a{:}B$, then we can take $!a$ for t. Suppose A is obtained by modus ponens from $B \to A$ and B. Thus $\mathsf{GLA} \vdash B \to A$ and $\mathsf{GLA} \vdash B$. By (IH) we have terms t_1 and t_2 such that $\mathsf{GLA} \vdash t_1{:}(B \to A)$ and $\mathsf{GLA} \vdash t_2{:}B$. And thus for t we can take $t_1{\cdot}t_2$. Suppose that A is obtained from B using necessitation. Thus $\mathsf{GLA} \vdash B$. By (IH) we have $\mathsf{GLA} \vdash t{:}B$ for some t. By Lemma 1 we that have for some s that $\mathsf{GLA} \vdash s{:}\Box B$.

Lemma 3 (Lifting lemma). *If*

$$\mathsf{GLA}: \; x_1{:}X_1, \ldots, x_n{:}X_n \vdash Y$$

then for some term t *we have*

$$\mathsf{GLA}: \quad x_1{:}X_1, \ldots, x_n{:}X_n \vdash t{:}Y.$$

Moreover the proof-variables in t *are all among* $\{x_1, \ldots, x_n\}$.

Proof. Induction on a derivation of Y from the $x_i{:}X_i$'s. If Y is one of the X_i's, say X_{i_0} then for t we can take x_{i_0}. If Y is a theorem of GLA the required t is given by Lemma 2. The inductive case when Y is obtained by modes ponens from previously obtained formulas is similar as in Lemma 2.

3.1 The Trouble with the Forgetful Projection in GLA

Obviously, since LP is a sub-system of GLA we have that $\mathsf{LP} \vdash A$ implies $\mathsf{GLA} \vdash A$. And thus in particular by Theorem 1 we have the following. (Recall that an LP formula is normal when all negative occurrences of proof-terms are variables, we use the same terminology for the more general formulas of GLA.)

Theorem 2. *If* $\mathsf{S4} \vdash A$ *then for some normal* B *with* $B^\circ \equiv A$ *we have* $\mathsf{GLA} \vdash B$.

It is also true that GLA does not realize all the theorems of GL. For suppose that for some terms t and r we have

$$\mathsf{GLA} \vdash x{:}(r{:}\bot \to \bot) \to t{:}\bot.$$

Since $\mathsf{GLA} \vdash c{:}(r{:}\bot \to \bot)$ we thus have $\mathsf{GLA} \vdash t[x/c]{:}\bot$ and by reflection $\mathsf{GLA} \vdash \bot$. A contradiction.

As we will see below the theorems of GL that can be realized in GLA are precisely those formulas that are also theorems of $\mathsf{S4}$. Clearly to show this it suffices to show the the other direction of Theorem 2, this however is less straightforward then in the $\mathsf{S4}/\mathsf{LP}$ case. One easily sees that if $\mathsf{LP} \vdash A$ then $\mathsf{S4} \vdash A^\circ$. If we however in the most straightforward way extend the definition of forgetful projection to formulas in the language of GLA, then the set of theorems of GLA

under this projection is not even closed under modus ponens. For the following three formulas are theorems of GLA.

- $\Box(\Box p \to p) \to \Box p$,
- $x{:}p \to p$,
- $\Box(x{:}p \to p)$.

Their forgetful projections are respectively

- $\Box(\Box p \to p) \to \Box p$,
- $\Box p \to p$,
- $\Box(\Box p \to p)$.

From which p follows using modus ponens. Obviously p is not the forgetful projection of any theorem of GLA. The trick is to not study the 'plain' forgetful projection but a variant that remembers which $\Box$'s came from proof terms and which where already there. This is what we will carry out in the coming sections.

4 The System EI

In this section we will show that only the theorems of S4 can be realized in GLA. The main tool is a modal propositional logic with two modalities $\Box$ and $\boxtimes$. In particular we will be interested in the modal formulas in this language that constitute images of the following generalization of forgetful projection to GLA formulas.

Definition 1 (Forgetful projection). *For an* GLA *formula A we define the* forgetful projection A° *with induction on A as follows.*

- $p^\circ \equiv p$ *and* $\bot^\circ \equiv \bot$,
- $(A \to B)^\circ \equiv (A^\circ \to B^\circ)$,
- $(\Box A)^\circ \equiv \Box(A^\circ)$,
- $(t{:}A)^\circ \equiv \boxtimes(A^\circ)$.

Let EI (for Explicit/Implicit) be the normal bi-modal logic axiomatized by the following axiom schemata and rules.

CP 'Classical Propositional Logic',
L1 $\Box(A \to B) \to (\Box A \to \Box B)$,
L2 $\Box A \to \Box\Box A$,
L3 $\Box(\Box A \to A) \to \Box A$,
S1 $\boxtimes(A \to B) \to (\boxtimes A \to \boxtimes B)$,
S2 $\boxtimes A \to \boxtimes \boxtimes A$,
S3 $\boxtimes A \to A$,
C1 $\boxtimes A \to \Box A$,
C2 $\neg \boxtimes A \to \Box \neg \boxtimes A$,
C3 $\boxtimes \Box A \to A$,
R $\vdash A$ implies $\vdash \boxtimes A$.

Lemma 4. GLA $\vdash A$ *implies* EI $\vdash A^\circ$.

Proof. Induction on a GLA derivation of A. If A is an instance of **A5** then A is of the form $c{:}B$, B° is an axiom of EI and $\boxtimes B^\circ (\equiv (c{:}B)^\circ)$ is derivable using $\boxtimes$ necessitation. If A is an instance of any of the other axiom schemata then A° is an axiom of EI. Suppose $A \equiv \Box B$, and the last step in the derivation of A is necessitation. By (IH) we obtain $\mathsf{EI} \vdash B^\circ$. By $\boxtimes$ necessitation we obtain $\mathsf{EI} \vdash \boxtimes B^\circ$ and by **C1** and modus ponens we get $\mathsf{EI} \vdash \Box B^\circ$. The case 'the last step is modus ponens' is trivial.

Next we formulate a Kripke semantics for EI. A binary relation R is *conversely well-founded* when every R *increasing path* $x_0 R x_1 R x_2 \cdots$ is finite.

Definition 2 (EI-frame). *A bi-modal Kripke frame $\langle W, R^\Box, R^\boxtimes \rangle$ is an* EI-frame *if*

1. *$R^\Box$ is transitive and conversely well-founded,*
2. *$R^\boxtimes$ is transitive and reflexive,*
3. *$xR^\Box y$ implies $xR^\boxtimes y$,*
4. *$xR^\Box y$ and $xR^\boxtimes z$ implies $yR^\boxtimes z$,*
5. *for all x there exists y such that $xR^\boxtimes y$ and $yR^\Box x$.*

Notice that no finite frames satisfying both 1 and 5 exist. For if 5 holds then one inductively constructs a sequence

$$x_1 R^\boxtimes x_2 R^\boxtimes x_3 \cdots$$

For which we in addition have $\cdots x_3 R^\Box x_2 R^\Box x_1$. Thus by transitivity of $R^\Box$ we have for all $i < j$

$$x_j R^\Box x_i.$$

But if the frame is finite then for some $i < j$ we must have $x_i = x_j$, contradicting the conversely well-foundedness of $R^\Box$.

Nevertheless, as is shown in [10], EI can be embedded in a sub-logic that does have finite models and is complete for a class of finite frames.

Lemma 5 (Modal soundness). *If $\mathsf{EI} \vdash A$ then A is valid on any EI-frame.*

Proof. As usual it suffices to show the lemma for A an axiom of EI. All instances of GL and $\mathsf{S4}$ are well-known to hold because of properties 1 and 2 of EI-frames.

We show that $\boxtimes A \to \Box A$ is valid. Suppose $w \Vdash \boxtimes A$ and suppose $wR^\Box x$. By 3 we have $wR^\boxtimes x$ and thus $x \Vdash A$.

To show that also $\boxtimes A \to \Box \boxtimes A$ is valid, suppose that in addition $xR^\boxtimes y$ then by 2 $wR^\boxtimes y$ and thus $y \Vdash A$ as well.

Now we show that $\neg \boxtimes A \to \Box \neg \boxtimes A$ is valid. Suppose $w \Vdash \neg \boxtimes A$ and $wR^\Box x$. For some y with $wR^\boxtimes y$ we have $y \Vdash \neg A$. By 4 we have $xR^\boxtimes y$ and thus $x \Vdash \neg \boxtimes A$.

Finally we show that $\boxtimes \Box A \to A$ is valid. Suppose $w \Vdash \boxtimes \Box A$. By 5 there exists some x with $wR^\boxtimes x$ and $xR^\Box w$. We thus have $x \Vdash \Box A$ and thus $w \Vdash A$.

We aim at showing that the $\Box$-free fragment of EI coincides with $\mathsf{S4}$. One direction, namely that $\mathsf{S4}$ is a subset of the $\Box$-free fragment is obvious. For the other direction we will make use of the completeness of $\mathsf{S4}$ with respect to transitive and reflexive Kripke frames [7]. We will use bounded morphisms to connect these frames with our EI-frames.

Definition 3 (Bounded morphism). *Let M and M' be two Kripke models. A* bounded morphism *from M to M' is a surjective mapping $M \longrightarrow M'$ such that for all $x, y \in M$ we have*

- *$x \Vdash p$ iff $f(x) \Vdash' p$,*
- *xRy implies $f(x)R'f(y)$,*
- *If $f(x)R'y'$ then for some y we have $f(y) = y'$ and xRy.*

The following lemma is well-known, see [7].

Lemma 6. *If f is a bounded morphism from M to M' then for all $w \in M$, M, w is bi-similar with $M', f(w)$. Consequently for any formula A, $M \models A$ iff $M' \models A$.*

Proposition 1. *For any transitive and reflexive Kripke model M there exists an EI model $M_\omega = \langle W_\omega, R^\Box, R^\boxtimes, \Vdash\rangle$ such that there exists a bounded morphism from $\langle W_\omega, R^\boxtimes, \Vdash\rangle$ to M.*

Proof. Let $M = \langle W, R, \Vdash\rangle$ be an $\mathsf{S4}$ model. Let W_1, W_2, ... be countably many disjoint copies of W. For $x \in W$ we denote with x_i the copy of x in W_i. Define $M_\omega = \langle W_\omega, R^\boxtimes, R^\Box, \Vdash\rangle$ as follows.

- $W_\omega = \bigcup_{i \geq 1} W_i$,
- $x_i R^\boxtimes y_j$ iff xRy,
- $x_i R^\Box y_j$ iff $x = y$ and $j < i$,
- $x_i \Vdash p$ iff $x \Vdash p$.

First we will show that M_ω is based on an EI-frame. That $R^\boxtimes$ is transitive and reflexive is immediate. That $R^\Box$ is transitive and conversely well-founded is also easy to see. Suppose that $x_i R^\Box y_j$. Then we thus have in particular that $x = y$ and by reflexivity of R we get $x_i R^\boxtimes y_j$. Suppose that $x_i R^\Box y_j$ and $x_i R^\boxtimes z_k$. We have to show that yRz. But this follows immediately since from our assumptions it follows that $y = x$ and xRz. Let $x_i \in W_\omega$. We have to show that for some $y_j \in W_\omega$ we have $x_i R^\boxtimes y_j$ and $y_j R^\Box x_i$. Just take $y_j = x_{i+1}$. This completes the proof that M_ω is based on an EI-frame.

We show that the mapping f defined by $f(x_i) = x$ is a bounded morphism from $\langle W_\omega, R^\boxtimes, \Vdash\rangle$ to M. f is clearly surjective and by definition of $\Vdash$ we have $x_i \Vdash p$ iff $f(x_i) \Vdash p$. Suppose $x_i R^\boxtimes y_j$. Then xRy and thus $f(x_i)Rf(y_j)$. Suppose $f(x_i)Ry$. $f(x_i) = x$, thus xRy. By definition of $R^\boxtimes$ we have $x_i R^\boxtimes y_i$. And by definition of f we have $f(y_i) = y$.

Theorem 3. *If A is $\Box$-free and $\mathsf{EI} \vdash A$ then $\mathsf{S4} \vdash A$.*

Proof. We show that any A satisfying the assumptions of the theorem is valid on all transitive and reflexive frames. The theorem then follows from the modal completeness of $\mathsf{S4}$ ([7]). So let F be some $\mathsf{S4}$ frame and let M be a model based on F. Let M_ω be the EI-model from Proposition 1. By Lemma 5 of EI we have that $M_\omega \models A$. And since M is a bounded morphic image from M_ω we also have by Lemma 6 that $M \models A$.

Theorem 4. *Any modal formula that is realizable in* GLA *is a theorem of* $\mathsf{S4}$.

Proof. Let B be a realization of A (that is B is $\Box$-free and $B^\circ \equiv A$) such that $\mathsf{GLA} \vdash B$. By Lemma 4 $\mathsf{EI} \vdash B^\circ$ and thus by Theorem 3 $\mathsf{S4} \vdash B^\circ$.

5 The Disjunction Property for GLA

In this section we will prove the disjunction property for GLA. The analog for LP was first established in [11] using a minimal model construction for LP and we will use the same technique here.

With a *constant specification* we mean a set $\mathcal{CS}$ of pairs $\langle c, A\rangle$ where $c{:}A$ is an instance of **A5**. With $\mathsf{GLA}(\mathcal{CS})$ we denote the fragment of GLA where **A5** is restricted to $c{:}A$ for $\langle c, A\rangle \in \mathcal{CS}$. For the sake of completeness we repeat some definitions from [4].

Definition 4 (GLA-model). *A* GLA-model *is a structure* $\langle W, R, \Vdash\rangle$ *where*

1. *R is a transitive conversely well-founded relation on W,*
2. *$\Vdash$ is a forcing relation satisfying for all $w, v \in W$,*
   (a) *the usual constraint on boolean connectives and $\Box$,*
   (b) *for all $t{:}F$, $w \Vdash t{:}F$ iff $v \Vdash t{:}F$,*
   (c) *$w \Vdash t{:}F$ implies $w \Vdash F$,*
   (d) *$w \Vdash s{:}(F \to G)$ and $w \Vdash t{:}F$ implies $w \Vdash (s{\cdot}t){:}G$,*
   (e) *$w \Vdash t{:}F$ implies $w \Vdash (t+s){:}F$ and $w \Vdash (s+t){:}F$,*
   (f) *$w \Vdash t{:}F$ implies $w \Vdash !t{:}(t{:}F)$.*

Let F be a formula and let $\mathrm{Sub}(F)$ be the set of sub-formulas of F. Put

$$\mathrm{S}(F) = \bigwedge\{\Box A \to A \mid \Box A \in \mathrm{Sub}(F)\}.$$

We say that a rooted GLA-model $\langle W, R, \Vdash\rangle$ with root r is *F-sound* when

$$r \Vdash \mathrm{S}(F).$$

A rooted GLA-model is a *$\mathcal{CS}$-model* when it is A-sound for all $\langle c, A\rangle \in \mathcal{CS}$ and $c{:}A$ holds in the whole model. The following theorem is shown in [4].

Theorem 5 (Modal completeness). $\mathsf{GLA}(\mathcal{CS}) \vdash A$ *iff A is valid in all A-sound $\mathcal{CS}$-models.*

For the remainder of this section we fix a finite constant specification $\mathcal{CS}$. Let $*$ be the least map

$$* :\ \mathsf{GLA}\text{-terms} \longrightarrow \mathcal{P}(\mathsf{GLA}\text{-formulas})$$

for which

- $*(c) = \{A \mid \langle c, A\rangle \in \mathcal{CS}\}$,
- $F \to G \in *(s)$ and $F \in *(t)$ implies $G \in *(s{\cdot}t)$,
- $F \in *(s)$ implies $F \in *(s+t)$ and $F \in *(t+s)$,
- $F \in *(t)$ implies $t{:}F \in *(!t)$.

The following lemma follows immediately from minimality of $*$.

Lemma 7. *1. For all variables x, $*(x) = \emptyset$,*
2. for all constants c, $*(c) = \{A \mid \langle c, A\rangle \in \mathcal{CS}\}$,
3. $F \in *(t+s)$ implies $F \in *(t)$ or $F \in *(s)$,
4. $F \in *(s{\cdot}t)$ implies that for some G, $G \in *(t)$ and $G \to F \in *(s)$,
5. $F \in *(!t)$ implies that for some $G \in *(t)$, $F \equiv t{:}G$.

Corollary 1. *If $F \in *(t)$ then $\mathsf{GLA}(\mathcal{CS}) \vdash t{:}F$.*

Proof. Induction on the complexity of t using Lemma 7.

Now given this map $*$ we define a GLA-model $M = \langle W, R \Vdash\rangle$ as follows.

- $W = \{w_0, w_1, w_2, \ldots\}$,
- $w_i R w_j$ iff $i > j$,
- $w \Vdash p$ for all $w \in W$ and all p,
- $w \Vdash t{:}A$ iff $A \in *(t)$ and for all $v \in W$, $v \Vdash A$.

Lemma 8. *M is a GLA-model. Moreover it is a GLA-model in which $c{:}A$ holds for all $\langle c, A\rangle \in \mathcal{CS}$.*

Proof. R is clearly transitive and conversely well-founded. All constraints on $\Vdash$ hold by definition and the properties of the map $*$. For $\langle c, A\rangle \in \mathcal{CS}$ we have by Theorem 5 that A holds in M. We also have that $A \in *(c)$ and thus $c{:}A$ holds in M.

The next lemma implies that for any F there exists some $w \in W$ such that w generates an F-sound $\mathcal{CS}$-model.

Lemma 9. *Let $X = \{\Box F_i \to F_i \mid 0 \le i < n\}$. There exists some $k \le n$ such that $w_k \Vdash \bigwedge X$.*

Proof. If not then by a pigeon hole argument it follows that for some $i < n$ and $r < s \le n$ we have $w_r \Vdash \Box F_i \wedge \neg F_i$ and $w_s \Vdash \Box F_i \wedge \neg F_i$. But $w_s R w_r$. A contradiction.

Theorem 6 (Disjunction property). *If $\mathsf{GLA}(\mathcal{CS}) \vdash t{:}A \vee s{:}B$ then*

$$\mathsf{GLA}(\mathcal{CS}) \vdash t{:}A \quad \textit{or} \quad \mathsf{GLA}(\mathcal{CS}) \vdash s{:}B.$$

Proof. Suppose $\mathsf{GLA}(\mathcal{CS}) \vdash t{:}A \vee s{:}B$. Let M be the model defined above. For any $i \ge 0$ let M_i be the sub-model of M generated by w_i. Since $\mathcal{CS}$ is finite by Lemma 9 there exists an $i \ge 0$ such that M_i is an $(t{:}A \vee s{:}B)$-sound $\mathcal{CS}$-model. By Theorem 5 we have $w_i \Vdash t{:}A \vee s{:}B$. But then $w_i \Vdash t{:}A$ or $w_i \Vdash s{:}B$. In the first case by Corollary 1 we get $\mathsf{GLA}(\mathcal{CS}) \vdash t{:}A$ and in the second case $\mathsf{GLA}(\mathcal{CS}) \vdash s{:}B$.

Notice that Theorem 6 generalizes to arbitrary constant specifications.

6 The Intersection of S4 and GL

We give an axiomatization of all formulas in the intersection of S4 with GL. We show that this normal modal logic has the Craig-interpolation property.

We follow the terminology from [14]. That is, a *modal logic* is a proper subset of the set of all modal formulas closed under substitution and modes ponens. A modal logic is *normal* when it is also closed under necessitation and contains all instances of $\Box(A \to B) \to (\Box A \to \Box B)$. The following lemmata are folklore.

Lemma 10. S4 *is the smallest modal logic that contains all the theorems of* K4 *and all instances of* $\Box A \to A$ *and* $\Box(\Box A \to A)$.

Lemma 11. GL *is the smallest modal logic that contains all the theorems of* K4 *and all instances of* $\Box(\Box A \to A) \to \Box A$ *and* $\Box(\Box(\Box A \to A) \to \Box A)$.

In what follows we abbreviate

$$\begin{aligned} \mathrm{L}(p) &\equiv \Box(\Box p \to p) \to \Box p, \\ \mathrm{R}(p) &\equiv \Box p \to p. \end{aligned}$$

Lemma 12. $\mathsf{S4} \vdash \neg\mathrm{L}(\bot)$ *and* $\mathsf{GL} \vdash \mathrm{L}(\bot)$.

Proof. We have $\mathsf{S4} \vdash \Box\Diamond\top$ and $\mathsf{S4} \vdash \Diamond\top$ and thus $\mathsf{S4} \vdash \neg\mathrm{L}(\bot)$. $\mathsf{GL} \vdash \mathrm{L}(\bot)$ is clear.

For F a formula we write $\boxdot F$ for $\Box F \wedge F$. Let $\mathsf{K4L_0T_0}$ be the smallest normal modal logic that contains all the instances of

4 $\Box A \to \Box\Box A$,
$\mathsf{L_0}$ $\mathrm{L}(\bot) \to \boxdot\mathrm{L}(A)$,
$\mathsf{T_0}$ $\neg\mathrm{L}(\bot) \to \boxdot\mathrm{R}(A)$.

We write $\mathsf{S4} \cap \mathsf{GL} \vdash A$ for $\mathsf{S4} \vdash A$ and $\mathsf{GL} \vdash A$.

Theorem 7. $\mathsf{S4} \cap \mathsf{GL} \vdash A$ *iff* $\mathsf{K4L_0T_0} \vdash A$

Proof. The right to left direction is immediate from Lemma 12. To show the other direction let A be a theorem of both S4 and GL. Then by Lemma 10 we get some $X_1, \ldots X_k$ such that

$$\mathsf{K4} \vdash \bigwedge_{1\le i\le k} \boxdot(\Box X_i \to X_i) \to A.$$

And by Lemma 11 we get some $Y_1, \ldots, Y_n$ such that

$$\mathsf{K4} \vdash \bigwedge_{1\le i\le n} \boxdot(\Box(\Box Y_i \to Y_i) \to \Box Y_i) \to A.$$

As $\mathsf{K4} \subseteq \mathsf{K4L_0T_0}$ and

$$\mathsf{K4L_0T_0} \vdash \neg\mathrm{L}(\bot) \vee \mathrm{L}(\bot) \to \bigwedge_{1\le i\le k} \boxdot(\Box X_i \to X_i) \vee \bigwedge_{1\le i\le n} \boxdot(\Box(\Box Y_i \to Y_i) \to \Box Y_i)$$

we have $\mathsf{K4L_0T_0} \vdash \neg\mathrm{L}(\bot) \vee \mathrm{L}(\bot) \to A$ and thus $\mathsf{K4L_0T_0} \vdash A$.

Theorem 8. $\mathsf{K4L_0T_0}$ *enjoys the Craig-interpolation property.*

Proof. Suppose $\mathsf{K4L_0T_0} \vdash A \to B$. Then both $\mathsf{GL} \vdash A \to B$ and $\mathsf{S4} \vdash A \to B$. By the interpolation theorems for GL ([8]) and $\mathsf{S4}$ ([7]) we find I_1 and I_2, in the common language of A and B such that $\mathsf{GL} \vdash A \to I_1 \to B$ and $\mathsf{S4} \vdash A \to I_2 \to B$. Now put

$$I \equiv (I_1 \wedge \mathrm{L}(\bot)) \vee (I_2 \wedge \neg \mathrm{L}(\bot)).$$

Since $\mathsf{GL} \vdash \mathrm{L}(\bot)$ we have $\mathsf{GL} \vdash I \leftrightarrow I_1$ and since $\mathsf{S4} \vdash \neg\mathrm{L}(\bot)$ we have $\mathsf{S4} \vdash I \leftrightarrow I_2$. Thus I is an interpolant for $A \to B$ in both $\mathsf{S4}$ and GL.

We have shown in the main body of this paper that the intersection of $\mathsf{S4}$ with GL is of interest when studying combined logics of explicit and formal proofs. Therefore the standard questions in the studies of modal logic are in order. However, intersections of modal logics are in general not the nicest objects in existence [13]. Apparently, by Theorem 8 with GL and $\mathsf{S4}$ we might be more lucky and therefore desirable and well-behaved answers to the questions below are plausible.

Question 1. Is there a nice cut-free formulation of $\mathsf{K4L_0T_0}$?

Question 2. What is the explicit version of $\mathsf{K4L_0T_0}$?

Question 3. What is the closed fragment of $\mathsf{K4L_0T_0}$?

Acknowledgements

The author would like to thank Professor Sergei Artemov and the anonymous referees for useful comments and suggestions.

References

1. Artemov, S.N.: Operational modal logic. Technical Report MSI 95-29, Cornell University (1995)
2. Artemov, S.N.: Explicit provability and constructive semantics. Bulletin of Symbolic Logic **7**(1) (2001) 1–36
3. Artemov, S.N., Beklemishev, L.D.: Provability logic. In Gabbay, D., Guenthner, F., eds.: Handbook of Philosophical Logic. Volume 13. 2nd edn. Kluwer (2004) 229–403
4. Artemov, S.N., Nogina, E.: Logic of knowledge with justifications from the provability perspective. Technical Report TR-2004011, CUNY Ph.D. Program in Computer Science (2004)
5. Artemov, S.N., Straßen, T.: The basic logic of proofs. In Börger, E., Jäger, G., Kleine Büning, H., Martini, S., Richter, M.M., eds.: Selected papers of the 6th Workshop on Computer Science Logic (CSL-1992), San Miniato, Italy, September 28–October 2, 1992. Volume 702 of Lecture Notes in Computer Science., Springer-Verlag (1993) 14–28

6. Beklemishev, L.D., Visser, A.: Problems in the logic of provability. Technical Report Logic Group Preprint Series 235, Department of Philosophy of Utrecht University, Utrecht (2005)
7. Blackburn, P., de Rijke, M., Venema, Y.: Modal logic. Cambridge University Press, New York, NY, USA (2001)
8. Boolos, G.: The Logic of Provability. Cambridge University Press, Cambridge (1993)
9. Goris, E.: Logic of proofs for bounded arithmetic. In Dima Grigoriev, J.H., Hirsch, E.A., eds.: Computer Science - Theory and Applications: First International Computer Science Symposium in Russia, CSR 2006, St. Petersburg, Russia, June 8-12. 2006. Volume 3967 of Lecture Notes in Computer Science., Elsevier (2006) 191–201
10. Goris, E.: Explicit proofs in formal provability logic. Technical Report TR-2006003, CUNY Ph.D. Program in Computer Science (2006)
11. Krupski, N.V.: On the complexity of the reflected logic of proofs. Technical Report TR-2003007, CUNY Ph.D. Program in Computer Science (2003)
12. Nogina, E.: On logic of proofs and provability. Bulletin of Symbolic Logic **12** (2006) 2005 Summer Meeting of the ASL.
13. Schumm, G.F.: Some failures of interpolation in modal logic. Notre Dame Journal of Formal Logic **27**(1) (1986) 108–110
14. Segerberg, K.: Post completeness in modal logic. Journal of Symbolic Logic **37** (1972) 711–715
15. Solovay, R.M.: Provability interpretations of modal logic. Israel Journal of Mathematics **25** (1976) 287–304
16. Yavorskaya-Sidon, T.L.: Logic of proofs and provability. Annals of Pure and Applied Logic **113**(1–3) (2002) 345–372

A Synthesis Algorithm for Hybrid Systems

Srikanth Gottipati[1] and Anil Nerode[2]

[1] The Graduate Center, CUNY, New York, NY 10016
[2] Cornell University, Ithaca, NY 14853

Abstract. Hybrid systems are systems of continuous plants, subject to disturbances, interacting with sequential automata in a network. By the synthesis problem for hybrid systems we mean extracting a finite state digital controller automaton from the system equations, constraints, and cost function which define the hybrid system. This automaton senses system state, and on the basis of its state, changes state and issues a chattering control to the actuators to control the system with epsilon optimal control for a fixed epsilon in real time. We address this problem by extracting a local cost function for the control system which transforms the infinite-dimensional optimization problem into a finite-dimensional problem.

1 Introduction

Hybrid systems are complex dynamic systems composed of an interacting network of subsystems with continuous dynamics (we refer to them as plants) and subsystems with discrete dynamics (we refer to them as digital controllers). The continuous and discrete dynamics interact. Changes occur in the plant states in response to changes in the environment and changes in states of the digital controllers. Changes in the states of the digital controllers occur due to changes in the plant state and the environment. The digital controllers are finite state real time input-output devices, the plant evolutions are governed by differential or difference equations. The hybrid system is the coupled system of discrete and continuous elements (see Figure 1).

Hybrid systems arise in many contexts, both in man-made systems and in nature. Continuous systems which have a phased operation, such as walking robots, insect motion, or biological cell growth and division, are well-suited to be modeled as hybrid systems, as are continuous systems which are controlled by a discrete logic, such as a chemical plant controlled with valves and pumps, or the autopilot modes for controlling an aircraft. Hybrid systems are also natural models for systems comprised of many interacting subsystems or processes, such as air or ground transportation systems. In these examples, the continuous dynamics model system state evolution, biochemical or chemical reactions, while the discrete dynamics control the sequence of contacts or collisions in the gait cycle, cell divisions, valves and pumps switching, and coordination protocols. In all of these examples, the system dynamics are complex enough that traditional analysis and control methods based solely on differential equations have not been computationally feasible for complex systems. Another important way in which hybrid systems arise is from the hierarchial organization of complex systems. In these systems, a hierarchial organization helps manage complexity and higher levels in the hierarchy require less detailed models of the functioning of the lower levels, necessitating the interaction of discrete and continuous components. Example of such systems include flexible manufacturing and chemical process control systems, intelligent highway systems, air traffic management systems, etc.

To understand the behavior of hybrid systems, to simulate, and to control these systems, theoretical advances and analytical tools are needed. Recent developments include building theory and analytical tools, and applying them to interesting and large scale systems which are currently not known how to analyze, control, or simulate like an automated air traffic system, biological control circuit, etc. The predominant model being used in the literature for hybrid systems is to partition the entire state space into finite regions, possibly overlapping, where in each region has its own invariant. Once the evolved state of the system in a particular region violates this invariant condition, the state of the system is thrown by the digital control program into a different region with possible reset of the state and from then on evolves according to the state dynamics specified for that region, as long as the invariant for that region is maintained.

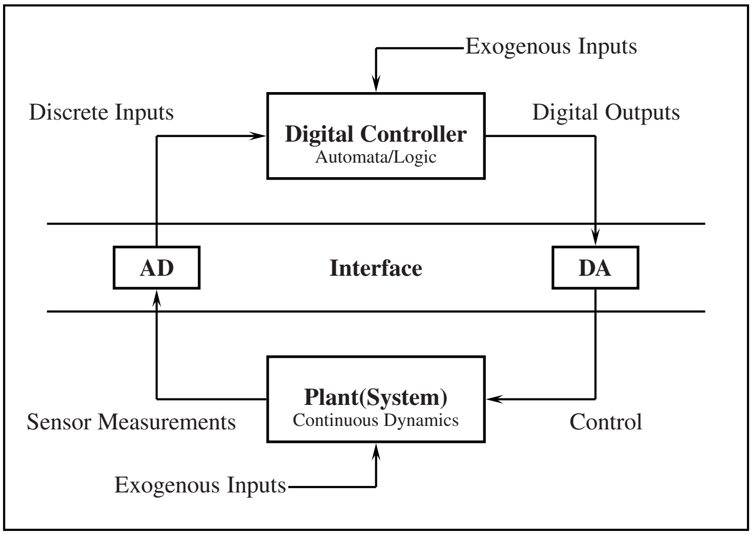


Fig. 1. Hybrid system

The other modeling perspective is that the state space has a finite number of open regions which are represented by logical propositions which involve systems of differential equalities and inequalities. When the state enters one of these regions one of the logical rules might fire in which case the system dynamics and constraints might change and this in effect is reflected in the change in control policy. When several of the rules fire, there is supposed to be consistent behavior.

The analysis of behavior of classes hybrid systems has become quite popular, there are many meetings now. We are primarily interested in extraction of controls which guarantee epsilon optimal control relative to a cost function and constraints. We are less concerned with taking a given hybrid system (with controls given too) and trying to figure out its behavior as a dynamical system, which is a principal focus of much current research.

1.1 History of Control

Here is some background. We first recapitulate some of the evolution of control theory for continuous plants. The modern control of industrial devices by feedback goes back at least to the steam engine governor of James Watt (1787), the feedback linear amplifiers of E. H.Armstrong's patent (1914), the theory of stability of linear feedback amplifier developed at Bell Labs by H. S. Black (1927), H. Nyquist (1932), H. W. Bode (1940) (Laplace transform frequency domain analysis), the theory of linear servomechanisms, developed at MIT Radiation labs, during World War II, emerging in the book "Theory of Servomechnisms" by James, Nichols, and Phillips, 1947. Complex interacting linear systems can be represented in matrix form, giving rise to linear control theory. In 1960 Kalman gave a straightforward matrix calculus algorithm for finding optimal linear controllers for linear models with a linear quadratic cost function when they exist, based on simple facts about matrices and quadratic forms. This remains the principal optimal control tool for the control engineer to this day, especially for process control.

For non-linear optimization in the 1950's Richard Bellman [1] invented dynamic programming for finding paths satisfying constraints and minimizing a cost function assuming his "principle of optimality", based on backwards search for an optimal trajectory starting at the goal. In the 1950's L.S. Pontryagin [12] developed a differential equation necessary condition for optimal trajectories subject to differential equality and inequality constraints. His maximum principle can be expressed in terms of an associated symplectic geometry. His principle, earlier discovered by Hestenes in 1950, in the context of the calculus of variations, is a local necessary condition for a minimum, just as in calculus setting a derivative equal to zero is a local necessary condition for a minimum. The purpose of local necessary conditions is to cut down the set of controls that have to be considered to find an optimal control. Pontryagin's maximum principle plays for optimality the same role that the first and second derivative tests for minima do in calculus. It is a variant and generalization of the first and second variation necessary tests of the calculus of variations. Most calculus of variation books explore local necessary conditions and local sufficient conditions for minima, but go no further.

What are the mathematical theorems that prove the existence of optimal curves from start to goal? In calculus we use Weierstrass's sufficiency theorem where the variational operator operates on f and converge to a point where the minimum is achieved (Specker, 1959). But if we say that in practice, we only need to be within epsilon of the minimum, there are algorithms, even efficient algorithms when some smoothness is assumed. The situation is similar in the calculus of variations and optimal control. Hilbert's "direct method" (1901) proves the existence of curves minimizing from start to goal by application of Arzela-Ascoli lemma on uniformly bounded equicontinuous families. Application of this to suitable function spaces produces a compact function space such that, on applying the theorem that an upper semicontinuous function on a compact set attains its minimum, one gets the minimizing curve. Compactness is achieved by extending the space of piecewise smooth curves to a completion using the method first introduced in the calculus of variations by E. J. MacShane (1939-41). These are the "weak solutions", also called relaxed solutions, or measure valued solutions according to the Riesz representation theorem. This was developed further by L.C. Young, Pontryagin, and their followers. Here too algorithms do not exist which will in general compute optimal measure-valued controls. But, just as in calculus, if one asks only for controls which yield trajectories with cost within a prescribed epsilon of optimal cost, there are such algorithms yielding piecewise smooth epsilon optimal control functions, and becomes a question of developing efficient algorithms. Wolf Kohn and Anil Nerode in 1992-4 began the development and implementation of efficient real time algorithms for epsilon optimal control of complex systems. They founded a company, Clearsight Systems, which has demonstrated the real time feasibility of using such chattering approximations to optimal control for many applications such as logistics enterprize systems, financial analysis, and many other areas.

What does this have to do with hybrid systems? The algorithms for real time epsilon optimal control for a fixed epsilon extract a finite state digital controller automaton from the system equations, constraints, and cost function. This automaton senses system state, and on the basis of its state, changes state and issues a chattering control to the actuators to control the system. This was the motivation for Kohn-Nerode defining hybrid automata in the 1992 volume "Hybrid Systems", LNCS, [11]. So hybrid automata and hybrid systems emerge when one extracts epsilon optimal control for continuous non-linear systems and implements it.

There is a second place where the hybrid system concept is important. In many complex systems, behavior is constrained by logical rules. If an order is given by a commander to troops or by a high business administrator to his organization, the course of action must satisfy doctrine, regulations, and other constraints which are expressed in logic. Traditional control theory simply does not deal with this. Those who study planning do deal with this, but primarily by logical deduction of a plan meeting logical constraints and achieving a goal. Prolog is suited to this. But when the system being controlled is a physical system like an airplane or a chemical process plant, we now have purely digital rules in logical form interacting with an evolving physical dynamical system, a hybrid system. The Kohn-Nerode approach converts the logical constraints and goals into (generally non-convex) continuous constraints and goals, and couples the continuous constraints on the controlled physical system with the continualization of the discrete constraints.

Our intention, following Kohn-Nerode [7], [11], is to cut down the search space and to improve the algorithms for computing optimal trajectories using Finsler geometries associated with optimal control problems, specifically to introduce use of Finsler curvature formulas to guide computations. This entails an extensive analysis of Finsler geometry in control theory.

1.2 Optimal Control: Synthesis

Optimal control theory could be viewed as a generalization of Calculus of Variations problems. The problem of optimal plant control is that of constructing a control law or policy $u(t) \in U$, where is the set of admissible controls, which will drive the plant from its initial state, $x_0 \in M$, into some set of goal states, $G \subset M$, along an optimal path $x(t)$. Optimality is generally formulated as minimizing some cost functional on paths, which we label $J(x)$. In this proposal, the cost functionals will always be assumed to be of the form

$$J(x) := \int_{t_0}^{t_1} L(x(t), u(t))dt, \tag{1}$$

where L is the cost function. In practice, moreover, the plant dynamics are constrained to take place on a constraint sub-manifold $N \subset M$. In our work, N is determined by differential-algebraic constraints of the form $F(x, \dot{x}, u) = 0$. These constraints arise from dynamic concerns and from performance specifications. For a particular problem we will have to specify the set of admissible paths from which we will draw candidates for optimality. We shall work with piecewise smooth paths. We shall also assume controllability, i.e., we will assume the existence of a path on the constraint sub-manifold which connects x_0 to G.

In conventional optimal control theory two methods for computing optimal controls stand out. One is the Bellman's *Hamilton-Jacobi-Bellman* (HJB) method and the other is the Pontryagin's *Maximum Principle* method. From an engineer's perspective, the HJB framework is preferred for solving optimal control problems since it naturally leads to a design of optimal *feedback* (or *closed-loop*) controllers, i.e. control laws as a function of state. However, except for some very simple problems, this approach is bereft with major theoretical and numerical difficulties. The alternative framework avoids the pitfalls associated with the HJB theory but generates *open-loop* controls, i.e. control laws as a function of time. However, it has been known since the birth of optimal control that if open-loop controls can be generated in real-time, they are basically equivalent to feedback controls. In modern terminology this is *model-predictive control*(MPC) with the horizon being time-to-go. In a later chapter we shall give a full account of the state-of-art methods for computing optimal controls for practical nonlinear systems.

With respect to types of Hybrid models that we described above, they require only open-loop controls if we have full knowledge of the system at the time of computation of optimal controls. With any unforeseen contingencies, for example disturbances or changes in the logic database, one needs to develop feedback controls. Control laws are implemented via a finite state machine. The following simple example shows how the feedback law could be represented by a hybrid automaton.

1.3 An Intuitive Example: Double Integrator Problem

Consider the control system

$$\frac{d^2x}{dt^2} = u, \tag{2}$$

where u is a real control parameter constrained by the condition that $|u| \leq 1$. In the phase coordinates $x_1 = x$ and $x_2 = dx/dt$, this equation may be rewritten in the form of the following system:

$$\frac{dx_1}{dt} = x_2, \qquad \frac{dx_2}{dt} = u. \tag{3}$$

Let us consider the problem of getting to the origin $(0, 0)$ from a given initial state x_{init} in the shortest time. In other words, we shall consider the time-optimal problem for the

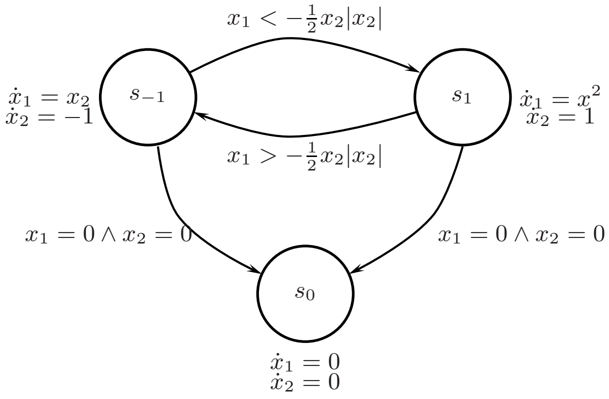


Fig. 2. Hybrid Automaton: Double Integrator

case where the origin $(0, 0)$ is the terminal position. The feedback control for the double integrator problem is given by

$$u(t, x_1, x_2) = \begin{cases} -1, & \text{if } x_1 > -\frac{1}{2}x_2|x_2|, \\ 1, & \text{if } x_1 > 0 \text{ and } x_1 = -\frac{1}{2}x_2|x_2|, \\ 1, & \text{if } x_1 < -\frac{1}{2}x_2|x_2|, \\ -1, & \text{if } x_1 < 0 \text{ and } x_1 = -\frac{1}{2}x_2|x_2|. \end{cases} \tag{4}$$

Current numerical techniques for the Dynamic Programming method cannot be implemented to problems of high dimensions due to the "curse of dimensionality". Hence numerical computation of feedback control laws seems to be a far-fetched proposition at the moment. We need more information on the geometric structure of our optimal control problem. The hope is that it would be possible to derive feedback controls from geometrical objects like curvature, torsion, etc which are measures of deviation from flat Euclidean spaces. Physics has especially benefited from the study of curved manifolds through Riemannian geometry. However, optimal control problems do not lie on Riemannian manifolds but on more general spaces called Finsler spaces.

In this paper we develop a synthesis algorithm by transforming the optimal control problem into an finite dimensional optimization problem. The paper is organized as follows: section 2 introduces sprays, section 3 gives a detailed study of the inverse problem of the calculus of variations, and section 4 outlines the synthesis algorithm.

2 Sprays

Second order ordinary differential equations (SODEs) arise in many areas of natural science. A special class of SODEs come from the variational problems of Lagrange metrics including Finsler metrics. Many geometers like L. Berwald, T. Y. Thomas, O. Veblen, J. Douglas, D. Kosambi and E. Cartan, have studied SODEs using geometric methods. SODEs in general are given in the following form

$$\frac{d^2x^i}{ds^2} = \Phi^i\left(s, x, \frac{dx}{ds}\right), \qquad i = 2, \ldots, n \ . \tag{5}$$

Riemannian and Finsler geodesics are given as solutions of a homogeneous system of SODEs which are known in the differential geometry literature as sprays. One can study such SODEs which are more general than the geodesic equations obtained from Riemannian or Finsler metrics, see [15]. We first define a spray as follows:

Definition 1. *Let M be a manifold. A spray on M is a smooth vector field G on $TM \setminus \{0\}$ expressed in a standard local coordinate system (x^i, v^i) in TM as follows*

$$\mathbf{G} = v^i \frac{\partial}{\partial x^i} - 2G^i(x,v) \frac{\partial}{\partial v^i}, \tag{6}$$

where $G^i(x,v)$ are local functions on TM satisfying

$$G^i(x, \lambda v) = \lambda^2 G^i(x,v), \qquad \lambda > 0 \ . \tag{7}$$

A manifold with a spray is called a spray space.

Definition 2. *A regular curve c in M is called a geodesic of* $\mathbf{G}$ *if it is the projection of an integral curve of* $\mathbf{G}$*. We shall call G^i the spray coefficients of* $\mathbf{G}$*.*

For a vector $v \in T_xM$, let $\gamma : (-\varepsilon, \varepsilon) \to M$ be the curve in $\mathcal{G}$ with $\dot{\gamma}(0) = v$. Define

$$G^i(v) := -\frac{1}{2} \frac{d^2\gamma^i}{dt^2}(0), \tag{8}$$

where $(\gamma^i(t))$ are the local coordinates of the curve $\gamma(t)$ and $G^i(v)$ are positive homogeneous of degree 2 in v. It is then easy to prove that any curve $c : (a,b) \to M$ in $\mathcal{G}$ satisfies the following system,

$$\frac{d^2\gamma^i}{dt^2} + 2G^i\left(\frac{dc}{dt}\right) = 0 \ . \tag{9}$$

Every system of SODEs (also called a semispray) can be studied via the associated system of sprays, and every Lagrange metric can be studied via the associated Finsler metric. Let us assume that a system of semisprays and sprays are related by

$$\Phi^i(s, x, \dot{x}) = 2\dot{x}^i G^1(s, x, 1, \dot{x}) - 2G^i(s, x, 1, \dot{x}). \tag{10}$$

Clearly, for a given set of functions Φ^i, there infinitely many sets of homogeneous functions G^i satisfying (). A special case is when G^i are given by

$$\begin{cases} G^1(x,y) = 0 \\ G^i(x,y) = -\frac{1}{2} y^1 y^1 \Phi^i(s, \eta, \xi) \end{cases} \tag{11}$$

where $s = x^1$, $\eta^i = x^i$, and $\xi^i = y^i / y^1$.

3 Inverse Problem of the Calculus of Variations

The inverse problem in the calculus of variations involves deciding whether the solutions of a given system of second-order ordinary differential equations

$$\ddot{x}^i = F^i(t, x^b, \dot{x}^b), \quad a, b = 1, \ldots, n, \tag{12}$$

are the solutions of a set of Euler-Lagrange equations

$$\frac{\partial L}{\partial x^j} - \frac{d}{dt}\frac{\partial L}{\partial \dot{x}^j} = \frac{\partial L}{\partial x^j} - \frac{\partial^2 L}{\partial \dot{x}^k \partial \dot{x}^j}\ddot{x}^j - \frac{\partial^2 L}{\partial x^k \partial \dot{x}^j}\dot{x}^j - \frac{\partial^2 L}{\partial t \partial \dot{x}^j} = 0 \tag{13}$$

for some Lagrangian $L(t, x, \dot{x})$.

This problem, referred to as the inverse problem of the calculus of variations was first posed by H. von Helmholtz in 1887. In his paper [3], Helmholtz studied a system of n second order differential equations of the form

$$B_{jk}(t, x^i, \dot{x}^i)\ddot{x}^k + A_j(t, x^i, \dot{x}^i) = 0, \tag{14}$$

where $1 \leq i, j, k \leq n$. He found necessary conditions for the existence of a Lagrange function L such that

$$B_{jk}\ddot{x}^k + A_j = \frac{\partial L}{\partial x^j} - \frac{d}{dt}\frac{\partial L}{\partial \dot{x}^j} \quad . \tag{15}$$

These conditions, referred to as *Helmholtz* conditions, are of the form

$$B_{jk} = B_{kj} \tag{16}$$

$$\frac{\partial B_{jk}}{\partial \dot{x}^i} = \frac{\partial B_{ji}}{\partial \dot{x}^k} \tag{17}$$

$$\frac{\partial A_j}{\partial \dot{x}^k} + \frac{\partial A_k}{\partial \dot{x}^j} = 2\frac{\widehat{d}}{dt}B_{jk} \tag{18}$$

$$\frac{\partial A_j}{\partial x^k} - \frac{\partial A_k}{\partial x^j} = \frac{1}{2}\frac{\widehat{d}}{dt}\left(\frac{\partial A_j}{\partial \dot{x}^k} - \frac{\partial A_k}{\partial \dot{x}^j}\right) \tag{19}$$

where $1 \leq j, k \leq n$, and the operator $\widehat{d}/dt$ is defined by

$$\frac{\widehat{d}}{dt} = \frac{\partial}{\partial t} + \dot{x}^i \frac{\partial}{\partial x^i} \quad . \tag{20}$$

In 1896 Mayer [10] has proved that the *Helmholtz* conditions are also sufficient for (14) to be variational.

The inverse problem originally posed by Helmholtz is naturally generalized to the following question: which also is referred to as inverse problem of the calculus of variations: Under what conditions are the SODE (12) equivalent to some Euler-Lagrange equations? In other words, one asks whether there exist Euler-lagrange equations the solutions of which coincide with the solutions of equations (12). In this formulation the problem is very general and its solution is not yet known. However, one can simplify this problem restricting the class of equivalent equations, namely, to equations of the form

$$f_{ij}(\ddot{x}^j - F^j) = \frac{d}{dt}\frac{\partial L}{\partial \dot{x}^i} - \frac{\partial L}{\partial x^i}, \tag{21}$$

where f_{ij} are functions of $(t, x, \dot{x})$. Such a matrix f_{ij} is called a variational multiplier and the problem is often called the multiplier problem. Hirsch [5] was the first person to pose the multiplier problem. Hirsch produced certain self-adjointness conditions which, of themselves, are not particularly useful in classifying SODEs according to the existence and uniqueness of the corresponding multipliers. The most commonly used set of necessary and sufficient conditions for the existence of the multiplier f_{ij} are also called Helmholtz conditions due to Fields medalist J. Douglas, who in his landmark paper [2] goes on to completely solve the problem for $n = 2$ case, and put in the following form by Sarlet [14]:

$$f_{ij} = f_{ji} \tag{22}$$

$$\Gamma(f_{ij}) = f_{ik}\Gamma^k_j + f_{jk}\Gamma^k_i \tag{23}$$

$$f_{ik}\Phi^k_j = f_{jk}\Phi^k_i \tag{24}$$

$$\frac{\partial f_{ij}}{\partial \dot{x}^k} = \frac{\partial f_{ik}}{\partial \dot{x}^j} \tag{25}$$

where

$$\Gamma^i_j := -\frac{1}{2}\frac{\partial F^i}{\partial \dot{x}^j}, \qquad \Phi^i_j := -\frac{\partial F^i}{\partial x^j} - \Gamma^k_j\Gamma^i_k - \Gamma\left(\Gamma^i_j\right), \tag{26}$$

and the operator Γ is defined by

$$\Gamma := \frac{\partial}{\partial t} + \dot{x}^i\frac{\partial}{\partial x^i} + F^i\frac{\partial}{\partial \dot{x}^i} \quad . \tag{27}$$

In [2], Douglas does a painstaking study of four main cases and many subcases. These cases are distinguished by the vanishing or otherwise of quantities constructed from F^i. Douglas expresses the necessary conditions to be satisfied by the multiplier matrix as partial differential equations and uses Riquier-Janet theory to solve them. In most cases, Douglas decides the existence question and gives all the possible Lagrangians; in the remaining cases the problem becomes a question of the closure of a certain 1-form. In his paper Douglas also avoids the Hirsch's self-adjointness conditions. Douglas's solution is so comprehensive that his conditions formed the basis for subsequent attempts on the 3 and higher-degrees of freedom cases. Following Douglas the next significant contribution to solving the Helmholtz conditions was that of the Henneaux. Henneaux [4] developed an algorithm for solving the Helmholtz conditions for any given system of SODEs, and in particular, he solved the problem for spherically symmetric problems in dimension 3.

Example 3. For an arbitrary linear 1-dimensional SODE of the form

$$\ddot{y} + a(x)\dot{y} + b(x)y = f(x) \tag{28}$$

the Lagrangian is given by

$$L(x, y, \dot{y}) = \left[\frac{1}{2}\dot{y}^2 + \frac{b(x)}{2}y^2 - f(x)y\right] e^{\int a(x)dx} \tag{29}$$

Example 4. For a linear control spray of the following form;

$$\ddot{\mathbf{x}} = A \cdot \mathbf{x} + B \cdot \mathbf{u}, \quad \mathbf{x} \in R^n, \ \mathbf{u} \in R^m \tag{30}$$

where A is symmetric and $\mathbf{u}$ is assumed to be a parameter, we get the following Lagrangian

$$L = \frac{1}{2}\dot{\mathbf{x}}^\top \dot{\mathbf{x}} + \frac{1}{2}\mathbf{x}^\top A\mathbf{x} + \mathbf{x}^\top B\mathbf{u} \tag{31}$$

3.1 Finsler Metrizability

In Finsler geometry, the geodesics of a Finsler function, parametrized so that the tangent vector has constant Finsler length, define a spray, see [9], [13], [15]. However, not every spray is obtainable in this way. The inverse problem for sprays is the problem of determining, for a given spray, whether or not there exists a Finsler function of which it is the geodesic spray; or more broadly, of giving criteria for distinguishing those sprays which are geodesic. In fact the inverse problem in the broad sense refers to a whole projective class of sprays rather an individual one. The inverse problem for sprays, or the Finsler-metrizability problem as it is sometimes called, is thus a special case of the inverse problem of calculus of variations for arbitrary systems of SODEs, or in other words for semi-sprays.

Curvature terms play an important role in the Finsler-metrizability problem. Those sprays with $n > 1$ degrees of freedom which have vanishing Riemann curvature (also called R-flat) are Finsler-metrizable. Moreover, if any particular spray is Finsler-metrizable one can conclude that all sprays projectively equivalent to it are also metrizable. It turns out that the isotropic sprays are projectively equivalent to R-flat sprays.

The Riemann curvature R^i_{jkl} of a spray determines and is determined by the type $(1, 1)$ tensor $R^i_j = R^i_{kjl}u^k u^l$; the spray is isotropic if there is a function λ and a vector field μ_i such that

$$R^i_j = \lambda\delta^i_j + \mu_j u^i \ . \tag{32}$$

The property of being isotropic is easily seen to be projectively invariant. One can construct from the Riemann curvature a projectively invariant tensor P^i_{jkl}, related to it in much the same way as the projective curvature is to the Riemann curvature of an affine connection. Douglas has also studied such tensors with respect to the inverse problem and has called it the Weyl tensor. It turns out that for $n > 2$ a spray is isotropic iff $P^i_{jkl} = 0$.

Another important class of sprays are the projectively affine sprays which are defined to be those sprays whose Douglas curvature vanishes. The Douglas curvature tensor defined on sprays is projective invariant constructed from the Berwald curvature of sprays. These curvatures are non-Riemannian. A spray is called projectively R-flat iff both the Douglas and Weyl curvatures vanish. A nice theorem by Berwald shows that the structure of projectively affine sprays should be a polynomial of $\dot{x}$ of degree ≤ 3, i.e.,

$$F^i(t,x,\dot{x}) = A^i(t,x) + B^i_j(t,x)\dot{x}^j + C^i_{jk}(t,x)\dot{x}^j\dot{x}^k + D^i_{jkl}(t,x)\dot{x}^j\dot{x}^k\dot{x}^l, \tag{33}$$

where $1 \leq i,j,k,l \leq n$.

One possible method for solving the inverse problem for a system of SODEs is to construct the spray by passing to the homogeneous formalism; if this spray is projectively Finslerian then the original equations are variational. Of course this will in general be only a local construction: that is to say, the spray is only locally defined, and one cannot expect to do more than find a local Finsler function.

Since any isotropic spray is projectively Finslerian, it is natural to ask for the conditions under which a system of SODEs give rise to an isotropic spray. For this purpose it is necessary to express the Riemann curvature of the spray in term of quantities derived from the differential equations, that is from the coefficients F^i. It turns out that

$$R^0_0 = R^0_i = 0 \tag{34}$$

$$R^i_0 = \dot{x}^0\dot{x}^j\Phi^i_j \tag{35}$$

$$R^i_j = -(\dot{x}^0)^2\Phi^i_j, \tag{36}$$

where Φ^i_j is the so-called Jacobi endomorphism of the system of differential equations,

$$\Phi^i_j = \frac{\partial F^i}{\partial x^j} + \frac{d\gamma^i_j}{dt} + \gamma^i_k\gamma^k_j, \qquad \gamma^i_j = -\frac{1}{2}\frac{\partial F^i}{\partial \dot{x}^j}. \tag{37}$$

It turns out that when SODEs are expressed in terms of spray geometry, Douglas's case I criterion corresponds exactly to the condition for a spray to be isotropic. It follows that systems of SODEs in case I are variational.

4 Synthesis

Given a control SODE system

$$\ddot{x} = f(t,x,\dot{x},u), \quad u \in U \subseteq R^m, \quad x \in M \subseteq R^n \tag{38}$$

the objective is to synthesize feedback control policies $u(t,x,\dot{x})$ to steer the system from any given initial point $x \in M$ to desired goal $x_g \in M$. The goal is to synthesize a feedback controller $u(t,x,\dot{x})$ for the above optimal control problem. For the sake of clarity we shall initially consider time-optimal control problems. However it should be noted that all optimal control problems could be transformed to time-optimal control problems. The algorithm involves the construction of an inverse Lagragian for the control system which is used to construct local goals. Before we outline our synthesis algorithm we shall give two motivating examples.

Example 5. The control SODE for the double integrator problem as shown before is given by

$$\ddot{x} = u, \quad |u| \leq 1, \quad x \in R. \tag{39}$$

The objective is to steer the system to $x = 0$ with zero terminal velocity $\dot{x} = 0$ in minimum possible time. Let us assume the control parameter u to be a constant for the synthesis purposes. We know that the first order Lagrangian for the SODE is given by

$$L(t, x, \dot{x}, u) = \frac{1}{2}\dot{x}^2 + ux \tag{40}$$

The first step for any feedback control $u(t, x, \dot{x})$ synthesis is to find the optimal controls

$$u(t, x, \dot{x}) = \arg\min_{u \in U} L(t, x, \dot{x}, u) \tag{41}$$

The extremal controls u^* are given by

$$u^*(x, \dot{x}) = \begin{cases} 1, & x > 0, \\ -1, & x < 0. \end{cases} \tag{42}$$

The trajectories that lead us to the goal should be the minimal curves which are the curves in the $(x, \dot{x})$ plane given by the following equations

$$\frac{1}{2}\dot{x}^2 + x = 0, \qquad \frac{1}{2}\dot{x}^2 - x = 0, \tag{43}$$

the solutions to which are parabolas. The complete synthesis is shown in Figure 3.

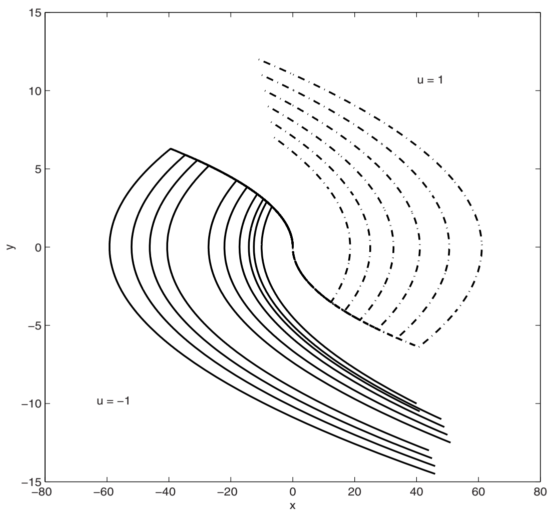


Fig. 3. Double Integrator optimal paths

Example 6. For the SODE

$$\ddot{x} = x + u, \quad |u| \leq 1, \quad x \in R, \tag{44}$$

the inverse Lagrangian is given by

$$L(t, x, \dot{x}, u) = \frac{1}{2}\dot{x}^2 + \frac{1}{2}x^2 + ux \tag{45}$$

The extremal controls u^* in order to steer the system to the point $(x, \dot{x}) = (0, 0)$, are given by

$$u^*(x, \dot{x}) = \begin{cases} 1, & x > 0, \\ -1, & x < 0. \end{cases} \tag{46}$$

The trajectories that lead us to the goal should be the minimal curves which are given by the following equations

$$\frac{1}{2}\dot{x}^2 + \frac{1}{2}x^2 + x = 0, \qquad \frac{1}{2}\dot{x}^2 + \frac{1}{2}x^2 - x = 0 \tag{47}$$

the solutions to which are circles with centers $(1, 0)$ and $(-1, 0)$ and radius 1.

It is quite evident from the above examples that the chief ingredient for the synthesis of feedback optimal controls is the inverse Lagrangian for the control SODEs. Given an optimal control problem as following:

$$\min_{u \in U} \int_0^T L(t, x, u)dt \tag{48}$$

subject to the constraints

$$\dot{x} = f(t, x, u), \qquad g_1(x, \dot{x}, u) \leq 0, \qquad x(T) = 0. \tag{49}$$

1. Reduction to a time-optimal control problem: The following idea of transforming the variational problem into a time-optimal control problem was used earlier by Gamkralidze to prove some existence results. We first reparametrize the time variable t with a new variable s

$$s(t) = \int_0^t L(\tau, x(\tau), u(\tau))d\tau, \qquad t \in [a, b], \tag{50}$$

which is a strictly monotonous continuous function of t, for any pair $(x(t), u(t))$ satisfying $\dot{x}(t) = F(t, x(t), u(t))$. As far as

$$\frac{ds(t)}{dt} = L(t, x(t), u(t)) > 0, \tag{51}$$

then $s(\cdot)$ admits a monotonous inverse function $t(\cdot)$ defined on $[0, T]$, such that

$$\frac{dt}{ds}(s) = \frac{1}{L(t(s), x(s), u(s))} \ . \tag{52}$$

Obviously

$$\frac{dx(t(s))}{ds} = \frac{dx(t(s))}{dt}\frac{dt(s)}{ds} = \frac{F(t(s),x(s),u(s))}{L(t(s),x(s),u(s))} \tag{53}$$

We can transform the original problem into the following time-optimal problem

$$\min_{u\in U} T \tag{54}$$

subject to

$$\dot{t} = \frac{1}{L(t,x,u)} \tag{55}$$

$$\dot{x} = \frac{f(t,x,u)}{L(t,x,u)} \tag{56}$$

$$g(x,\dot{x},u) \leq 0 \tag{57}$$

2. Construct a first-order inverse Lagrangian $L^{inv}(t,x,\dot{x},u)$ for the control system (assuming u to be a constant parameter), such that the control system is indeed its Euler-Lagrange equations.
3. Having thus derived a inverse Lagrangian for the dynamic optimal control problem, we have essentially transformed the problem into the following finite dimensional optimization problem

$$u^*(t,x,\dot{x}) = \arg\min_{u\in U} \int_{V\subset R^{2n}} L^{inv}(t,z,u)dz, \tag{58}$$

where $z = (x,\dot{x})$, and $U = \{u|g(z,u) \leq 0\}$. If the inverse Lagrangian L^{inv} is convex in u, then we have an unique extremal control u^*, otherwise, one will have to optimize a relaxed inverse Lagrangian. The relaxation could be done by twice applying the Young-Fenchel transform on the inverse Lagrangian.

$$C_u(L(z,u)) = L^{**}(z,u) = \max_{u^*}\{u^* \cdot u^* - L^*(z,u^*)\} \tag{59}$$

where L^* is defined as the conjugate of L by

$$L^*(z,u^*) = \max_{u}\{u^* \cdot u - L(z,u)\} \ . \tag{60}$$

When $L(z,u)$ is not convex in u then the transform C_u when applied on L gives the convex envelope of L.
4. Finally we compute the optimal trajectories which steer the system towards the goal by solving for z in the following equation

$$L(z,u^*) = 0. \tag{61}$$

The hypersurfaces which satisfy the above condition bifurcate the extremal trajectories computed in step 3 and steer the system towards its goal in an optimal sense. This concludes the synthesis algorithm.

We have sketched an algorithm which synthesizes feedback controls along the lines of Hilbert's direct method for the multidimensional calculus of variations problems. The algorithm needs to be developed to include state constraints (physical) and logical constraints. This would probably require introducing potential fields into system. The challenging part though of this algorithm is the computation of the inverse Lagrangian. However, we believe that the computation of a symbolic inverse Lagrangian would greatly help in the modeling and design of hybrid systems.

References

1. R. Bellman, *Dynamic Programming*, Princeton Un iversity Press, Princeton, N.J., 1957.
2. J. Douglass, *Solution of the inverse problem of the calculus of variations*, Trans. Am. Math. Soc.,50, 71-128. 1941.
3. H. Helmholtz, *Uber der physikalische Bedeutung des Princips der kleinsten Wirkung*, J. Reine Angew. Math., 100, 137-166. 1957.
4. M. Henneaux, *On the inverse problem of the calculus of variations*, J. Phys. A: Math. Gen. 15, L93-L96, 1957.
5. A. Hirsch, *Die Existenzbedingungen des verallgemeinerten kinetischen Potentials*, Math. Ann., 50, 429-441. 1957.
6. W. Kohn, A. Nerode, J. Remmel, *Hybrid systems as Finsler manifolds: finite state control as approximation to connections*, Hybrid systems, II (Ithaca, NY, 1994), 294-321, Lecture Notes in Comput. Sci., 999, Springer, Berlin, 1995.
7. W. Kohn, V. Brayman, and A. Nerode, *Control synthesis in Hybrid systems with Finsler dynamics*, Houston Journal of Mathematics, Vol.28, No. 2, 2002.
8. W. Kohn, V. Brayman, P. Cholewinski and A. Nerode, *Control in Hybrid Systems*, International Journal of Hybrid Systems, Volume 3, No 2 & 3, 2003.
9. M. Matsumoto, *Foundations of Finsler geometry and special Finsler spaces*, Kaiseicha Press, Otsu, 1986.
10. A. Mayer, *Die Existenzbedingungen ines kinetischen potentiales*, Berich. Verh. Konig. Sach. Gesell. wissen, Leipzig, Math. Phys, Kl. 84, 519-529 1957.
11. A. Nerode and W. Kohn, *Models for Hybrid Systems: Automata, Topologies, Controllability, Observability*, LNCS Hybrid Systems, 1993.
12. L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, and E.F. Michenko, *The Mathematical Theory of Optimal Processes*, John-Wiley Interscience publishers 1962.
13. H. Rund, *The differential geometry of Finsler spaces*, Springer-Verlag, Berlin-Gottingen-Heidelberg, 1959.
14. W. Sarlet, *The Helmholtz conditions revisited. A new approach to the inverse problem of Lagrangian dynamics*, J. Phys. A: Math. Gen. 15, 1503-1517, 1993.
15. Z. Shen, *Differential Geometry of Spray and Finsler Spaces*, Kluwer Academic Publishers, 2001.

Including the Past in ‘Topologic’

Bernhard Heinemann

Fakultät für Mathematik und Informatik,
FernUniversität in Hagen,
58084 Hagen, Germany
bernhard.heinemann@fernuni-hagen.de

Abstract. In this paper, we extend Moss and Parikh’s topo-logical view of knowledge. We incorporate a further modality, denoted P, into the original system. This operator describes the increase of sets. Regarding the usual logic of knowledge, P corresponds to no learning of agents. In the context of ‘topologic’, however, P represents the reverse effort operator and is related to the past therefore. It is our objective to prove nice properties of the accompanying logic like soundness and completeness with respect to the intended class of structures, or decidability. To this end, we take up a hybrid logic point of view, among other things. This not only yields the desired results, but also has some interesting consequences with regard to applications.

Keywords: modal logic, the logic of knowledge, topological reasoning, hybrid logic, temporal operators

1 Introduction

In the paper [1], Moss and Parikh presented a certain bi-modal logic of *knowledge and effort.* On the one hand, the language underlying that logic makes possible a qualitative description of procedures gaining knowledge, and on the other hand some expressive power concerning spatial concepts is provided. In fact, since knowledge is represented through *knowledge states,* which are the sets of states an agent in question considers possible at a time, knowledge acquisition appears as a *shrinking procedure* regarding the space of all such sets. Thus ideas from topology like *closeness* or *neighbourhood* turn up together with knowledge in a natural way.

Moss and Parikh suggestively called their system topologic, and we adopt this naming here. In the following, we briefly recall the basics of the language of topologic. As it has just been indicated, formulas may contain two one-place operators: a modality K describing the knowledge of an agent and another one, $\Box$, describing (e.g., computational) effort. The domains for evaluating formulas are *subset spaces* $(X, \mathcal{O}, V)$ consisting of a non-empty set X of states, a set $\mathcal{O}$ of subsets of X representing the knowledge states of the agent, and a valuation V determining the states where the atomic propositions are true. The operator K then quantifies across any $U \in \mathcal{O}$, whereas $\Box$ quantifies 'downward' across $\mathcal{O}$ since shrinking and acquiring knowledge correspond to each other.

After [1] was published, several classes of subset spaces, including the ordinary topological ones, could be characterized within the framework of topologic; cf [2,3,4,5]. Actually, the topological language proved to be quite suitable for dealing with 'local' properties of points and sets (whereas more expressive power is needed to capture 'non-local' notions); see the corresponding chapter of the handbook [6] for more details regarding this and an overview of the current state of the art.

In the present paper, we are less interested in spatial aspects than in those related to time. Note that a temporal dimension already inheres in the effort modality $\Box$ since $\Box$ quantifies across future knowledge states; this was made more explicit, e.g., in the paper [7]. The new operator we consider, P, is the converse of $\Box$. Thus P refers to properties in the past by quantifying 'upward' across $\mathcal{O}$, i.e., over all knowledge states that contain the actual one. We shall, therefore, call P the *past operator* more often than not, and only sometimes emphasize its connection with the increase of sets.[1]

The epistemic relevance of P (in case this operator is regarded independent of $\Box$) is worth mentioning. While the effort operator is associated with *no forgetting* of agents (also called *perfect recall* in the literature), the increase operator P comes along with *no learning;* cf [10].

Integrating the past operator into topologic seems to be very natural. The reader might wonder why this has not been done up to now. However, sometimes this modality turns out to be temperamental, and we must use methods going beyond ordinary modal logic for taming it. But it is worth doing this since we really get a useful means of expression; e.g., the *overlap operator* studied in the paper [11] will be definable then. This was the main motivation for us to examine the properties of P. In retrospect, the promising results we obtained justify our approach.

The subsequent technical part of the paper is organized as follows. In the next section we recapitulate the language of topologic from [12] (which is the journal version of [1]), and we define the semantics of the past operator at the same time. We then give some examples of valid formulas of the extended language. Moreover, we touch on the question of the expressiveness of P. In Section 3, we present a list of axioms for topologic including the past operator. Unfortunately, we do not have a corresponding completeness theorem. However, as the first of the main issues of this paper we prove that the resulting logic is at least decidable. In Section 4, we deal with a certain hybridization of topologic including the past. Concerning the concepts from basic hybrid logic we need for that, see, e.g., [13], Sec. 7.3. In this case we actually obtain the desired

[1] It should be noted that a related modality was examined in the paper [8] with regard to certain systems of *linear time*. However, the framework here is much more general, and the technical details are completely different from those there. – For a treatment of the past in temporal logic of concurrency, see the paper [9].

meta-theorems for the arising logic, i.e., soundness and completeness with respect to subset spaces, and decidability. Furthermore, we point to a nice application of that system. Finally, we give a brief summary of the paper and mention future research.

2 The Extended Modal Language

In this section, we first define the modal language, including the past operator, for subset spaces. Second, we give some examples of valid formulas. Finally, the expressive power of the new language is compared to that of a previous extension of topologic.

Let PROP $= \{A, B, \ldots\}$ be a denumerable set of symbols. The elements of PROP are called *proposition letters*. We define the set WFF of *well-formed formulas over* PROP by the rule

$$\alpha \ ::= \ A \mid \neg\alpha \mid \alpha \wedge \beta \mid \mathsf{K}\alpha \mid \Box\alpha \mid \mathsf{P}\alpha.$$

The operators K and $\Box$ represent *knowledge* and *effort,* respectively, as it is common for topologic. The operator P is the counterpart of $\Box$. Since P is, therefore, related to the past we call this modality the *past operator*. The duals of K, $\Box$ and P are denoted L, $\Diamond$ and $\langle\mathsf{P}\rangle$, respectively. The missing boolean connectives $\top$, $\bot$, $\vee$, $\rightarrow$, $\leftrightarrow$ are treated as abbreviations, as needed.

We now turn to semantics. First, we define the domains for interpreting formulas. Given a set X, let $\mathcal{P}(X)$ be the powerset of X.

Definition 1 (Subset frames and subset spaces)

1. *Let $X \neq \emptyset$ be a set and $\mathcal{O} \subseteq \mathcal{P}(X)$ a set of subsets of X. Then, $\mathcal{F} := (X, \mathcal{O})$ is called a* (general) subset frame*.*
2. *A subset frame $\mathcal{F} = (X, \mathcal{O})$ satisfying $\{\emptyset, X\} \subseteq \mathcal{O}$ is called* special*.*
3. *Let $\mathcal{F} = (X, \mathcal{O})$ be a subset frame. The* set $\mathcal{N}_{\mathcal{F}}$ of neighbourhood situations *of $\mathcal{F}$ is defined by $\mathcal{N}_{\mathcal{F}} := \{(x, U) \mid x \in U \text{ and } U \in \mathcal{O}\}$. (Mostly, neighbourhood situations are written without brackets later on.)*
4. *Let $\mathcal{F}$ be a subset frame. A mapping V : PROP $\longrightarrow \mathcal{P}(X)$ is called an $\mathcal{F}$–*valuation*.*
5. *A* subset space *is a triple $\mathcal{M} := (X, \mathcal{O}, V)$, where $\mathcal{F} = (X, \mathcal{O})$ is a subset frame and V an $\mathcal{F}$–valuation; $\mathcal{M}$ is called* based on $\mathcal{F}$*.*

The requirement '$\{\emptyset, X\} \subseteq \mathcal{O}$' from item 2 of this definition is convenient, but in a sense insignificant for topologic because of the forward looking nature of the effort operator; cf [12], Sec. 1.1. This is no longer true for the extended system, as one will see later on. Nevertheless, we continue to deal with special subset frames in this paper as long as it is possible.

The next definition concerns the relation of satisfaction, which is defined with regard to subset spaces now.

Definition 2 (Satisfaction and validity). *Let $\mathcal{M} = (X, \mathcal{O}, V)$ be a subset space.*

1. *Let x, U be a neighbourhood situation of $\mathcal{F} = (X, \mathcal{O})$. Then*

$$\begin{array}{lcl} x, U \models_{\mathcal{M}} A & :\Longleftrightarrow & x \in V(A) \\ x, U \models_{\mathcal{M}} \neg\alpha & :\Longleftrightarrow & x, U \not\models_{\mathcal{M}} \alpha \\ x, U \models_{\mathcal{M}} \alpha \wedge \beta & :\Longleftrightarrow & x, U \models_{\mathcal{M}} \alpha \text{ and } x, U \models_{\mathcal{M}} \beta \\ x, U \models_{\mathcal{M}} \mathsf{K}\alpha & :\Longleftrightarrow & \forall\, y \in U : y, U \models_{\mathcal{M}} \alpha \\ x, U \models_{\mathcal{M}} \Box\alpha & :\Longleftrightarrow & \forall\, U' \in \mathcal{O} : (x \in U' \subseteq U \Rightarrow x, U' \models_{\mathcal{M}} \alpha) \\ x, U \models_{\mathcal{M}} \mathsf{P}\alpha & :\Longleftrightarrow & \forall\, U' \in \mathcal{O} : (U' \supseteq U \Rightarrow x, U' \models_{\mathcal{M}} \alpha)\,, \end{array}$$

   where $A \in$ PROP and $\alpha, \beta \in$ WFF. In case $x, U \models_{\mathcal{M}} \alpha$ is true we say that α holds in $\mathcal{M}$ at *the neighbourhood situation x, U.*
2. *A formula α is called* valid in $\mathcal{M}$ *(written '$\mathcal{M} \models \alpha$'), iff it holds in $\mathcal{M}$ at every neighbourhood situation of the frame $\mathcal{M}$ is based on.*

Note that the meaning of proposition letters is independent of neighbourhoods by definition, thus 'stable' with respect to $\Box$ and P. This fact is reflected by a special axiom below; see Section 3.

We now look for the formulas which are valid in all subset spaces. It is a known fact that the schema

$$\mathsf{K}\Box\alpha \rightarrow \Box\mathsf{K}\alpha$$

is a typical validity of topologic. This schema was called the *Cross Axiom* in the paper [12] and plays a key role in the completeness and the decidability proof for that logic. The Cross Axiom describes the basic interaction between knowledge and effort. It is not very surprising that there is a complementary schema for P.

Proposition 1. *Let $\mathcal{M}$ be any subset space. Then, for all $\alpha \in$ WFF we have that*

$$\mathcal{M} \models \mathsf{PK}\alpha \rightarrow \mathsf{KP}\alpha.$$

The easy proof of Proposition 1 is omitted here. – Quite recently, a modal operator O was considered which describes *overlapping* of sets within the framework of topologic; cf [11]. With the aid of O, it is possible to access also points that are distant from the actual one. The precise semantics of O in subset spaces $\mathcal{M} = (X, \mathcal{O}, V)$ at neighbourhood situations is as follows:

$$x, U \models_{\mathcal{M}} \mathsf{O}\alpha :\Longleftrightarrow \forall\, U' \in \mathcal{O} : \text{ if } x \in U', \text{ then } x, U' \models_{\mathcal{M}} \alpha$$

where $\alpha \in$ WFF. Compared to the semantics of the effort operator, the condition $U' \subseteq U$ obviously was left out; cf Definition 2. Thus shrinking appears as a special case of overlapping. It turned out that O is rather strong. In fact, with the aid of O even the *global modality* A (cf [13], Sec. 7.1) is definable in subset spaces based on special subset frames, actually through

$$\mathsf{A}\alpha :\equiv \mathsf{OK}\Box\alpha.$$

Note that this is true since $X \in \mathcal{O}$. But now the overlap operator itself can be defined with regard to such subset spaces. The defining clause reads

$$\mathsf{O}\alpha :\equiv \mathsf{P}\Box\alpha.$$

Consequently, the approach from [11] is subsumed under the present one in a sense. (For a fair comparison, however, one should take into account the issues of the next section as well.)

3 A Decidable Fragment of the Logic

Our starting point to this section is the system of axioms for topologic from [12]. We then add several schemata involving the new modality. After that we derive some auxiliary theorems of the resulting logic, topP, which are used for the proof of the decidability of topP. This proof makes up the main part of Section 3.

Unless otherwise noted, all subset spaces occurring in this section are based on special subset frames.

The complete list of axioms for topologic reads as follows:

1. All instances of propositional tautologies.
2. $\mathsf{K}(\alpha \to \beta) \to (\mathsf{K}\alpha \to \mathsf{K}\beta)$
3. $\mathsf{K}\alpha \to (\alpha \wedge \mathsf{K}\mathsf{K}\alpha)$
4. $\mathsf{L}\alpha \to \mathsf{K}\mathsf{L}\alpha$
5. $\Box(\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$
6. $(A \to \Box A) \wedge (\Diamond A \to A)$
7. $\Box\alpha \to (\alpha \wedge \Box\Box\alpha)$
8. $\mathsf{K}\Box\alpha \to \Box\mathsf{K}\alpha$,

where $A \in$ PROP and $\alpha, \beta \in$ WFF. In this way, it is expressed that for every Kripke model M validating these axioms

– the accessibility relation $\xrightarrow{\mathsf{K}}$ of M belonging to the knowledge operator is an equivalence,
– the accessibility relation $\xrightarrow{\Box}$ of M belonging to the effort operator is reflexive and transitive,
– the composite relation $\xrightarrow{\Box} \circ \xrightarrow{\mathsf{K}}$ is contained in $\xrightarrow{\mathsf{K}} \circ \xrightarrow{\Box}$ (this is usually called the *cross property*), and
– the valuation of M is constant along every $\xrightarrow{\Box}$–path, for all propositional letters.

We now turn to the axioms in which P occurs:

9. $\mathsf{P}(\alpha \to \beta) \to (\mathsf{P}\alpha \to \mathsf{P}\beta)$
10. $\alpha \to \Box\langle\mathsf{P}\rangle\alpha$
11. $\alpha \to \mathsf{P}\Diamond\alpha$
12. $\langle\mathsf{P}\rangle\mathsf{P}\alpha \to \mathsf{P}\langle\mathsf{P}\rangle\mathsf{L}\alpha$,

where A and α, β are as above. – Some comments on these axioms seem to be appropriate. Item 9 contains the usual distribution schema being valid for every normal modality. The schemata 10 and 11 say that the accessibility relations belonging to $\Box$ and P, respectively, are inverse to each other. Note that the latter relation is reflexive and transitive, which follows logically from the respective property of the former one. All this is well-known from usual tense logic; cf, e.g., [14], § 6. The final schema is to capture the property '$X \in \mathcal{O}$' of subset spaces axiomatically; cf the remark following Definition 1.

We obtain a logical system by adding the standard proof rules of modal logic, i.e., *modus ponens* and *necessitation with respect to each modality.* We call this system topP, indicating both topologic and the past operator.

Definition 3 (The logic). *Let* topP *be the smallest set of formulas which contains the axiom schemata 1 – 12 and is closed under the application of the following rule schemata:*

$$(\text{MODUS PONENS})\ \frac{\alpha \to \beta, \alpha}{\beta} \qquad (\Delta\text{–NECESSITATION})\ \frac{\alpha}{\Delta\alpha},$$

where $\alpha, \beta \in \mathrm{WFF}$ *and* $\Delta \in \{\mathsf{K}, \Box, \mathsf{P}\}$*. Then we let* $\vdash$ *denote* topP*–derivability.*

Some useful derivations are contained in the following lemma.

Lemma 1. *For all* $A \in \mathrm{PROP}$ *and* $\alpha \in \mathrm{WFF}$*, we have that*

1. $\vdash (A \to \mathsf{P}A) \wedge (\langle\mathsf{P}\rangle A \to A)$
2. $\vdash \mathsf{LP}\alpha \to \mathsf{PL}\alpha$
3. $\vdash \mathsf{PK}\alpha \to \mathsf{KP}\alpha$.

Proof. 1. From the second conjunct of Axiom 6 we get $\vdash \mathsf{P}\Diamond A \to \mathsf{P}A$ with the aid of P–necessitation, Axiom 9 and propositional reasoning. Axiom 11 yields $\vdash A \to \mathsf{P}\Diamond A$. This gives us the first conjunct of the desired schema. From the first conjunct of Axiom 6 we infer $\vdash \langle\mathsf{P}\rangle A \to \langle\mathsf{P}\rangle\Box A$. The dual of Axiom 11 and propositional reasoning then imply that $\vdash \langle\mathsf{P}\rangle A \to A$, as desired.

2. The formula $\mathsf{LP}\alpha \to \mathsf{P}\Diamond\mathsf{LP}\alpha$ is an instance of Axiom 11. By using, among other things, the dual of Axiom 8 we get $\vdash \mathsf{P}\Diamond\mathsf{LP}\alpha \to \mathsf{PL}\Diamond\mathsf{P}\alpha$. As a consequence of the dual of Axiom 10, we have that $\vdash \mathsf{PL}\Diamond\mathsf{P}\alpha \to \mathsf{PL}\alpha$. Propositional reasoning now yields $\vdash \mathsf{LP}\alpha \to \mathsf{PL}\alpha$.
3. From Axiom 3 we infer $(*)\ \vdash \mathsf{K}\alpha \to \alpha$, for all $\alpha \in \mathrm{WFF}$. Thus $\vdash \alpha \to \mathsf{L}\alpha$, for all formulas α. From that we get $\vdash \mathsf{PK}\alpha \to \mathsf{LPK}\alpha$ since this formula is an instance of the just derived schema. Axiom 4 implies $\vdash \mathsf{LPK}\alpha \to \mathsf{KLPK}\alpha$. With the aid of item 2 of this lemma we obtain $\vdash \mathsf{KLPK}\alpha \to \mathsf{KPLK}\alpha$. The dual of Axiom 4 gives us $\vdash \mathsf{KPLK}\alpha \to \mathsf{KPK}\alpha$, and $(*)$ finally implies $\vdash \mathsf{KPK}\alpha \to \mathsf{KP}\alpha$. It follows that $\vdash \mathsf{PK}\alpha \to \mathsf{KP}\alpha$, as desired.

The schema from item 1 of this lemma is complementary to Axiom 6. Both schemata together correspond to the stability condition mentioned in Section 2, right after Definition 2. The schema contained in item 3 is the one from Proposition 1 and will be called the *Reverse Cross Axiom;* see the discussion preceding Proposition 1.

The next proposition is quite obvious.

Proposition 2 (Soundness). *Let $\alpha \in \mathrm{WFF}$ be formula. If α is* topP*–derivable, then α is valid in all subset spaces.*

A possible proof of completeness must use the canonical model of topP in some way. We fix several notations concerning this model. Let $\mathcal{C}$ be the set of all maximal topP–consistent sets of formulas. Furthermore, let $\xrightarrow{\mathsf{K}}$, $\xrightarrow{\Box}$ and $\xrightarrow{\mathsf{P}}$ be the accessibility relations on $\mathcal{C}$ induced by the modalities K, $\Box$ and P, respectively.

Three useful properties of the canonical model are listed in the subsequent lemma.

Lemma 2. *1. For all $\Psi, \Gamma, \Theta \in \mathcal{C}$ satisfying $\Psi \xrightarrow{\mathsf{K}} \Gamma \xrightarrow{\mathsf{P}} \Theta$ there exists $\Xi \in \mathcal{C}$ such that $\Psi \xrightarrow{\mathsf{P}} \Xi \xrightarrow{\mathsf{K}} \Theta$.*

2. For all $\Psi, \Gamma, \Theta \in \mathcal{C}$ satisfying $\Psi \xrightarrow{\mathsf{P}} \Gamma$ and $\Psi \xrightarrow{\mathsf{P}} \Theta$ there exist $\Xi, \Phi \in \mathcal{C}$ such that $\Gamma \xrightarrow{\mathsf{P}} \Xi$ and $\Theta \xrightarrow{\mathsf{P}} \Phi \xrightarrow{\mathsf{L}} \Xi$.

3. $\left(\xrightarrow{\mathsf{K}} \cup \xrightarrow{\Box} \cup \xrightarrow{\mathsf{P}}\right)^* \subseteq \xrightarrow{\mathsf{P}} \circ \xrightarrow{\mathsf{K}} \circ \xrightarrow{\Box}$.

We only give some ideas, but not a detailed proof of Lemma 2 here. (The missing details will be contained in the full version of this paper.) The condition stated in item 1 follows from the Reverse Cross Axiom (see Lemma 1.3) and is, therefore, called the *reverse cross property.* The condition from item 2 is, among other things, a consequence of Axiom 12. Since a kind of confluence of the accessibility relation $\xrightarrow{\mathsf{P}}$ is forced, we call item 2 the *pseudo Church-Rosser property.* Quite some modal proof theory has to be applied in the proofs of items 1 and 2 of the lemma. Item 3 ensues from an iterated application of the previous items and the usual cross property, respectively.

Unfortunately, we do not have a proof of the completeness of topP with respect to the class of all subset spaces based on special frames. An adaption to the new system, including a suitable extension, of the corresponding proof for topologic (cf [12], Sec. 2.2) does not work anyhow.[2] This seems to be true even if the speciality condition (item 2 of Definition 1) is weakened a little, in the following way.

[2] If we drop Axiom 12, then completeness for subset spaces based on general frames can be proved in the just indicated way; cf [15]. The presence of that axiom, however, may be viewed as an indication of incompleteness. This is due to the formal similarity of Axiom 12 to the *Weak Directedness Axiom for intersection spaces* from [12], Sec. 2.4. It is a known fact that the latter is incomplete. In particular, the question arises whether all formulas of the form $\langle\mathsf{P}\rangle\mathsf{P}\alpha \rightarrow \mathsf{P}\langle\mathsf{P}\rangle\alpha$, which are sound for special subset spaces, are topP–derivable.

Definition 4 (Past-directed subset frames). *A (general) subset frame $\mathcal{F} = (X, \mathcal{O})$ is called* past-directed, *iff for all $U_1, U_2 \in \mathcal{O}$ there exists some $U \in \mathcal{O}$ such that $U \supseteq U_1 \cup U_2$.*

On the other hand, Lemma 2.2 suggests that completeness could hold for the larger class of all spaces based on past-directed frames. We will refer to this class of structures in the next section.

Contrasting those bad news, we at least can show that topP is decidable. This is done in the following.

Since $\mathsf{topologic}$ does not satisfy the finite model property with respect to subset spaces (cf [12], Sec. 1.3), topP too lacks this property. However, as in the former case this deficiency can be circumvented by a detour via suitable Kripke models.

Subsequently, let R and K, S and $\Box$, and T and P, respectively, correspond to each other.

Definition 5 (topP–model). *Let $M := (W, \{R, S, T\}, V)$ be a trimodal model, i.e., $R, S, T \subseteq W \times W$ are binary relations and V is a valuation in the usual sense. Then, M is called a* topP–model *iff the following conditions are satisfied:*

1. *R is an equivalence relation, and S is reflexive and transitive,*
2. *S and T are inverse to each other,*
3. *$S \circ R \subseteq R \circ S$,*
4. *M satisfies the pseudo Church-Rosser property with respect to T and R,*
5. *for all $w, w' \in W$ and $A \in \mathrm{PROP}$: if $w \, S \, w'$, then $w \in V(A)$ iff $w' \in V(A)$.*

It is easy to see that all the axioms considered above are sound with respect to the class of all topP–models. Furthermore, the canonical model of topP is an example of a topP–model. (As to item 4 from Definition 5, cf Lemma 2.2). This gives us the following theorem.

Theorem 1 (Kripke completeness). *The logic topP is sound and complete with respect to the class of all topP–models.*

We now use the method of *filtration* for proving the finite model property of topP with respect to topP–models. The proceeding here follows the one from [12], Sec. 3.3, to a large degree. Thus we may be brief regarding this and stress the new aspects only.

For a given topP–consistent formula $\alpha \in \mathrm{WFF}$, we define a filter set $\Sigma \subseteq \mathrm{WFF}$ as follows. We first let

$$\Sigma_0 := \mathrm{sf}(\alpha) \cup \{\neg\beta \mid \beta \in \mathrm{sf}(\alpha)\},$$

where $\mathrm{sf}(\alpha)$ denotes the set of all subformulas of α. Second, we form the closure of Σ_0 under finite conjunctions of pairwise distinct elements of Σ_0. Third, we close under single applications of the operator L. And finally, we form the set of all subformulas of the formulas obtained up to now.[3] Let Σ denote the resulting set. Then Σ is finite and subformula closed.

[3] Note that this step is really necessary since L is an abbreviation.

We consider the respective *smallest* filtrations of the accessibility relations $\xrightarrow{\mathsf{K}}$, $\xrightarrow{\Box}$, and $\xrightarrow{\mathsf{P}}$, of the canonical model of topP; cf [14], § 4. Let

$$M := (W, \{R, S, T\}, V)$$

be the corresponding filtration of a suitably generated submodel M' of that model in which the valuation V assigns the empty set to all proposition letters not occurring in Σ. Then we have the following lemma.

Lemma 3. *The structure M is a finite* topP*–model of which the size depends computably on the length of α.*

Proof. Most of it is clear from the definitions and the proof of [12], Theorem 2.11. In particular, the finiteness of W follows from the finiteness of Σ. Only items 2 and 4 of Definition 5 have to be checked yet.

For item 2, let Γ, Θ be two points of the canonical model such that $[\Gamma]\ S\ [\Theta]$, where the brackets $[\ldots]$ indicate the respective classes. Due to the definition of the minimal filtration there are $\Gamma' \in [\Gamma]$ and $\Theta' \in [\Theta]$ such that $\Gamma' \xrightarrow{\Box} \Theta'$. Since $\Theta' \xrightarrow{\mathsf{P}} \Gamma'$ then holds because of the dual of Axiom 10, we conclude $[\Theta]\ T\ [\Gamma]$ from that with the aid of the first filtration condition (marked (F1) in [14], § 4). This shows $S^{-1} \subseteq T$. By interchanging the roles of S and T, the property $T^{-1} \subseteq S$ can be established in a similar manner. Now, the condition stated in item 2 of Definition 5 easily follows.

Item 4 is a bit harder to prove. We make use of the following property $(*)$ of the intermediate model M' introduced above:

$(*)$ For every finite set $\{\Gamma_1, \ldots, \Gamma_n\}$ of points of M' there exists a point Γ of M' such that $(\Gamma, \Gamma_i) \in \xrightarrow{\mathsf{K}} \circ \xrightarrow{\Box}$ for all $i = 1, \ldots, n$ (where $n \geq 1$).

This is consequence of Lemma 2.3 and the fact that M' is generated. Now, it is clear from the first filtration condition again that the property

$$(\Psi, \Theta) \in \xrightarrow{\mathsf{K}} \circ \xrightarrow{\Box}$$

passes down to the filtration (where Ψ, Θ are any maximal topP–consistent sets). Thus M is generated in the following sense. For all classes $[\Phi] \in W$ we have that

$$([\Gamma], [\Phi]) \in R \circ S,$$

where Γ is obtained according to $(*)$ after arbitrary representatives of the finitely many classes contained in W have been chosen. From that, the properties of R, and item 2, the validity of the pseudo Church-Rosser property for M is clear.

Since the topP–model M from Lemma 3 realizes α, the claimed decidability result follows readily.

Theorem 2 (Decidability). *The set of all* topP*–derivable formulas is decidable.*

Concerning the complexity of the topP–satisfiability problem, we only mention that this problem is very likely hard for EXPTIME. This is due to the (implicit) presence of the global modality (see Section 2); cf [16], Ch. 2.2.

4 Hybridization

Concluding the technical part of the paper, we develop a hybrid logic version of topP. This extension, HtopP, rectifies the shortcomings of the former system to some extent and simultaneously generalizes previous hybridizations of topologic (e.g., the one from [11]). By enriching the language once again we first obtain much more expressive power concerning properties of relations, and this can already be achieved by simply adding suitable sets of nominals to the ground language; see the just cited paper for some examples. Second, HtopP turns out to be sound and complete for subset spaces based on past-directed frames. And finally, HtopP is decidable, too. We outline only the completeness proof in this section.[4] Actually, we will have 'almost canonical' completeness. As an application, we show that the hybrid logic of *directed spaces,* cf [5], is finitely axiomatizable and decidable.[5] The first property is false for the modal fragment, and the second one apparently was unknown before the hybrid methods came into play.

We now carry out all these things. For a start, we define the extended language. As we already indicated above, we merely add two sets of nominals to the language underlying topP. If the denotation of a nominal is non-empty, then it should be either a unique state or a distinguished set of states. Let

$$\mathrm{N}_{stat} = \{i, j, \ldots\} \text{ and } \mathrm{N}_{sets} = \{I, J, \ldots\}$$

be the corresponding sets of symbols, which we call *names of states* and *names of sets,* respectively. We assume that PROP, N_{stat} and N_{sets} are mutually disjoint.

Definition 6 (Subset spaces with names)

1. *Let $\mathcal{F} = (X, \mathcal{O})$ be a past-directed subset frame. A* hybrid $\mathcal{F}$–valuation *is a mapping*
$$V : \mathrm{PROP} \cup \mathrm{N}_{stat} \cup \mathrm{N}_{sets} \longrightarrow \mathcal{P}(X)$$
such that
(a) $V(i)$ is either $\emptyset$ or a singleton subset of X, for every $i \in \mathrm{N}_{stat}$, and
(b) $V(A) \in \mathcal{O}$ for every $A \in \mathrm{N}_{sets}$.
2. *A* subset space with names *(or, in short, an* SSN*) is a triple $(X, \mathcal{O}, V)$, where $\mathcal{F} = (X, \mathcal{O})$ is a subset frame as in item 1 and V a hybrid $\mathcal{F}$–valuation.*

Note that nominals may have an empty denotation. This is appropriate for our purposes since it simplifies the proof of the subsequent Theorem 3 a little, but not common in standard hybrid logic.

[4] In order to establish decidability, the techniques from the previous section have to be tailored to the hybrid context. Due to space limitations we cannot give too many details regarding this here.

[5] A subset frame $\mathcal{F} = (X, \mathcal{O})$ is called *directed,* iff $\forall U_1, U_2 \in \mathcal{O}, \forall x \in X :$ if $x \in U_1 \cap U_2,$ then $\exists U \in \mathcal{O} : x \in U \subseteq U_1 \cap U_2$.

Definition 7 (Satisfaction for nominals). *Let $\mathcal{M} := (X, \mathcal{O}, V)$ be an SSN and x, U a neighbourhood situation of the underlying frame. Then*

$$\begin{aligned} x, U \models_{\mathcal{M}} i &:\Longleftrightarrow x \in V(i) \\ x, U \models_{\mathcal{M}} I &:\Longleftrightarrow V(I) = U, \end{aligned}$$

for all $i \in \mathrm{N}_{stat}$ and $I \in \mathrm{N}_{sets}$.

We now turn to the axioms for nominals. These axioms are divided into two groups. The formulas of the first group provide for the right interpretation of the names of states and sets, respectively, in the canonical model which is used for the proof of completeness below.

13. $i \wedge \alpha \rightarrow \mathsf{K}(i \rightarrow \alpha)$
14. $I \rightarrow \mathsf{K}I$
15. $\mathsf{K}\Box\,(I \wedge \alpha \rightarrow \mathsf{L}\beta) \vee \mathsf{K}\Box\,(I \wedge \beta \rightarrow \mathsf{L}\alpha)$,

where $i \in \mathrm{N}_{stat}$, $I \in \mathrm{N}_{sets}$ and $\alpha, \beta \in \mathrm{WFF}$.[6] The schema 13 says that only one point of any $\xrightarrow{\mathsf{K}}$–equivalence class can be named by a fixed state name. And the schema 14 guarantees that a set name of such a class is valid across the whole class. Finally, Axiom 15 captures the property that every element of N_{sets} denotes at most one $\xrightarrow{\mathsf{K}}$–class. Note the formal similarity of this schema to the linearity schema L from classical modal logic (cf [14], p 22).

The following two axioms are responsible for the fact that really a structure of subset space can be ensured with the aid of that model.

16. $i \wedge I \rightarrow \Box\,(\Diamond(i \wedge I) \rightarrow i \wedge I)$
17. $\mathsf{K}(\Diamond J \rightarrow \Diamond I) \wedge \mathsf{L}\Diamond J \rightarrow \Box\,(I \rightarrow \mathsf{L}\Diamond J)$,

where $i, j \in \mathrm{N}_{stat}$ and $I, J \in \mathrm{N}_{sets}$. Axiom 16 corresponds to the antisymmetry of the relation $\xrightarrow{\Box}$ on the canonical model we construct later on. The part Axiom 17 plays is less obvious. Roughly speaking, the inclusion relation of subsets is arranged correctly by this axiom.

By fixing the hybrid proof rules we get to the logical system HtopP. The new rules are called NAME and ENRICHMENT, respectively. We comment on these rules right after the next definition.

Definition 8 (The hybrid logic). *Let* HtopP *be the smallest set of formulas which contains the axiom schemata 1 – 17 and is closed under the application of the standard modal rule schemata and the following ones:*

$$(\textsc{name}_{stat})\ \ \frac{j \rightarrow \beta}{\beta} \qquad (\textsc{name}_{sets})\ \ \frac{J \rightarrow \beta}{\beta}$$

$$(\nabla\text{–}\textsc{enrichment})\ \ \frac{\langle \mathsf{P}\rangle \mathsf{L}\Diamond\,(i \wedge I \wedge \nabla(j \wedge J \wedge \alpha)) \rightarrow \beta}{\langle \mathsf{P}\rangle \mathsf{L}\Diamond\,(i \wedge I \wedge \nabla\alpha) \rightarrow \beta},$$

where $\alpha, \beta \in \mathrm{WFF}$, $i, j \in \mathrm{N}_{stat}$, $I, J \in \mathrm{N}_{sets}$, $\nabla \in \{\mathsf{L}, \Diamond, \langle \mathsf{P}\rangle\}$, and j, J are new *each time, i.e., do not occur in any other syntactic building block of the respective rule.*

[6] From now on, WFF denotes the set of formulas of the enriched language.

For the reader being not familiar with these unorthodox proof rules a 'contrapository' reading is suggested; e.g., the rule ($\textsc{name}_{stat}$) is to be read 'if β is satisfiable, then $j \wedge \beta$ is satisfiable, too' (provided that the nominal j does not occur in β). The soundness of the hybrid rules should be apparent from that now. Technically, the NAME and ENRICHMENT rules are used for establishing an appropriate *Lindenbaum Lemma,* which makes up the first step in the proof of the following theorem.

Theorem 3 (Hybrid completeness). HtopP *is sound and complete with respect to the class of all subset spaces with names.*

Proof. The proof of Theorem 3 goes along the lines of the proof of Theorem 3.2 from [17], but some modifications are required during the construction of the model refuting a given non-derivable formula α. We start off with the canonical model of HtopP. Let us retain the designations $\mathcal{C}$, $\xrightarrow{\mathsf{K}}$, $\xrightarrow{\Box}$, and $\xrightarrow{\mathsf{P}}$ from the previous section, and let Γ_0 be a maximal consistent set containing $\neg\alpha$. If Γ is any element of $\mathcal{C}$, then Γ is called

- *named* iff Γ contains some $i \in \mathrm{N}_{stat}$ and some $I \in \mathrm{N}_{sets}$, and
- *enriched* iff, for every $\nabla \in \{\mathsf{L}, \Diamond, \langle\mathsf{P}\rangle\}$, whenever $\langle\mathsf{P}\rangle\mathsf{L}\Diamond\,(i \wedge I \wedge \nabla\alpha) \in \Gamma$, then there are $j \in \mathrm{N}_{stat}$ and $J \in \mathrm{N}_{sets}$ such that

$$\langle\mathsf{P}\rangle\mathsf{L}\Diamond\,(i \wedge I \wedge \nabla(j \wedge J \wedge \alpha)) \in \Gamma.$$

Now, it is our first task to extend Γ_0 to a named and enriched maximal consistent set Γ' in the language to which enough new constants have been added. This can be achieved in the usual way by means of the Lindenbaum Lemma mentioned above.

Secondly, we consider the canonical model of HtopP which is formed with respect to the latter language. Let $\mathcal{M}_0$ be the submodel generated by Γ' of this model. We then have that the interpretation of the set names in $\mathcal{M}_0$ is uniquely determined.[7]

Lemma 4. *Let D be the domain of $\mathcal{M}_0$ and let I be a set name. Assume that both of $\Gamma, \Gamma' \in D$ contain I. Then $\Gamma \xrightarrow{\mathsf{K}} \Gamma'$.*

Proof. Suppose on the contrary that there are two points $\Gamma, \Gamma' \in D$ such that $I \in \Gamma \cap \Gamma'$, but not $\Gamma \xrightarrow{\mathsf{K}} \Gamma'$. Then there is some $n \in \mathbb{N}$ and a sequence $\Gamma_0, \Gamma_1, \ldots, \Gamma_n$ of elements of D such that $\Gamma_0 = \Gamma$, $\Gamma_n = \Gamma'$, and Γ_i, Γ_{i+1} are connected by either $\xrightarrow{\mathsf{K}}$, $\xrightarrow{\Box}$, or $\xrightarrow{\mathsf{P}}$, for all $i = 0 \ldots n-1$. This is true since $\mathcal{M}_0$ is generated. Now, Lemma 2.3 implies that there exist $\Psi, \Theta \in D$ such that

$$\Gamma \xrightarrow{\mathsf{P}} \Psi \xrightarrow{\mathsf{K}} \Theta \xrightarrow{\Box} \Gamma'.$$

The desired contradiction then follows with the aid of Axiom 15.

[7] We exemplarily focus on this point of the completeness proof a little more detailedly.

Let D' be the set of all points of D that are named, and let $\mathcal{M}'$ be the corresponding substructure of $\mathcal{M}_0$. $\mathcal{M}'$ is the model we want to operate on. The subsequent *Existence Lemma* is due to our special notion of enrichment.

Lemma 5. *Assume that $\Gamma \in D'$ contains $\nabla\alpha$, where $\nabla \in \{\mathsf{L}, \Diamond, \langle\mathsf{P}\rangle\}$ is the respective dual of $\Delta \in \{\mathsf{K}, \Box, \mathsf{P}\}$. Then there exists some $\Theta \in D'$ satisfying $\Gamma \xrightarrow{\Delta} \Theta$ and $\alpha \in \Theta$.*

The desired model falsifying α can be obtained as a certain space of partial functions, X, over the model $\mathcal{M}'$ now. The domain $\mathrm{dom}(h)$ of every such function $h \in X$ is a subset of the set

$$\mathcal{Q} := \{[\Gamma] \mid \Gamma \in D'\}$$

of all equivalence classes $[\Gamma] := \{\Gamma' \in D' \mid \Gamma \xrightarrow{\mathsf{K}} \Gamma'\}$ of the accessibility relation induced by the modality K. Actually, $\mathrm{dom}(h)$ is maximal in $\mathcal{Q}$ with regard to the following two conditions:

1. $h([\Gamma]) \in [\Gamma]$ for all $[\Gamma] \in \mathrm{dom}(h)$, and
2. $h([\Gamma]) \xrightarrow{\Box} h([\Theta])$ for all $[\Gamma], [\Theta] \in \mathrm{dom}(h)$ satisfying $[\Gamma] \preccurlyeq [\Theta]$;

the *precedence relation* $\preccurlyeq$ occurring in the second item is defined by

$$[\Gamma] \preccurlyeq [\Theta] :\Longleftrightarrow \exists\, \Gamma' \in [\Gamma], \Theta' \in [\Theta] : \Gamma' \xrightarrow{\Box} \Theta',$$

for all points $\Gamma, \Theta \in D'$.[8] We write $h_\Gamma := h([\Gamma])$ in case $h([\Gamma])$ exists. Furthermore, we let

- $U_{[\Gamma]} := \{h \in X \mid h_\Gamma \text{ exists}\}$, for all $\Gamma \in D'$,
- $\mathcal{O} := \{U_{[\Gamma]} \mid \Gamma \in D'\}$, and
- $V : \mathrm{PROP} \cup \mathrm{N}_{stat} \cup \mathrm{N}_{sets} \longrightarrow \mathcal{P}(X)$ be defined by

  $$h \in V(c) :\Longleftrightarrow c \in h_\Gamma \text{ for some } \Gamma \in D' \text{ for which } h_\Gamma \text{ exists},$$

  for all $c \in \mathrm{PROP} \cup \mathrm{N}_{stat} \cup \mathrm{N}_{sets}$.

We then get that $\mathcal{M} := (X, \mathcal{O}, V)$ is a past-directed subset space for which the relevant *Truth Lemma* is valid. With that, the completeness of the system HtopP with respect to the intended class of structures can be concluded in a standard manner.

In addition, we obtain that the hybrid logic of past-directed subset spaces is decidable.

Theorem 4 (Hybrid decidability). *The set of all* HtopP*–derivable formulas is decidable.*

[8] Note that again quite some proof theory has to be used for establishing that h is well-defined.

As aforementioned, we have to skip the proof of this theorem here. For getting an idea of the proceeding the reader is referred to [17], proof of Theorem 4.4. But we at least want to emphasize the point crucial to hybrid decidability. This concerns the verification of the final group of axioms (16 and 17 above) with regard to a filtration. Actually, it is not necessary to establish the corresponding property on the filtrated model, but only to validate those instances of the schemata in which nominals from the filter set occur. This can be achieved by modifying that filter set appropriately.

At the end of this section, we revisit the 'guarded jump'–operators investigated in the paper [18]. For convenience, we remind the reader of the semantics of these operators. Let $\mathcal{M} = (X, \mathcal{O}, V)$ be an SSN and x, U a neighbourhood situation of the underlying frame. Then

$$x, U \models_{\mathcal{M}} [\epsilon_I]\alpha :\Longleftrightarrow \text{ if } x \in V(I), \text{ then } x, V(I) \models_{\mathcal{M}} \alpha,$$

for all $I \in \mathrm{N}_{sets}$ and $\alpha \in$ WFF. The 'guarded jump'–operators are, therefore, named variants of the overlap operator considered in Section 2. Thus it is hardly surprising $[\epsilon_I]$ can be defined as well:

$$[\epsilon_I]\alpha :\equiv \mathsf{P}\Box(I \rightarrow \alpha),$$

for all $\alpha \in$ WFF. As a consequence, we get that all the results obtained in the paper [18] can be inferred from the theorems of the present section. This applies to the hybrid logic of directed SSNs[9] and, in particular, the modal logic of directed spaces; cf [18], Theorem 20 and Corollary 21, and the discussion in the introduction to this section.

Theorem 5 (The logic of directed spaces)

1. *The hybrid logic of directed SSNs is finitely axiomatizable and decidable.*
2. *The topo-logic of directed spaces is decidable.*

More details concerning this application of the hybrid formalism will be contained in the full version of this paper.

5 Concluding Remarks

In this paper, Moss and Parikh's topologic was extended by a modality reversing the effort operator. While this goes smoothly for general subset frames, some complications arise if special frames form the semantic basis. In this case we could identify a natural fragment, topP, of the accompanying logic and prove its decidability. Incorporating concepts from hybrid logic then yielded more satisfactory results. We established the soundness, completeness, and decidability, of the hybrid logic of past-directed subset spaces. From that, the decidability of the logic of directed spaces was obtained as a corollary.

It remains to solve the completeness problem for topP. Moreover, the complexity of the logics must be determined each time. All this is postponed to future research.

[9] The axiom capturing directedness reads $\Diamond I \wedge \Diamond J \rightarrow \Diamond \mathsf{K}\left(\langle\epsilon_I\rangle\top \wedge \langle\epsilon_J\rangle\top\right)$, where $I, J \in \mathrm{N}_{sets}$ and $\langle\epsilon_I\rangle$ denotes the dual of $[\epsilon_I]$.

References

1. Moss, L.S., Parikh, R.: Topological reasoning and the logic of knowledge. In Moses, Y., ed.: Theoretical Aspects of Reasoning about Knowledge (TARK 1992), Los Altos, CA, Morgan Kaufmann (1992) 95–105
2. Georgatos, K.: Knowledge theoretic properties of topological spaces. In Masuch, M., Pólos, L., eds.: Knowledge Representation and Uncertainty, Logic at Work. Volume 808 of Lecture Notes in Artificial Intelligence., Springer (1994) 147–159
3. Georgatos, K.: Knowledge on treelike spaces. Studia Logica **59** (1997) 271–301
4. Heinemann, B.: Topological Modal Logics Satisfying Finite Chain Conditions. Notre Dame Journal of Formal Logic **39** (1998) 406–421
5. Weiss, M.A., Parikh, R.: Completeness of certain bimodal logics for subset spaces. Studia Logica **71** (2002) 1–30
6. Aiello, M., Pratt-Hartmann, I., van Benthem, J.: The logic of space (2007?) To appear. See URL http://dit.unitn.it/ aiellom/hsl/.
7. Heinemann, B.: Temporal Aspects of the Modal Logic of Subset Spaces. Theoretical Computer Science **224** (1999) 135–155
8. Heinemann, B.: Linear tense logics of increasing sets. Journal of Logic and Computation **12** (2002) 583–606
9. Lichtenstein, O., Pnueli, A., Zuck, L.: The glory of the past. In Parikh, R., ed.: Logics of Programs. Volume 193 of Lecture Notes in Computer Science., Berlin, Springer (1985) 196–218
10. Halpern, J.Y., van der Meyden, R., Vardi, M.Y.: Complete Axiomatizations for Reasoning about Knowledge and Time. SIAM Journal on Computing **33** (2004) 674–703
11. Heinemann, B.: Regarding overlaps in ‘topologic’. In Governatori, G., Hodkinson, I., Venema, Y., eds.: Advances in Modal Logic 6, London, College Publications (2006) 259–277
12. Dabrowski, A., Moss, L.S., Parikh, R.: Topological reasoning and the logic of knowledge. Annals of Pure and Applied Logic **78** (1996) 73–110
13. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Volume 53 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Cambridge (2001)
14. Goldblatt, R.: Logics of Time and Computation. second edn. Volume 7 of CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, CA (1992)
15. Heinemann, B.: Some spatial and spatio-temporal operators derived from the topological view of knowledge. In Wilson, D., Sutcliffe, G., eds.: Proceedings 20th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2007), Menlo Park, CA, AAAI Press (2007) To appear.
16. Spaan, E.: Complexity of Modal Logics. PhD thesis, ILLC, Universiteit van Amsterdam (1993)
17. Heinemann, B.: A hybrid logic for reasoning about knowledge and topology (2005) Under review for journal publication. For a preliminary version see URL http://www.informatik.fernuni-hagen.de/thi1/ber.ps.
18. Heinemann, B.: A two-sorted hybrid logic including guarded jumps. In Schmidt, R., Pratt-Hartmann, I., Reynolds, M., Wansing, H., eds.: Advances in Modal Logic 5, London, King’s College Publications (2005) 73–92

A Note on Rewriting Proofs and Fibonacci Numbers

Max Kanovich

Computer Science Dept., Queen Mary, University of London,
Mile End Road, London, E1 4NS, UK
mik@dcs.qmul.ac.uk

Abstract. One basic activity in combinatorics is to establish combinatorial identities by so-called 'bijective proofs,' which amounts to constructing explicit bijections between two types of the combinatorial objects in question.

The aim of this paper is to show how techniques from the formal logic world can be applied directly to such problems studied completely independently in the world of combinatorics.

The basic idea is to characterize equinumerous partition ideals in terms of the minimal elements generating the order filters, the complements to the original ideals.

For the case where the minimal elements of each of these order filters are disjoint, we have developed a general automated method to remove the mystery of certain known results and establish new results in the theory of integer partitions: Kanovich [Finding direct partition bijections by two-directional rewriting techniques. Discrete Math., 285 (1-3) (2004) 151-166], and Kanovich [The two-way rewriting in action: Removing the mystery of Euler-Glaisher's map. Discrete Math., (2006), doi:10.1016/j.disc.2006.10.005].

Here we address the case of filters having overlapping minimal elements. Essentially this is a case study related to the Fibonacci numeration system.

In addition to a 'bijective proof' for Zeckendorf's theorem - that every positive integer is uniquely representable within the Fibonacci numeration system, we establish 'bijective proofs' for a new series of partition identities related to Fibonacci and Lucas numbers.

The main result is proved using the idea of filter supports, and rewriting systems on multisets having overlapping patterns.

The main technical step is a suitable construction of such a rewriting system and then a proof that this rewriting system and the system consisting of its reverse rewriting rules, both have the Church-Rosser property, with providing the bijections we are looking for.

Keywords: multiset rewriting, Church-Rosser property, confluence, termination, strong normalization, combinatorics, integer partitions, partition identities, Fibonacci numbers.

1 Motivating Examples and Summary

One basic activity in combinatorics is to establish partition identities by so-called 'bijective proofs,' which amounts to constructing explicit bijections between two classes of the partitions at hand.

Forming an interdisciplinary bridge between Theoretical Computer Science and Combinatorics, the aim of this paper is to show how techniques from the formal logic world can be applied directly to specific problems studied completely independently in the theory of integer partitions.

The basic idea is to characterize equinumerous partition ideals in terms of the minimal elements generating the order filters, the complements to the original ideals.

The *novelty* of our approach to the combinatorics of partitions is in the use of rewriting techniques (*two-directional* in the sense that forward and backward applications of rewrite rules head respectively for two different normal forms) for the purpose of establishing *explicit relevant bijections* between partitions of two different types (represented by these normal forms).

The case where the minimal elements of each of the order filters mentioned above are disjoint is fully covered by a general approach developed in Kanovich [7,8].

Here we will address the challenging case of filters having overlapping minimal elements.

An inspiring example is the Fibonacci numeration system:

Theorem 1 (Zeckendorf [14]). *Every positive integer is uniquely representable as a sum of distinct Fibonacci numbers, but where no two consecutive Fibonacci numbers are used.*

We will use the two-directional rewriting technique to construct a 'bijective proof' for Zeckendorf's theorem and to establish 'bijective proofs' for a new series of partition identities related to Fibonacci and Lucas numbers.
The Fibonacci numbers u_i and Lucas numbers l_i are defined as

$$\begin{array}{ll} u_1 = 1,\ \ u_2 = 1,\ \ u_{i+2} = u_i + u_{i+1} & (i = 1, 2, 3, \ldots), \\ l_1 = 2,\ \ \ l_2 = 1,\ \ \ l_{i+2} = l_i + l_{i+1} & (i = 1, 2, 3, \ldots), \end{array}$$

2 Backgrounds

Let us recall the background material with which we are dealing (see, for instance, Andrews[1]).

An *integer partition* of

$$n = m_1 + m_2 + \cdots + m_k$$

can be identified as a *multiset* M consisting of positive integers m_1, m_2,..., m_k.

We will represent this M as

$$M = \{m_1, m_2, \ldots, m_k\},$$

where the number of copies of some m shows the multiplicity of the m within M.

Each m_i is called a *part* of the partition. This sum $m_1+m_2+\cdots+m_k$ will be also denoted by $\|M\|$:

$$\|M\| = m_1 + m_2 + \cdots + m_k.$$

Definition 1. *Let $p(\mathcal{C}, n)$ stand for the number of partitions of n that belong to a given class $\mathcal{C}$.*

Two classes of partitions $\mathcal{C}_1$ and $\mathcal{C}_2$ are called equinumerous, *if for all n:*

$$p(\mathcal{C}_1, n) = p(\mathcal{C}_2, n).$$

Example 1. Zeckendorf's theorem says that:

> For any positive integer n, the number of partitions of n into ones equals the number of partitions of n into distinct Fibonacci parts with no consecutive Fibonacci parts (both numbers are equal to 1).

Definition 2. *(Andrews[1]) Generally, the classes $\mathcal{C}$ of partitions considered in the literature have the 'local' property that if M is a partition in $\mathcal{C}$ and one or more parts are removed from M to form a new partition M', then M' is also in $\mathcal{C}$.*

Such a class $\mathcal{C}$ is called a partition ideal, *or an* order ideal *in terms of the lattice $\mathcal{P}$ of finite multisets of positive integers, ordered by $\subseteq$.* [1]

Dually, a class $\mathcal{F} \subseteq \mathcal{P}$ is an order filter *if $M' \in \mathcal{F}$, whenever $M \in \mathcal{F}$ and $M \subseteq M'$.*

It is readily seen that $\mathcal{C}$ is a partition ideal if and only if its complement $\overline{\mathcal{C}}$ is an order filter.

Definition 3. *M is* minimal *in an order filter $\mathcal{F}$, if $M \in \mathcal{F}$ and $M' = M$, for any multiset $M' \in \mathcal{F}$ such that $M' \subseteq M$.*

The support *of $\mathcal{F}$ is defined as the set of all its minimal elements.*

The case where the minimal elements of certain order filters are disjoint is covered by the following general criterion from Kanovich[7].

Theorem 2 (Kanovich[7]). *Let $\mathcal{C}$ and $\mathcal{C}'$ be partition ideals such that the support of the filter $\overline{\mathcal{C}}$ (the filter which is the complement to $\mathcal{C}$) is made of pairwise disjoint multisets, say*

$$C_1, C_2, \ldots, C_i, \ldots,$$

and the support of the filter $\overline{\mathcal{C}'}$ (the filter which is the complement to $\mathcal{C}'$) is made of pairwise disjoint multisets

$$C'_1, C'_2, \ldots, C'_i, \ldots.$$

[1] We say that $M' \subseteq M$ if M' can be formed by removing a number of parts from M. For instance,

$$\{1, 5\} \subseteq \{1, 1, 3, 5, 5, 5\}.$$

Assume that these supports are sorted as lists so that the sequence of integers

$$\|C_1\|,\ \|C_2\|,\ \dots, \|C_i\|, \dots$$

is non-decreasing, and the sequence of integers

$$\|C_1'\|,\ \|C_2'\|,\ \dots, \|C_i'\|, \dots$$

is non-decreasing.

Then $\mathcal{C}$ and $\mathcal{C}'$ are equinumerous if and only if:

$$\|C_i\| = \|C_i'\|, \quad \textit{for all } i.$$

In addition to that, the system Γ consisting of the following multiset rewriting rules:

$$\gamma_1 : C_1 \to C_1', \quad \gamma_2 : C_2 \to C_2', \quad \dots, \quad \gamma_i : C_i \to C_i', \quad \dots$$

provides[2] *a bijection h between $\mathcal{C}'$ and $\mathcal{C}$.*

Moreover, the inverse bijection h^{-1} from $\mathcal{C}$ onto $\mathcal{C}'$ is computed here with the help of the system Γ^{-1} consisting of the 'reverse' rewriting rules:

$$\gamma_1^{-1} : C_1' \to C_1, \quad \gamma_2^{-1} : C_2' \to C_2, \quad \dots, \quad \gamma_i^{-1} : C_i' \to C_i, \quad \dots$$

3 The Main Result

We introduce a two-directional *rewriting* scheme in the following way.

Definition 4. *Let Γ be a set of reduction rules:*

$$\gamma_1 : A_1 \to B_1,\ \gamma_2 : A_2 \to B_2,\ \dots,\ \ \gamma_i : A_i \to B_i,\ \dots .$$

The reversed rules $\gamma_i^{-1} : B_i \to A_i$ form Γ^{-1}:

$$\gamma_1^{-1} : B_1 \to A_1,\ \gamma_2^{-1} : B_2 \to A_2,\ \dots,\ \ \gamma_i^{-1} : B_i \to A_i,\ \dots .$$

*The $\mathcal{A}$-*normal forms *are the Γ-irreducible forms, i.e. the forms that do not contain any of the following list:*

$$\mathcal{A} = A_1, A_2, \dots, A_i, \dots ,$$

*and the $\mathcal{B}$-*normal forms *are the Γ^{-1}-irreducible forms, i.e. the forms that do not contain any of the following list:*

$$\mathcal{B} = B_1, B_2, \dots, B_i, \dots .$$

An intended bijection h between $\mathcal{B}$-normal forms and $\mathcal{A}$-normal forms is defined as follows:

$$h(M) := \widetilde{M}, \textit{ if } \widetilde{M} \textit{ is an } \mathcal{A}\textit{-normal form, and } M \textit{ is } \Gamma\textit{-reducible to } \widetilde{M}. \tag{1}$$

[2] We say "*provides a bijection h between two classes of integer partitions $\mathcal{C}_1$ and $\mathcal{C}_2$*" meaning that, for any n, function h is a bijection between the partitions of n that belong to $\mathcal{C}_1$ and the partitions of the n that belong to $\mathcal{C}_2$.

Definition 5. *We will say that Γ is $\mathcal{B}$-*terminating *if every sequence of Γ-reductions must terminate in a finite number of steps, whenever it started from a $\mathcal{B}$-normal form.*

Proposition 1. *Let both Γ and Γ^{-1} be confluent, and Γ be $\mathcal{B}$-terminating, and Γ^{-1} be $\mathcal{A}$-terminating.*

Then the above h is a well-defined bijection between $\mathcal{B}$-normal forms and $\mathcal{A}$-normal forms.

Remark 1. As a matter of fact, we need a strong "stratified" version of the conclusion of Proposition 1 to supply a bijection between two sets of *partitions* of a fixed n, and to show thereby that the two sets of partitions of the n are equinumerous:

For any fixed n, the above h should be a bijection between $\mathcal{B}$-normal partitions of the n and $\mathcal{A}$-normal partitions of the same n.

The most natural way to guarantee this property is to invoke the following 'balance conditions' - that for all i,

$$\sum_{a\in A_i} a = \sum_{b\in B_i} b \ .$$

In this 'balanced case' only partitions of one and the same n may appear within any sequence of Γ-reductions. ■

3.1 The 'bijective proof' for Zeckendorf's Theorem

Within Zeckendorf's theorem we are dealing with a partition ideal consisting of partitions into distinct Fibonacci numbers with no consecutive Fibonacci parts.

The order filter, the complement to this ideal, is generated by its minimal elements:

$$\{1,1\},\{1,2\},\{2,2\},\{2,3\},\{3,3\},\{3,5\},\{5,5\},\ldots,\{4\},\{6\},\{7\},\ldots,$$

Since some of these minimal elements overlap, general 'non-overlapping' theorems, like Theorem 2, do not work, and we have to treat such an 'overlapping case' in a specific way.

Below we give a 'bijective proof' for Zeckendorf's theorem.

Theorem 3. *Let $\mathcal{C}$ be a partition ideal consisting of partitions with no Fibonacci parts being repeated or consecutive Fibonacci numbers, and $\mathcal{C}'$ be a partition ideal consisting of partitions that have no Fibonacci parts but 1; in both cases non-Fibonacci parts may appear without restriction.*

We can construct an explicit bijection between $\mathcal{C}'$ and $\mathcal{C}$.

Proof Sketch.
Call the $\mathcal{A}$-normal forms the multisets that do not have any of the minimal elements of the order filter $\overline{\mathcal{C}}$ (which is the complement to $\mathcal{C}$):

$$\{1,1\},\{1,2\},\{2,2\},\{2,3\},\{3,3\},\{3,5\},\ldots,\{u_i,u_i\},\{u_i,u_{i+1}\},\ldots$$

As the $\mathcal{B}$-normal forms, we take the multisets that do not have any of the minimal elements of the order filter $\overline{\mathcal{C}'}$ (which is the complement to $\mathcal{C}'$):

$$\{2\}, \{3\}, \{5\}, \{8\}, \{13\}, \ldots, \{u_{i+1}\}, \ldots$$

We introduce the following system of rewriting rules $\Gamma_{(2)}$:

$$\begin{aligned}
&\gamma_1\colon \{1,1\} \to \{2\},\ \gamma_1'\colon \{2,2\} \to \{1,3\},\\
&\gamma_2\colon \{1,2\} \to \{3\},\ \gamma_2'\colon \{3,3\} \to \{1,5\},\\
&\gamma_3\colon \{2,3\} \to \{5\},\ \gamma_3'\colon \{5,5\} \to \{2,8\},\ \ldots\ ,\\
&\gamma_i\colon \{u_i, u_{i+1}\} \to \{u_{i+2}\},\ \gamma_i'\colon \{u_{i+2}, u_{i+2}\} \to \{u_i, u_{i+3}\},\ \ldots\ (i = 1, 2, \ldots)
\end{aligned} \tag{2}$$

Accordingly, the 'reverse' system $\Gamma_{(2)}^{-1}$ will consist of the following reverse rewriting rules:

$$\begin{aligned}
&\gamma_1^{-1}\colon \{2\} \to \{1,1\},\ {\gamma_1'}^{-1}\colon \{1,3\} \to \{2,2\},\\
&\gamma_2^{-1}\colon \{3\} \to \{1,2\},\ {\gamma_2'}^{-1}\colon \{1,5\} \to \{3,3\},\\
&\gamma_3^{-1}\colon \{5\} \to \{2,3\},\ {\gamma_3'}^{-1}\colon \{2,8\} \to \{5,5\},\ \ldots\ ,\\
&\gamma_i^{-1}\colon \{u_{i+2}\} \to \{u_i, u_{i+1}\},\ {\gamma_i'}^{-1}\colon \{u_i, u_{i+3}\} \to \{u_{i+2}, u_{i+2}\},\ \ldots\ (i = 1, 2, \ldots)
\end{aligned} \tag{3}$$

Example 2. We illustrate the proof of Theorem 3 by applying $\Gamma_{(2)}$ to the partition of 7:

$$M = \{1,1,1,1,1,1,1\}.$$

We can proceed, for instance, in the following way:

$$\{1,1,1,1,1,1,1\} \xrightarrow{\gamma_1} \{2,1,1,1,1,1\} \xrightarrow{\gamma_1} \{2,2,1,1,1\} \xrightarrow{\gamma_1}$$
$$\{2,2,2,1\} \xrightarrow{\gamma_2} \{2,2,3\} \xrightarrow{\gamma_1'} \{1,3,3\} \xrightarrow{\gamma_2'} \{1,1,5\} \xrightarrow{\gamma_1} \{2,5\}$$

terminating in $\{2,5\}$, the correct representation of 7 in the Fibonacci numeration system.

On the reverse, starting from a partition of the form $\{2,5\}$, $\Gamma_{(2)}^{-1}$ terminates in the original $\{1,1,1,1,1,1,1\}$.

Lemma 1. *Systems $\Gamma_{(2)}$ and $\Gamma_{(2)}^{-1}$ are confluent and strongly normalizing.*

Proof. Here we suppose that partitions are thought of as non-increasing sequences of integers.

(a) For the forward direction, notice that each of the rules (2) yields a partition that takes a higher position in the lexicographical order.

On the other hand, each of the rules (2) preserves the partition norm $\|M\|$.

Since this 'combined' ordering is well-founded, any sequence of $\Gamma_{(2)}$-reductions must terminate.

On a case-by-case basis, $\Gamma_{(2)}$ is proved to be locally confluent.

These two facts - that $\Gamma_{(2)}$ is strongly normalizing and locally confluent, imply that $\Gamma_{(2)}$ is confluent (cf. Newman's Lemma [3,9]).

(b) For the reverse direction, notice that the reverse rules reduce any partition M lexicographically down to a partition with no Fibonacci parts but 1. Therefore, any sequence of $\Gamma_{(2)}^{-1}$-reductions terminates with a unique normal form. ■

Now we can complete the prove of Theorem 3.

Since our $\mathcal{A}$-normal forms turn out to be exactly the $\Gamma_{(2)}$-irreducible forms, and our $\mathcal{B}$-normal forms turn out to be exactly the $\Gamma_{(2)}^{-1}$-irreducible forms, Proposition 1 provides a bijection h between $\mathcal{C}'$ and $\mathcal{C}$. ■

Remark 2. Theorem 3 is not within reach of known 'bijective' methods [13].

In particular

(a) There is no way to sort these two lists

$$\{1,1\},\{1,2\},\{2,2\},\{2,3\},\{3,3\},\{3,5\},\ldots,\{u_i,u_i\},\{u_i,u_{i+1}\},\ldots$$

and

$$\{2\},\{3\},\{5\},\{8\},\{13\},\ldots,\{u_{i+1}\},\ldots$$

to be suited for Remmel's algorithm[11].

(b) Similarly, the sieve method from Wilf [13] cannot be applied to Theorem 3.

3.2 The 'bijective proofs' for New Partition Identities Related to Generalized Fibonacci Numbers

The rewriting construction introduced in the proof of Theorem 3 can be easily generalized for Fibonacci-like numbers produced by the following scheme:

Definition 6. *Given positive integers a and b, let define* F-numbers v_i *as follows:*

$$v_1 = a,\ v_2 = b,\ v_{i+2} = v_i + v_{i+1} \quad (i = 1,2,3,\ldots) \tag{4}$$

For $a = b = 1$, we have Fibonacci numbers: 1, 1, 2, 3, 5, ...
For $a = 2,\ b = 1$, we get Lucas numbers: 2, 1, 3, 4, 7, ...

Our goal is to prove the following:

Theorem 4. *For any n, the number of partitions of n into F-numbers whose parts are neither repeated F-numbers with indices greater than* 2 *nor consecutive F-numbers equals the number of partitions of n into a's and b's.*

Proof. Similar to the bijective proof of Theorem 3, we introduce the following system $\Gamma_{(5)}$ consisting of the rewriting rules:

$$\begin{aligned}
&\gamma_1\colon \{v_1, v_2\} \to \{v_3\},\ \gamma_1'\colon \{v_3, v_3\} \to \{v_1, v_4\},\\
&\gamma_2\colon \{v_2, v_3\} \to \{v_4\},\ \gamma_2'\colon \{v_4, v_4\} \to \{v_2, v_5\},\\
&\gamma_3\colon \{v_3, v_4\} \to \{v_5\},\ \gamma_3'\colon \{v_5, v_5\} \to \{v_3, v_6\},\ \dots\ ,\\
&\gamma_i\colon \{v_i, v_{i+1}\} \to \{v_{i+2}\},\ \gamma_i'\colon \{v_{i+2}, v_{i+2}\} \to \{v_i, v_{i+3}\},\ \dots\ (i = 1, 2, 3, \dots)
\end{aligned} \tag{5}$$

Accordingly, the 'reverse' system $\Gamma_{(5)}^{-1}$ will consist of the reverse rewriting rules:

$$\begin{aligned}
&\gamma_1^{-1}\colon \{v_3\} \to \{v_1, v_2\},\ \gamma_1'^{-1}\colon \{v_1, v_4\} \to \{v_3, v_3\},\\
&\gamma_2^{-1}\colon \{v_4\} \to \{v_2, v_3\},\ \gamma_2'^{-1}\colon \{v_2, v_5\} \to \{v_4, v_4\},\\
&\gamma_3^{-1}\colon \{v_5\} \to \{v_3, v_4\},\ \gamma_3'^{-1}\colon \{v_3, v_6\} \to \{v_5, v_5\},\ \dots\ ,\\
&\gamma_i^{-1}\colon \{v_{i+2}\} \to \{v_i, v_{i+1}\},\ \gamma_i'^{-1}\colon \{v_i, v_{i+3}\} \to \{v_{i+2}, v_{i+2}\},\ \dots\ (i = 1, 2, \dots)
\end{aligned} \tag{6}$$

By the same token of Lemma 1, we prove that both systems $\Gamma_{(5)}$ and $\Gamma_{(5)}^{-1}$ are confluent and strongly normalizing.

The effect is that Proposition 1 provides a bijection h between the partition ideals in question. ■

Theorem 4 yields a series of new partition identities.

For instance, by taking $a = 2$, $b = 1$, and $a = 1$, $b = 2$, respectively, Theorem 4 yields the following Corollary 1:

Corollary 1. *The following classes of partitions are pairwise equinumerous:*

(a) *The class of partitions into ones and twos.*
(b) *The class of partitions into Lucas parts whose parts are neither repeated Lucas numbers greater than* 2 *nor consecutive Lucas numbers.*
(c) *The class of partitions into Fibonacci parts whose parts are neither repeated Fibonacci numbers greater than* 2 *nor distinct consecutive Fibonacci numbers.*

Proof

(i) Take $a = 2$, $b = 1$, to cover the case of partitions into Lucas parts.
(ii) Take $a = 1$, $b = 2$, to cover the case of partitions into Fibonacci parts. ■

4 Concluding Remarks

The *novelty* of our formal systems approach to the combinatorics is in the use of rewriting techniques (*two-directional* in the sense that forward and backward applications of rewrite rules head respectively for two different normal forms) for the purpose of establishing *explicit relevant bijections* between combinatorial objects of two different types (represented by these normal forms).

As compared to Kanovich[7,8], here we have discussed the challenges of the 'overlapping' multiset rewriting systems.

As a possible step towards the much loftier goal of classifying all equinumerous partition ideals, we have shown the 'bijective proofs' of a new series of partition identities related to Fibonacci and Lucas numbers.

Acknowledgments

I owe special thanks to George Andrews and Herb Wilf for their inspiring introduction to the world of integer partitions and for valuable discussions on how techniques from the rewriting world can be useful to sort out specific mathematical problems studied completely independently in the world of combinatorics.

References

1. G.E.Andrews. The Theory of Partitions, Cambridge University Press, 1998.
2. F.Baader and T.Nipkow. Term Rewriting and All That. Cambridge University Press, 1998.
3. N.Dershowitz and J.-P.Jouannaud. Rewrite systems. In J.van Leeuwen, ed., Handbook of Theoretical Computer Science, volume B, pp 243–320. Elsevier, 1990.
4. A.M.Garsia and S.C.Milne, Method for constructing bijections for classical partition identities. Proc. Nat. Acad. Sci. U.S.A. **78** (1981), no. 4(1), 2026–2028.
5. Basil Gordon, Sieve-equivalence and explicit bijections, J. Combin. Theory Ser. A **34** (1983), no. 1, 90–93.
6. K.M. O'Hara, Bijections for partition identities. J. Combin. Theory Ser. A **49** (1988), no. 1, 13–25.
7. M.Kanovich. Finding direct partition bijections by two-directional rewriting techniques. *Discrete Mathematics*, Volume 285, Issues 1-3, 6 August 2004, pp. 151-166. The preliminary version: Bijections between partitions by two-directional rewriting techniques. In: *Proc. Annual Conference of the European Association for Computer Science Logic, CSL'02,* September 22-25, 2002 Edinburgh, Scotland, Lecture Notes in Computer Science 2471, J.Bradfield (Ed.), pp. 44–58, 2002.
8. M.Kanovich. The two-way rewriting in action: Removing the mystery of Euler-Glaisher's map. *Discrete Mathematics* (2006), doi:10.1016/j.disc.2006.10.005, 27p.
9. J.W.Klop. Term rewriting systems. In: S.Abramsky, D.M.Gabbay, and T.S.E.Maibaum, ed., Handbook of Logic in Computer Science, volume 2, pp 1–116. Oxford University Press, New York, 1992.
10. D.Knuth. The Art of Computer Programming, Volume 2: Seminumerical Algorithms, Addison-Wesley, 1997.
11. J.B.Remmel, Bijective proofs of some classical partition identities, J. Combin. Theory Ser. A **33** (1982), 273–286.
12. H.S.Wilf, Sieve equivalence in generalized partition theory, J. Combin. Theory Ser. A **34** (1983), 80–89.
13. H.S.Wilf, Lectures on Integer Partitions, University of Victoria, Victoria, B.C., Canada, 2000, (`<http://cis.upenn.edu/~wilf>`).
14. E. Zeckendorf, A Generalized Fibonacci Numeration, Fibonacci Quarterly, vol 10 (1972), 365–372.

On Complexity of Ehrenfeucht-Fraïssé Games

Bakhadyr Khoussainov[1] and Jiamou Liu[2]

[1] Department of Computer Science
University of Auckland
New Zealand
[2] Department of Computer Science
University of Auckland
New Zealand

Abstract. In this paper we initiate the study of Ehrenfeucht-Fraïssé games for some standard finite structures. Examples of such standard structures are equivalence relations, trees, unary relation structures, Boolean algebras, and some of their natural expansions. The paper concerns the following question that we call Ehrenfeucht-Fraïssé problem. Given $n \in \omega$ as a parameter, two relational structures $\mathcal{A}$ and $\mathcal{B}$ from one of the classes of structures mentioned above, how efficient is it to decide if Duplicator wins the n-round EF game $G_n(\mathcal{A}, \mathcal{B})$? We provide algorithms for solving the Ehrenfeucht-Fraïssé problem for the mentioned classes of structures. The running times of all the algorithms are bounded by constants. We obtain the values of these constants as functions of n.

1 Introduction

Ehrenfeucht-Fraïssé (EF) games constitute an important tool in both finite and infinite model theory. For example, in infinite model theory these games can be used to prove Scott Isomorphism Theorem showing that all countable structures are described (up to isomorphism) in $L_{\omega_1,\omega}$-logic. In finite model theory these games and their different versions are used for establishing expressibility results in the first order logic and its extensions. These results can be found in standard books in finite and infinite model theory (e.g. [6], [11]) or relatively recent papers (e.g. [2], [12]). In this paper all EF games are considered on finite structures.

Despite significant use of EF games in finite and infinite model theory there has not been, with some exceptions, much work in addressing efficiency of these games. M. Grohe studied EF games with fixed number of pebbles and showed that the problem of deciding the winner is complete for PTIME [5]. E. Pezzoili showed that deciding the winner of EF games is PSPACE-complete [14]. In [9] P. Kolaitis and J. Panttaja prove that the following problem is EXPTIME-complete: given a natural number k and structures $\mathcal{A}$ and $\mathcal{B}$, does Duplicator win the k pebble existential EF game on $\mathcal{A}$ and $\mathcal{B}$? In [1] sufficient conditions are provided for Duplicator to win EF games. These conditions are then used to prove some inexpressibility results, e.g reachability in undirected graphs is not in monadic NP. These results suggest that developing tools and algorithms for finding winners of EF are of interest. We also point out that there has recently been an interest in EF games to collapse results in database theory [16]. In addition, we think that algorithms that solve EF games can be used in data matching and data transformation problems in databases.

In this paper we initiate the study of EF games for some standard finite structures. Examples of such standard structures are equivalence relations, trees, unary relation structures, Boolean algebras, and some of their natural expansions. The paper concerns the following question that we call the Ehrenfeucht-Fraïssé problem. Given $n \in \omega$ as a parameter, two relational structures $\mathcal{A}$ and $\mathcal{B}$, how efficient is it to decide if Duplicator wins the n-round EF game $G_n(\mathcal{A}, \mathcal{B})$? We provide algorithms for solving the Ehrenfeucht-Fraïssé problem for the structures mentioned above. The running times of all the algorithms are bounded by constants. We obtain the values of these constants as functions of n.

By a structure we always mean a finite relational structure over a language without functional symbols. Let $\mathcal{A}$ and $\mathcal{B}$ be structures and $n \in \omega$. EF game, denoted by $G_n(\mathcal{A}, \mathcal{B})$, on these two structures is played as follows. There are two players, Duplicator and Spoiler, both provided with $\mathcal{A}$ and $\mathcal{B}$. The game consists of n rounds. Informally, Duplicator's goal is to show that these two structures are similar, while Spoiler needs to show the opposite. At round i, Spoiler selects structure $\mathcal{A}$ or $\mathcal{B}$, and then takes an element from the selected structure. Duplicator responds by selecting element from the other structure. Say, the players have produced the following play consisting of pairs of elements $(a_1, b_1), \ldots, (a_n, b_n)$, where $a_i \in A$ and $b_i \in B$ for $i = 1, \ldots, n$. Note that if Spoiler selected a_i (or b_i) then Duplicator selected b_i (or a_i, respectively). Duplicator wins the play if the mapping $a_i \to b_i$, $i = 1, \ldots, n$, extended by mapping the values of constant symbols $c^{\mathcal{A}}$ to $c^{\mathcal{B}}$, is a partial isomorphism between $\mathcal{A}$ and $\mathcal{B}$. It is clear that if $\mathcal{A}$ and $\mathcal{B}$ are isomorphic then Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ no matter what n is. The opposite is not always true. However, for large n if Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ then $\mathcal{A}$ and $\mathcal{B}$ are isomorphic. Thus, solving the EF problem can be thought as an approximation to the isomorphism problem.

One can do the following rough estimates for finding the winner of the game $\mathcal{G}_n(\mathcal{A}, \mathcal{B})$. There are finitely many, up to logical equivalence, formulas ϕ_1, …, ϕ_k of quantifier rank n (see for example [11]). It is well known that Duplicator wins $G_n(\mathcal{A}, \mathcal{B})$ if and only if for all ϕ_i (with $i = 1, \ldots, k$) the structure $\mathcal{A}$ satisfies ϕ_i if and only if $\mathcal{B}$ satisfies ϕ_i [11]. Thus, the question if Duplicator wins $G_n(\mathcal{A}, \mathcal{B})$ can be solved in polynomial time. However, there are two important issues here. The first issue concerns the number k that depends on n; k is approximately bounded by the n-repeated exponentiations of 2. The second issue concerns the degree of the polynomial for the running time that is bounded by n. Thus, the questions arise as to for which standard structures the value of k is feasible as a function of n, and whether the degree of the polynomial for the running time can be pushed down. As an example consider the class of linear orders. It is well-known that Duplicator wins $G_n(\mathcal{A}, \mathcal{B})$, where $\mathcal{A}$ and $\mathcal{B}$ are linear orders, if and only if either $|A| = |B|$ or both $|A| > 2^n$ and $|B| > 2^n$ (e.g. [11]). In this example, the number k, roughly, equals to 2^n. The degree of polynomial for the running time is 0. Thus, when n is fixed the winner of the game can be found in constant time, and the constant that bounds the time is 2^n.

A brief overview of this paper is as follows. The next section gives an elementary solution to EF games in the case when the language contains unary predicates only. The third, fourth and fifth sections are quite technical and devoted to solving EF games for equivalence structures and some of their extensions. Equivalence structures are natural models of university or large company databases. For example, in a university database there could be the *SameFaculty* and the *SameDepartment* relations. The first relation stores all tuples (x, y) such that x and y belong to the same faculty; similarly, the second relation stores all tuples (u, v) such that u and v are in the same department. These relations are equivalence relations. Moreover, the set-theoretic connection between these relations is that the relation *SameDepartment* is a subset of the *SameFaculty* relation. We call such structures embedded equivalence relation structures. Section 6 reduces the question of deciding EF games for trees of a given height to solving the EP games for embedded equivalence structures introduced in the previous sections. Finally, the main structures in the last section are Boolean algebras with distinguished ideals.

Each of these sections provides an algorithm that decides EF games $G_n(\mathcal{A}, \mathcal{B})$, where $\mathcal{A}$ and $\mathcal{B}$ are structures considered in the section. These algorithms run in *constant times* with n being a parameter. We also bound the value of the constants as a function of n. Clearly, the constants obtained depend on the representations of the structures. In each case, it will be clear from the content how we represent our structures. As an example we state two results of Sections 4 and 5. Section 4 is devoted to structures of the type $(A; E, P_1, ..., P_s)$, where E is an equivalence relation on A and $P_1, ..., P_s$ are unary predicates. We call these structures equivalence structures with s colors. The main result of Section 4 is the following:

Theorem 1. Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on equivalence structures with s colors. The constant that bounds the running time is n^{2^s+1}.

Section 5 is devoted to the structures of type $(A; E_1, \dots, E_h)$, where each E_i is an equivalence relation on A and $E_1 \subseteq E_2 \subseteq \dots \subseteq E_h$. These structures are called embedded equivalence structures of height h. The main result of Section 5 is:

Theorem 2. Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on embedded equivalence structures of height h. The constant that bounds the running time is $(n+1)^{\cdot^{\cdot^{\cdot (n+1)^n}}}$ where the tower has height h.

2 Simple Example: Structures with Unary Predicates

This is an elementary section that gives a full solution for EF games in the case when the language contains unary predicates only. Here is the main result of this section.

Theorem 3. *Fix the language $L = (P_1, \dots, P_s)$, where each P_i is a unary predicate symbol. Let $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on structures $\mathcal{A}$ and $\mathcal{B}$ of the language. The constant that bounds the running time is $2^s \cdot n$.*

Proof. Let $\mathcal{A} = (A; P_1, P_2, ..., P_s)$ and $\mathcal{B} = (B; P_1, P_2, ..., P_s)$ be structures of the language given. For structure $\mathcal{A} = (A; P_1, P_2, ..., P_s)$, we set $P_{s+1} = \bigcap_i \neg P_i$.

Lemma 1. *Suppose $P_1, P_2, ..., P_s$ are pairwise disjoint. Then Duplicator wins $G_n(\mathcal{A}, \mathcal{B})$ if and only if for all $1 \le i \le s+1$ if $|P_i^A| < n$ or $|P_i^B| < n$ then $|P_i^A| = |P_i^B|$. In particular, when Duplicator wins it is the case that for all $1 \le i \le s+1$, $|P_i^A| \ge n$ if and only if $|P_i^B| \ge n$.*

To prove the lemma suppose that there is $1 \le i \le k+1$ such that $|P_i^A| < n$ but $|P_i^A| \ne |P_i^B|$. Assume $|P_i^B| < |P_i^A|$. Then Spoiler selects $|P_i^A|$ elements from P_i^A. This strategy is clearly a winning strategy for Spoiler. For the other direction, assume that hypothesis of the lemma holds. Duplicator has a winning strategy as follows. At round k, assume that the players have produced the k-round play $(a_1, b_1), ..., (a_k, b_k)$ such that for each $1 \le i \le k$, $a_i \in A, b_i \in B$. If Spoiler selects $a_{k+1} \in A$, then Duplicator responds by selecting $b_{k+1} \in B$ as follows: If $a_{k+1} = a_i$ for some i then $b_{k+1} = b_i$. Otherwise if $a_{k+1} \in P_j^A$ for some $1 \le j \le k$, then $b_{k+1} \in P_j^B$ so that $b_{k+1} \notin \{b_1, \ldots, b_k\}$. The case when Spoiler selects an element from B is treated similarly. The strategy is clearly winning. □

Now assume that for a structure $\mathcal{A}$, the unary predicates $P_1, P_2, ..., P_s$ are not necessarily pairwise disjoint. For each element $x \in A$, define the **characteristic** of x, $ch(x)$, as a binary sequence $(t_1, t_2, ..., t_s)$ such that for each $1 \le i \le s$, $t_i \in \{0, 1\}$ if $x \in P_i$ and $t_i = 0$ otherwise. There are 2^s pairwise distinct characteristics, and we order them in lexicographic order: $ch_1, ..., ch_{2^s}$. Construct the structure $\mathcal{A}' = (A; Q_1, ..., Q_{2^s})$ such that for all $1 \le i \le 2^s$, $Q_i = \{x \in A \mid ch(x) = ch_i\}$. The following is now an easy lemma.

Lemma 2. *Duplicator wins $G_n(\mathcal{A}, \mathcal{B})$ if and only if Duplicator wins $G_n(\mathcal{A}', \mathcal{B}')$.* □

We now represent $\mathcal{A}$ and $\mathcal{B}$ by 2^s lists, and the i^{th} list lists all elements with characteristic ch_i. To solve the game $G_n(\mathcal{A}', \mathcal{B}')$, the algorithm checks the conditions in Lemma 1 by reading the lists. The process takes time bounded by $2^s \cdot n$ as required. □

3 Equivalence Structures

An **equivalence structure** is a structure $\mathcal{A}$ of the type $(A; E)$ where E is an equivalence relation on A. We list all the equivalence classes of $\mathcal{A}$ as $A_1, ..., A_k$ such that $|A_i| \le |A_{i+1}|$ for all $1 \le i < k$. Let $q_{\mathcal{A}}$ be the number of equivalent classes in $\mathcal{A}$; for each $t < n$, let q_t^A be the number of equivalence classes in $\mathcal{A}$ with size t. Finally, let $q_{\ge r}^{\mathcal{A}}$ be the number of equivalence classes in $\mathcal{A}$ of size at least r. For an equivalence structure $\mathcal{B}$ we have similar notations as $B_1, B_2, \ldots$ to denote its equivalence classes, and the associated numbers $q_{\mathcal{B}}$, $q_t^{\mathcal{B}}$, and $q_{\ge r}^{\mathcal{B}}$.

Lemma 3. *If Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on equivalence structures $\mathcal{A}$ and $\mathcal{B}$, then the following must be true:*

1. *If $q_{\mathcal{A}} < n$ or $q_{\mathcal{B}} < n$ then $q_{\mathcal{A}} = q_{\mathcal{B}}$; and*
2. *$q_{\mathcal{A}} \ge n$ if and only if $q_{\mathcal{B}} \ge n$.* □

In our analysis below, by the above lemma, we always assume that $q_{\mathcal{A}} = q_{\mathcal{B}}$ or $q_{\mathcal{A}} \ge n$ if and only if $q_{\mathcal{B}} \ge n$. We need the following notation for the next lemma and definition. For $t \le n$, let $q^t = min\{q_{\ge t}^{\mathcal{A}}, q_{\ge t}^{\mathcal{B}}\}$. Let $\mathcal{A}_t$ and $\mathcal{B}_t$ be equivalence structures obtained by taking out exactly q^t equivalence classes of size $\ge t$ from $\mathcal{A}$ and $\mathcal{B}$ respectively. We also set $n - q^t$ to be 0 in case $q^t \ge n$; and otherwise $n - q^t$ has its natural meaning.

Lemma 4. *1. Assume that there is a $t < n$ such that $q_t^{\mathcal{A}} \neq q_t^{\mathcal{B}}$ and $n - q^t > t$. Then Spoiler wins the game $G_n(\mathcal{A}, \mathcal{B})$.*

2. Assume that there is a $t \leq n$ such that $n - q^t > 0$ and one of the structures $\mathcal{A}_t$ or $\mathcal{B}_t$ has an equivalence class of size $\geq n - q^t$ and the other structure does not. Then Spoiler wins the game $G_n(\mathcal{A}, \mathcal{B})$.

Proof. We prove the first part of the lemma. The second part is proved similarly. Assume, without loss of generality, that $q_t^{\mathcal{A}} > q_t^{\mathcal{B}}$ and $n - q_t^{\mathcal{B}} > t$. Spoiler's strategy is the following. First, select elements $a_1, \dots, a_{q_t^{\mathcal{B}}}$ from distinct equivalence classes of size t in $\mathcal{A}$. Next, select t distinct elements in the equivalence class of size t in $\mathcal{A}$. This leads Spoiler to win. □

Definition 1. *1. We say that $G_n(\mathcal{A}, B)$* **has small disparity** *if there is a $t < n$ such that either $q_t^{\mathcal{A}} \neq q_t^{\mathcal{B}}$ and $n - q^t > t$.*

2. We say that $G_n(\mathcal{A}, B)$ **has large disparity** *if there exists a $t \leq n$ such that $n - q^t > 0$ and one of the structures $\mathcal{A}_t$ or $\mathcal{B}_t$ has an equivalence class of size $\geq n - q^t$ and the other structure does not.*

Lemma 5. *Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ if and only if the game $G_n(\mathcal{A}, \mathcal{B})$ has neither small nor large disparity.*

Proof. The previous lemma proves one direction. For the other, we assume that neither small nor large disparity occurs in the game. We describe a winning strategy for Duplicator.

Let us a assume that the players have produced a k-round play $(a_1, b_1), (a_2, b_2), \ldots, (a_k, b_k)$. In case $k = 0$, we are at the start of the game $G_n(\mathcal{A}, \mathcal{B})$. Our inductive assumptions on this k-round play are the following:

1. $E(a_i, a_j)$ is true in $\mathcal{A}$ if and only if $E(b_i, b_j)$ is true in $\mathcal{B}$, and the map $a_i \to b_i$ is one-to-one.
2. For all a_i, $\|[a_i]\| \geq n - i$ if and only if $\|[b_i]\| \geq n - i$, where $[x]$ denotes the equivalence class of x.
3. For a_i if $\|[a_i]\| < n - i$ then $\|[a_i]\| = \|[b_i]\|$.
4. Let $\mathcal{A}'$ and $\mathcal{B}'$ be the equivalence structures obtained by removing the equivalence classes $[a_1]$, ..., $[a_k]$ from $\mathcal{A}$ and the equivalence classes $[b_1]$, ..., $[b_k]$ from $\mathcal{B}$, respectively. We assume that $\mathcal{A}'$ and $\mathcal{B}'$ satisfy the following conditions:
   (a) In game $G_{n-k}(\mathcal{A}', \mathcal{B}')$ no small disparity occurs.
   (b) In game $G_{n-k}(\mathcal{A}', \mathcal{B}')$ no large disparity occurs.

Assume that Spoiler selects $a_{k+1} \in A$. Duplicator responds by choosing b_{k+1} as follows. If $a_{k+1} = a_i$ then $b_{k+1} = b_i$. Otherwise, if $E(a_i, a_{k+1})$ is true in $\mathcal{A}$ then Duplicator chooses a new b_{k+1} such that $E(b_i, b_{k+1})$ is true in $\mathcal{B}$. Assume a_{k+1} is not equivalent to any of the elements $a_1, \ldots, a_k$. If $\|[a_{k+1}]\| \geq n - k$ then Duplicator chooses b_{k+1} such that b_{k+1} is not equivalent to any of the elements $b_1, \ldots, b_k$ and $\|[b_{k+1}]\| \geq n - k$. Duplicator can select such an element as otherwise large disparity would occur in the game. If $\|[a_{k+1}]\| < n - k$ then Duplicator chooses b_{k+1} such that $\|[b_{k+1}]\| = \|[a_{k+1}]\|$ and b_{k+1} is not equivalent to any of the elements $b_1, \ldots, b_k$. The case when Spoiler selects an element from B is treated similarly.

Now we show that the $(k+1)$-round play $(a_1, b_1), (a_2, b_2), ..., (a_k, b_k), (a_{k+1}, b_{k+1})$ satisfies the inductive assumptions. The inductive assumptions (1), (2), and (3) can easily be checked to be preserved. To show that the assumption (4) is preserved, consider the equivalence structures $\mathcal{A}''$ and $\mathcal{B}''$ obtained by removing the equivalence classes $[a_1]$, ..., $[a_k]$, $[a_{k+1}]$ from $\mathcal{A}$ and the equivalence classes $[b_1]$, ..., $[b_k]$, $[b_{k+1}]$ from $\mathcal{B}$, respectively. In game $G_{n-k-1}(\mathcal{A}'', \mathcal{B}'')$ small disparity does not occur as otherwise the game $G_{n-k}(\mathcal{A}', \mathcal{B}')$ would have small disparity. Thus, assumption $(4a)$ is also preserved. Similarly, if $G_{n-k-1}(\mathcal{A}'', \mathcal{B}'')$ had large disparity then the game $G_{n-k}(\mathcal{A}', \mathcal{B}')$ would also have large disparity contradicting the inductive assumption. Hence, the strategy described must be a winning strategy due to the fact that Duplicator preserves the inductive assumption (1) at each round. □

For the next theorem, we represent each equivalence structure $\mathcal{A}$ and $\mathcal{B}$ in two lists. For example, the first list for the structure $\mathcal{A}$ lists all equivalence classes of $\mathcal{A}$ in increasing order; the second list is $q^{\mathcal{A}}$, $q_1^{\mathcal{A}}$, $q_{\geq 1}^{\mathcal{A}}$, $q_2^{\mathcal{A}}$, $q_{\geq 2}^{\mathcal{A}}$, ... The lemmas above give us the following:

Theorem 4. *Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on equivalence structures $\mathcal{A} = (A; E)$ and $\mathcal{B} = (B; E)$. The constant that bounds the running time is n.* □

We can extend the above theorem by defining the following structures:

Definition 2. *A* **homogeneous equivalence structure** *is $(A; E, P_1, \ldots, P_s)$ such that*

- *$(A; E)$ is an equivalence structure; and*
- *Each P_i is a homogeneous unary relation on A meaning that for all $x, y \in A$ if $E(x, y)$ then $x \in P_i$ if and only if $y \in P_i$.*

For a homogeneous equivalence structure $\mathcal{A}$, define the characteristic $ch(x)$ of an element $x \in A$ as in Section 2. Represent $\mathcal{A}$ as a disjoint union of equivalence structures $\mathcal{A}_1$, ..., $\mathcal{A}_{2^s}$, where A_ϵ consists of elements with characteristic ϵ. The above theorem is thus extended to:

Theorem 5. *There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on homogeneous equivalence structures $\mathcal{A}$ and $\mathcal{B}$. The constant that bounds the running time is $2^s \cdot n$.* □

4 Equivalence Structures with Colors

In this section structures $\mathcal{A}$ are of the form $(A; E, P_1, ..., P_s)$, where E is an equivalence relation on A and $P_1, ..., P_s$ are unary predicates on A. We call these **equivalence structures with** s **colors**. We start with the case when $s = 1$. The case for $s > 2$ will be explained later.

Let $\mathcal{A} = (A; E, P)$ be a equivalence structure with one color. Say $x \in A$ is **colored** if $P(x)$ is true; otherwise x is **non-colored**. An equivalence class X has **type** $tp(X) = (i, j)$, if the number of colored elements of X is i, non-colored elements is j; thus, $i + j = |X|$.

Definition 3. *Given two types (i, j) and (i', j') respectively. We say that (i, j) is* **colored n-equivalent** *to (i', j'), denoted by $(i, j) \equiv_n^C (i', j')$, if the following holds.*

1. *If $i < n$ then $i' = i$; otherwise $i' \geq n$.*
2. *If $j < n - 1$ then $j' = j$; otherwise $j' \geq n - 1$.*

We say that (i, j) is **non-colored n-equivalent** *to (i', j'), denoted by $(i, j) \equiv_n^N (i', j')$, if the following holds.*

1. *If $j < n$ then $j' = j$; otherwise $j' \geq n$.*
2. *If $i < n - 1$ then $i' = i$; otherwise $i' \geq n - 1$.*

For $X \subseteq A$, we use $(X; E \restriction X, P \restriction X)$ to denote the equivalence structure obtained by restricting E and P on X. Note that given two equivalence classes X and Y of types (i, j) and (i', j') respectively, if (i, j) is colored (non-colored) n-equivalent to (i', j'), then Duplicator wins the n-round game played on structures $(X; E \restriction X, P \restriction X)$ and $(Y, E \restriction Y, P \restriction Y)$, given the fact that Spoiler chooses a colored (non-colored) element in the first round.

Lemma 6. *If either $(i', j') \equiv_n^C (i, j)$ or $(i', j') \equiv_n^N (i, j)$, then $(i', j') \equiv_{n-1}^C (i, j)$ and $(i', j') \equiv_{n-1}^N (i, j)$.* □

For an equivalence structure $\mathcal{A} = (A; E, P)$, we need the following notations:

- For type (i, j) and $k \geq 1$, Set $C_{(i,j),k}^{\mathcal{A}}$ be the set $\{X \mid X$ is an equivalence class of $\mathcal{A}$ and $tp(X) \equiv_k^C (i, j)\}$. Set $N_{(i,j),k}^{\mathcal{A}}$ be the set $\{X \mid X$ is an equivalence class of $\mathcal{A}$ and $tp(X) \equiv_k^N (i, j)\}$
- For type (i, j) and $k \geq 1$, set $q_{(i,j),k}^{\mathcal{A},C} = |C_{(i,j),k}^{\mathcal{A}}|$, and set $q_{(i,j),k}^{\mathcal{A},N} = |N_{(i,j),k}^{\mathcal{A}}|$.
- For $\mathcal{A}$ and $\mathcal{B}$, set $q_{(i,j),k}^{C} = min\{q_{(i,j),k}^{\mathcal{A},C}, q_{(i,j),k}^{\mathcal{B},C}\}$ and $q_{(i,j),k}^{N} = min\{q_{(i,j),k}^{\mathcal{A},N}, q_{(i,j),k}^{\mathcal{B},N}\}$
- Set $\mathcal{A}^C((i, j), k)$ be the structure obtained from $\mathcal{A}$ by removing $q_{(i,j),k}^{C}$ equivalence classes in $C_{(i,j),k}^{\mathcal{A}}$.
- Set $\mathcal{A}^N((i, j), k)$ be the structure obtained from $\mathcal{A}$ by removing $q_{(i,j),k}^{N}$ equivalence classes in $N_{(i,j),k}^{\mathcal{A}}$.

Observe the following. If Spoiler selects a colored element from an equivalence class X in $\mathcal{A}$, and Duplicator responds by selecting a colored element from an equivalence class Y such that $tp(Y) \equiv_n^C tp(X)$, there is no point for Spoiler to play inside X because this will guarantee a win for Duplicator. Conversely, suppose Spoiler selects a colored element from an equivalence class X in $\mathcal{A}$, and there is no equivalence class in $\mathcal{B}$ whose type is colored n-equivalent to $tp(X)$. Then Spoiler has a winning strategy by playing inside X and Y.

Definition 4. *Consider the game $G_n(\mathcal{A}, \mathcal{B})$ played on equivalence structures with one color. We say that a* **colored disparity** *occurs if there exists a type (i, j) and $n > k \geq 0$ such that the following holds:*

1. *$k = q^C_{(i,j),n-k}$.*
2. *In one of $\mathcal{A}^C((i, j), n-k)$ and $\mathcal{B}^C((i, j), n-k)$, there is an equivalence class whose type is colored $(n-k)$-equivalent to (i, j), and no such equivalence class exists in the other structure.*

We say that a **non-colored disparity** *occurs if there exists a type (i, j) and $n > k \geq 0$ such that the following holds:*

1. *$k = q^N_{(i,j),n-k}$.*
2. *In one of $\mathcal{A}^N((i, j), n-k)$ and $\mathcal{B}^N((i, j), n-k)$, there is an equivalence class whose type is non-colored $(n-k)$-equivalent to (i, j), and no such equivalence class exists in the other structure.*

Lemma 7. *Suppose $\mathcal{A}$ and $\mathcal{B}$ are two equivalence structures with one color. Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ if and only if neither colored disparity nor non-colored disparity occurs in the game.*

Proof. If either colored or non-colored disparity occurs in the game, then it is not too hard to see that Spoiler wins the game. Suppose that neither colored disparity nor non-colored disparity occurs in the game $G_n(\mathcal{A}, \mathcal{B})$, we describe a strategy for Duplicator. Let us assume that the players have produced a k-round play $(a_1, b_1), (a_2, b_2), ..., (a_k, b_k)$. Let (i_l, j_l) and (i'_l, j'_l) be the types of a_l and b_l,respectively with $1 \leq l \leq k$. Our inductive assumptions on this k-round play are the following:

1. For any $1 \leq l \leq k$, a_l is a colored element if and only if b_l is a colored element.
2. For any $1 \leq l, m \leq k$, $E(a_l, a_m)$ if and only if $E(b_l, b_m)$.
3. For any $1 \leq l \leq k$, $(i_l, j_l) \equiv^C_{n-l} (i'_l, j'_l)$ and $(i_l, j_l) \equiv^N_{n-l} (i'_l, j'_l)$.
4. Let $\mathcal{A}'$ and $\mathcal{B}'$ be the equivalence structures obtained by removing equivalence classes $[a_1], ..., [a_k]$ from $\mathcal{A}$ and $[b_1], ..., [b_k]$ from $\mathcal{B}$, respectively. We assume in game G_{n-k} neither colored disparity nor non-colored disparity occurs.

Assume that Spoiler selects an element $a_{k+1} \in A$. Duplicator responds to this move by choosing b_{k+1} as follows. If $a_{k+1} = a_l$ then $b_{k+1} = b_l$. Otherwise, if $E(a_{k+1}, a_l)$ is true in $\mathcal{A}$, then Duplicator chooses a new b_{k+1} such that $E(b_{k+1}, b_l)$ and a_{k+1} is a colored element if and only if b_{k+1} is a colored element. By (3) of the inductive assumption, Duplicator can always select such an element b_{k+1}.

Assume a_{k+1} is not equivalent to any of the element $a_1, ..., a_k$. Let X be the equivalence class of a_{k+1} in $\mathcal{A}$. If a_{k+1} is a colored element, then Duplicator chooses a colored element b_{k+1} from an equivalence class Y of $\mathcal{B}$ such that $tp(X) \equiv^C_{n-k} tp(Y)$. If a_{k+1} is a non-colored element, then Duplicator chooses a non-colored b_{k+1} from an equivalence class Y of $\mathcal{B}$ such that $tp(X) \equiv^N_{n-k} tp(Y)$. Note that such an equivalence class Y must exist in $\mathcal{B}$ as otherwise either colored or non-colored disparity would occur in $G_{n-k}(\mathcal{A}', \mathcal{B}')$ as witnessed by $tp(X)$ and 0. The case when Spoiler selects an element from B is treated in a similar manner.

On the play $(a_1, b_1), ..., (a_k, b_k), (a_{k+1}, b_{k+1})$, the inductive assumption (1) and (2) can be easily checked to hold. To prove that inductive assumption (3) holds, let (i_{k+1}, j_{k+1}) and (i'_{k+1}, j'_{k+1}) be the type of $[a_{k+1}]$ and $[b_{k+1}]$ respectively. The strategy ensures one of $(i_{k+1}, j_{k+1}) \equiv^C_{n-k} (i'_{k+1}, j'_{k+1})$ and $(i_{k+1}, j_{k+1}) \equiv^N_{n-k} (i'_{k+1}, j'_{k+1})$ is true, and by Lemma 6, $(i_{k+1}, j_{k+1}) \equiv^C_{n-k-1} (i'_{k+1}, j'_{k+1})$ and $(i_{k+1}, j_{k+1}) \equiv^N_{n-k-1} (i'_{k+1}, j'_{k+1})$. It is now routine to show, by using Lemma 6, that inductive assumption (4) is preserved.

Thus, the strategy is winning for Duplicator by inductive assumptions (1) and (2).

□

For the next theorem we represent colored equivalence structures $\mathcal{A}$ in three lists. The first one lists equivalence classes of $\mathcal{A}$ in increasing order of their types; the second and the third list the sequences $\{q^{\mathcal{A},C}_{(i,j),k}\}_{0\le i,j,k\le n}$ and $\{q^{\mathcal{A},N}_{(i,j),k}\}_{0\le i,j,k\le n}$ respectively:

Theorem 6. *Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on equivalence structures with one color $\mathcal{A}$ and $\mathcal{B}$. The constant that bounds the running time is n^3.* □

Fix $s > 1$, let $\mathcal{A}$ be an equivalence structure with s many colors. For each element x of $\mathcal{A}$, define the **characteristic** of x as defined in the previous sections. There are 2^s distinct characteristics. Order them in lexicographic order: $ch_1, ..., ch_{2^s}$. Construct the structure $\mathcal{A}' = (A; E, Q_1, ..., Q_{2^s})$ such that for all $1 \le i \le 2^s$, $Q_i = \{x \in A \mid ch(x) = ch_i\}$. Clearly, for distinct characteristics ch_i and ch_j we have $Q_i \cap Q_j = \emptyset$. Moreover, $\mathcal{A}$ and $\mathcal{B}$ are isomorphic if and only if $\mathcal{A}'$ and $\mathcal{B}'$ are isomorphic.

For an equivalence class X, we define the **type** of X, $tp(X)$, as a sequence $(i_1, i_2, ..., i_{2^s})$ such that in X the number of element with characteristic ch_j is i_j for all $1 \le j \le 2^s$.

Definition 5. *Let $\kappa = (i_1, ..., i_{2^s})$ and $\lambda = (i'_1, ..., i'_{2^s})$ be two types of equivalence classes. For $1 \le j \le 2^s$, we say that κ is (j, n)-**equivalent** to λ, denoted by $\kappa \equiv^j_n \lambda$, if the following holds.*

1. *If $i_j < n$ then $i'_j = i_j$, otherwise $i'_j \ge n$; and*
2. *For all $1 \le l \le 2^s$ where $l \ne j$, if $i_l < n - 1$ then $i'_l = i_l$, otherwise $i'_l \ge n - 1$.*

Let X and Y be equivalence classes of types κ and λ respectively. If $\kappa \equiv^j_n \lambda$, then Duplicator wins the n-round EF game played on structures $(X; E \upharpoonright X, P_1 \upharpoonright X, ..., P_s \upharpoonright X)$ and $(Y; E \upharpoonright Y, P_1 \upharpoonright Y, ..., P_s \upharpoonright Y)$, given that Spoiler selects an element $x \in X$ with characteristic ch_j.

For type λ, $1 \le j \le 2^s$ and $k \ge 1$, we set $C^{\mathcal{A},j}_{\lambda,k}$ be the set $\{X \mid X$ is an equivalence class of $\mathcal{A}$ and $tp(X) \equiv^j_k \lambda\}$. Similar to the case of equivalence structures with one color, one introduces notations $q^{A,j}_{\lambda,k}$, $q^j_{\lambda,k}$, and $\mathcal{A}^j(\lambda, k)$.

Definition 6. *Consider the game $G_n(\mathcal{A}, \mathcal{B})$ played on equivalence structures with s colors. For $1 \le j \le 2^s$, we say that a* **disparity occurs with respect to** *ch_j if there exists a type $\lambda = (i_1, ..., i_{2^s})$ and $n > k \ge 0$ such that the following holds:*

1. *$k = q^j_{\lambda,n-k}$*
2. *In one of $\mathcal{A}^j(\lambda, n-k)$, there is an equivalence class whose type is $(j, n-k)$-equivalent to λ, and no such equivalence class exists in the other structure.*

The proof of the following are similar to Lemma 7 and Theorem 6:

Lemma 8. *Let $\mathcal{A}$ and $\mathcal{B}$ be equivalence structures with s colors. Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ if and only if no disparity occurs with respect to ch_j for any $1 \leq j \leq 2^s$.* □

Theorem 7. *Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on equivalence structures with s colors. The constant that bounds the running time is n^{2^s+1}.* □

5 Embedded Equivalence Structures

An ***embedded equivalence structure of height*** h is a structure $\mathcal{A} = (A; E_1, E_2, ..., E_h)$ such that each E_i, $1 \leq i \leq h$, is an equivalence relation, and $E_i \subseteq E_j$ for $i < j$. In this section we give a full solution for EF played on embedded equivalence structures of height h. We start with the case when $h = 2$. The case for $h > 2$ will be explained later.

Let $\mathcal{A} = (A; E_1, E_2)$ be an embedded equivalence structure of height 2. We say that an E_2-equivalence class X has type $tp(X) = (q_1, \ldots, q_t)$ if the largest E_1-equivalence class contained in X has size t and for all $1 \leq i \leq t$, q_i is the number of E_1-equivalence classes of size i contained in X. Thus, $\sum_{i=1}^{t}(q_i \times i) = |X|$. For two types $\sigma = (q_1, \ldots, q_{t_1})$ and $\tau = (q'_1, \ldots, q'_{t_2})$, we say $\sigma = \tau$ if $t_1 = t_2$ and $q_i = q'_i$ for all $1 \leq i \leq t_1$.

For $X \subseteq A$, we use $(X; E_1 \upharpoonright X)$ to denote the equivalence structure obtained by restricting E_1 on X. Given two E_2-equivalence classes X and Y of types σ and τ respectively, we say that σ is n-equivalent to τ, denoted by $\sigma \equiv_n \tau$, if Duplicator wins the n-round game played on structures $(X; E_1 \upharpoonright X)$ and $(Y; E_1 \upharpoonright Y)$. Note that if $\sigma \equiv_n \tau$, then $\sigma \equiv_i \tau$ for all $i \leq n$.

We need the following notations:

- For type σ and $i \geq 1$, set $C^{\mathcal{A}}_{\sigma,i}$ be the set $\{X \mid X$ is an E_2-equivalence class of $\mathcal{A}$ and $tp(X) \equiv_i \sigma\}$.
- Set $q^{\mathcal{A}}_{\sigma,i} = |C^{\mathcal{A}}_{\sigma,i}|$.
- For embedded equivalence structure $\mathcal{A}$ and $\mathcal{B}$, set $q^{\sigma,i} = min\{q^{\mathcal{A}}_{\sigma,i}, q^{\mathcal{B}}_{\sigma,i}\}$
- Set $\mathcal{A}(\sigma, i)$ be the embedded equivalence structure of height 2 obtained from $\mathcal{A}$ by removing $q^{\sigma,i}$ equivalence classes whose types are i-equivalent to σ.

Observe in round k of the game $G_n(\mathcal{A}, \mathcal{B})$, if Spoiler selects an element from an E_2-equivalence class X in $\mathcal{A}$, and Duplicator responds by selecting another element from an E_2-equivalence class Y in $\mathcal{B}$ such that $tp(Y) \equiv_{n-k} tp(X)$, there is no point for Spoiler to keep playing inside X because this will guarantee a win for Duplicator. Intuitively, $\mathcal{A}(\sigma, n-k)$ contains all the E_2-equivalence classes for Spoiler to choose elements from after $q^{\sigma,n-k}$ many E_2-equivalence classes whose types are $(n-k)$-equivalent to σ have been chosen.

Definition 7. *Consider the game $G_n(\mathcal{A}, \mathcal{B})$ played on embedded equivalence structures of height 2. We say that a* **disparity** *occurs if there exists a type σ and $n > k \geq 0$ such that the following holds.*

1. *$k = q^{\sigma,n-k}$.*
2. *In one of $\mathcal{A}(\sigma, n-k)$ and $\mathcal{B}(\sigma, n-k)$, there is an E_2-equivalence class whose type is $(n-k)$-equivalent to σ, and no such E_2-equivalence class exists in the other structure.*

Lemma 9. *Suppose $\mathcal{A}$ and $\mathcal{B}$ are two embedded equivalence structures of height 2. Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ if and only if no disparity occurs.*

Proof. Suppose disparity occurs in $G_n(\mathcal{A}, \mathcal{B})$ witnessed by σ and k, in $\mathcal{A}(\sigma, n-k)$ there is an E_2-equivalence class whose type is $(n-k)$-equivalent to σ, and no such E_2-equivalence class exists in $\mathcal{B}(\sigma, n-k)$. Using these, it is not hard to prove that Spoiler wins the game.

Suppose that no disparity occurs in the game $G_n(\mathcal{A}, \mathcal{B})$, we describe a strategy for Duplicator. Let us assume that the players have produced a k-round play (a_1, b_1), $(a_2, b_2), \ldots, (a_k, b_k)$. Let σ_i and τ_i be the types of a_i and b_i, respectively with $1 \leq i \leq k$. Our inductive assumptions on this k-round play are the following:

1. The map $a_i \to b_i$ is partial isomorphism.
2. For all $1 \leq i \leq k$, $\sigma_i \equiv_{n-i} \tau_i$.
3. Let $\mathcal{A}'$ and $\mathcal{B}'$ be the equivalence structures obtained by removing the E_2-equivalence classes $[a_1]_{E_2}, \ldots, [a_k]_{E_2}$ from $\mathcal{A}$ and the equivalence classes $[b_1]_{E_2}, \ldots, [b_k]_{E_2}$ from $\mathcal{B}$, respectively. We assume in game $G_{n-k}(\mathcal{A}', \mathcal{B}')$ no disparity occurs.

Assume that Spoiler selects an element $a_{k+1} \in A$. The case when Spoiler selects an element from B is treated as below. Duplicator responds to this move by choosing b_{k+1} as follows. If $a_{k+1} = a_i$ then $b_{k+1} = b_i$. Otherwise, if $E_1(a_i, a_{k+1})$ is true in $\mathcal{A}$, then Duplicator chooses a new b_{k+1} such that $E_1(b_i, b_{k+1})$. If $E_2(a_i, a_{k+1})$ is true in $\mathcal{A}$ and there is no j such that $E_1(a_j, a_{k+1})$, then Duplicator chooses a new b_{k+1} such that $E_2(b_i, b_{k+1})$ and there is no j such that $E_1(b_j, b_{k+1})$. By (2) of the inductive assumption Duplicator can always select such an element b_{k+1} by following its winning strategies.

Assume a_{k+1} is not equivalent to any of the elements $a_1, ..., a_k$. Let X be the E_2-equivalence class in $\mathcal{A}$ that contains a_{k+1}. Duplicator selects b_{k+1} from an E_2-equivalence class Y in $\mathcal{B}$ such that $tp(X) \equiv_{n-k} tp(Y)$. Duplicator is able to select such an element as otherwise disparity would occur as witnessed by the type of X and 0.

The inductive assumption (1) and (2) can be easily checked to hold on the play $(a_1, b_1), \ldots, (a_k, b_k), (a_{k+1}, b_{k+1})$. To show that the assumption (3) is preserved, consider the structures $\mathcal{A}''$ and $\mathcal{B}''$ obtained by removing $[a_1]_{E_2}, \ldots, [a_k]_{E_2}, [a_{k+1}]_{E_2}$ and $[b_1]_{E_2}, \ldots, [b_k]_{E_2}, [b_{k+1}]_{E_2}$ from $\mathcal{A}$ and $\mathcal{B}$, respectively. Suppose disparity occurs in $G_{n-k-1}(\mathcal{A}'', \mathcal{B}'')$ as witnessed by some type τ and $t < n-k-1$. There are two cases. If $tp([a_{k+1}]) \equiv_{n-k-t-1} \tau$, then $tp([b_{k+1}]) \equiv_{n-k-t-1} \tau$, and disparity must occur in $G_{n-k}(\mathcal{A}', \mathcal{B}')$ as witnessed by τ and $t+1$. If $tp([a_{k+1}]) \not\equiv_{n-k-t-1} \tau$, then $tp([b_{k+1}]) \not\equiv_{n-k-t-1} \tau$, and disparity must occur in $G_{n-k}(\mathcal{A}', \mathcal{B}')$ as witnessed by τ and t, contradicting our assumption. Hence the strategy is a winning strategy. □

Theorem 8. *Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on embedded equivalence structures of height 2. The constant that bounds the running time is $(n+1)^n$.*

Proof. We represent structure $\mathcal{A} = (A; E_1, E_2)$ by a tree and a list. The tree has height 3. The leaves of the tree are all elements in A. Two leaves x, y have the same parent if $E_1(x, y)$, and x, y have the same ancestor at level 1 if $E_2(x, y)$. Intuitively, we can view the root of tree as A, the internal nodes at level 1 represent all E_2-equivalence classes on A, and the children of each E_2-equivalence class X at level 2 are all E_1-equivalence classes contained in X. We further require that representations of E_2 and E_1-equivalence classes are put in left-to-right order according to their cardinalities.

The list is $q^{\mathcal{A}}_{\sigma_1,1}, \dots, q^{\mathcal{A}}_{\sigma_t,1}, \dots, q^{\mathcal{A}}_{\sigma_1,n}, \dots, q^{\mathcal{A}}_{\sigma_t,n}$ where each σ_i is a type of E_2-equivalence class, and $q^{\mathcal{A}}_{\sigma_i,j}$ is as defined above. Each $q^{\mathcal{A}}_{\sigma_i,j}$ has a value between 0 and n and if it is greater than n, we set it to n. The algorithm checks whether disparity occurs in $G_n(\mathcal{A}, \mathcal{B})$ by examining the list. There can be at most $(n + 1)^n$ pairwise non-n-equivalent types. Therefore, checking disparity requires a time bounded by $(n + 1)^{n+1}$. □

For the case when $\mathcal{A}$ and $\mathcal{B}$ are two embedded equivalence structures of height h, where $h > 2$, we give a similar definition of the type of an E_h-equivalence class. We can then describe the winning conditions for Spoiler and Duplicator in a similar way.

Let $\mathcal{A}$ be an embedded equivalence structure of height h where $h > 2$. For an E_h-equivalence class X, we recursively define $tp(X)$, the type of X. Set $tp(X)$ be $(q_{\sigma_1}, \dots, q_{\sigma_t})$ that satisfies the following property.

1. Each σ_i is the type of an E_{h-1}-equivalence class.
2. σ_t is the maximum type in lexicographic order among all types of E_{h-1}-equivalence classes contained in X.
3. The list $\sigma_1, ..., \sigma_t$ contains all possible types of E_{h-1}-equivalence classes less or equal to σ_t ordered lexicographically.
4. For all $1 \le i \le t$, q_{σ_i} is the number of all E_{h-1}-equivalence classes contained in X whose type are σ_i.

We note that these types allow us to solve the isomorphism problem for embedded equivalence structures of height h in linear time on the size of the structures.

Let $\kappa = (q_{\sigma_1}, ..., q_{\sigma_s})$ and $\lambda = (q'_{\sigma_1}, ..., q'_{\sigma_t})$ be types of two E_h-equivalence classes X and Y, respectively. We say $\kappa = \lambda$ if $s = t$ and $q_{\sigma_i} = q'_{\sigma_i}$ for all $1 \le i \le s$. We say $\kappa \equiv_n \lambda$ if the structures $(X; E_1 \upharpoonright X, \dots, E_{h-1} \upharpoonright X)$ and $(Y; E_1 \upharpoonright Y, \dots, E_{h-1} \upharpoonright Y)$ are n-equivalent.

Similarly to the case of embedded equivalence structures of height 2, we re-introduce the notions $C^{\mathcal{A}}_{\sigma,i}$, $q^{\mathcal{A}}_{\sigma,i}$, $q^{\sigma,i}$, $\mathcal{A}(\sigma, i)$ and disparity in the game $G_n(\mathcal{A}, \mathcal{B})$.

Lemma 10. *Suppose $\mathcal{A}$ and $\mathcal{B}$ are two embedded equivalence structures of height h where $h \ge 2$. Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ if and only if no disparity occurs.* □

The number of pairwise non-n-equivalent types of E_h-equivalence classes is at most $(n + 1)^{\cdot^{\cdot^{\cdot^{(n+1)^n}}}}$ where the tower of $(n + 1)$ has height h. Thus, by the lemma above, we have:

Theorem 9. *Fix $n \in \omega$. There is an algorithm that runs in constant time and decides if Duplicator wins the game $G_n(\mathcal{A}, \mathcal{B})$ on embedded equivalence structures $\mathcal{A}$ and $\mathcal{B}$ of height h. The constant that bounds the running time is $(n + 1)^{\cdot^{\cdot^{\cdot^{((n+1)}}}}$ where the tower has height h.* □

6 Trees

In this section we are interested in **trees**; these are finite structures of the type $\mathcal{T} = (T, \leq)$, where the relation $\leq$ is a partial order on T such that $\mathcal{T}$ has the greatest element (the root), and the set $\{y \mid x \leq y\}$ for any given $x \in T$ is a linearly ordered set under $\leq$. We call an element a **leaf** of the tree $\mathcal{T}$ if it is a minimal element; otherwise we call it an **internal node**. A **path** in $\mathcal{T}$ is a maximal linearly order subset of T. The length of a given path is the number of elements in the path. The **height** of $\mathcal{T}$ is the length of the largest path in $\mathcal{T}$. We say that the **level** of a node $x \in T$ is j if the distance from x to the root is j. We fix number $h \geq 2$, and restrict ourselves to the class $\mathcal{K}_h$ of all trees of height at most h. Deciding Ehrenfeucht-Fraïssé games on trees from $\mathcal{K}_h$ can be done directly by using the techniques from the previous section. Instead, we reduce the problem of deciding Ehrenfeucht-Fraïssé games on trees in K_h to one for embedded equivalence structures of height $h + 1$.

We transform trees from the class $\mathcal{K}_h$ into the class of embedded equivalence structures of height h in the following manner. Let $\mathcal{T}$ be a tree in $\mathcal{K}_h$. We now define an embedded equivalence structure $\mathcal{A}(\mathcal{T})$ as follows. The domain D of $\mathcal{A}(T)$ is now $T \cup \{a_x \mid x$ is a leaf of $\mathcal{T}\}$. We define the equivalence relation E_i , $1 \leq i \leq h$, on the domain as follows. The relation E_1 is the minimal equivalence relation that contains $\{(x, a_x) \mid x$ is a leaf of $\mathcal{T}\}$. Let $x_1, \ldots, x_s$ be all elements of $\mathcal{T}$ at level $h - i + 1$, where $1 \leq i < h$. Let $\mathcal{T}_1$, $\ldots, \mathcal{T}_s$ be the subtrees of $\mathcal{T}$ whose roots are $x_1, \ldots, x_s$, respectively. Set E_i be the minimal equivalence relation that contains $E_{i-1} \cup T_1^2 \cup \ldots \cup T_s^2$. It is clear that $E_i \subseteq E_{i+1}$ for all $1 \leq i \leq h$. Thus we have the embedded equivalence structure $\mathcal{A}(T) = (D; E_1, ..., E_h)$.

Lemma 11. *For trees $\mathcal{T}_1$ and $\mathcal{T}_2$, $\mathcal{T}_1 \cong \mathcal{T}_2$ if and only if $\mathcal{A}(\mathcal{T}_1) \cong \mathcal{A}(\mathcal{T}_2)$. In particular, Duplicator wins $G_n(\mathcal{T}_1, \mathcal{T}_2)$ if and only if Duplicator wins $G_n(\mathcal{A}(\mathcal{T}_1), \mathcal{A}(T_2))$.*

Proof. Suppose $\mathcal{T}$ is a tree in the class $\mathcal{K}_h$. Take an element $x \in T$. By construction of $\mathcal{A}(\mathcal{T})$, the following statements are true.

– x is a leaf in $\mathcal{T}$ if and only if $|\{y \mid E_1(x, y)\}| = 2$ in $\mathcal{A}(\mathcal{T})$.
– x is the root of $\mathcal{T}$ if and only if $|\{y \mid E_h(x, y)\}| = 1$ in $\mathcal{A}(\mathcal{T})$.

We define the **level** of x in $\mathcal{A}(\mathcal{T})$ as follows. If x is the root of $\mathcal{T}$, the level of x is 0. Otherwise, if x is an internal node, the level of x in $\mathcal{A}(\mathcal{T})$ is the largest l such that $|\{y \mid E_{h-l+1}(x, y)\}| > 1$. If x is a leaf, we define the level of x in $\mathcal{A}(\mathcal{T})$ to be the largest $l + 1$ such that there is an internal node y such that $E_{h-l+1}(x, y)$.

By definition, for all $x \in T$, the level of x in $\mathcal{T}$ is the level of x in $\mathcal{A}(\mathcal{T})$. For $x, y \in T$, $x \leq y$ in $\mathcal{T}$ if and only in $\mathcal{A}(\mathcal{T})$ x has level s and y has level t such that $s \geq t$ and $E_{h-t+1}(x, y)$. Thus, for two trees from $\mathcal{K}_h$, $\mathcal{T}_1$ and $\mathcal{T}_2$, and a mapping $f : T_1 \to T_2$, f is an isomorphism between $\mathcal{T}_1$ and $\mathcal{T}_2$ if and only if f is an isomorphism between $\mathcal{A}(\mathcal{T}_1)$ and $\mathcal{A}(\mathcal{T}_2)$.

To prove the second part of the lemma, one direction is clear. For the other direction, assume that there is a winning strategy for Duplicator on the game $G_n(\mathcal{T}_1, \mathcal{T}_2)$. We describe a strategy for Duplicator on the game $G_n(\mathcal{A}(\mathcal{T}_1), \mathcal{A}(\mathcal{T}_2))$ where $\mathcal{A}(\mathcal{T}_1) = (D_1; E_1, ..., E_h)$ and $\mathcal{A}(\mathcal{T}_2) = (D_2; E_1, ..., E_h)$. Let us assume that the players have produced a k-round play $(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)$. Assume on this k-round play the map $x_i \to y_i$ is a partial isomorphism between $\mathcal{A}(\mathcal{T}_1)$ and $\mathcal{A}(\mathcal{T}_2)$.

Assume that Spoiler selects an element $x_{k+1} \in D_1$. Duplicator responds to this move by choosing x_{k+1} as follows. If $x_{k+1} = x_i$ then $y_{k+1} = y_i$. Otherwise, if $x_{k+1} \in T_1$, then Duplicator selects an element $y_{k+1} \in T_2$ according to its winning strategy on $G_n(\mathcal{T}_1, \mathcal{T}_2)$. If $x_{k+1} = a_x$ for some leaf $x \in \mathcal{T}_1$. Then Duplicator responds by selecting $y_{k+1} = a_y$ where y is the leaf in T_2 that corresponds to x in Duplicator's winning strategy in $G_n(\mathcal{T}_1, \mathcal{T}_2)$. It is clear that $x_i \to y_i$ where $1 \le i \le k+1$ is also a partial isomorphism between $\mathcal{A}(\mathcal{T}_1)$ and $\mathcal{A}(\mathcal{T}_2)$. Therefore the strategy described is a winning strategy for Duplicator on game $G_n(\mathcal{A}(\mathcal{T}_1), \mathcal{A}(\mathcal{T}_2))$. □

Using the lemma above, one can now prove this:

Theorem 10. *Fix $n \in \omega$. There is an algorithm that runs in constant time and decides if Duplicator wins the game $G_n(\mathcal{T}_1, \mathcal{T}_2)$, where $\mathcal{T}_1, \mathcal{T}_2 \in \mathcal{K}_h$. The constant that bounds the running time is $(n+1)^{\cdot^{\cdot^{\cdot^{(n+1)^{(n+1)}}}}}$ where the tower has height h.* □

7 Boolean Algebras with Distinguished Ideals

A **Boolean algebra (BA) with distinguished ideals** is a structure $\mathcal{A} = (A; \le, 0, 1, I_1, \ldots, I_s)$, where $(A; \le, 0, 1)$ forms a BA and each I_j is an ideal of the algebra $(A; \le, 0, 1)$. The set of **atoms** of $\mathcal{A}$, denoted $At(\mathcal{A})$, is the set $\{a \mid \forall y(0 \le y \le a \to y = 0 \vee y = a)\}$. Since we restrict ourselves to finite structures, the BA $(A; \le, 0, 1)$ can be identified with the structure $(2^{X_A}; \subseteq, \emptyset, X_A)$, where $X_A = At(\mathcal{A})$ and 2^{X_A} is the collection of all subsets of X_A. Moreover, for each ideal I_j there exists a set $A_j \subset At(\mathcal{A})$ such that $I_j = 2^{A_j}$. Hence the original structure $\mathcal{A}$ can be identified with the structure: $(2^{X_A}; \subseteq, \emptyset, X_A, 2^{A_1}, \ldots, 2^{A_s})$. For each element $x \in At(\mathcal{A})$, define the **characteristic** of x, $ch(x)$, as a binary sequence $(t_1, t_2, ..., t_s)$ such that for each $1 \le i \le s$, $t_i \in \{0, 1\}$, $t_i = 1$ if $x \in A_i$ and $t_i = 0$ otherwise. For each characteristic $\epsilon \in \{0, 1\}^s$ consider the set $A_\epsilon = \{x \in At(\mathcal{A}) \mid ch(x) = \epsilon)\}$. This defines the ideal I_ϵ in the Boolean algebra $(2^{X_A}; \subseteq, \emptyset, X_A)$. Moreover, we can also identify this ideal with the Boolean algebra $(2^{A_\epsilon}; \subseteq, \emptyset, A_\epsilon)$. There are 2^s pairwise distinct characteristics. Let $\epsilon_1, \ldots, \epsilon_{2^s}$ be the list of all characters. We denote by $\mathcal{A}'$ the following structure: $(2^X; \subseteq, \emptyset, X, 2^{A_{\epsilon_1}}, \ldots, 2^{A_{\epsilon_{2^s}}})$.

Lemma 12. *Let $\mathcal{A} = (2^{X_A}; \subseteq, \emptyset, X_A, 2^{A_1}, \ldots, 2^{A_s})$ be a Boolean algebra with distinguished ideals*

1. *For any two distinct characteristics ϵ and δ we have $I_\epsilon \cap I_\delta = \{\emptyset\}$.*
2. *For any element $a \in 2^X$ there are elements $a_\epsilon \in I_\epsilon$ such that $a = \cup_\epsilon a_\epsilon$.*
3. *The Boolean algebra $(2^{X_A}; \subseteq, \emptyset, X_A)$ is isomorphic to the Cartesian product of the Boolean algebras I_ϵ.*
4. *$\mathcal{A}$ and $\mathcal{B}$ are isomorphic if and only if $\mathcal{A}'$ and $\mathcal{B}'$ are isomorphic.* □

The next lemma connects the structure $\mathcal{A}'$ and $\mathcal{A}$ in terms of characterizing the winner of the game $G_n(\mathcal{A}, \mathcal{B})$.

Lemma 13. *Duplicator wins the game $G_{n+1}(\mathcal{A}, \mathcal{B})$ if and only if each of the following two conditions are true:*

1. *For each characteristic ϵ, $|A_\epsilon| \ge 2^n$ if and only if $|B_\epsilon| \ge 2^n$.*
2. *For each characteristic ϵ, if $|A_\epsilon| < 2^n$ then $|A_\epsilon| = |B_\epsilon|$.*

Proof. Assume that for some ϵ, we have $|A_\epsilon| \neq |B_\epsilon|$ and $|B_\epsilon| < 2^n$. Let us assume that $|A_\epsilon| \geq 2^n$. The case when $|A_\epsilon| < 2^n$ is treated in a similar manner. We describe a winning strategy for Spoiler. Spoiler starts by taking elements $a_1, a_2, \ldots$ in A_ϵ. For each $i \leq n$ the element a_i is such that $|At(a_i)| \geq 2^{n-i}$, where $At(a)$ denotes the set of atoms below a. The elements $a_1, a_2, \ldots$ are such that for each i, either $a_i \subset a_{i-1}$ or $a_i \cap a_{i-1} = \emptyset$. Consider the k round play $(a_1, b_1), \ldots, (a_k, b_k)$ where $k < n$. Let $e < k$ be the last round for which $a_k \subset a_e$. If no such e exists, let $a_e = 2^{A_\epsilon}$ and $b_e = 2^{B_\epsilon}$. We have the following inductive assumptions.

– $|At(a_k)| \geq 2^{n-k}$ and $|At(a_e \setminus (a_{e+1} \cup \ldots \cup a_k))| \geq 2^{n-k}$.
– Either $|At(b_k)| < 2^{n-k}$ or $|At(b_e \setminus (b_{e+1} \cup \ldots \cup b_k))| < 2^{n-k}$.

There are two cases.

Case 1. Assume that $|At(b_k)| < 2^{n-k}$ and $|At(a_k)| \geq 2^{n-k}$. In this case Spoiler selects a_{k+1} such that $a_{k+1} \subset a_k$, $a_{k+1} \neq \emptyset$, $|At(a_{k+1})| \geq 2^{n-k-1}$ and $|At(a_k \setminus a_{k+1})| \geq 2^{n-k-1}$. Note that Duplicator must choose b_{k+1} strictly below b_k. Then either $|At(b_{k+1})| < 2^{n-k-1}$ or $|At(b_k \setminus b_{k+1})| < 2^{n-k-1}$

Case 2. Assume that $|At(b_k)| \geq 2^{n-k}$ and $|At(a_k)| \geq 2^{n-k}$. In this case, Spoiler selects a_{k+1} such that $a_{k+1} \subset a_e$, $a_{k+1} \neq \emptyset$, $a_{k+1} \cap (a_{e+1} \cup \ldots \cup a_k) = \emptyset$, $|At(a_{k+1})| \geq 2^{n-k-1}$, and $|At(a_e \setminus (a_{e+1} \cup \ldots \cup a_{k+1}))| \geq 2^{n-k-1}$. Note that by definition of e, $|At(b_e)| < 2^{n-k}$ and for each $e + 1 \leq i \leq k - 1$, $|At(b_i)| \geq 2^{n-i}$ as otherwise b_k would be below b_i. Hence $|At(b_k \setminus (b_{e+1} \cup \ldots \cup b_k))| < 2^{n-k}$. Duplicator must choose b_{k+1} strictly below b_e and disjoint with $b_{e+1}, \ldots, b_k$. Therefore either $|At(b_{k+1})| < 2^{n-k-1}$ or $|At(b_e) \setminus At(b_{e+1} \cup \ldots \cup b_{k+1})| < 2^{n-k-1}$.

After n rounds, by the inductive assumption, it is either $|At(b_n)| = 0$ or $|At(b_e \setminus (b_{e+1} \cup \ldots \cup b_n))| = 0$. If the former, then Spoiler wins by selecting $a_{n+1} \subset At(a_n)$;otherwise, Spoiler wins by selecting $a_{n+1} \subset a_e \setminus (a_{e+1} \cup \ldots \cup a_n)$.

Now we prove that the conditions stated in the lemma suffice Duplicator to win the $(n+1)$-round game $G_{n+1}(\mathcal{A}, \mathcal{B})$. Let us assume that the players have produced a k-round play $(a_1, b_1), (a_2, b_2), ..., (a_k, b_k)$. Our inductive assumptions on this k-round play are the following:

1. The map $a_i \to b_i$ is a partial isomorphism.
2. For each a_i, $1 \leq i \leq k$, let $a_i = \cup_\epsilon a_\epsilon$ be as stipulated in Lemma 12(2). For each a_ϵ, let e be the last round such that $a_\epsilon \subseteq a_e$, if there is no such round, then assume $a_e = At(I_\epsilon)$. Let d be the last round such that $a_d \subseteq a_\epsilon$, if there is no such round, then assume $a_d = \emptyset$. Let $b_i = \cup_\epsilon b_\epsilon$. The conditions for b_ϵ are the following:
   – $|At(a_\epsilon \setminus a_d)| \geq 2^{n-i}$ if and only if $|At(b_\epsilon \setminus a_d)| \geq 2^{n-i}$; $|At(a_e \setminus a_\epsilon)| \geq 2^{n-i}$ if and only if $|At(b_e \setminus b_\epsilon)| \geq 2^{n-i}$.
   – If $|At(a_\epsilon \setminus a_d)| < 2^{n-i}$ then $|At(b_\epsilon \setminus a_d)| = |At(a_\epsilon \setminus a_d)|$; If $|At(a_e \setminus a_\epsilon)| < 2^{n-i}$ then $|At(b_e \setminus b_\epsilon)| = |At(a_e \setminus a_\epsilon)|$.

Assume that Spoiler selects an element $a_{k+1} \in A$. Duplicator responds to this move by choosing b_{k+1} as follows. If $a_{k+1} = a_i$ then $b_{k+1} = b_i$. Otherwise, suppose $a_{k+1} = \cup a_\epsilon$ as stipulated in Lemma 12(2). For each a_ϵ, let d, e be as described in the inductive assumptions. We select each b_ϵ by the following rules.

– If $|At(a_\epsilon \setminus a_d)| \geq 2^{n-k-1}$ then select b_ϵ such that $|At(b_\epsilon \setminus a_d)| \geq 2^{n-k-1}$; If $|At(a_e \setminus a_\epsilon)| \geq 2^{n-k-1}$ then $|At(b_e \setminus b_\epsilon)| \geq 2^{n-k-1}$.
– If $|At(a_\epsilon \setminus a_d)| < 2^{n-k-1}$ then select b_ϵ such that $|At(b_\epsilon \setminus a_d)| = |At(a_\epsilon \setminus a_d)|$; If $|At(a_e \setminus a_\epsilon)| < 2^{n-k-1}$ then $|At(b_e \setminus b_\epsilon)| = |At(a_e \setminus a_\epsilon)|$.

Finally, Duplicator selects $b_{k+1} \in B$ such that $b_{k+1} = \cup_\epsilon b_\epsilon$.

Note the inductive assumptions guarantee that Duplicator is able to make such a move. It is clear that the inductive assumptions also hold on the $(k+1)$-round play $(a_1, b_1), \ldots, (a_{k+1}, b_{k+1})$. Hence, the strategy described must be a winning strategy due to the fact that Duplicator preserves the inductive assumption (1) at each round. The lemma is proved. □

For the next result, we represent the Boolean algebras by listing their atoms in 2^s lists, where the i^{th} list lists all atoms with characteristic ϵ_i:

Theorem 11. *Fix $n \in \omega$. There exists an algorithm that runs in constant time and decides whether Duplicator wins the game $G_{n+1}(\mathcal{A}, \mathcal{B})$ on BAs $\mathcal{A}$ and $\mathcal{B}$ with s distinguished ideals. The constant that bounds the running time is $2^s \cdot 2^n$.* □

References

1. S. Arora, R. Fagin, *On winning strategies in EF games*. Theoretical Computer Science 174, 97-121, 1997.
2. A. Dawar, *Infinitary logic and inductively definability over finite structures*. Information and Computation, volume 119, 2:160-175, 1995.
3. A. Ehrenfeucht, *An appliction of games to the completeness problem for formalized theories*. Fundamenta Mathematicae,49:129-141,1961.
4. R. Fraïssé, *Sur quelques classfications des systèm de relations*. Université d Alger, Publication Scientifiques, Sér. A,1:35-182, 1954.
5. M. Grohe. *Equivalence in finite-variable logics is complete for polynomial time*. Proceeding FOCS 96.
6. W. Hodeges, *Model theory*, Combridge University Press, 1993.
7. N. Immerman, D. Kozen, *Definability with bounded number of bounded variables*. Logic in Computer Science, 236-244, 1987.
8. C. Karp, *Finite quantifier equivalence*, In J.W. Addison, L. Henkin, and A. Tarski, editors, The Theory of Models, 407-412. North Holland, 1965.
9. P. Kolaitis, J. Panttaja, *On the complexity of existential pebble games*. In M. Baaaz and J.A. Makowsky (Eds): CSL 2003, LNCS2803, pp. 314-329, 2003.
10. C. Lautemann, N. Schweikardt, *An EF approach to collapse results for first-order queries over embedded databases*. In A. Ferreira and H. Reichel, editors, STACS'01:18th Annual Symposium on Theoretical Aspects of Computer Science, volume 2010 of Lecture Notes in Computer Science, 455-466, Dresden, Germany, Springer, 2001.
11. L. Libkin, *Elements of finite model theory*. Springer-Verlag, 2004.
12. J. Marcinkowski, *Directed Reachability: From Ajtai-Fagin to Ehrenfeucht-Fraïssé games*. Proceedings of CSL99, Springer LNCS, 1999.
13. A. Mekler, S. Shelah, J. Väänänen, *The EF-game of length ω_1*. Transactions of the American mathematical Society, Volume 339, Number 2, 1993.

14. E. Pezzoli, *Computational complexity of EF games on finite structures*. In Computer Science Logic, 12th International Workshop, CSL'98, 159-170, 1999.
15. D. Scott, *Logic with denumerable long formulas and finite strings of quantifiers*. In J.W. Addison, L. Henkin, and A. Tarski, editors, The Theory of Models, 329-341. North Holland, 1965.
16. N. Schweikardt, *An EF game appraoch to collapse results in database theory*. To appear in Information and Computation, 2006.

The Law of the Iterated Logarithm for Algorithmically Random Brownian Motion

Bjørn Kjos-Hanssen and Anil Nerode

Department of Mathematics, Cornell University,
Ithaca, NY 14853, U.S.A.
{bjoern,anil}@math.cornell.edu

Abstract. Algorithmic randomness is most often studied in the setting of the fair-coin measure on the Cantor space, or equivalently Lebesgue measure on the unit interval. It has also been considered for the Wiener measure on the space of continuous functions. Answering a question of Fouché, we show that Khintchine's law of the iterated logarithm holds at almost all points for each *Martin-Löf random path of Brownian motion*. In the terminology of Fouché, these are the *complex oscillations*. The main new idea of the proof is to take advantage of the Wiener-Carathéodory measure algebra isomorphism theorem.

Keywords: Brownian motion, randomness, law of the iterated logarithm, Kolmogorov complexity.

1 Introduction

Algorithmic randomness for Brownian motion was introduced by Asarin and Pokrovskii. They defined what they called (according to the English translation [1]) *truly random* continuous functions. Fouché [3] called these functions *complex oscillations*.

In this article we answer a question of Fouché (see [5]) by showing that for each complex oscillation, Khintchine's law of the iterated logarithm holds at almost every point. To that end, in Section 2 we will borrow a construction from the proof of the Wiener-Carathéodory measure algebra isomorphism theorem. For the full statement of this theorem, the reader may consult for example Royden [6], Theorem 15.3.4; we shall not need it. We believe our method based on this isomorphism theorem can be used to yield other results than the one presented here. Namely, algorithmic randomness for the unit interval $[0,1]$ has been studied more extensively than algorithmic randomness for the space $C[0,1]$ of continuous functions, and the isomorphism theorem allows a transfer of some results. For a general introduction to computability theory and algorithmic randomness on $[0,1]$, the reader may consult [2].

Definition 1. *Suppose Ω is a set, $\mathscr{F} = \{T_i : i \in \mathbb{N}\}$ a countable Boolean algebra of subsets of Ω, μ a probability measure on the σ-algebra generated by $\mathscr{F}$. Let $t \in [0,1]$. Suppose $\phi : \mathbb{N}^2 \to \mathbb{N}$ is a total function Turing reducible to t.*

The sequence $U_n = \bigcup_m T_{\phi(n,m)}$, $n \in \mathbb{N}$ is called a t-uniform sequence of $\Sigma_1^t(\mathscr{F})$ sets.

A t-effective $\mathscr{F}$-null set is a set $A \subseteq \Omega$ such that for some such ϕ,

1. *$A \subset \bigcap_n U_n$, and*
2. *μU_n goes effectively to 0 as $n \to \infty$. That is, there is a computable function ψ such that whenever $n \geq \psi(k)$, we have $\mu U_n \leq 2^{-k}$.*

We review the Wiener measure W on $\Omega = C[0,1]$. It is such that for $\omega \in \Omega$, and $t_0 < t_1 < \cdots < t_n$, the values of $\omega(t_0)$ and $\omega(t_{i+1} - t_i)$ are independent random variables. Moreover, the probability that $\omega(s+t) - \omega(s) \in A$, where A is some set of reals, is $\int_A (2\pi t)^{-1/2} \exp(-x^2/2t)dx$. This says that $\omega(t)$ is normally distributed with standard deviation $\sqrt{t}$ (variance t) and mean 0. Informally, a sufficiently random member of Ω with respect to W is called a path of Brownian motion.

Let $\overline{G}$ denotes the closure of G, and let G^0 denotes the complement of G; moreover, $O_\epsilon(G)$ is the open ϵ-ball around G.

Definition 2 (Fouché [3]). *A sequence $\mathscr{F}_0 = (F_i : i < \omega)$ of Borel subsets of Ω is a t-effective generating sequence if*

(1) for $F \in \mathscr{F}_0$, for $\epsilon > 0$ and $\delta \in \{0,1\}$, if $G = O_\epsilon(F^\delta)$ or if $G = F^\delta$, then $W(\overline{G}) = W(G)$;
(2) there is a t-effective procedure that yields, for each sequence $0 \leq i_1 < \cdots < i_n < \omega$ and $k < \omega$, a binary number β_k such that $|W(\bigcap_{1 \leq k \leq n} F_{i_k}) - \beta_k| < 2^{-k}$; and
(3) for $n, i < \omega$, for rational numbers $\epsilon > 0$ and for $x \in C_n$, both the relations $x \in O_\epsilon(F_i)$ and $x \in O_\epsilon(F_i^0)$ are t-recursive in x, ϵ, i and n.

A t-effectively generated algebra is the Boolean algebra generated from an t-effective generating sequence.

A set $A \subset C[0,1]$ is of t-constructive measure 0 if, for some t-effectively generated algebra $\mathscr{F}$, A is a t-effective $\mathscr{F}$-null set.

If ω belongs to no t-effective $\mathscr{F}$-null set, then we say that ω is t-$\mathscr{F}$-random or $\mathscr{F}$-random relative to t. (This also applies when $\mathscr{F}$ is not effective.) If t is computable then we may omit mention of t.

Finally, if there exists a t such that $\mathscr{F}_0$ is a t-effective generating sequence, then $\mathscr{F}_0$ is called a generating sequence. *The algebra it generates is similarly called a* generated algebra.

The precise definition of complex oscillations is immaterial to the present paper, but we include it for completeness.

Definition 3. *For $n \geq 1$, we write C_n for the class of continuous functions on $[0,1]$ that vanish at 0 and are linear with slope $\pm\sqrt{n}$ on the intervals $[(i-1)/n, i/n]$, $i = 1, \ldots, n$.*

To every $x \in C_n$ one can associate a binary string in $\{1, -1\}^*$, $a_1 \cdots a_n$, of length n by setting $a_i = 1$ or $a_i = -1$ according to whether x increases or *decreases on the interval* $[(i-1)/n, i/n]$. *We call the word* $a_1 \cdots a_n$ *the* code *of* x *and denote it by* $c(x)$.

A sequence $\{x_n\}_{n\in\mathbb{N}}$ *in* $C[0,1]$ *is* complex *if* $x_n \in C_n$ *for each* n *and there is some constant* $d > 0$ *such that* $K(c(x_n)) \geq n - d$ *for all* n*, where* K *denotes prefix-free Kolmogorov complexity.*

A function $x \in C[0,1]$ *is a complex oscillation if there is a complex sequence* $\{x_n\}_{n\in\mathbb{N}}$ *such that* $x_n - x$ *converges effectively to* 0 *as* $n \to \infty$*, in the uniform norm.*

The only use we make of Definition 3 is to quote the following result.

Theorem 1 (Fouché [3]; see also [4]). *Each complex oscillation is in the complement of each set of constructive measure 0.*

Let $LIL(\omega, t)$ be the statement that

$$\limsup_{h\to 0} \frac{|\omega(t+h) - \omega(t)|}{\sqrt{2|h| \log\log(1/|h|)}} = 1.$$

Thus $LIL(\omega, t)$ says that Khintchine's Law of the Iterated Logarithm holds for ω at t.

Theorem 2 (following Fouché [5]). *If* $t \in [0,1]$*, and* f *is* t*-*$\mathscr{F}$*-random for each* t*-effectively generated algebra* $\mathscr{F}$*, then the Law of the Iterated Logarithm holds for* f *at* t*.*

The proof is a straightforward relativization to t of Fouché's argument (which covers the case where t is computable).

2 Isomorphism Theorem

Let $\mathscr{A}_0$ be a generating sequence. Write $\mathscr{A}_0 = \{A_n\}_{n\in\mathbb{N}}$.

- Let $\mathfrak{A}_n$ be the Boolean algebra generated by $\{A_1, \ldots, A_n\}$.
- Let $\mathscr{A} = \mathfrak{A}_\infty = \bigcup_n \mathfrak{A}_n$, the Boolean algebra generated by $\mathscr{A}_0$.
- Let $\mathfrak{I}$ be the Boolean algebra of finite unions of half-open intervals $[a, b)$ in $[0, 1)$.

A *Boolean measure algebra homomorphism* is a map that preserves measure, unions, and complements.

Theorem 3 (Wiener, Carathéodory). *There is a Boolean measure algebra homomorphism* $\Phi : \mathscr{A} \to \mathfrak{I}$*.*

Proof. To start, since $\mathfrak{A}_1 = \{\emptyset, A_1, A_1', \Omega\}$ and $\mu A_1 + \mu A_1' = \mu\Omega = 1$, we let $\Phi(A_1) = [0, \mu A_1)$, $\Phi(A_1') = [\mu A_1, 1)$, $\Phi(\emptyset) = \emptyset$, and $\Phi(\Omega) = [0, 1)$. Then Φ is clearly a Boolean measure algebra homomorphism from $\mathfrak{A}_1$ into $\mathfrak{I}$.

Suppose now that Φ has been defined on $\mathfrak{A}_{n-1}$ so that it is a Boolean measure algebra homomorphism from $\mathfrak{A}_{n-1}$ onto the algebra generated by $k \in \mathbb{N}$ many half open intervals $[x_0, x_1), [x_1, x_2), \ldots, [x_{k-1}, x_k)$, where $x_0 = 0$ and $x_k = 1$.

We wish to extend the mapping Φ to $\mathfrak{A}_n$. Let B_i be the set in $\mathfrak{A}_{n-1}$ which is mapped onto the interval $[x_j, x_{j+1})$, for $j < k$. Then $\mathfrak{A}_{n-1}$ consists of all finite unions of the sets B_j, $j < k$, and $\mathfrak{A}_n$ consists of all finite unions from the $2k$ sets $A_n \cap B_j$, $A_n^0 \cap B_j$, $j < k$. Let

$$\Phi(A_n \cap B_j) = [x_j, x_j + \mu(A_n \cap B_j))$$
$$\Phi(A_n^0 \cap B_j) = [x_j + \mu(A_n \cap B_j), x_{j+1})$$

This might define Φ of some sets to be of the form $[x_j, x_j) = \emptyset$.

Clearly Φ as so defined is measure-preserving, $\Phi(A_n \cap B_j) \cup \Phi(A'_n \cap B_j) = [x_j, x_{j+1}) = \Phi(B_j)$, and $\mu(A_n \cap B_j) + \mu(A'_n \cap B_j) = \mu(B_j) = x_{j+1} - x_j$. From this it follows that we can extend Φ to all of $\mathfrak{A}_n$ so that it is a Boolean measure algebra homomorphism. Since $\mathfrak{A}_\infty = \bigcup_n \mathfrak{A}_n$, we have thus defined Φ on all of $\mathfrak{A}_\infty$.

Remark 1. The function Φ is effective in the following sense: if $\mathscr{F} = \{T_k : k \in \mathbb{N}\}$ is a t-effectively generated algebra, then the measure of $\Phi(T_k)$ can be computed t-effectively, uniformly in k.

Lemma 1. *Suppose $\mathcal{I}_n = (a_n, b_n)$, $n \in \mathbb{N}$, is a sequence of open intervals with $(a_{n+1}, b_{n+1}) \subseteq (a_n, b_n)$. Suppose $\bigcap_n (a_n, b_n) = \emptyset$. Then either $\{a_n\}_{n\in\mathbb{N}}$ or $\{b_n\}_{n\in\mathbb{N}}$ is an eventually constant sequence.*

The proof is routine. The set of Martin-Löf real numbers in $[0, 1]$ is denoted RAND, and relativized to t, RAND^t.

An effectively generated algebra $\mathscr{F} = \{T_k : k \in \mathbb{N}\}$ is *non-atomic* if for any $b : \mathbb{N} \to \{0, 1\}$, we have $W(\bigcap_k T_k^{b(k)}) = 0$. Here $T_k^1 = T_k$ and as before T_k^0 is the complement of T_k.

Lemma 2. *Let $t \in [0, 1]$ and let $\mathscr{F} = \{T_k : k \in \mathbb{N}\}$ be a non-atomic, t-effectively generated algebra. Let a function φ from $C[0, 1]$ to 2^ω be defined by: $\varphi(\omega) =$ the unique member of $\cap\{\Phi(T_k) : \omega \in T_k\}$, if it exists.*

(1) The domain of φ includes all t-$\mathscr{F}$-randoms.
(2) If $\varphi(\omega)$ is defined then for each k,

$$\omega \in T_k \leftrightarrow \varphi(\omega) \in \Phi(T_k).$$

Proof. (1): Suppose ω is not in the domain of φ. That is, $S = \cap\{\Phi(T_k) : \omega \in T_k\}$ does not have a unique element. It is clear that S is an interval. Since $\mathscr{F}$ is non-atomic, this interval must have measure zero. Thus, since S does not have exactly one element, S must be empty.

By Remark 1 and Lemma 1, there is a t-computable point a or b such that $a_n \to a$ or $b_n \to b$, where $(a_n, b_n) = \cap_{k\le n}\Phi(T_k)$. Using this point a or b one can t-effectively determine whether $\omega \in T_k$, given any $k \in \mathbb{N}$. Thus ω is not t-$\mathscr{F}$-random.

(2)$\leftarrow$: Since $\{T_k\}_{k\in\mathbb{N}}$ is a Boolean algebra and so closed under complements.

(2)$\rightarrow$: By definition of φ.

2.1 Effectiveness Lemmas

A *presentation* of a real number a is a sequence of open intervals I_n with rational endpoints, containing a, such that I_n has diameter $\leq 2^{-n}$.

Lemma 3. *There is a Turing machine which, given a presentation of $a = a_0 \oplus a_1$ as oracle, terminates iff $a_0 < a_1$.*

Proof (Proof sketch.). The presentation provides some information about a_0 and a_1. The machine terminates once enough information has been found to conclude that $a_0 < a_1$.

On the other hand, it is well known that if $a = b$ then no algorithm will be able to verify this in general. For intervals (a, b), (c, d), we say (a, b) is *bi-properly* contained in (c, d) if $c < a \leq b < d$.

Lemma 4. *(1) The set of pairs σ, k such that $\Phi(T_k)$ is bi-properly contained in $i([\sigma])$, is computably enumerable.*
(2) The set of pairs σ, k such that $i([\sigma])$ is bi-properly contained in $\Phi(T_k)$ is computably enumerable.

Proof. The endpoints of $\Phi(T_k)$ and $i([\sigma])$ have computable presentations. Thus the result follows from Lemma 3.

Lemma 5. *Let $t \in [0, 1]$ and let $\mathscr{F} = \{T_k : k \in \mathbb{N}\}$ be a t-effectively generated algebra.*

(1) If $\mathscr{F}$ is non-atomic, then for each t-ML-test $\{U_n\}_{n\in\mathbb{N}}$ there is a t-computable function $f : \mathbb{N}^2 \to \mathbb{N}$ such that

$$U_n \cap RAND = \bigcup_m \Phi(T_{f(n,m)}) \cap RAND.$$

(2) If for a t-computable function $f : \mathbb{N}^2 \to \mathbb{N}$, we have $U_n = \bigcup_m \Phi(T_{f(n,m)})$ then U_n has a subset U'_n such that $U_n \cap RAND = U'_n \cap RAND$ and $\{U'_n\}_{n\in\mathbb{N}}$ is uniformly $\Sigma^0_1(t)$.

Proof. (1): We can enumerate the cones $[\sigma]$ contained in U_n. Once we see some $[\sigma]$ get enumerated and then see (using Lemma 4(1)) that some $\Phi(T_k)$ is bi-properly contained in $[\sigma]$, we can enumerate $\Phi(T_k)$. Since $\mathscr{F}$ is non-atomic, we will gradually enumerate all of $[\sigma]$ except for possibly one or more of its computable endpoints.

(2): By Lemma 4(2). Given an enumeration of the sets $\Phi(T_{f(n,m)})$, $m \in \mathbb{N}$, we can enumerate sets $[\sigma_{n,m,p}]$ that are bi-properly contained in $\Phi(T_{f(n,m)})$ to ensure that $\bigcup_m \Phi(T_{f(n,m)} \cap \text{RAND} = \bigcup_{p,m}[\sigma_{n,m,p}] \cap \text{RAND}$. The endpoints of $\Phi(T_k)$ are computable and hence not in RAND.

The only result of this section that will be used in the next is the following:

Theorem 4. *Let $\omega \in \Omega$, $t \in [0,1]$, and let $\mathscr{F}_0 = \{T_k : k \in \mathbb{N}\}$ be a non-atomic t-effective generating sequence, and $\mathscr{F}$ its generated algebra. The following are equivalent:*

(1) ω is t-$\mathscr{F}$-random;
(2) $\varphi(\omega) \in RAND^t$.

Proof. (2) implies (1): Suppose ω is not t-$\mathscr{F}$-random, so $\omega \in \bigcap_n V_n$, a t-$\mathscr{F}$-null set. Then $V_n = \bigcup_m T_{f(n,m)}$ for some t-computable f.

Let $U_n = \bigcup_m \Phi(T_{f(n,m)})$. Note
(a) U_n is uniformly $\Sigma_1^0(t)$ by Lemma 5(2).
(b) Since Φ is measure preserving on $\mathscr{F}$ and is a Boolean algebra homomorphism by Theorem 3,

$$\mu(\cup_{i=1}^n \Phi(T_{a_i})) = \mu(\Phi(\cup_{i=1}^n T_{a_i})) = \mu(\cup_{i=1}^n T_{a_i})$$

Since the measure of a countable union is the limit of the measures of finite unions, $\mu U_n = \mu V_n \leq 2^{-n}$.
By (a) and (b), $\{U_n\}_{n\in\omega}$ is a t-ML-test.

If ω is not in the domain of φ then ω is not $\mathscr{F}$-random, by Lemma 2(1); so we may assume $\varphi(\omega)$ exists. Hence, since $\omega \in \bigcap_n V_n$, by definition of φ, we have $\varphi(\omega) \in \bigcap_n U_n$. Thus $\varphi(\omega) \notin \text{RAND}^t$.

(1) implies (2): Suppose $\varphi(\omega)$ is not 1-t-random, so $\varphi(\omega) \in \bigcap_n U_n$, for some t-Martin-Löf test $\{U_n\}_{n\in\mathbb{N}}$. Let $V_n := \cup_m T_{f(n,m)}$ with f as in Lemma 5. So by its definition, V_n is uniformly $\Sigma_1^t(\mathscr{F}_0)$. As in the proof that (2) implies (1), V_n and U_n have the same measure. Since $\varphi(\omega) \in \bigcap_n U_n$, by Lemma 2(2) we have $\omega \in \bigcap_n V_n$.

Remark 2. Our identification of binary sequences with numbers in $[0,1]$ deserves some comment. A number $t \in [0,1]$ is a dyadic rational if it is of the form $\frac{p}{2^n}$, for $p, n \in \mathbb{N}$; otherwise, t is called a dyadic irrational. The set of dyadic irrationals can be identified with a full-measure subset of $2^{\mathbb{N}}$ via the map ι such that $\iota(\sum_{i\geq 1} b_i 2^{-i}) = \{b_i\}_{i\geq 1}$. This also gives an identification of cones $[\sigma] = \{A \in 2^{\mathbb{N}} : \forall n < |\sigma|\ A(n) = \sigma(n)\}$ for $\sigma \in \{0,1\}^*$ with intervals in the dyardic irrationals. Formally, we can let $i(2^\omega) = (0,1)$ and if $i([\sigma]) = (a,b)$ then $i([\sigma 0]) = (a, a+(b-a)/2)$ and $i([\sigma 1] = (a+(b-a)/2, b)$, but we leave i implicit. We only need φ restricted to the $\mathscr{F}$-random functions $\omega \in \Omega$. By Theorem 4, φ maps such functions ω into RAND and in particular into the dyadic irrationals.

3 Khintchine's Law for Complex Oscillations

It is common in probability theory to write, for $\omega \in \Omega$ and $x \in [0,1]$, $B_x(\omega) = \omega(x)$. This allows us to refer to the set $\{\omega \in \Omega : \omega(x) < y\}$, for example (where x, y are fixed rational numbers) as the event that $B_x < y$, and as a set this is written $[B_x < y]$. In words, the value of the Brownian motion at time x is less than y.

Let, for each $t \in [0,1]$, $\mathscr{F}_t$ be an t-effectively generated algebra that contains the one used in Theorem 2, and that moreover is non-atomic. The latter is achieved by including all events of the form $[B_x < y]$ for rational $x \in [0,1]$ and arbitrary rational y. Note that if $\mathscr{F}$ and $\mathscr{F}'$ are effectively generated algebras, and $\mathscr{F} \subseteq \mathscr{F}'$, then each $\mathscr{F}'$-random function $\omega \in \Omega$ is also $\mathscr{F}$-random, since adding elements to an effective generating sequence only adds new effective null sets.

Lemma 6. *For each $t \in [0,1]$ and each ω with $\varphi(\omega) \in RAND^t$, we have $LIL(\omega, t)$. In particular, for each $t \in RAND$ and each ω with $\varphi(\omega) \in RAND^t$, we have $LIL(\omega, t)$.*

Proof. Suppose $t \in [0,1]$ and $\varphi(\omega) \in \mathrm{RAND}^t$. By Theorem 4, ω belongs to no t-effective $\mathscr{F}_t$-null set. Hence by Theorem 2, $LIL(\omega, t)$.

The point now is that in the image of φ, we already know more of what is going on. Let $A \oplus B = \{2n : n \in A\} \cup \{2n+1 : n \in B\}$, for reals A, B (equivalently, $A, B \subseteq \mathbb{N}$).

Theorem 5 (van Lambalgen's Theorem). *Let A, B be reals. The following are equivalent.*

- *$A \in RAND$ and $B \in RAND^A$;*
- *$A \oplus B \in RAND$;*
- *$B \in RAND$ and $A \in RAND^B$.*

We can now approach our desired result:

Lemma 7. *If $\varphi(\omega) \in RAND$ and $t \in RAND^{\varphi(\omega)}$ then $LIL(\omega, t)$.*

Proof. Suppose $\varphi(\omega) \in \mathrm{RAND}$ and $t \in \mathrm{RAND}^{\varphi(\omega)}$. By Theorem 5 with $A = \varphi(\omega)$ and $B = t$, we have that $t \in \mathrm{RAND}$ and $\varphi(\omega) \in \mathrm{RAND}^t$. Hence by Lemma 6, we have $LIL(\omega, t)$.

Theorem 6. *If ω is a complex oscillation, then for almost all t, $LIL(\omega, t)$.*

Proof. Suppose ω is a complex oscillation. By Theorem 4, $\varphi(\omega) \in \mathrm{RAND}$. By Lemma 7, $LIL(\omega, t)$ holds for each $t \in \mathrm{RAND}^{\varphi(\omega)}$. Since RAND^A has measure 1 for each $A \in 2^\omega$, we are done.

Finally, we remark that our main result can be extended from Martin-Löf randomness to Schnorr randomness, using a weak version of van Lambalgen's theorem that holds for Schnorr randomness.

References

1. E.A. Asarin and A.V. Pokrovskii, Use of the Kolmogorov complexity in analyzing control system dynamics, *Automation and Remote Control* **47**, 21-28 (1986).
2. R.G. Downey and D.R. Hirschfeldt, *Algorithmic Randomness and Complexity*, to appear. Available on the home page of the first author: http://www.mcs.vuw.ac.nz/people/Rod-Downey

3. W. Fouché, Arithmetic Representations of Brownian Motion I, *J. Symbolic Logic* **65** No. 1 (2000), 421-442.
4. W. Fouché, The descriptive complexity of Brownian motion, *Adv.Math.* **155** (2000), no. 2, 317–343.
5. W. Fouché, Dynamics of a Generic Brownian Motion: Recursive Aspects. To appear in A. Beckmann, E. Beggs, B. Löwe (eds.), *From Gödel to Einstein: Computability between Logic and Physics at CiE 2006.* Special issue of the journal *Theoretical Computer Science A*, 2007.
6. H.L. Royden, *Real Analysis*, third edition, Prentice-Hall, 1988.

Hypersequent Calculus for Intuitionistic Logic with Classical Atoms

Hidenori Kurokawa

Department of Philosophy
The City University of New York, Graduate Center
365 Fifth Avenue New York, NY 10016
hkurokawa@gc.cuny.edu

Abstract. We introduce a hypersequent calculus for intuitionistic logic with classical atoms, i.e. intuitionistic logic augmented with a special class of propositional variables for which we postulate the decidability property. This system combines classical logical reasoning with constructive and computationally oriented intuitionistic logic in one system. Our main result is the cut-elimination theorem with the subformula property for this system. We show this by a semantic method, namely via proving the completeness theorem of the hypersequent calculus without the cut rule. The cut-elimination theorem gives a semantic completeness of the system, decidability, and some form of the disjunction property.

1 Introduction

Combining logics is a widely discussed topic in logical studies these days, and combining intuitionistic logic and classical logic is no exception. Several ways of combining the two logics have been proposed. One way is to introduce two (intuitionistic and classical) implications or negations in one system, [10], [6] and [13], and another is to introduce a two-sorted language of propositional logic with two different kinds (intuitionistic and classical) of propositional variables, and the law of excluded middle is postulated only for classical variables, [16], [18], [17] and [12].[1] Combining intuitionistic and classical logic in the second way above can be motivated by the following consideration: even in constructive mathematics some formulas are decidable, so we may need some logic that can have both decidable propositions and not necessarily decidable ones anyway. Also, our two-sorted approach has some connection to the proof theory of basic intuitionistic logic of proofs ($iBLP$) [1]. $iBLP$ has formulas of the form "$x : F$" that read "x is a proof of F," which are decidable.

In this paper, we present a Gentzen-style sequent calculus in which we can combine intuitionistic propositional logic (IPC) and classical propositional logic

[1] Strictly speaking, [16] does not discuss a two-sorted language, but it is obvious that we can expand the language of their logic to a two-sorted one to obtain a combined system for IPC and CPC. Also, there are yet other ways of combining logics. See [9],[14] and [8].

(CPC) in the second way. It is not entirely trivial to formulate such a system especially if you want to formulate a system where cut-elimination and the full subformula property holds. We extend the framework of sequent calculus to a hypersequent calculus, which has already been used in many contexts in non-classical logics. We show cut-elimination by a semantical method and some other properties of the hypersequent system for IPC$_{CA}$.

2 Hypersequent Calculus

First, we give a specification of our language and fix some notational conventions. The language of IPC$_{CA}$, $\mathcal{L}_{IPC_{CA}}$, consists of the usual intuitionistic propositional connectives and two sets of propositional variables: intuitionistic propositional variables, $Var_I := \{p_1, \ldots, p_n, ...\}$, and classical propositional variables $Var_C := \{X_1, \ldots, X_n, \ldots\}$. The latter variables will satisfy an additional constraint that will provide their classical behavior. A formula F in $\mathcal{L}_{IPC_{CA}}$ is specified as follows.[2] $F ::= p_i|X_i|\bot|F_1 \to F_2|F_1 \wedge F_2|F_1 \vee F_2$.

We use the following notational convention: 1) B° is a formula containing only classical variables. 2) A,B,C,... (without any extra symbol) can be any formula.

We adopt a multi-conclusion intuitionistic sequent calculus where only $R \to$ lacks symmetry. We assume that our sequents are sets of formulas and our hypersequents are multisets of sequents. Here is the system of mLIC (multi-conclusion logical calculus for intuitionistic logic with classical atoms).

1) **Axioms**: $A \Rightarrow A$ $\quad \bot\Rightarrow$

2) **External structural rules**:

EW $\dfrac{G}{G|H}$ **EC** $\dfrac{G|\Gamma \Rightarrow \Delta|\Gamma \Rightarrow \Delta|H}{G|\Gamma \Rightarrow \Delta|H}$

3) **Internal structural rules**:

LW $\dfrac{G|\Gamma \Rightarrow \Delta}{G|A, \Gamma \Rightarrow \Delta}$ **RW** $\dfrac{G|\Gamma \Rightarrow \Delta}{G|\Gamma \Rightarrow \Delta, A}$

4)**Logical rules**

L$\wedge$ $\dfrac{G|A, B, \Gamma \Rightarrow \Delta}{G|A \wedge B, \Gamma \Rightarrow \Delta}$ **R**$\wedge$ $\dfrac{G|\Gamma \Rightarrow \Delta, A \qquad G|\Gamma \Rightarrow \Delta, B}{G|\Gamma \Rightarrow \Delta, A \wedge B}$

L$\vee$ $\dfrac{G|A, \Gamma \Rightarrow \Delta \qquad G|B, \Gamma \Rightarrow \Delta}{G|A \vee B, \Gamma \Rightarrow \Delta}$ **R**$\vee$ $\dfrac{G|\Gamma \Rightarrow \Delta, A, B}{G|\Gamma \Rightarrow \Delta, A \vee B}$

L $\to$ $\dfrac{G|\Gamma \Rightarrow \Delta, A \quad G|B, \Gamma \Rightarrow \Delta}{G|A \to B, \Gamma \Rightarrow \Delta}$ **R** $\to$ $\dfrac{G|A, \Gamma \Rightarrow B}{G|\Gamma \Rightarrow A \to B, \Delta}$

[2] We take $\neg\varphi$ as an abbreviation of $\varphi \to \bot$.

5) **Classical Splitting** $$\frac{G|\Gamma_1, \Gamma_2^\circ \Rightarrow \Delta_1, \Delta_2^\circ|H}{G|\Gamma_1 \Rightarrow \Delta_1|\Gamma_2^\circ \Rightarrow \Delta_2^\circ|H}$$

6) **Cut** $$\frac{G_1|\Gamma_1 \Rightarrow \Delta_1, A|H_1 \qquad G_2|A, \Gamma_2 \Rightarrow \Delta_2|H_2}{G_1|G_2|\Gamma_1, \Gamma_2 \Rightarrow \Delta_1, \Delta_2|H_1|H_2}$$

Remark 1. Hypersequents are introduced by Avron to formulate cut-free sequent calculi for various non-classical logics, in particular Gödel-Dummett logic.

2. This Splitting would give classical logic if we did not have any restriction of the language of formulas. (See [4].) Without Classical Splitting, the hypersequent calculus would be a hypersequent for IPC in $\mathcal{L}_{IPC_{CA}}$ ([5]).

3. $(X \to p) \vee (p \to X)$ is valid w.r.t. the class of Kripke models for IPC$_{CA}$ (cf.[11]). However, it seems difficult to add a rule to an ordinary cut-free sequent calculus so that the above formula can be derived in it. Due to Classical Splitting, we have a cut-free proof of the formula in mLIC.

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{X \Rightarrow X}{X \Rightarrow \,|\, \Rightarrow X}\ \text{Classical Splitting}}{X \Rightarrow p|p \Rightarrow X}\ LW,\ RW}{\Rightarrow X \to p, p \to X| \Rightarrow X \to p, p \to X}\ R\to}{\Rightarrow (X \to p) \vee (p \to X)| \Rightarrow (X \to p) \vee (p \to X)}\ R\vee}{\Rightarrow (X \to p) \vee (p \to X)}\ EC$$

3 Kripke Models for Intuitionistic Logic with Classical Atoms

Now we state the main theorem. mLIC$^-$ means mLIC without Cut.

Theorem 1. *If mLIC* $\vdash \Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_n \Rightarrow \Delta_n$*, then mLIC*$^-$ $\vdash \Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_n \Rightarrow \Delta_n$.

We show this theorem by a semantic method, using finite tree Kripke models for IPC$_{CA}$. A Kripke model $\mathcal{K}$ for the language of IPC$_{CA}$ is defined as an ordered triple $(K, \leq, \Vdash)$, where K is a nonempty set and called a set of states, $\leq$ is a partial order of the states, and $\Vdash$ is a forcing relation. The ordered pair of the first two components $(K, \leq)$ is called a Kripke frame. A forcing relation satisfies the condition of "monotonicity" propositional variables: for any $s, t \in K$ and for any propositional variable p, if $s \leq t$ and $s \Vdash p$, then $t \Vdash p$.

Also, $\Vdash$ satisfies the standard inductive clauses for logical connectives $\to$, $\wedge$,$\vee$ for intuitionistic logic.[3] Without loss of generality, we only think about finite tree Kripke models. In addition to these, we have the following condition.

The new condition for classical atoms (Stability) : Let s_0 be the root node of a finite Kripke tree model. For each $X_i \in Var_C$, one of the following holds: $s_j \Vdash X_i$ for all $s_j \geq s_0$ or $s_j \nVdash X_i$ for all $s_j \geq s_0$

[3] We have $\bot$ in the language, and $\bot$ is never forced at any state.

Here $\mathcal{K}, s \Vdash \psi$ means that a formula ψ is forced at state s in a Kripke model $\mathcal{K}$. Also, $\mathcal{K} \Vdash \psi$ means that a formula ψ is valid in $\mathcal{K}$, which means that ψ is forced in all the states in $\mathcal{K}$. A formula ψ is "valid" if it is valid in all Kripke models. We extend our forcing relation and the notion of validity to hypersequents.

Definition 1. *1. $\mathcal{K}, s \nVdash \Gamma \Rightarrow \Delta$, if $\exists s' \in K, s' \geq s$, s.t. for any $\varphi \in \Gamma$, $\mathcal{K}, s' \Vdash \varphi$ and for any $\psi \in \Delta$, $\mathcal{K}, s' \nVdash \psi$. 2. $\mathcal{K}, s \Vdash \Gamma \Rightarrow \Delta$, otherwise.*

Definition 2. *$\mathcal{K}, s \Vdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ iff $\mathcal{K}, s \Vdash \Gamma_1 \Rightarrow \Delta_1$ or ... or $\mathcal{K}, s \Vdash \Gamma_n \Rightarrow \Delta_n$. Also, a hypersequent $G = \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ is valid if for any $\mathcal{K}$ and any $s \in K$, $\mathcal{K}, s \Vdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$.*

4 The Completeness Theorem for the Multi-conclusion Hypersequent Calculus Without the Cut-Rule

Cut-elimination is obtained as a consequence of the following theorem.

Theorem 2. *$mLIC^- \vdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ if $\Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ is valid in any Kripke model of mLIC.*

To show cut-elimination, we also need to show the soundness theorem "mLIC$\vdash$ $\Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ only if $\Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ is valid." However, since we already proved that Hilbert-style system of IPC$_{CA}$ is sound with respect to the relevant class of Kripke models in [11], our proof of soundness is done, provided that the provability of a hypersequent in mLIC implies that of the translated formula in IPC$_{CA}$.[4] Now we prove the contrapositive of the completeness theorem of hypersequent calculus mLIC$^-$.

4.1 Saturation Lemma

Definition 3. *A saturated hypersequent $G' = \Gamma_1' \Rightarrow \Delta_1' | \ldots | \Gamma_n' \Rightarrow \Delta_n'$ for $mLIC^-$ is a hypersequent satisfying the following conditions.*[5]

1. *For any component $\Gamma_i' \Rightarrow \Delta_i'$ of G', if $A \vee B \in \Gamma_i'$, then $A \in \Gamma_i'$ or $B \in \Gamma_i'$.*
2. *For any component $\Gamma_i' \Rightarrow \Delta_i'$ of G', if $A \vee B \in \Delta_i'$, then $A \in \Delta_i'$ and $B \in \Delta_i'$.*
3. *For any component $\Gamma_i' \Rightarrow \Delta_i'$ of G', if $A \wedge B \in \Gamma_i'$, then $A \in \Gamma_i'$ and $B \in \Gamma_i'$.*

[4] This fact, i.e. the translation is sound, has to be shown. The proof is given by induction on the length of proof. The crucial case is Classical Splitting. We have to show that the soundness of the translation has to be preserved under the application of the rule. Assuming that IPC$_{CA}\vdash (\bigwedge \Gamma_1 \wedge \bigwedge \Gamma_2^\circ) \rightarrow (\bigvee \Delta_1 \vee \bigvee \Delta_2^\circ)$, we want to show that IPC$_{CA}\vdash (\bigwedge \Gamma_1 \rightarrow \bigvee \Delta_1) \vee (\bigwedge \Gamma_2^\circ \rightarrow \bigvee \Delta_2^\circ)$. The inference here is obviously valid with respect to the semantics of IPC$_{CA}$, so by soundness and completeness of IPC$_{CA}$, the proof is essentially done and the translation is sound.

[5] In general, we call each sequent $\Gamma_i \Rightarrow \Delta_i$ of a hypersequent G a *component* of G. Here i is an index for a component in a hypersequent. $'$ means that the sequent is saturated.

4. For any component $\Gamma'_i \Rightarrow \Delta'_i$ of G', if $A \wedge B \in \Delta'_i$, then $A \in \Delta'_i$ or $B \in \Delta'_i$.
5. For any component $\Gamma'_i \Rightarrow \Delta'_i$ of G', if $A \to B \in \Gamma'_i$, then $A \in \Delta'_i$ or $B \in \Gamma'_i$.
6. For any component $\Gamma'_i \Rightarrow \Delta'_i$ of G', if $A \to B \in \Delta'_i$, then there exists a component $\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}$ of G', s.t. $A \in \Gamma'_{i\sigma}$ and $B \in \Delta'_{i\sigma}$.[6]

Also, if the conditions 1-5 are satisfied for a component, we call the component "saturated component."[7]

Definition 4. *A hypersequent $\Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_i \Rightarrow \Delta_i|\Gamma_i \cup \{A\} \Rightarrow B|\ldots|\Gamma_n \Rightarrow \Delta_n$is an associated hypersequent of a hypersequent$\Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_i \Rightarrow \Delta_i|\ldots|\Gamma_n \Rightarrow \Delta_n$ with respect to $\Gamma_i \Rightarrow \Delta_i$, s.t. $A \to B \in \Delta_i$. Also, we call this $\Gamma_i \cup \{A\} \Rightarrow B$ an associated component of $\Gamma_i \Rightarrow \Delta_i$, s.t. $A \to B \in \Delta_i$.*

Lemma 1 (Saturation Lemma)
For any hypersequent $G = \Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_n \Rightarrow \Delta_n$, s.t. $mLIC^- \nvdash \Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_n \Rightarrow \Delta_n$, there exists a saturated hypersequent[8] *$G' = \Gamma'_1 \Rightarrow \Delta'_1|\ldots|\Gamma'_{n\sigma_n} \Rightarrow \Delta'_{n\sigma_n}$ satisfying the following conditions.*

(α) For each component $\Gamma_i \Rightarrow \Delta_i$ and its associated components $\Gamma_{i\sigma} \Rightarrow \Delta_{i\sigma}$,[9]
(α_1) $\Gamma_{i\sigma} \subseteq \Gamma'_{i\sigma}$; ($\alpha_2$) $\Delta_{i\sigma} \subseteq \Delta'_{i\sigma}$; ($\alpha_3$) $\Gamma'_{i\sigma} \cap \Delta'_{i\sigma} = \emptyset$ and $\bot \notin \Gamma'_{i\sigma}$.
(β) Let P and N be the following.
$P := \bigcup\{\Gamma'_{i\sigma}|(\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma})$ is a component of saturated hypersequent $G'\}$
$N := \bigcup\{\Delta'_{i\sigma}|(\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma})$ is a component of saturated hypersequent $G'\}$.
Then $\{X \in Var_C|X \in P\} \cap \{X \in Var_C|X \in N\} = \emptyset$.

Proof. The proof is done by describing the saturation procedure.

1) We start from the leftmost component $\Gamma_1 \Rightarrow \Delta_1$ of the original hypersequent $G = \Gamma_1 \Rightarrow \Delta_1|\ldots|\Gamma_n \Rightarrow \Delta_n$, and we continue to construct saturated components and associated components for "$\to$" formulas on the succedent of components until we develop all the saturated components. The following rules give us how to construct a saturated *component* of a saturated hypersequent. For $\Gamma_i \Rightarrow \Delta_i$ in G, go through all of the following steps.

1. If $A \vee B \in \Gamma_i$, then put A into Γ_i or put B into Γ_i.
2. If $A \vee B \in \Delta_i$ then put A into Δ_i and put B into Δ_i.
3. If $A \wedge B \in \Gamma_i$, then put A into Γ_i and put B into Γ_i.
4. If $A \wedge B \in \Delta_i$, then put A into Δ_i or put B into Δ_i.

[6] σ stands for a finite sequence of numbers. When we have an implication on the succedent, we will have a new sequent and we name it by an appropriate number. So, we have some sequence of number. The particular construction of a new component will be given below. σ's are convenient labels. The construction does not essentially depends on the labels.
[7] We call this "saturated component" regardless of whether it has $A \to B$ in Δ'_i or not.
[8] Here σ_n stands for a sequence for n-th component.
[9] We identify $\Gamma_i \Rightarrow \Delta_i$ with $\Gamma_{i0} \Rightarrow \Delta_{i0}$, and once saturated, it becomes $\Gamma'_{i0} \Rightarrow \Delta'_{i0}$. An index of an associated component starts with 1. Also, the notion of "associatedness" is transitive.

5. If $A \to B \in \varGamma_i$, then put A into $\varDelta_i$ or put B into $\varGamma_i$.
6. If $\varGamma_i \cap \varDelta_i \neq \emptyset$ or $\bot \in \varGamma_i$, then backtrack.

2) We go through the procedure until we can no longer parse any formula in the set,[10] except for implicational formulas on the succedent. If we obtain a sequent that satisfies all the conditions, then we can terminate.[11] If we have the case 6, then we backtrack. For $\varGamma_1 \Rightarrow \varDelta_1$, we may terminate with "success," i.e. terminate by constructing a saturated component without any application of 1-6 anymore, or terminate with "failure," i.e. terminate with no possibility of backtracking any more and have only the cases where $\varGamma_1' \cap \varDelta_1' \neq \emptyset$ or $\bot \in \varGamma_1'$. If $\varGamma_1' \Rightarrow \varDelta_1'$ does not have $A \to B$ in $\varDelta_1'$, then we are done with $\varGamma_1' \Rightarrow \varDelta_1'$.

3) If there is a saturated component $\varGamma_1' \Rightarrow \varDelta_1'$ for $\varGamma_1 \Rightarrow \varDelta_1$ satisfying all the conditions listed above and that $A_1 \to B_1, \ldots, A_k \to B_k \in \varDelta_1'$, then we construct k-many associated components for $\varGamma_1 \Rightarrow \varDelta_1$. Let $\varGamma_{1j} \Rightarrow \varDelta_{1j}$ be as $\varGamma_{1j} = \varGamma \cup \{A_j\}$ and $\varDelta_{1j} = \{B_j\}$ $(1 \leq j \leq k)$. So, our new hypersequent looks like $\varGamma_1' \Rightarrow \varDelta_1' | \varGamma_{11} \Rightarrow \varDelta_{11} | \ldots | \varGamma_{1k} \Rightarrow \varDelta_{1k} | \ldots | \varGamma_n \Rightarrow \varDelta_n$.

4) By going through from 1) to 3) for $\varGamma_{11} \Rightarrow \varDelta_{11}$, we get $\varGamma_{11}' \Rightarrow \varDelta_{11}'$. If there is any $A \to B \in \varDelta_{11}'$, then we have to construct an associated component $\varGamma_{111} \Rightarrow \varDelta_{111}$ (possibly $\varGamma_{112} \Rightarrow \varDelta_{112}, \ldots, \varGamma_{11l} \Rightarrow \varDelta_{11l}$) for $\varGamma_{11} \Rightarrow \varDelta_{11}$. We also saturate $\varGamma_{111} \Rightarrow \varDelta_{111}, \ldots, \varGamma_{11l} \Rightarrow \varDelta_{11l}$.[12] So, we get $\varGamma_1' \Rightarrow \varDelta_1' | \varGamma_{11}' \Rightarrow \varDelta_{11}' | \varGamma_{111}' \Rightarrow \varDelta_{111}' | \ldots | \varGamma_{1k}' \Rightarrow \varDelta_{1k}' | \varGamma_{1k1}' \Rightarrow \varDelta_{1k1}' | \ldots | \varGamma_n \Rightarrow \varDelta_n$.

5) We go through all the steps in 1)-4) until we finish saturating all the associated components for $\varGamma_n \Rightarrow \varDelta_n$. Since the number of $A \to B$ formulas on the succedents is finite, our procedure terminates in a finite number of steps.[13]

6) After we systematically construct all the saturated components (including all associated ones), we take the union of $\varGamma_{i\sigma}'$ and the union of $\varDelta_{i\sigma}'$ (P and N) and check whether the condition of disjointness of the classical variables (β) in P and N. If (β) is violated, then we backtrack. We have the two possible cases of termination: (1) We terminate with "success," i.e. by constructing a saturated component without violating (α_3) or (β); (2) We terminate with "failure," i.e. violating (α_3) or (β) with no case of backtracking.

If we have a successful case, then the other conditions of (α) are indeed satisfied with respect to any component $\varGamma_{i\sigma}' \Rightarrow \varDelta_{i\sigma}'$. In all the steps except the case $A \to B \in \varDelta_{i\tau}'$ we simply add some new formulas. So, once $\varGamma_{i\sigma} \Rightarrow \varDelta_{i\sigma}$ is constructed, we only add formulas on $\varGamma_{i\sigma}$ and $\varDelta_{i\sigma}$. Hence, the condition (α_1) and (α_2) of the lemma are satisfied for all the successful cases of components.

On the other hand, the case (2) is impossible.

[10] After a formula is used once, it becomes unavailable in one component.

[11] The procedure for the component $\varGamma_i \Rightarrow \varDelta_i$ will terminate because we only go through the subformulas of the component, and the complexity of them strictly goes down at each step.

[12] For any implication formula on any succedent, we unfold all associated components.

[13] We may have a repetition of having the same formula every time we construct a new associated sequent. But then we can stop whenever we get into the repetition of the same step.

Claim. If a component (or any pair of components) in a candidate of a saturated hypersequent G' violate(s) the condition (α_3) or the condition (β) of the disjointness of classical variables or both without any possibility of backtracking, then we can construct a cut-free proof of the hypersequent G.

Proof. Essentially by tracing the saturation procedure backwards (from the rightmost component to the leftmost one), we first construct a cut-free derivation of the original hypersequent G from all the cases of saturation in which the condition (α_3) or (β) is violated. Assume that we have violations of (α_3) or (β) with implications on the succedent developed and we have no case of backtracking.

We arrange all those possible alternatives of saturated hypersequents which violate the condition (α_3) or (β)[14] so that we can construct a derivation tree of G whose leaves are saturated hypersequents. We first construct a tree labeled by hypersequents (at this point, not necessarily a derivation tree yet) by the rules: 1) The root is the original hypersequent G; 2)-1. if a saturation step is deterministic, then put the resulting G_2 above G_1; 2)-2. if a saturation step is non-deterministic, then put the two alternatives G_2 and G_3 above the previous one G_1; 2)-3. if a saturation step is $A \to B \in \Delta'_{i\sigma}$,[15] then put the hypersequents as follows.

$$\frac{\Gamma'_1 \Rightarrow \Delta'_1 | \dots | \Gamma'_i \Rightarrow \Delta'_i | \Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma} | \Gamma'_{i\sigma 1}, A \Rightarrow B | \dots | \Gamma_n \Rightarrow \Delta_n}{\Gamma'_1 \Rightarrow \Delta'_1 | \dots | \Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma} | \dots | \Gamma_n \Rightarrow \Delta_n.}$$

3) Leaf nodes are saturated hypersequents with (α_3) or (β) violated.

Subclaim 1: With some minor modifications, our finite tree of saturated hypersequents becomes a derivation of the original hypersequent G from all the alternative saturated hypersequents with violation of either (α_3) or (β).

Proof. About 1) and 3): first, the construction traces saturation of G backwards, so we end with G. By assumption, all the topmost saturated hypersequents satisfy the following conditions: in case of (α_3), all such saturated hypersequents have a component $\Gamma'_{i\sigma}, A \Rightarrow \Delta'_{i\sigma}, A$ or $\Gamma'_{i\sigma}, \bot \Rightarrow \Delta'_{i\sigma}$; in case of (β) (possibly with (α_3), all such saturated hypersequents have at least one pair of components $\Gamma'_{i\sigma}, X \Rightarrow \Delta'_{i\sigma}$ and $\Gamma'_{j\tau} \Rightarrow \Delta'_{j\tau}, X$ (or a component $\Gamma'_{i\sigma}, A \Rightarrow \Delta'_{i\sigma}, A$ or $\Gamma'_{i\sigma}, \bot \Rightarrow \Delta'_{i\sigma}$).

About 2): (Outline) We show inductively (on the number of applications of logical rules) that the constructed tree with some modifications is the desired derivation. For 2)-1,2, by IH, we have a derivation up to G_2 (and G_3) from the failed cases of saturated hypersequents. We can take this as a derivation

[14] Even in a case with violation of (β), in some alternatives, there may be a violation of (α_3).

[15] If the succedent has more than one implication formula, then we have to deal with all of them by putting one new line of a hypersequent whenever we apply a case of $\to$ in $\Delta'_{i\sigma}$.

of the lower hypersequent G_1 from G_2 (and G_3), where a principal formula is obtained as a result of applying the rule. So, we have a desired derivation of G_1 from the failed saturated hypersequents. For 2)-3: ($A \to B \in \Delta'_{i\sigma}$) In this case, we need a slight modification. In our saturation, we put some new sequents adjacent to the component that has $\to$ on the succedent. To accommodate this, we insert one intermediate line that has another copy of $\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}$ to the tree. So,

$$\dfrac{\dfrac{\Gamma'_1 \Rightarrow \Delta'_1|\dots|\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}|\Gamma'_{i\sigma 1}, A \Rightarrow B|\dots|\Gamma_n \Rightarrow \Delta_n}{\Gamma'_1 \Rightarrow \Delta'_1|\dots|\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}|\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}|\dots|\Gamma_n \Rightarrow \Delta_n}\, R\to}{\Gamma'_1 \Rightarrow \Delta'_1|\dots|\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}|\dots|\Gamma_n \Rightarrow \Delta_n}\, \text{EC}$$

Here $\Gamma'_{i\sigma 1} = \Gamma'_{i\sigma}$, and by IH, we have a derivation up to the top line from the failed saturated hypersequents. Note that the first step is just $R \to$ and the second is EC. So, we have a derivation of the bottom line. ⊠ (subclaim 1)

Subclaim 2: Any hypersequent of the form $\Gamma_1 \Rightarrow \Delta_1|\dots|A, \Gamma_i \Rightarrow \Delta_i, A|\dots|\Gamma_n \Rightarrow \Delta_n$ or $\Gamma_1 \Rightarrow \Delta_1|\dots|\bot, \Gamma_i \Rightarrow \Delta_i, |\dots|\Gamma_n \Rightarrow \Delta_n$ is provable in mLIC$^-$.
Subclaim 3: Any hypersequent of the form $\Gamma_1 \Rightarrow \Delta_1|\dots|X, \Gamma_i \Rightarrow \Delta_i|\dots|\Gamma_j \Rightarrow \Delta_j, X|\Gamma_n \Rightarrow \Delta_n$ is provable in mLIC$^-$.

Proof. (For 2) The entire hypersequent can be taken as the result of applying LW, RW and EW to an axiom. (For 3) We can have the following proof.

$$\dfrac{\dfrac{\dfrac{X \Rightarrow X}{X \Rightarrow | \Rightarrow X}\,\text{Classical Splitting}}{X, \Gamma_i \Rightarrow \Delta_i|\Gamma_j \Rightarrow \Delta_j, X}\,\text{several LW or RW}}{\Gamma_1 \Rightarrow \Delta_1|\dots|X, \Gamma_i \Rightarrow \Delta_i|\dots|\Gamma_j \Rightarrow \Delta_j, X|\dots|\Gamma_n \Rightarrow \Delta_n}\,\text{several EW}$$

The failed saturated hypersequents have the form of the hypersequents in the above subclaims. So, by putting proofs in the subclaim 2, 3 on top of our cut-free derivation from saturated hypersequents, we can construct a cut-free *proof* of the original hypersequent G based on all the cases of violation of the condition (α_3) or (β) (possibly with those of (α_3), respectively. This completes transforming the cases of failure into a cut-free proof in mLIC$^-$. ⊠ (claim)

By the claim, the existence of a failed case (2) would be contradictory to the assumption of unprovability of the original hypersequent. So, the existence of a saturated hypersequent satisfying the conditions has been proven. ⊠(Lemma)

4.2 Constructing a Kripke Countermodel

Let G' be a saturated hypersequent satisfying all the conditions in the lemma. First, list up the classical variables $X_1, \dots, X_p$, "safe variables," that are in the set $P = \bigcup\{\Gamma_{i\sigma}|\Gamma_{i\sigma} \Rightarrow \Delta_{i\sigma}$ is a component of the saturated hypersequent $G'\}$. We consider the hypersequent all of whose components are of the form $\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}$ such that 1. $\Gamma^{+\prime}_{i\sigma} = \Gamma'_{i\sigma} \cup \{X_1, \dots X_p\}$ and 2. $\Delta'_{i\sigma}$ is as before. We call this hypersequent a "modified saturated hypersequent" G^+.

Proposition 1. *Suppose mLIC^{-} $\nvdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$ $(= G)$, and let G' be a saturated hypersequent satisfying the conditions of Saturation Lemma with "safe variables" $X_1, \ldots, X_p$. Then, there is a modified saturated hypersequent G^+, s.t. for each saturated component $\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}$ of G^+, the following are satisfied:*

(α) *$\Gamma^{+\prime}_{i\sigma} \cap \Delta'_{i\sigma} = \emptyset$ and $\bot \notin \Gamma^{+\prime}_{i\sigma}$.*
(β) *Let P^+ and N be the following.*
- *$P^+ := \bigcup\{\Gamma^{+\prime}_{i\sigma} | (\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma})$ is a component of $G^+\}$.*
- *$N := \bigcup\{\Delta'_{i\sigma} | (\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma})$ is a component of $G^+\}$.*

Then $\{X \in Var_C | X \in P^+\} \cap \{X \in Var_C | X \in N\} = \emptyset$.

Proof. Given a G' and $X_1, \ldots, X_p$, add X_i's to the antecedent of each $\Gamma'_{i\sigma} \Rightarrow \Delta'_{i\sigma}$ in G'. We check that the resulting hypersequent satisfies the conditions of G^+ in the proposition. The hypersequent is still *saturated* even after X_i's are added since these are all variables and inactive in the saturation procedure. The condition (α) is satisfied because, by definition, safe variables X_i's are the classical variables such that $X_i \notin N$. So, adding X_i to the antecedent keeps the antecedent of each saturated component in G' disjoint with its succedent. Similarly, for the condition (β), there are no X_i's such that $X_i \notin N$. Also, it is obvious that this is a modified saturated hypersequent in the sense defined above. $\boxtimes$

Based on this modified saturated hypersequent G^+, we construct a Kripke countermodel for G.[16] Except for classical variables, a model to be constructed from a (modified) saturated hypersequent must have almost the same structure as a Kripke model for intuitionistic logic. However, to construct a Kripke model from one saturated hypersequent, we first construct Kripke models for components, and we glue those Kripke models to construct a Kripke model for G.

Suppose we are given a modified saturated sequent $G^+ = \Gamma^{+\prime}_{10} \Rightarrow \Delta'_{10} | \Gamma^{+\prime}_{11} \Rightarrow \Delta'_{11} | \ldots | \Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0} | \ldots | \Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma} | \ldots | \Gamma^{+\prime}_{n0} \Rightarrow \Delta'_{n0} | \ldots | \Gamma^{+\prime}_{n\tau} \Rightarrow \Delta'_{n\tau}$. We construct Kripke models that falsify the original components $\Gamma_i \Rightarrow \Delta_i$ $(1 \leq i \leq n)$.

Let a Kripke model $\mathcal{K}_i$ be the following triple $(S^+_i, \leq_i, \Vdash_i)$ based on G^+:

1. $S^+_i = \{\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0}, \Gamma^{+\prime}_{i1} \Rightarrow \Delta'_{i1}, \ldots, \Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}\}$ (finite).
2. $\leq_i$ is defined as: $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \leq_i (\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau})$ iff $\Gamma^{+\prime}_{i\sigma} \subseteq \Gamma^{+\prime}_{i\tau}$
3. $\Vdash_i$ is defined as follows[17]: for any $p, X \in \bigcup_{i \leq n}(Sb(\Gamma_i) \cup Sb(\Delta_i))$,
1) $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \Vdash_i p$ iff $p \in \Gamma^{+\prime}_{i\sigma}$; 2) $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \Vdash_i X$ iff $X \in \Gamma^{+\prime}_{i\sigma}$.

For any $p, X \notin \bigcup_{i \leq n}(Sb(\Gamma_i) \cup Sb(\Delta_i))$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i p$ and $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i X$. By this definition, we can immediately obtain some desirable properties of a model of our logic. First, $\geq_i$ is a partial order, since this is an inclusion. Secondly, for any intuitionistic variable $p \in \bigcup_{i \leq n}(Sb(\Gamma_i) \cup Sb(\Delta_i))$, monotonicity clearly holds. Thirdly, about classical variables, the following hold.

[16] In the following, we say "model" unless we emphasize that one is a countermodel.
[17] $Sb(\Gamma)$ stands for the set of subformulas contained in the set of formulas Γ.

Claim. For any $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $\{X | X \in \Gamma^{+\prime}_{i0}\} = \{X | X \in \Gamma^{+\prime}_{i\sigma}\}$.

Proposition 2. *For any $X \in Var_C$, the stability holds, i.e.,*
For any $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \Vdash_i X$ or
for any $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i X$.

Proof. If $X \notin \bigcup_{i \leq n}(Sb(\Gamma_i) \cup Sb(\Delta_i))$, then by definition, for any $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma})$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i X$. The statement easily follows. If $X \in \bigcup_{i \leq n}(Sb(\Gamma_i) \cup Sb(\Delta_i))$, then suppose that for $X \in \bigcup_{i \leq n}(Sb(\Gamma_i) \cup Sb(\Delta_i))$ of G, $\exists(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i X$ and $\exists(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \Vdash_i X$. For particular ρ and τ, $(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$ and $(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \nVdash_i X$, and $(\Gamma^{+\prime}_{i\rho} \Rightarrow \Delta'_{i\rho}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta^{+\prime}_{i0})$ and $(\Gamma^{+\prime}_{i\rho} \Rightarrow \Delta'_{i\rho}) \Vdash_i X$. So, by definition, $X \notin \Gamma^{+\prime}_{i\tau}$ and $X \in \Gamma^{+\prime}_{i\rho}$. However, by the claim, $X \in \Gamma^{+\prime}_{i0}$ iff $X \in \Gamma^{+\prime}_{i\tau}$ and $X \in \Gamma^{+\prime}_{i0}$ iff $X \in \Gamma^{+\prime}_{i\rho}$. Then, $X \in \Gamma^{+\prime}_{i0}$ iff $X \notin \Gamma^{+\prime}_{i0}$. Contradiction. ⊠

Proposition 3. *Let $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma})$ be a component of G^+ in S^+_i and $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$. For any $X \in Sb(\Gamma^+_i) \cup Sb(\Delta_i)$,*[18]

1. *$X \in \Gamma^{+\prime}_{i\sigma} \Longrightarrow \forall(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \Vdash_i X$ and*
2. *$X \in \Delta'_{i\sigma} \Longrightarrow \forall(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \nVdash_i X$*

Proposition 4. *For any $\psi \in Sb(\Gamma^+_i) \cup Sb(\Delta_i)$, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \leq_i (\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau})$ and $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \Vdash_i \psi \Longrightarrow (\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \Vdash_i \psi$.*

Proof. By induction on the complexity of formulas. ⊠

For purely classical formulas, we have another statement to show.

Proposition 5. *For any formula $\psi^\circ \in Sb(\Gamma^+_i) \cup Sb(\Delta_i)$ and for any $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma'_{i0} \Rightarrow \Delta'_{i0})$, 1. $\psi^\circ \in \Gamma^{+\prime}_{i\sigma} \Longrightarrow \forall(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$, $(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \Vdash_i \psi^\circ$; 2. $\psi^\circ \in \Delta'_{i\sigma} \Longrightarrow \forall(\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0}), (\Gamma^{+\prime}_{i\tau} \Rightarrow \Delta'_{i\tau}) \nVdash_i \psi^\circ$.*

Proof. By induction of the complexity of ψ°.

Lemma 2 (Semantic Lemma).
Let G^+ be a modified saturated hypersequent, and let $(\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$ be a component of G^+ in S^+_i. For any $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \geq_i (\Gamma^{+\prime}_{i0} \Rightarrow \Delta'_{i0})$ and for any formula $\varphi \in Sb(\Gamma^+_i) \cup Sb(\Delta_i)$,
1. $\varphi \in \Gamma^{+\prime}_{i\sigma} \Longrightarrow (\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \Vdash_i \varphi$; 2. $\varphi \in \Delta'_{i\sigma} \Longrightarrow (\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i \varphi$.

Proof. By induction on the complexity of formulas.
Case 1.1) $\varphi = p$. For 1. By definition. For 2, suppose $p \in \Delta'_{i\sigma}$. By the condition 3. of Saturation Lemma , $p \notin \Gamma^{+\prime}_{i\sigma}$. So, by definition, $(\Gamma^{+\prime}_{i\sigma} \Rightarrow \Delta'_{i\sigma}) \nVdash_i p$.
Case 1.2) $\varphi = X$. This is a corollary of proposition 12.

[18] Γ^+_i is the antecedent of the original component with safe variables added. We assume we have fixed one saturated hypersequent G^+ to construct a Kripke model.

Case 2) $\varphi = A * B$ ($* = \rightarrow, \wedge, \vee$). We have four subcases of combinations of a mixed formula and a classical formula (1) $A = C^{\circ}$ and $B = D^{\circ}$, (2) A is mixed and B is mixed, (3) $A = C^{\circ}$ and B is mixed, and (4) A is mixed and $B = D^{\circ}$. However, classical formulas are special cases of mixed for intuitionistic formulas, so if the lemma holds for general cases, it obviously holds for classical formulas. It suffices to prove the lemma for generic formulas without specifying whether these are classical or not. The proof is similar to the case of IPC. $\boxtimes$.

4.3 Proof of Completeness and Cut-Elimination

Assuming mLIC$^-$ $\nvdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$, we first construct a Kripke countermodel for a component $\Gamma_i \Rightarrow \Delta_i$. Out of a modified saturated hypersequent G^+, we have constructed Kripke models $\mathcal{K}_i$ ($1 \leq i \leq n$). For each of these, by construction, we have a (root) saturated component ($\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}$) in S_i^+, s.t. $\{X_1, ..., X_p\} \cup \Gamma_i \subseteq \Gamma_{i0}^{+\prime}$ and $\Delta_i \subseteq \Delta_{i0}^{\prime}$. By Semantic Lemma, definition $\Vdash_i$ for sequent and reflexivity of $\leq_i$, $\mathcal{K}_i, (\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}) \nVdash_i \Gamma_i \Rightarrow \Delta_i$ ($1 \leq i \leq n$).

Next, we show the completeness theorem of the hypersequent calculus itself. Assume mLIC$^-$ $\nvdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$. We want to show that there is a Kripke model $\mathcal{K} = (K, \leq, \Vdash)$ and there is a state $s_r \in K$ s.t. $\bigwedge_{1 \leq i \leq n}(\mathcal{K}, s_r \nVdash \Gamma_i \Rightarrow \Delta_i)$. We construct the desired Kripke model by "gluing" constructed Kripke models.

For all the safe variables $X_1, ..., X_p$ in G^+, $X_1, \ldots, X_p \in \Gamma_{i0}^{+\prime}$ ($1 \leq i \leq n$). We already have $\mathcal{K}_i = (S_i^+, \geq_i, \Vdash_i)$ s.t. $\mathcal{K}_i, (\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}) \nVdash_i \Gamma_i \Rightarrow \Delta_i$ ($1 \leq i \leq n$). Then, we first take the disjoint union[19] of the models $\mathcal{K}_i$ ($1 \leq i \leq n$) and add a new node below the roots of the models. Let $s_r = (\Gamma_r^{\prime} \Rightarrow \Delta_r^{\prime})$, $\Gamma_r^{\prime} = \{X_1, ..., X_p\}$ and $\Delta_r^{\prime} = \emptyset$. s_r is the new root node. Let $(\Gamma_{i\sigma}^{+\prime} \Rightarrow \Delta_{i\sigma}^{\prime})$ be a component of G^+.

Now let $\mathcal{K} = (S^+, \leq, \Vdash)$, where 1. $S^+ = \{s_r\} \cup \biguplus_{1 \leq i \leq n} S_i^+$; 2. $\leq = \{(s_r, s) \in \{s_r\} \times S^+ | \Gamma_r^{\prime} \subseteq \Gamma_{i\sigma}^{+\prime}$ or $\Gamma_r^{\prime} \subseteq \Gamma_r^{\prime}\} \cup \biguplus_{1 \leq i \leq n}(\leq_i)$, where either $s = (\Gamma_r^{\prime} \Rightarrow \Delta_r^{\prime})$ or $s = (\Gamma_{i\sigma}^{+\prime} \Rightarrow \Delta_{i\sigma}^{\prime})$ s.t. $(\Gamma_{i\sigma}^{+\prime} \Rightarrow \Delta_{i\sigma}^{\prime}) \in \biguplus_{1 \leq i \leq n} S_i^+$; 3. $\Vdash = \{(s_r, X_l) | X_l \in \Gamma_r^{\prime}\} \cup \biguplus_{1 \leq i \leq n}(\Vdash_i)$. Such a model $\mathcal{K}$ must exist, since the only case where the glued model does not exist is the case where we have a conflict among classical variables, but by construction we never have such a case here.

Since the partial order in the new model is the inclusion on the antecedents of the saturated sequents, monotonicity obviously holds for intuitionistic atoms. The new condition for classical variables must be a consequence of the claim: for any $(\Gamma_{i\sigma}^{+\prime} \Rightarrow \Delta_{i\sigma}^{\prime}) \geq (\Gamma_r^{\prime} \Rightarrow \Delta_r^{\prime})$ in S^+, $\{X | X \in \Gamma_r^{\prime}\} = \{X | X \in \Gamma_{i\sigma}^{+\prime}\}$. The proof is essentially the same as that given to the nodes of $\mathcal{K}_i$.

Also, the following are the immediate consequences of the definition.

1. $s_r \Vdash X_l$ iff $X_l \in \Gamma_r^{\prime}$ iff $\forall s \in S^+$, $s \Vdash X_l$.
2. $\forall s \in S_i^+$, $s \Vdash \varphi \Longleftrightarrow s \Vdash_i \varphi$ for any formula φ,.

Claim. If mLIC$^-$ $\nvdash \Gamma_1 \Rightarrow \Delta_1 | \ldots | \Gamma_n \Rightarrow \Delta_n$, then $\bigwedge_{1 \leq i \leq n}(\mathcal{K}, s_r \nVdash \Gamma_i \Rightarrow \Delta_i)$.

[19] $\biguplus$ stands for the disjoint union operator.

Proof. In $\mathcal{K}_i$, at $(\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime})$, each component is falsified, respectively. So, for each i $(1 \leq i \leq n)$, at the state $(\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}) \in S^{+}$, the following hold: 1) $(\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}) \geq (\Gamma_{r}^{\prime} \Rightarrow \Delta_{r}^{\prime})$; 2) $(\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}) \Vdash \varphi$ for all $\varphi \in \Gamma_i$; 3) $(\Gamma_{i0}^{+\prime} \Rightarrow \Delta_{i0}^{\prime}) \nVdash \psi$ for all $\psi \in \Delta_i$. So, by definition, $\bigwedge_{1 \leq i \leq n}(\mathcal{K}, s_r \nVdash \Gamma_i \Rightarrow \Delta_i)$ ⊠(claim)

Hence, mLIC$^-$ $\nvdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$ implies $\bigwedge_{1 \leq i \leq n}(\mathcal{K}, s_r \nVdash \Gamma_i \Rightarrow \Delta_i)$. So, if mLIC$^-$ $\nvdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$, then $\mathcal{K}, s_r \nVdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$. So, if $\Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$ is valid, then mLIC$^-$ $\vdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$.

We have shown completeness of mLIC$^-$. On the other hand, the soundness theorem for mLIC itself holds with respect to the same class of Kripke models, as already discussed. Then, if there is a Kripke model $\mathcal{K}$ and $s \in K$, s.t. $\mathcal{K}, s \nVdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$, then mLIC$\nvdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$.

So, if mLIC$\vdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$, then mLIC$^-$ $\vdash \Gamma_1 \Rightarrow \Delta_1 | \dots | \Gamma_n \Rightarrow \Delta_n$. This gives a semantic proof of the cut-elimination theorem for mLIC.

As a corollary, the *subformula property* holds for mLIC, since Cut is the only rule that spoils the subformula property in mLIC. Due to this corollary, IPC$_{CA}$ is a *conservative extension* of its intuitionistic and classical fragments, respectively, since for any purely intuitionistic (or classical) formula provable in mLIC, there is a proof using only subformulas of the formula. The model we constructed is finite, so we have *decidability* of mLIC.

We have another corollary of completeness. A sequent system S has the *disjunction property* (DP) if $S \vdash \Rightarrow A \vee B \Longrightarrow S \vdash \Rightarrow A$ or $S \vdash \Rightarrow B$. We have a refinement of DP. To state the proposition, we need a definition of a positive and negative occurrence of a formula in a sequent. We put $+$ and $-$ symbol in front of a formula to state that the given formula has a positive occurrence and a negative occurrence. "$+\varphi$" means φ has a positive occurrence and "$-\varphi$" means φ has a negative occurrence. The occurrences of a subformula of a given formula is determined inductively as follows. For a sequent $\Gamma \Rightarrow \Delta$, 1) If $\varphi \in \Gamma$, then $-\varphi$: if $-\varphi$ and $\varphi = (A \wedge B)$, then $-A$ and $-B$; if $-\varphi$ and $\varphi = (A \vee B)$, then $-A$ and $-B$; if $-\varphi$ and $\varphi = (A \rightarrow B)$, then $+A$ and $-B$; 2) If $\varphi \in \Delta$, then $+\varphi$: if $+\varphi$ and $\varphi = (A \wedge B)$, then $+A$ and $+B$; if $+\varphi$ and $\varphi = (A \vee B)$, then $+A$ and $+B$; if $+\varphi$ and $\varphi = (A \rightarrow B)$, then $-A$ and $+B$.

We say "the *Extended Disjunction Property* (EDP) holds" when the following statement holds, since DP for IPC is a special case of this.

Proposition 6. *In mLIC, if no classical subformula of $A \vee B$ has both negative and positive occurrences in A and B of $\Rightarrow A \vee B$, then $\Rightarrow A$ or $\Rightarrow B$ holds.*

Proof. Proof by contradiction. Suppose that there exists $A \vee B$ such that not $\Rightarrow A$ and not $\Rightarrow B$, but that $\Rightarrow A \vee B$ s.t. there is no classical subformula of $A \vee B$ whose positive and negative occurrences appear in A and B of $A \vee B$.

First, observe that the above inductive characterization of positive and negative occurrences of a subformula of φ in Γ or Δ corresponds to a step in the saturation procedure that puts a subformula of a formula in Γ and Δ. The inductive characterization obviously gives us: $-\varphi$ in G iff φ occurs in the antecedent of some saturated component of some G'; $+\varphi$ in G iff φ occurs in the succedent of some saturated component of some G'.

By assumption, we have a case of $\Rightarrow A \vee B$ where we do not have any occurrence of a classical subformula φ° in A (in B) s.t. $+\varphi^\circ$ and in B (in A) s.t. $-\varphi^\circ$. So, in particular, there is no X such that $+X$ $(-X)$ in mLIC$^-$ $\nvdash\Rightarrow A$ and $-X$ $(+X)$ in mLIC$^-$ $\nvdash\Rightarrow B$. So, we can construct saturated hypersequents G'_A of mLIC$^-$ $\nvdash\Rightarrow A$ and G'_B of mLIC$^-$ $\nvdash\Rightarrow B$ such that there is no classical variable in some succedent (antecedent) of G'_A and in some antecedent (succedent) of G'_B.

By completeness, we can construct countermodels $\mathcal{K}_A$ and $\mathcal{K}_B$ based on G'_A and G'_B s.t. $\mathcal{K}_A, r_A \nVdash\Rightarrow A$ and $\mathcal{K}_B, r_B \nVdash\Rightarrow B$. By construction, there is no classical variable X such that at some s_A of $\mathcal{K}_A$, $s_A \Vdash X$ (or $s_A \nVdash X$) and at some s_B of $\mathcal{K}_B$, $s_B \nVdash X$ (or $s_B \Vdash X$). This is obviously sufficient to use the same gluing method as used in the proof of completeness. So, we can construct a countermodel $\mathcal{K}, r \nVdash\Rightarrow A \vee B$. So, by soundness, $\Rightarrow A \vee B$ is not mLIC-provable, which contradicts our assumption $\Rightarrow A \vee B$.[20] ⊠

References

1. S. Artemov and R. Iemhoff. The Basic Intuitionistic Logic of Proofs. *Journal of Symbolic Logic*, to appear in 2007.
2. A. Avron. Hypersequents, Logical Consequence, and Intermediate Logics for Concurrency. *Annals of Mathematics and Artificial Intelligence*, 4:225–248, 1991.
3. A. Avron. The Method of Hypersequents in the Proof Theory of Propositional Non-classical Logics. In Wilfrid Hodges, Martin Hyland, Charles Steinhorn, and John Truss, editors, *Logic: from foundations to applications. Proc. Logic Colloquium, Keele, UK, 1993*, pages 1–32. Oxford University Press, New York, 1996.
4. A. Avron. Two Types of Multiple-Conclusion Systems. *Journal of the Interest Group in Pure and Applied Logics*, 6 (5):695–717, 1998.
5. M. Baaz, A. Ciabattoni, and C. G. Fermüller. Hypersequent Calculi for Gödel Logics - a Survey. *Jounal of Logic and Computation*, 13:1–27, 2003.
6. L. Fariñas del Cerro and A. Herzig. Combining Classical and Intuitionistic Logic, or: Intuitionistic Implication as a Conditional. In F. Baader and K. U. Schulz, editors, *Frontiers of Combining Systems: Proceedings of the 1st International Workshop, Munich (Germany)*, pages 93–102. Kluwer Academic Publishers, 1996.
7. M. Fitting. *Intuitionistic Logic, Model Theory and Forcing.* North Holland, 1969.
8. R. C. Flagg. Integrating classical and intuitionistic type theory. *Annals of Pure and Applied Logic*, 32:27–51, 1986.
9. J-Y. Girard. On the Unity of Logic. *Annals of Pure and Applied Logic*, 59:201–217, 1993.
10. L. Humberstone. Interval Semantics for Tense Logic: Some Remarks. *Journal of Philosophical Logic*, 8, 1979.
11. H. Kurokawa. Intuitionistic Logic with Classical Atoms. Technical report, CUNY Ph.D. Program in Computer Science Technical Report TR-2004003, 2004.
12. O. Laurent and K. Nour. Parametric mixed sequent calculus. Technical report, Université de Savoie, 2005. Prépublication du Laboratoire de Mathmatiques num 05-05a.

[20] Note that the converse of the proposition does not hold. Even if there is such a subformula in the formula, we may not have to apply Classical Splitting to get a cut-free proof of the formula, and we have an intuitionistic proof of it. Here is a simple example, $\Rightarrow ((X \wedge p) \rightarrow p) \vee (p \rightarrow X)$.

13. P. Lucio. Structured Sequent Calculi for Combining Intuitionistic and Classical First-Order Logic. In H. Kirchner and Ch. Ringeissen, editors, *Proceedings of the Third International Workshop on Frontiers of Combining Systems, FroCoS 2000, LNAI 1794*, pages 88–104. Springer, 2000.
14. P. Miglioli, U. Moscato, M. Ornaghi, and G. Usberti. A Constructivism Based on Classical Truth. *Notre Dame Journal of Formal Logic*, 30(1):67–90, 1989.
15. G. Mints. *A Short Introduction to Intuitionistic Logic*. Kluwer Academic - Plenum, 2000.
16. S. Negri and J. von Plato. *Structural Proof Theory*. Cambridge University Press, Cambridge, UK, 2001.
17. K. Nour and A. Nour. Propositional mixed logic : its syntax and semantics. *Journal of Applied Non-Classical Logics*, 13:377–390, 2003.
18. A. Sakharov. Median Logic. Technical report, St. Petersberg Mathematical Society. http://www.mathsoc.spb.ru/preprint/2004/index.html.
19. C. Smorynski. Application of Kripke models. In A. Troelstra, editor, *Metamathematical Investigations of Intuitionistic Arithmetic and Analysis*. Springer Verlag, 1973.

Proof Identity for Classical Logic: Generalizing to Normality

Roman Kuznets

CUNY Graduate Center
365 Fifth Avenue, New York City, NY 10016, USA
kuznets@gmail.com

Abstract. The problem of the identity criteria for proofs can be traced to Hilbert and Prawitz. One of the approaches, which uses the concept of generality of proofs, was suggested in 1968 by Lambek. Following his ideas, we propose a language and a logic to represent Hilbert-style proofs for classical propositional logic by adapting the Logic of Proofs (LP) introduced by Artemov in 1994. We prove that proof polynomials, the objects representing Hilbert derivations in LP, are sufficient to realize all propositional derivations, with or without hypotheses. We also show that proof polynomials respect the ideas of generality and provide an algorithm for determining whether two given proof polynomials represent the same proof. These results naturally extend similar properties of combinatory logic demonstrated by Hindley. The language of LP allows us to formally capture more structure of Hilbert-style proofs. In particular, we show how the well-known phenomenon of proof composition in classical logic manifests itself in the case of Hilbert proofs.

1 Introduction

It was recently discovered that the 24th of the famous 23 Hilbert problems, which was dropped from the list, was to develop such a theory of proofs that would allow to compare proofs and provide criteria for judging which is the simplest proof of a given theorem (see [TW02]).

Several decades later, the question reappeared independently. This is how Prawitz formulates it in [Pra71]: "In the same way as one asks when two formulas define the same set or two sentences express the same proposition, one asks when two derivations represent the same proof; in other words, one asks for identity criteria for proofs or for a "synonymity" (or equivalence) relation between derivations."

In his series of papers [Lam68, Lam69, Lam72], Lambek suggested considering two proofs identical iff they have the same generalizations; this supposition is sometimes called *Generality Conjecture*. By *generalization* we mean here a more general statement that can be proven by applying the same principles in the same order. Here is an example:

Example 1. The tautology

$$A \to A \vee (B \to B) \tag{1}$$

can be derived in several ways:

1. (a) $A \to A \vee (B \to B)$ *disjunction axiom*

2. (a) ... *derivation of tautology $B \to B$*
   (b) $B \to B$
   (c) $(B \to B) \to A \vee (B \to B)$ *disjunction axiom*
   (d) $A \vee (B \to B)$ *modus ponens from (b) and (c)*
   (e) $A \vee (B \to B) \to (A \to A \vee (B \to B))$ *implication axiom*
   (f) $A \to A \vee (B \to B)$ *modus ponens from (d) and (e)*

The two derivations are easily distinguished according to Generality Conjecture. Indeed a more general theorem proven by derivation 1 is

$$A \to A \vee (B \to C) \tag{2}$$

(or even $A \to A \vee D$), while a more general conclusion of derivation 2 is

$$A \to C \vee (B \to B)\ . \tag{3}$$

Our original tautology (1) is a substitution instance of both (2) and (3), but neither (3) can be derived by derivation 1 nor (2) can be derived by derivation 2.

To be able to distinguish and/or identify proofs, one would need a sufficiently concise way of representing them that complies with the ideas of generality. Many systems for naming and/or identifying proofs have been developed, in particular very robust systems for intuitionistic and linear logics. Designing such a system for classical logic, however, has presented various challenges. As a result, there exist multiple proof-naming notations based on various classical proof systems, but none of them holds a dominant position unlike, for instance, the uniformly accepted λ-terms for intuitionistic logic. These notations and proof systems include, but are not limited to, Andrews's matings method ([And76]), Bibel's matrix method ([Bib79]), proof-nets for a certain classical sequent calculus by Lamarche and Straßburger ([LS05]), classical implication bases method for cut-free atomically closed tableaux by Fitting ([Fit06]), classical λ-terms by Nour ([Nou06]). Some of these approaches, e.g. [Fit06], rely on cut-free properties. Others restrict the language, e.g. in [LS05] only formulas in negation normal form[1] are considered. Identity of proofs in these proof systems is often related to normalization in the respective proof system rather than to generality of proofs.

[1] Negation is restricted to atomic formulas.

We will concentrate on Hilbert-style proofs, for which few proof-naming systems have been developed. Arguably, Hilbert derivations resemble the way human mathematicians work the most. On the other hand, Hilbert-style proofs are difficult to work with because their are inherently not cut-free (the only inference rule, *modus ponens*, essentially is the cut); furthermore, they do not have a precisely defined notion of normalization. It would seem that combinatory logic is a ready-made system of terms for Hilbert-style derivations. Yet applications of combinatory logic to proof identity via generality concept are scarce.

We would further argue that combinatory logic does not, in fact, capture all the facets of derivations from hypotheses. To substantiate this argument we will use a richer language, namely the language of Artemov's Logic of Proofs (LP) developed in [Art95, Art01]. LP is equipped with a construct $t{:}F$ that resembles a combinatory statement and is read as 'proof polynomial t is a proof of formula F.' But LP also considers boolean combinations of such statements and allows to iterate the :-operator, e.g. $t{:}s{:}F$. As a result, proof polynomials in LP are not limited to propositional derivations only. Any derivation in LP can be represented by a proof polynomial, thus LP is a proof-naming system for itself rather than for classical propositional logic.

In this paper, we discuss how the machinery of proof polynomials can be applied to the problem of identifying Hilbert-style proofs in classical propositional logic. In Sect. 2, we present LP and identify its fragment that seems to best suit this purpose. In Sect. 3, we show that proof polynomials in the chosen fragment comply with the idea of generality. We also describe how to extract from a proof polynomial the most general theorem proven by it, thereby providing a decision procedure that determines whether two given proof polynomials represent the same proof. In Sect. 4, we suggest a natural definition of a normal form for valid statements about Hilbert proofs in the chosen fragment of LP. In a *normal* formula all proof polynomials occurring negatively are atomic, which is always the case for combinatory terms in combinatory logic. This prompted Goris to formulate a natural conjecture about proof polynomials that every valid statement is a substitution instance of a normal valid statement. We provide a counterexample to this conjecture. However, we note that the conjecture does indeed hold for formulas in Horn clause form, which can be viewed as a conservativity result with respect to combinatory logic. Finally, in Sect. 5, we show how to refine the conjecture using the proof composition operation $+$ explicitly present in the language of LP, prove the amended conjecture, and discuss the impact of $+$ on proof identity.

2 Logic of Proofs LP: Choosing a Fragment

The system LP was originally presented in [Art95] (see also [Art01]) as a logic of formal mathematical proofs. The idea of introducing explicit proofs into the propositional language dates back to Gödel's lecture in 1938, although it was published only in 1995 (see [Göd95]). Independently Artemov implemented the same idea and thus solved a long-standing problem of an adequate provability semantics for modal logic S4. The success of the project was largely due to the operation of proof composition, denoted in LP by $+$, which was not present in Gödel's lecture.

2.1 Full LP

Proof polynomials t are built from proof constants c_i and proof variables x_i by means of three operations: unary ! and binary $+$ and $\cdot$

$$t ::= c_i \mid x_i \mid !\,t \mid t \cdot t \mid t + t$$

The language of LP is obtained by adding to the propositional language a new construct, $t\!:\!F$, where F is a formula and t is a proof polynomial. The axioms of LP_0 are obtained by adding the following schemas to a fixed finite set of axiom schemas of classical propositional logic:

LP1	$s\!:\!(F \to G) \to (t\!:\!F \to (s \cdot t)\!:\!G)$	(application)
LP2	$t\!:\!F \to !\,t\!:\!t\!:\!F$	(proof checker)
LP3	$s\!:\!F \to (s+t)\!:\!F, \quad t\!:\!F \to (s+t)\!:\!F$	(proof composition)
LP4	$t\!:\!F \to F$	(reflexivity)

The only inference rule of LP_0 is *modus ponens*. The usual way to define the full LP is to add to LP_0 the rule of axiom necessitation:

If $\mathcal{A}$ is a propositional axiom or one of LP1–4 *and c is a constant, infer $c\!:\!\mathcal{A}$.*

The system LP behaves in many respects as a normal propositional logic. In particular, LP is closed under substitutions (of both formulas for sentence letters and proof polynomials for proof variables) and enjoys the deduction theorem. The informal semantics for proof polynomials in LP considers variables x_i to be unspecified proofs, and constants c_i to be unanalyzed proofs of elementary facts, i.e., logical axioms.

2.2 Choosing a Constant Specification

LP has various subsystems, many of which can be created by changing the constant specification.

A *constant specification* $\mathcal{CS}$ is a set of LP-formulas of form $c\!:\!\mathcal{A}$, where c is a proof constant and $\mathcal{A}$ is an axiom.

A constant specification is called *injective* if no proof constant is assigned to two different axiom instances. In such specifications, each constant carries complete information about the axiom instance the proof of which it represents.

A constant specification is called *schematic* if every proof constant is assigned to one or several axiom schemas ([Mil07]).

A constant specification is called *schematically injective* if it is schematic and every proof constant is assigned to at most one axiom schema ([Mil07]). Note that a schematically injective $\mathcal{CS}$ is not injective unless it is empty.

A constant specification is called *axiomatically appropriate* if every axiom instance has a proof constant assigned to it ([Fit05]).

The *maximal* constant specification is that in which each proof constant is assigned to every axiom ([Kuz00]). This corresponds to the unrestricted use of the axiom necessitation rule.

$\mathsf{LP}_{\mathcal{CS}}$ is defined as the result of adding constant specification $\mathcal{CS}$ as new axioms to LP_0. LP then is $\mathsf{LP}_{\mathcal{CS}}$ for the maximal constant specification $\mathcal{CS}$.

Using the full LP to compare proofs is not viable because the maximal constant specification erases the differences between axiom schemas. But trying to distinguish between any two axiom instances through the use of an injective constant specification would go against the spirit of Lambek's idea of generality. Generality hinges on the fact that substitution instances of the same axiom schema amount to the same proof. Hence, the best choice to tackle proof identity is through axiomatically appropriate, schematically injective constant specifications. Axiomatic appropriateness will ensure that a full scope of valid facts is covered, whereas schematic injectivity will give the desired level of details.

2.3 Choosing a Fragment

As discussed earlier, our goal is to build a language describing proofs for classical propositional logic, not LP itself; hence we have to restrict the scope of proof polynomials to purely classical formulas:

Definition 1. *The language of the* Propositional Logic of Proofs, PLP, *consists of all* LP*-formulas that satisfy the following restriction:* F *in any subformula* $t{:}F$ *must be purely propositional. The only operations on proof polynomials are* $\cdot$ *and* $+$.

As a consequence, we must get rid of axiom LP2 and similarly restrict the use of the axiom necessitation rule to propositional axioms (in the propositional language). So instead of axiomatic appropriateness, we will require *propositional appropriateness*:

Definition 2. *A constant specification is* propositionally appropriate *if every propositional axiom instance has a proof constant assigned to it.*

We will formulate two versions of the Propositional Logic of Proofs: with and without the +-operation, which corresponds to the union, or composition, of proofs.

Definition 3. PLP^- *is a logic in the language of* PLP *whose axioms are a finite set of propositional axiom schemas together with axiom schemas LP1 and LP4. The inference rules are modus ponens and the axiom necessitation rule, the latter being restricted to a fixed propositionally appropriate, schematically injective constant specification* $\mathcal{CS}$.

PLP^+ *is obtained by adding the axiom schema LP3 to* PLP^-.

PLP *will be used as a generic name for both* PLP^- *and* PLP^+.

Note 1. Strictly speaking, the definition above depends on the chosen axiomatization of classical propositional logic and on the selected constant specification. In the paper, we will be working with one such axiomatization and a corresponding constant specification $\mathcal{CS}$, which will be fixed throughout the paper.

2.4 Reflexive Fragment of PLP

Nikolai Krupski in [Kru06] developed calculus $\mathsf{rLP}_{\mathcal{CS}}$, which he showed to be sound and complete with respect to the so-called reflexive fragment of $\mathsf{LP}_{\mathcal{CS}}$:

$$\mathsf{rLP}_{\mathcal{CS}} = \{s\!:\!G \mid \mathsf{LP}_{\mathcal{CS}} \vdash s\!:\!G\}$$

The set of axioms of $\mathsf{rLP}_{\mathcal{CS}}$ coincides with $\mathcal{CS}$. The rules are

$$(\mathrm{R1})\frac{s\!:\!G \quad r\!:\!(G \to H)}{(r \cdot s)\!:\!H} \qquad (\mathrm{R2})\frac{s_i\!:\!G}{(s_1 + s_2)\!:\!G} \quad i = 1,2 \qquad (\mathrm{R3})\frac{s\!:\!G}{!\,s\!:\!s\!:\!G}$$

To adapt this calculus for PLP^+, we have to get rid of rule (R3). In addition, for PLP^- we also omit rule (R2). The resulting calculi for the previously fixed $\mathcal{CS}$ will be denoted by rPLP^+ and rPLP^- respectively.

Note 2. Calculus rPLP^- closely resembles combinatory logic without reduction rules. Proof constants play the role of combinators providing at least one combinator for each axiom schema in the chosen axiomatization of propositional logic. The following theorem then proves conservativity of PLP^- over such a combinatory logic.

Theorem 1. *1.* $\mathsf{PLP}^- \vdash t\!:\!F$ *iff* $\mathsf{rPLP}^- \vdash t\!:\!F$
2. $\mathsf{PLP}^+ \vdash t\!:\!F$ *iff* $\mathsf{rPLP}^+ \vdash t\!:\!F$

Proof. The "if" direction is trivial.
For the "only if" direction, let $\mathsf{PLP} \vdash t\!:\!F$. Then $\mathsf{LP}_{\mathcal{CS}} \vdash t\!:\!F$. By the completeness theorem for $\mathsf{rLP}_{\mathcal{CS}}$ proven in [Kru06], $\mathsf{rLP}_{\mathcal{CS}} \vdash t\!:\!F$. Since proof polynomial t is !-free (also $\{+\}$-free in case of PLP^-), this $\mathsf{rLP}_{\mathcal{CS}}$ derivation can also be performed within rPLP. □

3 PLP and Generality Paradigm

3.1 Proof Polynomials Represent All Derivations

The following property is an analog of Curry-Howard isomorphism:

Lifting Lemma (Artemov, *[Art95, Art01]*). *1. If $\vdash_{\mathsf{LP}} F$, then there exists a $\{+\}$-free ground*[2] *proof polynomial t such that $\vdash_{\mathsf{LP}} t\!:\!F$.*
2. If $B_1, \dots, B_n \vdash_{\mathsf{LP}} F$, then there exists a $\{+\}$-free proof polynomial t depending on distinct fresh proof variables x_i such that

$$\vdash_{\mathsf{LP}} x_1\!:\!B_1 \wedge \dots \wedge x_n\!:\!B_n \to t(x_1, \dots, x_n)\!:\!F \enspace .$$

Proof. The proof is by an easy induction on a Hilbert derivation of F (in the first part, F is a theorem; in the second part, it is derived from hypotheses B_i). This proof also allows to introduce a "canonical" proof polynomial for each Hilbert-style derivation:

[2] *Ground* proof polynomial does not have proof variables occurring within it.

Definition 4. *The proof polynomial* t representing *a Hilbert-style derivation* $\mathcal{D}$ *of a formula* F *from hypotheses* $B_1, \dots, B_n$ *is defined by induction on* $\mathcal{D}$*:*

1. *An axiom instance is represented by a proof constant corresponding to the axiom schema used.*
2. *A hypothesis* B_i *is represented by a fresh proof variable* x_i*.*
3. *An application of* modus ponens *to* $P \to Q$ *and* P *is represented by the proof polynomial* $t \cdot s$*, where* t *and* s *are the representations of the derivations of* $P \to Q$ *and* P *respectively.*
4. *An application* $c{:}\mathcal{A}$ *of the axiom necessitation rule is represented by* $!\,c$*.*

Note that the proof polynomial representation of a given derivation is $\{+\}$-free. Also note that the last case is not required for purely propositional derivations. Hence terms representing propositional derivations are $\{+, !\}$-free. Finally, no proof variables are used in the absence of hypotheses, so that in this case the representing polynomial is ground. □

This proof, if applied to the restricted fragment PLP^-, yields the following statement:

Theorem 2 (Sufficiency of PLP^--proofs). *1. If* $\mathcal{D}$ *is a propositional derivation of* F *and* t *is the* $\{+\}$*-free*[3] *ground proof polynomial representing* $\mathcal{D}$*, then* $\vdash_{\mathsf{PLP}^-} t{:}F$*.*
2. If $\mathcal{D}$ *is a propositional derivation of a formula* F *from hypotheses* $B_1, \dots, B_n$ *and* $t(x_1, \dots, x_n)$ *is the* $\{+\}$*-free proof polynomial representing* $\mathcal{D}$*, then*

$$\vdash_{\mathsf{PLP}^-} x_1{:}B_1 \wedge \ldots \wedge x_n{:}B_n \to t(x_1, \dots, x_n){:}F \enspace .$$

This theorem shows that PLP^- is already sufficient to realize all stand-alone Hilbert-style propositional proofs, with or without hypotheses.

3.2 Proof Polynomials Respect Proof Identity

A tautology F has many proof polynomials t such that $\mathsf{PLP}^- \vdash t{:}F$:

Example 2. The two proofs of (1) from Example 1 are represented by proof polynomials a and $k(d((sk)k))$ respectively, where a, s, k, and d are proof constants representing the following axiom schemas $a{:}\Big(P \to P \vee Q\Big)$, $d{:}\Big(Q \to P \vee Q\Big)$, $s{:}\Big((P \to (Q \to R)) \to ((P \to Q) \to (P \to R))\Big)$, and $k{:}\Big(P \to (Q \to P)\Big)$.[4]

We must verify that proof polynomials are faithful representations of derivations, i.e., we must ensure that distinct proofs are not represented by the same proof polynomial.

[3] Note that PLP-terms do not contain !.
[4] Here proof polynomial $(sk)k$ represents the standard derivation of $P \to P$ familiar from combinatory logic.

Definition 5. *A propositional formula G is called a* generalization *of a formula F if F is obtained from G by substituting propositional formulas for sentence letters: $F = G\sigma$.*

Definition 6. *Two Hilbert-style derivations $\mathcal{D}$ and $\mathcal{D}'$ of a tautology F are called* similar[5] (according to the generality paradigm) *if every generalization G of F that can be derived by $\mathcal{D}$ (using the same axiom schemas and applying* modus ponens *in the same order) can also be derived by $\mathcal{D}'$, and vice versa.*

Lemma 1. *A derivation $\mathcal{D}$ proves a propositional tautology F iff $\mathsf{PLP}^- \vdash t{:}F$ for the proof polynomial t representing $\mathcal{D}$.*

Proof. The "only if" direction follows from Theorem 2.
For the "if"-direction, let $\mathsf{PLP}^- \vdash t : F$. By Theorem 1, $\mathsf{rPLP}^- \vdash t : F$. Stripping all proof polynomials from this rPLP^--derivation, we obtain a propositional derivation of F. Moreover, according to Definition 4, this derivation is exactly $\mathcal{D}$. □

The desired property follows from Lemma 1 trivially:

Theorem 3. *If propositional derivations $\mathcal{D}$ and $\mathcal{D}'$ are represented by the same proof polynomial, they are similar.*

Thus we proved that proof polynomials respect proof identity. Note that the converse of Theorem 3 does not hold. For instance, as Evan Goris observed, it is easy to add some redundancy to the standard skk derivation of $P \to P$ mentioned in Example 2; the representing proof polynomial would expand accordingly. At the same time, within the generality paradigm $P \to P$ cannot have more than one proof since there are no non-trivial derivable generalization of $P \to P$.

3.3 Proof Identity Is Decidable

PLP also allows to determine effectively whether two given proof polynomials represent similar derivations. For each proof polynomial t there exists the most general tautology (if any) proven by t:

Lemma 2. *If $\mathsf{PLP}^- \vdash t{:}F$, then there exists a tautology G such that*
1. $\mathsf{PLP}^- \vdash t{:}G$, and
2. any tautology H such that $\mathsf{PLP}^- \vdash t{:}H$ is a substitution instance of G.
Moreover, this G can be effectively constructed (in polynomial time).

Proof. By Theorem 1, $\mathsf{rPLP}^- \vdash t : F$. Consider this rPLP^--derivation. By induction on the derivation depth, we simultaneously construct a generalization G and a substitution σ that maps G back to F.

Each axiom instance in a leaf can be replaced by the axiom schema corresponding to the proof constant used. This axiom schema is a generalization of

[5] We use the term "similar" instead of "identical" to avoid an undesirable association with syntactical identity.

the original axiom instance used, as well as of any other instance of this axiom schema. The substitution is easily read from the original axiom instance. We use fresh sentence letters for each of the leaves, which allows to combine their substitutions seamlessly.

When the rule (R1) is used in the original derivation for formulas A and B

$$\frac{s{:}A \quad r{:}(A \rightarrow B)}{(r \cdot s){:}B} ,$$

by IH, we have A replaced by its generalization A_1 and $A \rightarrow B$ replaced by its generalization $A_2 \rightarrow B_2$. In order to apply the same rule, we find the most general unifier (mgu) of A_1 and A_2 and apply it to all the formulas in the generalized derivation. By IH, there exists a substitution σ that maps A_1 to A and $A_2 \rightarrow B_2$ to $A \rightarrow B$ (and hence A_2 to A); therefore, σ is a unifier of A_1 and A_2. Hence mgu τ of A_2 and A_1 must exist, and the substitution σ must have form $\tau\theta$ for some substitution θ. So B is still a substitution of $B_2\tau$. Clearly, τ is the most general substitution under which we could apply the rule. Hence any other B' such that $\mathsf{rPLP}^- \vdash r \cdot s{:}B'$ will be a substitution instance of $B_2\tau$. □

Remark 1. If the axiom schemas are written in the form of dags, the unifications can be performed polynomially via modified Robinson's algorithm (see [CB83]).

Remark 2. An analogous statement for combinatory logic is due to Hindley (see [Hin69]).

Theorem 4. *Given two proof polynomials t and s such that $\mathsf{PLP}^- \vdash t{:}F$ and $\mathsf{PLP}^- \vdash s{:}F$, it is possible to determine effectively whether t and s represent similar derivations under the generality paradigm.*

Proof. Construct the most general tautology G_t proven by t and the most general tautology G_s proven by s. If G_t and G_s are the same modulo renaming of sentence letters (more precisely, each one is a substitution instance of the other), then t and s represent similar derivations. Otherwise, they represent different derivations as either G_t or G_s will not be a substitution instance of the other. □

4 Normal Form for PLP

It was noted in Theorem 2.2 that each propositional derivation of F from hypotheses B_i can be represented by a proof polynomial t such that

$$\vdash_{\mathsf{PLP}^-} x_1{:}B_1 \wedge \ldots \wedge x_n{:}B_n \rightarrow t{:}F \ , \tag{4}$$

which is a PLP^--theorem in Horn clause form. An arbitrary PLP^--theorem can therefore be viewed as a statement describing the relationships between classical Hilbert-style derivations represented by the proof polynomials occurring in the theorem. Such a reading suggests that proof polynomials occurring negatively signify the use of a hypothesis (similar to x_i in (4)) whereas proof polynomials occurring positively represent derivations from these hypotheses (similar to t in (4)). In particular, a statement $t{:}F \to s{:}G$ should be read as 'if t stands for the hypothesis F, then s represents a derivation of G from F.' Under this reading, it seems that the structure of t should not play any role in this statement: after all, t is nothing but a label for the place(s) where the hypothesis was used. In that case, it would seem reasonable to denote such a label by an atomic proof polynomial, i.e., a proof variable, as is done in (4). One would also expect to see occurrences of t within s.

Definition 7. *A* PLP*-formula is called* normal *if all proof polynomials occurring negatively in it are distinct proof variables.*

This prompted Evan Goris to formulate a conjecture that, applied to PLP,[6] becomes

Goris's Conjecture. *For each* PLP^-*-theorem F there exists a normal* PLP^-*-theorem N such that*

$$F = N\sigma,$$

where substitution σ only replaces proof variables by proof polynomials (sentence letters remain unaffected).

This conjecture turns out to be true for the Horn-clause-like fragment shown earlier to be sufficient for representing all classical derivations. The proof of this statement, for which we develop an apparatus later in the paper, will be given at the end of Sect. 5. The statement can be viewed as an extended conservativity result with respect to combinatory logic: any PLP^- statement that has combinatory, i.e., Horn clause, format is indeed an instance of a statement derivable in combinatory logic.

However, as a PLP^--derivation may in fact use the inner structure of a negative proof polynomial, it is possible to construct a counterexample to Goris's Conjecture when applied to the full PLP^-. Departing from Horn-clause-like form allows us to combine alternative hypotheses by purely propositional methods, without the use of $+$.

Theorem 5 (Counterexample to Goris's Conjecture). *The formula*

$$F = \Big(x{:}P \wedge y{:}(P \to Q)\Big) \vee (y \cdot x){:}Q \to (y \cdot x){:}Q \tag{5}$$

is derivable in PLP^- *but is not a substitution of any normal derivable formula.*

The formula states that a sentence letter Q can be derived either from Q itself or by *modus ponens* from hypotheses $P \to Q$ and P for some sentence letter P.

Proof (by contradiction). Suppose F is a substitution instance of a normal theorem N. There are three occurrences of negative proof polynomials in (5):

[6] Originally the conjecture was formulated for the full LP.

x in front of P, y in front of $P \to Q$, and $y \cdot x$ in the antecedent. Polynomials x and y are already proof variables, so we only need $y \cdot x$ to be replaced by a proof variable z; w.l.o.g. we may assume $z \notin \{x, y\}$. Then

$$N = \Big(x\!:\!P \wedge y\!:\!(P \to Q)\Big) \vee z\!:\!Q \to t\!:\!Q \tag{6}$$

such that $z\sigma = y\cdot x$ and $t\sigma = y\cdot x$ for some substitution σ and proof polynomial t. We will show that either (6) is not derivable or no substitution of t yields $y \cdot x$. Since x and y are both atomic, t must be either a proof variable z_1 or have form $z_1 \cdot z_2$ for some proof variables z_1 and z_2. In the former case, (6) cannot be a theorem because none of

$$\Big(x\!:\!P \wedge y\!:\!(P \to Q)\Big) \vee z\!:\!Q \to x\!:\!Q$$
$$\Big(x\!:\!P \wedge y\!:\!(P \to Q)\Big) \vee z\!:\!Q \to y\!:\!Q$$
$$\Big(x\!:\!P \wedge y\!:\!(P \to Q)\Big) \vee z\!:\!Q \to z\!:\!Q$$
$$\Big(x\!:\!P \wedge y\!:\!(P \to Q)\Big) \vee z\!:\!Q \to z_1\!:\!Q$$

is derivable, where $z_1 \notin \{x, y, z\}$. Similarly, none of the 16 possibilities for t having form $z_1 \cdot z_2$ works. □

In the counterexample, for N to be a theorem, term t must combine the proof by *modus ponens* from $P \to Q$ and P (represented by $y \cdot x$) with the proof by invoking the hypothesis Q (represented by z). In the absence of $+$, the only way to achieve it is to match z with $y\cdot x$. Alternatively, to represent this propositional reasoning, we could use $+$, the operation of proof composition:

$$\mathsf{PLP}^{+} \vdash \Big(x\!:\!P \wedge y\!:\!(P \to Q)\Big) \vee z\!:\!Q \to (y \cdot x + z)\!:\!Q \tag{7}$$

Note that in general, matching one derivation with another will not always be possible. Thus PLP^{-} lacks machinery to represent the method of case analysis:

Example 3. $P \vee Q$ can be derived either from P or from Q, but there is no proof polynomial t such that

$$\mathsf{PLP}^{-} \vdash x\!:\!P \vee y\!:\!Q \to t\!:\!(P \vee Q)\ .$$

This example shows that PLP^{-} does not implement all facets of derivations from hypotheses. Combinatory logic, being less expressive, is even further from performing this function.

5 Proof Composition in PLP

There are advantages and drawbacks of adding $+$ and switching to PLP^{+}. The most obvious drawback is:

Theorem 6. *There exists a tautology F with two different propositional derivations represented in* PLP^+ *by the same proof polynomial.*

Of course, we need to extend our definition of derivation representation to proof polynomials with occurrences of $+$ since all proof polynomials in Definition 4 were $\{+\}$-free. The only extension true to the meaning of $+$ is:

Definition 8. *$t + s$ represents all derivations represented by either t or s.*

Proof. Consider two distinct derivations. Find their $\{+\}$-free representations t and s. Then $\mathsf{PLP}^+ \vdash (t + s) : F$. Both derivations are represented by the proof polynomial $t + s$. □

This seems like a reincarnation of a problem pointed out in [Str05] for the sequent calculus (see also [GTL89]) in Hilbert derivations. As Straßburger puts it in [Str05], "there is no clear notion of composing proofs in classical logic." So the composition of proofs seems to be an inherent property of classical logic. Neither combinatory logic nor PLP^- had tools to express it. On the other hand, adding such a tool cost us correctness with respect to generality.

A possible way of straightening this out is to view a proof polynomial with $+$ as representing a collection of derivations rather than a single derivation. Each derivation in such a collection will once again have the most general tautology proven by it. To compare two such proof polynomials one will need to compute all the most general tautologies proven by each and to compare the resulting sets. Naturally, there will be an exponential blow-up in complexity, but the question whether two given terms represent the same set of derivations will remain decidable.

5.1 Normal Form Theorem for PLP^+

The major advantage of using PLP^+ is what we call the Normal Form Theorem for PLP^+. This is an adaptation of Bryan Renne's patch to Goris's formulation of the conjecture, which proved unsuccessful for the full LP but works for PLP^+.

Consider the normal form suggested in (7). After substituting $y \cdot x$ for z we will not get (5). The result of this substitution yields

$$\Big(x:P \wedge y:(P \to Q)\Big) \vee (y \cdot x):Q \to (y \cdot x + y \cdot x):Q \ ,$$

which, nevertheless, is very close to (5). The only difference is that $y \cdot x$ in the consequent is duplicated.

Normal Form Theorem (for PLP^+). *For each* PLP^+*-theorem G there exists a normal* PLP^+*-theorem N and a substitution σ of proof polynomials for proof variables (sentence letters remain unaffected) such that G can be obtained from $N\sigma$ by eliminating duplicate polynomials, i.e., replacing $t + t$ by t (possibly multiple times).*

The proof of this theorem (see Section 5.3) was largely inspired by Fitting's method of classical implication bases (see [Fit06]), although the method itself will not be directly used.

The only tableau system for LP in existence up until now was developed by Renne in [Ren04]. As Fitting mentions in [Fit06], for the method of classical implication bases to work, it is essential that the tableau system used be cut-free. Renne's tableau system does not have the cut rule itself. But it has the so-called β^X-rules:

$$\frac{\mathbf{F}\ \ (s\cdot t){:}Y}{\mathbf{F}\ \ s{:}(X\rightarrow Y)\ \ \Big|\ \ \mathbf{F}\ \ t{:}X}$$

These rules resemble the cut-rule in that (a) they violate the subformula property; (b) they do not respect polarities. This makes it hard to use Renne's tableaux in our proof. Hence we will adopt a different tableau system prompted by the decision algorithm from [Mkr97, Kuz00].

5.2 Tableaux for PLP^+

It is easier to devise tableaux for $\mathsf{LP}_{\mathcal{CS}}$ first and then adapt them to PLP^+. To obtain a tableau system for $\mathsf{LP}_{\mathcal{CS}}$, we add the following two rules to the standard set of propositional tableau rules:

$$\alpha\text{-rule:}\quad \frac{\mathbf{T}\ \ s{:}G}{\begin{array}{c}\mathbf{T}\ \ G\\ \mathbf{T}\ \ s\gg G\end{array}} \qquad\qquad \beta\text{-rule:}\quad \frac{\mathbf{F}\ \ s{:}G}{\mathbf{F}\ \ G\ \ \Big|\ \ \mathbf{F}\ \ s\gg G}$$

Here $s\gg G$ is a notation suggested by Renne that replaces an older notation $G\in *(s)$ from [Kuz00, Mkr97]. It roughly means that s can be thought of as a proof of G. According to Mkrtychev semantics for LP (see [Mkr97]),

$s{:}G$ is true iff both G and $s\gg G$ are true.

If this statement is recast in a tableau format, it yields the α- and β-rules above.

Closure conditions. Unlike in classical propositional tableaux, there is an additional closure condition for a branch:

Definition 9. *A branch of an* $\mathsf{LP}_{\mathcal{CS}}$*-tableau is* closed *if either*

1. *it contains both* $\boldsymbol{F}G$ *and* $\boldsymbol{T}G$ *for some formula*[7] G *or*
2. *it contains expressions* $\boldsymbol{T}s_1\gg G_1,\ldots,\boldsymbol{T}s_n\gg G_n$*, and* $\boldsymbol{F}r\gg H$ *such that*

$$s_1{:}G_1,\ldots,s_n{:}G_n\vdash_{\mathsf{rLP}_{\mathcal{CS}}} r{:}H\ .$$

Let us call a branch that is closed according to case 1 propositionally closed*; similarly, a branch that is closed according to case 2 will be called* $\gg$-closed.

A tableau is closed *if all its branches are closed.*

[7] For technical reasons, we prohibit G from having form $s\gg B$.

Soundness and completeness of this tableau system with respect to $\mathsf{LP}_{\mathcal{CS}}$ was proven in [Mkr97], although the notation was slightly different and the word "tableau" was never used.

It is well known that G is a propositional tautology iff there is an atomically closed propositional tableau for $\mathbf{F}G$, i.e., each branch of this tableau has a pair of formulas $\mathbf{F}\,P$ and $\mathbf{T}\,P$, where P is a sentence letter. The tableau system for $\mathsf{LP}_{\mathcal{CS}}$ described above enjoys a similar property:

Definition 10. *An $\mathsf{LP}_{\mathcal{CS}}$-tableau is called* pseudo-atomically closed *if a) it is closed and, in addition, b) each propositionally closed branch has a pair of formulas $\mathbf{F}\,P$ and $\mathbf{T}\,P$, where P is a sentence letter.*

Lemma 3 (Mkrtychev, [Mkr97]). *For each $\mathsf{LP}_{\mathcal{CS}}$-valid formula G there exists a pseudo-atomicaly closed tableau for $\mathbf{F}\,G$.*

To adapt this tableau system to PLP, we restrict the language to that of PLP and change the definition of a $\gg$-closed branch:

Definition 11. *A* $\mathsf{PLP}^+(\mathsf{PLP}^-)$-tableau *is an $\mathsf{LP}_{\mathcal{CS}}$-tableau in PLP-language.*
A branch of a $\mathsf{PLP}^+(\mathsf{PLP}^-)$-tableau is closed *if either*

1. *it is propositionally closed in the sense of $\mathsf{LP}_{\mathcal{CS}}$ or*
2. *it contains expressions $\boldsymbol{T}s_1 \gg G_1, \dots, \boldsymbol{T}s_n \gg G_n$, and $\boldsymbol{F}r \gg H$ such that*

$$s_1 {:} G_1, \dots, s_n {:} G_n \vdash_{\mathsf{rPLP}^+(\mathsf{rPLP}^-)} r{:}H \ .$$

Theorem 7 (Completeness Theorem for PLP-tableaux). *PLP-formula G is derivable in PLP^+ (PLP^-) iff there is a pseudo-atomically closed $\mathsf{PLP}^+(\mathsf{PLP}^-)$-tableau for $\mathbf{F}\,G$.*

Proof. Mkrtychev models for $\mathsf{LP}_{\mathcal{CS}}$ from [Mkr97] can be adapted to PLP^+ (PLP^-) in a way similar to our adaptation of the tableau system. The completeness proof from [Mkr97], which uses the maximal consistent sets construction, applies to these adapted models with cosmetic changes. Then a standard proof of tableau completeness can be used w.r.t. these adapted models. □

5.3 Proof of the Normal Form Theorem

Proof. Consider an arbitrary theorem G of PLP^+. By Completeness Theorem 7, there exists a pseudo-atomically closed tableau for $\mathbf{F}\,G$. All proof polynomials in this closed tableau can be broken into families of related occurrences:

Definition 12. *The proof polynomial r in the parent formula of an α-rule*

$$\frac{\boldsymbol{T}\ \ r{:}B}{\begin{array}{c}\boldsymbol{T}\ \ B\\ \boldsymbol{T}\ \ r \gg B\end{array}}$$

is directly related *to r in the second child. Note that B may not contain proof polynomials. Cases for the β-rule and all the propositional rules are similar. A* family *of occurrences is an equivalence class with respect to the transitive reflexive closure of this direct relationship.*

A direct consequence of this definition is the following

Lemma 4. *The simultaneous replacement of all occurrences of a proof polynomial from a given family in a given tableau by another proof polynomial produces a new tableau.*

According to the Normal Form Theorem, negative families of proof polynomials should be replaced by distinct proof variables. By Lemma 4, this will result in a new tableau for $\mathbf{F}\, N'$, where N' is the normal formula obtained from G in the course of these replacements. However, simply producing a tableau for $\mathbf{F}\, N'$ is insufficient to show that N' is a theorem of PLP^+. In order for N' to be derivable, the tableau must be closed. The counterexample from Theorem 5 shows that this may not be the case.

Each propositionally closed branch will remain propositionally closed after the replacements. Indeed, the tableau was pseudo-atomically closed, i.e., each propositionally closed branch contained a pair $\mathbf{F}\, P$ and $\mathbf{T}\, P$ for some sentence letter P. This P will not be affected by the replacements.

By contrast, $\gg$-closed branches may cease to be closed. Such a branch was closed because it contained some negatively occurring expressions $\mathbf{T}\, s_i \gg G_i$, where $i = 1, \ldots, n$ and some positively occurring expression $\mathbf{F}\, r \gg H$ such that

$$s_1 : G_1, \ldots, s_n : G_n \vdash_{\mathsf{rPLP}^+} r : H \quad . \tag{8}$$

Let us associate such a derivation with each $\gg$-closed branch. The replacements of negative families by proof variables will affect the leaves of these derivations as well as their conclusions. Leaves of the derivations correspond to negative families in the tableau; hence the replacements in the tableau will be matched in the derivations. By contrast, conclusions of the derivations correspond to positive families in the tableau, which are not affected by the replacements at this stage.

Let r' be obtained from r via the replacements. Since $\mathbf{F}\, r \gg H$ in the tableau was not replaced by $\mathbf{F}\, r' \gg H$, (8) may cease to be a derivation, so there is no guarantee that its associated branch will remain $\gg$-closed after the replacement.

We cannot replace the family of the positive occurrence of r in this $\mathbf{F}\, r \gg H$ by r' because $\mathbf{F}\, r \gg H$ may participate in several derivations that generate different r's. Let $r_1, \ldots, r_n$ be all such r's for all derivations involving $\mathbf{F}\, r \gg H$. Replace the family of this positive occurrence of r by $r_1 + \ldots + r_n$. Let N be the result of applying such additional replacements to N' for all $\gg$-closed branches. Since

$$r_i : H \vdash_{\mathsf{rPLP}^+} (r_1 + \ldots + r_n) : H \quad ,$$

each derivation of $r_i : H$ can be appended to a derivation of $(r_1 + \ldots + r_n) : H$. Thus all branches that were $\gg$-closed in the original tableau will be $\gg$-closed after these additional replacements.

We applied two series of replacements to the original closed tableau for $\mathbf{F}\, G$ and obtained a closed tableau for $\mathbf{F}\, N$, where all negative occurrences of proof polynomials are proof variables and positive occurrences have form $r_1 + \ldots + r_n$. It follows that N is a normal theorem of PLP^+. It remains to note that reversing the replacement of negative occurrences in N will bring all the negative families to the initial state, whereas in a positive family $r_1 + \ldots + r_n$ each of r_i's will be mapped back to r so that the family will take form $r + \ldots + r$. $\square$

In Theorem 2, formulas of the form $x_1 : B_1 \wedge \ldots \wedge x_n : B_n \rightarrow t : G$ were shown to be sufficient for representing all stand-alone propositional derivations from hypotheses. If the proof of the Normal Form Theorem is restricted to theorems of form $s_1 : B_1 \wedge \ldots \wedge s_n : B_n \rightarrow t : G$, a stronger normal form can be obtained. In this Horn-clause-like fragment, the original Goris's Conjecture is true:

Corollary 1. *For each* PLP^+*(*PLP^-*)-theorem* $H = s_1 : B_1 \wedge \ldots \wedge s_n : B_n \rightarrow t : G$ *there exists a normal* PLP^+*(*PLP^-*)-theorem* N *and a substitution* σ *of proof polynomials for proof variables (sentence letters remain unaffected) such that* $H = N\sigma$*.*

Proof. Consider a completed pseudo-atomically closed PLP^+(PLP^-)-tableau for $\mathbf{F}\, H$. All $\gg$-closed branches in it (if any) share the same set of $\gg$-expressions:

$$\{\mathbf{T}\, s_1 \gg B_1, \ldots, \mathbf{T}\, s_n \gg B_n, \mathbf{F}\, t \gg G\}$$

Thus one rPLP^+(rPLP^-)-derivation may be associated with all such branches. Any PLP^--tableau (rPLP^--derivation) is also a PLP^+-tableau (an rPLP^+-derivation). Hence for either theory we can apply the algorithm from the proof of the Normal Form Theorem for PLP^+. Since there is only one rPLP^+-derivation, the positive occurrences of t will be replaced by only one proof polynomial t' so that no duplicates will be introduced. Consequently, there is no need to eliminate duplicates, and $N\sigma$ already equals H. It remains to note that in the case of PLP^- the original rPLP^--derivation did not contain any $+$'s. Since t is replaced by only one t' no rule for $+$ is needed, hence the derivation remains indeed an rPLP^- one. $\square$

Acknowledgements. I am deeply grateful to my research advisor Sergei Artemov for many fruitful discussions. I also thank Evan Goris and Bryan Renne for the interest they took in the problem and for many valuable comments and suggestions. Finally, I am indebted to the anonymous referees for their insights that helped to shape the final version of the paper.

References

[And76] Peter B. Andrews. Refutations by matings. *IEEE Transactions on Computers*, 25(8):801–807, 1976.

[Art95] Sergei N. Artemov. Operational modal logic. Technical Report MSI 95–29, Cornell University, 1995.

[Art01] Sergei N. Artemov. Explicit provability and constructive semantics. *Bulletin of Symbolic Logic*, 7(1):1–36, 2001.

[Bib79] Wolfgang Bibel. Tautology testing with a generalized matrix reduction method. *Theoretical Computer Science*, 8:31–44, 1979.

[CB83] Jacques Corbin and Michel Bidoit. A rehabilitation of Robinson's unification algorithm. In R. E. A. Mason, editor, *Information Processing 83*, pages 909–914. North-Holland/IFIP, 1983.

[Fit05] Melvin Fitting. The Logic of Proofs, semantically. *Annals of Pure and Applied Logic*, 132(1):1–25, 2005.

[Fit06] Melvin Fitting. Realizing substitution instances of modal theorems. Unpublished, available at `http://comet.lehman.cuny.edu/fitting/`, 2006.

[Göd95] Kurt Gödel. Vortrag bei Zilsel (*1938a). In Solomon Feferman, John W. Dawson, Jr., Warren Goldfarb, Charles Parsons, and Robert N. Solovay, editors, *Kurt Gödel Collected Works*, volume III, pages 86–113. Oxford University Press, 1995.

[GTL89] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

[Hin69] R. Hindley. The principal type-scheme of an object in combinatory logic. *Transactions of the American Mathematical Society*, 146:29–60, 1969.

[Kru06] Nikolai V. Krupski. On the complexity of the reflected logic of proofs. *Theoretical Computer Science*, 357(1–3):136–142, 2006.

[Kuz00] Roman Kuznets. On the complexity of explicit modal logics. In Peter Clote and Helmut Schwichtenberg, editors, *CSL 2000*, volume 1862 of *Lecture Notes in Computer Science*, pages 371–383. Springer, 2000.

[Lam68] Joachim Lambek. Deductive systems and categories I. Syntactic calculus and residuated categories. *Mathematical Systems Theory*, 2(4):287–318, 1968.

[Lam69] Joachim Lambek. Deductive systems and categories II. Standard constructions and closed categories. In *Category Theory, Homology Theory and their Applications I*, volume 86 of *Lecture Notes in Mathematics*, pages 76–122. Springer, 1969.

[Lam72] Joachim Lambek. Deductive systems and categories III. Cartesian closed categories, intuitionist propositional calculus, and combinatory logic. In *Toposes, Algebraic Geometry and Logic*, volume 274 of *Lecture Notes in Mathematics*, pages 57–82. Springer, 1972.

[LS05] François Lamarche and Lutz Straßburger. Naming proofs in classical propositional logic. In Pawel Urzyczyn, editor, *TLCA 2005*, volume 3461 of *Lecture Notes in Computer Science*, pages 246–261. Springer, 2005.

[Mil07] Robert Milnikel. Derivability in certain subsystems of the Logic of Proofs is Π_2^p-complete. *Annals of Pure and Applied Logic*, 145(3):223–239, 2007.

[Mkr97] Alexey Mkrtychev. Models for the Logic of Proofs. In Sergei I. Adian and Anil Nerode, editors, *LFCS 1997*, volume 1234 of *Lecture Notes in Computer Science*, pages 266–275. Springer, 1997.

[Nou06] Karim Nour. Classical combinatory logic. In René David, Danièle Gardy, Pierre Lescanne, and Marek Zaionc, editors, *Computational Logic and Applications 2005*, pages 87–96. DMTCS proc. AF, 2006.

[Pra71] Dag Prawitz. Ideas and results in proof theory. In J. E. Fenstad, editor, *Proceedings of the 2nd Scandinavian Logic Symposium*, volume 63 of *Studies in Logic and the Foundations of Mathematics*, pages 235–307. North-Holland, 1971.

[Ren04] Bryan Renne. Tableaux for the Logic of Proofs. Technical Report TR–2004001, CUNY Ph.D. Program in Computer Science, 2004.

[Str05] Lutz Straßburger. What is a logic, and what is a proof? In Jean-Yves Beziau, editor, *Logica Universalis*, pages 135–145. Birkäuser, 2005.

[TW02] Ruediger Thiele and Larry Wos. Hilbert's twenty-fourth problem. *Journal of Automated Reasoning*, 29(1):67–89, 2002.

On the Constructive Dedekind Reals: Extended Abstract

Robert S. Lubarsky[1] and Michael Rathjen[2,⋆]

[1] Department of Mathematical Sciences, Florida Atlantic University
rlubarsk@fau.edu
[2] Department of Pure Mathematics, University of Leeds
rathjen@math.ohio-state.edu

Abstract. In order to built the collection of Cauchy reals as a set in constructive set theory, the only Power Set-like principle needed is Exponentiation. In contrast, the proof that the Dedekind reals form a set has seemed to require more than that. The main purpose here is to show that Exponentiation alone does not suffice for the latter, by furnishing a Kripke model of constructive set theory, CZF with Subset Collection replaced by Exponentiation, in which the Cauchy reals form a set while the Dedekind reals constitute a proper class.

1 Introduction

In classical mathematics, one principal approach to defining the real numbers is to use equivalence classes of Cauchy sequences of rational numbers, and the other is the method of Dedekind cuts wherein reals appear as subsets of $\mathbb{Q}$ with special properties. Classically the two methods are equivalent in that the resulting field structures are easily shown to be isomorphic. As often happens in an intuitionistic setting, classically equivalent notions fork. Dedekind reals give rise to several demonstrably different collections of reals when only intuitionistic logic is assumed (cf. [17], Ch.5, Sect.5). Here we shall be concerned with the most common and fruitful notion of Dedekind real which crucially involves the (classically superfluous) condition of locatedness of cuts. These Dedekind reals are sometimes referred to as the *constructive Dedekind reals* but we shall simply address them as the *Dedekind reals*. Even in intuitionistic set theory, with a little bit of help from the countable axiom of choice ($\mathbf{AC}(\mathbb{N},2)$[1] suffices; see [4], 8.25), $\mathbb{R}^d$ and $\mathbb{R}^c$ are isomorphic (where $\mathbb{R}^d$ and $\mathbb{R}^c$ denote the collections of Dedekind reals and Cauchy reals, respectively). As $\mathbb{R}^c$ is canonically embedded in $\mathbb{R}^d$ we can view $\mathbb{R}^c$ as a subset of $\mathbb{R}^d$ so that the latter result can be stated as $\mathbb{R}^d = \mathbb{R}^c$. The countable axiom of choice is accepted in Bishop-style constructive mathematics but cannot be assumed in all intuitionistic contexts. Some choice is necessary for equating $\mathbb{R}^d$ and $\mathbb{R}^c$ as there are sheaf models of higher order intuitionistic logic in which $\mathbb{R}^d$ is not isomorphic to $\mathbb{R}^c$ (cf. [6]). This paper will

⋆ This material is based upon work supported by the National Science Foundation under Award No. DMS-0301162.

[1] $\forall r \subseteq \mathbb{N} \times 2[\forall n \in \mathbb{N}\, \exists i \in \{0,1\}\ \langle n,i\rangle \in r\ \rightarrow\ \exists f : \mathbb{N} \rightarrow 2\ \forall n \in \mathbb{N}\ \langle n, f(n)\rangle \in r]$.

show that the difference between $\mathbb{R}^d$ and $\mathbb{R}^c$ can be of a grander scale. When is the continuum a set? The standard, classical construction of $\mathbb{R}$ as a set uses Power Set. Constructively, the weaker principle of Subset Collection (in the context of the axioms of Constructive Zermelo-Fraenkel Set Theory CZF) suffices, as does even the apparently even weaker principle of Binary Refinement [5]. In contrast, we shall demonstrate that there is a Kripke model of CZF with Exponentiation in lieu of Subset Collection in which the Cauchy reals form a set while the Dedekind reals constitute a proper class. This shows that Exponentiation and Subset Collection Axiom have markedly different consequences for the theory of Dedekind reals.

This paper proves the following theorems:

Theorem 1. *(Fourman-Hyland [6]) IZF$_{Ref}$ does not prove that the Dedekind reals equal the Cauchy reals.*

Theorem 2. *CZF$_{Exp}$ (i.e. CZF with Subset Collection replaced by Exponentiation) does not prove that the Dedekind reals are a set.*

Even though the proof of the first theorem given here has the flavor of the original Fourman-Hyland proof of the same, it is still included because it is not obvious (to us at least) how their sheaf proof could be converted to the current Kripke model construction, which might therefore be of independent interest. In addition, it is helpful as background to understand the construction of the second proof, which we do not know how to convert to a purely topological or sheaf-theoretic argument.

The paper is organized as follows. After a brief review of Constructive Zermelo-Fraenkel Set Theory and notions of real numbers, section 2 features a Kripke model of IZF_{Ref} in which $\mathbb{R}^d \neq \mathbb{R}^c$. Here IZF_{Ref} denotes Intuitionistic Zermelo-Fraenkel Set Theory with the Reflection schema.[2]

In section 3 the model of section 2 undergoes refinements and pivotally techniques of [8] are put to use to engender a model of CZF_{Exp} in which $\mathbb{R}^d$ is a proper class.

1.1 Constructive Zermelo-Fraenkel Set Theory

In this subsection we will summarize the axioms for CZF. For more detail, see [1], [2], [3], or [4]. Among the non-logical axioms of CZF are *Extensionality*, *Pairing* and *Union* in their usual forms. CZF has additionally axiom schemata which we will now proceed to summarize.

Infinity: $\exists x \forall u \big[u \in x \leftrightarrow \big(\emptyset = u \ \vee \ \exists v \in x\ u = v+1\big)\big]$ where $v+1 = v \cup \{v\}$.

Set Induction: $\forall x[\forall y \in x \phi(y) \rightarrow \phi(x)] \rightarrow \forall x \phi(x)$

Bounded Separation: $\forall a \exists b \forall x[x \in b \leftrightarrow x \in a \wedge \phi(x)]$

[2] Reflection, Collection and Replacement are equivalent in classical set theory. Intuitionistically, Reflection implies Collection which in turn implies Replacement, however, these implications cannot be reversed (see [7] for the latter).

for all *bounded* formulae ϕ. A set-theoretic formula is *bounded* or *restricted* if it is constructed from prime formulae using $\neg, \wedge, \vee, \rightarrow, \forall x \in y$ and $\exists x \in y$ only.

Strong Collection: For all formulae ϕ,

$$\forall a\big[\forall x \in a \exists y \phi(x,y) \ \rightarrow\ \exists b\,[\forall x \in a\,\exists y \in b\,\phi(x,y) \wedge \forall y \in b\,\exists x \in a\,\phi(x,y)]\big].$$

Subset Collection: For all formulae ψ,

$$\begin{aligned}&\forall a \forall b \exists c \forall u\,\big[\forall x \in a\,\exists y \in b\ \psi(x,y,u) \ \rightarrow \\ &\quad \exists d \in c\,[\forall x \in a\,\exists y \in d\,\psi(x,y,u) \wedge \forall y \in d\,\exists x \in a\,\psi(x,y,u)]\big].\end{aligned}$$

An additional axiom we shall consider is:

Exponentiation: $\forall x \forall y \exists z\, z = {}^{x}y$.

Proposition 1. $CZF \vdash$ Exponentiation.

Proof. [1], Proposition 2.2. □

1.2 The Cauchy and Dedekind Reals

Definition 1. *A* fundamental sequence *is a sequence* $(r_n)_{n\in\mathbb{N}}$ *of rationals, together with is a* (Cauchy-)modulus $f : \mathbb{N} \rightarrow \mathbb{N}$ *such that*

$$\forall k\, \forall mn \geq f(k)\ \ |r_m - r_n| < \frac{1}{2^k},$$

where all quantifiers range over $\mathbb{N}$.

Two fundamental sequences $(r_n)_{n\in\mathbb{N}}, (s_n)_{n\in\mathbb{N}}$ *are said to coincide (in symbols* $\approx$*) if*

$$\forall k \exists n \forall m \geq n\ \ |r_m - s_m| < \frac{1}{2^k}.$$

$\approx$ *is indeed an equivalence relation on fundamental sequences. The set of Cauchy reals* $\mathbb{R}^c$ *consists of the equivalence classes of fundamental sequences relative to* $\approx$*. For the equivalence class of* $(r_n)_{n\in\mathbb{N}}$ *we use the notation* $(r_n)_{n\in\mathbb{N}}/\approx$*.*

The development of the theory of Cauchy reals in [17], Ch.5, Sect.2-4 can be carried out on the basis of CZF_{Exp}. Note that the axiom AC-NN! [3] is deducible in CZF_{Exp}.

Definition 2. *Let* $S \subseteq \mathbb{Q}$. S *is called a* left cut *or* Dedekind real *if the following conditions are satisfied:*

1. $\exists r(r \in S)\ \wedge\ \exists r'(r' \notin S)$ (boundedness)
2. $\forall r \in S\, \exists r' \in S\,(r < r')$ (openness)
3. $\forall rs \in \mathbb{Q}\,[r < s \rightarrow r \in S \vee s \notin S]$ (locatedness)

$\mathbb{R}^d$ is the set of Dedekind reals.

Lemma 1. $\mathbb{R}^c$ *is a subfield of* $\mathbb{R}^d$.

Proof: Exercise or see [4], Section 8.4. □

[3] $(\forall m \in \mathbb{N}\, \exists! n \in \mathbb{N}\, \phi(m,n)) \rightarrow (\exists f : \mathbb{N} \rightarrow \mathbb{N}\, \forall m \in \mathbb{N}\, \phi(m, f(m)))$.

2 $\mathbb{R}^d \neq \mathbb{R}^c$

Theorem 3. *(Fourman-Hyland [6]) IZF_{Ref} does not prove that the Dedekind reals equal the Cauchy reals.*

2.1 Construction of the Model

Let $M_0 \prec M_1 \prec ...$ be an ω-sequence of models of ZF set theory and of elementary embeddings among them, as indicated, such that the sequence from M_n on is definable in M_n, and such that each thinks that the next has non-standard integers. Notice that this is easy to define (mod getting a model of ZF in the first place): an iterated ultrapower using any non-principal ultrafilter on ω will do. The reader is invited to think of each model as being countable, in case that makes it easier to think about. We will ambiguously use the symbol f to stand for any of the elementary embeddings inherent in the M_n-sequence.

Definition 3. *The underlying partial order of the various Kripke models M will be a rooted tree with ω-many levels; the bottom node (level 0) will be notated by $\perp$, and the branching at a node of level n will be of size continuum in the sense of M_{n+1}. (For those thinking of each model as being countable, this makes the p.o. $\omega^{<\omega}$, that is, the countably branching tree.)*

Definition 4. *The ground Kripke model has, at each node of level n, a copy of M_n. The transition functions (from a node to a following node) are the elementary embeddings given with the original sequence of models (and therefore will be notated by f again).*

Note that by the elementarity of the extensions, this Kripke model is a model of classical ZF. More importantly, the model restricted to any node of level n is definable in M_n, because the original M-sequence was so definable.

The final model M will be an extension of the ground model that will be described like a forcing extension. That is, M will consist of (equivalence classes of) the terms from the ground model.

Definition 5. *The terms are defined at each node separately, inductively on the ordinals in that model. At any stage α, a term of stage α is a set σ of the form $\{\langle \sigma_i, J_i \rangle \mid i \in I\}$, where I is some index set, each σ_i is a term of stage $< \alpha$, and each J_i is an open subset of the real line.*

As a condition, each open set J is saying "the real in question is in me". For each node of level n there will be an associated real r (in the sense of M_n) such that at that node the true J's will be those containing r. The assignment of the r's will happen shortly, when the motivation should be clear.

Note in any case that all sets from the ground model have canonical names, by choosing each J_i to be the whole real line, hereditarily.

Notice also that the definition of the terms given above will be interpreted differently at each node of the ground Kripke model, as the reals change from node to node. However, any term at a node gets sent by the transition function f to a corresponding term at any given later node. The definitions given later, such as the forcing relation $\Vdash$, are all interpretable in each M_n, and coherently so, via the elementary embeddings.

To make this precise, we need to define the primitive relations at each node, $=_M$ and $\in_M$ (the subscript being used to prevent confusion with equality and membership of the ambient models M_n). This will be done via a forcing relation $\Vdash$.

Definition 6. *$J \Vdash \sigma =_M \tau$ iff for all $\langle \sigma_i, J_i \rangle \in \sigma$ $J \cap J_i \Vdash \sigma_i \in_M \tau$ and vice versa*

$J \Vdash \sigma \in_M \tau$ iff for all $r \in J$ there is a $\langle \tau_i, J_i \rangle \in \tau$ and $J' \subseteq J$ such that $r \in J' \cap J_i \Vdash \sigma =_M \tau_i$

(We will later extend this forcing relation to all formulas.)

Definition 7. *At a node (with associated real r, as described below), for any two terms σ and τ, $\sigma =_M \tau$ iff, for some J with $r \in J, J \Vdash \sigma =_M \tau$.*

Also, $\sigma \in_M \tau$ iff for some J with $r \in J, J \Vdash \sigma \in_M \tau$.

Thus we have a first-order structure at each node.

The transition functions are the same as before. That is, if σ is an object at a node, then it's a term, meaning in particular it's a set in some M_n. Any later node has for its universe the terms from some $M_m, m \geq n$. With f the elementary embedding from M_n to M_m, f can also serve as the transition function between the given nodes. These transition functions satisfy the coherence conditions necessary for a Kripke model.

To have a Kripke model, f must also respect $=_M$ and $\in_M$; in other words, f must be an $=_M$- and $\in_M$-homomorphism (i.e. $\sigma =_M \tau \rightarrow f(\sigma) =_M f(\tau)$, and similarly for $\in_M$). In order for these to be true, we need an additional restriction on the model. By way of motivation, one requirement is, intuitively speaking, that the sets σ can't shrink as we go to later nodes. That is, once σ_i gets into σ at some node, it can't be thrown out at a later node. σ_i gets into σ because $r \in J_i$ (where $\langle \sigma_i, J_i \rangle \in \sigma$). So we need to guarantee that if $r \in J$ and r' is associated to any extending node then $r' \in J$ for any open set J. This holds exactly when r' is infinitesimally close to r. We henceforth take this as an additional condition on the construction: once r is associated to a node, then any r' associated to an extending node must be infinitesimally close to r. It will actually turn out to be necessary to include all such r' as immediate successors to r.

Definition 8. *The association of reals to tree nodes: Associate 0 to the bottom node. Inductively up the tree, at a node of level n labeled with some real $r \in M_n$, extend the labeling by bijecting r's immediate successors in the tree with the set of reals in M_{n+1} infinitesimally close to r.*

Lemma 2. *f is an $=_M$ and $\in_M$-homomorphism.*

We can now conclude that we have a Kripke model.

Lemma 3. *This Kripke model satisfies the equality axioms:*

1. $\forall x\, x = x$
2. $\forall x, y\, x = y \rightarrow y = x$
3. $\forall x, y, z\, x = y \wedge y = z \rightarrow x = z$
4. $\forall x, y, z\, x = y \wedge x \in z \rightarrow y \in z$
5. $\forall x, y, z\, x = y \wedge z \in x \rightarrow z \in y$.

Proof. 1: It is easy to show with a simultaneous induction that, for all J and σ, $J \Vdash \sigma =_M \sigma$, and, for all $\langle \sigma_i, J_i \rangle \in \sigma$, $J \cap J_i \Vdash \sigma_i \in_M \sigma$.

2: Trivial because the definition of $J \Vdash \sigma =_M \tau$ is itself symmetric.

3: For this and the subsequent parts, we need some lemmas.

Lemma 4. *If J' $\subseteq$ J $\Vdash \sigma =_M \tau$ then J' $\Vdash \sigma =_M \tau$, and similarly for $\in_M$.*

Lemma 5. *If $J \Vdash \rho =_M \sigma$ and $J \Vdash \sigma =_M \tau$ then $J \Vdash \rho =_M \tau$.*

These can be proven by an induction on terms. Returning to proving property 3, the hypothesis is that for some J and K containing r, $J \Vdash \rho =_M \sigma$ and $K \Vdash \sigma =_M \tau$. By the first lemma, $J \cap K \Vdash \rho =_M \sigma, \sigma =_M \tau$, and so by the second, $J \cap K \Vdash \rho =_M \tau$, which suffices.

4: Let $J \Vdash \rho =_M \sigma$ and $K \Vdash \rho \in_M \tau$. We will show that $J \cap K \Vdash \sigma \in_M \tau$. Let $r \in J \cap K$. By hypothesis, let $\langle \tau_i, J_i \rangle \in \tau$, $J' \subseteq K$ be such that $r \in J' \cap J_i \Vdash \rho =_M \tau_i$; WLOG $J' \subseteq J$. By the first lemma, $J' \cap J_i \Vdash \rho =_M \sigma$, and by the second, $J' \cap J_i \Vdash \sigma =_M \tau_i$.

5: Similar, and left to the reader.

With this lemma in hand, we can now mod out by $=_M$, so that the symbol "=" is interpreted as actual set-theoretic equality. We will henceforth drop the subscript $_M$ from $=$ and $\in$, although we will not distinguish notationally between a term σ and the model element it represents, σ's equivalence class.

By way of notation, a node will be named by its associated real. Hence "$r \models \phi$" means ϕ holds at the node with real r. Strictly speaking there is an ambiguity here, as each node with real r has a successor node with real r. However, by elementarity, anything we can express in set theory true at one node will be true at the other, so this should lead to no confusion.

Note that, at any node of level n, the choice of r's from that node on is definable in M_n. This means that the evaluation of terms (at and beyond the given node) can be carried out over M_n, and so the Kripke model (from the given node on) can be defined over M_n, truth predicate and all.

2.2 The Forcing Relation

Which J's count as true determines the interpretation of all terms, and hence of truth in the end model. We need to get a handle on this. As with forcing, we need a relation $J \Vdash \phi$, which holds at r iff $r \models \phi$. Note that, by elementarity, it doesn't matter in which classical model M_n or at what node in the ground Kripke model $\Vdash$ is being interpreted (as long as the parameters are in the interpreting model, of course).

Definition 9. *$J \Vdash \phi$ is defined inductively on ϕ:*
$J \Vdash \sigma = \tau$ and $J \Vdash \sigma \in \tau$ are as above
$J \Vdash \phi \wedge \psi$ iff $J \Vdash \phi$ and $J \Vdash \psi$
$J \Vdash \phi \vee \psi$ iff for all $r \in J$ there is a J' containing r such that J' $\Vdash \phi$ or J' $\Vdash \psi$
$J \Vdash \phi \rightarrow \psi$ iff for all J' $\subseteq J$ if J' $\Vdash \phi$ then J' $\Vdash \psi$
$J \Vdash \exists x\ \phi(x)$ iff for all $r \in J$ there is a J' containing r and a σ such that $J \cap J' \Vdash \phi(\sigma)$
$J \Vdash \forall x\ \phi(x)$ iff for all $r \in J$ and σ there is a J' containing r such that $J \cap J' \Vdash \phi(\sigma)$

Lemma 6. *1. For all ϕ $\emptyset \Vdash \phi$.*
2. If J' $\subseteq J \Vdash \phi$ then J' $\Vdash \phi$.
3. If $J_i \Vdash \phi$ for all i then $\bigcup_i J_i \Vdash \phi$.
4. $J \Vdash \phi$ iff for all $r \in J$ there is a J' containing r such that $J \cap$ J' $\Vdash \phi$.
5. Truth Lemma: For any node r, $r \models \phi$ iff $J \Vdash \phi$ for some J containing r.

Proof. Each part either is by induction or follows trivially from an earlier part.

2.3 The Final Proof

Theorem 4. *This Kripke model satisfies IZF_{Ref}.*

Proof. Infinity, Pairing, and Union are easy.

- Extensionality: We need to show that $\forall x\ \forall y\ [\forall z\ (z \in x \leftrightarrow z \in y) \rightarrow x = y]$. So let σ and τ be any terms at a node r such that $r \models$ "$\forall z\ (z \in \sigma \leftrightarrow z \in \tau)$". We must show that $r \models$ "$\sigma = \tau$". By the Truth Lemma, let $r \in J \Vdash$ "$\forall z\ (z \in \sigma \leftrightarrow z \in \tau)$"; i.e. for all $r' \in J, \rho$ there is a J' containing r' such that $J \cap J' \Vdash \rho \in \sigma \leftrightarrow \rho \in \tau$. We claim that $J \Vdash$ "$\sigma = \tau$", which again by the Truth Lemma suffices. To this end, let $\langle \sigma_i, J_i \rangle$ be in σ; we need to show that $J \cap J_i \Vdash \sigma_i \in \tau$. Let r' be an arbitrary member of $J \cap J_i$ and ρ be σ_i. By the choice of J, let J' containing r' be such that $J \cap J' \Vdash \sigma_i \in \sigma \leftrightarrow \sigma_i \in \tau$; in particular, $J \cap J' \Vdash \sigma_i \in \sigma \rightarrow \sigma_i \in \tau$. It has already been observed in 3, part 1, that $J \cap J' \cap J_i \Vdash \sigma_i \in \sigma$, so $J \cap J' \cap J_i \Vdash \sigma_i \in \tau$. By going through each r' in $J \cap J_i$ and using 6, part 3, we can conclude that $J \cap J_i \Vdash \sigma_i \in \tau$, as desired. The other direction ("$\tau \subseteq \sigma$") is analogous.
- Set Induction (Schema): Suppose $r \models$ "$\forall x\ ((\forall y \in x\ \phi(y)) \rightarrow \phi(x))$"; by the Truth Lemma, let J containing r force as much. We must show $r \models$ "$\forall x\ \phi(x)$". Suppose not. Using the definition of satisfaction in Kripke models, there is an r' extending (i.e. infinitesimally close to) r (hence in J) and a σ such that $r' \not\models \phi(\sigma)$. By elementarity, there is such an r' in M_n, where n is the level of r. Let σ be such a term of minimal V-rank among all r's $\in J$. Fix such an r'. By the Truth Lemma (and the choice of J), $r' \models$ "$(\forall y \in \sigma\ \phi(y)) \rightarrow \phi(\sigma)$". We claim that $r' \models$ "$\forall y \in \sigma\ \phi(y)$". If not, then for some r'' extending r' (hence in J) and $\tau, r'' \models \tau \in \sigma$ and $r'' \not\models \phi(\tau)$. Unraveling the interpretation of $\in$, this choice of τ can be substituted by a term τ of lower

V-rank than σ. By elementarity, such a τ would exist in M_n, in violation of the choice of σ, which proves the claim. Hence $r' \models \phi(\sigma)$, again violating the choice of σ. This contradiction shows that $r \models$ "$\forall x\ \phi(x)$".
– Separation (Schema): Let $\phi(x)$ be a formula and σ a term. Then $\{\langle \sigma_i, J \cap J_i\rangle \mid \langle \sigma_i, J_i\rangle \in \sigma \text{ and } J \Vdash \phi(\sigma_i)\}$ will do.
– Power Set: A term $\hat{\sigma}$ is a canonical subset of σ if for all $\langle \sigma_i, \hat{J}_i\rangle \in \hat{\sigma}$ there is a $J_i \supseteq \hat{J}_i$ such that $\langle \sigma_i, J_i\rangle \in \sigma$. $\{\langle \hat{\sigma}, \mathbb{R}\rangle \mid \hat{\sigma}$ is a canonical subset of $\sigma\}$ is a set (in M_n), and will do.
– Reflection (Schema): Recall that the statement of Reflection is that for every formula $\phi(x)$ (with free variable x and unmentioned parameters) and set z there is a transitive set Z containing z such that Z reflects the truth of $\phi(x)$ in V for all $x \in Z$. So to this end, let $\phi(x)$ be a formula and σ be a set at a node r of level n (in the tree which is this Kripke model's partial order). Let k be such that the truth of $\phi(x)$ at node r and beyond is Σ_k definable in M_n. In M_n, let X be a set containing σ, r, and ϕ's parameters such that $X \prec_k M_n$. Let τ be $\{\langle \rho, \mathbb{R}\rangle \mid \rho \in X \text{ is a term}\}$. τ will do.

We are interested in the canonical term $\{\langle r, J\rangle \mid r$ is a rational, J is an open interval from the reals, and $r < J\}$, where $r < J$ if r is less than each element of J. We will call this term X. Note that at node r (of level n), every standard (in the sense of M_n) real less than r gets into X, and no standard real greater than r will ever get into X. Of course, non-standard reals infinitesimally close to r are still up for grabs.

Proposition 2. $\perp \models$ *"X is a constructive Dedekind real".*

Proof. First off, $\perp \models$ "$-1 \in X \wedge 1 \notin X$". Secondly, if $r \models$ "$s < t \in X$", then $\langle t, J\rangle \in X$, where $t < J$ and $r \in J$. Hence $s < J$, so $\langle s, J\rangle \in X$, and $r \models s \in X$. Finally, suppose $r \models$ "$s, t \in \mathbf{Q} \wedge s < t$". Either $s < r$ or $r < t$. Since s and t are both standard (in the sense of M_n, n the level of r), either $r \models s \in X$ or $r \models t \notin X$ respectively.

Proposition 3. $\perp \models$ *"The Dedekind real X is not a Cauchy real."*

Proof. Recall that a Cauchy real is a Dedekind real Y for which there are two functions, f_0 and f_1, with domain $\mathbb{N}$, such that f_0 is a sequence of rationals and f_1 is a modulus of convergence (i.e. for $n, m > f_1(k)$, $f_0(n)$ and $f_0(m)$ are within 2^{-k} of each other), and $r \in Y$ iff $r < f_0(f_1(k)) - 2^{-k}$ for some k.

For notational ease we will restrict attention to what happens at node $\perp$. The argument carries over mutatis mutandis to all other nodes.

Suppose $\perp \models$ "f_0 and f_1 witness that X is a Cauchy real". By the Truth Lemma, there is an open set J containing 0 forcing the same. There are two cases.

CASE I: There is some open set J' containing 0 forcing a value $f_0(n)$ for each (standard) integer n. In this case, f_0 is a ground model function; that is, in M_0, hence in each M_n, $g(n)$ can be defined as the unique m such that $J' \Vdash f_0(n) = m$, and then f_0 is the internalization of g. Since classical logic holds in the ground model, either f_0 is bounded away from 0, say by 2^{-k}, or it's not. If it is, then for a contradiction f_0 can't witness that X is a Cauchy real. For indices n past $f_1(k)$, either $f_0(n) < -2^{-k}$ for all such n or $f_0(n) > 2^{-k}$ for all such n. In the former case, f_0 will never put -2^{-k} itself into X, even though $\perp \models -2^{-k} \in X$; in the latter, $f_0(f_1(k+1))$ would put $2^{-(k+1)}$ into X, even though $\perp \models 2^{-(k+1)} \notin X$. If on the other hand f_0 is not bounded away from 0, then the condition "$r < f_0(f_1(k)) - 2^{-k}$ for some k" becomes simply "$r < 0$". So then f_0, f_1 would witness that $r \in X$ iff $r < 0$. But this is false: if r is an infinitesimal less than 0, then $r \models r < 0$ but $r \not\models r \in X$, and if r is an infinitesimal greater than 0, then $r \models r/2 > 0$ but $r \models r/2 \in X$.

CASE II: Not case I. That means that for any interval J' around 0, however small, there is some argument n to f_0 such that J' does not force any value $f_0(n)$. By elementarity, in M_1 pick J' to be some infinitesimally small neighborhood around 0, and n such an argument. Pick some value q that $f_0(n)$ could have and the maximal (hence non-empty, proper, and open) subset of J' forcing $f_0(n) = q$. Pick the maximal (hence non-empty, proper, and open) subset of J' forcing $f_0(n) \neq q$. These two subsets must be disjoint, lest the intersection force a contradiction. But an open interval cannot be covered by two disjoint, non-empty open sets. Hence there is an infinitesimal r in neither of those two subsets. Now consider the Kripke model at node r. $f_0(n)$ is undefined at r. Otherwise, by the Truth Lemma, there would be some interval J containing r such that $J \Vdash f(n) = p$ for some particular rational p. Whether or not $p = q$ would force r into one of the subsets or the other. Therefore, the node $\perp$ cannot force that f_0 is total, and so f_0 does not witness that X is a Cauchy real.

3 The Dedekind Reals Are Not a Set

Theorem 5. *CZF_{Exp} (i.e. CZF with Subset Collection replaced by Exponentiation) does not prove that the Dedekind reals form a set.*

3.1 The Construction

Any model showing what is claimed must have certain properties. For one, the Dedekind reals cannot equal the Cauchy reals (since CZF_{Exp} proves that the Cauchy reals are a set). Hence the current model takes its inspiration from the previous one. Also, it must falsify Subset Collection (since CZF proves that the Dedekind reals are a set). Hence guidance is also taken from [8], where such a model is built.

The idea behind the latter is that a relation keeps on being introduced into the model but at a later node disappears. Since this relation is chosen so that it doesn't help build any functions, it can be ignored when proving Exponentiation. On the other hand, while you're free to include this relation in an alleged full set of relations, that relation that you choose is gone by the next node, so when it reappears later your attempt at a full set no longer works.

In the present context, we will do something similar. The troublesome relation will be (essentially) the Dedekind real X from the previous construction. It will "disappear" in that, instead of continuing to change its mind about what it is at all future nodes, it will settle down to one fixed, standard real at all next nodes. But then some other real just like X will appear and pull the same stunt.

We now begin with the definition of the Kripke model, which ultimately is distributed among the next several definitions.

Definition 10. *The underlying p.o. of the Kripke model is the same as above: a tree with height ω and continuum-in-the-sense-of-M_0 many nodes at each level. There is a bijection between the nodes of height n and the reals of M_n. A node will be identified with its corresponding real. s of height n+1 is an immediate successor of r of height n iff s is infinitesimally close to r.*

Definition 11. *A term at a node of height n is a set of the form $\{\langle \sigma_i, J_i\rangle \mid i \in I\} \cup \{\langle \sigma_h, r_h\rangle \mid h \in H\}$, where each σ is (inductively) a term, each J an open set of reals, each r a real, and H and I index sets, all in the sense of M_n.*

The first part of each term is as in the first theorem: at node r, J_i counts as true iff $r \in J_i$. The second part plays a role only when we decide to have the term settle down and stop changing. This settling down in described as follows.

Definition 12. *For a term σ and real $r \in M_n$, σ^r is defined inductively in M_n on the terms as $\{\langle \sigma_i^r, \mathbb{R}\rangle \mid \langle \sigma_i, J_i\rangle \in \sigma \wedge r \in J_i\} \cup \{\langle \sigma_h^r, \mathbb{R}\rangle \mid \langle \sigma_h, r\rangle \in \sigma\}$.*

Note that σ^r is a canonical term, meaning that it's the canonical image of a set from the ambient universe. It bears observation that $(\sigma^r)^s = \sigma^r$.

What determines when a term settles down in this way is the transition function. In fact, from any node to an immediate successor, there will be two transition functions, one the embedding f as before and the other the settling down function. This fact of the the current construction does not quite jive with the standard definition of a Kripke model, which has no room for alternate ways to go from one node to another. However, this move is standard (even tame) for categorical models, which allow for arbitrary arrows among objects. So while the standard categorical description of a partial order is a category where the objects are the elements of the order and there's an arrow from p to q iff $p \leq q$, the category we're working in has two arrows from p to q (for immediate successors). If you're still uncomfortable with this double arrow, or object to calling this object a Kripke model, then double not the arrows but the nodes. That is, replace each node s by two nodes s_{old} and s_{new}, and have the two arrows go to these two separate nodes. Now you have a very traditional Kripke model again. To save on subscripts, we will work instead with two arrows going from r to s.

Definition 13. *If s is an immediate successor of r, then there are two transition functions from r to s, called f and g. f is the elementary embedding from M_n to M_{n+1} as applied to terms. $g(\sigma) = f(\sigma)^s$. Transition functions to non-immediate successors are arbitrary compositions of the immediate transition functions.*

When considering $g(\sigma)$, note that $\sigma \in M_n$ and $s \in M_{n+1}$. However, for purposes other than the transition functions, we will have occasion to look at σ^s for both σ and s from M_n. In this case, please note that, since f is an elementary embedding, $(f(\sigma))^s = f(\sigma^s)$.

It's easy to see that for σ canonical $f(\sigma)$ is also canonical, and for τ canonical (such as $f(\sigma)$) so is τ^r. Hence in this case $f(\sigma) = g(\sigma)$.

We do not need to show that the transition functions are well-defined, since they are defined on terms and not on equivalence classes of terms. However, once we define $=$, we will show that $=$ is an equivalence relation and that f and g respect $=$, so that we can mod out by $=$ and still consider f and g as acting on these equivalence classes.

Definition 14. *$J \Vdash \sigma =_M \tau$ iff for all $\langle \sigma_i, J_i \rangle \in \sigma$ $J \cap J_i \Vdash \sigma_i \in_M \tau$ and for all $r \in J$ $\sigma^r = \tau^r$, and vice versa.*

$J \Vdash \sigma \in_M \tau$ iff for all $r \in J$ there is a $\langle \tau_i, J_i \rangle \in \tau$ and J' $\subseteq$ J such that $r \in J' \cap J_i \Vdash \sigma =_M \tau_i$, and for all $r \in J$ $\langle \sigma^r, \mathbf{R} \rangle \in \tau^r$.

Definition 15. *At a node r, for any two terms σ and τ, $r \models \sigma =_M \tau$ iff, for some J with $r \in J$, $J \Vdash \sigma =_M \tau$.*

Also, $r \models \sigma \in_M \tau$ iff for some J with $r \in J$, $J \Vdash \sigma \in_M \tau$.

Thus we have a first-order structure at each node.

Corollary 1. *The model just defined is a Kripke model. That is, the transition functions are $=_M$ and $\in_M$-homomorphisms.*

Lemma 7. *This Kripke model satisfies the five equality axioms (as in lemma 3).*

With this lemma in hand, we can now mod out by $=_M$ at each node, and have a model in which equality is actually $=$.

3.2 The Forcing Relation

In the first proof, it didn't matter where $\Vdash$ was interpreted, whether in the Kripke model or at any M_n, by elementarity. Here, in contrast, the transition functions are not the elementary embeddings. So we must be more careful about where the definition of $\Vdash$ takes place. There are two, essentially equivalent, choices. We could work in the Kripke term model, where the terms are uninterpreted; the condition J would mean "the cut will happen somewhere in me, and at all future nodes the cut will be at a fixed real also somewhere in me". The choice we will pursue is to work in the classical models M_n.

So pick a formula with parameters from M_n, and work in M_n. (One could point out that, if the parameters are from M_n, then they are also from $M_{n'}$, for all $n' > n$, and then ask if there is any ambiguity where the following definition takes place. If one did, the answer would be "no, by elementarity.")

Definition 16. *For $\phi = \phi(\sigma_0, ..., \sigma_i)$ a formula with parameters $\sigma_0, ..., \sigma_i$, ϕ^r is $\phi(\sigma_0^r, ..., \sigma_i^r)$.*

Definition 17. *$J \Vdash \phi$ is defined inductively on ϕ:*

$J \Vdash \sigma = \tau$ and $J \Vdash \sigma \in \tau$ are as above
$J \Vdash \phi \wedge \psi$ iff $J \Vdash \phi$ and $J \Vdash \psi$
$J \Vdash \phi \vee \psi$ iff for all $r \in J$ then there is a J' $\subseteq J$ containing r such that J' $\Vdash \phi$ or J' $\Vdash \psi$
$J \Vdash \phi \rightarrow \psi$ iff for all J' $\subseteq J$ if J' $\Vdash \phi$ then J' $\Vdash \psi$, and, for all $r \in J$, if $\mathbf{R} \Vdash \phi^r$ then $\mathbf{R} \Vdash \psi^r$
$J \Vdash \exists x\ \phi(x)$ iff for all $r \in J$ there is a J' containing r and a σ such that $J \cap J' \Vdash \phi(\sigma)$
$J \Vdash \forall x\ \phi(x)$ iff for all $r \in J$ and σ there is a J' containing r such that $J \cap J' \Vdash \phi(\sigma)$, and for all $r \in J$ and σ there is a J' containing r such that J' $\Vdash \phi^r(\sigma)$.

(Notice that in the last clause, σ is not interpreted as σ^r.)

Lemma 8. *1. For all ϕ $\emptyset \Vdash \phi$.*
2. If J' $\subseteq J \Vdash \phi$ then J' $\Vdash \phi$.
3. If $J_i \Vdash \phi$ for all i then $\bigcup_i J_i \Vdash \phi$.
4. $J \Vdash \phi$ iff for all $r \in J$ there is a J' containing r such that $J \cap$ J' $\Vdash \phi$.
5. For all ϕ, J if $J \Vdash \phi$ then for all $r \in J$ $\mathbf{R} \Vdash \phi^r$.
6. Truth Lemma: For any node r, $r \models \phi$ iff $J \Vdash \phi$ for some J containing r.

Proof. Induction.

3.3 The Final Proof

Theorem 6. $\perp \models CZF_{Exp}$

Proof. Infinity, Pair, and Union are as in the previous section. Extensionality and Set Induction are too, with appropriate minor changes.

– Exponentiation: Let σ and τ be terms at node r. In what follows, we will identify sets (in an M_n) with their canonical terms. Let C be $\{\langle \rho, J\rangle \mid J \Vdash \rho : \sigma \rightarrow \tau \text{ is a function}\} \cup \{\langle h, s\rangle \mid h : \sigma^s \rightarrow \tau^s \text{ is a function}\}$. We claim that C suffices.

  Let s be any immediate extension of r. (The case of non-immediate extensions follows directly from this case.) If $s \models$ "$\rho : f(\sigma) \rightarrow f(\tau)$ is a function", then $s \models \rho \in C$ by the first clause in the definition of C. If $s \models$ "$\rho : g(\sigma) \rightarrow g(\tau)$ is a function" and ρ is canonical, then $s \models \rho \in C$ by the second clause. What we must show is that for any node r and sets X and Y, if $r \models$ "$\rho : X \rightarrow Y$ is a function", then ρ is canonical.

  By the Truth Lemma, let $r \in J \Vdash$ "$\rho : X \rightarrow Y$ is a function". We claim that for all $x \in X$ there is a $y \in Y$ such that for each $s \in J$ $s \models \rho(x) = y$. If not, let x be otherwise. Let y be such that $r \models \rho(x) = y$. For each immediate successor s of r, $s \models f(\rho)(f(x)) = f(y)$. By overspill the same holds for some neighborhood around r (sans the f's). If this does not hold for all $s \in J$, let s be an endpoint in J of the largest interval around r for which this does hold. Repeating the same argument around s, there is a y' such that, for all t in some neighborhood of s, $t \models \rho(x) = y'$. This neighborhood of s must overlap that of r, though. So $y = y'$, contradicting the choice of s. So the value $\rho(x)$ is fixed on the whole interval J, and ρ is forced by J to equal a particular canonical function.

– Separation: Although CZF contains only Δ_0 Separation, full Separation holds here. Let $\phi(x)$ be a formula and σ a term. Then, again identifying sets (in M_n) with their canonical terms, $\{\langle \sigma_i, J \cap J_i \rangle \mid \langle \sigma_i, J_i \rangle \in \sigma$ and $J \Vdash \phi(\sigma_i)\} \cup \{\langle x, s \rangle \mid x \in \sigma^s$ and $\mathbb{R} \Vdash \phi^s(x)\}$ will do.
– Strong Collection: If $r \models \forall x \in \sigma\ \exists y\ \phi(x, y)$, let $r \in J$ force as much. For each $\langle \sigma_i, J_i \rangle \in \sigma$ and $s \in J \cap J_i$, let $\tau_{i,s}$ and $J_{i,s}$ be such that $s \in J_{i,s} \Vdash \phi(\sigma_i, \tau_{i,s})$. Also, for each $s \in J$ and $x \in \sigma^s$, let $\tau_{x,s}$ be such that $\mathbb{R} \Vdash \phi^s(x, \tau_{x,s})$. Then $\{\langle \tau_{i,s}, J_{i,s} \rangle \mid i \in I, s \in J\} \cup \{\langle \tau_{x,s}, s \rangle \mid s \in J, x \in \sigma^s\}$ suffices.

Theorem 7. $\perp \models$ *The Dedekind reals do not form a set.*

Proof. Let σ be a term. At some future node s, $g(\sigma)$ is a canonical term, meaning it names a set from the ambient model. But the prototypical set from the first proof, $\{\langle r, J \rangle \mid r$ is a rational, J is an open interval from the reals, and $r < J\}$, at node s is a Dedekind real which is not from the ground model, and so is not in $g(\sigma)$. So σ cannot name the set of Dedekind reals.

References

1. P. Aczel, The type theoretic interpretation of constructive set theory, in A. MacIntypre, L. Pacholski, and J. Paris (eds.), Logic Colloquium '77 (North-Holland, Amsterdam, 1978), p. 55-66
2. P. Aczel, The type theoretic interpretation of constructive set theory: choice principles, in A.S. Troelstra and D. van Dalen (eds.), The L.E.J. Brouwer Centenary Symposium (North-Holland, Amsterdam, 1982), p. 1-40
3. P. Aczel, The type theoretic interpretation of constructive set theory: inductive definitions, in R.B. Marcus et al. (eds.), Logic, Methodology and Philosophy of Science VII (North-Holland, Amsterdam, 1986), p. 17-49
4. P. Aczel and M. Rathjen, Notes on constructive set theory, Technical Report 40, 2000/2001, Mittag-Leffler Institute, Sweden, 2001.
5. L. Crosilla, H. Ishihara, and P. Schuster, On constructing completions, *The Journam of Symbolic Logic*, 2005, v. 70, p. 969-978
6. M.P. Fourman and J.M.E. Hyland, Sheaf models for analysis, in: M.P. Fourman, C.J. Mulvey and D.S. Scott (eds.), Applications of sheaves (Springer, Berlin, 1979), p. 280-301.
7. H. Friedman and A. Scedrov, The lack of definable witnesses and provably recursive functions in intuitionistic set theories, *Advances in Math*, 1985, p. 1-13
8. R. Lubarsky, Independence results around Constructive ZF, *Annals of Pure and Applied Logic*, 2005, v. 132, p. 209-225
9. R. Lubarsky, CZF + full Separation is equivalent to second order arithmetic, *Annals of Pure and Applied Logic*, 2006, v. 141, pp. 29-34
10. P. Martin-Löf, An intuitionistic theory of types: predicative part, in H.E. Rose and J. Sheperdson (eds.), Logic Colloquium '73 (North-Holland, Amsterdam, 1975), p. 73-118

11. P. Martin-Löf, Intuitionistic Type Theory (Bibliopolis, Naples, 1984)
12. J. Myhill, Constructive set theory, *Journal of Symbolic Logic*, 1975, p. 347-382
13. M. Rathjen, The strength of some Martin–Löf type theories, *Archive for Mathematical Logic*, 1994, p. 347-385
14. M. Rathjen, The higher infinite in proof theory, in J.A. Makowsky and E.V. Ravve (eds.), Logic Colloquium '95, Springer Lecture Notes in Logic, Vol. 11 (Springer, New York, Berlin, 1998), p. 275-304
15. A. Simpson, Constructive set theories and their category-theoretic models, in L.Crosilla and P.Schuster, eds., From Sets and Types to Topology and Analysis (Oxford Logic Guides, Oxford University Press, forthcoming)
16. T. Streicher, Realizability Models for IZF and CZF + $\neg$ Pow via the Aczel Construction, personal communication
17. A.S. Troelstra and D. van Dalen, Constructivism in Mathematics, Volumes I, II (North Holland, Amsterdam, 1988).

Verifying Balanced Trees

Zohar Manna[1], Henny B. Sipma[1], and Ting Zhang[2]

[1] Stanford University
{zm,sipma}@cs.stanford.edu
[2] Microsoft Research Asia
tingz@microsoft.com

Abstract. Balanced search trees provide guaranteed worst-case time performance and hence they form a very important class of data structures. However, the self-balancing ability comes at a price; balanced trees are more complex than their unbalanced counterparts both in terms of data structure themselves and related manipulation operations. In this paper we present a framework to model balanced trees in decidable first-order theories of term algebras with Presburger arithmetic. In this framework, a theory of term algebras (i.e., a theory of finite trees) is extended with Presburger arithmetic and with certain connecting functions that map terms (trees) to integers. Our framework is flexible in the sense that we can obtain a variety of decidable theories by tuning the connecting functions. By adding *maximal path* and *minimal path* functions, we obtain a theory of red-black trees in which the transition relation of tree self-balancing (rotation) operations is expressible. We then show how to reduce the verification problem of the red-black tree algorithm to constraint satisfiability problems in the extended theory.

1 Introduction

Balanced search trees provide guaranteed worst-case time performance and hence they form a very important class of data structures. Also they are the basis of efficient implementations of many advanced data structures such as associative arrays and associative sets. However, the self-balancing ability comes at a cost; balanced trees are more complex than their unbalanced counterparts both in terms of data structure themselves and related manipulation operations. Moreover, as balanced trees are not regular trees, their properties cannot be directly characterized by standard tree automata techniques [4].

In this paper we present a framework to model balanced trees in decidable first-order theories of term algebras with Presburger arithmetic [23]. In this framework, a theory of term algebras (i.e., a theory of finite trees) is extended with Presburger arithmetic and certain connecting functions that map terms (trees) to integers. Given connecting functions and a fixed signature of a term

[1] The first and the second author were supported in part by NSF grants CCR-01-21403, CCR-02-20134, CCR-02-09237, CNS-0411363, and CCF-0430102, by ARO grant DAAD19-01-1-0723, and by NAVY/ONR contract N00014-03-1-0939.

algebra, the corresponding extended theory is two sorted with *integer sort* and *term sort*. The language is the set-theoretic union of the language of term algebras and the language of Presburger arithmetic augmented with connecting functions from $\mathbb{T}$ to $\mathbb{N}$. Formulae are formed from term literals and *integer literals* using logical connectives and quantifications.

Our framework is flexible in the sense that we can obtain a variety of decidable theories by varying the connecting functions. By adding *maximal path* and *minimal path* functions, we obtain a theory of red-black trees in which the transition relation of tree self-balancing (color exchange and rotation) operations are expressible. We then show how to reduce the verification problem of the red-black tree algorithm to a constraint satisfiability problem in the extended theory.

Related Work and Comparison. There has been a considerably amount of work in *shape analysis*, a kind of *pointer analysis* aimed at statically inferring properties on heap allocated data structures. Shape analysis tools can partially detect pointer linkage properties such as sharing, aliasing, cyclicity and reachability. The property of being a balanced tree, however, is a much higher-level property that can not be inferred from pointed-to relations on heaps. Rugina presents a method, called quantitative shape analysis, to verify rebalancing operations on AVL trees [19]. Based on abstract interpretation, the method performs forward propagation in an abstract heap where each location (node) is associated with quantitative attributes and relations to characterize the balancing property.

Tree automata techniques are widely used in solving constraints on tree languages [4]. However, balanced trees are not regular trees (by the pumping lemma), and hence their corresponding tree languages cannot be directly characterized by standard tree automata [4]. Habermehl et al. present extended tree automata with size constraints on transition relations (TASC) [10]. TASCs are able to represent pre- and post-conditions of a program involving tree rotation operations. Hence, given the right invariants, the verification of a program reduces to checking the validity of Hoare triples that state that after execution of the program from the starting state satisfying the pre-condition, the resulting reachable states are included in the states represented by the post-condition. However, TASCs that encode transition relations of tree operations are fairly complicated. The lack of intuitive connections between low level program statements and the corresponding automata representations makes this formalism unattractive for use in practical verification tools.

Baldan et al. treat red-black trees as hypergraphs and tree update operations as rewritings on hypergraphs [2]. They use approximate unfolding to compute the reachable states of a graph rewriting system to prove the property that no red node has a red parent. The balancing property itself, however, is not expressible in graph rewriting grammar and an additional type system is introduced to prove it. Calcagno et al. present a context logic to model trees with *local updates*, which are destructive operations at pointed locations [3]. They present a deductive proof system based on Hoare triples and prove soundness and completeness of the proof system. The balancing property, however, is not expressible this formal system and the verification of Hoare triples is not fully automatic.

Like [10] we reduce the verification problem to checking the validity of Hoare triples. In our approach, however, the pre- and post-conditions and transition relations are all represented directly by the first-order formulas in the extended theory of term algebras. Different from [10,2,3], we do not have an update function in the theory and hence we can not express local updates at an arbitrary pointed location. On the other hand, local updates only affect subtrees around the focus point, and hence our theory can still express and prove verification conditions of step-by-step tree operations. An informal but easy induction will give us global safety properties.

The contributions of this paper are the following: (1) we develop a first-order theory of red-black trees, that is, a theory of term algebras augmented with Presburger arithmetic; (2) we show how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples; (3) we provide a decision procedure for automatically checking validity of the resulting verification conditions. To the best of our knowledge, this is the first decidable logic theory for red-black trees. Moreover, it can be easily generalized to model other balanced tree structures, such as AVL trees and B-trees.

Paper Organization Section 2 presents the notation and terminology for term algebras. Section 3 introduces the theory of red-black trees and states its decidability result. Section 4 shows how to use the theory to analyze the red-black tree insertion algorithm. Section 5 concludes with a discussion of future work. Because of space limitations all decidability proofs have been omitted. They are available for reference on the third author's web site.

2 Preliminaries

We assume the first-order syntactic notions of variables, parameters and quantifiers, and semantic notions of structures, satisfiability and validity as in [9]. We use $[\![x]\!]$ to denote the value given by an assignment and $\bar{x}$ to denote a sequence of variables.

Definition 1 (Term Algebras). *A term algebra* TA : $\langle \mathbb{T}; \mathcal{C}, \mathcal{A}, \mathcal{S}, \mathcal{T} \rangle$ *consists of*

1. $\mathbb{T}$*: The term domain, which exclusively consists of terms recursively built up from constants by applying non-nullary constructors. Objects in* $\mathbb{T}$ *are called* TA-terms. *The type of a term t, denoted by* $\text{type}(t)$*, is the outermost constructor symbol of t. We say that t is α-typed (or is an α-term) if* $\text{type}(t) = \alpha$.
2. $\mathcal{C}$*: A set of constructors: α, β, γ, ... The arity of α is denoted by* $\text{ar}(\alpha)$.
3. $\mathcal{A}$*: A set of constants: a, b, c, ... We require $\mathcal{A} \neq \emptyset$ and $\mathcal{A} \subseteq \mathcal{C}$. For $a \in \mathcal{A}$,* $\text{ar}(a) = 0$ *and* $\text{type}(a) = a$.

4. *$\mathcal{S}$: A set of selectors. For a constructor α with arity $k > 0$, there are k selectors $\mathsf{s}_1^\alpha, \ldots, \mathsf{s}_k^\alpha$ in $\mathcal{S}$. We call s_i^α ($1 \leq i \leq k$) the i^{th} α-selector. For a term x, $\mathsf{s}_i^\alpha(x)$ returns the i^{th} immediate subterm of x if x is an α-term and x itself otherwise.*
5. *$\mathcal{T}$: A set of testers. For each constructor α there is a corresponding tester Is_α. For a term x, $\text{Is}_\alpha(x)$ is true if and only if x is an α-term. For a constant a, $\text{Is}_a(x)$ is just $x = a$. In addition there is a special tester Is_{A} such that $\text{Is}_{\mathsf{A}}(x)$ is true if and only if x is a constant.*

We use $\mathscr{L}_{\mathbb{T}}$ to denote the language of term algebras.

Term domain $\mathbb{T}$ consists of only ground terms built from constructors. Selectors only exist in the *formal* language. In fact selectors can be defined by constructors in the existential fragment of the language; for a quantifier-free formula $\Phi(\bar{x})$ containing selectors, we can obtain an equivalent and selector-free formula $\exists \bar{y}\, \Phi'(\bar{x}, \bar{y})$ where $\Phi'(\bar{x}, \bar{y})$ is quantifier-free and $\bar{y}$ is fresh.

A term t is called a *constructor term* if t is a variable or the outermost function symbol of t is a constructor. Constants are constructor terms. A term t is called a *selector term* if either t is a variable or the outermost function symbol of t is a selector. Variables are both constructor terms and selector terms. We assume that no constructors appear immediately inside selectors as simplification can always be done. A term is called *proper* if it is not a constant or a variable.

The first-order theory of term algebras was shown to be decidable by Mal'cev using quantifier elimination [15]. Decision procedures for the quantifier-free theory were discovered by Nelson, Oppen et al. [16,17,8]. Oppen gave a linear algorithm for acyclic structures [17] and (with Nelson) a quadratic algorithm for cyclic structures [16]. If the values of the selector functions on constants are specified, then the problem is NP-complete [17].

Presburger arithmetic is the first-order theory of addition in the arithmetic of integers. The corresponding structure is denoted by $\text{PA} = \langle \mathbb{Z}; 0, +, < \rangle$. We use $\mathscr{L}_{\mathbb{Z}}$ to denote the formal language of PA. The first-order theory of Presburger arithmetic (PA) was first shown to be decidable in 1929 by the quantifier elimination method [9]. More efficient algorithms were later discovered by [6] and further improved in [18].

There has been a great interest in generalizing Mal'cev's result on term algebras. Maher showed the decidability of the theory of infinite and rational trees [14]. Comon and Delor presented an elimination procedure for term algebras with membership predicate in the regular tree language [5]. Backofen presented an elimination procedure for structures of feature trees with arity constraints [1]. Rybina and Voronkov showed the decidability of term algebras with queues [20]. Kuncak and Rinard showed the decidability of term powers, which are term algebras augmented with coordinate-wise defined predicates [13]. A combination of Presburger arithmetic and term algebras was used by Korovin and Voronkov to show that the quantifier-free theory of term algebras with Knuth-Bendix order is NP-complete [11,12]. In [21,23] we presented decision procedures both for the first-order theory and the corresponding quantifier-free fragments of term algebras with integer functions. In [22] we extended the decidability result to the first-order theory of term algebras with Knuth-Bendix order.

3 The Theory of Red-Black Trees

In this section we present a theory of a term algebra with two integer functions to express the properties of red-black trees.

Definition 2 (Red-black Trees [7]). *A* red-black *tree is a binary tree with the following coloring properties:*

1. *Every node is either red or black.*
2. *Every leaf node is black.*
3. *The root is black.*
4. *Every red node has two black children.*
5. *All paths from the root to leaf nodes contain the same number of black nodes.*

Properties (1)-(3) can be modeled in a theory of term algebras as follows:

Definition 3 (Structure of Colored Trees). *The structure of red-black colored trees is*

$$\begin{aligned}\mathrm{RB} = \langle\ \mathbb{T}_{\mathrm{rb}};\ &\{\mathrm{red}, \mathrm{black}, \mathrm{nil}\},\ \{\mathrm{nil}\},\\ &\{\mathrm{car}^{\mathrm{red}}, \mathrm{cdr}^{\mathrm{red}}, \mathrm{car}^{\mathrm{black}}, \mathrm{cdr}^{\mathrm{black}}\},\ \{\mathrm{Is}_{\mathrm{red}}, \mathrm{Is}_{\mathrm{black}}, \mathrm{Is}_{\mathrm{nil}}\}\ \rangle,\end{aligned}$$

where $\mathbb{T}_{\mathrm{rb}}$ *denotes the domain,* nil *denotes a leaf,* red *and* black *are binary constructors,* $\mathrm{car}^{\sharp}$ *and* $\mathrm{cdr}^{\sharp}$*, respectively, are the left and the right* $\sharp$*-selectors (*$\sharp \in \{\mathrm{red}, \mathrm{black}\}$*). The corresponding language is denoted by* $\mathscr{L}_{\mathrm{RB}}$*.*

For notation simplicity we use car to denote either $\mathrm{car}^{\mathrm{red}}$ or $\mathrm{car}^{\mathrm{black}}$, which should be clear from the context. Similar for the use of cdr. If a term t appears in a selector, we assume either $\mathrm{Is}_{\mathrm{red}}(t)$ or $\mathrm{Is}_{\mathrm{black}}(t)$ holds. For example, $\mathrm{car}(x) = y$ should be understood as an abbreviation of

$$(\mathrm{Is}_{\mathrm{red}}(x)\ \wedge\ y = \mathrm{car}^{\mathrm{red}}(x))\ \vee\ (\mathrm{Is}_{\mathrm{black}}(x)\ \wedge\ y = \mathrm{car}^{\mathrm{black}}(x)).$$

In the following we use *terms* and *trees*,respectively, to refer to syntactic objects and semantic objects. We call terms (trees) of red-type (resp. of black-type) red-*terms (-trees)* (resp. black-*terms (-trees)*).

We extend RB with PA to express balancing properties (4)-(5):

Definition 4 (Structure of Red-black Trees). *The structure of red-black trees is*

$$\mathrm{RB}_{\mathbb{Z}} = \langle\, \mathrm{RB};\ \mathrm{PA};\ |\cdot|_{\max}, |\cdot|_{\min} : \mathbb{T}_{\mathrm{rb}} \to \mathbb{N}\,\rangle,$$

where, $|\cdot|_{\max}$ *and* $|\cdot|_{\min}$ *are two integer functions defined recursively as*

$$|x|_\star = \begin{cases} 1 & x = \text{nil}, \\ 0 & \text{Vio}(x), \\ \star(|x_1|_\star, |x_2|_\star) + 1 & \text{GB}(x, x_1, x_2), \\ \star(|x_1|_\star, |x_2|_\star) & \text{GR}(x, x_1, x_2). \end{cases}$$

where $\star \in \{\max, \min\}$ *and* $\text{GB}(x, x_1, x_2)$*,* $\text{GR}(x, x_1, x_2)$ *and* $\text{Vio}(x)$ *are*

$$\begin{aligned} \text{Vio}(x) &\overset{\text{def}}{=\!=} x \neq \text{nil} \ \wedge \ \forall x_1 \forall x_2 \left(\neg\text{GB}(x, x_1, x_2) \ \vee \ \neg\text{GR}(x, x_1, x_2)\right), \\ \text{GB}(x, x_1, x_2) &\overset{\text{def}}{=\!=} x = \text{black}(x_1, x_2) \ \wedge \ |x_1|_{\max} \neq 0 \ \wedge \ |x_2|_{\max} \neq 0, \\ \text{GR}(x, x_1, x_2) &\overset{\text{def}}{=\!=} x = \text{red}(x_1, x_2) \ \wedge \ |x_1|_{\max} \neq 0 \ \wedge \ |x_2|_{\max} \neq 0 \\ &\qquad\qquad \wedge \ \neg\text{Is}_{\text{red}}(x_1) \ \wedge \ \neg\text{Is}_{\text{red}}(x_2). \end{aligned}$$

We denote the corresponding language by $\mathscr{L}_{\text{RB}}^{\mathbb{Z}}$.

$\text{Vio}(x)$ states that x violates property (4) of red-black trees. $\text{GB}(x, x_1, x_2)$ states x is a black tree with two good subtrees x_1 and x_2. Similarly for $\text{GR}(x, x_1, x_2)$. $|x|_{\max}$ (resp. $|x|_{\min}$) gives the maximal (resp. minimal) number of black nodes that x can have on a maximal path. A maximal path of x that contains the largest (resp. smallest) number of black nodes is called a *maximal black path* (resp. *minimal black path*) of x. We call $|x|_{\max}$ the *maximal black length* of x, $|x|_{\min}$ the *minimal black length*, and the pair $(|x|_{\max}, |x|_{\min})$ the *measure* of x, denoted by $\|x\|$.

In this theory, properties (1) and (2) of Definition 2, which state that every node is either black or red, and that a nil node is black, are trivially satisfied by the choice of signature and the integer functions. Therefore x is a red-black tree if x satisfies the following three conditions.

(§1)	$\lvert x\rvert_{\max} = \lvert x\rvert_{\min}$	*any maximal path of x contains the same number of black nodes,*
(§2)	$\lvert x\rvert_{\max} > 0$	*any red node of x must have two black children,*
(§3)	$\text{Is}_{\text{black}}(x)$	*the root of x is black.*

We denote by $\varphi_{\text{RB}}^-(x)$ the conjunction of (§1) and (§2), and by $\varphi_{\text{RB}}(x)$ the conjunction of (§1)-(§3). We note that $\varphi_{\text{RB}}(x)$ defines a subdomain of $\mathbb{T}_{rb}$ and the theory of this subdomain can be obtained by relativizing quantifiers to $\varphi_{\text{RB}}(x)$. Formally, $\forall x(\varphi_{\text{RB}}(x) \to \Phi(x))$ (resp. $\exists x(\varphi_{\text{RB}}(x) \wedge \Phi(x))$) expresses that $\Phi(x)$ is a universal (resp. existential) property of red-black trees. We have

Theorem 1 (Decidability of $\text{RB}_{\mathbb{Z}}$**)**

1. *The first-order theory of* $\text{RB}_{\mathbb{Z}}$ *is decidable and admits quantifier elimination.*
2. *The decision problem for the quantifier-free fragment is NP-complete.*

4 Analysis of Red-Black Trees

4.1 Algorithm and Example

In this section we consider the insertion-fixup operation of red-black trees represented by Algorithm 2, a slightly modified version of the algorithm given in [7]. We illustrate the algorithm on the same example as in [7], inserting 4 at the bottom of the tree and showing how the algorithm restores the red-black tree property.

Algorithm 2 (RB-Insertion-Fixup)
Input: root, T, x.

```
1:  while ($x \neq root$ and $T[x\text{-}1].color$ = RED) do
2:      if $T[x\text{-}1].dir = right$ then
3:          if ($T[x\text{-}1].tree$ IS RED) {Case 1} then
4:              $T[x\text{-}1].tree$ := BLACK(car($T[x\text{-}1].tree$),cdr($T[x\text{-}1].tree$))
5:              $T[x\text{-}1].color$ := BLACK
6:              $T[x\text{-}2].color$ := RED
7:              $x := x\text{-}2$
8:          else
9:              if ($T[x].dir = left$) {Case 2} then
10:                 swap($T[x].tree$, $T[x+1].tree$)
11:                 $T[x].dir := right$
12:                 $T[x+1].dir := left$
13:             end if
14:             $T[x\text{-}1].color$ := BLACK {Case 3}
15:             $T[x].tree$ := RED($T[x].tree$, $T[x\text{-}1].tree$)
16:             $T[x\text{-}1].tree := T[x\text{-}2].tree$
17:             $T[x\text{-}1].dir := T[x\text{-}2].dir$
18:             if ($x\text{-}2 \neq root$) then
19:                 $x\text{-}3+1 := x\text{-}1$
20:             else
21:                 $root := x\text{-}1$
22:             end if
23:         end if
24:     else if ($T[x\text{-}1].dir = left$) then
25:         similar code as the then clause with left and right swapped
26:     end if
27: end while
28: $T[root].color$ := BLACK
```

Recall that our language does not have an update function to express the relation between the original tree and updated tree if the update happens at an unbounded depth inside the tree. We know that the restoring updates will begin at the newly inserted node and traverse upwards to the root and that all local updates will happen on this path. We represent the tree as a sequence of subtrees indexed by nodes on the path from the root to the newly inserted

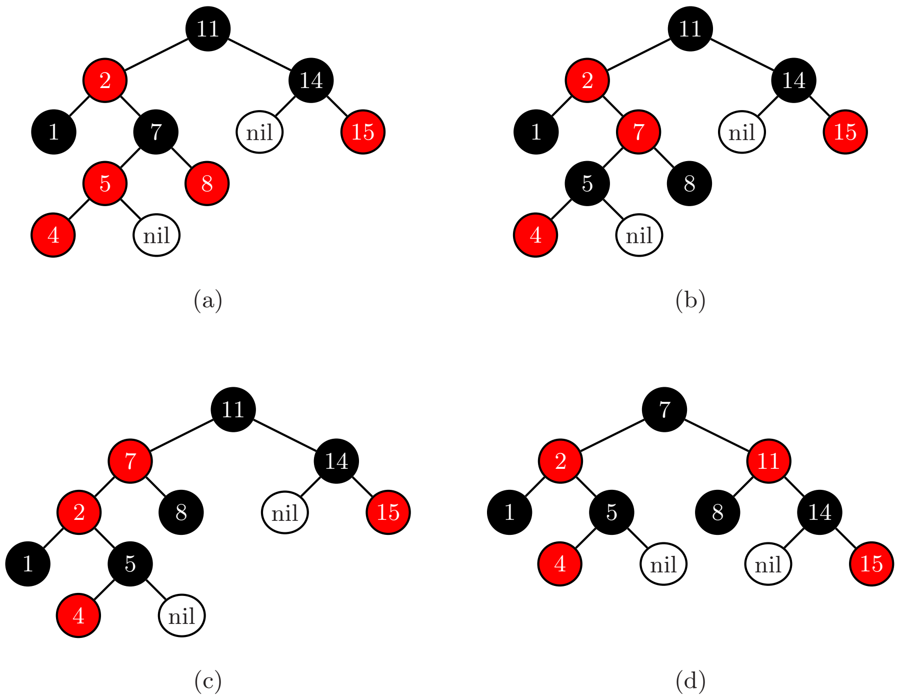


Fig. 1. A run of RB-Insertion-Fixup

node. We treat the path as a doubly linked list (denoted by T in the algorithm) in which each element contains three fields, *.color*, *.dir* and *.tree*. Field *.color* denotes the type of the node. Field *.dir* indicates whether the subtree at this node is the left child or the right child. Field *.tree* denotes the sibling subtree of this node. We have $root.dir = \bot$ and $root.tree = \bot$. For simplicity, we omit the value field as it has no role in restoring the red-black tree property. We treat *root* and x as iterators and we use array notation $T[x]$ to denote the element pointed to by x. We use $x + 1$ and $x - 1$ to denote previous iterator and next iterator of x, respectively. For example, the statement $x + 3 - 1 = x - 1$ at line 19 means $x.pre.pre.pre.next := x.pre$.

Figure 1 shows the results of the operations performed to restore the balanced-tree property after inserting 4. Figure 2 gives a more detailed picture of the data structures of the nodes on the path from the root to x. Figure 1 (b) shows the tree obtained by recoloring. The new violation now corresponds to Case 2 in Algorithm 2. Figure 1 (c) shows the tree obtained from a left rotation. There is still a violation which corresponds to Case 3 in Algorithm 2. Figure 1 (d) shows a new red-black tree after a right rotation. Figures 2 (b)-(d) show the corresponding changes of the data structure during the run[1].

[1] To save space, in all figures we do not draw a nil node if its sibling is not nil, and for this reason, we do not draw nil nodes black.

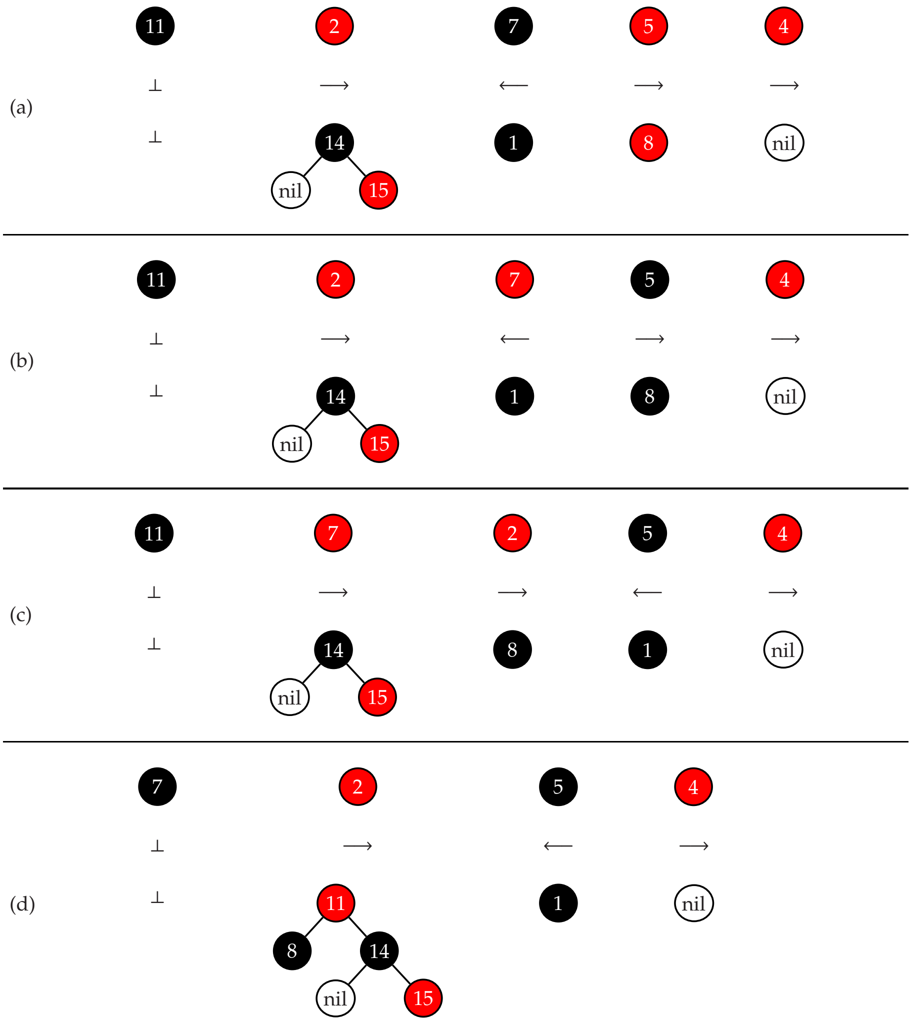


Fig. 2. Paths from the root of the tree to x. In each of (a)-(d), the first row shows the sequence of nodes from the root to x; the second row shows whether the node above it is a left ($\longleftarrow$) or right ($\longrightarrow$) sibling; the third row shows the sibling tree of the node in the top row.

4.2 Verification Conditions

We now show how to use $\mathscr{L}_{\mathrm{RB}}^{\mathbb{Z}}$ to express the verification conditions for statements restoring that red-black tree property in Algorithm 2. Recall that in the algorithm x is an iterator and $T[x]$ is a node pointed by x in a linked list and it contains three fields, *.dir*, *.color* and *.tree*. At the semantic level, however, we view x as an integer index and $T[x]$ as a subtree indexed by x. If $x \neq \mathit{root}$, then $T[x].\mathit{tree}$ denotes the sibling tree of $T[x]$, and $T[x-1]$ represents the immediate super-tree containing $T[x]$. For example, if $T[x].\mathit{dir}$ is right and $T[x-1].\mathit{color}$ is red, then $T[x-1] = \mathrm{red}(T[x], T[x].\mathit{tree})$. We have three field operators, *.dir*, *.color* and *.tree*. Among them *.dir* can only take three values, left, right and $\bot$, so expressions involving *.dir* can be removed by disjunctive splitting. Similar for *.color*, but it can be directly expressed in $\mathscr{L}_{\mathrm{RB}}$ as below.

$$\begin{aligned} T[x].\mathit{color} = \mathrm{red} &\overset{\mathsf{def}}{==} \mathrm{Is}_{\mathrm{red}}(T[x]), \\ T[x].\mathit{color} = \mathrm{black} &\overset{\mathsf{def}}{==} x = \mathrm{nil} \ \vee \ \mathrm{Is}_{\mathrm{black}}(T[x]). \end{aligned}$$

With the help of *.dir*, *.tree* can be expressed in $\mathscr{L}_{\mathrm{RB}}$ as follows.

$$\begin{aligned} T[x].\mathit{tree} = y \neq \bot \quad &\overset{\mathsf{def}}{==} \quad x \neq \mathit{root} \ \wedge \ \Big((y = \mathrm{car}(T[x-1]) \ \wedge \ T[x].\mathit{dir} = \mathit{right}) \\ &\qquad\qquad \vee \ (y = \mathrm{cdr}(T[x-1]) \ \wedge \ T[x].\mathit{dir} = \mathit{left}) \Big) \ , \\ T[x].\mathit{tree} = \bot \quad &\overset{\mathsf{def}}{==} \quad x = \mathit{root} \ . \end{aligned}$$

Therefore from now on we treat field access expressions as abbreviations in $\mathscr{L}_{\mathrm{RB}}$. Note that we use array and record notations for clarity. At the formula level terms of index access or field access are simply variables. For example, $T[x]$, $T[x].\mathit{tree}$, $T[x].\mathit{color}$ and $T[x].\mathit{dir}$ can be represented by variables f_x, g_x, h_x and k_x indexed by x, respectively. Similarly for terms indexed by $x-i$ and $x+i$.

Let $\bar{v}$ denote the variables in the current state and $\bar{v}'$ denote the corresponding variables in the next state. The transition relation of a statement q is denoted by $\rho_q(\bar{v}, \bar{v}')$. The post-condition $\mathit{post}(q, \varphi)$ of $\varphi(\bar{v})$ after executing a statement q is

$$(\exists \bar{v}^0)\ \big(\ \rho_q(\bar{v}^0, \bar{v}) \ \wedge \ \varphi(\bar{v}^0)\ \big).$$

The transition relation of two sequential statements can be computed as follows. Let $\rho_q(\bar{v}, \bar{v}^1)$ and $\rho_r(\bar{v}^1, \bar{v}')$ be the transition relations for statements q and r respectively. Then the transition relation of the composite statement $\langle q; r\rangle$ is

$$(\exists \bar{v}^1)\ \big(\ \rho_q(\bar{v}, \bar{v}^1) \ \wedge \ \rho_r(\bar{v}^1, \bar{v}')\ \big).$$

The validity checking of a Hoare triples $\{\varphi\}q\{\psi\}$ is equivalent to proving that $\mathit{post}(q, \varphi) \rightarrow \psi$.

The lack of update functions makes it impossible to express the tree operational semantics precisely in a finite formula. For example, when $T[x]$ is changed, not only should $T'[x]$ appear in $\rho_q(\bar{v}, \bar{v}')$, but also all ancestors of $T[x]$. In fact $\rho_q(\bar{v}, \bar{v}')$ has an unbounded number of conjuncts of the form $\mathrm{car}(T'[x-i]) = T'[x-i+1]$ or $\mathrm{cdr}(T'[x-i]) = T'[x-i+1]$. We can still, however, prove *safety properties* about tree operations with the help of an informal induction. As an example, we show that $\varphi^-_{\mathrm{RB}}(T[x])$, introduced in Section 3, is an invariant with respect to each code fragment (corresponding to Case 1, 2 or 3 in Algorithm 2). This can be obtained by establishing the Hoare triple $\{\varphi\}Q\{\psi\}$ where φ is the pre-condition

$$\begin{aligned} &x \neq \mathit{root} \ \wedge \ x-1 \neq \mathit{root} \rightarrow \\ &\quad \big(\varphi^-_{\mathrm{RB}}(T[x]) \ \wedge \ \varphi^-_{\mathrm{RB}}(T[x].\mathit{tree}) \ \wedge \ \neg\varphi^-_{\mathrm{RB}}(T[x-1]) \ \wedge \ \varphi^-_{\mathrm{RB}}(T[x-1].\mathit{tree})\big) \end{aligned}$$

ψ is the post-condition $\varphi_{\mathrm{RB}}^{-}(T[x])$, and Q is a code fragment corresponding to Case 1, 2 or 3. Here we need another invariant $\forall x(x \neq \mathit{root} \rightarrow \varphi_{\mathrm{RB}}^{-}(T[x].\mathit{tree}))$. This invariant can not be formally proved in our theory because of the universal quantification on indexes. But it is easy to verify that the parametric Hoare triples

$$\{x \neq \mathit{root} \rightarrow \varphi_{\mathrm{RB}}^{-}(T[x \pm i].\mathit{tree})\} \quad q \quad \{x \neq \mathit{root} \rightarrow \varphi_{\mathrm{RB}}^{-}(T[x \pm i].\mathit{tree})\}$$

can be established for each statement q not modifying index x (see below for the transition relations of those statements). In the following we list *local* transition relations of all statements involving tree update and use guard conditions to simplify those transition relations.

Case 1 is implemented by statements 4-7. The guard conditions are $x \neq \mathit{root} \wedge \mathrm{Is}_{\mathrm{red}}(T[x-1])$ (line 1), $T[x-1].\mathit{dir} = \mathit{right}$ (line 2) and $\mathrm{Is}_{\mathrm{red}}(T[x-1].\mathit{tree})$ (line 3). Under these conditions the transition relations for statements 4-7 are, respectively,

$$\begin{aligned} T'[x-1].\mathit{tree} &= \mathrm{cdr}(T'[x-2]) \\ &= \mathrm{black}(\mathrm{car}(T[x-1].\mathit{tree}), \mathrm{cdr}(T[x-1].\mathit{tree})), \end{aligned} \tag{S-4}$$

$$\mathrm{car}(T'[x-2]) = T'[x-1] = \mathrm{black}(\mathrm{car}(T[x-1]), \mathrm{cdr}(T[x-1])), \tag{S-5}$$

$$T'[x-2] = \mathrm{red}(\mathrm{car}(T[x-2]), \mathrm{cdr}(T[x-2])), \tag{S-6}$$

$$x' = x - 2. \tag{S-7}$$

The composite transition relation for statements 4-7 is

$$\begin{aligned} & T'[x-1].\mathit{tree} = \mathrm{black}(\mathrm{car}(T[x-1].\mathit{tree}), \mathrm{cdr}(T[x-1].\mathit{tree})) \\ \wedge\quad & T'[x-1] = \mathrm{black}(\mathrm{car}(T[x-1]), \mathrm{cdr}(T[x-1])) \\ \wedge\quad & T'[x-2] = \mathrm{red}(T'[x-1], T'[x-1].\mathit{tree}) \\ \wedge\quad & x' = x - 2. \end{aligned}$$

Recall that $T[x]$, $T[x].\mathit{tree}$, $T[x].\mathit{color}$ and $T[x].\mathit{dir}$ are just more informative aliases of indexed variables f_x, g_x, h_x and k_x, respectively. Similarly for terms indexed by $x-1$ and $x-2$. The next state variable for $T[x]$ should be $T'[x']$, but by default we write $T'[x]$ when $x' = x$. When we do transition relation composition, statement 7 requires us to hard code the integer indexing properties in the formula by adding equalities like $T'[x-1] = T'[x'+1]$, $T'[x-2].\mathit{tree} = T'[x'].\mathit{tree}$ and so on. To save space, however, we omit them in the above example.

Figure 3 illustrates Case 1 by a run on tree (a) in Figure 1 (copied as (b-0)). Trees (b-1), (b-2) and (b-3) are the outcomes of statements 4, 5 and 6, respectively.

The code fragment for Case 2 consists of statements 10-12. We take into account the conditions $T[x-1].\mathit{color} = \mathrm{red}$ (line 1), $T[x-1].\mathit{dir} = \mathit{right}$ (line 2)

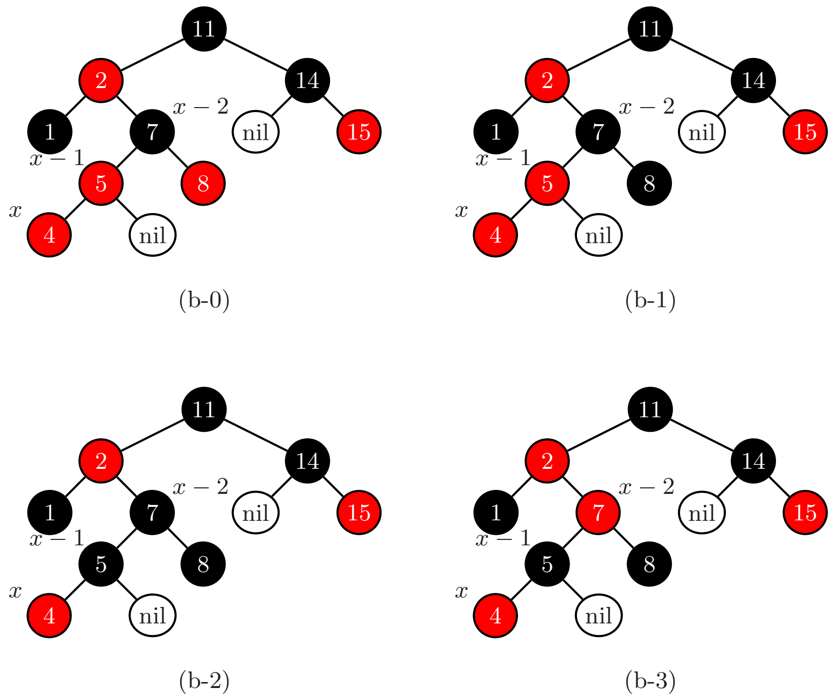


Fig. 3. A detailed run of RB-INSERTION-FIXUP step (b)

and $T[x].dir = left$ (line 9). Under these condition the transition relations for statements 10-12 are, respectively,

$$\begin{aligned} &\mathrm{cdr}(T'[x-1]) = T'[x] \ \wedge\ (T'[x+1].tree = \mathrm{cdr}(T'[x]) = T[x].tree) \\ &\qquad\qquad \wedge\ (T'[x].tree = \mathrm{car}(T'[x-1]) = T[x+1].tree), \quad \text{(S-10)} \\ &T'[x].dir = right \ \wedge\ T'[x-1] = \mathrm{red}(\mathrm{cdr}(T[x-1]), \mathrm{car}(T[x-1])), \quad \text{(S-11)} \\ &T'[x+1].dir = left \ \wedge\ \mathrm{car}(T'[x-1]) = T'[x] \\ &\qquad\qquad \wedge\ T'[x] = \mathrm{red}(\mathrm{cdr}(T[x]), \mathrm{car}(T[x])). \quad \text{(S-12)} \end{aligned}$$

The composite transition relation for statements 10-12 is

$$\begin{aligned} & T'[x+1].tree = T[x].tree \ \wedge\ T'[x].tree = T[x+1].tree \\ \wedge\ & T'[x].dir = right \ \wedge\ T'[x-1] = \mathrm{red}(T'[x], T[x+1].tree) \\ \wedge\ & T'[x+1].dir = left \ \wedge\ T'[x] = \mathrm{red}(T[x].tree, \mathrm{car}(T[x])). \end{aligned}$$

Figure 4 illustrates Case 2 by a run on tree (b) in Figure 1 (copied as (c-0)). Trees (c-1), (c-2) and (c-3) are the outcomes of statements 10, 11 and 12, respectively. Recall that we ignored value labels at internal nodes as they are irrelevant to the red-black tree properties. But the binary search tree property may be violated without adjusting the value labels. So for the sake of illustration we switched positions of 2 and 7 in Figure 4 (c-1) although statement 10 does not have this effect.

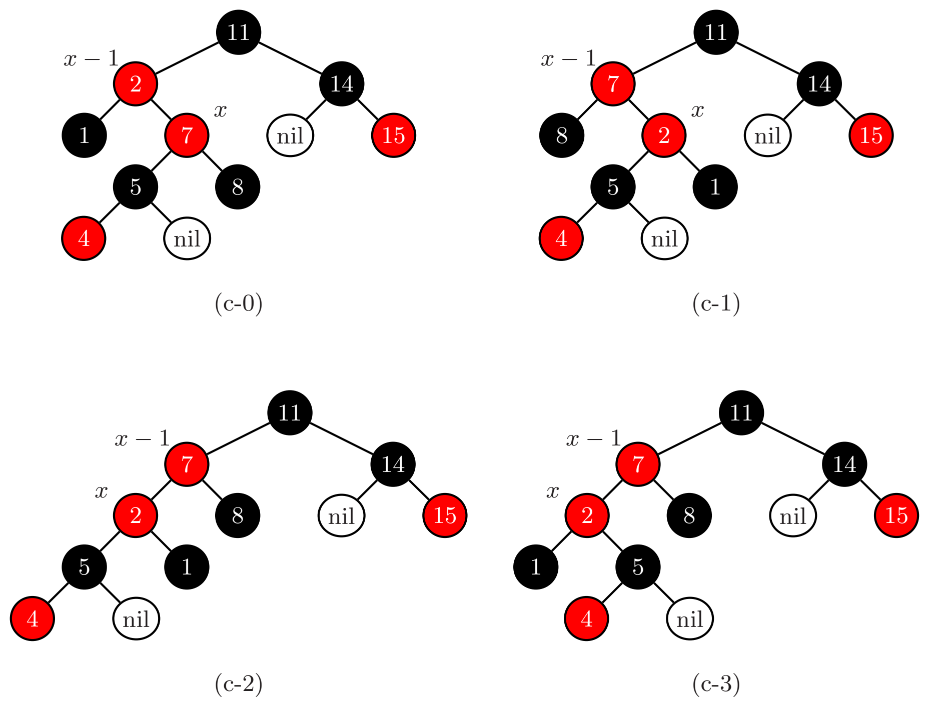


Fig. 4. A detailed run of RB-Insertion-Fixup step (c)

Case 3 consists of statements 14-21. We take into account the conditions $T[x-1].color = \text{red}$ (line 1), $T[x-1].dir = right$ (line 2) and $T[x].dir = right$ (line 11). Under these conditions the transition relations for statements 14-21 are, respectively,

$$\text{car}(T'[x-2]) = T'[x-1] = \text{black}(\text{car}(T[x-1]), \text{cdr}(T[x-1])), \tag{S-14}$$

$$\text{cdr}(T'[x-1]) = T'[x].tree = \text{red}(T[x].tree, T[x-1].tree), \tag{S-15}$$

$$\begin{aligned} &T'[x-1].tree = T[x-2].tree \\ &\quad \wedge\ (x-2 \neq root\ \rightarrow\ \text{cdr}(T'[x-2]) = T[x-2].tree), \end{aligned} \tag{S-16}$$

$$\begin{aligned} &T'[x-1].dir = T[x-2].dir \\ &\quad \wedge\ \big(x-2 \neq root\ \wedge\ T[x-1].dir \neq T[x-2].dir\ \rightarrow \\ &\qquad\quad T'[x-2] = \text{black}(\text{cdr}(T[x-2]), \text{car}(T[x-2]))\big), \end{aligned} \tag{S-17}$$

$$\begin{aligned} x'-2 = x-3\ \wedge\ &\big((\text{cdr}(T'[x-3]) = T[x-1]\ \wedge\ T[x-2].dir = left) \\ &\vee\ (\text{car}(T'[x-3]) = T[x-1]\ \wedge\ T[x-2].dir = right)\big), \end{aligned} \tag{S-19}$$

$$x'-1 = root. \tag{S-21}$$

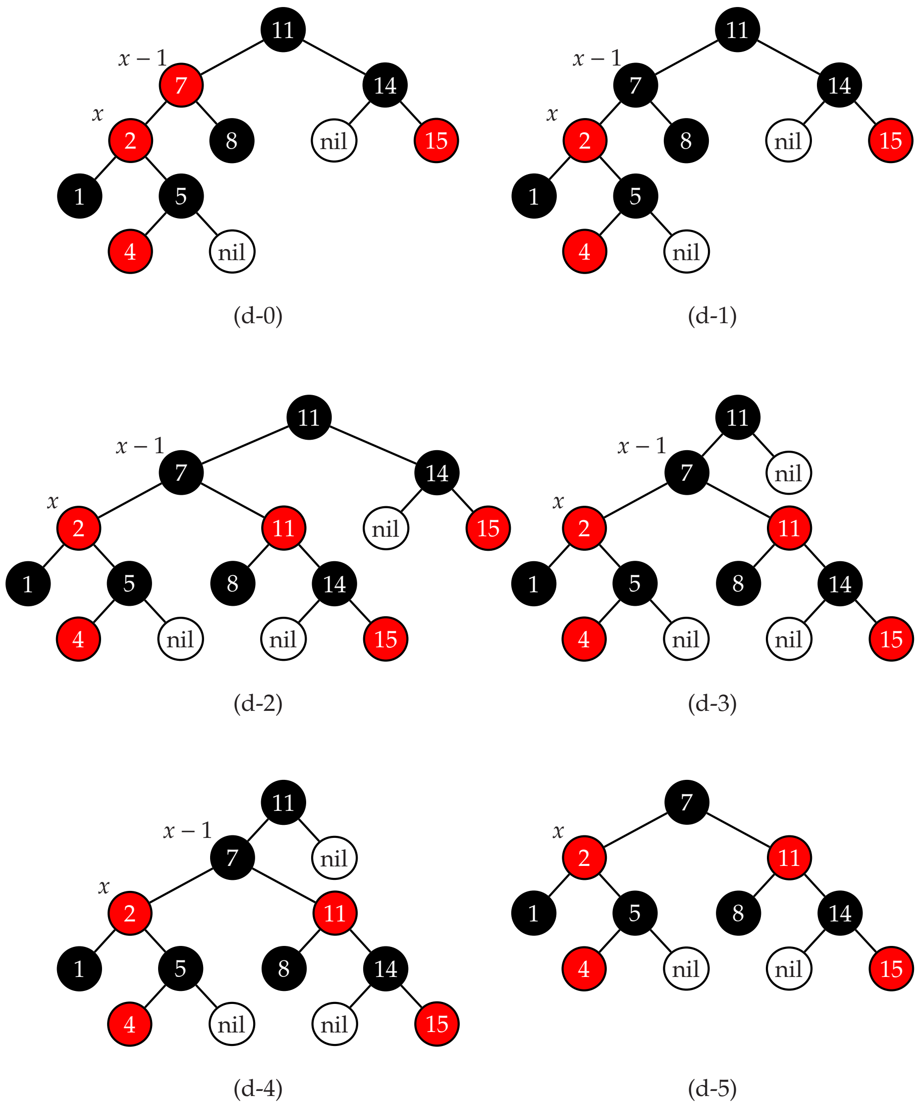


Fig. 5. A detailed run of RB-INSERTION-FIXUP step (d) with $x - 2 = root$

Assuming $x-2 = root$, the composite transition relation for statements 14-21 is

$$\begin{array}{ll} & \text{car}(T'[x-2]) = T'[x-1] = \text{black}(\text{car}(T[x-1]), T'[x].tree) \\ \wedge & T'[x].tree = \text{red}(T[x].tree, T[x-1].tree) \\ \wedge & T'[x-1].tree = T[x-2].tree \;\wedge\; T'[x-1].dir = T[x-2].dir \\ \wedge & x' - 1 = root. \end{array}$$

Assuming $x-2 \neq root$, the composite transition relation for statements 14-21 is

$$\begin{aligned}
& \mathrm{car}(T'[x-2]) = T'[x-1] = \mathrm{black}(\mathrm{car}(T[x-1]), T'[x].tree) \\
\wedge \quad & T'[x].tree = \mathrm{red}(T[x].tree, T[x-1].tree) \\
\wedge \quad & T'[x-1].tree = T[x-2].tree \ \wedge\ \mathrm{cdr}(T'[x-2]) = T[x-2].tree \\
\wedge \quad & T'[x-1].dir = T[x-2].dir \ \wedge\ \Big(T[x-1].dir \neq T[x-2].dir \ \rightarrow \\
& \qquad\qquad T'[x-2] = \mathrm{black}(\mathrm{cdr}(T[x-2]), \mathrm{car}(T[x-2]))\Big) \\
\wedge \quad & x'-2 = x-3 \ \wedge\ \Big((\mathrm{cdr}(T'[x-3]) = T[x-1] \ \wedge\ T[x-2].dir = left) \\
& \qquad\qquad \vee\ (\mathrm{car}(T'[x-3]) = T[x-1] \ \wedge\ T[x-2].dir = right)\Big) \ .
\end{aligned}$$

Figure 5 illustrates Case 3 by a run on tree (c) in Figure 1 (copied as (d-0)). Trees (d-1)-(d-5) are the outcomes of statements 14-17 and 21, respectively, under the assumption that $x - 2 = root$. Here (d-3) and (d-4) are the same because $T[x-1].dir = T[x-2].dir$ and hence statement 17 has no effect.

5 Conclusion

We presented a decidable theory of red-black trees, which is an extension of the theory of term algebras with two size functions. We showed how the red-black tree insertion algorithm can be analyzed using this theory. We plan to extend this theory to express local updates at an arbitrary pointed location in a tree. We note that adding a standard update function easily makes the first-order theory undecidable. We will investigate ways to enhance the expressiveness of the theory while maintaining the decidability.

References

1. Rolf Backofen. A complete axiomatization of a theory with feature and arity constraints. *Journal of Logical Programming*, 24(1&2):37–71, 1995.
2. Paolo Baldan, Andrea Corradini, Javier Esparza, Tobias Heindel, Barbara König, and Vitali Kozioura. Verifying red-black trees. In *Proceedings of the 1st International Workshop on the Verification of Concurrent Systems with Dynamic Allocated Heaps (COSMICAH 2005)*, 2005.
3. Cristiano Calcagno, Philippa Gardner, and Uri Zarfaty. Context logic and tree update. In *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 271–282. ACM Press, 2005.
4. Hubert Comon, Max Dauchet, Remi Gilleron, Denis Lugiez, Sophie Tison, and Marc Tommasi. *Tree Automata Techniques and Applications*. Electronic edition at `http://l3ux02.univ-lille3.fr/tata/tata.pdf`, 2002.
5. Hubert Comon and Catherine Delor. Equational formulae with membership constraints. *Information and Computation*, 112(2):167–216, 1994.
6. D. C. Cooper. Theorem proving in arithmetic without multiplication. In *Machine Intelligence*, volume 7, pages 91–99. American Elsevier, 1972.

7. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, Cambridge, Massachusetts, 2001.
8. J. Downey, R. Sethi, and R. E. Tarjan. Variations of the common subexpression problem. *Journal of the ACM*, 27:758–771, 1980.
9. H. B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 2001.
10. Peter Habermehl, Radu Iosif, and Tomas Vojnar. Automata-based verification of programs with tree updates. In *Proceedings of 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*, volume 3920 of *Lecture Notes in Computer Science*, pages 350–364. Springer-Verlag, 2006.
11. Konstantin Korovin and Andrei Voronkov. A decision procedure for the existential theory of term algebras with the Knuth-Bendix ordering. In *Proceedings of 15th IEEE Symposium on Logic in Computer Science*, pages 291 – 302. IEEE Computer Society Press, 2000.
12. Konstantin Korovin and Andrei Voronkov. Knuth-Bendix constraint solving is NP-complete. In *Proceedings of 28th International Colloquium on Automata, Languages and Programming (ICALP'01)*, volume 2076 of *Lecture Notes in Computer Science*, pages 979–992. Springer-Verlag, 2001.
13. Viktor Kuncak and Martin Rinard. The structural subtyping of non-recursive types is decidable. In *Proceedings of 18th IEEE Symposium on Logic in Computer Science*, pages 96–107. IEEE Computer Society Press, 2003.
14. M. J. Maher. Complete axiomatizations of the algebras of finite, rational and infinite tree. In *Proceedings of the 3rd IEEE Symposium on Logic in Computer Science*, pages 348–357. IEEE Computer Society Press, 1988.
15. A. I. Mal'cev. Axiomatizable classes of locally free algebras of various types. In *The Metamathematics of Algebraic Systems, Collected Papers*, chapter 23, pages 262–281. North Holland, 1971.
16. Greg Nelson and Derek C. Oppen. Fast decision procedures based on congruence closure. *Journal of the ACM*, 27(2):356–364, April 1980.
17. Derek C. Oppen. Reasoning about recursively defined data structures. *Journal of the ACM*, 27(3):403–411, July 1980.
18. C. R. Reddy and D. W. Loveland. Presburger arithmetic with bounded quantifier alternation. In *Proceedings of the 10th Annual Symposium on Theory of Computing*, pages 320–325. ACM Press, 1978.
19. Radu Rugina. Quantitative shape analysis. In *Proceedings of the 11th International Static Analysis Symposium (SAS'04)*, volume 3148 of *Lecture Notes in Computer Science*, pages 228–245. Springer-Verlag, 2004.
20. Tatiana Rybina and Andrei Voronkov. A decision procedure for term algebras with queues. *ACM Transactions on Computational Logic*, 2(2):155–181, 2001.
21. Ting Zhang, Henny B. Sipma, and Zohar Manna. Decision procedures for recursive data structures with integer constraints. In *the 2nd International Joint Conference on Automated Reasoning (IJCAR'04)*, volume 3097 of *Lecture Notes in Computer Science*, pages 152–167. Springer-Verlag, 2004.
22. Ting Zhang, Henny B. Sipma, and Zohar Manna. The decidability of the first-order theory of term algebras with Knuth-Bendix order. In Robert Nieuwenhuis, editor, *the 20th International Conference on Automated Deduction (CADE'05)*, volume 3632 of *Lecture Notes in Computer Science*, pages 131–148. Springer-Verlag, 2005.
23. Ting Zhang, Henny B. Sipma, and Zohar Manna. Decision procedures for term algebras with integer constraints. *Information and Computation*, 204:1526–1574, October 2006.

Compactness Properties for Stable Semantics of Logic Programs

Victor W. Marek[1] and Jeffrey B. Remmel[2]

[1] Department of Computer Science, University of Kentucky
Lexington, KY 40506-0046, USA
[2] Department of Mathematics,
University of California at San Diego, La Jolla, CA 92093

Abstract. Logic programming with stable logic semantics (SLP) is a logical formalism that assigns to sets of clauses in the language admitting negations in the bodies a special kind of models, called stable models. This formalism does not have the compactness property. We show a number of conditions that entail a form of compactness for SLP.

1 Introduction

When logicians look at a knowledge representation formalism they ask a variety of questions. They looks at a syntax, and on the semantics. In semantical considerations, since Gödel and Hilbert, logicians recognize that not all models of the formalism are *intended*. Quite often semantics that the agent (user of the formalism) assigns to to the syntax require significant restrictions. For instance logicians will often limit the classes of intended models to ones that give intended meaning to one or more predicates. An examples of this approach is in studies of ω-models of theories interpreting the arithmetic of positive integers. The reasoning agent may limit models by looking only at models of some size of the universe (for instance denumerable models or finite models) or of special form (for instance transitive models od set theories, or models that are levels of cumulative hierarchy, V_α. The rise of Computer Science and of Artificial Intelligence introduced new classes of formalisms and new classes of their models. While previously, the mathematical practice and imagintion of logicians provided the basic intuitions in selecting formalisms and their semantics, the applications on Computer Science and Artificial Intelligence resulted in a variety of new formalisms with intuitions in the areas previously limited to informal, philosophical investigations only.

It should be observed that since Gödel [13] logicians were concerned with the nature of computation. Related isssues, such as constructivity of argument were also discussed by logicians, esp. intuitionists, before the advent of modern computers. The relationship of complexity of sets of natural numbers, and existence of algorithms for deciding some mathematical theories has a very long history, and was present in the minds of mathematicians long before the viable computing devices were introduced.

Practical computing, and availability of symbolic computation in particular revolutionized the approach to foundational issues. today we know that logic and computation are inextricably connected. It is not our imagination only. We use computers not only as tools for solving equations, or finding values of functions but also as means to perform tasks that were reserved to human mental activities not long time ago.

As more and more solving tasks associated with human mental capabilities are delegated to computers, a variety of new formalisms for formaliztions of these tasks are introduced. Here we will be interested in one such formalism, so called *logic programming with stable semantics* which we will abbreviate SLP. This formalism, although using the name logic *programming* is grounded in the work of logicians on so-called *Horn formulas.* These formulas, introduced by A. Horn in late 1940ies, has been extensively studied for a variety of reasons and in various contexts. In the 1960ies, S.C. Kleene and R. Smullyan, showed that partial recursive functions can be represented in Horn logic. Then R. Kowalski introduced algorithms for testing the presence of an atom in the least model of Horn theory. These ideas were transformed into a practical programming language, called PROLOG by A. Colmerauer. This language one of the family of *declarative* programming languages and many of its implementations resulted in practical tools and are used in a variety of applications. The basic construct (in a somewhat idealized PROLOG) is a clause (also known as a rule), an expression of the form $p \leftarrow l_1, \ldots, l_k$ where p is an atom, and $l_1, \ldots, l_k$ are literals (atoms or negated atoms). This is a step beyond Horn logic proper, where $l_1, \ldots, l_k$ need to be atoms[1]. From the very beginning, the implementations of PROLOG admitted admitted negated atoms among $l_1, \ldots, l_k$ and attempted to assign to the clauses a meaning that may informally be stated like this: "*if* $l_1, \ldots, l_k$ *have been established then we establish* p". This works well in Horn case, resulting in the *intended* semantics of the least model. But when we admit negated literals among $l_1, \ldots, l_k$ then the interpretation of rules is not clear. Among a large number of proposals, the one that, essentially, gained concensus as a correct one, is the *stable semantics* due to M. Gelfond and V. Lifschitz [12]. Stable semantics is, in effect, an interpretation of rules as *default rules* studied by R. Reiter. The consensus on stable semantics has been reached in the mid 1990ies. Today, stable sematics and its extensions to so-called answer sets is a basis for so-called *Answer Set Programming* (ASP) with several practical implementations.

When logicians look at a formalism such as SLP, they ask a variety of questions pertaining both to the expressive power of the formalism, and to logical properties of its semantics. The question of expressibility has been studied early and it turned out that the problem of existence of stable models of *finite* propositional logic programs (i.e. finite collections of propositional clauses) is an NP-complete problem. For infinite programs (and in particular for finite programs admitting function symbols -such programs are interpreted as their *ground versions*) the SLP turned out to be overexpressive [21]. That is very complex sets of

[1] We did not discuss here the issue of unification and its role in PROLOG. We limit ourselveds to the propositional case, although we admit infinite programs.

integers, more complex than anything considered computable, are expressed by such models (see also remarks below). This resulted in very significant limitations for ASP.

One question that the logicians will ask when dealing with a formalism is its compactness. This question has been asked in case of SLP by M. Gelfond. Motivated by this question we quickly found counterexamples that will be presented in the next Section. But a natural way to look at negative answers is to ask if general properties that are not true, can be approximated by some other properties. In the case of our problem, the issue was this: *compactness for* SLP *does not hold in general; are there some classes of programs where compactness is guaranteed*? In fact, one such class has been found by Bonatti [5]. Those are programs which split into two parts, lower and upper in such a way that the upper part reduces to a Horn program on every stable model of the lower part. Here we will discuss three different syntactically motivated classes of programs with the compactness property. We introduce a proof-theoretic techniques for handling stable models. This technique uses *proof schemes*, an extension of the concept of proof to non-monotonic context. The main difference betwee the proof schemes and proofs as studied in proof theory is that proof schemes have guards (we call them *supports*) that allow the proof to be executed. The support is a set of atoms that need to be *absent* in the context to execute the proof scheme. We can consider inclusion-minimal supports, and distinguish programs which we call finite-support programs which have only finite number of such minimal supports. One of our results is that such programs are characterized by the continuity of so-called Gelfond-Lifschitz operator, and that these programs do have compactness property. The second class of programs is based on existence of schemes of a certain form. This class of programs, called locally-finite programs has been introduced by Cenzer and Remmel in [6]. We show that the programs in this class also have compactness property. The third class considered in this paper is based on Scott's notion of consistency property. Again we find that this is syntactic property (different from the previous two) that entail compactness. We discuss the ramifications in the Conclusions section.

2 Motivating Examples

The Compactness Theorem for propositional logic says that if Θ is a collection of sentences and every finite subset of Θ has a model, then Θ has a model. In this paper, we study the analogue of compactness for the stable logic semantics of propositional logic programs. At first glance, there are some obvious differences between stable models of propositional logic programs and models of sets of sentences in a propositional logic. For example, if T is a set of sentences in a propositional logic and $S \subseteq T$, then it is certainly the case that every model of T is a model of S. Thus a set of propositional sentences T has the property that if T has a model, then every subset of T has a model. This is certainly not true for propositional logic programs. For example, consider the following logic example.

Example 1. Let P consists of the following two clauses:

$C_1 = a \leftarrow \neg a, \neg b$ and
$C_2 = b \leftarrow$

Then it is easy to see that $\{b\}$ is stable model of P. However the subprogram P' consisting of just clause C_1 does not have a stable model. That is b is not in any stable model of P' since there is no clause in P' whose head is b. Thus the only possible stable models of P' would be $M_1 = \emptyset$ and $M_2 = \{a\}$. But it is easy to see that both M_1 and M_2 are not stable models of P'. △

Thus the best that we could hope for as an analogue of compactness for propositional logic programs would be the following principle which we will call *Comp*: (*Comp*) *If for any finite program $P' \subseteq P$, there exist a finite program P'' such that $P' \subseteq P'' \subseteq P$ such that P'' has a stable model, then P has a stable model.*

Our next example, will show that (Comp) is not true for propositional logic programs.

Example 2. Let P be an infinite propositional program consisting of the following clauses.

1. $a \leftarrow \neg a, \neg b$
2. $b \leftarrow \neg c_i$, for all $i \geq 0$
3. $c_i \leftarrow$ for all $i \geq 0$.

Note that P has no stable model. That is, if M is a stable model of P, then $c_i \in M$ for all i by the clauses in the group (3). Thus $b \notin M$ since the only way to derive b is from one of the clauses in the group (2) and these are all blocked for M. Now consider a. We cannot have $a \in M$ since the only way to derive a is via the clause (1) but it would be blocked if $a \in M$. Thus $a \notin M$. However if $a \notin M$ and $b \notin M$, then we must have $a \in M$ by clause (1). This is a contradiction so that P has no stable model.

Now we fix $n \geq 0$ and consider a finite subprogram P_n of P. consisting of the following clauses.

1. $a \leftarrow \neg a, \neg b$
2. $b \leftarrow \neg c_i$, for all $0 \leq i \leq n+1$
3. $c_i \leftarrow$, for all $0 \leq i \leq n$.

It is easy to see that P_n has an exactly one stable model, namely $\{b, c_1, \ldots, c_n\}$. It is easy see that any finite subset F of P is contained in some P_n. Therefore the principle (*Comp*) fails for P. △

Since the analogue of compactness fails for propositional logic programs, it is natural to ask if there are certain conditions (*Cond*) such that whenever a propositional logic program statisfies (*), then (*Comp*) holds. In turns out that there are several conditions that have appeared in the literature that ensure that (*Comp*) holds. The main goal of this paper is survey those conditions.

3 Stable Models and Proof Schemes

Before we can state some conditions which ensure property (*Comp*), we need to formally introduce stable models and the concept of proof schemes.

For an introductory treatment of logic programs, see [1]. Here is a brief self-contained account of their stable models [12]. Let us assume as given a fixed first order language $\mathcal{L}$ based on predicate letters, constants, and function symbols. The Herbrand base of the language is defined as the set $B_{\mathcal{L}}$ of all ground atoms of the language $\mathcal{L}$. A literal is an atom or its negation, a ground literal is an a ground atom or its negation. A logic program P is a set of "program clauses", that is, an expression of the form:

$$p \leftarrow l_1, \ldots, l_k \tag{1}$$

where p is an atom, and $l_1, \ldots, l_k$ is a list of literals.

Then p is called the conclusion of the clause, the list $l_1, \ldots, l_k$ is called the body of the clause. Ground clauses are clauses without variables. Horn clauses are clauses with no negated literals, that is, with atomic formulas only in the body. We will denote by $Horn(P)$ the part of the program P consisting of its Horn clauses. Horn clause programs are programs P consisting of Horn clauses. That is for Horn programs P, $P = Horn(P)$. Each such program has a least model in the Herbrand base determined as the least fixed point of a continuous operator T_P representing 1-step Horn clause logic deduction ([16]).

Informally, the *knowledge* of a logic program P is the set of clauses in P with no negated literals in the bodies, that is, the Horn clauses. The set of beliefs of a logic program P is the set of clauses with negated literals occurring in the bodies of clauses of P. This use of language is sufficiently suggestive to guide the reader to translations of many nonmonotone theories in other reasoning systems into equivalent logic programs so that extensions as models for the non-monotonic theory correspond to stable models as models for the logic program.

A ground instance of a clause is a clause obtained by substituting ground terms (terms without variables) for all variables of the clause. The set of all ground instances of the program P is called $ground(P)$.

Let M be any subset of the Herbrand base. A ground clause is said to be M-*applicable* if the atoms whose negations are literals in the body are not members of M. Such clause is then *reduced* by eliminating remaining negative literals. This *monotonization* $GL(P, M)$ of P with respect to M is the propositional Horn clause program consisting of reducts of M-applicable clauses of $ground(P)$ (see Gelfond-Lifschitz [12]). Then M is called a *stable model* for P if M is the least model of the Horn clause program $GL(M, P)$. We denote this least model as $F_P(M)$. It is easy to see that a stable model for P is a minimal model of P ([12]). We denote by $Stab(P)$ the set of all stable models of P. There may be no, one, or many stable models of P.

We should note that the syntactical condition of stratification of Apt, Blair, and Walker [3] singles out programs with a well-behaved, unique stable model, but there is no reason to think that in belief revision one could move from stratified program to stratified program; but how one might do this is an interesting and challenging question.

What kind of proof theory is appropriate for logic programs? The key idea for our proofs is that of a proof scheme with conclusion an atom p. Proof schemes are intended to reflect exactly how p is a finitary but non-monotonic consequence of P.

Proof schemes represent a form of resolution proofs tailored to fit non-monotonic proofs. Two items distinguis them from the resolution proof trees. first, the derivation process is represented in a linear fashion, as sequences. Second, the negative information, controlling applicability of proof schemes is represented explicitely.

A proof scheme uses, as in Horn logic, the positive information present in the positive literals of bodies of clauses of P, but proof schemes also have to respect the negative information present in the negative literals of bodies of clauses. With this motivation, here is the definition. A *proof scheme* for p with respect to P is a sequence of triples $< \langle p_l, C_l, S_l \rangle >_{1 \le l \le n}$, with n a natural number, such that the following conditions all hold.

1. Each p_l is in $B_{\mathcal{L}}$. Each C_l is in $ground(P)$. Each S_l is a finite subset of $B_{\mathcal{L}}$.
2. p_n is p.
3. The S_l, C_l satisfy the following conditions. For all $1 \le l \le n$, one of **(a)**, **(b)**, **(c)** below holds.
   (a)C_l is $p_l \leftarrow$, and S_l is S_{l-1},
   (b)C_l is $p_l \leftarrow \neg s_1, \ldots, \neg s_r$ and S_l is $S_{l-1} \cup \{s_1, \ldots, s_r\}$, or
   (c)C_l is $p_l \leftarrow p_{m_1}, \ldots, p_{m_k}, \neg s_1, \ldots, \neg s_r$, $m_1 < l$,...,$m_k < l$, and S_l is $S_{l-1} \cup \{s_1, \ldots, s_r\}$.

(We put $S_0 = \emptyset$).

The atoms p_i occur in a proof schem in the order they are derived. Because an atom may be the head of several rules, we also list the specific rule C_i that is used to derive p_i. The sets S_i control the applicability of proof scheme. The idea here is that, unlike usual proofs, the proof schemes carry within themself the information on its applicability with respect to potential stable models. To have the scheme applicable with respect to M, the support of that scheme (S_n) must be disjoint from M. Let us suppose that $\varphi =< \langle p_l, C_l, S_l \rangle >_{1 \le l \le n}$ is a proof scheme. Then $conc(\varphi)$ denotes atom p_n and is called the *conclusion* of φ. Also, $supp(\varphi)$ is the set S_n and is called the *support* of φ.

Condition (3) tells us how to construct the S_l inductively, from the S_{l-1} and the C_l. The set S_n consists of the negative information of the proof scheme.

A proof scheme may not need all its lines to prove its conclusion. It may be possible to omit some clauses and still have a proof scheme with the same conclusion. If we omit as many clauses as possible, retaining the conclusion but still maintaining a proof scheme, this is a *minimal proof scheme* with that conclusion. It may be possible to do this with many distinct results, but obviously there are only a finite number of ways altogether to trim a proof scheme to a minimal proof scheme with the same conclusion, since no new clauses are ever introduced. Of course, a given atom may be the conclusion of no, one, finitely many, or infinitely many different minimal proof schemes. These differences are clearly computationally significant if one is searching for a justification of a conclusion. The apparatus needed to discuss this was introduced in [17].

Formally, we preorder proof schemes φ, ψ by $\varphi \prec \psi$ if

1. φ, ψ have same conclusion,
2. Every clause in φ is also a clause of ψ.

The relation $\prec$ is reflexive, transitive, and well-founded. Minimal elements of $\prec$ are called minimal proof schemes.

Here are some propositions from [17,18].

Proposition 1. *Let P be a program and $M \subseteq B_{\mathcal{L}}$. Let p be an atom. Then p is in $F_P(M)$ if and only if there exists a proof scheme with conclusion p whose support is disjoint from M.*

If Z is a set of atoms we let $\neg Z$ be the conjunction of all the negations of atoms of Z. Now we fix program P and atom p for the discussion. Associate with the atom p a (possibly infinitary) Boolean equation E_p

$$p \leftrightarrow (\neg Z_1 \vee \neg Z_2 \vee \ldots), \tag{2}$$

where the $Z_1, Z_2 \ldots$ is a (possibly infinite) list of supports of all minimal proof schemes with conclusion p with respect to P. In fact, for our purposes it is enough to list only the inclusion-minimal supports. This is called a *defining equation* for p with respect to P. If there are infinitely many distinct minimal supports for proof schemes with conclusion p, this will be an infinitary equation. We make two other conventions about the defining equation of p. Namely

1. If p is not the conclusion of any proof scheme with respect to P, then the defining equation for p is $p \leftrightarrow \perp$, which is equivalent to $\neg p$. Hence in this case, $\neg p$ must hold in every stable model of P.
2. If p has a proof scheme with empty support, that is, a proof scheme which uses only Horn clauses, then the defining equation for p is equivalent to $\top$. In this case, p belongs to all stable models of P.

The set Eq_P of all equations E_p obtained as p ranges over the Herbrand base is called a defining system of equations for program P.

Example 3. Let P be a program:

$p(0) \leftarrow \neg q(X)$
$nat(0) \leftarrow$
$nat(s(X)) \leftarrow nat(X)$.

Then for each n, $< \langle p(0), p(0) \leftarrow \neg q(s^n(0)), \{q(s^n(0)\} \rangle >$ is a minimal proof scheme with conclusion $p(0)$. Thus atom $p(0)$ has an infinite number of minimal proof schemes with respect to program P. $\triangle$

The defining equations characterize stable models of logic programs.

Proposition 2. *Let P be a logic program with defining system of equations Eq_P. Let M be a subset of the Herbrand universe $B_{\mathcal{L}}$. Then M is a stable model for P if and only if $M \cup \{\neg q \colon q \in B_{\mathcal{L}} \setminus M\}$ is a solution of the system Eq_P.*

Here is another characterization of stable models, this time via proof schemes.

Proposition 3. *Let P be a program. Also, suppose that M is a subset of the Herbrand universe $B_{\mathcal{L}}$. Then M is a stable model of P if and only if, for every $p \in B_{\mathcal{L}}$, it is true that p is in M if and only if there exists a proof scheme φ with conclusion p such that the support of φ is disjoint from M.*

Given a logic program P, Dung and Kanchanasut in [10] introduce a construction of a purely negative propositional program P' (that is a program consisting of clauses of the form $p \leftarrow \neg q_1, \ldots, \neg q_n$, $n \geq 0$, where $p, q_1, \ldots q_n$ are ground atoms) with the property that P and P' have the same stable models. It is very easy to construct P' out of the set of proof schemes (minimal proof schemes are sufficient for that purpose). Namely, let, for a given $p \in B_{\mathcal{L}}$, $Z_1^p, Z_2^p, \ldots$ list the supports of all minimal proof schemes for p in P. Then the set of purely negative clauses

$$\{p \leftarrow \neg Z_i^p : p \in B_{\mathcal{L}}, i \in n\}$$

is a purely negative program P' with the desired property that $Stab(P) = Stab(P')$.

4 FSP Logic Programs and Their Continuity Properties

Our first condition that ensures a program P has property ($Comp$) is the notion of *finitary support programs* introduced by Marek, Nerode, and Remmel [21]. That is, Marek, Nerode, and Remmel examined logic programs P such that every defining equation for every atom p is finite. We will use show in the next section that FSP indeed the property *Comp*. In this section we show continuity property of an operator associated with FSP logic programs.

The FSP property is equivalent to requiring that every atom has only a finite number of inclusion-minimal supports of minimal proof schemes. Such a program may have the property that there is an atom which is the conclusion of infinitely many different minimal proof schemes, but these schemes have only finitely many supports altogether among them.

Example 4. Let P be the program:

$p(0) \leftarrow q(X)$
$q(X) \leftarrow \neg r(0)$
$nat(0) \leftarrow$
$nat(s(X)) \leftarrow nat(X)$.

Then the atom $p(0)$ is the conclusion of infinitely many proof schemes:

$$< \langle q(s^n(0)), q(s^n(0)) \leftarrow \neg r(0), \{r(0)\}\rangle, \langle p(0), p(0) \leftarrow q(s^n(0)), \{r(0)\}\rangle >$$

as n ranges over ω. But the single minimal support of all these proof schemes is $\{r(0)\}$. That is, whenever $r(0)$ is not in M, then $p(0)$ will be in $F_P(M)$. △

The above program motivates the following definition.

Definition 1. *We say that a* finitary support *program (*FSP *program, for short) is a logic program such that for every atom p, there is a finite set of finite sets S, which are exactly the inclusion-minimal supports of all those minimal proof schemes with conclusion p.*

We now study the FSP property. It turns out that this property is equivalent to the continuity property for a suitably defined operator. This is precisely the same operator whose square (that is two-fold application) determines the *monotonic* operator whose least and largest fixpoints determine the well-founded model of the program ([25]).

We associate an operator with each logic program as follows.

Definition 2. *Let P be a program. The operator $F_P : \mathcal{P}(B_\mathcal{L}) \to \mathcal{P}(B_\mathcal{L})$ is defined as follows: If $S \subseteq B_\mathcal{L}$ then $F_P(S)$ is the set of all atoms in $B_\mathcal{L}$ for which there exists a proof scheme p such that $supp(p) \cap S = \emptyset$. Thus F_P assigns to S the set $F_P(S)$.*

Proposition 4. *The operator F_P is anti–monotonic, that is, if $S_1 \subseteq S_2$, then $F_P(S_2) \subseteq F_P(S_1)$.*

Proposition 5. *The operator F_P is lower half-continuous; that is, if $\langle S_n \rangle_{n\in\omega}$ is a monotone decreasing sequence of subsets of $B_\mathcal{L}$ then $\bigcup_{n\in\omega} F_P(S_n) = F_P(\bigcap_{n\in\omega} S_n)$.*

Proposition 6. *Let P be a logic program. Then following conditions are equivalent:*

(a) P is an FSP logic program.
(b) F_P is an upper half-continuous operator; that is, whenever $\langle S_n \rangle_{n\in\omega}$ is a monotone increasing sequence of subsets of $B_\mathcal{L}$, we have

$$\bigcap_{n\in\omega} F_P(S_n) = F_P(\bigcup_{n\in\omega} S_n).$$

5 Coding Stable Models into Trees

The key result to prove that at FSP programs have property (*Comp*) is the proof of a result of Marek, Remmel, and Nerode [18,21] to the effect that for any recursive program P, there is recursive tree $T_P \subseteq \omega^\omega$ such that there is an effective one-one degree preserving correspondence between the set of stable models of P, $Stab(P)$, and the set of infinite paths throught T_P, $Path(T_P)$.

In this section, we shall give the neccessary recursion theoretic background to make precise and to prove the result of Marek, Nerode, and Remmel that given any recursive logic program P, there is a recursive tree T such that there is an effective 1-1 degree-preserving map between the set of stable models of P and the set of paths through T. Then we shall show that the fact that condition (*Comp*) holds for FSP programs follows from their proof.

5.1 Recursive Programs

When we discuss finite programs then we can easily read off a recursive representation of the Herbrand base. The reason is that the alphabet of such a program, that is, the set of predicate symbols and function symbols that appear in the program, is finite. The situation changes when P is an infinite predicate logic program representable with a recursive set of Gödel numbers. When we read off the enumeration of the alphabet of the program from an enumeration of the program itself, there is no guarantee that the alphabet of P is recursive. In particular the Herbrand base of the program is recursively enumerable but may not necessarily be recursive.

For the purposes of this paper, we define a program P to be recursive if not only the set of its Gödel numbers is recursive, but also the resulting representation of the Herbrand base is recursive.

5.2 Recursively FSP Programs

A *recursively* FSP program is an FSP recursive program such that we can uniformly compute the finite family of supports of proof schemes with conclusion p from p. The meaning of this is obvious, but we need a technical notation for the proofs. Start by listing the whole Herbrand base of the program, $B_{\mathcal{L}}$ as a countable sequence in one of the usual effective ways. This assigns an integer (Gödel number) to each element of the base, its place in this sequence. This encodes finite subsets of the base as finite sets of natural numbers, all that is left is to code each finite set of natural numbers as a single natural number, its *canonical index*. To the finite set $\{x_1, \ldots, x_k\}$ we assign as its canonical index $can(\{x_1, \ldots, x_k\}) = 2^{x_1} + \ldots + 2^{x_k}$. We also set $can(\emptyset) = 0$. If program P is FSP, and the list, in order of magnitude, of Gödel numbers of all minimal support of schemes with conclusion p is

$$Z_1^p, \ldots, Z_{l_r}^p,$$

then we define a function $su^P \colon B_{\mathcal{L}} \to \omega$ as below.

$$p \mapsto can(\{can(Z_1^p), \ldots, can(Z_{l_r}^p)\})$$

We call a logic program P a *recursively* FSP program if it is FSP and the function su^P is recursive.

5.3 Tools from Recursion Theory

Let $\omega = \{0, 1, 2, \ldots\}$ denote the set of natural numbers and let $<,>\colon \omega \times \omega \to \omega - \{0\}$ be some fixed one-to-one and onto recursive pairing function such that the projection functions π_1 and π_2 defined by $\pi_1(< x, y >) = x$ and $\pi_2(< x, y >) = y$ are also recursive. We extend our pairing function to code n-tuples for $n > 2$ by the usual inductive definition, that is $< x_1, \ldots, x_n >=< x_1, < x_2, \ldots, x_n >>

for $n \geq 3$. We let $\omega^{<\omega}$ denote the set of all finite sequences from ω and $2^{<\omega}$ denote the set of all finite sequences of 0's and 1's. Given $\alpha = (\alpha_1, \ldots, \alpha_n)$ and $\beta = (\beta_1, \ldots, \beta_k)$ in $\omega^{<\omega}$, we write $\alpha \sqsubseteq \beta$ if α is initial segment of β, that is, if $n \leq k$ and $\alpha_i = \beta_i$ for $i \leq n$. For the rest of this paper, we identify a finite sequence $\alpha = (\alpha_1, \ldots, \alpha_n)$ with its code $c(\alpha) =< n, < \alpha_1, \ldots, \alpha_n >>$ in ω. We let 0 be the code of the empty sequence $\emptyset$. Thus, when we say a set $S \subseteq \omega^{<\omega}$ is recursive, recursively enumerable, etc., we mean the set $\{c(\alpha) \colon \alpha \in S\}$ is recursive, recursively enumerable, etc. A *tree* T is a nonempty subset of $\omega^{<\omega}$ such that T is closed under initial segments. A function $f \colon \omega \to \omega$ is an infinite *path* through T if for all n, $(f(0), \ldots, f(n)) \in T$. We let $[T]$ denote the set of all infinite paths through T. A set A of functions is a Π^0_1-class if there is a recursive predicate R such that $A = \{f \colon \omega \to \omega \ \colon \forall_n (R(< n, < f(0), \ldots, f(n-1) >>)\}$. A Π^0_1-class A is *recursively bounded* if there is a recursive function $g \colon \omega \to \omega$ such that $\forall_{f \in A} \forall_n (f(n) \leq g(n))$. It is not difficult to see that if A is a Π^0_1-class, then $A = [T]$ for some recursive tree $T \subseteq \omega^{<\omega}$. We say that a tree $T \subseteq \omega^{<\omega}$ is *highly recursive* if T is a recursive, finitely branching tree such that there is a recursive procedure which given $\alpha = (\alpha_1, \ldots, \alpha_n)$ in T produces a canonical index of the set of immediate successors of α in T, that is, produces a canonical index of $\{\beta = (\alpha_1, \ldots, \alpha_n, k) \colon \beta \in T\}$. If A is a recursively bounded Π^0_1-class, then $A = [T]$ for some highly recursive tree $T \subseteq \omega^{<\omega}$, see [14]. We let A' denote the jump of the set A and $\mathbf{0}'$ denote the jump of the empty set. Thus $\mathbf{0}'$ is the degree of any complete r.e. set. We say that a tree $T \subseteq \omega^{<\omega}$ is *highly recursive in* $\mathbf{0}'$ if T is a finitely branching tree such that T is recursive in $\mathbf{0}'$ and there is an effective procedure which given an $\mathbf{0}'$-oracle and an $\alpha = (\alpha_1, \ldots, \alpha_n)$ in T produces a canonical index of the set of immediate successors of α in T, that is, produces a canonical index of $\{\beta =< \alpha_1, \ldots, \alpha_n, k) \colon \beta \in T\}$.

We say that there is an effective one-to-one degree preserving correspondence between the set of stable models of a recursive program P, $\mathit{Stab}(P)$, and the set of infinite paths $[T]$ through a recursive tree T if there are indices e_1 and e_2 of oracle Turing machines such that

(i) $\forall_{f \in [T]} \{e_1\}^{gr(f)} = M_f \in \mathit{Stab}(P)$,
(ii) $\forall_{M \in \mathit{Stab}(P)} \{e_2\}^M = f_M \in [T]$, and
(iii) $\forall_{f \in [T]} \forall_{M \in \mathit{Stab}(P)} (\{e_1\}^{gr(f)} = M$ if and only if $\{e_2\}^M = f)$.

Here $\{e\}^B$ denotes the function computed by the e^{th} oracle machine with oracle B. We write $\{e\}^B = A$ for a set A if $\{e\}^B$ is a characteristic function of A. If f is a function $f \colon \omega \to \omega$, then $gr(f) = \{< x, f(x) > \colon x \in \omega\}$. Condition (i) says that the infinite paths of the tree T, uniformly produce stable models via an algorithm with index e_1. Condition (ii) says that stable models of P uniformly produce branches of the tree T via an algorithm with index e_2. A is *Turing reducible* to B, written $A \leq_T B$, if $\{e\}^A = B$ for some e. A is *Turing equivalent* to B, written $A \equiv_T B$, if both $A \leq_T B$ and $B \leq_T A$. Thus condition (iii) asserts that our correspondence is one-to-one and if $\{e_1\}^{gr(f)} = M_f$, then f is Turing equivalent to M_f. Finally, given sets A and B, we let $A \oplus B = \{2x : x \in A\} \bigcup \{2x+1 : x \in B\}$.

5.4 Representing Programs by Trees

Theorem 1. *We suppose that the first order language $\mathcal{L}$ has infinitely many ground atoms.*

1. *Then for any recursive program P in $\mathcal{L}$, there exists a recursive tree $T \subseteq \omega^{<\omega}$ and an effective one-to-one degree preserving correspondence between the set of all stable models of P, $Stab(P)$ and $[T]$, the set of all infinite paths through T.*
2. *If, in addition to the hypothesis of (1), program P is FSP, then the tree T is finite splitting.*
3. *If, in addition to the hypothesis of (2), program P is recursively FSP, then the tree T is a highly recursive tree.*

Theorem 1 allows us to prove that the class of denumerable *FSP* has the property)*Comp*).

Theorem 2. *Let us suppose that P is a countable FSP program. Then P satisifies property (Comp). That is, if for any finite program $P' \subseteq P$, there exist a finite program P'' such that $P' \subseteq P'' \subseteq P$ such that P'' has a stable model, then P has a stable model.*

6 Locally Determined Logic Programs

In this section, we shall describe another condition on propositional logic programs that will ensure property (*Comp*) that has appeared in the literature. Namely, we shall introduce the notion of locally determined logic programs due to Cenzer and Remmel [6]. For the rest of this paper, we shall only consider countable logic programs P. Thus whenever we say that P is a logic program, we shall always assume that P is countable.

The informal notion of a locally determined logic program P is one in which the existence of a proof scheme for an atom a_i (or the lack of existence thereof) can be determined by examining only clauses or proof schemes involving some initial segment of the Herbrand base of P. More formally, we fix some countable logic program P and some listing $a_0, a_1, \ldots$ of the atoms of Herbrand base of P without repetitions. (We shall make the convention that if P is a recursive logic program, then there is some recursive function $h : \omega \to \omega$ such that $h(i) = a_i$.) Then given a proof scheme or a clause ψ, we write $max(\psi)$ for the $max(\{i : a_i \text{ occurs in } \psi\})$. We shall write P_n for the set of all clauses $C \in P$ such that $max(C) \leq n$ and let $A_n = \{a_0, \ldots, a_n\}$.

Definition 3. *We shall say that n is a level of P if for all $S \subseteq \{a_0, \ldots, a_n\}$ and all $i \leq n$, whenever there exists a proof scheme ϕ such that $cln(\phi) = a_i$ and $supp(\phi) \cap S = \emptyset$, then there exists a proof scheme ψ such that $cln(\psi) = a_i$, $supp(\psi) \cap S = \emptyset$ and $max(\psi) \leq n$. Note that by definition, the Herbrand base H_{P_n} of P_n is contained in A_n. We let $lev(P) = \{n : n \text{ is a level of } P\}$.*

The following result has essentially been proven in [9,15]

Theorem 3. *Suppose that n is a level of P and E is a stable model of P. Then $E_n = E \cap \{a_0, \ldots, a_n\}$ is a stable model of P_n.*

Definition 4. *We shall say that a logic program P is* **locally determined** *if P is countable and there are infinitely many n such that n is a level of P.*

Example 5. Let P be the infinite logic program with the following set of clauses.

1. $a_{2i} \leftarrow \neg a_{2i+1}$, for all $i \in \omega$
2. $a_{2i+1} \leftarrow \neg a_{2i}$, for all $i \in \omega$.

Thus the Herbrand base of P is $\{a_0, a_1, \ldots\}$. It is easy to see that S is stable model of P if and only if $|S \cap \{a_{2i}, a_{2i+1}\}| = 1$ for all $i \in \omega$. In fact, one can easily prove that $lev(P) = \{2i+1 : i \in \omega\}$. Moreover, it is clear that P is locally finite. △

Example 6. Let Q be the logic program with the following set of clauses for all $i \in \omega$.

1. $a_{3i} \leftarrow \; \neg a_{3i+1}, \neg a_{3i-2}, \ldots \neg a_1, \neg a_{3i+2}, \neg a_{3i-1}, \ldots \neg a_2$
2. $a_{3i+1} \leftarrow \; \neg a_{3i}, \neg a_{3i-3}, \ldots \neg a_0, \neg a_{3i+2}, \neg a_{3i-1}, \ldots \neg a_2$
3. $a_{3i+2} \leftarrow \; \neg a_{3i}, \neg a_{3i-3}, \ldots \neg a_0, \neg a_{3i+1}, \neg a_{3i-2}, \ldots \neg a_1$
4. $a_{3i} \leftarrow \; a_{3i+3}$
5. $a_{3i+1} \leftarrow \; a_{3i+4}$
6. $a_{3i+2} \leftarrow \; a_{3i+5}$

The Herbrand base of Q is $\{a_0, a_1, \ldots\}$. It is easy to see that P has exactly 3 stable models, namely, $S_0 = \{a_{3i} : i \in \omega\}$, $S_1 = \{a_{3i+1} : i \in \omega\}$ and $S_2 = \{a_{3i+2} : i \in \omega\}$. In this case, Q is not locally finite since for any $i > 0$, the following set of clauses can be used to construct a minimal proof scheme of a_0 with support equal to

$$\{a_{3i+1}, a_{3i-2}, \ldots, a_1, a_{3i+2}, a_{3i-1}, \ldots, a_2\}.$$

$$\begin{aligned}
&a_{3i} \leftarrow \neg a_{3i+1}, \neg a_{3i-2}, \ldots \neg a_1, \neg a_{3i+2}, \neg a_{3i-1}, \ldots \neg a_2 \\
&a_{3i-3} \leftarrow a_{3i} \\
&a_{3i-6} \leftarrow a_{3i-3} \\
&\vdots \\
&a_0 \leftarrow a_3.
\end{aligned}$$

However we claim that $lev(P) = \{3i+2 : i \in \omega\}$. That is, let us fix $n \geq 0$ and let us suppose that $T \subseteq \{a_i : i \leq 3n+2\}$ and let us suppose that ψ is a proof scheme with conclusion a_r where $supp(\psi) \cap T = \emptyset$ and $r \leq 3n+2$. We shall consider three cases.

Case 1. $T = \emptyset$.
In this case it is easy to see that every element of $\{a_i : i \leq 3n+2\}$ which is the conclusion of a proof scheme of P_{3n+2} of length one using one of the clauses (1), (2), and (3).
Case 2. There exist a_{3i+s} and a_{3j+t} in T where $s \neq t$ and $s, t \in \{0, 1, 2\}$.

In this case all clauses of the form (1), (2) or (3) where the head of the clause is some a_k with $k > 3n+2$ cannot be part of a proof scheme ψ such that $supp(\psi) \cap T = \emptyset$. Thus the only clauses of the form (1), (2), or (3) that can be part of ψ are clauses from P_{3n+2}. However, in that case, the only clauses of the form (4), (5), and (6) that can be part of ψ must also be in P_{3n+2} because there is no way that we can derive an element a_k with $k \geq 3n+2$ that is in the body of a clause of the form (4), (5), and (6) if we can only use clauses in ψ of type (1), (2), and (3) in ψ from P_{3n+2}. Here we are using the fact that ψ is a minimal proof scheme. It follows that ψ must be a proof scheme for P_{3n+2}.
Case 3. Conditions of Case 1 or Case 2 do not hold.

In this case, T must be contained in one of the stable models S_0, S_1, or S_2. We shall assume that $T \subseteq S_0$ since the other two cases are similar. Since $T \cap S_0 \neq \emptyset$, there can be no clause of type (2) and (3) in ψ which in not in P_{3n+2}. Now we suppose that a clause of type (3) occurs in ψ with head a_{3j} where $j > n$. Then this clause combined with clauses of type (4) in ψ can be used to derive elements of the form a_{3i} with $i \leq n$. But for all $i \leq n$, we can clearly immediately derive a_{3i} form the clause

$$a_{3i} \leftarrow \neg a_{3i+1}, \neg a_{3i-2}, \ldots, \neg a_1, \neg a_{3i+2}, \neg a_{3i-1}, \ldots \neg a_2 \tag{3}$$

which lies in P_{3n+2} and whose constraints do not intersect T. It follows that we can construct an minimal proof scheme ψ' in P_{3n+2} with the same conclusion as ψ such that $supp(\psi') \cap T = \emptyset$.

It follows that $3n+2$ is level of P for all n. Moreover it is clear from the clauses of type (1) and (2) that $3n$ and $3n+1$ are not levels of P so that $lev(P) = \{3n+2 : n \in \omega\}$ as claimed. △

Example 7. In this example, we give a program which is very similar to example 6, but is not locally determined. Let R be the logic program with the following set of clauses. The parameter i ranges over ω in all six groups of clauses.

1. $a_{3i} \leftarrow \neg a_{3i+1}, \neg a_{3i+2}$
2. $a_{3i+1} \leftarrow \neg a_{3i}, \neg a_{3i+2}$
3. $a_{3i+2} \leftarrow \neg a_{3i}, \neg a_{3i+1}$
4. $a_{3i} \leftarrow a_{3i+3}$
5. $a_{3i+1} \leftarrow a_{3i+4}$
6. $a_{3i+2} \leftarrow a_{3i+5}$

Just as in example 6, the Herbrand base of R is $\{a_0, a_1, \ldots\}$ and it easy see that P has exactly 3 stable models, namely, $S_0 = \{a_{3i} : i \in \omega\}$, $S_1 = \{a_{3i+1} : i \in \omega\}$ and $S_2 = \{a_{3i+2} : i \in \omega\}$. In this case, R has no levels. Again it is easy to see that the clauses of the form (1) and (2) ensure that $3n$ and $3n+1$ are not levels of P. However in this case, $3n+2$ is also not a level of P. That is consider $T = \{a_{3n}, a_{3n+1}\}$. It is easy to see that there is no minimal proof scheme ψ of P_{3n+2} such that $supp(\psi) \cap T = \emptyset$ and the conclusion of ψ is a_{3n+2}. However the following is a minimal proof ψ' of P with conclusion a_{3n+2} such that $supp(\psi') \cap T = \emptyset$.

$\langle\langle a_{3n+5}, a_{3n+5} \leftarrow \neg a_{3n+3}, \neg a_{3n+4}, \{a_{3n+3}, a_{3n+4}\}\rangle\langle a_{3n+2}, a_{3n+2} \leftarrow a_{3n+5}\rangle, \{a_{3n+3}, a_{3n+4}\}\rangle$.

Example 8. Suppose that we are given a set $L = \{l_0 < l_1 < \ldots\}$. Then we can construct a program P such that $H_P = \{a_0, a_1, \ldots\}$ and $lev(P) = L$ as follows. Let $l_{-1} = -1$. Then for each $n \geq 0$, we add the clause

$$a_{l_n} \leftarrow$$

to P if $l_n - l_{n-1} = 1$. Otherwise we add the following clauses to P

$$a_{l_{n-1}+k} \leftarrow \neg a_{l_{n-1}+1}, \ldots, \neg a_{l_{n-1}+k-1}, \neg a_{l_{n-1}+k+1}, \ldots, \neg a_{l_{n-1}+(l_n - l_{n-1})}$$

for all $k = 1, \ldots, l_n - l_{n-1}$. $\triangle$

Definition 5. *Suppose that P is a recursive logic program. Then we say that P is* **effectively locally determined** *if P is locally determined and there is a recursive function f such that for all i, $f(i) \geq i$ and $f(i)$ is a level of P.*

In [21], Marek, Nerode, and Remmel showed that the problem of finding a stable model of a locally finite recursive logic program can be reduced to finding an infinite path through a finitely branching recursive tree and the problem of finding a stable model of a highly recursive logic program can be reduced to finding an infinite path through a highly recursive tree. A locally determined logic program is not always locally finite since it is possible that a given atom has infinitely many proof schemes which involves arbitrarily large atoms as in Example 6 above. Vice versa, it is possible to give examples of locally finite logic programs which is not locally determined. Nevertheless, we shall see that we get similar results to those of Marek, Nerode, and Remmel for locally determined and effectively locally determined recursive logic programs.

Theorem 4. *Let P be a recursive logic program.*

1. If P is locally determined, then there is a recursive finitely branching tree T and a one-to-one degree preserving correspondence between the set of stable models $\mathcal{S}(P)$ of P and $[T]$ and

2. If P is effectively locally determined, then there is a highly recursive finitely branching tree T and a one-to-one degree preserving correspondence between the set set of stable models $\mathcal{S}(P)$ of P and $[T]$.

A careful inspection of the tree T of Theorem 4 allows us to establish the following fact.

Corollary 1. *Suppose that P is a countable locally determined logic program such that there are infinitely many n such that P_n has a stable model E_n. Then P has a stable model.*

We also have the following.

Corollary 2. *Supose that P is infinite locally determined logic program, then P satisfies property (Comp).*

One can immediately apply a number of known results from the theory of recursively bounded Π_1^0 classes to derive corresponding results about the set of stable models of an effectively locally determined recursive logic program.

Corollary 3. *Suppose that P is an effectively locally determined recursive logic program which has at least one stable model. Then*

1. *P has a stable model whose Turing jump is recursive in $\mathbf{0}'$.*
2. *If P has no recursive stable model, then P has $2^{\aleph_0}$ stable models.*
3. *If P has only finitely many stable models, then each of these stable models is recursive.*
4. *There is a stable model E of P in an r.e. degree.*
5. *There exist stable models E_1 and E_2 of P such that any function, recursive in both E_1 and E_2, is recursive.*
6. *If P has no recursive stable model, then there is a nonzero r.e. degree $\mathbf{a}$ such that P has no stable model recursive in $\mathbf{a}$.*

A similar corollary holds for locally determined recursive logic programs where the statements in Corollary 3 are replaced by versions which are relativized to a $\mathbf{0}'$ oracle.

7 FC-Normal Logic Programs

In this section with the notion of FC-normal logic programs as defined by Marek, Nerode, and Remmel [22]. These programs always have a stable model. In fact, they have an even stronger property that property (*Comp*). Namely, an FC-normal program P has the property that if P' is a finite subprogram of P which has a stable model E, then P has a stable model S which extends E.

Recall that $Horn(P)$ is the set of Horn clauses of a logic program P. We let $T_{Horn(P)}$ denote the one-step provability operator associated with $Horn(P)$, see [1]. That is, if $Q \subseteq H_P$, then $T_{Horn(P)}(Q)$ equals

$$\{p \in H_P : (\exists C = p \leftarrow q_1, \dots q_m \in Horn(P))\ (q_1, \dots, q_m \in Q)\}.$$

We call a family of subsets of H_P, Con, a **consistency property** over P if it satisfies the following conditions:

1. $\emptyset \in Con$.
2. If $A \subseteq B$ and $B \in Con$, then $A \in Con$.
3. Con is closed under directed unions.
4. If $A \in Con$ then $A \cup T_{Horn(P)}(A) \in Con$.

Conditions (1)-(3) are Scott's conditions for information systems. Condition (4) connects "consistent" sets of atoms to the Horn part of the program; if A is consistent then adding atoms provable from A preserves "consistency". The following fact is easy to prove.

Proposition 7. *If Con is a consistency property with respect to P and $A \in Con$, then $T_{Horn(P)} \Uparrow \omega(A) \in Con$.*

Here, for a Horn program Q, $T_Q \Uparrow \omega(A)$ is the cumulative fixpoint of T_Q over A. Proposition 7 says that our condition (4) in the definition of consistency property implies that the cumulative closure of a "consistent" set of atoms under $T_{H(P)}$ is still "consistent".

Given a consistency property, we define the concept of an **FC-normal** program with respect to that property. Here FC stands for "Forward Chaining".

Definition 6. (a) *Let P be a program, let Con be a consistency property with respect to P. Call P* **FC-normal with respect to** *Con if for every clause $C = p \leftarrow q_1, \ldots, q_n, \neg r_1, \ldots, \neg r_m$ such that $C \in ground(P) - ground(Horn(P))$ and every consistent fixpoint A of $T_{Horn(P)}$ such that $q_1, \ldots, q_n \in A$ and $p, r_1, \ldots, r_m \notin A$ we have*

(1) *$A \cup \{p\} \in Con$ and*
(2) *$A \cup \{p, r_i\} \notin Con$ for all $1 \leq i \leq m$.*
(b) *P is called* **FC-normal** *if there exists a consistency property Con such that P is FC-normal with respect to Con.*

Example 9. Let the Herbrand base consist of atoms a, b, c, d, e, f. Let the consistency property be defined by the following condition:
$A \notin Con$ if and only if either $\{c, d\} \subseteq A$ or $\{e, f\} \subseteq A$.

Now let us consider the following program.

(1) $a \leftarrow$
(2) $b \leftarrow c$
(3) $c \leftarrow b$
(4) $c \leftarrow a, \neg d$
(5) $e \leftarrow c, \neg f$

It is not difficult to check that this program is FC-normal with respect to the consistency property described above. Moreover, one can easily check that P possesses a unique stable model $M = \{a, b, c, e\}$.
If we add to this program the clause $f \leftarrow c, \neg e$, the resulting program is still FC-normal but now there are two stable models, $M_1 = \{a, b, c, e\}$ and $M_2 = \{a, b, c, f\}$. △

Marek, Nerode, and Remmel [22] showed that FC-normal normal programs have many of the properties that are possessed by normal default theories.

Theorem 5. *If P is an FC-normal program, then P possesses a stable model.*

Theorem 6. *If P is an FC-normal program with respect to the consistency property Con and $I \in Con$, then P possesses a stable model I' such that $I \subseteq I'$.*

Marek, Nerode, and Remmel proved Theorem 5 and 6 via a generally forward chaining algorithm which can be applied to FC-normal programs of any cardinality. Since in our case, we are dealing with only recursive and hence countable programs, we shall give only the countable version of their forward chaining construction. That is, let us suppose we fix some well-ordering $\prec$ of $ground(P) - ground(H(P))$ of order type ω. Thus, the well-ordering $\prec$ determines some listing of the clauses of $ground(P) - ground(H(P))$, $\{c_n : n \in \omega\}$. Their forward chaining construction then defines an increasing sequence of sets $\{T_n^{\prec}\}_{n\in\omega}$ in stages.

The countable forward chaining construction of $T^{\prec} = \bigcup_{n\in\omega} T_n^{\prec}$.

<u>Stage 0</u>. Let $T_0^{\prec} = T_{Horn(P)} \Uparrow \omega(\emptyset)$.

<u>Stage $n+1$</u>. Let $\ell(n+1)$ be the least $s \in \omega$ such that
$c_s = \varphi \leftarrow \alpha_1, \ldots, \alpha_k, \neg\beta_1, \ldots, \neg\beta_m$ where $\alpha_1, \ldots, \alpha_k \in T_n^{\prec}$ and
$\beta_1, \ldots, \beta_m, \varphi \notin T_n^{\prec}$. If there is no such $\ell(n+1)$, let $T_{n+1}^{\prec} = T_n^{\prec}$. Otherwise let

$$T_{n+1}^{\prec} = T_{Horn(P)} \Uparrow \omega(T_n^{\prec} \cup \{p_{\ell(n+1)}\})$$

where $p_{\ell(n+1)}$ is the head of $c_{\ell(n+1)}$.

Example 10. If we consider the final extended program of Example 9, it is easy to check that any ordering $\prec_1$ in which the clause $C_1 = e \leftarrow c, \neg f$ precedes the clause $C_2 = f \leftarrow c, \neg e$ will have $T^{\prec_1} = M_1$ while any ordering $\prec_2$ in which C_2 precedes C_1 will have $T^{\prec_2} = M_2$. △

This given, Marek, Nerode, and Remmel proved the following results.

Theorem 7. *If P is a countable FC-normal program and $\prec$ is* any *well-ordering of $ground(P) - ground(H(P))$ of order type ω, then :*

(1) *$T^{\prec}$ is a stable model of P where $T^{\prec}$ is constructed via the countable forward chaining algorithm.*
(2) *(completeness of the construction). Every stable model model of P is of the form $T^{\prec}$ for a suitably chosen ordering $\prec$ of $ground(P) - ground(H(P))$ of order type ω where $T^{\prec}$ is constructed via the countable forward chaining algorithm.*

Theorem 8. *If P is an FC-normal logic program with respect to Con, then every stable model M of P is in Con.*

Theorem 9. *Let P be an FC-normal logic program with respect to a consistency property Con. Then if E_1 and E_2 are two distinct stable models of P, then $E_1 \cup E_2 \notin Con$.*

Given a logic program P and a stable model M, we let $NG(M,P)$, the set of non-Horn generating clauses of P be equal to the set of all clauses $c = \varphi \leftarrow \alpha_1, \ldots, \alpha_k, \neg\beta_1, \ldots, \neg\beta_m$ in $ground(P)$ such that $\alpha_1, \ldots, \alpha_k \in M$ and $\beta_1, \ldots, \beta_m \notin M$.

FC-normal programs possess the following key "semi-monotonicity" property.

Theorem 10. *Let P_1, P_2 be two programs such that $P_1 \subseteq P_2$ but $H(P_1) = H(P_2)$. Let us assume, in addition, that both are FC-normal with respect to the same consistency property. Then for every stable model M_1 of P_1, there is a stable model M_2 of P_2 such that:*
(1) *$M_1 \subseteq M_2$ and*
(2) *$NG(M_1, P_1) \subseteq NG(M_2, P_2)$.*

As mentioned in the introduction, a recursive FC-normal logic program P is guaranteed to have at least one relatively well behaved stable model

Theorem 11. *Suppose that P is a recursive logic program and P is FC-normal. Then P has a stable model S such that S is r.e. in $\mathbf{0}'$ and hence $E \leq_T \mathbf{0}''$.*

We observe that this should be contrasted with the following result from [21]

Proposition 8. *There exists a recursive logic program P such that P possesses a stable model, but no stable model of P is hyperarithmetic.*

We note that Theorem 11 is in some sense the best possible. That is, results from [22] show that the following holds. Given sets $A, B \subseteq \omega$, let $A \oplus B = \{2x : x \in A\} \cup \{2x+1 : x \in B\}$.

Theorem 12. *Let A be any r.e. set and B be any set which is r.e. in A, i.e. $B = \{x : \phi_e^A(x) \downarrow\}$. Then there is a recursive FC-normal logic program P such P has a unique stable model S and $S \equiv_T A \oplus B$. In particular, if B is any set which is r.e. in $\mathbf{0}'$ and $B \geq_T \mathbf{0}'$, then there is an FC-normal recursive logic program P such that P has a unique stable model S and $S \equiv_T B$.*

However if either $Horn(P)$ or $P - Horn(P)$ is finite, then one can improve on Theorem 11. That is, the following was proved in [22].

Theorem 13. *Let P be a FC-normal recursive logic program such that $P - Horn(P)$ is finite, then every stable model of $\mathcal{S}$ is r.e..*

We say that a recursive logic program P is **monotonically decidable** if for any finite set $F \subseteq \mathcal{H}(P)$, $T_{Horn(P)} \Uparrow \omega(F)$ is recursive and there is a uniform effective procedure to go from a canonical index of a finite set F to a recursive index of the $T_{Horn(P)} \Uparrow \omega(F)$, i.e. if there is a recursive function f such that for all k, $\phi_{f(k)}$ is the characteristic function of $T_{Horn(P)} \Uparrow \omega(D_k)$. It is easy to see that if $Horn(P)$ is finite, then the recursive program P is automatically monotonically decidable.

Theorem 14. *Let P be a recursive logic program such that P is FC-normal and monotonically decidable, then P has an stable model which is r.e.*

We end this section by giving complexity results for finite FC-normal logic program where the forward chaining algorithm runs in polynomial time.

For complexity considerations, we shall assume that the elements of H_P are coded by strings over some finite alphabet Σ. Thus every $a \in H_P$ will have some length which we denote by $||a||$. Next, for a clause

$$r = c \leftarrow a_1, \ldots, a_n, \neg b_1, \ldots, \neg b_m,$$

we define $||r|| = (\sum_{i \leq n} ||a_i||) + (\sum_{i \leq m} ||b_j||) + ||c||$. Finally, for a set Q of clauses, we define

$$||Q|| = \sum_{r \in Q} ||r||.$$

Theorem 15. *Suppose P is a finite FC-normal logic program and $\prec$ is some well-ordering of $P - Horn(P)$. Then $E^{\prec}$ as constructed via our forward chaining algorithm can be computed in time*

$O(||Horn(P)|| \cdot ||P - Horn(P)|| + ||P - Horn(P)||^2)$.

We note that none of the theorems above make any explicit assumptions that the underlying consistency property of a recursive FC-normal logic P is in any way effective. Indeed none of the above results require that the underlying consistency property has any effective properties.

Finally Marek, Nerode, and Remmel [22] proved the following result about recursive FC-normal logic programs.

Theorem 16. *Let T be a recursive tree in $2^{<\omega}$ such that $[T] \neq \emptyset$. Then there is a FC-normal recursive logic program P such that there is an effective one-to-one degree preserving correspondence between $[T]$ and $\mathcal{S}(P)$.*

Reiter ([23]) proved that there is a recursive normal default theory with no recursive extension. Theorem 16, which was originally proved for nonmonotonic rule systems of which logic programs and default theories are special cases, contains Reiter's result as special case. In addition, it gives much finer information even for recursive normal default theories since the set of degrees of paths through highly recursive trees have been extensively studied. For example, our correspondence allows us to transfer results about the possible degrees of paths through highly recursive trees to results about the degrees of stable models of recursive FC-normal logic programs. A number of results of this kind has been proved in [22].

8 Conclusions

In this paper we established three classes $\mathcal{C}$ of logic programs such that $\mathcal{C}$ have compactness property. Two of these classes are based on enforcing tighter restriction on proof theoretic characterizations of stable models. The third one is based on the form of consistency property of Scott applied to logic programs. It is likely that there are other compact classes of programs and, in fact, one such class has been identified in [5]. The classes we found are quite rich, and it is likely that additional strong restrictions would have to be imposed on compact classes to make them a viable tool for applications. But what would be such applications? As is clear from the results stated in this paper, unlimited negation allows the programmer to write programs that have a unique stable model, but that model is so complex that it cannot be effectively queried. Even the stratified negation does not prevent stable models that are too complex for querying [2]. The logic programming community answered these challenges by limiting programs to DATALOG$^{\neg}$ programs (which essentially means eliminating untabled functions). In such setting the Herbrand base $B_{\mathcal{L}}$ is finite and so practical systms based on this mechanism can be built [24,4]. Yet, the current implementations of PROLOG mostly work and allow for use of negation. Thus there is hope that the quest for some reasonable and *practical* semantics of logic programs with negation is not entirely futile. Our paper is a small step in this direction.

Acknowledgements. V.W. Marek was partially supported by NSF grant no. IIS-0325063. J.B. Remmel was partially supported by NSF grant DMS-0400507.

References

1. K. Apt. Logic programming, In: J. van Leeuven, ed, Handbook of Theoretical Computer Science, pages 493–574, MIT Press, Cambridge, MA", 1990.
2. K. R. Apt, H. A. Blair, Arithmetical Classification of Perfect Models of Stratified Programs, Fundamenta Informaticae 13 (1990) 1-17.
3. K. Apt, H. Blair, and A. Walker. Towards a Theory of Declarative Knowledge. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–142, Los Altos, CA, 1987. Morgan Kaufmann.
4. Y. Babovich and V. Lifschitz. *Cmodels*, 2002. `http://www.cs.utexas.edu/users/tag/cmodels.html`.
5. P.A. Bonatti. Resolution for Skeptical Stable Model Semantics. *Journal of Automated Reasoning* 27:391–421, 2001.
6. D. Cenzer and J. B. Remmel, Index Sets for Π^0_1-classes, Annals of Pure and Applied Logic 93 (1998), 3-61.
7. D. Cenzer and J. B. Remmel, Π^0_1-classes in Mathematics, in: *Handbook of Recursive Mathematics: Volume 2*, eds. Yu. L. Ershov, S.S. Goncharov, A. Nerode, and J.B. Remmel, Studies in Logic and the Foundations of Mathematics, vol. 139, Elsevier, 1998, pp. 623-822.
8. D. Cenzer, J. B. Remmel, and A. K. C. S. Vanderbilt, Locally Determined Logic Programs, in: *Proceedings of the 6th international Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR99)*, Springer-Verlag, 1999, pp. 34-49.
9. P. Cholewiński, Stratified default theories, in *Proceedings of CSL'94*, Lecture Notes in Computer Science, vol. 933, Springer-Verlag, 1995.
10. P.M. Dung and K. Kanchanasut. A Fixpoint Approach to Declarative Semantics of Logic Programs, In: E.L. Lusk and R.A. Overbeek eds, *Logic Programming, Proceedings of North American Conference*, pages 604–625, 1989
11. A. Ferry, A topological characterization of the stable and minimal model classes of propositional logic programs, Ann. Math. Artificial Intelligence 15 (1995), 325-355.
12. M. Gelfond and V. Lifschitz. The Stable Semantics for Logic Programs. In *Proceedings of the 5th International Symposium on Logic Programming*, pages 1070–1080, Cambridge, MA., 1988. MIT Press.
13. K. Gödel. Über formal unentscheibare Sätze der Principia Mathemaica und verwandter Systeme I. *Monatshefte Math. Phys.* 38:173–198, 1931.
14. C.G. Jockusch and R.I. Soare. π^0_1 Classes and Degrees of Theories. *Transactions of American Mathematical Society*, 173:33–56, 1972.

15. V. Lifschitz and H. Turner, Splitting a logic program, in: *Proceedings of the Eleventh International Conference on Logic Programming*, ed. P. Van Hentenryck, 1994, pp. 23–37.
16. J. Lloyd, *Foundations of Logic Programming*, Springer-Verlag, 1989.
17. W. Marek, A. Nerode, and J.B. Remmel. Nonmonotonic Rule Systems I. *Annals of Mathematics and Artificial Intelligence*, 1:241–273, 1990.
18. W. Marek, A. Nerode, and J.B. Remmel. Nonmonotonic Rule Systems II. *Annals of Mathematics and Artificial Intelligence*, 5:229-264, 1992.
19. W. Marek, A. Nerode, and J.B. Remmel. A Context for Belief Revision: Normal Logic Programs (Extended Abstract) *Proceedings, Workshop on Defeasible Reasoning and Constraint Solving*, International Logic Programming Symposium, San Diego, CA., 1991.
20. W. Marek, A. Nerode, and J.B. Remmel. How Complicated is the Set of Stable Models of a Logic Program? *Annals of Pure and Applied Logic*, 56:119-136, 1992.
21. W. Marek, A. Nerode, and J. B. Remmel, The stable models of predicate logic programs. Journal of Logic Programming 21 (1994), 129-154.
22. W. Marek, A. Nerode, and J. B. Remmel, Context for belief revision: Forward chaining-normal nonmonotonic rule systems, Annals of Pure and Applied Logic 67 (1994), 269-324.
23. R. Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13:81–132, 1980.
24. P. Simons, I. Niemelä, and T. Soininen. Extending and implementing the stable model semantics. *Artificial Intelligence*, 138:181–234, 2002.
25. A. Van Gelder, K.A. Ross, and J.S. Schlipf. Unfounded sets and well-founded semantics for general logic programs. *Journal of the ACM*, 38:587, 1991.

Uniform Circuits, & Boolean Proof Nets

Virgile Mogbil[1,⋆] and Vincent Rahli[2,⋆⋆]

[1] LIPN – UMR7030, Université Paris 13 – CNRS, France
[2] ULTRA, Heriot-Watt University, Scotland

Abstract. The relationship between Boolean proof nets of multiplicative linear logic (*APN*) and Boolean circuits has been studied [Ter04] in a non-uniform setting. We refine this results by taking care of uniformity: the relationship can be expressed in term of the (Turing) polynomial hierarchy. We give a proofs-as-programs correspondence between proof nets and deterministic as well as non-deterministic Boolean circuits with a uniform depth-preserving simulation of each other. The Boolean proof nets class $m\&BN(poly)$ is built on multiplicative and additive linear logic with a polynomial amount of additive connectives as the non-deterministic circuit class $NNC(poly)$ is with non-deterministic variables. We obtain uniform-$APN = NC$ and $m\&BN(poly) = NNC(poly) = NP$.

1 Introduction

Linear Logic (LL) is a refinement of classical and intuitionist logic [Gir87]. The conjunction/disjunction are split into the multiplicative (M) and additive (A) connectives. The exponentials give a logical status to the structural rules of classical and intuitionist sequent calculus. The study of LL revealed the *proof nets* [Gir87, DR89]: a parallel syntax for logical proofs where some inessential sequential information are removed. In this way the global and sequential sequent calculus cut-elimination becomes local and parallel in the proof nets. The well known Curry-Howard isomorphism is a correspondence between proofs and programs which associates cut-elimination in proofs and execution in programs. We study its extension to models of parallel computation using proof nets.

Boolean circuits [Vol99, BS90] are a standard model of parallel computation as Turing machine are a model of sequential computation. Several important complexity classes are defined in terms of Boolean circuits, including *NC*. *NC* can be thought of as the problems being efficiently solved on a parallel computer just as the class *P* can be thought of as the tractable problems. Because a circuit has a fixed input size, Boolean circuits are so-called non-uniform models of computation: inputs of different lengths are processed by different circuits. A *uniformity* condition is often imposed on circuit families so that each circuit can be computed by some resource-bounded Turing machine. For instance *NC* is defined to be the set of Boolean functions that can be decided by uniform Boolean

⋆ Work partially supported by projects GEOCAL (ACI) and NO-CoST (ANR).
⋆⋆ Work partially supported by projects GEOCAL (ACI) and NO-CoST (ANR).

circuits of polynomial size in the length of the inputs and polylogarithmic depth. The depth is the time on a parallel computer where the size is the number of processors.

K. Terui (NII, Japan) introduced a proof-as-programs correspondence between a multiplicative Boolean proof nets class (APN) and Boolean circuits such that cut-elimination corresponds to evaluation [Ter04]. Defining a parallel cut-elimination in proof nets, the Terui's main result is $APN =$ non-uniform *NC*. This is the first time that a logical depth is taken into account: it makes possible to achieve a seed up over sequential computation. Without restricting the depth in *NC* or non-uniform *NC*, one obtains respectively *P* or *P/poly*. *P/poly* is the complexity class of languages recognized by a polynomial-time Turing machine with a polynomial-bounded advice function. So K. Terui gives a corollary: polynomially-sized multiplicative Boolean proof nets class is equivalent to *P/poly*. Our paper is firstly motivated by an algorithmic point of view, and therefore an important issue is the uniformity of circuits, because only a uniform circuit family $(C_n)_{n\in\mathbb{N}}$ can be regarded as an implementation of an algorithm. Indeed a description of the circuit C_n for inputs of size n can be obtained easily when the value of n is known: we need an efficient algorithm to built C_n given n, where different notions of efficient give rise to different notions of uniformity [Ruz81, All89, BIS90]. We introduce a suitable notion of uniformity for proof nets and adapt the proofs between proof nets and Boolean circuits to satisfy the uniformity condition. The method gives the uniform counter part of the Terui's results i.e. a proof nets characterisation of both *NC* and *P*.

Note that *P/poly* is not generally considered a practical class for computing. Indeed it contains every undecidable unary language, none of which can be solved in general by real computers. However *P/poly* is an important theoretical class because of the following fact: if $NP \subseteq P/poly$ then the polynomial hierarchy collapses to $\Sigma_2 P$ (i.e. *NP* with *NP* Oracle), and if *NP* is not a subset of *P/poly* then $P \neq NP$ [KL80, CK06]. Our work is motivated by such theoretical point of view and the logical study of complexity classes. We study a non-deterministic extension of the parallel Curry-Howard isomorphism in a uniform setting always by giving a uniform depth-preserving simulation of each classes. On one hand there are several characterizations of non-determinism in circuits [Ven92, Wol94]. We use *NNC*(*poly*) a class equivalent to *NP*, which is defined in the same way as *NC* but using at most a polynomial amount of non-deterministic variables. On the other hand, as suggested by K. Terui we enrich proof nets with additive connectives, because additive allow us to incorporate non-determinism. An encoding of a *co-NP* problem in the intuitionist fragment of MALL [MT03] illustrates why additives could be used. Contrary to the multiplicative case, the proof nets with additives have never been convincing: in the original syntax [Gir87], the additive connective & is associated to a box and the cut-elimination does not satisfy the Church-Rosser property. Nevertheless if a such proof net does not contain the connective & in its conclusions then it has a unique normal form [Tor03]. Our encoding and the defined Boolean proof nets satisfy this property, so we restrict our attention to this setting. Strong normalization and confluence of the additive proof nets have been studied in various directions [Tor03], in the polarized fragment of LL [LdF04] or recently with a set of linkings on a formula [HvG03, HvG05]. We have not yet fully explored this last approach which seems to give another kind of speed-up.

After some background on Boolean circuits in section 2, we present the Terui's approach extended to the uniform case in section 3. The circuit gates are unbounded like $(\wedge^n)_{n\in\mathbb{N}}$ and $(\vee^n)_{n\in\mathbb{N}}$. In particular the $stCONN_2$ gates which test the reachability between two nodes of the undirected graph given in input, are used to simulate the cut-elimination of proof nets. In the first subsection the definitions concerning MLL_u (a n-ary variant of MLL) and Boolean type are presented as in [Ter04]. We define the uniformity for Boolean proof net families: the class mBN denotes just uniform-APN. Then, we improve the Terui's results with two theorems (Thm.2, Thm.5): the translation and simulation between NC and mBN are done in logspace. Section 4 is devoted to the uniform non-deterministic Boolean proof nets called $m\&BN(poly)$. We define the proof nets for $\mathrm{M}_u\mathrm{ALL}$ which is the fragment MLL_u extended with additive connectives. We remind the standard definitions of the additives connectives: the slices and the cut-elimination. We easily obtain a parallel reduction theorem because of the particular setting. We introduce an extended version of the Boolean type for the non-deterministic variables. In the last section we establish the translation and the simulation theorems which imply $NNC(poly) = m\&BN(poly) = NP$.

2 Background

– Boolean Circuits –

Let F_n denote the set of all Boolean functions $f : \{0,1\}^n \to \{0,1\}$ for some $n \in \mathbb{N}$. A *basis* is a finite set consisting of Boolean functions or sequences of Boolean functions $(f_i)_{i\in\mathbb{N}}$ where $f_i \in F_i$. The standard basis are $\mathcal{B}_0 = \{\neg, \wedge, \vee\}$ and $\mathcal{B}_1 = \{\neg, (\wedge^n)_{n\in\mathbb{N}}, (\vee^n)_{n\in\mathbb{N}}\}$.

A *Boolean circuit* over a basis $\mathcal{B}$ is a directed acyclic graph with $n+1$ sources or inputs (vertices with no in-going edges), one sink or output (a vertex with no out-going edges) and all nodes in $\mathcal{B}$. Sources are labelled by literals from $\{x_1, \ldots, x_n\} \cup \{1\}$ and nodes of in-degree k are labelled by one of the k-ary Boolean functions of $\mathcal{B}$. A Boolean circuit (a circuit for short) computes a function in F_n in a natural way. Nodes are called *gates*, and in-degree and out-degree are called *fan-in* and *fan-out* respectively. The circuits over basis without infinite families of Boolean functions (as $\mathcal{B}_0$), are called *bounded fan-in* circuits. The other circuits are called *unbounded fan-in* circuits.

We say that a *family of circuits* $C = (C_n)_{n\in\mathbb{N}}$ computes a function $f : \{0,1\}^* \to \{0,1\}$ (or recognizes a language $L_C \in \{0,1\}^*$) if for every n the circuit C_n computes the restriction of f to F_n. I.e. $\forall x \in \{0,1\}^*, C_{|x|}(x) = f(x)$.

Let C be a circuit, the *size* denoted $size(C)$ is the number of gates of C. The *depth* denoted $d(C)$ is the length of a longest directed path.

A *non-deterministic* Boolean circuit C with m non-deterministic variables is a circuit with $n+m+1$ sources labelled by $\{x_1, \ldots, x_n\} \cup \{y_1, \ldots, y_m\} \cup \{1\}$.

It computes a function $f \in F_n$ as follows: for $x \in \{0,1\}^n$, $f(x) = 1$ iff there exist a setting of the non-deterministic variables $\{y_1, \ldots, y_m\}$ which makes the circuit output 1. We denote $C(x, y)$ the same circuit as C without distinction between non-deterministic variables and deterministic gates, $x \in L_C$ the language recognized by C, if $\exists w \in \{0,1\}^m$ a witness s.t. $C(x, w) = 1$. When needed we abusively denote $C(x)$ and $C(x, y)$ the distinct circuits.

– Circuit Uniformity –

As briefly presented in the introduction, there are different notions of circuit uniformity [Vol99]. In order to obtain a class containing P, it is necessary to impose a "P-uniform" condition on circuit families. That is, the description of the n^{th} circuit can be provided by a deterministic Turing machine operating in polynomial time [All89]. But P-uniformity is too weak to define subclasses of P; this leads one to consider L-uniformity [Ruz81]: the description is computable in logspace. It is the same when we want to consider the subclass of NC with $O(log\ n)$ depth (called NC^1): $DLOGTIME$-uniformity requires a somewhat more careful definition. Informally, there is a deterministic linear time in $O(log\ s(C_n))$ Turing machine that, given n and a name of a gate g can determine all the wanted information about gate g (like sort, predecessors, ...) belonging to the circuit C_n of size s. Unfortunately all these notions are more and more restrictive.

In our work we focus on a uniform approach of the well established relationship between non-uniform NC and the multiplicative Boolean proof net class APN. For the sake of simplicity we consider the L-uniformity: it is sufficient to investigate all classes containing L. Actually, only NC^1 and constant depth classes of circuits and proof nets need a uniformity notion stronger than the L-uniformity. Moreover the NC class remains the same if stronger notions of uniformity than L-uniformity are used [Ruz81].

The *direct connection language* of a family $C = (C_n)_{n\in\mathbb{N}}$ over basis $\mathcal{B}$, denoted $L_{DC}(C)$, is the set of tuples $\langle y, g, p, b\rangle$, where for $y = 1^n$, we have: g is the number of a gate v in C_n, and $p \in \{0,1\}^*$ is a binary word such that

- if $p = \varepsilon$ then b is the number of the function from $\mathcal{B}$ labeling v,
- if $p = bin(k)$ then b is the number of the k^{th} predecessor gate to v.

A circuit family $(C_n)_{n\in\mathbb{N}}$ is *L-uniform* if its direct connection language can be recognized in logspace by a deterministic Turing machine. Without precisions, we use in the rest of this paper the term *uniform* as a shorthand for L-uniform.

– Circuit Classes –

The *classes NC^i* and *AC^i* for $i \geqslant 0$ are the functions computable by uniform families of polynomial size, $O(log^i n)$ depth circuits over $\mathcal{B}_0$ and $\mathcal{B}_1$ respectively. $AC^i(stCONN_2)$ correspond to AC^i over $\mathcal{B}_1 \cup \{stCONN_2\}$. We denote NC the uniform circuit families which have polynomial size and polylogarithmic depth, i.e. $NC = \cup_{i \geqslant 0} NC^i$. We define AC in the same way. L, NL and P are in the time-space hierarchy of the Turing machines. The well known hierarchy is:

$$AC^0 \subsetneq NC^1 \subseteq L \subseteq NL \subseteq AC^1 \subseteq NC^2 \subseteq AC^2 \subseteq \cdots \subseteq NC \subseteq P$$
$$\forall i \in \mathbb{N}, AC^i \subseteq AC^i(stCONN_2) \subseteq AC^{i+1}$$

The *class* $NNC^i(f(n))$ is the class of languages accepted by L-uniform-NC^i circuit families with at most $O(f(n))$ non-deterministic variables, where n is the length of the input. We define $NAC^i(f(n))$ in the same way, but using AC^i. We abusively denote $NNC^i(poly)$ when $f(n)$ is a polynomial function. If $f(n) = log\, n$ then the amount of non-deterministic variables can be described by a polynomial number of NC gates [Wol94]:

$$NNC^i(log\ n) = NC^i, \text{ and then } NNC(log\ n) = \cup_{i \geqslant 0} NNC^i(log\ n) = NC.$$

So we don't consider these classes but we investigate the following classes [Wol94]:

$$NNC(poly) = \cup_{j \geqslant 0} NNC(n^j) = NP \text{ and } \forall i \in \mathbb{N}^*, NNC^i(poly) = NP.$$
$$NAC(poly) = \cup_{j \geqslant 0} NAC(n^j) = NP \text{ and } \forall i \in \mathbb{N}, NAC^i(poly) = NP.$$

3 Uniform Boolean Proof Nets

– MLL$_u$ and Boolean Type –

We recall in this section some basic definitions gave by Terui in [Ter04]. The *formulas* of MLL$_u$ are built on literals by n-ary versions of multiplicative conjunction and disjunction, for every $n \geqslant 2$. The negation of a non-literal formula is defined by de Morgan's duality (reversing the order of subformulas).

A *sequent* of MLL$_u$ is of the form $\vdash \Gamma$, where Γ is a multiset of formulas. The rules of MLL$_u$ are given in Fig.1(a) with the convention $\overrightarrow{A} \equiv A_1, \ldots, A_n$ and $\overleftarrow{A} \equiv A_n, \ldots, A_1$.

(a)
$$\frac{}{\vdash A, A^\perp}\ (axiom) \qquad \frac{\vdash \Gamma, C \quad \vdash \Delta, C^\perp}{\vdash \Gamma, \Delta}\ (cut)$$
$$\frac{\vdash \Gamma_1, A_1 \quad \ldots \quad \vdash \Gamma_n, A_n}{\vdash \Gamma_1, \ldots, \Gamma_n, \otimes^n(\overrightarrow{A})}\ \otimes^n \qquad \frac{\vdash \Gamma, A_n, \ldots, A_1}{\vdash \Gamma, \invamp^n(\overleftarrow{A})}\ \invamp^n$$

(b)
$$\frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \& B}\ \&$$
$$\frac{\vdash \Gamma, A_i}{\vdash \Gamma, A_1 \oplus A_2}\ \oplus_i {}_{i=1,2}$$

Fig. 1. (a) MLL$_u$ (a+b) M$_u$ALL

The corresponding *links* (Fig.2(a)) are of three *sorts* called: axiom-link, $\otimes^n$-link and $\invamp^n$-link. Each link has several ports. The ports numbered 0 are called the

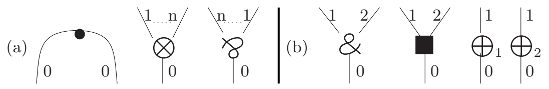


Fig. 2. (a) Multiplicative and (b) additive links

principal ports, while others are called auxiliary ports. By convention a principal port is always written below the link whereas an auxiliary port is above it.

A *pseudo net* is a triple $\langle L, \sigma, \sim\rangle$ s.t. L is a finite set of links, $\sigma : L \to \{\bullet\} \cup \{\otimes^n, ⅋^n\}_{n \geqslant 1}$ and $\sim$ is a symmetric relation on $(L, \mathbb{N})^2$.

A link p with $\sigma(p) = \bullet$ ($\otimes^n$, $⅋^n$, resp.) stands for an axiom-link ($\otimes^n$-link, $⅋^n$-link, resp.). When $(p, n) \sim (q, m)$, we say that there is an edge between (p, n) and (q, m), where (p, n) stands for the port number n of link p. A *cut* in a pseudo net is an unordered pair of link p, q s.t. $(p, 0) \sim (q, 0)$: we call it an *ax*-cut when either p or q is an axiom-link, otherwise we call it a *m*-cut.

A *proof net* of type $\vdash \Gamma$ is a pseudo net P inferred by a sequent calculus proof of $\vdash \Gamma'$ where Γ is a decoration of Γ' i.e. formulas of the form $p : A$ (see Fig. 5 for an example) (a proof net of type $\vdash p : A$ is simply called a proof net of type A) . The pseudo nets inferred in Fig.3(a) are respectively: $tensor_q^{p_1,\dots,p_n}(P_1, \dots, P_n)$, $par_q^{p_n,\dots,p_1}(P)$ and ax_p, $cut^{p,q}(P, Q)$.

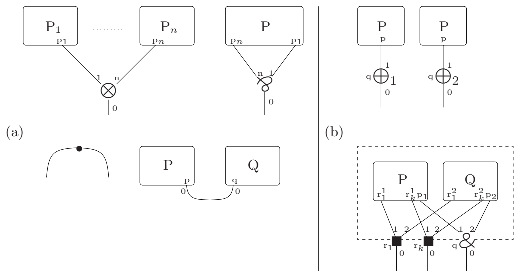


Fig. 3. (a) Multiplicative and (b) additive proof net constructors

The *size* $|P|$ is the number of links in P. The *depth* $d(A)$ of a formula A is given by $d(\alpha) = d(\alpha^\perp) = 1$ and $d(\otimes^n(\overrightarrow{A})) = d(⅋^n(\overleftarrow{A})) = max(d(A_1), \dots, d(A_n)) + 1$, with $\overrightarrow{A} \equiv A_1, \dots, A_n$. Given a derivation π of $\vdash \Gamma$ inferring P, its depth $d(\pi)$ is the maximal depth of cut formulas in it. The *depth* $d(P)$ of a proof net P is defined to be $min\{d(\pi) | \pi$ is a derivation of $\vdash \Gamma$ inferring P for some $\Gamma\}$.

Boolean values are represented with the type $\mathbf{B} = ⅋^3(\alpha^\perp, \alpha^\perp, \otimes^2(\alpha, \alpha))$.

There are exactly two cut-free proof nets of this type (*true* and *false* resp.) Fig.4(a): $b_1 \equiv par_s^{p,q,r}(tensor_r^{p,q}(ax_p, ax_q))$, $b_0 \equiv par_s^{q,p,r}(tensor_r^{p,q}(ax_p, ax_q))$.

A *Boolean proof net* with n inputs $\overrightarrow{p} \equiv p_1, \dots, p_n$ and one output is a proof net $P(\overrightarrow{p})$ of type: $\vdash p_1 : \mathbf{B}^\perp[A_1], \dots, p_n : \mathbf{B}^\perp[A_n], q : \otimes^{m+1}(\mathbf{B}, \overrightarrow{C})$, for some $\overrightarrow{A} \equiv A_1, \dots, A_n$ and $\overrightarrow{C} \equiv C_1, \dots, C_m$ (garbage due to the multiplicative framework) where $\mathbf{B}[A]$ denote the formula $\mathbf{B}$ where all occurrences of α are substituted by A. Given $\overrightarrow{b} \equiv b_{i_1}, \dots, b_{i_n}$, $P(\overrightarrow{b})$, of type $\otimes^{m+1}(\mathbf{B}, \overrightarrow{C})$, denotes the proof net obtained by connecting, $\forall j \in \{1, \dots, n\}$ $(i_j \in \{0,1\})$, b_{i_j} to p_j by a cut. $P(\overrightarrow{b})$ reduces to a cut-free proof net of the shape $tensor(b_i, \overrightarrow{Q})$, $i \in \{0,1\}$: we say that $P(\overrightarrow{b})$ evaluates to b_i. Let $w \equiv i_1 \dots i_n \in \{0,1\}^n$. $P(\overrightarrow{p})$ represents a function $f : \{0,1\}^n \to \{0,1\}$ if $P(b_{i_1}, \dots, b_{i_n})$ evaluates to $b_{f(w)}$. Thus, the language accepted by $P(\overrightarrow{p})$ is $f^{-1}(1)$.

APN^i for $i \in \mathbb{N}$, is the class of languages recognized by non-uniform Boolean proof net families of polynomial size and $O(log^i\ n)$-depth. APN$= \bigcup_{i\in\mathbb{N}} APN^i$.

– Uniform Proof Nets –

In the framework of polynomial size proof nets, we consider an extended description of a proof net P which is equivalent to the triple defining a pseudo net. We call it $Conf(P)$, the configuration of P.

Let $P = \{P_n\}_{n\in\mathbb{N}}$ be a family of Boolean proof nets. Links are identified by binary words. For all $n \in \mathbb{N}$, we fix the inputs $p_1, \dots, p_n$ of P_n to be identified by $0, \dots, bin(n-1)$ respectively, and the output to be identified by $bin(n)$ (where bin is the function which associates to a number in decimal base its value in binary base). *Conf(P)* denotes the set of tuples in $\{1\}^* \times (W \setminus \{\epsilon\}) \times W^2 \times \{0,1\}$ (W is the set of binary words), s.t. for $y = 1^n$:

– in $\langle y, u, \epsilon, \epsilon, 1\rangle$, u identifies a P_n link.
– in $\langle y, u, s, \epsilon, 1\rangle$, s is the sort's identifier of the link identified by u.
– in $\langle y, u, v, bin(i), 1\rangle$, u identifies a link connected by its principal port (or one of its two principal ports in the case of an axiom) to the port i of the link identified by v.
– in $\langle y, u, a, b, 0\rangle$, u, a, b are the same informations as above but are not concerned with P_n (i.e. the edges which does not appear in P_n).

Remark that if $P_n \in P$ then $Conf(P_n)$ is the set of tuples that belong to $Conf(P)$ s.t. the first word is of size n. A proof net family $\{P_n\}_{n\in\mathbb{N}}$ of polynomial size s is *L-uniform* (resp. *P-uniform*) iff there is a function which computes $Conf(P_n)$ from 1^n in space $O(log\ s)$ (resp. in time $s^{O(1)}$), for all $n \in \mathbb{N}$. For all $i \in \mathbb{N}$, uniform APN^i is denoted mBN^i.

– Uniform Terui's Translation of NC –

The *conditional* (if-then-else, Fig.4(b)) is the base of the Terui's gates translations: given two proof nets P_1 and P_2 of types $\vdash \mathit{\Gamma}, p_1 : A$ and $\vdash \mathit{\Delta}, p_2 : A$ resp., one can build a proof net $cond_r^{p_1,p_2}[P_1, P_2](q)$ of type $\vdash \mathit{\Gamma}, \mathit{\Delta}, q : \mathbf{B}[A]^\perp, r : A \otimes A$. Given a cut between b_i and q we have with the convention that the first component is considered as the output, the rest being the garbage:

$$cond_r^{p_1,p_2}[P_1, P_2](b_1) \to^* tensor_r^{p_1,p_2}(P_1, P_2),$$
$$cond_r^{p_1,p_2}[P_1, P_2](b_0) \to^* tensor_r^{p_2,p_1}(P_2, P_1).$$

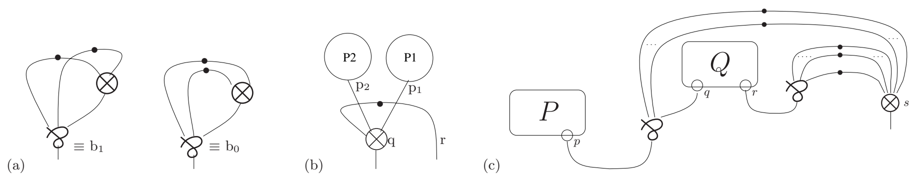


Fig. 4. (a) The Boolean b_1 and b_0 (b) The conditional (c) The composition

Disjunction, *conjunction* and *duplication* are based on the conditional: let $n \geqslant 2$ be an integer and $C \equiv \otimes^n(\mathbf{B}[A_1], \ldots, \mathbf{B}[A_n])$,

$$\begin{aligned}
&\mathit{or}(p_1,p_2) \equiv \mathit{cond}[b_1, ax_{p_1}](p_2) \text{ of type } \vdash p_1 : \mathbf{B}^\perp, p_2 : \mathbf{B}[\mathbf{B}]^\perp, q : \mathbf{B} \otimes \mathbf{B},\\
&\mathit{and}(p_1,p_2) \equiv \mathit{cond}[ax_{p_1}, b_0](p_2) \text{ of type } \vdash p_1 : \mathbf{B}^\perp, p_2 : \mathbf{B}[\mathbf{B}]^\perp, q : \mathbf{B} \otimes \mathbf{B},\\
&\mathit{copy}^n(p) \equiv \mathit{cond}[\mathit{tensor}(\overrightarrow{b_1}), \mathit{tensor}(\overrightarrow{b_0})](p) \text{ of type } \vdash p : \mathbf{B}^\perp[C], q : C \otimes C.
\end{aligned}$$

The *composition* (Fig.4(c)) of two translated circuits is defined as follows: let $\Gamma \equiv p'_1 : A'_1, \ldots, p'_n : A'_n$ and $\Delta \equiv q'_1 : B'_1, \ldots, q'_n : B'_m$, let $P(\overrightarrow{p'})$ and $Q(\overrightarrow{q'})$ be proof nets of type $\vdash \Gamma, p : \otimes^{1+m}(\mathbf{B}, \overrightarrow{C})$ and $\vdash q : \mathbf{B}^\perp[A], \Delta, r : \otimes^{1+m'}(\mathbf{B}, \overrightarrow{D})$, respectively. Then:

$$\mathit{comp}_s^{p,q,r}[P,Q](\overrightarrow{p'}, \overrightarrow{q'}) \text{ is of type } \vdash \Gamma[A], \Delta, s : \otimes^{1+m'+m}(\mathbf{B}, \overrightarrow{D}, \overrightarrow{C[A]}).$$

With this composition one can construct n-ary versions of conjunction and disjunction. The translation of a Boolean circuit follows from composition of gate translations and duplication for fan-out management:

Theorem 1 ([Ter04]). *For every unbounded fan-in Boolean circuit C of size s and depth d over the basis $\mathscr{B}_1(stCONN_2)$, there is a Boolean proof net of size $O(s^5)$ and depth $O(d)$, which accepts the same set as C does.*

Lemma 1. *Let Σ be a finite alphabet. If $g : \Sigma^* \to \Sigma^*$ and $f : \Sigma^* \to \Sigma^*$ are computable in logspace and the f output is polynomial in its input, then $g \circ f$ is computable in logspace.*

Theorem 2. *Let $i \in \mathbb{N}$. The Terui translation of a Boolean circuit family in $AC^i(stCONN_2)$, by a Boolean proof net family in mBN^i is logspace.*

Remark 1. In order to simplify this result we consider a translation of a circuit where the garbage of each translated gate is propagated directly to the output of the proof net. This version is equivalent to the one presented by Terui but in which some reductions have been performed.

Proof. Let $C = \{C_n\}_{n \in \mathbb{N}}$ be a L-uniform Boolean circuit family in AC^i $(stCONN_2)$. By uniformity, there is a logspace function f s.t. for every $n \in \mathbb{N}$, $f(1^n) = L_{DC}(C_n)$. Let $P = \{P_n\}_{n \in \mathbb{N}}$ be the Boolean proof net family obtained by translation of C. We show that there is a logspace function f' built from f s.t. for every $n \in \mathbb{N}$, $f'(1^n) = \mathit{Conf}(P_n)$. In order to do that, we use a function $f_{d \to c}$, logspace in the inputs of f, which associates $\mathit{Conf}(P_n)$ to $L_{DC}(C_n)$ for all $n \in \mathbb{N}$. Let $f_{d\to c} = f_3 \circ f_2 \circ f_1$ where we call module the composition of the translated gate and its translated fan-out, and ($k, j \in \{1, \dots, size(C_n)\}$):

- f_1 copies out its inputs adding to tuples which indicate that a gate v is the k^{th} predecessor of a gate u, that gate u is the j^{th} successor of gate v.
  Then f_1 computes for every gate its module composed with a pseudo net allowing garbage propagation, adding the identifier of the translated gate, and for each output, an identifier differentiating it from others.
- f_2 copies out and completes the information given by the modules created by f_1 with the help of the information added to $L_{DC}(C_n)$. That is, it adds information about edges between an output of a module and an input of another module, with the help of information added about translated gates and edges between gates.
- f_3 organizes tuples and deletes useless information.

$L_{DC}(C_n)$ like $Conf(P_n)$ has a polynomial size. An identifier of a gate or a link is logspace. At each step, these functions memorize a constant number of identifiers, so f_1, f_2 and f_3 are logspace. Then $f_{d\to c}$ is logspace. By lemma 1, $f' = f_{d\to c} \circ f$ is logspace in the inputs of f. □

– Parallel Cut-Elimination –

The elimination of a cut between two axioms cannot always be performed in parallel. So K. Terui considers another reduction step named *tightening reduction* which reduces a maximal sequence of cut axioms to an axiom. Such maximal sequence (ax-sequence) is defined as a set of axioms, each linked with another in this set s.t. there are not other axioms that verify this property.

If Q is obtained from P by elimination of all ax-cuts simultaneously (m-cuts, ax-sequences, resp.) then we write $P \Rightarrow_{ax} Q$ ($P \Rightarrow_m Q$, $P \Rightarrow_t Q$, resp.). We write $P \Rightarrow Q$ if $P \Rightarrow_{ax} Q$ or $P \Rightarrow_m Q$ or $P \Rightarrow_t Q$.

Theorem 3 (parallel cut-elimination, [Ter04]). *There is a sequence of parallel reductions $P \Rightarrow P_1 \cdots \Rightarrow P_k$, s.t. P_k is cut-free and $k \leqslant 3 \times d(P)$.*

– Uniform Terui's Simulation in NC –

We consider proof nets with links belonging to a fixed set L_0 with the convention $⅋^n$-link is of sort $⅋$, and a $\otimes^n$-link is of sort $\otimes$. A *configuration* Θ consists of the following Boolean values : $alive(p)$, $sort(p, s)$, $edge(p, 0, q, i)$, for every $p, q \in L_0$, $s \in \{\bullet, \otimes, ⅋\}$ and $i < |L_0|$.

A link $p \in L_0$ is said to be alive in Θ if $alive(p) = 1$. Given a proof net $P = \langle L, \sigma, \sim\rangle$, we write $\Theta \in Conf(P)$ if for every $p \in L_0$, $alive(p) = 1 \iff p \in L$ and for every alive links p, q in Θ, the following holds : $sort(p, s) = 1 \iff \sigma(p)$ is of sort s and $edge(p, 0, q, i) = 1 \iff (p, 0) \sim (q, i)$.

Lemma 2 ([Ter04]). *There is an unbounded fan-in Boolean circuit C of size $O(|P_0|^3)$ and constant depth with stCONN$_2$ gates s.t. whenever $\Theta \in Conf(P)$ is given as input of C and $P \Rightarrow P$', it outputs a configuration $\Theta' \in Conf(P')$.*

Theorem 4 ([Ter04]). *For every Boolean proof net P of size s and depth d, there is a Boolean circuit C over the basis $\mathscr{B}_1(stCONN_2)$ of size $O(s^4)$ and depth $O(d)$ which accepts the same set as P does.*

Theorem 5. *Let $i \in \mathbb{N}$. The Terui simulation of a Boolean proof net family in mBN^i by a Boolean circuit family in $AC^i(stCONN_2)$, is logspace.*

Proof. Let $P = \{P_n\}_{n\in\mathbb{N}}$ be a L-uniform Boolean proof net family in APN^i. By uniformity there is a logspace function f s.t. for all $n \in \mathbb{N}$, $f(1^n) = Conf(P_n)$. Let $C = \{C_n\}_{n\in\mathbb{N}}$ be the Boolean circuit family obtained by simulation of P. We show that there is a logspace function f' built from f s.t. for every $n \in \mathbb{N}$, $f'(1^n) = L_{DC}(C_n)$. In order to do that, we use a function $f_{c\to d}$, logspace in the inputs of f, which associates $L_{DC}(C_n)$ to $Conf(P_n)$ for all $n \in \mathbb{N}$. The simulation introduced by Terui is divided in three steps: creation of initial configuration, simulation of parallel cuts elimination and check of the the last configuration.

- $f_{c\to d}$ builds the part of the first configuration which represents the inputs of the proof net using the inputs of the Boolean circuit. It builds the part which represents the proof net using its input. Only a constant number of identifiers of links are memorized.
- $f_{c\to d}$ builds the circuit which simulates parallel cuts elimination. This part is only dependent of the size and depth of the proof net and not of its structure.
- $f_{c\to d}$ builds the circuit which check the result contained in the last configuration. This building is only dependent of the size of the proof net.

$Conf(P_n)$ like $L_{DC}(C_n)$ has a polynomial size. An identifier of a gate or a link is logspace. At each step, $f_{c\to d}$ memorizes a constant number of identifiers, so it is logspace. By lemma 1, $f' = f_{c\to d} \circ f$ is logspace in the inputs of f. □

Corollary 1. *$mBN = NC$ and P is the class of those languages for which there is a uniform polynomial size family of (multiplicative) Boolean proof nets.*

4 Non-deterministic Boolean Proof Nets $m\&BN()$

In this section we introduce the unbounded fan-in version of multiplicative linear logic (MLL$_u$) extended with the binary additive connectives. We denote it M$_u$ALL. The *formulas* of M$_u$ALL are built from literals by MLL$_u$ connectives and by binary additive conjunction $\&$ and additive disjunction $\oplus$. The *sequent calculus rules* are given in Fig.1(a+b).

The added *links* are called $\&$-link, $\oplus_1$-link and $\oplus_2$-link (Fig.2(b)). The *inference rules* for the new links follow the sequent calculus rules as expected. The resulting proof nets are depicted in Fig. 3(b). In a derivation, the two premises of a $\&$-rule, with the same context Γ, infer two proof nets called *components*. Each identical conclusion coming from the two components are merged with a binary co-additive-links (denoted ■-links or coad-links). So a $\&$-link arrives with an *additive box* bounding a (possibly empty) set of coad-links and the two components. This is described in Fig.3(b) with the associated ports. As previously the cuts are just edges between principal ports.

– *Slices and Additive Cut-Elimination* –

A *slice* of a proof net P is a proof net $sl(P)$ which may contain some unary $\&_1$ and $\&_2$ links: for every $\&$-link one of the premises is chosen. Then the non-corresponding component, the additive box and the coad-links are erased. In a slice, the additive cut-elimination between a $\oplus_i$-link and a $\&_j$-link, for $i, j \in \{1, 2\}$, is simply to check the *consistency*: if $i = j$ then the additive links are removed propagating the cut on the unique premises.

We define as usual [Gir87] the *additive cut-elimination*, denoted $\rightarrow_a$. The cut-elimination between a $\&$-link and a $\oplus_i$-link ($i \in \{1, 2\}$) is the choice of the i^{th} $\&$-link's premise and the same erasing as a slice. A cut-elimination with a coad-link is more complicated: the cut proof net is swallowed and duplicated in the additive box. The duplicated conclusions are merged with coad-links.

The cut-elimination with a coad-link is not confluent. A standard example is the proof net of type $\vdash A\&A, A^{\perp}$, cut to the proof net of type $\vdash A^{\perp}\&A^{\perp}, A$. This produces two possible proof nets of type $\vdash A\&A, A^{\perp}\&A^{\perp}$ where the $\&$-link of the additive box is one of the formulas, the other being a coad-link. Nevertheless the proof nets are stable by cut-elimination [Dan90, Tor03]. Roughly speaking if the conclusions of the proof nets are $\&$-free then the cut-elimination is confluent.

– *Parallel Reduction Theorem* –

It is difficult to have an additive cut-elimination from a parallel point of view (denoted $\Rightarrow_a$). We need a strategy e.g. as the parallel tightening reduction, one could want to reduce first the $\&$-links in parallel. But it is difficult to bound the number of parallel reduction steps because of the coad-links (it depends on maximal nesting of the additive boxes it contains). In the non-deterministic Boolean proof nets framework, it is somewhat different as explained after in lemma 3: proof nets are always cut against an additive witness s.t. this cut is eliminated equivalently to a slice choice. So we start the parallel reductions by choosing a slice, then additive cut-elimination becomes trivial and confluent: $\Rightarrow_a$ is only the disjoint consistency check done in parallel. We write $\Rightarrow$ if one of the parallel reduction occurs (i.e. $\Rightarrow_t, \Rightarrow_{ax}, \Rightarrow_m$ or $\Rightarrow_a$). Because in a slice each sequence of all distinct parallel reductions strictly decrease the depth, we have:

Theorem 6 (parallel cut-elimination). *For every consistent slice sl, there is a sequence of parallel reductions $sl(P) \Rightarrow P_1 \Rightarrow \cdots \Rightarrow P_k$ s.t. P_k is cut-free and $k \leqslant 4 \times d(P) + 1$.*

– *Extended Boolean Type for Non-Deterministic Variables* –

We define a non-deterministic Boolean type as $\mathbf{B_e} = (\mathbf{B} \otimes \alpha^{\perp}), (\alpha \& \alpha)$. There are four proof nets of this (yet strange) type. Two of them have the same behavior as the (deterministic) Boolean type $\mathbf{B}$. The two others can be arbitrarily used for our purpose: let the proof net depicted in Fig.5 be our choice of one of them. With this proof net we are able to internally choose between b_0 or b_1 using a slice or equivalently by cut elimination with a witness as explained in the beginning of this section. Without loss of generality, for an arbitrary C, we allow an input

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash B}\quad \dfrac{}{\vdash \alpha,\alpha^{\perp}}\,(ax)}{\vdash B\otimes\alpha^{\perp},\alpha}\,(\otimes)\qquad \dfrac{\dfrac{\vdots}{\vdash B}\quad \dfrac{}{\vdash \alpha,\alpha^{\perp}}\,(ax)}{\vdash B\otimes\alpha^{\perp},\alpha}\,(\otimes)}{\vdash B\otimes\alpha^{\perp},\alpha\&\alpha}\,(\&) \qquad \mapsto$$

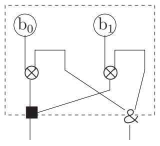


Fig. 5. From a sequent calculus proof of $\mathbf{B_e}$ to the chosen proof net

type to be of the form $\mathbf{B_e}[C] = \mathbf{B_e}[\mathbf{B}[C]/\mathbf{B}, id/\alpha] = (\mathbf{B}[C] \otimes id^{\perp})\invamp(id\&id)$ where $id = \alpha\invamp\alpha^{\perp}$ is a technical trick to simplify reductions in the translation and in the simulation.

– m&BN() Description and Hierarchy –

A *non-deterministic Boolean proof net* with n inputs and $O(f(n))$ additive links is a proof net $P(\overrightarrow{b})$ of type (for some $\overrightarrow{D} = D_1, \ldots, D_k$ and $m = f(n)$):

$$\vdash p_1 : \mathbf{B}^{\perp}[A_1], \ldots, p_n : \mathbf{B}^{\perp}[A_n], q : \otimes^{1+k+m}(\mathbf{B}, \overrightarrow{D}, \overrightarrow{id}), r : \invamp^{m}(\overrightarrow{id^{\perp}\&id^{\perp}}).$$

Clearly, for w a proof net of type $\otimes^{m}(\overrightarrow{id \oplus id})$ and $\overrightarrow{b} \equiv b_{i_1}, \ldots, b_{i_n}$ ($\forall j \in \{1, \ldots, n\}$, $i_j \in \{0,1\}$), the proof net $cut(P(\overrightarrow{b}), w)$ reduces to a cut-free proof net of type $\otimes^{1+k+m}(b_i, \overrightarrow{D}, \overrightarrow{id})$ ($i \in \{0,1\}$): we say that $P(\overrightarrow{b})$ evaluates to b_i if there exists such a w. The language accepted by a non-deterministic Boolean proof net is defined in the same way as for Boolean proof nets in section 3.

Similarly to the Terui *APN* hierarchy and to the *NNC*() hierarchy, we define the *m&BN*() hierarchy: a uniform family $(P_n)_{n\in\mathbb{N}}$ of non-deterministic Boolean proof nets *accepts* a language $X \subseteq \{0,1\}^*$ if P_n is n-ary and accepts $X \cap \{0,1\}^n$ for every $n \geqslant 1$. A language $X \subseteq \{0,1\}^*$ belongs to the class $m\&BN^i(poly)$ iff X is accepted by a uniform polynomial size, log^i-depth family of non-deterministic Boolean proof nets with a polynomial number of additive links.

5 *NNC*(*poly*) = *m&BN*(*poly*)

In order to obtain the equality between *NNC*(*poly*) and *m&BN*(*poly*), we use the class *NAC*(*poly*) whose relations with *NNC*(*poly*) were described in section 2.

– Translation of NAC(poly) –

Let $C(x)$ be a circuit of input length n belonging to a family in $NAC^i(poly)$ for an arbitrary $i \in \mathbb{N}$. By definition $C(x,y)$ and $C(x)$ have the same dimensions and $m = \mid y \mid = n^{O(1)}$. Thus $C(x,y) \in AC^i$ w.r.t. the size of x, if we just omit the distinction between non-deterministic variables and deterministic variables. As in [Ter04] (see section 3) let P be the translation of $C(x,y)$. For each y we cut the builtproof net P with exactly one representation of $\mathbf{B}_e$. We manage the garbage as it can be seen in Fig.6 to obtain the

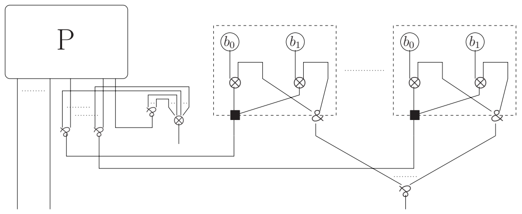
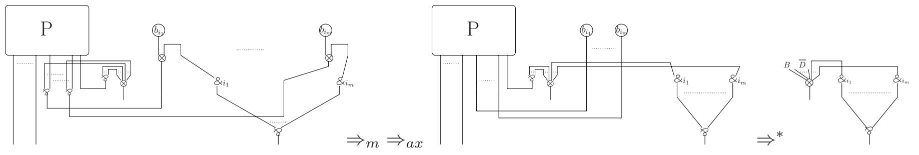


Fig. 6. Translation

proof net T. Such *translated* proof net is in $m\&BN^i(poly)$. Let sl be a slice of T. We have the following parallel reduction from $sl(T)$:



Another way could be to choose a witness w, i.e a proof net of type $\otimes^m(\overrightarrow{id \oplus id})$, and then to reduce the cut between T and w. By only one additive parallel reduction we obtain confluently a multiplicative proof net: it corresponds to $C(-, w)$. From the translation it immediately holds that to have a slice or a witness is equivalent in this setting, as is stated in this lemma:

Lemma 3. *Let $C \in NAC(poly)$ be a circuit family and let $P = (P_n)_{n\in\mathbb{N}}$ be the uniform non-deterministic Boolean proof net family obtained by the translation of C. Let $C(x, y) \in AC$ be the deterministic circuit family corresponding to C s.t. $|y|= m$. We have $x \in L_C$, the language recognized by C*
$\iff \exists w \in \{0, 1\}^m$ *a witness s.t.* $C_n(x, w) = 1$
$\iff \exists sl$ *a slice of* P_n *s.t.* $sl(P_n(x)) \Rightarrow^* \otimes^{1+n+m}(b_1, \overrightarrow{D}, \overrightarrow{\alpha}), \parr_{1\leqslant j\leqslant m}(\&_{i_j}(\alpha))$
$\iff \exists w$ *a witness of type* $\otimes^m(\overrightarrow{id \oplus id})$ *s.t.* $cut(P_n(x), w) \Rightarrow^* \otimes^{1+l+m}(b_1, \overrightarrow{D}, \overrightarrow{id})$
$\iff x \in L_P$, *which is the language recognized by P*

Theorem 7 (translation). *Let $i \in \mathbb{N}$. For every Boolean circuit $C \in NAC^i(poly)$ of size s and depth d, there is a Boolean proof net in $m\&BN^i(poly)$ of size $O(s^5)$ and depth $O(d)$ which accepts the same set/language as C does.*

Proof. Let $C \in NAC^i(poly)$ be a circuit of input length n. Let $s = size(C)$ and $d = d(C)$. Let $m =|y|= n^{O(1)} \leqslant s$ be the amount of non-deterministic variables. Following Terui's theorem, every gate of fan-in f and fan-out o can be encoded by a proof net of size $O(f^4 + o) \leqslant O(s^4)$ and of constant depth. The depth increase is linear in d. Because $C(x, y)$ and C have the same dimensions, we obtain the same result as Terui. The last construction due to the non-deterministic variables translation doesn't change the proof net dimensions: we add $O(m + s)$ links and a constant to the depth. □

– Cut-Elimination Simulation –

Let $P \in m\&BN^i(poly)$ be a proof net of size s and depth d. Let $NAC^i(poly)$ $(stCONN)$ corresponds to $NAC^i(poly)$ over $\mathcal{B}_1 \cup \{stCONN\}$. In order to have a corresponding non-deterministic circuit $C \in NAC^i(poly)(stCONN)$, we represent P by a set of Boolean values: the configurations that we extend to take care of additive sorts and a $box(p, q)$ relation to describe that p is associated to the additive box of the $\&$-link q. After that in the same way as [Ter04] (see section 3), the cut-elimination in P is simulated by the construction of layers of a circuit for each parallel cut-elimination step s.t. the evaluation of the constructed circuit C simulates the cut-elimination procedure. For simplicity we describe it following the parallel reductions as in lemma 3 with the help of non-deterministic variables, one for each link of the proof net:

– The circuit simulates the choice of a slice (its edges) s.t. every distinct choice can be done in parallel, depending on the non-deterministic variables:
  - selection of one premise for each $\&$-links, with the help of a constant depth circuit of size $O(s^3)$,
  - selection of one premise for each coad-links depending on the selected premise of the associated $\&$-links, with the help of a constant depth circuit of size $O(s^4)$. Moreover we "erase" the coad-links themselves: by updating the *alive* value of a coad-link and the *edge* values concerning its incident edges and by linking the premise and the conclusion of a coad-link, with the help of a constant depth circuit of size $O(s^5)$,
  - selection of one component associated to each $\&$-links using a *stCONN* gate, with the help of a constant depth circuit of size $O(s^4)$.
– The cut between a witness (depending on the non-deterministic variables) and the modified slice is reduced in a known result due to our choice of the *id* type. This is done by a constant depth circuit of size $O(s^5)$.
– Finally as many time as $4 \times d(P)+1$ we simulate one parallel reduction of the modified slice ($\Rightarrow$, in the same way as in [Ter04] with the help of a constant depth circuit of size $O(s^3)$ that simulate $\Rightarrow_a$). We check consistency at each step with the help of a constant depth circuit of size $O(s^3)$.

The most essential cost is due to the erasing of the coad-links and the elimination of the cut between a witness and a slice of a Boolean proof net. Each small circuit used here can be found in the appendix with precise depth and size.

Theorem 8 (simulation). *Let $i \in \mathbb{N}$. For every Boolean proof net $P \in m\&BN^i(poly)$ of size s and depth d, there is a Boolean circuit in $NAC^i(poly)$ $(stCONN)$ of size $O(s^5)$ and depth $O(d)$ which accepts the same set/language as P does.*

Since the considered non-deterministic circuit class collapse we can replace $NAC^i(poly)(stCONN)$ by $NAC^i(poly)$ in the previous theorem.

6 Conclusion

Focusing on uniformity, we strengthened the connection between proof nets and deterministic Boolean circuits by giving a uniform depth-preserving simulation of each other. This kind of Curry-Howard isomorphism for models of parallel computation as been extended to the non deterministic case. Because the uniformity arguments apply to the translation and simulation theorems, we have:

Theorem 9. *$\forall i \in \mathbb{N}$, $m\&BN^i(poly) = NAC^i(poly) = NP$.*

Corollary 2. *$m\&BN(poly) = NNC(poly) = NP$.*

The substitution on the non-deterministic Boolean type B_e deals with id to simplify the general case which can also be treated without difficulties. The most simple form of non-deterministic Boolean is $1 \oplus 1$ i.e. just replace $id = \alpha \parr \alpha^\perp$ by 1: the semantic is clearly the same but the fragment of linear logic considered is extended to neutrals. Even without this, the Boolean proof nets of type $\vdash \mathbf{B}^\perp, \ldots, \mathbf{B}^\perp, \mathbf{B}\& \ldots \&\mathbf{B}$ are in a way canonical (up to substitutions) but the simulation is more complicated. A more general work can be done using the additive fragment expressiveness fully: a garbage is no more needed but the cut-elimination is no more confluent. Other approaches are interesting like the additives à la Hughes and van Glabbeek [HvG03, HvG05]: we believe to realize a stronger seed-up.

References

[All89] Eric W. Allender. P-uniform circuit complexity. *Journal of the Association for Computing Machinery*, 36(4):912–928, 1989.

[BIS90] David A. Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1. *J. of Comput. and System Science*, 41(3):274–306, 1990.

[BS90] R. B. Boppana and M. Sipser. *The complexity of finite functions*. MIT Press, 1990.

[CK06] S. Cook and J. Krajicek. Consequences of the provability of NP⊆P/poly, 2006.

[Dan90] V. Danos. *La logique linéaire appliquée à l'étude de divers processus de normalisation (et principalement du λ-calcul)*. PhD thesis, Univ. Paris VII, 1990.

[DR89] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28(3):181–203, 1989.

[Gir87] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50(1):1–102, 1987.

[HvG03] D. J. D. Hughes and R. J. van Glabbeek. Proof nets for unit-free multiplicative-additive linear logic. In *Proc. IEEE Logic in Comput. Sci.*, 2003.

[HvG05] D. J. D. Hughes and R. J. van Glabbeek. Proof nets for unit-free multiplicative-additive linear logic. *ACM Trans. on Comput. Logic*, 2005.

[KL80] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proc. 12th ACM Symp. on Theory of Computing*, pages 302–309, 1980.

[LdF04] O. Laurent and L. Tortora de Falco. Slicing polarized additive normalization. *Linear Logic in Computer Science*, 2004.

[MT03] H. Mairson and K. Terui. On the computational complexity of cut-elimination in linear logic. *Theoretical Computer Science*, 2841:23–36, 2003.
[Ruz81] W. Ruzzo. On uniform circuit complexity. *J. of Computer and System Science*, 21:365–383, 1981.
[Ter04] K. Terui. Proof nets and boolean circuits. In *Proc. IEEE Logic in Comput. Sci.*, pages 182–191, 2004.
[Tor03] L. Tortora De Falco. Additives of linear logic and normalization - part i: a (restricted) church-rosser property. *T.C.S.*, 294(3):489–524, 2003.
[Ven92] H. Venkateswaran. Circuit definitions of nondeterministic complexity classes. *Siam J. Comput.*, 21(4):655–670, 1992.
[Vol99] H. Vollmer. *Introduction to Circuit Complexity – A Uniform Approach.* Texts in Theoretical Computer Science. Springer Verlag, 1999.
[Wol94] Marty J. Wolf. Nondeterministic circuits, space complexity and quasigroups. *Theoretical Computer Science*, 125(2):295–313, 1994.

A Some Corollaries

As a corollary for the theorem 7, we have:

Corollary 3. *For every Boolean circuit $C \in NNC^i(poly)$ of size s and depth d there is a Boolean proof net in $m\&BN(poly)$ of size $O(s^5)$ and depth $O(d)$ which accept the same set/language as C does.*

The most essential cost of the simulation of a Boolean proof net in $m\&BN^i(poly)$ ($i \in \mathbb{N}$) by a boolean circuit in $NNC^i(poly)$, is due to the translates of *stCONN* gates (a log-depth circuit of size $O(n^4)$ for a n-ary gate). Thus as a corollary for the theorem 8, we have:

Corollary 4. *For every Boolean proof net $P \in m\&BN^i(poly)$ ($i \in \mathbb{N}$) of size s and depth d, there is a Boolean circuit in $NNC^{i+2}(poly)$ of size $O(s^{12})$ and depth $O(d)$ which accepts the same set/language as P does.*

B Non-deterministic Simulation

In this section, we adopt these conventions:

- $alive(p)$ is write p
- $sort(p, s)$ is write $p(s)$
- $edge(p, 0, q, i)$ is write (p, q, i)

The circuit on Fig. 7, depicts the selection of one premise of a $\&$-links. The two ouputs are the new values for the *edge* values. This circuit is realized for all links p, q and r. For all links p and q, the new value of an $edge(p, 0, q, 1)$ or an $edge(p, 0, q, 2)$ value corresponds to the conjunction of its values at ouputs of these circuits. We obtain a circuit (named Sg) of size n^{3j} and constant depth, with n^j the number of links.

It is the same for the selection of one premise of a coad-links (Fig. 8), but it depend on the selected premise of the associated $\&$-links. The two ouputs are

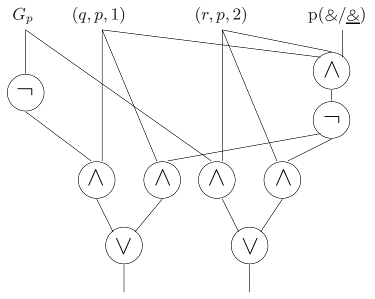


Fig. 7. Selection of one premise of a &-link

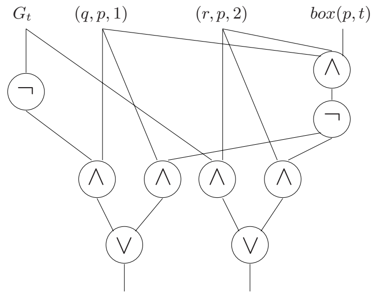


Fig. 8. Selection of one premise of a coad-link

the new values for the *edge* values. This circuit is realized for all links p, q, r and t. For all links p and q, the new value of an $edge(p, 0, q, 1)$ or an $edge(p, 0, q, 2)$ value corresponds to the conjunction of its values at ouputs of these circuits. We obtain a circuit of size n^{4j} and constant depth, with n^j the number of links.

The circuit on Fig. 9, depicts the erasing of information on unary coad-links. This circuit is realized for all links p, q, r, t and i. First, the new value of a *edge* value corresponds to the disjunction of its values at the first output ($edge(q, 0, t, i)$ on the Fig. 9) of these circuits. Second, the new value of an *edge*, a *box*, or an *alive* value corresponds to the conjunction of its values at the five

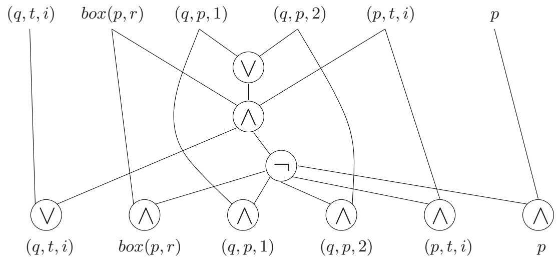


Fig. 9. Erasing of a unary coad-link

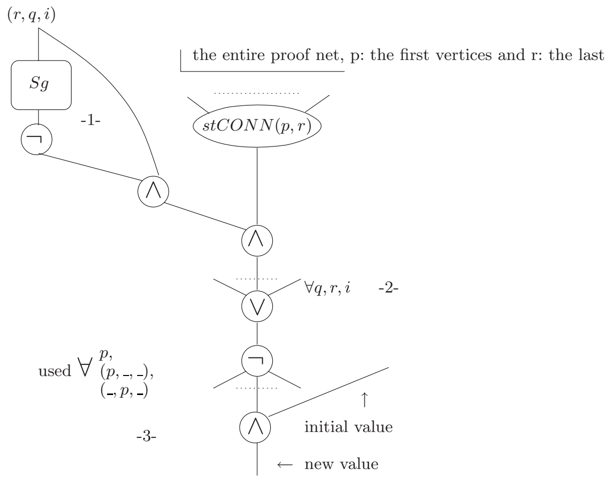

(r, q, i)
Sg
-1-
the entire proof net, p: the first vertices and r: the last
$stCONN(p, r)$
$\forall q, r, i$
-2-
used $\forall$ p, $(p, \_, \_)$, $(\_, p, \_)$
initial value
-3-
new value

Fig. 10. Erasing of a useless component in a slice

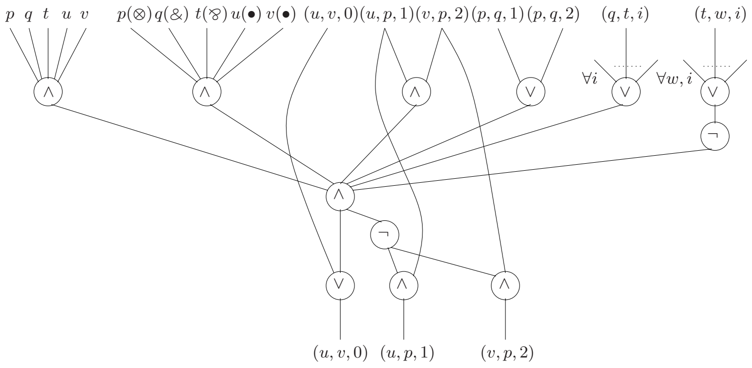

p q t u v
$p(\otimes)$ $q(\&)$ $t(⅋)$ $u(\bullet)$ $v(\bullet)$
$(u, v, 0)$ $(u, p, 1)$ $(v, p, 2)$ $(p, q, 1)$ $(p, q, 2)$
(q, t, i)
(t, w, i)
$\forall i$
$\forall w, i$
$(u, v, 0)$ $(u, p, 1)$ $(v, p, 2)$

Fig. 11. Fast cut-elimination of a witness cut to a boolean proof net

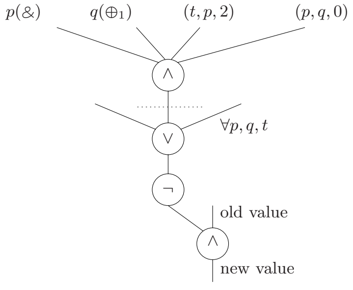


Fig. 12. Check of consistency property

last outputs of these circuits and at the output just created. We obtain a circuit of size n^{5j} and constant depth, with n^j the number of links.

The circuit on Fig. 10, depicts the erasing of the useless components of each additive box. The part above the mark -2- is realized for all links q, r and i. This circuit above the mark -3- is realized for all links p. The new value of an *edge* or an *alive* value corresponds to the conjunction of its values at ouputs of these circuits. We obtain a circuit of size n^{4j} and constant depth, with n^j the number of links.

Since the cut between a witness and the modified slice is reduced in a known result, the following circuit on Fig. 11 depicts elimination of this cut. This circuit is realized for all links p, q, t, y and v. For all links p and q, the new value of an $edge(p, 0, q, 1)$ or an $edge(p, 0, q, 2)$ value corresponds to the conjunction of its values at ouputs of these circuits. For an $edge(p, 0, q, 0)$ value, we realize a disjunction. We obtain a circuit of size n^{5j} and constant depth, with n^j the number of links.

The circuit on Fig. 12 allows to check consistency of a slice. It is of size n^{3j} and constant depth, with n^j the number of links.

C Some Proofs

– Descriptions of Proof Nets –

The description of a Boolean proof net (the triples defining a pseudo net introduced in section 3.3) can expressed as following:

Definition 1 (triple(P)). *Let $P = \{P_n\}_{n\in\mathbb{N}}$ be a family of Boolean proof nets. Links are identified by binary numbers. For all $n \in \mathbb{N}$, we fix the inputs $p_1, \dots, p_n$ of P_n to be identified by $O, \dots, bin(n-1)$ respectively, and the output to be identified by $bin(n)$.* triple(P) *denotes the set of tuples in $\{1\}^* \times (W \setminus \{\varepsilon\}) \times W^2$ (where W is the set of binary words), s.t. for $y = 1^n$:*

- *$\langle y, u, \epsilon, \epsilon\rangle$, u identifies a link belonging to P_n.*
- *$\langle y, u, s, \epsilon\rangle$, s identifies the sort of the link identified by u.*
- *$\langle y, u, v, bin(i)\rangle$, u identifies a link connected by its principal port (or one of its two principal ports in the case of an axiom) to the port i of the link identified by v.*

Let P_n be a Boolean proof net belonging to the family $\{P_i\}_{i\in\mathbb{N}}$. We'll show that there exists a function f computable in space $O(log(n))$, which associates $Conf(P_n)$ to 1^n iff. there exists a function g computable in space $O(log(n))$, which associate $triple(P_n)$ to 1^n. In order to do that, we will use two function h and h', both logspace in n, which associate respectively $triple(P_n)$ to $Conf(P_n)$ and $Conf(P_n)$ to $triple(P_n)$.

For every tuple belonging to its input, h write this tuple, without its last word, on its ouput, if the last word is 1.

The function h' handles the links by decreasing order on their identifiers, after having found the biggest identifier amoung the tuples $\langle \_, \_, \epsilon, \epsilon\rangle$, t. For all identifier of link $i \in \{0, \ldots, t\}$, if the tuple $\langle \_, i, \epsilon, \epsilon \rangle$ is in the input of h', it copy out on its output with the fifth word 1, else with 0. It works similarly for the tuples whose the forth word is ϵ (or none of the four).

Functions h and h' memorize a constant number of tuples, of identifier of links and of counters of size logspace in n.

– Proof of Lemma 1 –

Proof. We don't want to memorize the output of f on a input w. We have to notice that at each step of the computation of g on $f(w)$, it's necessary to read only one bit of $f(w)$. Thus, it's sufficient to maintain a counter (on the binary base) of size $O(log(n))$ (where n is the size of w), because of the polynomial size of the output of f. Then, we simulate the computation of g, in space $O(log(n))$, on the input $f(w)$ without write it entirely. Each time we need to read a bit, we use a space $O(log(n))$ to find it. Hence, we use a space $O(log(n))$ to compute $g \circ f$. □

– Proof of Theorem 6 –

Proof. As in a multiplicative framework, only elimination of ax-cuts cannot be performed in parallel in a slice of a proof-net built with multiplicative and additive links. Hence, we'll show that the sequence $\Rightarrow_t; \Rightarrow_{ax}; \Rightarrow_a; \Rightarrow_m$ decrease the depth a slice of a proof-net. Let P be a proof-net and $sl(P)$ one of its consistent slice.

– Let Q such that $sl(P) \Rightarrow_t Q$. If the only one cuts of $sl(P)$ are some cuts of ax-sequences, they are all removed. Then, the depth of Q is 0 which if less or equal to the depth of $sl(P)$. Else, there exist some other cuts in $sl(P)$. Let π be any sequentialization of $sl(P)$). Let $\{A_1, \ldots, A_k\}$ (where $k \in \mathbb{N}$), the set of ax-sequences of $sl(P)$. For all $i \in \{1, \ldots, k\}$, the formulae (of depth p_i) associates to the ports of the axiom which substitute the ax-sequence A_i are the same than those of any axiom in this ax-sequence. Let $p = max(p_1, \ldots, p_k)$. After reduction of all the ax-sequences, the depths of the cut-formulae (which are not between two axioms) in π remain unchanged. If the maximal depth (q) of cuts (we define the depth of a cut as the depth of formulae associated to the ports of the links connected by this cut) in $sl(P)$, which are not cuts between two axioms, is less than p, then the depth of Q is $q < p = d(sl(P))$. Else, the depth of Q is the same than the one of $sl(P)$: $q \geq p$.
– Let R such that $Q \Rightarrow_{Ax} R$. If the only one cuts of Q are ax-cuts, they are all removed. Then, the depth of R is 0 which if less than the depth of Q (since these cuts are between two links whose one is not an axiom, see below). Else, there exist some other cuts in Q. Let π be any sequentialization of Q. Let $\{v_1, \ldots, v_k\}$ (where $k \in \mathbb{N}$), the set of axioms which are cut to other links than an axiom in Q. For all $i \in \{1, \ldots, k\}$, let v_i^1 and v_i^2 be the two principal ports of v_i. Let v_i cut by its port v_i^1 to the principal port u_i^0 of a link u_i (its only one principal port because u_i cannot be an axiom). Let w_i the link connected to v_i by v_i^2. In π, the formula associated to u_i^0 is the same

than the one associated to v_i^2, the dual of the one associated to v_i^1. So, they are the same depth p_i. Let $p = max(p_1, \ldots, p_k)$. The elimination of this cut remove, for all $i \in \{1, \ldots, k\}$, v_i and connect w_i to u_i. So, after elimination of all the ax-cuts, the depths of cut-formulae (those which are not ax-cuts) in π remain unchanged. If the maximal depth (q) of cuts in Q, which are not ax-cuts, is less than p, then the depth of R is $q < p = d(Q)$. Else, the depth of R is the same than the one of Q: $q \geq p$.

– R do not contain cuts involving axioms any more. The only one cuts of R are m-cuts or additive cuts.
  - Let S such that $R \Rightarrow_a S$. Let R contain m m-cuts and n additive cuts. Let π be any sequentialization of R. An additive cut between two additive links, in a consistent slice of a proof net, whose the formulae associated to their principal ports in π are $\&_i(A)$ and $\oplus_i(B, C)$ $(i \in \{1, 2\})$ is replaced by a cut between two links whose the formulae associated to the principal ports connected by the cut in π are A and B or A and C. The deepest formulae in π (p be the greatest depth) are formulae associate to m-cuts or formulae $F_1 \in \{\&_{i_1}(A_1), \oplus_1(B_1, C_1)\}, \ldots, F_k \in \{\&_{i_k}(A_k), \oplus_{i_k}(B_k, C_k)\}$, where $0 \leq k \leq n$ and $\forall j \in \{1, \ldots, k\}, i_j \in \{1, 2\}$. The set of deepest cut-formulae contain now the formulae A_i, B_i or A_i, C_i, for all $i \in \{1, \ldots, k\}$), of depth $p-1$, which are cut-formulae. In π, an additive cut of depth p generate cuts of depth at most $p-1$ (since $\&_i(A)$ or $\oplus_i(B, C)$ $(i \in \{1, 2\})$ are formulae of depth $d(A) + 1$). The other cuts, of depth $q < p$, generate cuts of depth at most $q - 1 < p - 1$.
  - Let T such that $S \Rightarrow_m T$. Let π be any sequentialization of S. By the previous item, the deepest cut-formulae are formulae in π, associate to m-cuts. A m-cut between two links whose the formulae associate to their principal ports in π are $\otimes(\overrightarrow{A})$ and $\parr(\overleftarrow{B})$, is replaced by n cuts between pairs of links whose formulae in π associate to ports connected by cuts are A_i and B_i, for all $i \in \{1, \ldots, n\}$ where $n \in \mathbb{N}$. The deepest cut-formulae in π (p be the greatest depth) are $F_1 \in \{\otimes^{k_1}(A_1^1, \ldots, A_1^{k_1}), \parr^{k_1}((A_1^{k_1})^\perp, \ldots, (A_1^1)^\perp)\}, \ldots, F_r \in \{\otimes^{k_k}(A_r^1, \ldots, A_r^{k_r}), \parr^{k_r}((A_r^{k_r})^\perp, \ldots, (A_r^1)^\perp)\}$, where $r \in \mathbb{N}$ and $\forall j \in \{1, \ldots, r\}, k_j \in \mathbb{N}$. Then, The deepest cut-formulae are some formulae $A_i^j, (A_i^j)^\perp$ where $i \in \{1, \ldots, r\}$ and $j \in \{1, \ldots, k_r\}$, of depth $p - 1$. In π, a cut of depth p generate some cuts of depth at most $p - 1$. Moreover, there exit at least one cut at this depth since $\otimes(\overrightarrow{A})$ or $\parr(\overleftarrow{B})$ $(n \in \mathbb{N})$ are formulae of depth $max(d(A_1), \ldots, d(A_n)) + 1$, so there exist a $j \in \{1, \ldots, n\}$ such that $A_j = max(d(A_1), \ldots, d(A_n))$. The other cuts of depth $q < p$ generate cuts of depth at most $q - 1 < p - 1$.

$\square$

Finite Automata Presentable Abelian Groups*

André Nies[1] and Pavel Semukhin[2]

[1] Department of Computer Science, University of Auckland, New Zealand
andre@cs.auckland.ac.nz

[2] Department of Computer Science, University of Auckland, New Zealand
pavel@cs.auckland.ac.nz

Abstract. We give new examples of FA presentable torsion-free abelian groups. Namely, for every $n \geqslant 2$, we construct a rank n indecomposable torsion-free abelian group which has an FA presentation. We also construct an FA presentation of the group $(\mathbb{Z}, +)^2$ in which every nontrivial cyclic subgroup is not FA recognizable.

1 Introduction

The study of finite-automata (FA) presentable, or automatic, structures began in the works by Hodgson [5,6] and then carried on in Khoussainov and Nerode [7]. A structure is called FA presented if its domain and atomic relations are recognized by finite automata operating synchronously on their input. So, these structures have finite presentations. The class of automatic structures is of special interest in the field of theoretical computer science. One of the reasons for this is that the first-order theory of an FA presentable structure is decidable, and in fact the model checking problem is decidable. For instance, Hodgson [5,6] was the first to use this property to give a new proof of the decidability of Presburger arithmetic $\mathrm{Th}(\mathbb{N}, +)$.

There is a complete description of FA presentable structures in the classes of Boolean algebras [9] and well-ordered sets [3], but for another classes of structures such as groups, rings, linear orders, etc., the situation is far from clear. For example, it is unknown whether the group of rationals under addition has an FA presentation. In [8] Khoussainov and Rubin posed the problem of characterizing automatic abelian groups (Problem 4).

In this paper we describe a new method for constructing FA presentable abelian groups and monoids using the notion of amalgamated product. We show that under certain conditions the amalgamated product of FA presentable groups or monoids is itself FA presentable. We then use this method to give new examples of FA presentable torsion-free abelian groups. The only known examples of such groups were $(\mathbb{Z}, +)$, $(R_p, +)$ (the group of rationals with denominators powers of p), and their finite direct products. Our examples are indecomposable and strongly indecomposable torsion-free abelian groups.

* This research was partially supported by the Marsden Fund of New Zealand, grant no. 03-UOA-130.

In the last section we will construct an FA presentation of the group $(\mathbb{Z},+)^2$ which is new in the sense that any nontrivial cyclic subgroup in that presentation is not FA recognizable.

We now give formal definitions that will be used in this paper.

Definition 1.1. *Let Σ be a finite alphabet, and $\bar{a} = (a_1, \ldots, a_k)$ be a tuple of words from Σ^*. A* **convolution** *of $\bar{a}$ is a word in alphabet $(\Sigma \cup \{\Box\})^k$ which is constructed by placing the words $a_1, \ldots, a_k$ one under another and adding a special symbol $\Box$ at the end of the words to get the same length. For example,*

$$\mathrm{Conv}(01, 1011, 100) = \begin{array}{cccc} 0 & 1 & \Box & \Box \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & \Box \end{array}$$

A **convolution of a relation** *$R \subseteq (\Sigma^*)^k$ is defined as $\mathrm{Conv}(R) = \{\mathrm{Conv}(\bar{a}) : \bar{a} \in R\}$.*

Definition 1.2. *A relation $R \subseteq (\Sigma^*)^k$ is* **FA recognizable***, or* **regular***, if $\mathrm{Conv}(R)$ is recognized by a finite automaton.*

Definition 1.3. *A structure $\mathcal{A} = (A; R_1, \ldots, R_n, f_1, \ldots, f_m)$ is* **FA presented** *if, for a finite alphabet Σ, $A \subseteq \Sigma^*$ is an FA recognizable set of word in Σ^*, and all the relations $R_1, \ldots, R_n$ together with the graphs of operations $f_1, \ldots, f_m$ are recognized by finite automata.*

A structure $\mathcal{A}$ is **FA presentable** *if it is isomorphic to an FA presented structure.*

In some cases to prove that a given structure is FA presentable we will not construct its automatic presentation explicitly. Instead we give its first-order interpretation in a structure already known to be FA presented. The description of this method together with formal definitions and proofs can be found in [2].

2 An FA Presentation of the Group R_p

Definition 2.1. *Let R_p be the subgroup of $(\mathbb{Q},+)$ consisting of elements of the form k/p^i.*

In the literature R_p is also denoted by $\mathbb{Q}^{(p)}$ (for example, see [4]) or $\mathbb{Z}[1/p]$. The next theorem shows that R_p is FA presentable, and we will use this particular presentation of R_p in section 5 to construct new examples of FA presentable abelian groups.

Theorem 2.1. *R_p is FA-presentable.*

First, we will construct an automatic presentation of R_p^+, the submonoid of R_p consisting of elements greater than or equal to 0. Later we describe how to obtain an FA presentation of the entire group R_p from the one for R_p^+.

The alphabet of the FA presentation for R_p^+ will be $\Sigma = \{\binom{n}{m} : n \in \{0,1\}$ and $m \in \{0, \ldots, p-1\}\}$. Every element $z \in R_p^+$ will be represented by two lines of digits,

n_1	n_2	$\cdots$	n_k
m_1	m_2	$\cdots$	m_k

where $n_1 n_2 \ldots n_k$ represents the integral part of z in binary presentation with the least significant digit first, and $m_1 m_2 \ldots m_k$ represents the fractional part of z in base p with the most significant digit first. If needed, we put additional zeros to the right to make the lengths of integral and fractional part equal. For example, if $p = 3$ then the element $14\dfrac{17}{27} \in R_3^+$ is represented by

0	1	1	1
1	2	2	0

Let the domain D of an FA presentation of R_p^+ consist of all words in Σ^* not ending in $\binom{0}{0}$, except for $\binom{0}{0}$ itself which represents 0. Clearly, D is FA recognizable.

Let *Add* be the graph of the addition operation. We prove that *Add* is FA recognizable. First, we construct an auxiliary automaton $\mathcal{A}$ whose alphabet is $(\Sigma \cup \{\square\})^3$. The states of $\mathcal{A}$ are q_0, $(0,0)$, $(0,1)$, $(1,0)$, $(1,1)$, where q_0 is an initial state and $(0,0)$ is a final state. The state (α, β) means that we have a carry bit α in the addition of integral parts, and a carry bit β in the addition of fractional parts.

The transitions of $\mathcal{A}$ are defined below. It is assumed there that the special symbol $\square$ is identical to the symbol $\binom{0}{0}$.

There is a transition from q_0 to (α, β) with the label $\left(\binom{n_1}{m_1}, \binom{n_2}{m_2}, \binom{n_3}{m_3}\right)^\top$ if and only if

$$\begin{cases} n_1 + n_2 = 2\alpha + n_3 \\ m_1 + m_2 + \beta = m_3 \end{cases} \quad \text{or} \quad \begin{cases} n_1 + n_2 + 1 = 2\alpha + n_3 \\ m_1 + m_2 + \beta = p + m_3 \ . \end{cases}$$

This means that from the first letter of the input $\mathcal{A}$ guesses the carry bit from the fractional part to the integral part: in the first case the carry bit is 0, while in the second case the carry bit is 1.

There is a transition from (α, β) to (α', β') with the label $\left(\binom{n_1}{m_1}, \binom{n_2}{m_2}, \binom{n_3}{m_3}\right)^\top$ if and only if

$$\begin{cases} n_1 + n_2 + \alpha = 2\alpha' + n_3 \\ m_1 + m_2 + \beta' = p\beta + m_3 \ . \end{cases}$$

Now as one can see $\mathrm{Conv}(Add) = L(\mathcal{A}) \cap \mathrm{Conv}(D^3)$. Therefore, since D^3 is FA recognizable, then so is *Add*.

Let us define an FA presentation for R_p. Consider the presentation of R_p^+ given above; let $\pi : D^2 \to D^2$ be the following function

$$\pi(x, y) = \begin{cases} (x - y, 0) & \text{if } x \geqslant y, \\ (0, y - x) & \text{if } x < y. \end{cases}$$

Note that the graph of π is an FA recognizable subset of D^4 since it can be defined in terms of *Add* and $\leqslant$ relations which are FA recognizable in our presentation of R_p^+. Now the domain of the FA presentation of R_p is

$$\{(x,y):\ x,y \in D \text{ and } (x=0 \vee y=0)\}$$

with addition operation defined as

$$(x_1,y_1)+(x_2,y_2)=(x_3,y_3) \text{ if and only if } (x_3,y_3)=\pi(x_1+x_2,y_1+y_2). \qquad \square$$

3 Amalgamations of Monoids and Abelian Groups

Before turning to abelian groups let us consider commutative monoids which have cancellation property, namely, $a+c=b+c$ implies $a=b$ for all elements a, b, c. In what follows, by **monoid** we will mean **commutative monoid with cancellation property**.

Proposition 3.1. *Let M, N, and U be monoids, and $f: U \to M$, $g: U \to N$ be isomorphic embeddings. Consider the direct product $M \times N$ of the monoids and a relation $\sim_U$ on $M \times N$ defined as follows:*

$$(x_0,y_0) \sim_U (x_1,y_1) \iff \exists u,v \in U\ \big(x_0+f(u)=x_1+f(v) \wedge y_0+g(u)=y_1+g(v)\big)$$

Then $\sim_U$ is a congruence on $M \times N$, and $M \oplus_U N$, the **amalgamated product** *of M and N over U, is the quotient structure $M \times N/\sim_U$, which also is a commutative monoid with cancellation property.*

Proof. It is straightforward to show that $\sim_U$ is a congruence, and that $M \oplus_U N$ is a commutative monoid. We prove that it possesses cancellation property. Suppose $(x_0,y_0)+(z,w) \sim_U (x_1,y_1)+(z,w)$; then $x_0+z+f(u)=x_1+z+f(v)$ and $y_0+w+g(u)=y_1+w+g(v)$ for some $u,v \in U$. Since M and N possess cancellation property, we have that $x_0+f(u)=x_1+f(v)$ and $y_0+g(u)=y_1+g(v)$, that is $(x_0,y_0) \sim_U (x_1,y_1)$. $\square$

We will use the notation $\langle x,y\rangle_U$ to denote the equivalence class of $(x,y) \in M \times N$ with respect to $\sim_U$.

Proposition 3.2. *Let $M \oplus_U N$ be an amalgamated product of monoids M and N over U. Then there are submonoids $\widetilde{M}$ and $\widetilde{N}$ in $M \oplus_U N$ such that $\widetilde{M} \cong M$, $\widetilde{N} \cong N$, and $M \oplus_U N = \widetilde{M} + \widetilde{N}$.*

Proof. Let $\widetilde{M} = \{\langle x,0\rangle_U : x \in M\}$ and $\widetilde{N} = \{\langle 0,y\rangle_U : y \in N\}$; as one can see, $\widetilde{M}$ and $\widetilde{N}$ are submonoids of $M \oplus_U N$, and $M \oplus_U N = \widetilde{M} + \widetilde{N}$. Consider the mappings $\varphi : M \to \widetilde{M}$ and $\psi : N \to \widetilde{N}$ such that $\varphi(x) = \langle x,0\rangle_U$ and $\varphi(y) = \langle 0,y\rangle_U$. Clearly, φ and ψ are epimorphisms. Let us show, for instance, that φ is one-to-one. Suppose $\langle x,0\rangle_U = \langle x',0\rangle_U$; then $x+f(u)=x'+f(v)$ and

$g(u) = g(v)$ for some $u, v \in U$. Therefore, $u = v$ and the cancellation property implies that $x = x'$. □

In the case of abelian groups we can define the notion of amalgamated product in a slightly different manner.

Definition 3.1. *Let A, B, and U be abelian groups and $f : U \to A$, $g : U \to B$ be isomorphic embeddings. Then $A \oplus_U B$, the* **amalgamated product** *of A and B over U, is the quotient group $A \oplus B/\widetilde{U}$, where $\widetilde{U} = \{(f(u), g(u)) \mid u \in U\}$.*

The next proposition is the strengthening of 3.2 for abelian groups.

Proposition 3.3. *Let $A \oplus_U B$ be an amalgamated product of A and B over U. Then there are subgroups $\widetilde{A}$ and $\widetilde{B}$ in $A \oplus_U B$ such that $\widetilde{A} \cong A$, $\widetilde{B} \cong B$, $A \oplus_U B = \widetilde{A} + \widetilde{B}$ and $\widetilde{A} \cap \widetilde{B} \cong U$.*

Proof. By definition $A \oplus_U B = A \oplus B/\widetilde{U}$. Let $\widetilde{A} = \{(a, 0) + \widetilde{U} \mid a \in A\}$ and $\widetilde{B} = \{(0, b) + \widetilde{U} \mid b \in B\}$. As one can see, $A \oplus_U B = \widetilde{A} + \widetilde{B}$ and $\widetilde{A} \cong A$, $\widetilde{B} \cong B$. We now prove that $\widetilde{A} \cap \widetilde{B} \cong U$. Let $x \in \widetilde{A} \cap \widetilde{B}$, then $x = (a, 0) + \widetilde{U}$ and $x = (0, b) + \widetilde{U}$; hence $(a, -b) \in \widetilde{U}$ and $a = f(u)$, $b = -g(u)$. Therefore, $x = (f(u), 0) + \widetilde{U}$ and $\widetilde{A} \cap \widetilde{B} = \{(f(u), 0) + \widetilde{U} \mid u \in U\}$, which is isomorphic to U. □

Remark 3.1. The intersection $\widetilde{M} \cap \widetilde{N}$ of the submonoids of $M \oplus_U N$ defined in the proof of Proposition 3.2 does not necessarily isomorphic to U. To show this let M, N, U be $(\mathbb{N}, +)$, and f, g be the identity embeddings. As one can see, $M \oplus_U N$ is isomorphic to $(\mathbb{Z}, +)$ because $\langle x, y\rangle_{\mathbb{N}} = \langle x', y'\rangle_{\mathbb{N}}$ iff $x - y = x' - y'$, and we can identify $\langle x, y\rangle_{\mathbb{N}}$ with $x - y \in \mathbb{Z}$. In this case $\widetilde{M}$ and $\widetilde{N}$ correspond to the submonoids of non-negative and non-positive numbers respectively. Thus $\widetilde{M} \cap \widetilde{N} = \{\langle 0, 0\rangle_{\mathbb{N}}\} \ncong \mathbb{N}$.

The converse of 3.3 also holds.

Proposition 3.4. *Let L be an abelian group, A, B subgroups of L, and $U = A \cap B$. Then*

$$A + B \cong A \oplus_U B,$$

where the embeddings f, g of U into A and B are the identity mappings.

Proof. In this case $A \oplus_U B = A \oplus B/\widetilde{U}$, where $\widetilde{U} = \{(u, u) \mid u \in U\}$. Let $\varphi : A \oplus_U B \to A + B$ be defined as follows:

$$\varphi((a, b) + \widetilde{U}) = a - b.$$

We show that φ is an isomorphism. First, note that it is well defined: if $(a, b) + \widetilde{U} = (a', b') + \widetilde{U}$ then $(a - a', b - b') = (u, u)$ for some $u \in U$; therefore $a - b = (a' + u) - (b' + u) = a' - b'$.

It is easy to see that φ is an epimorphism. We now prove that it is one-to-one. Let $a - b = a' - b'$; then $a - a' = b - b' \in A \cap B = U$. Therefore, $(a - a', b - b') = (u, u) \in \widetilde{U}$ and $(a, b) + \widetilde{U} = (a', b') + \widetilde{U}$. □

Remark 3.2. If M, N, and U are abelian groups then both definitions of amalgamated product, that is the one for groups and the one for monoids, give us the same structure $M \oplus_U N$.

4 Constructions of FA Presentable Monoids and Abelian Groups

In this section we will prove a version of Proposition 3.2 for FA presentable structures.

Theorem 4.1. *If M, N, and U are FA presented monoids and $f : U \to M$, $g : U \to N$ are isomorphic embeddings that are FA recognizable subsets of $U \times M$ and $U \times N$ respectively, then the amalgamated product $M \oplus_U N$ is FA presentable. Moreover, $M \oplus_U N$ contains FA recognizable submonoids $\widetilde{M}$ and $\widetilde{N}$ such that $\widetilde{M} \cong M$, $\widetilde{N} \cong M$ and $M \oplus_U N = \widetilde{M} + \widetilde{N}$.*

Proof. We prove that $M \oplus_U N$ is FA presentable by constructing an interpretation of it in the FA presentable structure $E = M \sqcup N \sqcup U$ enriched with unary predicates for subsets M, N, U and binary predicates R_f and R_g for the graphs of f and g. Let R^M and R^N be the graphs of the addition operation in M and N respectively.

The domain for $M \oplus_U N$ is defined in E^2 by the formula $\Delta(x_0, y_0) = M(x_0) \wedge N(y_0)$. Addition is defined by

$$\Phi(x_0, y_0, x_1, y_1, x_2, y_2) = R^M(x_0, x_1, x_2) \ \wedge \ R^N(y_0, y_1, y_2).$$

Equality is defined by

$$\begin{aligned}\epsilon(x_0, y_0, x_1, y_1) = \exists u, v \ (U(u) \wedge U(v) \wedge x_0 + f(u) = x_1 + f(v) \\ \wedge\, y_0 + g(u) = y_1 + g(v))\end{aligned}$$

or more formally

$$\begin{aligned}\epsilon(x_0, y_0, x_1, y_1) =& \exists u, v, w_0, w_1, w_2, w_3, z_0, z_1 \ (U(u) \wedge U(v) \wedge R_f(u, w_0) \\ & \wedge R_f(v, w_1) \wedge R_g(u, w_2) \wedge R_g(v, w_3) \wedge R^M(x_0, w_0, z_0) \\ & \wedge R^M(x_1, w_1, z_0) \wedge R^N(y_0, w_2, z_1) \wedge R^N(y_1, w_3, z_1)).\end{aligned}$$

From the proof of Proposition 3.2 it follows that $\widetilde{M}$ and $\widetilde{N}$ are defined by the formulas

$$\begin{aligned}(z_0, z_1) \in \widetilde{M} \iff & \exists x, u, v \ (M(x) \wedge U(u) \wedge U(v) \wedge \\ & z_0 + f(u) = x + f(v) \wedge z_1 + g(u) = g(v)), \\ (z_0, z_1) \in \widetilde{N} \iff & \exists y, u, v \ (N(y) \wedge U(u) \wedge U(v) \wedge \\ & z_0 + f(u) = f(v) \wedge z_1 + g(u) = y + g(v)).\end{aligned}$$

Therefore, $\widetilde{M}$ and $\widetilde{N}$ are FA recognizable submonoids. □

Theorem 4.2. *Let A and B be abelian groups such that B is a subgroup of A and $|A : B|$ is finite. If B is FA presentable, then so is A.*

Proof. Let $r_0, \dots, r_k$ be representatives of the cosets of B in A. Then there are a function $g : \{0, \dots, k\}^2 \to \{0, \dots, k\}$ and elements $b_{ij} \in B$ with the following property: for every i and j,

$$r_i + r_j = r_{g(i,j)} + b_{ij}.$$

We may assume that the FA presentation of B uses an alphabet Σ, such that $0, \dots, k \notin \Sigma$, and that the domain of this presentation is $D \subseteq \Sigma^*$. Let the alphabet of the FA presentation of A be $\Sigma \cup \{0, \dots, k\}$. Each element of A has the unique form $r_i + b$ for some $b \in B$, and is represented by the string iv, where $v \in D$ represents b. Since A is abelian,

$$(r_i + b_1) + (r_j + b_2) = r_{g(i,j)} + b_{ij} + b_1 + b_2.$$

Hence the graph of addition operation can be recognized by a finite automaton.

□

Example 4.1 (Two different presentations of R_6). Consider the presentation of R_6 described in Section 2. We will show that R_6 in this presentation does not have FA recognizable subgroup isomorphic to R_2. Suppose M is an FA recognizable subgroup of R_6 and $M \cong R_2$. Let $M^+ = \{(x, 0) : (x, 0) \in M\}$; then M^+ is FA recognizable and $M^+ \cong R_2^+$. Note that we can identify the FA presentation of R_6^+ and the FA recognizable submonoid of R_6 with domain $\{(x, 0) : (x, 0) \in R_6\}$. This implies that R_6^+ has an FA recognizable submonoid isomorphic to R_2^+.

Now if $M^+ \leqslant R_6^+$ is isomorphic to R_2^+, then for some $n_0, k_0 \in \mathbb{N}$

$$M^+ = \frac{n_0}{6^{k_0}} \cdot R_2^+ = \left\{ \frac{n_0 n 3^k}{6^{k_0+k}} \ : \ k, n \in \mathbb{N} \right\}.$$

For each k and let α_k be the smallest element of M^+ of the length $k_0 + k$ in this presentation. Obviously, $\alpha_k = n_0 3^k 6^{-(k_0+k)}$ and it has the form

$$\begin{array}{|c|c|c|c|c|c|c|}
\hline
0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\hline
0 & 0 & \cdots & 0 & \multicolumn{3}{c|}{r_k} \\
\hline
\end{array}$$

where

$$\lim_{k \to \infty} \frac{\mathit{length}(r_k)}{(k + k_0)} = \log_6 3. \tag{1}$$

Choosing sufficiently large k we will have enough leading zeros in the presentation of α_k to pump this string. This will give us a contradiction with the formula (1). Therefore, M^+ is not FA recognizable, and M is not FA recognizable too.

On the other hand, R_6 is isomorphic to $R_2^+ \oplus_{\mathbb{N}} R_3^+$. Indeed, $R_2^+ \oplus_{\mathbb{N}} R_3^+ = \{\langle x, y \rangle_{\mathbb{N}} : (x, y) \in R_2^+ \times R_3^+\}$ and

$$\langle x, y \rangle_{\mathbb{N}} = \langle x', y' \rangle_{\mathbb{N}} \text{ if and only if } x - y = x' - y'.$$

Let $z = m/6^k \in R_6$; there are $m_0, m_1 \in \mathbb{Z}$ such that $m = 3^k m_0 - 2^k m_1$; then $z = m_0/2^k - m_1/3^k = (m_0/2^k + l) - (m_1/3^k + l)$ for any $l \in \mathbb{Z}$. Choosing sufficiently large l we see that $z = x - y$, where $x \in R_2^+$, $y \in R_3^+$. Therefore, the mapping that sends $\langle x, y\rangle_{\mathbb{N}}$ to $x - y \in R_6$ gives us desired isomorphism.

Consider FA presentations for R_2^+ and R_3^+ described in Section 2. Recall that the integral part of every element is presented in base 2 both in R_2^+ and R_3^+. Thus if we take FA presentation of $\mathbb{N}$ in base 2, then the graphs of the identity embeddings $f : \mathbb{N} \to R_2^+$ and $g : \mathbb{N} \to R_3^+$ will be FA recognizable. Therefore, by Theorem 4.1, R_6 has an FA presentation which contains FA recognizable submonoids isomorphic to R_2^+ and R_3^+. Now if $M^+ \subseteq R_6$ is a submonoid isomorphic to R_2^+ then $M = M^+ \cup -M^+$ is a subgroup isomorphic to R_2. Clearly, M is definable in terms of M^+ and addition. Therefore, this presentation of R_6 contains FA recognizable subgroups isomorphic to R_2 and R_3. It is different from the presentation given in Section 2, in the sense that there is no automatic isomorphism between them.

5 Indecomposable FA Presentable Abelian Groups

We describe rank n torsion-free abelian groups, G_n and H_n, which are indecomposable and strongly indecomposable respectively. We then show how to apply the methods from the previous section to prove that they are FA presentable.

In what follows we will use expressions like $p^{-\infty}a$ as an abbreviation for an infinite set $p^{-1}a, p^{-2}a, \cdots$. For every $n \geqslant 2$, let G_n be a subgroup of $\mathbb{Q}^n$ generated by $p_1^{-\infty}e_1, \ldots, p_n^{-\infty}e_n$, $q^{-1}(e_1 + \cdots + e_n)$, where $q, p_1, \ldots, p_n$ are different primes and $e_1, \ldots, e_n$ are linear independent elements in $\mathbb{Q}^n$ considered as a $\mathbb{Q}$-vector space. An example of such a group was found in [4, vol. 2, §88, Exercise 2].

Theorem 5.1. *The group G_n is indecomposable for all $n \geqslant 2$.*

Proof. First, note that every $x \in G_n$ has the form

$$x = (p_1^{-k_1}m_1 + q^{-1}s)e_1 + \cdots + (p_n^{-k_n}m_n + q^{-1}s)e_n,$$

where $m_1, \ldots, m_n, s \in \mathbb{Z}$ and $k_1, \ldots, k_n \in \mathbb{N}$. Let $E_j = \langle p_j^{-\infty}e_j\rangle$ where $1 \leqslant j \leqslant n$. We show that the groups E_j are fully invariant in G_n, i.e. $\varphi(E_j) \subseteq E_j$ for any endomorphism φ of G_n. Let $x \in E_j$ and $\varphi(x) = \sum s_i e_i$. In G_n, x is divisible by all powers of p_j, and so is $\varphi(x)$. Hence, $s_i = 0$ for $i \neq j$ and $\varphi(x) = s_j e_j$.

Take any $i \neq j$. As mentioned above, s_j has the form $p_j^{-k_j}m_j + q^{-1}s$ and s_i has the form $p_i^{-k_i}m_i + q^{-1}s$. Since $s_i = 0$, $q^{-1}s$ must be an integer. Therefore, $\varphi(x) = s_j e_j$ belongs to E_j.

Now suppose that $G_n = A \oplus B$. If $x \in G_n$, then x has the unique form $x = a + b$, where $a \in A$, $b \in B$. Define the following endomorphisms of G_n: $\varphi^A(x) = a$ and $\varphi^B(x) = b$, where $x = a + b$. Obviously, $x = \varphi^A(x) + \varphi^B(x)$. If $x \in E_j$ then $\varphi^A(x) \in E_j \cap A$ and $\varphi^B(x) \in E_j \cap B$, since E_j is fully invariant. This means that $E_j = (E_j \cap A) \oplus (E_j \cap B)$.

Note that E_j is indecomposable, because it has rank 1. Therefore, $E_j \subseteq A$ or $E_j \subseteq B$. Assume there exists $1 \leqslant k < n$ such that $E_1, \dots, E_k \subseteq A$ and $E_{k+1}, \dots, E_n \subseteq B$. Let $q^{-1}(e_1 + \cdots + e_n) = a + b$, where $a \in A$ and $b \in B$. Then $e_1 + \cdots + e_k + e_{k+1} + \cdots + e_n = qa + qb$. Since $e_1 + \cdots + e_k \in A$ and $e_{k+1} + \cdots + e_n \in B$ we have that $a = q^{-1}(e_1 + \cdots + e_k)$.

We show that this is impossible. Let $a = (p_1^{-k_1} m_1 + q^{-1}s)e_1 + \cdots + (p_n^{-k_n} m_n + q^{-1}s)e_n$; since $p_n^{-k_n} m_n + q^{-1}s = 0$, $q^{-1}s$ must be an integer. Hence $p_1^{-k_1} m_1 + q^{-1}s = 0$ cannot be equal to q^{-1}.

So we can assume that $E_1, \dots, E_n \subseteq A$. If $B \neq \mathbf{0}$ then let $b \in B$ be a nonzero element. Then there exists $n > 0$ such that $nb \in \langle e_1, \dots, e_n \rangle \subseteq E_1 + \cdots + E_n \subseteq A$, which is impossible because $nb \neq 0$ and nb is an element of B. Therefore, $B = \mathbf{0}$. □

Definition 5.1 ([1]). *A torsion-free abelian group A is* **strongly indecomposable** *if whenever $0 \neq k \in \mathbb{N}$ and $kA \leqslant B \oplus C \leqslant A$ then $B = \mathbf{0}$ or $C = \mathbf{0}$.*

The group H_n from the next theorem was introduced in [1, Example 2.4].

Theorem 5.2. *The group $H_n = \langle p_1^{-\infty} e_1, \dots, p_n^{-\infty} e_n,\ q^{-\infty}(e_1 + \cdots + e_n) \rangle$ is strongly indecomposable for all $n \geqslant 2$.*

Proof. First, we show that any endomorphism of H_n is the same as multiplication by an integer. Let $x \in H_n$, by an argument similar to one in the beginning of the proof of Theorem 5.1, one can show that if x is divisible in H_n by all powers of p_i, then x has the form $mp_i^{-k} e_i$.

Now let $\widehat{e}_1 = -e_1, \dots, \widehat{e}_{n-1} = -e_{n-1}$, $\widehat{e}_n = e_1 + \cdots + e_n$. Then $e_n = \widehat{e}_1 + \cdots + \widehat{e}_n$ and we can write

$$H_n = \langle p_1^{-\infty} \widehat{e}_1, \dots, p_{n-1}^{-\infty} \widehat{e}_{n-1},\ p_n^{-\infty}(\widehat{e}_1 + \cdots + \widehat{e}_n),\ q^{-\infty} \widehat{e}_n) \rangle.$$

Therefore, if x is divisible in H_n by any power of q then it has the form $mq^{-k}\widehat{e}_n = mq^{-k}(e_1 + \cdots + e_n)$.

Let φ be an endomorphism of H_n; $\varphi(e_i) = r_i e_i$ where $r_i = m_i p_i^{-k_i}$, because $\varphi(e_i)$ is divisible by any power of p_i. Hence $\varphi(e_1 + \cdots + e_n) = \sum r_i e_i$. On the other hand, since $\varphi(e_1 + \cdots + e_n)$ is divisible by all powers of q, it has the form $mq^{-k}(e_1 + \cdots + e_n)$. Therefore, each r_i is equal to an integer number r and $\varphi(x) = rx$. Since the group is torsion-free, every nonzero endomorphism is one-to-one.

To conclude the proof we will show that if a torsion-free abelian group A has only one-to-one nonzero endomorphisms then it is strongly indecomposable. Assume that there are $k \neq 0$ and nonzero groups B and C such that $kA \leqslant B \oplus C \leqslant A$. Let ψ be an endomorphism of $B \oplus C$ defined as follows: if $x = b + c$ where $b \in B$, $c \in C$ then $\psi(x) = b$. Then the mapping φ defined by $\varphi(x) = \psi(kx)$ is an endomorphism of A. Take any $0 \neq c \in C$, then $\varphi(c) = \psi(kc) = 0$ and, therefore, φ is not one-to-one. Note that φ is also nonzero, since if $0 \neq b \in B$ then $\varphi(b) = kb \neq 0$. □

Theorem 5.3. *The group G_n is FA presentable.*

Proof. Since R_p is FA presentable, the direct sum $R_{p_1} \oplus \cdots \oplus R_{p_n}$ is also FA presentable. Note that $R_{p_1} \oplus \cdots \oplus R_{p_n}$ is a subgroup of finite index in G_n. Hence, by Theorem 4.2, G_n is also FA presentable. □

Remark 5.1. Note that unlike G_n, the group H_n is not an extension of finite index of any known example of an FA presentable group. To show that it is FA presentable we will use the method of amalgamated products described in section 4.

Theorem 5.4. *The group H_n is FA presentable.*

Proof. First, let us show that H_n is isomorphic to $(R^+_{p_1} \times \cdots \times R^+_{p_n}) \oplus_{\mathbb{N}} R^+_q$, an amalgamated product of monoids $R^+_{p_1} \times \cdots \times R^+_{p_n}$ and R^+_q over $\mathbb{N}$, where the isomorphic embeddings $f : \mathbb{N} \to R^+_{p_1} \times \cdots \times R^+_{p_n}$ and $g : \mathbb{N} \to R^+_q$ are chosen as follows: for all $m \in \mathbb{N}$, $f(m) = (m, \ldots, m)$ and $g(m) = m$.

Every element of $(R^+_{p_1} \times \cdots \times R^+_{p_n}) \oplus_{\mathbb{N}} R^+_q$ is of the form $\langle (a_1, \ldots, a_n), b\rangle_{\mathbb{N}}$, where $a_i \in R^+_{p_i}$, for $i = 1, \ldots, n$, and $b \in R^+_q$. Suppose that

$$\langle (a_1, \ldots, a_n), b\rangle_{\mathbb{N}} = \langle (a'_1, \ldots, a'_n), b'\rangle_{\mathbb{N}}$$

then there are $u, v \in \mathbb{N}$ such that

$$\begin{cases} (a_1 + u, \ldots, a_n + u) = (a'_1 + v, \ldots, a'_n + v) \\ b + u = b' + v. \end{cases}$$

This implies that $a_i - b = a'_i - b'$ for all $i = 1, \ldots, n$. Thus we can correctly define a function h on $(R^+_{p_1} \times \cdots \times R^+_{p_n}) \oplus_{\mathbb{N}} R^+_q$ such that

$$h(\langle (a_1, \ldots, a_n), b\rangle_{\mathbb{N}}) = (a_1 - b)e_1 + \cdots + (a_n - b)e_n.$$

As one can see, the range of h is a subset of H_n, and h is a homomorphism. To show that it is one-to-one, assume that

$$h(\langle (a_1, \ldots, a_n), b\rangle_{\mathbb{N}}) = h(\langle (a'_1, \ldots, a'_n), b'\rangle_{\mathbb{N}});$$

then

$$(a_1 - b)e_1 + \cdots + (a_n - b)e_n = (a'_1 - b')e_1 + \cdots + (a'_n - b')e_n.$$

Therefore, $a_i - a'_i = b - b' \in R_{p_i} \cap R_q$ for $i = 1, \ldots, n$. Since $R_{p_i} \cap R_q = \mathbb{Z}$ there is $w \in \mathbb{Z}$ such that

$$\begin{cases} (a_1, \ldots, a_n) = (a'_1 + w, \ldots, a'_n + w) \\ b = b' + w. \end{cases}$$

So $\langle (a_1, \ldots, a_n), b\rangle_{\mathbb{N}} = \langle (a'_1, \ldots, a'_n), b'\rangle_{\mathbb{N}}$.

Now to prove that h is onto, consider an element $z \in H_n$; it must be of the form

$$z = \Big(\frac{m_1}{p_1^{k_1}} + \frac{l}{q^r}\Big)e_1 + \cdots + \Big(\frac{m_n}{p_n^{k_n}} + \frac{l}{q^r}\Big)e_n.$$

for integers m_i, l, and natural numbers k_i, r. Obviously,

$$\frac{m_i}{p_i^{k_i}} + \frac{l}{q^r} = \Big(\frac{m_i}{p_i^{k_i}} + t\Big) - \Big(-\frac{l}{q^r} + t\Big)$$

for any $t \in \mathbb{Z}$. Choosing sufficiently large t we can make all $a_i = m_i/p_i^{k_i} + t$ and $b = -l/q^r + t$ to be positive. In this case $\langle (a_1, \dots, a_n), b\rangle_{\mathbb{N}}$ is an element of $(R^+_{p_1} \times \cdots \times R^+_{p_n}) \oplus_{\mathbb{N}} R^+_q$ and $h(\langle (a_1, \dots, a_n), b\rangle_{\mathbb{N}}) = z$. Therefore, the range of h is H_n, and hence it is an isomorphism.

Consider FA presentations of monoids $R^+_{p_1}, \dots, R^+_{p_n}$, and R^+_q described in Section 2. From this we can easily construct an FA presentation of $R^+_{p_1} \times \cdots \times R^+_{p_n}$ by putting strings representing elements of $R^+_{p_i}$ one under another in a column using an extra padding symbol when necessary. Recall that the integral part of an element of $R^+_{p_i}$ or R^+_q is presented in base 2. Therefore, if we consider the presentation of $\mathbb{N}$ in base 2, then the graphs of isomorphic embeddings $f : \mathbb{N} \to R^+_{p_1} \times \cdots \times R^+_{p_n}$ and $g : \mathbb{N} \to R^+_q$ will be FA recognizable. Now, by Theorem 4.1, the structure $(R^+_{p_1} \times \cdots \times R^+_{p_n}) \oplus_{\mathbb{N}} R^+_q$ is FA presentable, and as shown above it is isomorphic to H_n. $\square$

6 A New FA Presentation of $\mathbb{Z}^2$

Let $(\mathbb{Z}, +)$ be the group of integers under addition. In this section we will construct an FA presentation of $(\mathbb{Z}, +)^2$ in which no nontrivial cyclic subgroup is FA recognizable.

Consider $\mathbb{Z}[x]/\langle p_3\rangle$, the quotient of the polynomial ring $\mathbb{Z}[x]$ with respect to the ideal generated by $p_3(x) = x^2 + x - 3$. We will use the notation $p(x) \sim q(x)$ to denote that $p_3(x)$ divides $p(x) - q(x)$.

Remark 6.1. In the construction described below we can use any polynomial of the form $x^2 + x - q$, for a prime $q \geqslant 3$, instead of $p_3(x)$.

Let $A = (\mathbb{Z}[x]/\langle p_3\rangle, +)$ be the additive group of the ring $\mathbb{Z}[x]/\langle p_3\rangle$. It is not hard to see that A is isomorphic to $\mathbb{Z}^2$, since every polynomial in $\mathbb{Z}[x]$ is equivalent over $\langle p_3\rangle$ to a linear polynomial $kx + l$, which can be identified with a pair $(k, l) \in \mathbb{Z}^2$.

We say that a polynomial $a_n x^n + \cdots + a_0 \in \mathbb{Z}[x]$ is in **reduced form** (or briefly **reduced**) if $|a_i| \leqslant 2$ for all $i \leqslant n$.

Proposition 6.1. *For every $p(x) \in \mathbb{Z}[x]$, there is a reduced polynomial $\widetilde{p}(x)$ equivalent to it.*

Proof. This can be proved by induction: assume that $p(x)$ is in reduced form and show that $p(x) \pm x^n$ is equivalent to a reduced polynomial. It is enough to consider $p(x) \pm 1$. Note that if $p_0(x)$ and $p_1(x)$ are in reduced form and have non-negative coefficients then $p_0(x) - p_1(x)$ is reduced. Moreover, any reduced $p(x)$ is equal to the difference of such $p_0(x)$ and $p_1(x)$. So, it is enough to consider the case when $p(x)$ is reduced and has non-negative coefficients and show that $p(x)+1$ is equivalent to a reduced polynomial with non-negative coefficients.

We will actually prove a stronger statement: if $p(x)$ is a reduced polynomial with non-negative coefficients then $p(x)+(a_1x+a_0)$, where $0 \leqslant a_0, a_1 \leqslant 2$, is equivalent to a polynomial of the same sort. The proof is now by induction on the degree of $p(x)$.

Let us write $p(x)$ as $p(x) = p_1(x)x^2 + (b_1x + b_0)$; now using the fact that $3 \sim x^2 + x$ we have

$$\begin{aligned} p(x) + (a_1x + a_0) &= p_1(x)x^2 + (a_1 + b_1)x + (a_0 + b_0) \\ &\sim (p_1(x) + r_1x + (r_0 + r_1))x^2 + c_1x + c_0, \end{aligned}$$

where

$$c_0 = \begin{cases} a_0 + b_0, & \text{if } a_0 + b_0 < 3 \\ a_0 + b_0 - 3, & \text{otherwise} \end{cases} \qquad r_0 = \left[\frac{a_0 + b_0}{3}\right],$$

$$c_1 = \begin{cases} a_1 + b_1 + r_0, & \text{if } a_1 + b_1 + r_0 < 3 \\ a_1 + b_1 + r_0 - 3, & \text{otherwise} \end{cases} \qquad r_1 = \left[\frac{a_1 + b_1 + r_0}{3}\right].$$

Here $[v]$ is the integral part of v, defined by

$$[v] = \begin{cases} \max\{k \in \mathbb{Z} : k \leqslant v\} & \text{if } v \geqslant 0, \\ \min\{k \in \mathbb{Z} : v \leqslant k\} & \text{if } v < 0. \end{cases}$$

For example, $[1.5] = 1$ and $[-1.5] = -1$. Note that $0 \leqslant c_0, c_1 \leqslant 2$ and $0 \leqslant r_0, r_1 \leqslant 1$. By induction, $p_1(x) + r_1x + (r_0 + r_1)$ is equivalent to a reduced polynomial with non-negative coefficients; hence so is $p(x)$. □

We now give an automatic presentation for the group A. The alphabet of the presentation is $\Sigma = \{-2, -1, 0, 1, 2\}$. Each reduced polynomial $a_nx^n + \cdots + a_0$ is represented by a word $a_0 \ldots a_n \in \Sigma^*$. We say that two words $a_0 \ldots a_n$ and $b_0 \ldots b_m$ from Σ^* are **equivalent** if $a_nx^n + \cdots + a_0 \sim b_mx^m + \cdots + b_0$.

This equivalence relation is FA recognizable. An algorithm for checking it is as follows. Given two words $a_0 \ldots a_n$, $b_0 \ldots b_m$; we can assume that $n = m$ since one can always add extra zeros to the right. The algorithm needs to remember two carries r_0, r_1; initially $r_0 = r_1 = 0$. Note that since $3 \sim x + x^2$, whenever we subtract 3 from any digit we need to add 1 to the next two digits in order to get an equivalent word. That is why we need two carries here.

Now for every $i = 0, \ldots, n$ do the following. Check if 3 divides $a_i - b_i + r_0$. If 'no' then the words are not equivalent. If 'yes' then let $r_0^{\text{old}} = r_0$, $r_1^{\text{old}} = r_1$; redefine

$$r_0 = r_1^{\text{old}} + \frac{a_i - b_i + r_0^{\text{old}}}{3}, \qquad r_1 = \frac{a_i - b_i + r_0^{\text{old}}}{3},$$

and go to step $i+1$. If we reach in this way step n then the words are equivalent if and only if $a_n - b_n + r_0 = 0$ and $r_1 = 0$.

Since at every step $|r_0| \leqslant 4$ and $|r_1| \leqslant 2$, this algorithm requires a constant amount of memory. Now it is not hard to construct a finite automaton recognizing the equivalence.

Consider the following order on Σ: $-2 < -1 < 0 < 1 < 2$. It naturally extents to the length-lexicographical order on Σ^*, denoted as $<_{llex}$. Let the domain of the FA presentation of A be

$$\mathrm{Dom}(A) = \{w \in \Sigma^* : (\forall u <_{llex} w)\ u \text{ is not equivalent to } w\}.$$

This set is FA recognizable since $<_{llex}$ is an FA recognizable relation.

To define addition on $\mathrm{Dom}(A)$ consider the relation $R(x, y, z)$ such that if $x = a_0 \dots a_k$, $y = b_0 \dots b_l$ then $z = c_0 \dots c_n$ is obtained from x and y by applying the following algorithm. Again, let r_0, r_1 be two carries that are initially zero. At every step i starting from 0, let c_i be such that $|c_i| < 3$,

$$c_i \equiv a_i + b_i + r_0 \pmod 3$$

and c_i has the same sign as $a_i + b_i + r_0$. Let $r_0^{\mathrm{old}} = r_0$, $r_1^{\mathrm{old}} = r_1$. Now redefine

$$r_0 = r_1^{\mathrm{old}} + \left[\frac{a_i + b_i + r_0^{\mathrm{old}}}{3}\right], \qquad r_1 = \left[\frac{a_i + b_i + r_0^{\mathrm{old}}}{3}\right],$$

and go to step $i + 1$.

For example, if $x = 2211$ and $y = 22$ then this algorithm produces $z = 120021$. By construction, if $R(x, y, z)$ holds then the polynomial corresponding to z is equivalent over $\langle p_3 \rangle$ to the sum of polynomials represented by x and y. It is easy to see that at every step $|r_0| \leqslant 4$ and $|r_1| \leqslant 2$. Thus, as before, R can be recognized by a finite automaton.

Let $Add(x, y, z)$ be defined as

$$\begin{aligned} Add = \{(x, y, z) : x, y, z \in \mathrm{Dom}(A) \text{ and} \\ \exists w\ (R(x, y, w)\ \wedge\ w \text{ is equivalent to } z)\}. \end{aligned}$$

Since $\mathrm{Dom}(A)$, R, and the equivalence relation are FA recognizable Add is also FA recognizable. Obviously, Add is the graph of addition operation on $\mathrm{Dom}(A)$, and the FA presented structure $(\mathrm{Dom}(A), Add)$ is isomorphic to A.

Our next goal is to show that no nontrivial cyclic subgroup in this presentation of $\mathbb{Z}^2$ is FA recognizable.

Lemma 6.1. *Let $p(x)$ and $q(x)$ be reduced polynomials such that $p(x) \sim q(x)$ and $x^k \mid p(x)$, then $x^k \mid q(x)$.*

Proof. Suppose that $x^k \nmid q(x)$; then $p(x) - q(x) = x^l(a_0 + a_1 x + \cdots)$, where $l < k$, $|a_0| \leqslant 2$, and $a_0 \neq 0$. Since $3 \nmid a_0$, $p_3(x) = x^2 + x - 3$ cannot divide $p(x) - q(x)$, which gives a contradiction. □

For $p(x) \in \mathbb{Z}[x]$, consider the set of all words in Σ^* that represent polynomials equivalent to $p(x)$. All these words start with the same number of zeros. So we say that $p(x)$ **starts with k zeros in reduced form** if there is $w \in \Sigma^*$ representing $p(x)$ that starts with k zeros.

The following lemma will be used several times later on.

Lemma 6.2. *Let n be an integer, then $3^k \mid n$ if and only if n starts with k zeros in reduced form.*

Proof. Suppose that $3^k \mid n$; then $n = 3^k m \sim x^k(x+1)^k m$. Taking a reduced form for $(x+1)^k m$ and multiplying it by x^k we obtain a reduced form for n that starts with k zeros; thus n starts with k zeros in reduced form.

The other implication can be proved by induction on k. First, suppose that n starts with 0 in reduced form and $n = 3m + r$, where $0 < r \leqslant 2$. Take any reduced form for $3m$. Since it starts with 0, $n = 3m + r$ has a reduced form that starts with $r \neq 0$. This contradicts our assumption, and hence $3 \mid n$.

It is not hard to see that if $p(x)$ starts with exactly k zeros in reduced form, and $q(x)$ starts with exactly l zeros, then a reduced form for $p(x)q(x)$ starts with exactly $k + l$ zeros. Now suppose that n starts with $k + 1$ zeros in reduced form; then $n = 3m$ and m starts with k zeros, because $3 \sim 011$ starts with one 0. By induction, $3^k \mid m$, and so we have $3^{k+1} \mid n$. □

Let $\alpha = (\sqrt{13} - 1)/2$ be the positive root of $p_3(x) = x^2 + x - 3$. Consider a mapping $F : \mathbb{Z}[x] \to \mathbb{R}$ such that $F : p(x) \mapsto p(\alpha)$. Obviously,

$$(p + q)(\alpha) = p(\alpha) + q(\alpha) \quad \text{and} \quad (pq)(\alpha) = p(\alpha)q(\alpha).$$

Furthermore, if $p(x) \sim q(x)$ then $p(\alpha) = q(\alpha)$, since $p(\alpha) - q(\alpha) = p_3(\alpha)r(\alpha) = 0$.

Consider an arbitrary nontrivial cyclic subgroup in our presentation of $\mathbb{Z}^2$. It has the form

$$\langle w \rangle = \{n \cdot w : \ n \in \mathbb{Z}\}$$

for some $w \in \mathrm{Dom}(A)$. Let $q(x)$ be the polynomial that corresponds to w. Note that $q(\alpha) \neq 0$ since w represents nonzero element. Indeed, by applying the Euclidean algorithm one can see that there are polynomials s, r with $\deg(r) < 2$ such that $q = s \cdot p_3 + r$. Moreover, since the leading coefficient of p_3 is 1, s and r have integer coefficients. Now if $q(\alpha) = 0$, then $r(\alpha) = 0$, which implies that $r = 0$ since α is irrational. So $q = s \cdot p_3$ is equivalent to 0.

Suppose that $\langle w \rangle$ is recognized by a finite automaton with d states. We know that $3^d \cdot w \in \langle w \rangle$ starts with d zeros in reduced form. So let $3^d \cdot w$ be equivalent to $0^d v \in \mathrm{Dom}(A)$. By pumping lemma, there are $s_0, s_1, t \in \mathbb{N}$ with $t \neq 0$ such that $0^d v = 0^{s_0} 0^t 0^{s_1} v$ and $w_k = 0^{tk + s_0 + s_1} v \in \langle w \rangle$ for all $k \geqslant 0$.

Let $q_k(x)$ be the polynomial that corresponds to w_k. Since $w_k \in \langle w \rangle$, we have that $w_k \sim n_k \cdot w$ for some $n_k \in \mathbb{Z}$. If w starts with m zeros, then n_k starts with at least $tk - m + s_0 + s_1$ zeros in reduced form; thus $3^{tk - m + s_0 + s_1} \mid n_k$.

The fact that w_k is equivalent to $n_k \cdot w$ implies that $q_k(\alpha) = n_k q(\alpha)$. Now, on the one hand,

$$\begin{aligned} |q_k(\alpha)| &\leqslant 2(1 + \alpha + \cdots + \alpha^{|w_k| - 1}) \\ &= 2\frac{\alpha^{|w_k|} - 1}{\alpha - 1} \leqslant 8\alpha^{|w_k|} = 8\alpha^{s_0 + s_1 + |v|}\alpha^{tk} = C_0 \alpha^{tk}. \end{aligned}$$

On the other hand,

$$|q_k(\alpha)| = |n_k||q(\alpha)| \geqslant 3^{tk}3^{s_0+s_1-m}|q(\alpha)| = C_1 3^{tk}.$$

Thus

$$C_1 3^{tk} \leqslant C_0 \alpha^{tk} \quad \text{for all } k \geqslant 0,$$

which is impossible because $\alpha < 3$. Therefore, $\langle w \rangle$ is not FA recognizable.

References

1. D. Arnold, *Finite Rank Torsion Free Abelian Groups and Rings* (Springer-Verlag, 1982).
2. A. Blumensath, E. Grädel, Automatic structures, *15th Annual IEEE Symposium on Logic in Computer Science (Santa Barbara, CA, 2000)*, 51–62, *IEEE Comput. Soc. Press, Los Alamitos, CA,* 2000.
3. C. Delhommé, Non-automaticity of ω^ω, 2001, manuscript.
4. L. Fuchs, *Infinite Abelian Groups*, vol. 1, 2 (Academic Press, 1970, 1973).
5. B. R. Hodgson, *Théories décidables par automate fini*, PhD Thesis, University of Montreal, 1976.
6. B. R. Hodgson, Théories décidables par automate fini, *Annales de Sciences Mathématiques*, 7:39–57, 1983.
7. B. Khoussainov, A. Neorde, Automatic presentations of structures, in D. Leivant (ed.), *Logic and Computational Complexity* (LNCS 960, Springer-Verlag, 1995), 367–392.
8. B. Khoussainov, S. Rubin, Automatic structures: overview and future directions. Weighted automata: theory and applications (Dresden, 2002), *J. Autom. Lang. Comb.*, 2:287–301, 8 (2003).
9. B. Khoussainov, A. Nies, S. Rubin and F. Stephan, Automatic structures: richness and limitations, *Proceedings of the 19th IEEE Symposium on Logic in Computer Science* (IEEE Computer Society, 2004), 110–119.

Embeddings into Free Heyting Algebras and Translations into Intuitionistic Propositional Logic

Michael O'Connor

Cornell University

Abstract. We find a translation with particularly nice properties from intuitionistic propositional logic in countably many variables to intuitionistic propositional logic in two variables. In addition, the existence of a possibly-not-as-nice translation from any countable logic into intuitionistic propositional logic in two variables is shown. The nonexistence of a translation from classical logic into intuitionistic propositional logic which preserves $\wedge$ and $\vee$ but not necessarily $\top$ is proven. These results about translations follow from additional results about embeddings into free Heyting algebras.

1 Introduction

Intuitionistic logic has been explored for many years as a language for computer science, with a guiding principle being the Brouwer-Heyting-Kolmogorov interpretation, under which intuitionistic proofs of implication are functions and existence proofs require witnesses. Higher-order intuitionistic systems which can express a great deal of mathematics, such as Girard's System F and Martin-Löf's type theory (good references are [8] and [3]), have been developed and implemented by prominent computer scientists such as Constable, Huet and Coquand (see [2] and [1]). With all this development and with the existence of well-established topological, Kripke, and categorical semantics for intuitionistic systems, it may come as a surprise that many fundamental structural properties of intuitionistic propositional calculus have not been developed. By way of contrast, corresponding issues for classical logics have been settled for at least 75 years.

Heyting algebras are an equationally defined class of algebras with operations $\vee$, $\wedge$, and $\rightarrow$ and constants $\bot$ and $\top$ (representing "or," "and," "implies," "false," and "true" respectively) that stand in the same relation to intuitionistic propositional logic that Boolean algebras do to classical propositional logic. What follows is a very brief introduction to free Heyting algebras and a summary of the results that will be presented in this paper.

For each $n \in \mathbb{N}$, let $V_n = \{x_1, \ldots, x_n\}$ and let F_n be the set of propositional sentences in variables V_n. Let $\simeq_i^n$ and $\simeq_c^n$ be the intuitionistic and classical logical equivalence relations respectively.

The classical Lindenbaum algebra B_n is then defined as $F_n/\simeq_c^n$ and the intuitionistic Lindenbaum algebra H_n is defined as $F_n/\simeq_i^n$. The operations $\wedge$ and

$\vee$ and the constants $\top$ and $\bot$ are naturally defined on B_n and the operations $\wedge$, $\vee$, and $\rightarrow$ and the constants $\top$ and $\bot$ are naturally defined on H_n. B_n is then isomorphic to the free Boolean algebra on n generators and H_n is the free Heyting algebra on n generators. As usual, the order $\leq$ may be defined from $\wedge$ (or from $\vee$). Like all Heyting algebras, each H_n is also a distributive lattice.

The analogous statements are true for $V_\omega = \{x_1, x_2, \ldots\}$, F_ω, B_ω, and H_ω.

The structure of each B_n and of B_ω is well understood. However, among the free Heyting algebras, only H_1 is completely understood. It is known from [10] that if we let $\phi_1 = \neg x_1$, $\psi_1 = x_1$, $\phi_{i+1} = \phi_i \rightarrow \psi_i$, and $\psi_{i+1} = \phi_i \vee \psi_i$, then each propositional formula in the single variable x_1 is intuitionistically equivalent to exactly one formula in $\{\bot\} \cup \{\phi_i \mid i \in \omega\} \cup \{\psi_i \mid i \in \omega\} \cup \{\top\}$. Further, we can easily write down conditions characterizing the order on those formulas, so that the structure of H_1 is completely characterized.

Although the structure of H_n for $n \geq 2$ and of H_ω is not fully understood, there are a number of facts known. Although not a complete list, the reader is referred to [4], [7], [5], and [6]. A very useful construction is contained in [4] which will we avail ourselves of in this paper and which is described in Section 2 below.

The results of this paper are as follows: There is a lattice-embedding from H_ω into H_2. This obviously implies that there is a lattice-embedding from H_m into H_n for any $m \geq 1$, $n \geq 2$, but in these cases, more is true: For any $n \geq 2$ and $m \geq 1$, there is a ϕ and a $\psi \in H_n$ such that $[\phi, \psi] := \{\rho \in H_n \mid \phi \leq \rho \leq \psi\}$ and H_m are isomorphic as lattices. In addition, the isomorphism from $[\phi, \psi]$ to H_m can be extended to all of H_n, so that there is a surjective lattice-homomorphism from H_n to H_m.

Furthermore, we will show that any countable partial order can be order-embedded into H_2, and that the countable atomless boolean algebra B_ω cannot be lattice-embedded into H_ω.

Some of these results also have significance in terms of *translations* into intuitionistic propositional logic, a notion which we now define.

Let $\vdash$ be the intuitionistic consequence relation. Define a consequence-respecting translation from n-variable intuitionistic logic into m-variable intuitionistic logic to be a function $f\colon F_n \rightarrow F_m$ such that for all $\emptyset \neq \Gamma \subseteq F_n$, $\phi \in F_n$, $\Gamma \vdash \phi$ iff $f(\Gamma) \vdash f(\phi)$.

Define a tautology-respecting translation from n-variable intuitionistic logic into m-variable intuitionistic logic to be a function $f\colon F_n \rightarrow F_m$ such that for all $\phi \in F_n$, $\vdash \phi$ iff $\vdash f(\phi)$.

The term "respecting" is used to emphasize that the property of being consequence-respecting and the property of being tautology-respecting are stronger than the property of being consequence-preserving and tautology-preserving respectively.

We define a $(\wedge, \vee)$-preserving translation from n-variable propositional logic to m-variable propositional logic to be a function $f\colon F_n \rightarrow F_m$ such that for all ϕ, $\psi \in F_n$, $f(\phi \wedge \psi) = f(\phi) \wedge f(\psi)$ and $f(\phi \vee \psi) = f(\phi) \vee f(\psi)$.

We make the obvious modifications to the definitions for translations from classical logic to intuitionistic logic and for ω-variable logics.

Thus, Gödel's double-negation translation (see [9] or [3]) is a tautology-respecting but not consequence-respecting or $(\wedge, \vee)$-preserving translation from ω-variable classical logic to ω-variable intuitionistic logic and Gentzen's translation (again, see [9] or [3]) is a tautology-respecting and consequence-respecting but not $(\wedge, \vee)$-preserving translation from ω-variable classical logic to ω-variable intuitionistic logic. Both of these translations may be restricted to be from n-variable classical logic to n-variable intuitionistic logic for any n.

Some of the results of this paper may then be restated as follows: There is a consequence- and tautology-respecting, $(\wedge, \vee)$-preserving translation of ω-variable intuitionistic logic into 2-variable intuitionistic logic. We also get consequence- and tautology-respecting translations (which aren't $(\wedge, \vee)$-preserving) of ω-variable classical logic into 2-variable intuitionistic logic by composing with Gentzen's translation. This translation may be read off explicitly from the proof contained in this paper together with the construction of [4].

The disjunction property of intuitionistic logic implies that there can be no tautology-respecting translation of classical logic into intuitionistic logic. In addition, we will show that there is no merely consequence-respecting, $(\wedge, \vee)$-preserving translation of ω-variable classical logic into ω-variable intuitionistic logic (and thus not into n-variable intuitionistic logic for any n).

The result that any countable partial order can be embedded in H_2 implies that *any* logic may be translated in a consequence- and tautology-respecting but not necessarily $(\wedge, \vee)$-preserving way into 2-variable intuitionistic logic, as long as the logic is countable.

The author would like to acknowledge Richard Shore and Anil Nerode for many useful conversations and specific comments on this paper, and his parents for their love and guidance.

2 Notation, Terminology, and Bellissima's Construction

As above, let $V_n = \{x_1, x_2, \dots, x_n\}$.

We will use Bellissima's construction ([4]) of, for each n, a Kripke model K_n over V_n satisfying Propositions 1 and 2 below. The construction and relevant facts about it will be stated here.

Given a Kripke model K over V_n, let $\text{Nodes}(K)$ be the set of nodes of K and let $\leq(K)$ be the (non-strict) partial order on $\text{Nodes}(K)$ given by K. If no confusion will result, we may use K in place of $\text{Nodes}(K)$. Given $\alpha \in K$, let $w(\alpha) = \{x_i \in V_n \mid \alpha \Vdash x_i\}$.

We will define a Kripke model K_n in stages, so that $K_n = \bigcup_i K_n^i$ where the K_n^i are defined as follows:

$\text{Nodes}(K_n^0) = \mathcal{P}(V_n)$ and $\leq(K_n^0) = \{(\alpha, \alpha) \mid \alpha \in \text{Nodes}(K_n^0)\}$. For clarity, when we want to emphasize that we are thinking of $U \subseteq V_n$ as a node, we may write $\text{node}(U)$ or $\text{node}_{K_n}(U)$. If we want to also note that it is in K_n^0, we may write $\text{node}^0(U)$ or $\text{node}^0_{K_n}(U)$. For $U \subseteq V_n$, we let $w(\text{node}(U)) = U$.

Given $K_n^0, \ldots, K_n^i$, let $\mathcal{T}_{i+1}$ be the set of subsets T of $\bigcup_{j=0}^{i} K_n^j$ such that $T \cap (K_n^i - K_n^{i-1}) \neq \emptyset$ and such that the elements of T are pairwise incomparable with respect to $\leq(K_n^i)$. Then we define $\text{Nodes}(K_n^{i+1}) - \text{Nodes}(K_n^i)$ to be

$$\{\langle T, U\rangle \mid T \in \mathcal{T}_{i+1}, U \subseteq V_n, U \subseteq \bigcap_{\alpha \in T} w(\alpha) \text{ and if } T = \{\beta\}, \text{ then } U \subsetneq w(\beta)\}$$

For clarity, when we want to emphasize that we are thinking of $\langle T, U\rangle$ as a node, we may write $\text{node}(\langle T, U\rangle)$ or $\text{node}_{K_n}(\langle T, U\rangle)$. If we want to also note that it is in K_n^{i+1}, we may write $\text{node}^{i+1}(\langle T, U\rangle)$ or $\text{node}_{K_n}^{i+1}(\langle T, U\rangle)$.

We declare that $w(\langle T, U\rangle) = U$ and we let $\leq(K_n^{i+1})$ be the reflexive transitive closure of $\leq(K_n^i) \cup \{(\langle T, U\rangle, \beta) \mid \langle T, U\rangle \in K_n^{i+1} - K_n^i, \beta \in T\}$.

Let $k(\phi)$ denote the set of nodes in K_n which force ϕ, for ϕ a propositional formula in n variables.

Proposition 1 ([4]). *For ϕ and ψ propositional formulas in $x_1, \ldots, x_n$, $\phi \vdash \psi$ iff $k(\phi) \subseteq k(\psi)$.*

Proposition 2 ([4]). *For each node $\alpha \in K_n$, there is a ϕ_α such that $k(\phi_\alpha) = \{\beta \in K_n \mid \beta \geq \alpha\}$ and there is a ϕ'_α such that $k(\phi'_\alpha) = \{\beta \in K_n \mid \beta \nleq \alpha\}$.*

We now fix some terminology.

If $\alpha < \beta$ are nodes in some Kripke model, then β is called a successor of α and α a predecessor of β. If there is no γ with $\alpha < \gamma < \beta$, then β is called an immediate successor of α. The assertions "α is above β" and "β is below α" both mean $\alpha \geq \beta$.

For any node α in any Kripke model, $s(\alpha)$ is the set of α's immediate successors.

If $\alpha \in K_n$, then ϕ_α and ϕ'_α are as in Proposition 2.

For each m, $\text{Lev}_m^n = K_n^m - K_n^{m-1}$. This may also be called Lev_m if n is clear from context and may be denoted in English as "level m."

If $\alpha \in K_n$, then $\text{Lev}(\alpha)$ is the unique i such that $\alpha \in \text{Lev}_i^n$. Note that if $\alpha \leq \beta$, $\text{Lev}(\alpha) \geq \text{Lev}(\beta)$.

If T is a set of nodes in K_n, let $r(T) = \{\alpha \in T \mid \neg(\exists \beta \in T)(\beta < \alpha)\}$. Thus, for example, for any T with $|T| \geq 2$, $\langle r(T), \emptyset\rangle \in K_n$. The following facts will be used below and follow without much difficulty directly from the construction.

Fact 1. For $n \geq 2$ and $m \geq 0$, $|\text{Lev}_{m+1}^n| > |\text{Lev}_m^n|$. In particular, there are arbitrarily large levels of K_n.

The following fact is a more general version of the preceding fact.

Fact 2. Let $S \subseteq K_n$, $|S| \geq 3$ and let each element of S be at the same level. Let S' be the downward closure of S. Then $|S' \cap \text{Lev}_{m+1}^n| > |S' \cap \text{Lev}_m^n|$ for any m greater than or equal to the common level of the elements of S .

3 A Lattice Embedding from H_m to H_n for $m \geq 1$, $n \geq 2$

Theorem 3. *Let $n \geq 2$, $m \geq 1$. Then there are ϕ, $\psi \in H_n$ such that H_m is isomorphic to $[\phi, \psi]$. In addition, the isomorphism from $[\phi, \psi]$ to H_m can be extended to a surjective lattice-homomorphism from H_n to H_m.*

Proof. The main work is contained in the following proposition.

Proposition 4. *Let $m \geq 2$ and n be such that there is a level Lev_i^n of K_n and a set $A \subseteq \mathrm{Lev}_i^n$ such that $|A| = m$ and each $\alpha \in A$ has some immediate successor not above any other $\alpha' \in A$. Then there is a ϕ, $\psi \in H_n$ such that H_m is lattice-isomorphic to $[\phi, \psi]$ and in addition, the isomorphism from $[\phi, \psi]$ to H_m can be extended to a surjective lattice-homomorphism from H_n to H_m.*

Proof. Fix A and i from the hypothesis.

Let $A = \{\alpha_1, \ldots, \alpha_m\}$. Let ϕ be

$$\bigvee_i \phi_{\alpha_i}.$$

For each $A' \subseteq A$, let $\gamma_{A'}$ be the node $\langle r(T), \emptyset\rangle$ where $T = A' \cup \bigcup_{\alpha \notin A'} s(\alpha)$. This is valid as the elements of $r(T)$ are pairwise incomparable and $|r(T)| \geq 2$ since $m \geq 2$.

Note that $\gamma_{A'}$ is at level $i + 1$ if A' is nonempty and at level i if A' is empty. Since each α_i has a successor not above any other α_j, if $A' \neq A''$, $\gamma_{A'} \neq \gamma_{A''}$.

Let $S = \{\rho \in K_n^{i+1} \mid (\forall A' \subseteq A)\,(\rho \not\geq \gamma_{A'})\}$ and let ψ be $\psi_0 \wedge \psi_1$ where ψ_0 is

$$\left[\neg\neg(\bigvee_{A' \subseteq A} \phi_{\gamma_{A'}})\right]$$

and ψ_1 is

$$\bigwedge_{\rho \in S} \phi'_\rho$$

Define a function g with domain K_m as follows:

1. $g(\mathrm{node}_{K_m}(U)) = \gamma_{A'}$ where $A' = \{\alpha_k \mid x_k \in V_m - U\}$.

2. $g(\mathrm{node}^{j+1}_{K_m}(\langle T, U\rangle)) = \langle r(T'), \emptyset\rangle$, where $T' = \{g(\delta) \mid \delta \in T\} \cup \{\alpha_k \mid x_k \in V_m - U\}$.

We will show that the range of g is contained in K_n. By induction, what we must show is that $\langle r(T'), \emptyset\rangle$ is in K_n, which will hold as long as $|r(T')| \geq 2$.

Lemma 5. *The function g is into K_n and preserves order and nonorder. For all $\beta \in K_m$ and $x_k \in V_m$, $\beta \Vdash x_k$ iff $g(\beta) \not\leq \alpha_k$.*

Proof. We will prove by induction on i that g restricted to K_m^i satisfies the conditions in the statement of the lemma.

For $i = 0$, observe that $\{g(\mathrm{node}^0(U)) \mid U \subseteq V_m\}$ is pairwise incomparable and that if $U \neq U'$, $g(\mathrm{node}^0(U)) \neq g(\mathrm{node}^0(U'))$ as they have different immediate successors. It is also the case that for all $\mathrm{node}^0(U) \in K_m^0$ and $x_k \in V_m$, $\mathrm{node}^0(U) \Vdash x_k$ iff $x_k \in U$ iff $\gamma_{A'} \not\leq \alpha_k$, where $A' = \{\alpha_k \mid x_k \in V_m - U\}$.

Finally, since each $\gamma_{A'}$ is in K_n, the range of g restricted to K_m^0 is contained in K_n.

Now suppose g restricted to K_m^i satisfies the hypotheses in the statement of the lemma.

We first show that the range of g restricted to K_m^{i+1} is contained in K_n. Let $\langle T, U\rangle \in \mathrm{Lev}_{i+1}^m$. If $|T| \geq 2$, then $|r(T)| \geq 2$ and we are done. If $|T| = \{\beta\}$, then $U \subsetneq w(\beta)$ and T' must contain both $g(\beta)$ and α_k, where $x_k \in w(\beta) - U$. Since $\beta \Vdash x_k$, $g(\beta) \not\leq \alpha_k$. Since it is fairly easy to see that each α_k is not less than any element of the range of g, we must have $|r(T')| \geq 2$.

It is immediate then that g restricted to K_m^{i+1} is preserves order and the immediate successor relation. Each element of Lev_{i+1}^m is of the form $\mathrm{node}^{i+1}(T, U)$. Observe that if $U \neq U'$ and $\langle T, U\rangle, \langle T, U'\rangle \in K_m$, then $g(\mathrm{node}^{i+1}(\langle T, U\rangle)) \neq g(\mathrm{node}^{i+1}(\langle T, U'\rangle))$ as they have different immediate successors. Similarly, if $r(T) \neq r(T')$ then $g(\mathrm{node}^{i+1}(\langle r(T), U\rangle)) \neq g(\mathrm{node}^{i+1}(\langle r(T'), U'\rangle))$ as they have different immediate successors. We can now conclude that g preserves nonorder by using the inductive hypothesis and the fact that g preserves the immediate successor relation.

Lemma 6. *The sets* $\mathrm{ran}(g)$ *and* $k(\phi)$ *are disjoint and* $\mathrm{ran}(g) \cup k(\phi) = k(\psi)$.

Proof. It is immediate that $\mathrm{ran}(g)$ and $k(\phi)$ are disjoint.

We will first show that $\mathrm{ran}(g) \cup k(\phi) \subseteq k(\psi)$. It is clear that $k(\phi) \subseteq k(\psi)$. Since every node in $\mathrm{ran}(g)$ is at level $\geq i+1$, every node in $\mathrm{ran}(g)$ forces ψ_1. Since ψ_0 is doubly negated and every successor of a node in $\mathrm{ran}(g)$ is in $\mathrm{ran}(g)$ or $k(\phi)$, by induction every element of $\mathrm{ran}(g)$ forces ψ_0 and $\mathrm{ran}(g) \subseteq k(\psi)$.

We will now show that $k(\psi) \subseteq \mathrm{ran}(g) \cup k(\phi)$. By construction, $k(\psi) \cap K_n^i = k(\phi) \cup \{\gamma_A\}$ and $k(\psi) \cap \mathrm{Lev}_{i+1}^n = \mathrm{ran}(g) \cap \mathrm{Lev}_{i+1}^n = \mathrm{ran}(g|\mathrm{Lev}_0^m) - \gamma_A$. We will show that $k(\psi) \cap \mathrm{Lev}_j^n \subseteq \mathrm{ran}(g|\mathrm{Lev}_{j-(i+1)}^m)$ for all $j \geq i+1$ by induction on j. We just observed that this holds for $j = i+1$.

Suppose it holds for j. A node of $k(\psi) \cap \mathrm{Lev}_{j+1}^n$ must be of the form $\mathrm{node}^{j+1}(\langle T, \emptyset\rangle)$ for $T \subseteq k(\psi) \cap K_n^j$. Since T must contain an element of $k(\psi) \cap \mathrm{Lev}_j^n$ and every such node is below every element of $k(\psi) \cap \mathrm{Lev}_{i-1}^n$, T must be a subset of $k(\psi) \cap (K_n^j - K_n^{i-1})$. Let $S = g^{-1}(T \cap (K_n^j - K_n^i))$ and $U = \{x_k \mid \alpha_k \in T \cap \mathrm{Lev}_i\}$. Then $g^{-1}(\mathrm{node}_{K_n}^{j+1}(\langle T, \emptyset\rangle))$ is $\mathrm{node}_{K_m}^{j-i}(\langle S, \bigcap_{\mu \in S} w(\mu) - U\rangle)$

It follows from Lemmas 5 and 6 that g is an order-isomorphism from K_m to $k(\phi) - k(\psi)$.

Define $f\colon F_m \to F_n$ by:
1. $f(\bot) = \phi$
2. $f(x_i) = (\phi'_{\alpha_i} \vee \phi) \wedge \psi$
3. $f(\rho_0 \wedge \rho_1) = f(\rho_0) \wedge f(\rho_1)$.
4. $f(\rho_0 \vee \rho_1) = f(\rho_0) \vee f(\rho_1)$.
5. $f(\rho_0 \to \rho_1) = (f(\rho_0) \to f(\rho_1)) \wedge \psi$.

Lemma 7. *For any* $\rho \in F_m$, $\phi \vdash f(\rho) \vdash \psi$. *If* $\delta = g(\gamma)$ *then* $\gamma \Vdash \rho$ *iff* $\delta \Vdash f(\rho)$.

Proof. The proof that $\phi \vdash f(\rho) \vdash \psi$ is an easy proof by induction on ρ.

We now prove the second part of the lemma by induction on ρ.

For $\rho = \bot$, the result is immediate. The observation that $\gamma \Vdash x_i$ iff $\delta \not< \alpha_i$ furnishes the case where ρ is x_i. The inductive steps follow from the existence of the order-isomorphism g from K_m to $k(\psi) - k(\phi)$ and the fact that $\phi \vdash f(\rho)$ for all ρ.

Note that it follows from Lemma 7 that f is injective and hence an embedding.

We now define a function from F_n to F_m that is an inverse to f when restricted to $[\phi, \psi]$. Define h from F_n to F_m as follows:

1. $h(\bot) = h(x_i) = \bot$.
2. $h(\rho_0 \wedge \rho_1) = h(\rho_0) \wedge h(\rho_1)$.
3. $h(\rho_0 \vee \rho_1) = h(\rho_0) \vee h(\rho_1)$.
4. If there is some $\delta \in k(\phi) \cap K_n^{i-1}$ such that $\delta \nVdash \rho_0 \rightarrow \rho_1$, then $h(\rho_0 \rightarrow \rho_1) = \bot$. Otherwise,

$$h(\rho_0 \rightarrow \rho_1) = (h(\rho_0) \rightarrow h(\rho_1)) \wedge \bigwedge \{x_i \mid \alpha_i \nVdash \rho\}$$

Lemma 8. *Let $\delta = g(\gamma)$. For all $\rho \in F_n$, $\delta \Vdash \rho$ iff $\gamma \Vdash h(\rho)$.*

Proof. We will prove this by induction on the level of γ and the structure of ρ.

If ρ is $\bot$ or x_i, then $\delta \nVdash \rho$ and $\gamma \nVdash h(\rho)$.

The inductive step for $\rho = \rho_0 \vee \rho_1$ and $\rho = \rho_0 \wedge \rho_1$ is straightforward.

Let ρ be $\rho_0 \rightarrow \rho_1$. Suppose $\delta \Vdash \rho$. Then, since for every $\mu \in k(\phi) \cap K_n^{i-1}$, $\delta < \mu$, $h(\rho) = (h(\rho_0) \rightarrow h(\rho_1)) \wedge \bigwedge\{x_i \mid \alpha_i \nVdash \rho\}$. Since $\delta \Vdash \rho$, if $\alpha_i \nVdash \rho$, $\delta \not< \alpha_i$. It follows that $\gamma \Vdash x_i$. Thus γ forces the right conjunct of $h(\rho)$.

Suppose $\delta \Vdash \rho_0$ and $\delta \Vdash \rho_1$. Then we are done by the inductive hypothesis on the structure of ρ. Otherwise, suppose $\delta \nVdash \rho_0$. Then we are done by the inductive hypothesis on the structure of ρ and the level of γ.

Now suppose $\delta \nVdash \rho$. Then there is some $\mu \geq \delta$ such that $\mu \Vdash \rho_0$ and $\mu \nVdash \rho_1$. If μ is in the range of g then we are done by induction. If $\mu \in K_n^{i-1}$, then $h(\rho) = \bot$ and we are done. Otherwise $\mu \in K_n^i$ and is some α_j. Since $\delta < \alpha_j$, $\gamma \nVdash x_j$ and $\gamma \nVdash h(\rho)$.

It follows from Lemma 8 and Lemma 7 that if $\phi \vdash \rho \vdash \psi$, then $f(h(\rho)) = \rho$.

If $n \geq 2$, $m \geq 2$ by Fact 1 we can find a level in K_n satisfying the hypotheses of the Proposition. For example, we may pick a level in K_n of cardinality greater than $2m$, call $2m$ of its elements $\beta_1, \ldots, \beta_{2m}$, and let $A = \{\langle\{\beta_1, \beta_2\}, \emptyset\rangle, \ldots, \langle\{\beta_{2m-1}, \beta_{2m}\}, \emptyset\rangle\}$.

If $m = 1$, then we may let ϕ be $\bot$ and ψ be $x_2 \wedge \ldots \wedge x_n$. The embedding f from H_1 to $[\phi, \psi] \subseteq H_n$ sends ρ to $\rho \wedge x_2 \wedge \ldots \wedge x_n$. We may define a surjective lattice homomorphism h from H_n to H_1 that is an inverse to f as follows:

$h(x_1) = x_1$
$h(x_i) = \top$ for $1 < i \leq n$
$h(\phi \wedge \psi) = h(\phi) \wedge h(\psi)$
$h(\phi \vee \psi) = h(\phi) \vee h(\psi)$
$h(\phi \rightarrow \psi) = h(\phi) \rightarrow h(\psi)$

Corollary 9. *There is a consequence-respecting, $(\wedge, \vee)$-preserving translation but not tautology-respecting from m-variable intuitionistic logic to n-variable intuitionistic logic for $n \geq 2$.*

Proof. Immediate.

Corollary 10. *There is a consequence- and tautology-respecting translation from m-variable intuitionistic logic to n-variable intuitionistic logic for $n \geq 2$.*

Proof. Let $f\colon F_m \to F_n$ be a consequence-respecting translation from m-variable intuitionistic logic to n-variable intuitionistic logic. Define f' by $f'(\phi) = f(\top) \to f(\phi)$.

Then f' is consequence- and tautology-respecting. To see that it is consequence-respecting: If $\Gamma \vdash \phi$ then $f(\Gamma) \vdash f(\phi)$, so $f(\top) \to f(\Gamma), f(\top) \vdash f(\phi)$ and $f(\top) \to f(\Gamma) \vdash f(\top) \to f(\phi)$, where $f(\top) \to f(\Gamma)$ is an abbreviation of $\{f(\top) \to \psi \mid \psi \in f(\Gamma)\}$.

Conversely, if $f(\top) \to f(\Gamma), f(\top) \vdash f(\phi)$, then $f(\Gamma) \vdash f(\phi)$ since $f(\Gamma) \vdash f(\top) \to f(\Gamma)$ and $f(\Gamma) \vdash f(\top)$ (this last fact is due to the fact that f is consequence-preserving).

Corollary 11. *There is a consequence- and tautology-respecting, $(\wedge,\vee)$-preserving translation from m-variable intuitionistic logic to n-variable intuitionistic logic for $n \geq 2$.*

Proof. Let $f\colon F_m \to F_n$ be a consequence-respecting and $(\wedge, \vee)$-preserving translation from m-variable intuitionistic logic to n-variable intuitionistic logic. Define f' by

$$f'(\phi) = \begin{cases} f(\phi) & \nvdash^m_{\mathcal{I}} \phi \\ \top & \vdash^m_{\mathcal{I}} \phi \end{cases}$$

This is clearly still consequence-respecting and $\wedge$-preserving. The disjunction property of intuitionistic logic implies that it is also $\vee$-preserving.

Note that the translations given in Corollaries 9 and 10 can be done in linear time, while the one given in Corollary 11 cannot, as it requires deciding whether the given formula is a tautology.

By [4], H_n for $n \geq 2$ has an infinite descending chain, while H_1 does not, so there is no embedding of H_n into H_1 for $n \geq 2$.

4 A Lattice-Embedding from H_ω to H_n for $n \geq 2$

Theorem 12. *There is a lattice-embedding from H_ω into H_2*

Proof. Pick $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5 \in K_2$, all at the same level, say i. This may be done by Fact 1. Let $S = \{\rho \in K_2^i \mid \forall i \in \{1,2,3,4\}\, \rho \ngeq \alpha_i\}$. Let ϕ be

$$(\neg\neg(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4)) \wedge \bigwedge_{\beta \in S} \phi'_\beta.$$

Let $T = \{\rho \in K_2^i \mid \forall i \in \{1,2,3,4,5\}\, \rho \ngeq \alpha_i\}$. Let ψ be

$$(\neg\neg(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4 \vee \alpha_5)) \wedge \bigwedge_{\beta \in T} \phi'_\beta.$$

Define a sequence $\{\beta_i^j \mid i \in \omega, j \in \{1,2,3,4\}\}$ as follows: let $\beta_0^j = \alpha_j$. For $i \geq 0$, let $\{\beta_{i+1}^j \mid j = 1,2,3,4\}$ be a collection of four distinct nodes of the same level, with $\mathrm{Lev}(\beta_{i+1}^1) > \mathrm{Lev}(\beta_i^1)$ and such that they all force $\neg\neg(\beta_i^2 \vee \beta_i^3 \vee \beta_i^4)$. For example, we may take $\beta_{i+1}^1 = \langle\{\beta_i^2, \beta_i^3\}, \emptyset\rangle$, $\beta_{i+1}^2 = \langle\{\beta_i^2, \beta_i^4\}, \emptyset\rangle$, $\beta_{i+1}^3 = \langle\{\beta_i^3, \beta_i^4\}, \emptyset\rangle$, and $\beta_{i+1}^4 = \langle\{\beta_i^2, \beta_i^3, \beta_i^4\}, \emptyset\rangle$.

As in [4] (where a very similar construction is done), the nodes of $\{\beta_i^1 \mid i \in \omega\}$ are pairwise incomparable, and they all force ϕ.

Define a Kripke model K over the language $V_\omega = \{x_i \mid i \in \omega\}$ as follows: The set of nodes of K is the set $k(\psi) - k(\phi)$ and a node α forces x_i iff $\alpha \not\leq \beta_i^1$.

For all $\phi \in F_\omega$, let $k(\phi) = \{\alpha \in K \mid \alpha \Vdash \phi\}$.

Lemma 13. *For all $\phi, \psi \in F_\omega$, $k(\phi) \subseteq k(\psi)$ iff $\phi \vdash \psi$.*

Proof. Since K is a Kripke model, if $\phi \vdash \psi$, $k(\phi) \subseteq k(\psi)$.

Suppose $\phi \nvdash \psi$. Then there is a rooted finite Kripke model K' over V_ω such that $K' \Vdash \phi$ and $K' \nVdash \psi$. Since variables not occurring in ϕ or ψ are irrelevant, we may assume that each node of K' forces cofinitely many propositional variables.

Define a map $a \colon K' \to K$ inductively on K' as follows: If $\gamma \in K'$ is a node such that $a(\gamma')$ has defined for all immediate successors of γ, then let $a(\gamma)$ be a node whose set of successors in K_2 is the upward-closure of the set $\{\beta_i^1 \mid \gamma \nVdash x_i\} \cup \{a(\gamma') \mid \gamma' \geq \gamma\} \cup \{\alpha_5\}$.

For each i, $\gamma \Vdash x_i$ iff $a(\gamma) \Vdash x_i$. Since a is also order-preserving and its range is upward-closed in K, we have that if γ is the root of K', $a(\gamma) \Vdash \phi$ and $a(\gamma) \nVdash \psi$.

Now, as before, define $f \colon F_\omega \to F_2$ by:

1. $f(\bot) = \phi$
2. $f(x_i) = (\phi'_{\beta_i^1} \vee \phi) \wedge \psi$
3. $f(\rho_0 \wedge \rho_1) = f(\rho_0) \wedge f(\rho_1)$.
4. $f(\rho_0 \vee \rho_1) = f(\rho_0) \vee f(\rho_1)$.
5. $f(\rho_0 \to \rho_1) = (f(\rho_0) \to f(\rho_1)) \wedge \psi$.

By precisely the same argument as before, this is an embedding.

Note that, by [4], in any interval $[\phi, \psi] \subseteq H_n$, there are atomic elements. As there are no atomic elements in H_ω, H_ω cannot be embedded in H_n as an interval.

5 Impossibility of Lattice-Embedding B_ω into H_ω

Let B_ω be the countable atomless Boolean algebra. We will think of it as the Lindenbaum algebra of classical propositional logic on countably infinitely many variables.

Proposition 14. *There is no lattice embedding from B_ω into H_n for any n or into H_ω.*

Proof. By the previous theorem, it suffices to prove the proposition for H_2. Suppose there is a lattice embedding of B_ω into H_2. Call it f.

Let $f(\top)$ have n subformulas. Consider the 2^n formulas $\phi_1 = x_1 \wedge \cdots \wedge x_n$, $\phi_2 = x_1 \wedge \cdots \wedge \neg x_n, \ldots, \phi_{2^n} = \neg x_1 \wedge \cdots \wedge \neg x_n$. Since f preserves $\wedge$ and $\vee$ we must have that $\{k(f(\phi_i)) \mid 1 \leq i \leq 2^n\}$ is a partition of $k(f(\top)) - k(f(\bot))$ and that $k(f(\phi_i)) \cap (k(f(\top)) - k(f(\bot)))$ is non-empty for each i.

Lemma 15. *Let α be a node in a Kripke model with exactly two immediate successors, α_1 and α_2. Let ϕ be a formula. Suppose that for each subformula ϕ' of ϕ, $\alpha_1 \Vdash \phi'$ iff $\alpha_2 \Vdash \phi'$ and that for all propositional variables v appearing in ϕ, if α_1 and α_2 force v, then $\alpha \Vdash v$. Then for each subformula ϕ' of ϕ, $\alpha \Vdash \phi'$ iff $\alpha_1 \Vdash \phi'$. In particular, $\alpha \Vdash \phi$ iff $\alpha_1 \Vdash \phi$ iff $\alpha_2 \Vdash \phi$.*

Proof. By induction on the structure of ϕ. The conclusion is immediate if ϕ is atomic, and the $\wedge$ and $\vee$ cases are straightforward.

Suppose ϕ is $\phi_1 \to \phi_2$. By induction, we can conclude that $\alpha \Vdash \phi'$ iff $\alpha_1 \Vdash \phi'$ if ϕ' is a subformula of ϕ_1 or ϕ_2. We just have to verify that $\alpha \Vdash \phi$ iff $\alpha_1 \Vdash \phi$.

Suppose $\alpha \Vdash \phi$. Then, as $\alpha_1 \geq \alpha$, $\alpha_1 \Vdash \phi$.

Now suppose $\alpha \nVdash \phi$. Thus, there must be some $\alpha' \geq \alpha$ such that $\alpha' \Vdash \phi_1$ and $\alpha' \nVdash \phi_2$. If $\alpha' = \alpha$ then we are done by induction. Otherwise, we must have $\alpha' \geq \alpha_1$ or $\alpha' \geq \alpha_2$, and thus $\alpha_1 \nVdash \phi$.

For each i, let $\beta_i \in k(f(\phi_i)) \cap (k(f(\top)) - k(f(\bot)))$. By the pigeonhole principle, there must be some i and j, $i \neq j$, such that $\beta_i \Vdash \phi'$ iff $\beta_j \Vdash \phi'$ for all subformulas ϕ' of $f(\top)$. Let β be $\langle\{\beta_i, \beta_j\}, w(\beta_i) \cap w(\beta_j)\rangle$. We can easily verify that $\beta \in K_2$. By the lemma, $\beta \in f(\top)$. Thus, β is in $k(f(\phi_m)) \cap (k(f(\top)) - k(f(\bot)))$ for some m. Without loss of generality, say $m \neq i$. Then $\beta_i \Vdash f(\phi_m)$ and $\beta_i \Vdash f(\phi_i)$ but $\beta_i \nVdash f(\bot)$, a contradiction.

6 Order-Embeddings

Proposition 16. *Any countable partial ordering can be order-embedded into H_2 (and, therefore, into H_n for any $n \geq 2$).*

Proof. We first make the following definition:

Definition 17 ($\psi(\alpha_1, \ldots, \alpha_m)$, Permissive formulas). *Let $\{\alpha_1, \ldots, \alpha_m\}$ be a set of nodes of K_2 all with the same level. Let $S(\alpha_1, \ldots, \alpha_m) = \{\delta \in K_2 \mid \mathrm{Lev}(\delta) \leq \mathrm{Lev}(\alpha_1)$ and $\forall i\, \delta \not\geq \alpha_i\}$.*

We define $\psi(\alpha_1, \ldots, \alpha_m)$ to be

$$\left(\neg\neg \bigvee_{i=1}^{m} \phi_{\alpha_i}\right) \wedge \bigwedge_{\delta \in S(\alpha_1,\ldots,\alpha_m)} \phi'_\delta$$

If $T = \{\alpha_1, \ldots, \alpha_m\}$ then $\psi(T)$ will denote $\psi(\alpha_1, \ldots, \alpha_m)$. If some α_i is at a different level than some α_j, $\psi(\alpha_1, \ldots, \alpha_m)$ is not defined.

A formula of the form $\psi(\alpha_1, \ldots, \alpha_m)$ where $m \geq 3$ will be called permissive. The set $\{\alpha_1, \ldots, \alpha_m\}$ is called the set of generators of $\psi(\alpha_1, \ldots, \alpha_m)$ and $\mathrm{Lev}(\alpha_1)$ is called the level of $\psi(\alpha_1, \ldots, \alpha_m)$.

Lemma 18. *Given any permissive formula ψ, there exist permissive formulas ψ_n for $n \in \{0,1\}$ such that for each $n \in \{0,1\}$, $k(\psi_n) \subseteq k(\psi)$ and $k(\psi_0) \cap k(\psi_1)$ is finite.*

Proof. Let i be greater than the level of ψ with $|\mathrm{Lev}_i \cap k(\psi)| \geq 6$. We can find i by Fact 2. Let $\mathrm{Lev}_i \cap k(\psi) = \{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \ldots\}$, let $\psi_0 = \psi(\alpha_1, \alpha_2, \alpha_3)$ and $\psi_1 = \psi(\beta_1, \beta_2, \beta_3)$.

Definition 19 (ψ_σ). *We define ψ_σ, for $\sigma \in \{0,1\}^{<\omega}$ as follows: Let $\psi_\varepsilon = \top$. Given ϕ_σ, define $\phi_{\sigma n}$ for $n \in \{0,1\}$ so that ϕ_{σ_n} is permissive, $k(\phi_{\sigma n}) \subseteq k(\phi_\sigma)$, and $k(\phi_{\sigma 0}) \cap k(\phi_{\sigma 1})$ is finite as in the above lemma.*

Note that $\psi_\sigma \vdash \psi_{\sigma'}$ iff σ is an initial segment of σ' as a binary string. Note also that $\psi_\sigma \vdash \bigvee_i \psi_{\sigma_i}$ iff there is an i such that $\sigma = \sigma_i$.

Definition 20 (Complete Sets). *A set $S \subseteq H_2$ such that each element of S is a disjunction of the form $\bigvee_{i=1}^{n} \psi_{\sigma_i}$ is called complete if it satisfies the following property: Let S_1, S_2 be such that $S_1 \cup S_2 = S$, $S_1 \cap S_2 = \emptyset$, S_1 is upward closed, and S_2 is downward closed. Then there is some $\sigma(S_1, S_2)$ such that $|\sigma(S_1, S_2)| >$ $\max\{|\sigma| \mid \psi_\sigma$ a disjunct of a formula in $S\}$, $\psi_{\sigma(S_1,S_2)}$ implies every element of S_1, and $\psi_{\sigma(S_1,S_2)}$ implies no element of S_2.*

Note that the condition that

$$|\sigma(S_1, S_2)| > \max\{|\sigma| \mid \psi_\sigma \textit{ a disjunct of a formula in } S\}$$

means that no $\psi_\sigma \in S$ can imply $\psi_{\sigma(S_1,S_2)}$.

Note also that if σ_1 is such that $|\sigma_1| > \max\{|\sigma| \mid \psi_\sigma$ a disjunct of a formula in $S\}$, ψ_{σ_1} implies every element of S_1, and ψ_{σ_1} implies no element of S_2 then so does $\sigma_1\sigma_2$ for any σ_2 (where the juxtaposition indicates concatenation). Without loss of generality, then, we may assume that each $\sigma(S_1, S_2)$ has the same length.

The proposition will follow from the following lemma.

Lemma 21. *Suppose P is a finite partial order, $S \subseteq H_2$ is a complete set, and h is an isomorphism from P to $(S, \leq)$. For any partial order P' such that $|P'| = |P| + 1$, there is a ϕ of the form $\bigvee_i \psi_{\sigma_i}$ such that $S \cup \{\phi\}$ is complete, $P' \simeq (S \cup \{\phi\}, \leq)$ via an isomorphism extending h.*

Proof. If $P = \emptyset$, let ϕ be ψ_0. This is complete as we may let $\sigma(\{\phi\}, \emptyset) = 00$ and $\sigma(\emptyset, \{\phi\}) = 10$.

Suppose $S = T_1 \cup T_2 \cup T_3$ where T_1 is downward closed and T_2 is upward closed, and we would like to find ϕ so that ϕ is above all the elements of T_1, below all the elements of T_2 and incomparable with the elements of T_3.

Let $\mathcal{S}$ be the collection of all partitions (S_1, S_2) of S such that S_1 is upward closed and S_2 is downward closed.

Let ϕ be

$$\bigvee_{\chi \in T_1} \chi \vee \bigvee \{\psi_{\sigma(S_1,S_2)0} \mid (S_1, S_2) \in \mathcal{S} \text{ and } T_2 \subseteq S_1\}$$

Clearly, ϕ is above every $\chi \in T_1$. We also have that ϕ is below every $\rho \in T_2$, since every $\chi \in T_1$ must be below every $\rho \in T_2$, and by definition every $\psi_{\sigma(S_1,S_2)0}$ with $T_2 \subseteq S_1$ is below every $\rho \in T_2$.

ϕ is not above any element in $T_2 \cup T_3$: As noted above, $\psi_\sigma \vdash \bigvee \psi_{\sigma_i}$ implies $\sigma = \sigma_i$ for some i. But the disjuncts of ϕ are either elements of T_1 (which cannot be implied by elements of T_2 or T_3) or of the form ψ_σ where the length of σ is greater than the length of any σ' for $\psi_{\sigma'}$ some disjunct of a formula in $T_2 \cup T_3$.

ϕ is not below any element in $T_1 \cup T_3$: Let $\mu \in T_1 \cup T_3$. Let $S_2 = \{\mu' \mid \mu' \leq \mu\}$ and $S_1 = S - S_2$. Then $\psi_{\sigma'(S_1,S_2)0}$ is a disjunct of ϕ which does not imply μ.

To see that $S \cup \{\phi\}$ is complete: Let $(S_1, S_2) \in \mathcal{S}$ with $\phi \in S_1$. Since S_1 is upward closed, we must have $T_2 \cup S_1$. Thus $\psi_{\sigma(S_1-\{\phi\},S_2)0}$ is a disjunct of ϕ. We may therefore take $\sigma(S_1, S_2)$ to be $\sigma(S_1 - \{\phi\}, S_2)0$ concatenated with enough zeroes to make its length greater than $\max\{\sigma \mid \psi_\sigma$ a disjunct of a formula in $S \cup \{\phi\}\}$.

Let $(S_1, S_2) \in \mathcal{S}$ with $\phi \in S_2$. Thus $T_1 \subseteq S_2$. If there is any member of T_2 in S_2, then we may take $\sigma(S_1, S_2)$ to be any sufficiently long extension of $\sigma(S_1, S_2 - \{\phi\})$, since $\psi_{\sigma(S_1,S_2-\{\phi\})}$ cannot imply ϕ since ϕ implies each element of S_2.

Thus we may assume that $T_2 \subseteq S_1$. Therefore, $\psi_{\sigma(S_1,S_2-\{\phi\})0}$ is a disjunct of ϕ. By construction, there are no disjuncts of ϕ above it. We may take $\sigma(S_1, S_2)$ to be any sufficiently long extension of $\sigma(S_1, S_2 - \{\phi\})1$.

References

1. *The Coq proof assistant*, `http://coq.inria.fr`.
2. *The PRL project*, `http://www.nuprl.org`.
3. Michael Beeson, *Foundations of constructive mathematics: Metamathematical studies*, Springer, Berlin/Heidelberg/New York, 1985
4. Fabio Bellissima, *Finitely generated free Heyting algebras*, Journal of Symbolic Logic **51** (1986), 152–165
5. Carsten Butz, *Finitely presented Heyting algebras*, `http://www.itu.dk/~butz/research/heyting.ps.gz`, 1998
6. Luck Darnière and Markus Junker, *On finitely generated Heyting algebras*, `http://home.mathematik.uni-freiburg.de/junker/preprints/heyting-221005.pdf`, 2005
7. Silvio Ghilardi and Marek Zawadowski, *A sheaf representation and duality for finitely presented Heyting algebras*, Journal of Symbolic Logic **60** (1995), 911–939
8. Jean-Yves Girard, Yves Lafont, and Paul Taylor, *Proofs and types*, Cambridge University Press, Cambridge, 1989
9. Anil Nerode, George Odifreddi, and Richard Platek, *Constructive logics and lambda calculi*, in preparation
10. Iwao Nishimura, *On formulas of one variable in intuitionistic propositional calculus*, Journal of Symbolic Logic **25** (1960) 327–331
11. Alasdair Urquhart, *Free Heyting Algebras*, Algebra Universalis **3** (1973) 94–97

Some Puzzles About Probability and Probabilistic Conditionals[⋆]

Rohit Parikh

City University of New York
rparikh@gc.cuny.edu

Abstract. We examine some old and new paradoxes of probability, give a rough account of probabilistic conditionals, and prove a new result about non-monotonicity in probabilistic conditionals. It is well known that such conditionals are not monotonic – a conditional which is true can become false when additional hypotheses are added. We show that nonetheless, the conditionals are *usually* monotonic, or roughly speaking that we do not actually have to worry about non-monotonicity in practice.

Einstein famously said that *God does not play dice*. This attitude of his led to a certain estrangement with Quantum Mechanics and even with Physics during the last years of his life, otherwise marked by a close friendship with Kurt Gödel.

Einstein aside, probability is a puzzling phenomenon, for what exactly does it mean? One supposedly unproblematic way to define it is via frequencies. To say that a coin is fair, i.e., that the probability of heads is .5, can then be interpreted as, *If we toss the coin many times, then approximately half of the results will be heads*. But how many is many? Of course, if we toss the coin a hundred times, we probably will not get exactly fifty heads, and even if we toss it a thousand times, the proportion of heads can diverge quite a bit from .5. This is not *likely*, but explaining that word *likely* looks like it involves probability.

Moreover, if Jack wants to insure a car, the frequency definition is no good. The insurance company needs to know the probability that *he, Jack* will have an accident. And clearly there is no frequency available for the insurance company to resort to. Jack only wants to insure his car *once*. And even if he wants to insure the car a second time, he will be a year older, and hopefully more experienced. The probability of his having an accident, however we define it, will not be the same the second time.

An alternative approach is that due to Ramsey, de Finetti, Savage, etc. Ramsey defines probability subjectively, in terms of the bets which an agent is willing to accept. Your subjective probability of heads is .5 if you are willing to accept a double or nothing bet *in favour* of heads, and also a double or nothing bet *against* heads. But what bets you want to accept is up to you. There is nothing intrinsically objective about your bet-acceptance attitudes.

[⋆] Dedicated to Anil Nerode on his 75th birthday. Research supported in part by a grant from PSC-CUNY grants program. Versions of this paper were given in colloquia at Boston University, CUNY Graduate Center, and at IHPST in Paris.

There *are* standards of rationality which will apply. If you think in October 2004 that the probability that Bush will win is .5 and also that the probability that Kerry will win is .6, then bets can be placed in such a way that no matter what happens you will lose money. For an opponent can bet \$20 at even odds against Bush winning, and also \$16 at 3:2 odds against Kerry winning. Then if Bush wins, you gain \$20 from the first bet but lose \$24 on the second bet. If Kerry wins, you gain \$16 from the second bet and lose \$20 from the first, and finally if neither wins, you lose both bets. You lose in all three cases.

In other words, a *Dutch book* can be made against you, a term which already occurs with de Finetti who says that he is puzzled by the name – the Dutch are even more puzzled! It is of course well known that an agent against whom Dutch book cannot be made has a subjective probability which satisfies the Kolmogorov axioms like $0 \leq p(E) \leq 1$ and $p(E \vee F) = p(E) + p(F)$ when $E \wedge F$ is null.[1]

But even if two people are both rational in the sense that Dutch book cannot be made against either of them, it need not be the case that they both have the same subjective probability. Thus the question *What is the actual probability $p(E)$ of the event E?* cannot be answered. The only solace we have is that if two agents start by assigning positive (but different) probabilities to the same events, and they have the same experiences, and they both revise using Bayes' law, then their probabilities will converge in the long run.

So the notion of probability does have some foundational problems. But I want to put these problems aside, and talk about some paradoxes, both old and recent, and conclude with a new result about probabilistic conditionals.

The Saint Petersburg Paradox

"The St. Petersburg game is played by flipping a fair coin until it comes up tails, and the total number of flips, n, determines the prize, which equals 2^n. Thus if the coin comes up tails the first time, the prize (in dollars) is $2^1 = 2$, and the game ends. If the coin comes up heads the first time, it is flipped again. If it comes up tails the second time, the prize is $2^2 = 4$, and the game ends. If it comes up heads the second time, it is flipped again. And so on. There are an infinite number of possible consequences (runs of heads followed by one tail) possible. The probability of a consequence of n flips ($P(n)$) is 1 divided by 2^n, and the expected payoff of each consequence is the prize times its probability." *Stanford Encyclopedia of Philosophy.*

How much should you pay to participate in the St. Petersburg game? It is easily calculated that the expected payoff is infinite. You will win \$2 with probability .5, \$4 with probability .25, \$8 with probability .125, etc, adding up in all to ∞. Therefore *any* amount whatsoever is acceptable as payment to enter the game. But most people rebel at the thought.

One possible solution to this puzzle is to use utilities rather than payoffs. If we say that the utility value of a financial payoff is *logarithmic* in the payoff, then the expected utility of the St. Petersburg game will be finite and it would

[1] Isaac Levi [14] considers *families* of probabilities, but we shall not go into that here.

be a mistake to pay an amount whose utility would exceed the expected utility of the game. This was essentially the suggestion of Daniel Bernoulli.

But Bernoulli's suggestion would not prevent other paradoxes, for if the payoff were 2^{2n} with n = the number of heads before a tail, then the payoff from each outcome multiplied by the probability of that outcome would again be exponential in dollars, and hence again linear in *utility*. This would again give an infinite expected utility to the game. See also Paul Weirich [22].

The Sleeping Beauty

But let us go on to our second, much more modern paradox due to Adam Elga, based on an earlier paper of Ariel Rubinstein. *Sleeping Beauty* (SB) is put to sleep on a Sunday using some drug, and after that a fair coin is tossed. If the coin comes up heads, she is woken on Monday and asked the question Q (to be described later). If the coin comes up tails, she is woken up on Monday, asked the question Q, and then *put back* to sleep, woken up again on Tuesday and again asked the question Q. Sleeping Beauty knows this procedure and that the coin is fair. But when she is woken, she does not know what day of the week it is and whether she was woken before or not. Of course she does know that the day is either Monday or Tuesday, but she does not know which.
The question Q is,

What is the probability that the coin landed heads?

One answer is that it is .5. The coin is fair and SB knows this. Moreover, she knew all along that she would be woken up. The fact that she is woken up and asked Q is not surprising information which might change the probability. So the answer is .5. *Or is it?*

Suppose it is .5, and so she accepts a double or nothing bet on heads each time that Q is asked. Over a hundred trials, there will be fifty heads and fifty tails, roughly. When the coin lands heads, she will *win* one dollar (say) and when it lands tails, she will *lose* two dollars, one for each time she is asked Q. So she will end up losing \$50 net. This is not compatible with a .5 probability, and the right probability would be 1/3, or about .33.

Which is the right answer? And is there a right answer? Various people have written on this and let me just refer you to recent papers by Halpern, and by Bradley and Leitgeb [11,3]. Halpern claims that the problem here is *asynchrony*, an issue that arises often in distributed computing. Bradley and Leitgeb claim that the connection between betting and probability is only valid under certain presumptions, which fail in this case.

Probabilistic Conditionals

How should we interpret a conditional like, "If John comes to the party, then so will Mary", i.e., of the form *If A then B*? The standard interpretation used in mathematics is to treat it as equivalent to $A \rightarrow B$, i.e., as $\neg A \vee B$. See [20,17]

But often, this does not fit our intuition. The following example due to Dorothy Edgington is instructive. Clearly if God does not exist, then he cannot answer our prayers. Consider the statement S: *If God does not exist, then it is not the case that if I pray, my prayers will be answered.* Many who disagree about the existence of God will tend to accept S. There are two *if*s in S. Can they both be expressed as material conditionals? So suppose we symbolize S as $\neg G \rightarrow \neg(P \rightarrow A)$, interpreting both implications, the main one on the outside, and the subsidiary one inside the parentheses, as material implications. Also suppose I don't pray. Then P is false and $P \rightarrow A$ is true. Hence $\neg(P \rightarrow A)$ is false. But then for S to be true, $\neg G$ must be false and hence G must be true. *I can prove the existence of God simply by not praying!*

Even those who believe in God will find this argument fishy and will look more kindly on other ways, *beside the material conditional*, of interpreting the indicative conditional.[2]

One suggestion, associated with Ernest Adams and Edgington herself [1,5], is to treat the conditional probabilistically. Thus asserting *If A then B* is tantamount to saying that the probability of B given A is high. Someone who says, "If John comes to the party, Mary will come too," is asserting that $p(M|J)$ is high, perhaps more than .9.

Such probabilistic conditionals received a blow from results of David Lewis [15] who showed that (on pain of triviality) such conditionals cannot be interpreted as *propositions*. In other words, there cannot be a proposition (a set of possible worlds) C such that $p(C) = p(B|A)$ for all probability measures P.

But let us leave that worry aside and ask about the *logic* of probabilistic conditionals interpreted not as an implication (i.e., as a connective) but instead as a consequence relation $\mid\!\sim$. This avoids the Lewis' problem because we are *not saying* that *If A then B* is a proposition. Let us say that we accept $A \mid\!\sim B$ if $p(B|A)$ is high, where the $\mid\!\sim$ represents the indicative conditional interpreted probabilistically.

Horacio Arlo Costa and RP [2] looked at probabilistic conditionals in the context of cores,[3] a notion investigated by Bas van Fraasen and we looked at some conditions on non-monotonic relations considered by Dov Gabbay, and by Kraus, Lehmann and Magidor [8,13].

Various rules of inference apply to such consequence relations. Thus from $A \mid\!\sim B$ and $B \models C$ we can derive $A \mid\!\sim C$, where $\models$ represents the classical consequence. This rule is (RW) or right weakening and is sound. So is the rule (AND) which derives $A \mid\!\sim B \wedge C$ from $A \mid\!\sim B$ and $A \mid\!\sim C$ (provided we make some sacrifice in probability[4]). But a rule which does *not* hold is monotonicity (M), or

Strengthening the antecedent. This would be the rule, *from $A \mid\!\sim B$ and $C \models A$ we should be able to derive $C \mid\!\sim A$.* In particular we would like to be able to derive $(A \wedge X) \mid\!\sim B$ from $A \mid\!\sim B$.

$$\text{(M)} \qquad \frac{A \mid\!\sim B}{A \wedge X \mid\!\sim B}$$

[2] This particular variety of conditional is called the *indicative conditional*, to distinguish it from so called subjunctives or counterfactuals.

[3] [2] interpret "high" as 1.

[4] If $p(B|A) > .95$ and $p(C|A) > .95$, then $p(B \wedge C|A) > .9$.

Alas, the rule (M) is known not to be sound. The probability of B given A may be high, and the probability of B given A and X may be low. For instance, if our domain is integers upto 100, then the probability that n is odd given that it is prime is quite high. But the probability that it is odd given that it is prime *and* less than 4 is only .5.

A well known example involves birds. Given that Tweety is a bird there is a high probability that it flies. But given that Tweety is a bird and a penguin, the probability drops to 0.

What we show below is that the rule (M) is *mostly* sound. That is that provided that A is large enough, for *most* X, the conclusion continues to hold.

Probabilistic conditionals are *mostly* monotonic. And this is good news, for clearly, while accepting that the monotonicity condition does not hold *universally*, we do want it to hold *usually*.

For consider birds. If the dictum, "Birds fly'" could be destroyed at the drop of a hat, it would be useless. We could not conclude that female birds fly, that blue birds fly, or that the bird sitting on the window sill is likely to fly. It is almost always the case that when we know that some creature is a bird, we also have *some* additional information X. And usually, we do not drop the dictum *Birds fly* when we have some additional information. Thus it must be the case that the dictum is sort of sturdy. Information like, "It is a penguin," is *unusual.* It is this sturdiness that we will prove below.

We are representing propositions as sets of possible worlds. We have ourselves objected to this identification [16]. But the representation is commonly accepted and a result which uses it ought to have a relevance. The following theorem is stated rather loosely, but will be followed up by a more precisely stated theorem.

Theorem 1. *Suppose that W is a finite space, with all points equally likely. Suppose that A, B are large subsets of W and $p(B|A) > .95$, then for* most *randomly selected $X \subseteq W$, $p(B \cap X|A \cap X) = p(B|A \cap X) > .945$.*

Of course the set theoretic operation $\cap$ corresponds to the logical operation $\wedge$. In terms of $\mid\!\sim$ it means that if you know $A \mid\!\sim B$ and want to know if $A \wedge X \mid\!\sim B$, the answer will be, "Most likely".

We have used the numbers .95, .945, and the uniform distribution on W for convenience, but of course the result holds more generally, as will be evident from the proof. What 'large' B means will be made more explicit below. The technique of proof requires the binomial theorem and its Gaussian approximation. It is quite accessible.

We start by noting some simplifications. Since we are taking probabilities relative to A or its subsets, the points in W which are not in A play no role. So we shall assume that $A = W$. This automatically implies that $B \subseteq A$, an assumption which we could have justified independently.

Suppose now (using our assumption of largeness) that the set $A - B$ has cardinality 10,000 and hence, since $p(B|A) > .95$, B has cardinality (at least) 190,000. A random subset X of A has two parts: X_B which is simply $X \cap B$, and the remaining part X_R of X, which is $X \cap (A - B)$.

The expected size of X_B is 95,000 (half of 190,000) but it could be more or less. But by the weak law of large numbers, the standard deviation σ of the size of X_B is $.5 \times \sqrt{190,000}$, which is between 435 and 436. Thus $95,000 - 3\sigma$ is more than 93,682. It is unlikely that the actual value differs from the expected value by more than 3σ. Indeed, using standard tables, the probability that X_B has size more than 93,682 exceeds .9987. Similarly, the set X_R has expected size 5,000, but the standard deviation σ' is 50. Thus with the same probability .9987, X_R has size less than 5,150. Thus with probability greater than .9974, the ratio $card(X_B)/(card(X_B \cup X_R))$ is greater than 93,682/(93,682+5,150) which is .94789, or very nearly .95. (The figure .9974 comes from the fact that even if both errors of .0013 (1 - .9987) were to add up, we would still only get an error of .0026)

This means that if the sets A, B are both large, and $p(B|A) > .95$, then (when a random subset X of A is chosen) with probability greater than .9974, $p(B \cap X|A \cap X) > .94789$ We can show that similar results will hold if the random set X is chosen in some other way. E.g. if we toss a die for each point of A and put a point in X only if the die shows a 6, the results will still hold.

One could ask if the rule (M) can be called sound if it holds only for *most* X. However, note that $A \mathrel{|\!\!\sim} B$ does not say that if A is true, then B is also true 100% of the time. It only says that if A holds, then B is very likely to hold. If the rule applies 99.74% of the time, and the premise (which we accepted) only 'applies' 95% of the time then it is hard to justify the premise while rejecting the rule.

We now state a more precise result, which actually generalizes the observation above to the case where the probability α of B relative to A is positive, but not necessarily close to 1.[5]

The intuitive idea is that we can think of the set X as a random *sample* from the space W, in which case $X \cap A$ will be a random sample from A. The expected size of $X \cap A$ is half the cardinality $||A||$ of A and its standard deviation is $.5 \times \sqrt{||A||}$. The same holds for the expected size of $X \cap B$ except for the multiplier α. Now if the actual sizes of the two sets were the *same as* their expected sizes, then we would get $p(B \cap X|A \cap X)$ to be *equal* to $p(B|A)$. Of course we cannot expect to be so lucky, for actual size can deviate from the expected size. However, as the sizes of A and B go up, the deviation matters less and less, and so the difference between $p(B \cap X|A \cap X)$ and $p(B|A)$ will tend to zero. This gives us our second result.

Theorem 2. *Let $\alpha > 0$ be fixed and let sets A_n, B_n increase in size with $B_n \subseteq A_n$ and $|p(B_n|A_n) - \alpha| \to 0$. Let X_n be randomly chosen subsets of A_n and $\epsilon > 0$. Then as $n \to \infty$, $p[\ |(||B_n \cap X_n||/||A_n \cap X_n||) - \alpha| > \epsilon] \to 0$.*

In other words, if our prior probability of B given A was $\alpha > 0$, and we received additional information X, then provided A was large, we should expect the

[5] I am using α for this probability – a number, as I am using p for the probability *function.*

posterior probability of B given A to still be close to α, and the probability that it differs by more than ϵ goes to 0 as $||A||$ goes to infinity.

We have not looked at the case where A is not merely large but is actually infinite. Clearly, investigating that case will involve an excursion into measure theory – perhaps methods from Non-standard analysis will allow a transition from the large finite case to the infinite case.

Another rule,

$$\frac{A \mid\!\sim B}{\neg B \mid\!\sim \neg A}$$

is not capable of a similar treatment. If our universe consists solely of the innumerable pigeons and the relatively few penguins, then “Most birds fly” will be true, but “Most non-flyers are non-birds” will be false. Indeed all non-flyers will be birds in our universe!

Acknowledgement. We thank Evan Goris, Dexter Kozen, Roman Kuznets, Sergiu Hart, Matt Johnson, Bud Mishra and Samer Salame for comments.

References

1. E. Adams, “Probability and the logic of conditionals,” in *Aspects of Inductive Logic*, edited by Suppes and Hintikka, (1968), North Holland, 265-316.
2. Horacio Arlo Costa and Rohit Parikh “Conditional probability and defeasible inference,” *Journal of Philosophical Logic*, **34** (2005) 97-119.
3. Darren Bradley and Hannes Leitgeb, “When betting odds and credences come apart: more worries for the Dutch book arguments,” *Analysis*, **66.2**, 2006, 119-127.
4. The binomial theorem, http://www.stat.yale.edu/Courses/1997-98/101/binom.htm
5. Dorothy Edgington, “On Conditionals,” *Mind*, **104**, 1995, 235-329.
6. Adam Elga, “Self-locating belief and the Sleeping Beauty problem,” *Analysis*, **60**, (2000) 143-147.
7. Bruno de Finetti, “Foresight: its logical laws, its subjective sources,” in *Studies in Subjective Probability*, ed. Kyburg and Smokler, Krieger (1980) pp. 53-118 (translation by Kyburg, original French version, 1937).
8. Dov Gabbay, “Theoretical foundations for nonmonotonic reasoning in expert systems,” in *Proceedings NATO Advanced Study Institute on Logics and Models of Concurrent Systems*, (ed.) K.R. Apt, (1985), 439-457.
9. B.V. Gnedenko, *Theory of Probability*, (translated from the Russian by B.D. Seckler), Chelsea 1962.
10. James Hawthorne, “Nonmonotonic Conditionals that Behave Like Conditional Probabilities Above a Threshold,” *Journal of Applied Logic*, May 2006.
11. Joseph Halpern, “Sleeping Beauty reconsidered,” *Oxford Studies in Epistemology*, edited by Gendler and Hawthorne, (2005), pp. 111-142.
12. H. Jeffreys, *Theory of Probability*, New York, Oxford, 3rd edition, (1961).

13. S. Kraus, D. Lehmann, and M. Magidor, "Nonmonotonic reasoning, preferential models and cumulative logics," *Artificial Intelligence*, 44 (1990), 167-207.
14. Isaac Levi, "On indetrminate probabilities," *Journal of Philosophy*, 71 (1974), 391-418.
15. David Lewis, "Probabilities of conditionals and conditional probabilities," *Philosophical Review*, 85 (1976) 297-315.
16. R. Parikh, "Logical omniscience and common knowledge: WHAT do we know and what do WE know?," *Proceedings of the Tenth Conference on Theoretical Aspects of Rationality and Knowledge - 2005*, ed. Ron Meyden, National U. Singapore Press, pp. 62-78.
17. R. Parikh, review of [20], *Essays in Philosophy*, **7,1**, (2006).
18. F. P. Ramsey, "Truth and probability," (1926), reprinted in *F.P. Ramsey, Philosophical Papers*, edited by D.H.Mellor, Cambridge University Press 1990.
19. C. Schwarz, "Cumulative probability for the standard normal distribution," http://www.stat.sfu.ca/∼cschwarz/Stat-301/Handouts/nod122.html
20. David Sanford, *If P then Q*, second edition, Routledge 2003.
21. Daniel Stroock, *Probability Theory: An analytic view*, Cambridge U. Press, 1993.
22. Paul Weirich, "The St. Petersburg Gamble and Risk," *Theory and Decision* 17, (1984) 193-202.

A Temporal Dynamic Logic for Verifying Hybrid System Invariants[⋆]

André Platzer

University of Oldenburg, Department of Computing Science, Germany
Carnegie Mellon University, Computer Science Department, Pittsburgh, PA
platzer@informatik.uni-oldenburg.de

Abstract. We combine first-order dynamic logic for reasoning about possible behaviour of hybrid systems with temporal logic for reasoning about the temporal behaviour during their operation. Our logic supports verification of hybrid programs with first-order definable flows and provides a uniform treatment of discrete and continuous evolution. For our combined logic, we generalise the semantics of dynamic modalities to refer to hybrid traces instead of final states. Further, we prove that this gives a conservative extension of dynamic logic. On this basis, we provide a modular verification calculus that reduces correctness of temporal behaviour of hybrid systems to non-temporal reasoning. Using this calculus, we analyse safety invariants in a train control system and symbolically synthesise parametric safety constraints.

Keywords: dynamic logic, temporal logic, sequent calculus, logic for hybrid systems, deductive verification of embedded systems.

1 Introduction

Correctness of real-time and hybrid systems depends on a safe operation throughout *all* states of all possible trajectories, and the behaviour at intermediate states is highly relevant [1, 7, 9, 12, 14, 23].

Temporal logics (TL) use temporal operators to talk about intermediate states [1, 10, 11, 24]. They have been used successfully in model checking [1, 6, 14, 15, 18] of finite-state system abstractions. Continuous state spaces of hybrid systems, however, often do not admit equivalent finite-state abstractions [14, 18]. Instead of model checking, TL can also be used deductively to prove validity of formulas in calculi [9, 8]. Valid TL formulas, however, only express very generic facts that are true for all systems, regardless of their actual behaviour. Hence, the behaviour of a specific system first needs to be axiomatised declaratively to obtain meaningful results. Then, however, the correspondence between actual system operations and a declarative temporal representation may be questioned.

[⋆] This research was supported by a fellowship of the German Academic Exchange Service (DAAD). It was also sponsored by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS, see www.avacs.org).

Dynamic logic (DL) [13] is a successful approach for deductively verifying (infinite-state) systems [2, 3, 13, 16]. Like model checking, DL can analyse the behaviour of actual system models, which are specified operationally. Yet, operational models are *internalised* within DL-formulas, and DL is closed under logical operators. Thus, DL can refer to multiple systems and analyse their relationship. This can be important for verifying larger systems compositionally or for investigating refinement relations, see [22]. Further, Davoren and Nerode [9] argue that, unlike model checking, deductive methods support formulas with free parameters. However, DL only considers the behaviour at final states, which is insufficient for verifying safety invariants that have to hold all the time.

We close this gap of expressivity by combining first-order dynamic logic [13] with temporal logic [10, 11, 24]. Moreover, we generalise both operational system models and semantics to hybrid systems [14]. In this paper, we introduce a temporal dynamic logic dTL, which provides modalities for quantifying over traces of hybrid systems. We equip it with temporal operators to state what is true all along a trace or at some point during a trace. As in our non-temporal dynamic logic $\mathsf{d}\mathcal{L}$ [19, 20, 22], we use hybrid programs as an operational model for hybrid systems. They admit a uniform treatment of interacting discrete and continuous evolution in logic.

As a semantical foundation for combined temporal dynamic formulas, we introduce a hybrid trace semantics for dTL. We prove that dTL is a conservative extension of $\mathsf{d}\mathcal{L}$: for non-temporal specifications, trace semantics is equivalent to the non-temporal final state semantics of [19, 22].

As a means for verification, we introduce a sequent calculus for dTL that successively reduces temporal statements about traces of hybrid programs to non-temporal formulas. In this way, we make the intuition formally precise that safety invariants can be checked by augmenting proofs with appropriate assertions about intermediate states. Like in [22], our calculus supports compositional reasoning. It structurally decomposes correctness statements about hybrid programs into corresponding statements about its parts by symbolic transformation.

Our approach combines the advantages of DL in reasoning about the behaviour of (multiple and parametric) operational system models with those of TL to verify temporal statements about traces. On the downside, we show that our logic is incomplete. Yet, reachability in hybrid systems is already undecidable [14]. We argue that, despite this theoretical obstacle, dTL can verify practical systems and demonstrate this by studying safety invariants in train control [7, 12].

The first contribution of this paper is the logic dTL, which provides a coherent foundation for reasoning about the temporal behaviour of operational models of hybrid systems with symbolic parameters. The main contribution is our calculus for deductively verifying temporal statements about hybrid systems.

Hybrid Systems. The behaviour of safety-critical systems typically depends on both the state of a discrete controller and continuous physical quantities. Hybrid systems are mathematical models for dynamic systems with interacting discrete and continuous behaviour [9,14]. Their behaviour combines continuous evolution (called *flow*) characterised by differential equations and discrete jumps.

Dynamic Logic. The principle of dynamic logic is to combine system operations and correctness statements about system states within a single specification language (see [13] for a general introduction in the discrete case). By permitting system operations α as actions of modalities, dynamic logic provides formulas of the form $[\alpha]\phi$ and $\langle\alpha\rangle\phi$, where $[\alpha]\phi$ expresses that all terminating runs of system α lead to final states in which condition ϕ holds. Likewise, $\langle\alpha\rangle\phi$ expresses that it is possible for α to execute and result in a final state satisfying ϕ. In dTL, hybrid programs [19, 20, 22] play the role of α. In this paper, we modify the semantics of $[\alpha]$ to refer to all *traces* of α rather than only all final states reachable with α (similarly for $\langle\alpha\rangle$). For instance, the formula $[\alpha]\Box\phi$ expresses that ϕ is true at each state during all traces of the hybrid system α. With this, dTL can also be used to verify temporal statements about the behaviour of α at intermediate states during system runs.

Related Work. Based on [25], Beckert and Schlager [4] added separate trace modalities to dynamic logic and presented a relatively complete calculus. Their approach only handles discrete state spaces. In contrast, dTL works for hybrid programs with continuous state spaces. There, a particular challenge is that invariants may change their truth-value during a single continuous evolution.

Mysore et al. [18] analysed model checking of TCTL [1] properties for semi-algebraic hybrid systems and proved undecidability. Our logic internalises operational models and supports multiple parametric systems.

Zhou et al. [26] presented a duration calculus extended by mathematical expressions with derivatives of state variables. Their calculus is unwieldy as it uses a multitude of rules and requires external mathematical reasoning about derivatives and continuity.

Davoren and Nerode [9] extended the propositional modal μ-calculus with a semantics in hybrid systems and examine topological aspects. In [8], Davoren et al. gave a semantics in general flow systems for a generalisation of CTL* [11]. In both cases, the authors of [9] and [8] provided Hilbert-style calculi to prove formulas that are valid for all systems simultaneously using abstract actions.

The strength of our logic primarily is that it is a first-order dynamic logic: it handles actual hybrid programs like $x := x + 1; \dot{x} = 2y$ rather than only abstract actions of unknown effect. Our calculus directly supports verification of hybrid programs with first-order definable flows; first-order approximations of more general flows can be used according to [23]. First-order DL is more expressive and calculi are deductively stronger than other approaches [4, 17].

Structure of this Paper. After introducing syntax and semantics of the temporal dynamic logic dTL in Sect. 2, we introduce a sequent calculus for verifying temporal dTL specifications of hybrid systems in Sect. 4 and prove soundness. In Sect. 5, we prove safety invariants of the train control system presented in Sect. 3. Alternating path and trace quantifiers for liveness verification are discussed in Sect. 6. Finally, we draw conclusions and discuss future work in Sect. 7.

2 Temporal Dynamic Logic for Hybrid Systems

2.1 Overview: The Basic Concepts of dTL

The temporal dynamic logic dTL extends dynamic logic [13] with three concepts for verifying temporal specifications of hybrid systems:

Hybrid Programs. The behaviour of hybrid systems can be described by hybrid programs [19, 20, 22], which generalise real-time programs [15] to hybrid change. The distinguishing feature of hybrid programs in this context is that they provide uniform discrete jumps and continuous evolutions along differential equations. While hybrid automata [14] can be embedded, program structures are more amenable to compositional symbolic processing by calculus rules [19].

Modal Operators. Modalities of dynamic logic express statements about all possible behaviour ($[\alpha]\pi$) of a system α, or about the existence of a trace ($\langle\alpha\rangle\pi$), satisfying condition π. As in [19, 20, 22], the system α is described as a hybrid program. Yet, unlike in standard dynamic logic [13], π is a *trace formula* in dTL, and π is allowed to refer to all states that occur *during* a trace using temporal operators.

Temporal Operators. For dTL, the temporal trace formula $\Box\phi$ expresses that the formula ϕ holds all along a trace selected by $[\alpha]$ or $\langle\alpha\rangle$. For instance, the state formula $\langle\alpha\rangle\Box\phi$ says that the state formula ϕ holds at every state along at least one trace of α. Dually, the trace formula $\Diamond\phi$ expresses that ϕ holds at some point during such a trace. It can occur in a state formula $\langle\alpha\rangle\Diamond\phi$ to express that there is such a state in some trace of α, or as $[\alpha]\Diamond\phi$ to say that, along each trace, there is a state satisfying ϕ. In this paper, the primary focus of attention is on homogeneous combinations of path and trace quantifiers like $[\alpha]\Box\phi$ or $\langle\alpha\rangle\Diamond\phi$.

2.2 Syntax of dTL

State and Trace Formulas. The formulas of dTL are built over a non-empty set V of real-valued variables and a fixed signature Σ of function and predicate symbols. For simplicity, Σ is assumed to contain exclusively the usual function and predicate symbols for real arithmetic, such as $0, 1, +, \cdot, =, \leq, <, \geq, >$.

The set $\mathrm{Trm}(V)$ of *terms* is defined as in classical first-order logic. The formulas of dTL are defined similar to first-order dynamic logic [13]. However, the modalities $[\alpha]$ and $\langle\alpha\rangle$ accept trace formulas that refer to the temporal behaviour of *all* states along a trace. Inspired by CTL and CTL* [10, 11], we distinguish between state formulas, that are true or false in states, and trace formulas, that are true or false for system traces. The sets $\mathrm{Fml}(V)$ of state formulas, $\mathrm{Fml}_T(V)$ of trace formulas, and $\mathrm{HP}(V)$ of hybrid programs with variables in V are simultaneously inductively defined in Definition 1 and 2, respectively.

Definition 1 (Formulas). *The set* $\mathrm{Fml}(V)$ *of* (state) formulas *is simultaneously inductively defined as the smallest set such that:*

1. *If* $p \in \Sigma$ *is a predicate,* $\theta_1, \ldots, \theta_n \in \mathrm{Trm}(V)$*, then* $p(\theta_1, \ldots, \theta_n) \in \mathrm{Fml}(V)$*.*
2. *If* $\phi, \psi \in \mathrm{Fml}(V)$*, then* $\neg\phi, (\phi \wedge \psi), (\phi \vee \psi), (\phi \rightarrow \psi) \in \mathrm{Fml}(V)$*.*
3. *If* $\phi \in \mathrm{Fml}(V)$ *and* $x \in V$*, then* $\forall x\, \phi, \exists x\, \phi \in \mathrm{Fml}(V)$*.*
4. *If* $\pi \in \mathrm{Fml}_T(V)$ *and* $\alpha \in \mathrm{HP}(V)$*, then* $[\alpha]\pi, \langle\alpha\rangle\pi \in \mathrm{Fml}(V)$*.*

The set $\mathrm{Fml}_T(V)$ *of* trace formulas *is the smallest set with:*

1. *If* $\phi \in \mathrm{Fml}(V)$*, then* $\phi \in \mathrm{Fml}_T(V)$*.*
2. *If* $\phi \in \mathrm{Fml}(V)$*, then* $\Box\phi, \Diamond\phi \in \mathrm{Fml}_T(V)$*.*

Formulas without $\Box$ and $\Diamond$, i.e., without case 2 of the trace formulas, are called *non-temporal* $\mathsf{d}\mathcal{L}$ *formulas* [19, 22]. Unlike in CTL, state formulas are true on a trace (case 1) if they hold for the *last* state of a trace, not for the first. Thus, $[\alpha]\phi$ expresses that ϕ is true at the end of each trace of α. In contrast, $[\alpha]\Box\phi$ expresses that ϕ is true all along all states of every trace of α. This combination gives a smooth embedding of non-temporal $\mathsf{d}\mathcal{L}$ into dTL and makes it possible to define a compositional calculus. Like CTL, dTL allows nesting with a branching time semantics [10], e.g., $[\alpha]\Box(x \geq 2 \rightarrow \langle\gamma\rangle\Diamond x \leq 0)$.

Hybrid Programs. The hybrid programs [19, 20, 22] occurring in dynamic modalities of dTL are built from elementary discrete jumps and continuous evolutions using a regular control structure [13].

Definition 2 (Hybrid programs). *The set* $\mathrm{HP}(V)$ *of* hybrid programs *is inductively defined as the smallest set such that:*

1. *If* $x \in V$ *and* $\theta \in \mathrm{Trm}(V)$*, then* $(x := \theta) \in \mathrm{HP}(V)$*.*
2. *If* $x \in V$ *and* $\theta \in \mathrm{Trm}(V)$*, then* $(\dot{x} = \theta) \in \mathrm{HP}(V)$*.*
3. *If* $\chi \in \mathrm{Fml}(V)$ *is quantifier-free and first-order, then* $(?\chi) \in \mathrm{HP}(V)$*.*
4. *If* $\alpha, \gamma \in \mathrm{HP}(V)$ *then* $(\alpha \cup \gamma) \in \mathrm{HP}(V)$*.*
5. *If* $\alpha, \gamma \in \mathrm{HP}(V)$ *then* $(\alpha; \gamma) \in \mathrm{HP}(V)$*.*
6. *If* $\alpha \in \mathrm{HP}(V)$ *then* $(\alpha^*) \in \mathrm{HP}(V)$*.*

The effect of $x := \theta$ is an instantaneous discrete jump in state space or a mode switch. That of $\dot{x} = \theta$ is an ongoing continuous evolution regulated by the differential equation with time-derivative $\dot{x}$ of x and term θ (accordingly for systems of differential equations).

Test actions $?\chi$ are used to define conditions. Their semantics is that of a no-op if χ is true in the current state, and that of a dead end operator aborting any further evolution, otherwise. The sequential composition $\alpha; \gamma$, non-deterministic choice $\alpha \cup \gamma$, and non-deterministic repetition α^* of system actions are as usual [13]. They can be combined with $?\chi$ to form other control structures [13].

In dTL, there is no need to distinguish between discrete and continuous variables or between system parameters and state variables, as they share the same uniform semantics. For pragmatic reasons, an informal distinction can nevertheless improve readability. For instance, $\exists x\, [\dot{x} = -x]x \leq 5$ expresses that there is a choice of the initial value for x (which could be a parameter) such that after all evolutions along $\dot{x} = -x$, the outcome of the state variable x will be at most 5.

2.3 Trace Semantics of dTL

In standard dynamic logic [13] and d$\mathcal{L}$ [22, 19], modalities only refer to the final states of system runs and the semantics is a reachability relation on states: State ω is reachable from state ν using α if there is a run of α which terminates in ω when started in ν. For dTL, however, formulas can refer to intermediate states of runs as well. Thus, the semantics of a hybrid system α is the set of its possible *traces*, i.e., successions of states that occur during the evolution of α.

States contain values of system variables during a hybrid evolution. A *state* is a map $\nu : V \to \mathbb{R}$; the set of all states is denoted by $\mathrm{Sta}(V)$. In addition, we distinguish a state Λ to denote the failure of a system run when it is *aborted* due to a test $?\chi$ that yields *false*. In particular, Λ can only occur at the end of an aborted system run and marks that there is no further extension.

Hybrid systems evolve along piecewise continuous traces in multi-dimensional space as time passes. Continuous phases are governed by differential equations, whereas discontinuities are caused by discrete jumps in state space. Unlike in discrete cases [4, 25], traces are not just sequences of states, since hybrid systems pass through uncountably many states even in bounded time. Beyond that, continuous changes are more involved than in pure real-time [1, 15], because all variables can evolve along different differential equations. Generalising the real-time traces of [15], the following definition captures hybrid behaviour by splitting the uncountable succession of states into periods σ_i that are regulated by the same control law. For discrete jumps, some periods are point flows of duration 0.

Definition 3 (Hybrid Trace). *A* trace *is a (non-empty) finite or infinite sequence $\sigma = (\sigma_0, \sigma_1, \sigma_2, \dots)$ of functions $\sigma_i : [0, r_i] \to \mathrm{Sta}(V)$ with respective durations $r_i \in \mathbb{R}$ (for $i \in \mathbb{N}$). A* position *of σ is a pair (i, ζ) with $i \in \mathbb{N}$ and ζ in the interval $[0, r_i]$; the state of σ at (i, ζ) is $\sigma_i(\zeta)$. Positions of σ are ordered lexicographically by $(i, \zeta) \prec (j, \xi)$ iff either $i < j$, or $i = j$ and $\zeta < \xi$. Further, for a state $\nu \in \mathrm{Sta}(V)$, $\hat{\nu} : 0 \mapsto \nu$ is the* point flow *at ν with duration 0. A trace* terminates *if it is a finite sequence $(\sigma_0, \sigma_1, \dots, \sigma_n)$ and $\sigma_n(r_n) \neq \Lambda$. In that case, the last state* $\mathrm{last}\,\sigma$ *is denoted as $\sigma_n(r_n)$. The first state* $\mathrm{first}\,\sigma$ *is $\sigma_0(0)$.*

Unlike in [1, 15], the definition of traces also admits finite traces of bounded duration, which is necessary for compositionality of traces in $\alpha; \gamma$. The semantics of hybrid programs α as the set $\tau(\alpha)$ of its possible traces depends on valuations $val(\nu, \cdot)$ of formulas and terms at intermediate states ν. The valuation of terms [13], and interpretations of function and predicate symbols are as usual for real arithmetic. The valuation of formulas will be defined in Definition 5. We use $\nu[x \mapsto d]$ to denote the *modification* that agrees with state ν on all variables except for the symbol x, which is changed to $d \in \mathbb{R}$.

Definition 4 (Trace semantics of hybrid programs). *The* trace semantics, $\tau(\alpha)$, *of a hybrid* program α*, is the set of all its possible hybrid traces and is defined as follows:*

1. $\tau(x := \theta) = \{(\hat{\nu}, \hat{\omega}) \ : \ \omega = \nu[x \mapsto val(\nu, \theta)]$ *for* $\nu \in \mathrm{Sta}(V)\}$
2. $\tau(\dot{x} = \theta) = \{(f) \ : \ 0 \leq r \in \mathbb{R}$ *and* $f : [0, r] \to \mathrm{Sta}(V)$ *is such that the function* $val(f(\zeta), x)$ *is continuous in* ζ *on* $[0, r]$ *and has a derivative of value*

$val(f(\zeta), \theta)$ at each $\zeta \in (0, r)$. Variables without a differential equation do not change$\}$

3. $\tau(?\chi) = \{(\hat{\nu}) \ : \ val(\nu, \chi) = true\} \cup \{(\hat{\nu}, \hat{\Lambda}) \ : \ val(\nu, \chi) = false\}$
4. $\tau(\alpha \cup \gamma) = \tau(\alpha) \cup \tau(\gamma)$
5. $\tau(\alpha; \gamma) = \{\sigma \circ \varsigma \ : \ \sigma \in \tau(\alpha)\,, \varsigma \in \tau(\gamma)$ *when* $\sigma \circ \varsigma$ *is defined*$\}$*; the composition of* $\sigma = (\sigma_0, \sigma_1, \sigma_2, \dots)$ *and* $\varsigma = (\varsigma_0, \varsigma_1, \varsigma_2, \dots)$ *is*

$$\sigma \circ \varsigma = \begin{cases} (\sigma_0, \dots, \sigma_n, \varsigma_0, \varsigma_1, \dots) & \text{if } \sigma \text{ terminates at } \sigma_n \text{ and } \operatorname{last}\sigma = \operatorname{first}\varsigma \\ \sigma & \text{if } \sigma \text{ does not terminate} \\ \text{not defined} & \text{otherwise} \end{cases}$$

6. $\tau(\alpha^*) = \bigcup_{n \in \mathbb{N}} \tau(\alpha^n)$, *where* $\alpha^{n+1} = (\alpha^n; \alpha)$ *for* $n \geq 1$, *and* $\alpha^0 = (?true)$.

Time passes differently during discrete and continuous change. During continuous evolution, the discrete step index i of positions (i, ζ) remains constant, whereas the continuous duration ζ remains 0 during discrete point flows. This permits multiple discrete state changes to happen at the same (super-dense) continuous time, unlike in [1].

Definition 5 (Valuation of formulas). *The valuation of state and trace formulas is defined respectively. For state formulas, the* valuation $val(\nu, \cdot)$ *with respect to state* ν *is defined as follows:*

1. $val(\nu, p(\theta_1, \dots, \theta_n)) = p^\ell\big(val(\nu, \theta_1), \dots, val(\nu, \theta_n)\big)$, *where* p^ℓ *is the relation associated to* p.
2. $val(\nu, \phi \wedge \psi)$ *is defined as usual, the same holds for* $\neg, \vee, \rightarrow$.
3. $val(\nu, \forall x\, \phi) = true :\iff val(\nu[x \mapsto d], \phi) = true$ *for all* $d \in \mathbb{R}$
4. $val(\nu, \exists x\, \phi) = true :\iff val(\nu[x \mapsto d], \phi) = true$ *for some* $d \in \mathbb{R}$
5. $val(\nu, [\alpha]\pi) = true :\iff$ *for each trace* $\sigma \in \tau(\alpha)$ *that starts in* $\operatorname{first}\sigma = \nu$, *if* $val(\sigma, \pi)$ *is defined, then* $val(\sigma, \pi) = true$.
6. $val(\nu, \langle\alpha\rangle\pi) = true :\iff$ *there is a trace* $\sigma \in \tau(\alpha)$ *starting in* $\operatorname{first}\sigma = \nu$, *such that* $val(\sigma, \pi) = true$.

For trace formulas, the valuation $val(\sigma, \cdot)$ *with respect to trace* σ *is:*

1. *If* ϕ *is a state formula, then* $val(\sigma, \phi) = val(\operatorname{last}\sigma, \phi)$ *if* σ *terminates, whereas* $val(\sigma, \phi)$ *is not defined if* σ *does not terminate.*
2. $val(\sigma, \Box\phi) = true :\iff val(\sigma_i(\zeta), \phi) = true$ *for all positions* (i, ζ) *of* σ *with* $\sigma_i(\zeta) \neq \Lambda$.
3. $val(\sigma, \Diamond\phi) = true :\iff val(\sigma_i(\zeta), \phi) = true$ *for some position* (i, ζ) *of* σ *with* $\sigma_i(\zeta) \neq \Lambda$.

As usual, a (state) formula is *valid* if it is true in all states.

2.4 Conservative Temporal Extension

The following result shows that the extension of dTL by temporal operators does not change the meaning of non-temporal $\mathsf{d}\mathcal{L}$ formulas. The trace semantics given in Definition 5 is equivalent to the final state reachability relation semantics [22, 19] for the sublogic $\mathsf{d}\mathcal{L}$ of dTL. A proof for this can be found in [21].

Proposition 1. *The logic* dTL *is a* conservative extension *of non-temporal* d$\mathcal{L}$*, i.e., the set of valid* d$\mathcal{L}$*-formulas is the same with respect to transition reachability semantics of* d$\mathcal{L}$ *[22, 19] as with respect to the trace semantics of* dTL *(Definition 5).*

3 Safety Invariants in Train Control

In the European Train Control System (ETCS) [12], trains are coordinated by decentralised Radio Block Centres (RBC), which grant or deny movement authorities (MA) to the individual trains by wireless communication. In emergencies, trains always have to stop within the MA issued by the RBC, see Fig. 1. Following the reasoning pattern for traffic agents in [7], each train negotiates with the RBC to extend its MA when approaching the end, say m, of its current MA. Since wireless communication takes time, this negotiation is initiated in due time before reaching m. During negotiation, trains are assumed to keep their desired speed as in [7]. Before entering negotiation at some point ST, the train still has sufficient distance to MA (it is in *far* mode) and can regulate its speed freely within the track limits.

Depending on weather conditions, slope of track etc., the local train motion control determines a safety envelope s around the train, within which it considers driving safe, and adjusts its acceleration a in accordance with s (called *correction* [7]). In particular, depending on the maximum RBC response time, this determines the latest point, SB, on the track where a response from the RBC must have arrived to guarantee safe driving.

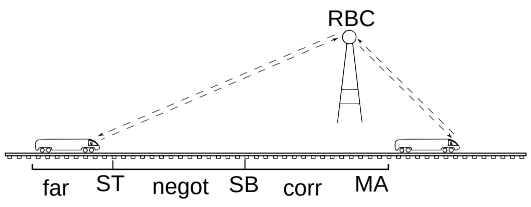


Fig. 1. ETCS train coordination by movement authorities

As a model for train movements, we use the ideal-world model adapted from [7]. It does not model friction, slopes, or mass of train but is perfectly suitable for analysing the cooperation level of train control [7]. The local safety properties that where used when verifying the cooperation protocol can then be shown for more detailed models of individual components.

For a safe operation of multiple traffic agents, it is crucial that the MA is respected at *every* point in time during this protocol, not only at its end. Hence, we need to consider temporal safety invariants. For instance, when the train has entered the negotiation phase at its current position z, dTL can analyse the following safety invariant of a part of the train controller:

$$\begin{aligned}
\psi \;&\rightarrow\; [\mathit{negot}; \mathit{corr}; \dot{z} = v, \dot{v} = a]\Box(\ell \leq L \rightarrow z < m) && (1)\\
\text{where } \mathit{negot} \;&\equiv\; \dot{z} = v, \dot{\ell} = 1\\
\mathit{corr} \;&\equiv\; (?m - z < s; a := -b) \cup (?m - z \geq s; a := \dots)\ .
\end{aligned}$$

It expresses that—under a sanity condition ψ for parameters—a train will *always* remain within its MA m, as long as the accumulated RBC negotiation latency ℓ is at most L. We refer to [12] for details on what contributes to ℓ. Like in [7], we model the train to first negotiate while keeping a constant speed ($\dot{z} = v$) in *negot*. Thereafter, in *corr*, the train corrects its acceleration or brakes with force b (as a failsafe recovery manoeuvre) on the basis of the remaining distance ($m - z$). Finally, the train continues moving according to the system ($\dot{z} = v, \dot{v} = a$) or, equivalently, $\ddot{z} = a$. Instead of manually choosing specific values for the free parameters of (1) as in [7,12], we will use the techniques developed in this paper to automatically synthesise constraints on the relationship of parameters that are required for a safe operation of cooperative train control.

4 A Verification Calculus for Safety Invariants

In this section, we introduce a sequent calculus for verifying temporal specifications of hybrid systems in dTL. With the basic idea being to perform a symbolic decomposition, hybrid programs are successively transformed into simpler logical formulas describing their effects. There, statements about the temporal behaviour of a hybrid program are successively reduced to corresponding non-temporal statements about the intermediate states.

For propositional logic, standard rules P1–P9 are listed in Fig. 2. The rule P10 is a shortcut to handle quantifiers of first-order real arithmetic, which is decidable. We use P10 as a modular interface to arithmetic and refer to [19] for a goal-oriented integration of arithmetic, which combines with dTL. Rules D1–D8 work similar to those in [13,3]. For handling discrete change, D8 inductively uses substitutions. D9–D10 handle continuous evolutions given a first-order definable flow y_x. In particular, in conjunction with P10, they fully encapsulate handling of differential equations within hybrid systems.

Rules T1–T10 successively transform temporal specifications of hybrid programs into non-temporal $\mathsf{d}\mathcal{L}$ formulas. The idea underlying this transformation is to decompose hybrid programs and recursively augment intermediate state transitions with appropriate specifications. D1–D2 are identical for dTL and $\mathsf{d}\mathcal{L}$ specifications, hence they apply for all trace formulas π and not just for state formulas. Rules for handling $[\alpha]\Diamond\phi$ and $\langle\alpha\rangle\Box\phi$ are discussed in Sect. 6.

4.1 Rules of the Calculus

A *sequent* is of the form $\Gamma \vdash \Delta$, where Γ and Δ are finite sets of formulas. Its semantics is that of the formula $\bigwedge_{\phi\in\Gamma} \phi \;\rightarrow\; \bigvee_{\psi\in\Delta} \psi$ and will be treated as an

$$\text{(P1)}\ \frac{\vdash \phi}{\neg\phi \vdash} \qquad \text{(P4)}\ \frac{\phi, \psi \vdash}{\phi \land \psi \vdash} \qquad \text{(P7)}\ \frac{\phi \vdash \qquad \psi \vdash}{\phi \lor \psi \vdash}$$

$$\text{(P2)}\ \frac{\phi \vdash}{\vdash \neg\phi} \qquad \text{(P5)}\ \frac{\vdash \phi \qquad \vdash \psi}{\vdash \phi \land \psi} \qquad \text{(P8)}\ \frac{\vdash \phi, \psi}{\vdash \phi \lor \psi}$$

$$\text{(P3)}\ \frac{\phi \vdash \psi}{\vdash \phi \to \psi} \qquad \text{(P6)}\ \frac{\vdash \phi \qquad \psi \vdash}{\phi \to \psi \vdash} \qquad \text{(P9)}\ \frac{}{\phi \vdash \phi}$$

$$\text{(P10)}\ \frac{F_0 \vdash G_0}{F \vdash G}$$

$$\text{(D1)}\ \frac{\langle\alpha\rangle\pi \lor \langle\gamma\rangle\pi}{\langle\alpha \cup \gamma\rangle\pi} \qquad \text{(D6)}\ \frac{\phi \lor \langle\alpha; \alpha^*\rangle\phi}{\langle\alpha^*\rangle\phi}$$

$$\text{(D2)}\ \frac{[\alpha]\pi \land [\gamma]\pi}{[\alpha \cup \gamma]\pi} \qquad \text{(D7)}\ \frac{\phi \land [\alpha; \alpha^*]\phi}{[\alpha^*]\phi}$$

$$\text{(D3)}\ \frac{\langle\![\alpha]\!\rangle\langle\![\gamma]\!\rangle\phi}{\langle\![\alpha; \gamma]\!\rangle\phi} \qquad \text{(D8)}\ \frac{F_x^\theta}{\langle\![x := \theta]\!\rangle F}$$

$$\text{(D4)}\ \frac{\chi \land \phi}{\langle ?\chi\rangle\phi} \qquad \text{(D9)}\ \frac{\exists t {\geq} 0\, \langle x := y_x(t)\rangle\phi}{\langle \dot{x} = \theta\rangle\phi}$$

$$\text{(D5)}\ \frac{\chi \to \phi}{[?\chi]\phi} \qquad \text{(D10)}\ \frac{\forall t {\geq} 0\, [x := y_x(t)]\phi}{[\dot{x} = \theta]\phi}$$

$$\text{(T1)}\ \frac{[\alpha]\Box\phi \land [\alpha][\gamma]\Box\phi}{[\alpha; \gamma]\Box\phi} \qquad \text{(T6)}\ \frac{\langle\alpha\rangle\Diamond\phi \lor \langle\alpha\rangle\langle\gamma\rangle\Diamond\phi}{\langle\alpha; \gamma\rangle\Diamond\phi}$$

$$\text{(T2)}\ \frac{\phi}{[?\chi]\Box\phi} \qquad \text{(T7)}\ \frac{\phi}{\langle ?\chi\rangle\Diamond\phi}$$

$$\text{(T3)}\ \frac{\phi \land [x := \theta]\phi}{[x := \theta]\Box\phi} \qquad \text{(T8)}\ \frac{\phi \lor \langle x := \theta\rangle\phi}{\langle x := \theta\rangle\Diamond\phi}$$

$$\text{(T4)}\ \frac{[\dot{x} = \theta]\phi}{[\dot{x} = \theta]\Box\phi} \qquad \text{(T9)}\ \frac{\langle \dot{x} = \theta\rangle\phi}{\langle \dot{x} = \theta\rangle\Diamond\phi}$$

$$\text{(T5)}\ \frac{[\alpha; \alpha^*]\Box\phi}{[\alpha^*]\Box\phi} \qquad \text{(T10)}\ \frac{\langle\alpha; \alpha^*\rangle\Diamond\phi}{\langle\alpha^*\rangle\Diamond\phi}$$

In these rules, ϕ and ψ are (state) formulas, whereas π is a trace formula. Unlike ϕ and ψ, the trace formula π may thus begin with $\Box$ or $\Diamond$. In D8, F is a first-order formula and the substitution of F_x^θ, which replaces x by θ in F, does not introduce new bindings. In D9–D10, t is a fresh variable and y_v the solution of the initial value problem $(\dot{x} = \theta, x(0) = v)$. In P10, $\mathrm{Cl}_\forall\,(F_0 \to G_0) \to \mathrm{Cl}_\forall\,(F \to G)$ is an instance of a first-order tautology of real arithmetic and $\mathrm{Cl}_\forall$ the universal closure.

Fig. 2. Rule schemata of the temporal dynamic dTL verification calculus

abbreviation. In the following, an *update* $\mathcal{U}$ is a list of discrete assignments of the form $x := \theta$ (see [3] for advanced update techniques, which can be combined with our calculus).

Definition 6 (Provability, derivability). *A formula ψ is* provable *from a set Φ of formulas, denoted by $\Phi \vdash_{\mathrm{dTL}} \psi$ iff there is a finite set $\Phi_0 \subseteq \Phi$ for which the sequent $\Phi_0 \vdash \psi$ is derivable. In turn, a sequent of the form $\Gamma, \langle \mathcal{U} \rangle \Phi \vdash \langle \mathcal{U} \rangle \Psi, \Delta$ (for some update $\mathcal{U}$, including the empty update, and finite sets Γ, Δ of context formulas) is* derivable *iff there is an instance*

$$\frac{\Phi_1 \vdash \Psi_1 \qquad \dots \qquad \Phi_n \vdash \Psi_n}{\Phi \vdash \Psi}$$

of a rule schema of the dTL *calculus in Fig. 2 such that*

$$\Gamma, \langle \mathcal{U} \rangle \Phi_i \vdash \langle \mathcal{U} \rangle \Psi_i, \Delta$$

is derivable for each $1 \leq i \leq n$. Moreover, the symmetric schemata Di and Ti can be applied on either side of the sequent (in context Γ, Δ and update $\langle \mathcal{U} \rangle$). The schematic modality $[\![\cdot]\!\rangle$ can be instantiated with both $[\cdot]$ and $\langle \cdot \rangle$ in all rule schemata. The same modality instance has to be chosen within a single schema instantiation, though.

As usual in sequent calculus—although the direction of entailment is from premisses (above rule bar) to conclusion (below)—the order of reasoning is *goal-directed*: Rules are applied starting from the desired conclusion at the bottom (goal) to the premisses (sub-goals).

Rule T1 decomposes invariants of $\alpha; \gamma$ into an invariant of α and an invariant of γ that holds when γ is started in *any* final state of α. T3 expresses that invariants of assignments need to hold before and after the discrete change (similarly for T2, except that tests do not lead to a state change). T4 can directly reduce invariants of continuous evolutions to non-temporal formulas as restrictions of solutions of differential equations are themselves solutions of different duration. T5 relies on T1 and is simpler than D7, because the other rules will inductively produce a premiss that ϕ holds in the current state. The dual rules T6–T10 work similarly. The usual induction schemes [13, 17] can be added to the dTL calculus. Inductive invariant properties can be handled by augmenting induction rules with an additional branch that takes care of the temporal properties.

4.2 Soundness and Incompleteness

The following result shows that verification with the dTL calculus always produces correct results about safety of hybrid systems, i.e., the dTL calculus is sound.

Theorem 1 (Soundness). *The* dTL *calculus is* sound*, i.e., derivable (state) formulas are valid. (See [21] for a proof.)*

Theorem 2 (Incompleteness). *Fragments of* dTL *are inherently incomplete, i.e. cannot have a complete calculus. (See [21] for a proof.)*

5 Verification of Train Control Safety Invariants

Continuing the ETCS study from Sect. 3, we consider a slightly simplified version of equation (1) that gives a more concise proof. By a safe abstraction (provable in dTL), we simplify *corr* to permit braking even when $m - z \geq s$, since braking remains safe with respect to $z < m$. We use the following abbreviations in addition to (1):

$$\begin{aligned} \psi &\equiv z < m \land v > 0 \land \ell = 0 \land L \geq 0 \\ \phi &\equiv \ell \leq L \to z < m \\ corr &\equiv a := -b \cup (?m - z \geq s; a := \dots) \ . \end{aligned}$$

Within the following proof, $\langle\!|\,|\!\rangle$ brackets are used instead of modalities to visually identify the update prefix (Definition 6). To give shorter formulas, we generalise update application D8 to work within quantifiers according to [3]. The dTL proof of the safety invariant in (1) splits into two cases as follows:

$$\begin{array}{rc}
 & \cfrac{\dots}{\psi \vdash [negot]\Box\phi} \qquad\qquad \cfrac{\dots}{\psi \vdash [negot][corr; \dot{z} = v, \dot{v} = a]\Box\phi} \\ \cline{2-2}
\text{T1} & \psi \vdash [negot; corr; \dot{z} = v, \dot{v} = a]\Box\phi \\ \cline{2-2}
\text{P3} & \vdash \psi \to [negot; corr; \dot{z} = v, \dot{v} = a]\Box\phi
\end{array}$$

There, the left branch proves that ϕ holds while negotiating and is as follows:

$$\begin{array}{rl}
 & \psi \vdash Lv + z < m \\ \cline{2-2}
\text{P10} & \psi \vdash \forall l{\geq}0\, (l \leq L \to lv + z < m) \\ \cline{2-2}
\text{D8} & \psi \vdash \forall l{\geq}0\, \langle\!| z := lv + z, \ell := l |\!\rangle \phi \\ \cline{2-2}
\text{D10} & \psi \vdash [negot]\phi \\ \cline{2-2}
\text{T4} & \psi \vdash [negot]\Box\phi
\end{array}$$

The right branch shows that ϕ continues to hold after negotiation has completed when continuing with an adjusted acceleration a:

$$\begin{array}{rl}
 & \psi, \ell{\geq}0 \vdash v^2 < 2b(m - Lv - z) \land Lv + z < m \\ \cline{2-2}
\text{P10} & \psi, \ell{\geq}0 \vdash \langle\!| z := \ell v{+}z, a := \text{-}b |\!\rangle \forall t{\geq}0\, (\ell{\leq}L \to \frac{a}{2}t^2{+}vt{+}z{<}m) \\ \cline{2-2}
\text{D8} & \psi, \ell{\geq}0 \vdash \langle\!| z := \ell v{+}z, a := \text{-}b |\!\rangle \forall t{\geq}0\, \langle\!| z := \frac{a}{2}t^2{+}vt{+}z |\!\rangle \phi) \\ \cline{2-2}
\text{T4,D10} & \psi, \ell{\geq}0 \vdash \langle\!| z := \ell v{+}z, a := \text{-}b |\!\rangle [\dot{z} = v, \dot{v} = a]\Box\phi \qquad \triangleright \\ \cline{2-2}
\text{D2} & \psi, \ell{\geq}0 \vdash \langle\!| z := \ell v{+}z |\!\rangle [corr][\dot{z} = v, \dot{v} = a]\Box\phi \qquad \triangleright \\ \cline{2-2}
\text{T1} & \psi, \ell{\geq}0 \vdash \langle\!| z := \ell v{+}z |\!\rangle [corr; \dot{z} = v, \dot{v} = a]\Box\phi \\ \cline{2-2}
\text{P3} & \quad \psi \vdash \ell{\geq}0 \to \langle\!| z := \ell v{+}z |\!\rangle [corr; \dot{z} = v, \dot{v} = a]\Box\phi \\ \cline{2-2}
\text{P10} & \quad \psi \vdash \forall \ell{\geq}0\, \langle\!| z := \ell v{+}z |\!\rangle [corr; \dot{z} = v, \dot{v} = a]\Box\phi \\ \cline{2-2}
\text{D10} & \quad \psi \vdash [negot][corr; \dot{z} = v, \dot{v} = a]\Box\phi
\end{array}$$

The application of T1 in this latter case spawns a third case (marked with $\triangleright$) to show that ϕ holds during *corr*. However, the reasoning in this third case is subsumed by the cases above, since the changes on a in *corr* do not interfere with condition ϕ. Generally, this optimisation of T1 is applicable whenever the modified vocabulary is disjoint from ϕ. Here, D10 and P10 are implemented in Mathematica to handle evolutions [19].

The leaves of the proof branches above can even be used to automatically *synthesise parameter constraints* that are necessary to avoid MA violation. The parametric safety constraint obtained by combining the open conditions conjunctively is $Lv + z < m \wedge v^2 < 2b(m - Lv - z)$. It simplifies to $v^2 < 2b(m - Lv - z)$ as $b > 0$. This yields bounds for the speed limit and negotiation latency in order to guarantee safe driving and closing of the proof. Similarly, D2 leads to a branch for the case $[?m - z \geq s; a := \ldots]$, from which corresponding conditions about the safety envelope s can be derived depending on the particular speed controller. Yet, this is beyond the scope of this paper.

6 Liveness by Quantifier Alternation

Liveness specifications of the form $[\alpha]\Diamond\phi$ or $\langle\alpha\rangle\Box\phi$ are sophisticated (Σ_1^1-hard because they can express infinite occurrence in Turing machines). Beckert and Schlager [4] say they failed to find sound rules for a discrete case that corresponds to $[\alpha; \gamma]\Diamond\phi$.

For *finitary liveness semantics*, we accomplish this as follows. In this section, we modify the meaning of $[\alpha]\Diamond\phi$ to refer to all *terminating* traces of α. Then, the straightforward generalisation T11 in Fig. 3 is sound, even in the hybrid case(see [21] for proofs). But T11 still leads to an incomplete axiomatisation as it does not cover the case where, in some traces, ϕ becomes true at some point during α, and in other traces, ϕ only becomes true during γ. To overcome this limitation, we use a program transformation approach. We instrument the hybrid program to monitor the occurrence of ϕ during all changes: In T12, $\breve{\alpha}$ results from replacing all occurrences of $x := \theta$ by $x := \theta; ?\phi \rightarrow t = 1$ and $\dot{x} = \theta$ by $\dot{x} = \theta \,\&\, (\phi \rightarrow t = 1)$. The latter denotes continuous evolution restricted to the region of the state space that satisfies $\phi \rightarrow t = 1$ (see [19] for details). The effect is that t detects whether ϕ has occurred during any change in α. In particular, t is guaranteed to be 1 after all runs, if ϕ occurs at least once along all traces of α. This trick directly works for quantifier-free first-order conditions ϕ. Using the combination presented in [22], nominals can be used as state labels to address the same issue for general ϕ.

$$\text{(T11)}\ \frac{\vdash [\alpha]\Diamond\phi, [\alpha][\gamma]\Diamond\phi}{\vdash [\alpha; \gamma]\Diamond\phi} \qquad \text{(T12)}\ \frac{\phi \vee \forall t\, [\breve{\alpha}]t = 1}{[\alpha]\Diamond\phi}$$

Fig. 3. Transformation rules for alternating temporal path and trace quantifiers

7 Conclusions and Future Work

For reasoning about hybrid systems, we have introduced a temporal dynamic logic, dTL, with modal path quantifiers over traces and temporal quantifiers along the traces. It combines the capabilities of dynamic logic [13] to reason about possible system behaviour with the power of temporal logic [24, 10, 11] in reasoning about the behaviour along traces. Furthermore, we have presented a calculus for verifying temporal safety specifications of hybrid programs in dTL.

Our sequent calculus for dTL is a modular combination of temporal and non-temporal reasoning. Temporal formulas are handled using rules that augment intermediate state transitions with corresponding sub-specifications. Purely non-temporal rules handle the effects of discrete and continuous evolution.

As an example, we demonstrate that our logic is suitable for reasoning about safety invariants in the European Train Control System [12]. Further, we have successfully applied our calculus to automatically synthesise (non-linear) parametric safety constraints for this system.

We are currently extending our preliminary verification tool for parametric hybrid systems to cover the full dTL calculus. Future work includes extending dTL with CTL*-like [11] formulas of the form $[\alpha](\psi \wedge \Box\phi)$ to avoid splitting of the proof into two very similar sub-proofs for temporal parts $[\alpha]\Box\phi$ and non-temporal parts $[\alpha]\psi$ arising in T1. Our combination of temporal logic with dynamic logic is more suitable for this purpose than the approach in [4], since dTL has uniform modalities and uniform semantics for temporal and non-temporal specifications. This extension will also simplify the treatment of alternating liveness quantifiers conceptually.

References

1. R. Alur, C. Courcoubetis, and D. L. Dill. Model-checking for real-time systems. In *LICS*, pages 414–425. IEEE Computer Society, 1990.
2. B. Beckert, R. Hähnle, and P. H. Schmitt, editors. *Verification of Object-Oriented Software: The KeY Approach*, volume 4334 of *LNCS*. Springer-Verlag, 2007.
3. B. Beckert and A. Platzer. Dynamic logic with non-rigid functions: A basis for object-oriented program verification. In U. Furbach and N. Shankar, editors, *IJCAR*, volume 4130 of *LNCS*, pages 266–280. Springer, 2006.
4. B. Beckert and S. Schlager. A sequent calculus for first-order dynamic logic with trace modalities. In R. Goré, A. Leitsch, and T. Nipkow, editors, *IJCAR*, volume 2083 of *LNCS*, pages 626–641. Springer, 2001.
5. A. Bemporad, A. Bicchi, and G. Buttazzo, editors. *Hybrid Systems: Computation and Control, 10th International Conference, HSCC 2007, Pisa, Italy, Proceedings*, volume 4416 of *LNCS*. Springer, 2007.
6. E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking.* MIT Press, Cambridge, MA, USA, 1999.
7. W. Damm, H. Hungar, and E.-R. Olderog. On the verification of cooperating traffic agents. In F. S. de Boer, M. M. Bonsangue, S. Graf, and W. P. de Roever, editors, *FMCO*, volume 3188 of *LNCS*, pages 77–110. Springer, 2003.
8. J. M. Davoren, V. Coulthard, N. Markey, and T. Moor. Non-deterministic temporal logics for general flow systems. In R. Alur and G. J. Pappas, editors, *HSCC*, volume 2993 of *LNCS*, pages 280–295. Springer, 2004.
9. J. M. Davoren and A. Nerode. Logics for hybrid systems. *Proceedings of the IEEE*, 88(7):985–1010, July 2000.
10. E. A. Emerson and E. M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Sci. Comput. Program.*, 2(3):241–266, 1982.
11. E. A. Emerson and J. Y. Halpern. "Sometimes" and "Not Never" revisited: on branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986.

12. J. Faber and R. Meyer. Model checking data-dependent real-time properties of the European Train Control System. In *FMCAD*, pages 76–77. IEEE Computer Society Press, Nov 2006.
13. D. Harel, D. Kozen, and J. Tiuryn. *Dynamic logic*. MIT Press, 2000.
14. T. A. Henzinger. The theory of hybrid automata. In *LICS*, pages 278–292, 1996.
15. T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. In *LICS*, pages 394–406. IEEE Computer Society, 1992.
16. D. Hutter, B. Langenstein, C. Sengler, J. H. Siekmann, W. Stephan, and A. Wolpers. Deduction in the verification support environment (VSE). In M.-C. Gaudel and J. Woodcock, editors, *FME*, volume 1051 of *LNCS*, pages 268–286. Springer, 1996.
17. D. Leivant. Partial correctness assertions provable in dynamic logics. In I. Walukiewicz, editor, *FoSSaCS*, volume 2987 of *LNCS*, pages 304–317. Springer, 2004.
18. V. Mysore, C. Piazza, and B. Mishra. Algorithmic algebraic model checking II: Decidability of semi-algebraic model checking and its applications to systems biology. In D. Peled and Y.-K. Tsay, editors, *ATVA*, volume 3707 of *LNCS*, pages 217–233. Springer, 2005.
19. A. Platzer. Differential dynamic logic for verifying parametric hybrid systems. 2007.
20. A. Platzer. Differential logic for reasoning about hybrid systems. In Bemporad et al. [5], pages 746–749.
21. A. Platzer. A temporal dynamic logic for verifying hybrid system invariants. Reports of SFB/TR 14 AVACS 12, February 2007. ISSN: 1860-9821, available at http://www.avacs.org.
22. A. Platzer. Towards a hybrid dynamic logic for hybrid dynamic systems. In P. Blackburn, T. Bolander, T. Braüner, V. de Paiva, and J. Villadsen, editors, *Proc., LICS International Workshop on Hybrid Logic, 2006, Seattle, USA*, ENTCS, 2007.
23. A. Platzer and E. M. Clarke. The image computation problem in hybrid systems model checking. In Bemporad et al. [5], pages 473–486.
24. A. Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57. IEEE, 1977.
25. V. R. Pratt. Process logic. In *POPL*, pages 93–100, 1979.
26. C. Zhou, A. P. Ravn, and M. R. Hansen. An extended duration calculus for hybrid real-time systems. In R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, editors, *Hybrid Systems*, volume 736 of *LNCS*, pages 36–59. Springer, 1992.

Multiplexor Categories and Models of Soft Linear Logic

Brian F. Redmond

Department of Mathematics & Statistics, University of Ottawa,
585 King Edward Avenue, Ottawa ON K1N 6N5, Canada
bredm066@uottawa.ca

Abstract. We give a categorical interpretation of Lafont's Soft Linear Logic, a logical system complete for polynomial time computation, in terms of multiplexor categories. We present two main classes of models and methods for constructing examples in each case. As a concrete example of the first type, we introduce a simple game semantics for Multiplicative Soft Linear Logic. To illustrate the second type, we give a realizability model and a new proof of the polytime soundness of Soft Linear Logic. These results further our semantic understanding of Soft Linear Logic and polynomial time.

Keywords: Categorical semantics, Soft Linear Logic, Polynomial time, Game semantics.

1 Introduction

Linear Logic was introduced by J.-Y. Girard in 1987 [6] as a logical system for managing resources and it has attracted considerable attention from logicians and computer scientists. Its novel handling of infinite resources makes Linear Logic ideal for studying aspects of time bounded computation directly within a logical system. By a careful consideration of the exponential fragment, researchers have be able to isolate subsystems complete for polynomial and elementary time computation. The first such system, Bounded Linear Logic (BLL), introduced by Girard, Scedrov and Scott, is complete for polynomial time computation [8]. Other systems include Girard's Light Linear Logic (LLL) and Elementary Linear Logic (ELL), complete for polynomial (resp. elementary) time computation [7]. Lafont's Soft Linear Logic can be seen as a subsystem of Bounded Linear Logic which is still powerful enough to encode polynomial time computation [10].

A good semantics for such a logic would lead to a semantic characterization of polynomial time computation [14]. One of the main points of this paper is to show that Soft Linear Logic possesses a very natural semantics, and therefore it is an ideal system for investigating the semantics of polynomial time computation.

We note that there has been previous work in the field of categorical models of various feasible fragments of Linear Logic. We note Baillot's *stratified coherence spaces* [3], which provides an example of a categorical model of Light

Linear Logic, Murawski and Ong's *discreet games* model of the intuitionistic multiplicative fragment of Light Affine Logic [14], Hofmann and Scott's realizability models of BLL [9], and O. Laurent and L. Tortora de Falco's *obsessional relational* models of ELL/SLL [11]. Our work on game semantics has been partially motivated by discussions with O. Laurent (private communication) who had considered an AJM-style game semantics similar to the one presented here for Soft Linear Logic.

But we should especially mention the unpublished work of Abramsky [2], which we were unaware of in the first draft of this paper. In his Clifford lectures at Tulane University, Abramsky gives several results which anticipate part of this work. In particular, he proposes the formula for $!A$ which we use in Sect. 4, and notes that such an exponential would not satisfy contraction. He also observes that the standard game models have sufficient limits for this formula to be applied.

In this paper, we shall always work in an intuitionistic setting even if it is not explicitly stated. Thus, for example, we shall write MELL as short for *Intuitionistic* Multiplicative Exponential Linear Logic.

2 Soft Linear Logic

In this section we review the theory of second-order Soft Linear Logic (SLL_2). Formulae are given by the following syntax:

$$A, B ::= \alpha | \mathbf{1} | A \& B | A \otimes B | A \multimap B | \forall \alpha . A | !A$$

Sequents are given intuitionistically: they have the form $\Gamma \vdash C$, where Γ is a finite (possibly empty) list of formulas. If Γ is the sequence of formulas $C_1, \ldots, C_n$, then $!\Gamma$ denotes the sequence $!C_1, \ldots, !C_n$. Also, if A is a formula and $n \in \mathbb{N}$, we write A^n for the formula $A \otimes \cdots \otimes A$ (n times) and $A^{(n)}$ for the sequence $A, \ldots, A$ (n times). Proofs are generated by a Gentzen style sequent calculus. The rules are the following (as originally presented in [10]):

- structural rules: *exchange*, *identity*, and *cut*

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C} \qquad \frac{}{A \vdash A} \qquad \frac{\Gamma \vdash A \quad \Delta, A \vdash C}{\Gamma, \Delta \vdash C}$$

- multiplicative logical rules:

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B} \qquad \frac{\Gamma \vdash A \quad \Delta, B \vdash C}{\Gamma, \Delta, A \multimap B \vdash C}$$

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \qquad \frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C} \qquad \frac{}{\vdash \mathbf{1}} \qquad \frac{\Gamma \vdash C}{\Gamma, \mathbf{1} \vdash C}$$

- additive logical rules:

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \& B} \qquad \frac{\Gamma, A \vdash C}{\Gamma, A \& B \vdash C} \qquad \frac{\Gamma, B \vdash C}{\Gamma, A \& B \vdash C}$$

– exponential logical rules: *soft (generic) promotion* and *multiplexing*

$$\frac{\mathit{\Gamma} \vdash A}{!\mathit{\Gamma} \vdash !A} \qquad \frac{\mathit{\Gamma}, A^{(n)} \vdash C}{\mathit{\Gamma}, !A \vdash C} \quad (n \text{ is any natural number})$$

– quantification logical rules:

$$\frac{\mathit{\Gamma} \vdash A}{\mathit{\Gamma} \vdash \forall\alpha.A} \quad (\alpha \text{ not free in } \mathit{\Gamma}) \qquad \frac{\mathit{\Gamma}, A[B/\alpha] \vdash C}{\mathit{\Gamma}, \forall\alpha.A \vdash C}$$

In multiplexing, the *rank* n can be any natural number. The notion of rank extends to proofs: the rank of a proof u in SLL$_2$ is the maximal rank of multiplexing in u. If all these ranks are the same, we say that u is *homogeneous*, and if there is no multiplexing, we say that u is *generic*. In SLL$_2$ programming, programs are represented by generic proofs and data by homogeneous proofs.

Note that one can derive the usual Linear Logic exponentials by adding the *digging* rule:

$$\frac{\mathit{\Gamma}, !!A \vdash C}{\mathit{\Gamma}, !A \vdash C}$$

to the soft exponential rules above. The following results are due to Lafont [10].

Theorem 1. *SLL$_2$ satisfies cut-elimination.*

Theorem 2. *SLL$_2$ is complete for polynomial time computation. In other words, any polynomial time algorithm is represented by a generic proof, and conversely, a generic proof defines a polynomial time algorithm via cut-elimination.*

Lafont conjectured in [10] that Multiplicative SLL$_2$ is also complete for polynomial time computation; this was later proved by Mairson and Terui in [13].

3 Multiplexor Categories

In this section we provide a categorical interpretation of the quantifier-free Multiplicative fragment of Soft Linear Logic (MSLL) in terms of multiplexor categories. We will tacitly assume that an interpretation of the additive connective can be obtained by assuming below that our categories have finite products.

Definition 1. *A* multiplexor category *consists of a triple $(\mathcal{C}, !, \{m_n\}_{n\geq 0})$, where $\mathcal{C} = (\mathcal{C}, \otimes, \multimap, \mathbf{1})$ is an autonomous category (=symmetric monoidal closed category) with structural maps α, λ, ρ and τ (twist), $! = (!, b_2, b_0)$ is a symmetric monoidal endofunctor on $\mathcal{C}$ with $\mathcal{C}$-maps $b_2 : !A \otimes !B \to !(A \otimes B)$ and $b_0 : \mathbf{1} \to !\mathbf{1}$, and for each natural number n,*

$$m_n : ! \stackrel{\cdot}{\to} (-)^{\otimes n}$$

is a monoidal natural transformation from ! to the n-fold tensor product functor, called a multiplexor of rank n. Implicit here is the commutativity of a number of diagrams. In particular, the definitions of monoidal functor and monoidal *natural transformation can be found in [12]. E.g., for multiplexing, in addition to naturality the following diagrams must commute:*

$$\begin{tikzcd}
!A \otimes !B \arrow[r, "b_2"] \arrow[d, "m_n \otimes m_n"'] & !(A \otimes B) \arrow[d, "m_n"] \\
A^{\otimes n} \otimes B^{\otimes n} \arrow[r, "\sim"] & (A \otimes B)^{\otimes n}
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbf{1} \arrow[r, "b_0"] \arrow[d, equal] & !\mathbf{1} \arrow[d, "m_n"] \\
\mathbf{1} \arrow[r, "\sim"] & \mathbf{1}^{\otimes n}
\end{tikzcd}$$

Many examples of multiplexor categories satisfy an additional condition.

Symmetry Condition. For each natural number n, the following diagram commutes:

$$\begin{tikzcd}
!A \arrow[r, "m_n"] \arrow[dr, "m_n"'] & A^{\otimes n} \arrow[d, "\sigma"] \\
 & A^{\otimes n}
\end{tikzcd}$$

for each permutation σ in S_n.

Definition 2. *A* symmetric multiplexor category *is a multiplexor category which satisfies the symmetry condition.*

The proof of the following is routine, but for space considerations we refer the reader to [15] for the details.

Theorem 3. *A multiplexor category provides a denotational semantics for MSLL. In other words, every proof in MSLL has an interpretation as an arrow in the category, and moreover, the interpretation is preserved under the cut-elimination procedure.*

For example, the interpretation of the sequent $!a \vdash a^n$, where a is an atom, is the morphism obtained by instantiating the natural transformation m_n with the interpretation of a.

4 Product Formulas

Our first examples of multiplexor categories come from models of Linear Logic. Multiplexing is a derived rule in the Multiplicative Exponential fragment of Intuitionistic Linear Logic (MELL):

$$\dfrac{\dfrac{A^{(n)} \vdash A^n}{!A^{(n)} \vdash A^n}}{!A \vdash A^n}$$

where the first double line represents dereliction rule(s) ($n \geq 1$), and the second double line represents either weakening ($n = 0$) or $n-1$ contraction rule(s).

In a similar way, a *Linear category* [4] is an example of a symmetric multiplexor category. The symmetry condition is given by the following diagram:

$$\begin{tikzcd}
!A \arrow[r] \arrow[dr] & (!A)^{\otimes n} \arrow[r] \arrow[d, "\sigma"] & A^{\otimes n} \arrow[d, "\sigma"] \\
 & (!A)^{\otimes n} \arrow[r] & A^{\otimes n}
\end{tikzcd}$$

The left triangle commutes by the commutative comonoid condition on $!A$ in a Linear category. The right square commutes by naturality of σ.

A good approximation to the soft exponential operator is provided by the following two theorems. In general such an interpretation satisfies neither the contraction rule nor the digging rule.

Theorem 4. *Let $\mathcal{C}$ be an autonomous category with countable products. Define:*

$$!A = \prod_n A^{\otimes n} = \mathbf{1} \times A \times (A \otimes A) \times (A \otimes A \otimes A) \times \cdots \quad . \tag{1}$$

Then $(\mathcal{C}, !)$ is a multiplexor category.

Theorem 5. *Let $\mathcal{C}$ be an autonomous category with countable products and equalizers of permutations (see below). Define:*

$$!A = \prod_n \mathcal{S}^n(A) = \mathbf{1} \times A \times (A \otimes_s A) \times (A \otimes_s A \otimes_s A) \times \cdots \quad . \tag{2}$$

where $A \otimes_s A$ is the following equalizer:

$$\begin{tikzcd}
A \otimes_s A \arrow[r, "e_2"] & A \otimes A \arrow[r, shift left, "id"] \arrow[r, shift right, "\tau"'] & A \otimes A
\end{tikzcd}$$

where τ is the twist map. The n-th symmetric power is defined analogously (as the equalizer of the set of permutations of n elements). Then $(\mathcal{C}, !)$ is a symmetric multiplexor category.

Proof. (Sketch) Given object A, multiplexing $!A \xrightarrow{m_n} A^{\otimes n}$ is defined as the composite $!A \xrightarrow{p_n} A^{\otimes_s n} \xrightarrow{e_n} A^{\otimes n}$, which is natural in A. A crucial part of the argument is in the construction of the map $b_2 :!A\otimes!B \to!(A \otimes B)$. The following diagram shows how this is constructed (n is any natural number):

$$\begin{tikzcd}
!A\otimes!B \arrow[r, "p_n\otimes p_n"] \arrow[d, dotted, "b_2"'] & A^{\otimes_s n} \otimes B^{\otimes_s n} \arrow[r, "e_n\otimes e_n"] \arrow[d, dotted] & A^{\otimes n} \otimes B^{\otimes n} \arrow[r, shift left=2, "id"] \arrow[r, phantom, "\vdots"] \arrow[r, shift right=2, "\sigma\otimes\sigma"'] \arrow[d, "\wr"] & A^{\otimes n} \otimes B^{\otimes n} \arrow[d, "\wr"] \\
!(A \otimes B) \arrow[r, "p_n"] & (A \otimes B)^{\otimes_s n} \arrow[r, "e_n"] & (A \otimes B)^{\otimes n} \arrow[r, shift left=2, "id"] \arrow[r, phantom, "\vdots"] \arrow[r, shift right=2, "\sigma"'] & (A \otimes B)^{\otimes n}
\end{tikzcd}$$

Note that this diagram also implies the first of the two monoidality diagrams for multiplexing (see Definition 1). Finally, the symmetry condition is justified by the following diagram:

$$\begin{tikzcd}
!A \arrow[r, "p_n"] \arrow[dr, "p_n"'] & A^{\otimes_s n} \arrow[r, "e_n"] \arrow[d, "id"] & A^{\otimes n} \arrow[d, "\sigma"] \\
& A^{\otimes_s n} \arrow[r, "e_n"] & A^{\otimes n}
\end{tikzcd}$$

The proof of the general case (Theorem 4) is similar. □

Theorems 4 and 5 lead to a large class of models. Indeed, many interesting models of MELL also possess a separate soft exponential operator. As a concrete example, we outline a basic game semantics for Soft Linear Logic.

Game Semantics. The simple two-player (Player vs. Opponent) game semantics described by Abramsky in [1] models the *negative fragment* of Intuitionistic Linear Logic. Recall:

A *game* G is a triple (M_G, λ_G, P_G) where

- M_G is the set of *moves* of the game;
- $\lambda_G : M_G \to \{P, O\}$ is a labeling function designating each move as by Player or Opponent;
- P_G (the game tree) is a non-empty, prefix closed subset of M_G^{alt}, the set of finite alternating sequences of moves in M_G, each beginning with an O-move.

Games are the objects of an autonomous category $\mathcal{G}$ with products, with morphisms given by strategies on games, as usual.

Given A, the Linear Logic exponential in $\mathcal{G}$ is roughly the infinite tensor product $A^{\otimes\omega}$ and validates contraction. The Soft Linear Logic exponential $!A$ is defined by (1), and can be described as follows:

$$\begin{aligned}
M_{!A} &= \{(i,j,a) | i,j \in \mathbb{N}^*, j \leq i, a \in M_A\} \\
\lambda_{!A}(i,j,a) &= \lambda_A(a) \\
P_{!A} &= \bigcup_i \{s \in M_{!A}^{\mathrm{alt}} | \forall j \in \{1,\ldots,i\}. s \upharpoonright (i,j) \in P_A \wedge \forall k \neq i. s \upharpoonright (k, \_) = \epsilon\}
\end{aligned}$$

where $\mathbb{N}^* = \mathbb{N} \backslash \{0\}$ and $s \upharpoonright (i,j)$ stands for the projection of s on the (i,j)-th copy of A. One can check directly that contraction no longer holds. One can also define $!A$ with (2), but the result is more complicated and strategies are no longer *history-free.*

5 Finitary Soft Linear Logic

In Soft Linear Logic $!A$ is not duplicable, but it is *cashable* on request (using multiplexing) for any number of copies of datum A required. (In this way $!A$ is analogous to a blank cheque, or a promissory note [2].) But from a computational perspective it may be useful to limit in advance the number of copies of datum A that may be paid.

To this end, we introduce a finitary version of Soft Linear Logic (SLL$_f$) by replacing the soft exponential operator ! with an $\mathbb{N}$-indexed family of operators $!_0, !_1, !_2, \ldots$, which are reminiscent of the *bounded reuse operators* $!_x$ of BLL [8], but have a different meaning. Intuitively, $!_n A$ is cashable for no more than n copies of datum A. Hence in the finitary version, the exponential rules of Soft Linear Logic are replaced by the following rules:

$$\frac{\Gamma \vdash A}{!_n \Gamma \vdash !_n A} \qquad \frac{\Gamma, A^{(n)} \vdash C}{\Gamma, !_{n+w} A \vdash C}$$

where w, n range over natural numbers. Cut-elimination works in a similar way to Soft Linear Logic, and there is an obvious translation into Soft Linear Logic which simply forgets the indices of the exponentials. Just as Soft Linear Logic is a subsystem of Linear Logic, finitary Soft Linear Logic can be seen as a subsystem of Bounded Linear Logic. We note that the following formula is provable in BLL but not in SLL$_f$:

$$!_{m+n} A \multimap !_m A \otimes !_n A \tag{3}$$

The categorical semantics developed for Soft Linear Logic generalizes to the finitary case as follows.

Definition 3. *A* finitary multiplexor category *consists of a triple of the form* $(\mathcal{C}, \{!_t\}_{t \geq 0}, \{m_{t,n}\}_{t \geq n \geq 0})$, *where* $\mathcal{C} = (\mathcal{C}, \otimes, \multimap, \mathbf{1})$ *is an autonomous category,* $!_t = (!_t, b_{2,t}, b_{0,t})$ *is a symmetric monoidal endofunctor on* $\mathcal{C}$ *for each natural number* t*, and for each pair of natural numbers* $t \geq n \geq 0$*,*

$$m_{t,n} : !_t \xrightarrow{\cdot} (-)^{\otimes n}$$

is a monoidal natural transformation from $!_t$ *to the* n*-fold tensor product functor, called a multiplexor of rank* (t, n).

Again, many finitary multiplexor categories also satisfy a further property.

Symmetry Condition. For each pair of natural numbers $t \geq n \geq 0$, the following diagram commutes:

$$\begin{tikzcd} !_t A \arrow[r, "m_{t,n}"] \arrow[dr, "m_{t,n}"'] & A^{\otimes n} \arrow[d, "\sigma"] \\ & A^{\otimes n} \end{tikzcd}$$

for each permutation σ in S_n.

Definition 4. *A* symmetric finitary multiplexor category *is a finitary multiplexor category which satisfies the symmetry condition.*

A finitary multiplexor category provides a denotational semantics for SLL_f. Moreover, any autonomous category with finite products has a canonical structure of a finitary multiplexor category (and a symmetric one if it has equalizers of permutations) using the finitary product formulas:

$$!_nA = \mathbf{1} \times A \times A^{\otimes 2} \times \cdots \times A^{\otimes n} \qquad \text{(general case)}$$
$$!_nA = \mathbf{1} \times A \times A^{\otimes_s 2} \times \cdots \times A^{\otimes_s n} \qquad \text{(symmetric case)}$$

The following is a concrete example of a symmetric finitary multiplexor category which has recently appeared in the literature [11].

Obsessional Cliques. Recall the category $\mathcal{OREL}$ introduced in [11], and described as follows:

- Objects: $\mathbb{N}$-sets. (A $\mathbb{N}$-set is given by a set A together with a multiplicative action $(k, a) \mapsto a^{(k)}$ from $\mathbb{N}^* \times A$ to A.)
- Morphisms: A morphism from $(A, \_^{(-)})$ to $(B, \_^{(-)})$ is a relation $R \subseteq A \times B$ which *preserves the action*, i.e.,

  $$\forall k \in \mathbb{N}^*, \forall (a, b) \in R.(a^{(k)}, b^{(k)}) \in R$$

  Such a relation is called an *obsessional clique.*
- Composition: usual relational composition.

The monoidal closed structure (cartesian product, $\{*\}$) is inherited from that of $\mathcal{REL}$ with pointwise action. For $t \in \mathbb{N}$, $!_tA$ is the set of finite multisets of elements of A with action given by:

$$[a_1, \ldots, a_n]^{(k)} = \begin{cases} [a_1^{(k)}, \ldots, a_n^{(k)}] & \text{if} \quad n \le t \\ [ka_1^{(k)}, \ldots, ka_n^{(k)}] & \text{if} \quad n > t \end{cases}$$

where ka is short for $a, \ldots, a$ (k times). Thus for a multiset of size less than or equal to t, the action is the usual pointwise action, allowing multiplexing of rank $\le t$. Moreover, this exponential structure rules out digging at all levels, and it rules out contraction for $t > 0$. (A good exercise is to show that bounded contraction, formula (3) above, is not valid in this category either when $m, n \ge 1$.) All together, $\mathcal{OREL}$ is a symmetric finitary multiplexor category, and a model of ELL with the ELL-exponential interpreted at the zeroth level by $!_0$. Finally, note that the forgetful functor $U : \mathcal{OREL} \to \mathcal{REL}$ preserves the structure strictly (see Sect. 6.1).

6 The S-Construction

Note that a multiplexor category is trivially a finitary multiplexor category in which $!_n = \,!$ for each natural number n. In this section, we are interested in the reverse direction. We give a categorical construction for extending a finitary multiplexor category $\mathcal{C}$ to a fully-fledged multiplexor category $\mathcal{C}^s$.

The category $\mathcal{C}^s$ is described as follows:

- Objects: Objects are sequences of $\mathcal{C}$-objects indexed by $\mathbb{N}$. Notation: $(A_i) = (A_0, A_1, A_2, \dots)$.
- Morphisms: A morphism from (A_i) to (B_i) is an equivalence class of sequences $(k : f_i) = (k : f_k, f_{k+1}, \dots)$, where k is a natural number and for each $i \geq k$, $f_i : A_i \to B_i$ in $\mathcal{C}$. Sequences $(k : f_i)$ and $(k' : f'_i)$ are equivalent iff there exists a natural number $m \geq k, k'$ such that for all $i \geq m$ we have $f_i = f'_i$ in $\mathcal{C}$. We shall abuse the notation and denote the equivalence class by one of its representatives.
- Composition: $(m : g_i) \circ (k : f_i) = (\max\{k, m\} : g_i \circ f_i)$. One must verify that this is well-defined. The identity on (A_i) is $(0 : id_{A_i})$.

There is an obvious embedding $J : \mathcal{C} \to \mathcal{C}^s$ defined by:

$$\begin{aligned} A &\mapsto (A, A, A, \dots) \\ f &\mapsto (0 : f) \end{aligned}$$

In a similar way, the entire structure lifts from that of the base category $\mathcal{C}$ with the following (well-defined) definitions:

$$\begin{aligned} (A_i) * (B_i) &= (A_i * B_i) \\ (k : f_i) * (m : g_i) &= (\max\{k, m\} : f_i * g_i) \\ !_n(A_i) &= (!_n A_i) \\ !_n(k : f_i) &= (k : !_n f_i) \\ (\mathbf{1}) &= (\mathbf{1}, \mathbf{1}, \dots) \end{aligned}$$

for $* \in \{\&, \otimes, \multimap\}$ and $n \in \mathbb{N}$. The structural isomorphisms lift in the obvious way to $(0 : \alpha), (0 : \lambda), (0 : \rho), (0 : \tau)$ for the monoidal structure, for example. This outlines the proof of the following proposition. We shall demonstrate one of the less obvious parts in detail.

Proposition 1. *$\mathcal{C}^s$ is a finitary multiplexor category. If $\mathcal{C}$ is symmetric, then so is $\mathcal{C}^s$.*

Proof. (Monoidal closedness.) The evaluation map ev is defined as $(0 : \mathrm{ev_i}) : ((A_i \multimap B_i) \otimes A_i) \to (B_i)$, where ev_i is the corresponding evaluation map in $\mathcal{C}$. Given any morphism $f = (k : f_i) : (X_i \otimes A_i) \to (B_i)$, we have a *unique* morphism $\hat{f} =^{\mathrm{def}} (k : \hat{f}_i) : (X_i) \to (A_i \multimap B_i)$ which satisfies $\mathrm{ev} \circ (\hat{f} \otimes id_A) = (k : ev_i \circ (\hat{f}_i \otimes id_{A_i})) = (k : f_i) = f$, where the second last equality follows from that in the base category $\mathcal{C}$. (Note that the assignment $f \mapsto \hat{f}$ is well-defined.) Once again, uniqueness of $\hat{f}$ follows from the uniqueness of the $\hat{f}_i$ in $\mathcal{C}$. □

The soft exponential operator is (well-)defined as follows:

$$\begin{aligned} !(A_i) &= (!_0 A_0, !_1 A_1, !_2 A_2, \dots) \\ !(k : f_i) &= (k : !_i f_i) \end{aligned}$$

The fact that this defines a symmetric monoidal endofunctor follows from the corresponding facts about the $!_i$'s.

We have the following proposition.

Proposition 2. *$\mathcal{C}^s$ is a multiplexor category. If $\mathcal{C}$ is symmetric, then so is $\mathcal{C}^s$.*

Proof. (Sketch) For each natural number n, multiplexing of rank n is defined as the equivalence class of the sequence $(n : m_{i,n}) = (n : m_{n,n}, m_{n+1,n}, \dots)$ where $m_{i,n} : !_i A_i \to A_i^{\otimes n}$ $(i \geq n)$ is bounded multiplexing in $\mathcal{C}$. For example, let us check the following diagram, which commutes in a multiplexor category, for any morphism $f = (k : f_i) : (A_i) \to (B_i)$,

$$\begin{tikzcd}
!A \arrow[r, "!f"] \arrow[dr, "m_0"'] & !B \arrow[d, "m_0"] \\
& \mathbf{1}
\end{tikzcd}$$

We have $m_0 \circ !f = (k : m_{i,0} \circ !_i f_i) = (k : m_{i,0}) \sim (0 : m_{i,0}) = m_0$. This shows why the equivalence relation is necessary. In a similar way, the remaining diagrams commute by the corresponding diagrams in the base category $\mathcal{C}$. We note that the equivalence relation is only used for multiplexing (as above, for example) and for lifting products from the base category $\mathcal{C}$. $\square$

6.1 A Variant of the S-Construction

In the applications, the base category $\mathcal{C}$ is often of a very simple type: a category of structured sets and structure-preserving functions (resp. relations). More explicitly, we have a finitary multiplexor category $\mathcal{C}$ together with a forgetful (or faithful) functor $U : \mathcal{C} \to \mathcal{SET}$ (resp. $U : \mathcal{C} \to \mathcal{REL}$), which preserves the structure strictly, e.g.:

$$\begin{aligned}
U(A \otimes B) &= U(A) \times U(B) \\
U(\mathbf{1}) &= \{*\} \\
U(A \multimap B) &= U(B)^{U(A)} && \text{(resp. } = U(A) \times U(B)) \\
U(!_n A) &= U(A) && \text{(resp. } = \mathcal{M}_f(U(A)))
\end{aligned}$$

where $\mathcal{M}_f(A)$ denotes the set of finite multisets of elements of A. Given a $\mathcal{C}$-object A, the underlying set $U(A)$ is denoted by $|A|$. In this case, the S-construction can be simplified and the category $\mathcal{C}^s$ is described as follows:

- Objects: Objects are sequences of $\mathcal{C}$-objects indexed by $\mathbb{N}$ (notation: $(A_i) = (A_0, A_1, A_2, \dots)$) with constant underlying set $|A|$.
- Morphisms: A morphism from (A_i) to (B_i) is a function (resp. relation) f from $|A|$ to $|B|$ such that there exists some $t \in \mathbb{N}$ with for any $n \geq t$, f is a morphism from A_n to B_n in $\mathcal{C}$ (i.e. structure-preserving at level n).
- Composition: Composition in the base category $\mathcal{C}$.

Everything follows as before, but without having to explicitly track the index k in the morphisms, and without the need of an equivalence relation. The following is an example of this type of model.

Obsessional Cliques (cont'd). Recall the category $\mathcal{SREL}$ introduced in [11], and described as follows:

- Objects: an object is a set A with a family of actions indexed by $\mathbb{N}$ giving it $\mathbb{N}$-set structures $(A_n)_{n \in \mathbb{N}}$.
- Morphisms: a morphism from (A_n) to (B_n) is a relation $R \subseteq A \times B$ such that there exists some $t \in \mathbb{N}$ with for any $n \geq t$, R is an obsessional clique of $A_n \times B_n$.
- Composition: usual relational composition.

There is an obvious embedding $\mathcal{OREL} \to \mathcal{SREL}$, and the monoidal closed structure of $\mathcal{OREL}$ lifts to $\mathcal{SREL}$ in the obvious way (i.e., componentwise). As a specific construction, we define:

$$!(A_n)_{n \in \mathbb{N}} = (!_n A_n)_{n \in \mathbb{N}}$$

This forms a symmetric multiplexor category. In fact $\mathcal{SREL}$ is the same as the category obtained by applying the S-construction to $\mathcal{OREL}$, i.e., $\mathcal{SREL} = \mathcal{OREL}^s$. This provides an alternative description of this model.

6.2 A Realizability Model for SLL$_2$

In this section, we give a realizability model of SLL$_2$ similar to the one for BLL in [9]. For space considerations we can only give a brief sketch here (see [15] for more details). It would be interesting to compare this approach to recent work done in [5].

In this section, we shall assume familiarity with the untyped lambda calculus. We shall also need the following preliminaries, which are taken from [9]:

Preliminaries. An untyped lambda term is *affine* (sometimes called *affine linear*) if each variable (free or bound) appears at most once (up to α-congruence). Such a term t is strongly normalizable in less than $|t|$ steps where $|t|$ is the size of the term. The runtime of the computation leading to the normal form is therefore $O(|t|^2)$. We will henceforth use the expression *affine lambda term* for an untyped affine lambda term which is in normal form. Given affine lambda terms s, t, their application st is defined as the normal form of the lambda term st. Finally, we write Λ_a for the set of closed affine lambda terms.

The category $\mathcal{R}_p^s$ is described as follows:

- Objects: A *realizability set* is a pair $(|A|, \Vdash_A)$, where $|A|$ is a set and $\Vdash_A$ is a ternary relation $\Vdash_A \subseteq \mathbb{N} \times \Lambda_a \times |A|$.
- Morphisms: A *morphism* from $(|A|, \Vdash_A)$ to $(|B|, \Vdash_B)$ is a function $f : |A| \to |B|$ such that there exists a natural number N and a sequence of terms $\{e_i\}_{i \geq N}$ (in Λ_a) such that:

$$i, t \Vdash_A a \qquad \text{implies} \qquad i, e_i t \Vdash_B f(a)$$

for all $t \in \Lambda_a$ and $a \in |A|$ and $i \geq N$. In this case we say that $\{e_i\}_{i \geq N}$ *witnesses* or *realizes* the function f. Moreover, we require that there exists

a polynomial $p(i)$ (over $\mathbb{N}$) such that $|e_i| \leq p(i)$ for all $i \geq N$. We call $p(i)$ a bounding polynomial.
- Composition: Usual function composition.

Let $A = (|A|, \Vdash_A)$ and $B = (|B|, \Vdash_B)$ be realizability sets. We have the following constructions on realizability sets, for example.

- *Tensor Product.* We define $A \otimes B = (|A| \times |B|, \Vdash_{A\otimes B})$ where:

$$i, t \Vdash_{A\otimes B} (a, b) \quad \text{iff} \quad t = \lambda f.ft_1t_2 \quad \text{where} \quad i, t_1 \Vdash_A a \quad \text{and} \quad i, t_2 \Vdash_B b$$

  The tensor unit is $\mathbf{1} = (\{*\}, \Vdash_{\mathbf{1}})$ where $i, \lambda x.x \Vdash_{\mathbf{1}} *$ for all $i \in \mathbb{N}$.
- *Linear Implication.* We define $A \multimap B = (|B|^{|A|}, \Vdash_{A\multimap B})$ where:

$$i, e \Vdash_{A\multimap B} f \quad \text{iff} \quad \text{whenever} \quad i, t \Vdash_A a \quad \text{then} \quad i, et \Vdash_B f(a)$$

- *Soft Exponential.* We define $!A = (|A|, \Vdash_{!A})$ where:

$$i, t \Vdash_{!A} a \quad \text{iff} \quad t = \lambda f.ft_1t_2 \ldots t_i \quad \text{where} \quad \forall j \in \{1, \ldots i\}.i, t_j \Vdash_A a$$

In [15], we show that $\mathcal{R}_p^s$ is a symmetric multiplexor category obtained by the S-construction. In fact, this was the original motivation for introducing the S-construction. Moreover, this model can be used to interpret second-order Soft Linear Logic [15]. As in [9], soundness of the model gives a new proof that all algorithms representable in SLL$_2$ are polynomial time. We have:

Theorem 6. *(Soundness) The category $\mathcal{R}_p^s$ provides a realizability semantics for SLL$_2$.*

We refer the reader to [15] for more details.

7 Conclusion

Ultimately, we are working towards a *fully complete* semantics of Soft Linear Logic [15]. But the models we have considered here are not yet complete. For example, in the product formula interpretation, there is a morphism $f : !a \to !(a \otimes a)$, but the formula $!a \multimap !(a \otimes a)$, where a is an atom, is not provable in Soft Linear Logic. An interesting question would be to explore the effects of adding this axiom to Soft Linear Logic, especially with regards to polytime soundness.

Lafont [10] also considers a possible extension of Soft Linear Logic with a hierarchy of modalities $!_{e_0}, !_{e_1}, !_{e_2}, \ldots$ and a hierarchical version of *digging*:

$$\frac{\Gamma, !_{e_n} \cdots !_{e_n} A \vdash C}{\Gamma, !_{e_{n+1}} A \vdash C}$$

This leads to a system for elementary recursive computation, as in ELL (see [7]). Both methods described in this paper can be used to model this extension.

For example, with the S-construction, one can define on objects (with a similar definition for morphisms):

$$\begin{aligned} !_{e_0}(A_i) &= (!_0A_0, !_1A_1, !_2A_2, \dots) && \text{(zeroth level)} \\ !_{e_1}(A_i) &= (!_0A_0, !_1!_1A_1, !_2!_2!_2A_2, \dots) && \text{(first level)} \\ !_{e_2}(A_i) &= (!_i^{(i+1)^2}A_i) && \text{(second level)} \\ &\dots \end{aligned}$$

Finally, it would be interesting to investigate connections with other bounded time complexity Linear Logics. Already, this work makes more explicit connections between Soft Linear Logic and Bounded Linear Logic, and a connection was made with Elementary Linear Logic in [11], which could be incorporated here by adding the following rule:

$$\frac{\Gamma, !_0A, !_0A \vdash C}{\Gamma, !_0A \vdash C}$$

to finitary Soft Linear Logic. (Note that $\mathcal{OREL}$ is also a model of this extended system [11].) It would be interesting if one could incorporate Girard's Light Linear Logic [7] into this setting also.

Acknowledgements. The author would like to sincerely thank his Ph.D. advisor Richard Blute and Phil Scott for their continuing encouragement and support for this project. This work has been greatly improved with suggestions by Olivier Laurent, Patrick Baillot and others at Universités Paris 7 and 13. Yves Lafont also provided helpful comments and answered many questions at the early stages of this project. Finally, we'd like to thank the anonymous referees for helpful comments on the draft version of this paper. This work is supported by the OGS program and NSERC of Canada

References

1. Abramsky, S.: Semantics of Interaction: an introduction to Game Semantics. In *Proceedings of the 1996 CLiCS Summer School, Isaac Newton Institute*, P. Dybjer and A. Pitts, eds., Cambridge University Press, 1–31, 1997
2. Abramsky, S.: Predicative Copying and Polynomial Time. Clifford Lectures, Tulane, 2002. Slides available at: http://web.comlab.ox.ac.uk/oucl/work/samson.abramsky/pcpt.pdf
3. Baillot, P.: Stratified coherence spaces: a denotational semantics for Light Linear Logic. *Theoretical Computer Science*, 318(1-2): 29–55, 2004
4. Bierman, G.M.: What is a Categorical Model of Intuitionistic Linear Logic?. In *Proceedings of Conference on Typed lambda calculus and Applications*, Springer-Verlag, LNCS 902: 78–93, 1995
5. Dal Lago, U., Hofmann, M.: Quantitative Models and Implicit Complexity. In *Proceedings of Foundations of Software Technology and Theoretical Computer Science*, Springer-Verlag, LNCS 3821: 189–200, 2005
6. Girard, J.-Y.: Linear logic. *Theoretical Computer Science*, 50: 1–102, 1987

7. Girard, J.-Y.: Light Linear Logic. *Information and Computation*, 143: 175–204, 1998
8. Girard, J.-Y., Scedrov, A., Scott, P.J.: Bounded linear logic: a modular approach to polynomial-time computability. *Theoretical Computer Science*, 97: 1–66, 1992
9. Hofmann, M., Scott, P.J.: Realizability Models for BLL-like Languages. *Theoretical Computer Science*, 318(1-2): 121–137, 2004
10. Lafont, Y.: Soft linear logic and polynomial time. *Theoretical Computer Science*, 318: 163–180, 2004
11. Laurent, O., Tortora de Falco, L.: Obsessional cliques: a semantic characterization of bounded time complexity. In *Proceedings of LICS*, IEEE Computer Society, 179–188, 2006
12. Mac Lane, S.: *Categories for the Working Mathematician, 2nd Edition*. Springer, 1998
13. Mairson, H., Terui, K.: On the Computational Complexity of Cut-Elimination in Linear Logic. In *Proceedings of ICTCS*, Springer-Verlag, LNCS 2841: 23–36, 2003
14. Murawski, A.S., Ong, C.-H.L.: Discreet Games, Light Affine Logic and PTIME Computation. In *Proceedings of CSL*, Springer-Verlag, LNCS 1862: 427–441, 2000
15. Redmond, B.F.: PhD Thesis. University of Ottawa. In preparation.

Until-Since Temporal Logic Based on Parallel Time with Common Past. Deciding Algorithms

V. Rybakov

Department of Computing and Mathematics,
Manchester Metropolitan University,
John Dalton Building, Chester Street, Manchester M1 5GD, U.K.
V.Rybakov@mmu.ac.uk

Abstract. We present a framework for constructing algorithms recognizing admissible inference rules (consecutions) in temporal logics with Until and Since based on Kripke/Hintikka structures modeling parallel time with common past. Logics $\mathcal{PTL}_\alpha$ with various branching factor $\alpha \in \mathcal{N} \cup \{\omega\}$ after common past are considered. The offered technique looks rather flexible, for instance, with similar approach we showed [33] that temporal logic based on sheafs of integer numbers with common origin is decidable by admissibility. In this paper we extend obtained algorithms to logics $\mathcal{PTL}_\alpha$. We prove that any logic $\mathcal{PTL}_\alpha$ is decidable w.r.t. admissible consecutions (inference rules), as a consequence, we solve satisfiability problem and show that any $\mathcal{PTL}_\alpha$ itself is decidable.

Keywords: Temporal logic, linear temporal logic, branching time logic, admissible inference rules.

1 Introduction

Temporal logic is a modern branch of mathematical logic which has diversified application in CS and AI. Technique of Temporal Logic has been involved to cover all approaches to the representation of temporal information within a logical framework. Significant part of this technique is based at modal-logic type of approach introduced around 1960 by Arthur Prior. Major part of applications was focused to reasoning about knowledge, time and computation (cf., for instance, Goldblatt [9], van Benthem [36]). Semantics approach to temporal logics has been based on Kripke/Hinttikka models and temporal algebras (cf. Thomason [35]), it has been noticeably influenced by developed tools of modal logics, which, in own turn, formed evolved branch in contemporary mathematical logic (cf. Goldblatt [10]).

Around 1970's multifarious variations of temporal logics focused to applications were developed (cf. Manna and Pnueli [20], Pnueli [23]). Linear temporal logic (LTL) has been quite successful in dealing with applications to systems specifications and verification (cf. [23]), with model checking (cf. [21,23]). Temporal logic has numerous applications to safety, liveness and fairness, to various problems arising in computing (cf. Eds. Barringer, Fisher, Gabbay and Gough, [1]).

Technique for model checking in linear temporal logic, LTL, based on Büchi automatons and Rabin tree automatons (cf. Carsten [2], Vardi [3,37]) formed a solid branch in computational logic. Significant part in investigation of LTL was focused to constructions of most efficient algorithms for verifying satisfiability in LTL, decidability of LTL (cf. [18]). E.g., the decidability of LTL can be shown by bounded finite model property (cf. Sistla, Clark [34]). A number of distinct variations of LTL by extension of the set of logical operations were suggested, e.g. Gabbay [5] proved that the formulas of LTL with *Past* can be rewritten in future of only LTL with possible exponential blowups, and in [17] it is shown that this exponential blow-up is necessary. Vardi [38] developed 2 way automata theory for a logic containing LTL+*Past*. Lange [16] has given a decision procedure and a complete axiomatization for LTL+*Past*. An axiomatization of the temporal logic with Until and Since over the real numbers was found in Gabbay, Hodkinson [6,7].

One of less investigated areas in temporal logic is the description of admissible inference rules (consecutions), the area which has been solidly developed for transitive modal logics and superintuitionistic logics. Admissible rules form the greatest class of rules which derive theorems of a logic $\mathcal{L}$ from $\mathcal{L}$-valid assumptions (actually, a rule is admissible in $\mathcal{L}$ if $\mathcal{L}$, as the set of all its theorems, is closed w.r.t. applications of this rule). Investigation of admissible rules was, in particular, initiated by H.Friedman problem about existence algorithms recognizing rules admissible in the intuitionistic logic IPC (cf. [4]).

The area of admissible inference rules was well developed for transitive modal and superintuitionistic logics (cf. Ghilardi [8], Iemhoff [13], Jerábek [14], Mints [22], Rybakov [24] – [30]). But for temporal logics not much is known, only I would refer Rybakov [32] and Rybakov [31]. In [32] it is proved that the (modal) standard linear temporal logic based on integer numbers (which does not use **U** and **S**) is decidable w.r.t. admissible rules. In this paper we extend our previous research to temporal logics with Until and Since.

Turning to the main subject of our paper, temporal logic is often used with a fixed semantics or *flow of time*. Common choices include the natural numbers, the integers, rationals, real numbers and various branching flows, depending on the application or on what time is perceived like. In [33] the almost linear logic with Until and Since based on sheafs of integers with common origin (0) has been studied, where an algorithm recognizing admissible rules was found. In present paper we extend this result to Until-Since Temporal Logics $\mathcal{PTL}_\alpha$ based on Parallel Time with Common Past, that is our logics have as semantics sheafs of integer numbers with common past - agglutinate negative numbers. Main result of the paper is the theorem stating that all logics $\mathcal{PTL}_\alpha$ are decidable w.r.t. admissible inference rules. This implies, in particular, that all these logics are decidable itself, and that satisfiability problem for logics $\mathcal{PTL}_\alpha$ is also decidable.

2 Notation, Preliminaries

The paper uses standard notation and known facts concerning modal, multimodal and temporal logics. Standard Linear Temporal logic (LTL in the sequel) has the language including Boolean logical operations and the temporal operations Next and Until. Formulas of LTL are built from a set *Prop* of atomic propositions and are closed w.r.t. applications of Boolean operations, the unary operation **N** (next) and the binary operation **U** (until). We extend the language of LTL by introduction the following counterpart operations to **N** and **U**: $\mathbf{N^{-1}}$ (previous) and **S** (since), so for any wffs φ and ψ, $\mathbf{N^{-1}}\varphi$ and $\varphi\mathbf{S}\psi$ are also wffs. This will be the language (notation $\mathcal{L}_{btl}$) of our temporal branching time logics. The semantics for formulas in $\mathcal{L}_{btl}$ is based on the set $\mathcal{Z}$ of all integer numbers with the standard linear order.

Our logic is generated by the following Kripke-Hintikka structures (models). Let $\mathcal{Z}$ be the set of all integer numbers, $\mathcal{N}$ be the set of all natural numbers, $\mathcal{Z}^- := \{m \mid m \in \mathcal{Z}, m \leq 0\}$, and for a set of indexes I and $i \in I$, $\mathcal{N}_i$ are disjoint copies of positive integer numbers. Numbers m from $\mathcal{N}_i$ will be denoted with subscripts, as m_i.

$$\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-) := \langle \bigcup_{i\in I} \mathcal{N}_i \cup \mathcal{Z}^-, R, Next, Prev, V\rangle$$

is the structure where relations $R, Next$ and $Prev$ are defined as follows.

$$\forall x, y \in \bigcup_{i\in I} \mathcal{N}_i \cup \mathcal{Z}^- \quad [(xRy \Leftrightarrow x \leq y)\&(x\ Prev\ y \Leftrightarrow y-1=x)\&$$

$$(x\ Next\ y \Leftrightarrow x+1=y)].$$

Relations *Next* and *Prev* are mutually converse and work in accordance with the standard intuition, and R is linear $\leq$ on $\mathcal{Z}^-$ and any $\mathcal{N}_i$. V is a valuation of a subset S of *Prop*, which assigns truth values to elements of S in $\bigcup_{i\in I}\mathcal{N}_i \cup \mathcal{Z}^-$. That is, for any $p \in S$, $V(p) \subseteq \bigcup_{i\in I}\mathcal{N}_i \cup \mathcal{Z}^-$. We will also use the notation $Next(a) := a+1, Prev(a) := a-1$, if $a \neq 0$, representing *Next* and *Prev* as functions.

The computational interpretation of $\bigcup_{i\in I}$ is as follows. The world 0 is the current time point of a computational process. The state -1 is the final state of the previous computation which then, in next step, results to 0 - our current time point. Since 0, the computational run may branch into states 1_i (taking various possible options, say, nondeterministic choice of instructions, expert's choice, etc...). In states after 0, and before 0, the structure behaves itself as the standard Kripke linear structure of LTL. Thus, in our interpretation, the worlds of $\bigcup_{i\in I}\mathcal{N}_i \cup \mathcal{Z}^-$ are *states*, R is the *transition* relation and V can be interpreted as *labeling* of the states with atomic propositions. If we refer to $\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$ as the Kripke frame, we mean $\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$ without valuation.

For $\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$, the truth values can be extended from propositions of S to arbitrary formulas constructed from these propositions as follows:

$$\forall p \in Prop,\ \forall a \in \otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-) \quad (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash p \Leftrightarrow a \in V(p);$$

$$(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \varphi \wedge \psi \Leftrightarrow (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \varphi \text{ and}$$

$$[(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \psi;$$

$$(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \neg\varphi \Leftrightarrow not[(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \varphi];$$

$$(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \mathbf{N}\varphi \Leftrightarrow \forall b[(a\ Next\ b) \Rightarrow (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), b) \Vdash \varphi];$$

$$(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \mathbf{N^{-1}}\varphi \Leftrightarrow \forall b[(a\ Prev\ b) \Rightarrow (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), b) \Vdash \varphi];$$

$$(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \varphi\mathbf{U}\psi \Leftrightarrow \exists b[(aRb) \land ((\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), b) \Vdash \psi) \land$$

$$\forall c[(aRcRb) \land \neg(bRc) \Rightarrow (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), c) \Vdash \varphi]];$$

$$(\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash \varphi\mathbf{S}\psi \Leftrightarrow \exists b[(bRa) \land ((\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), b) \Vdash \psi) \land$$

$$\forall c[(bRcRa) \land \neg(cRb) \Rightarrow (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), c) \Vdash \varphi]].$$

Note that all standard temporal or modal operations are definable through **U** and **S**. So, in particular, all modal operations $\Box^+$, $\Box^-$, $\Diamond^+$ and $\Diamond^-$ directed to the future and to the past are definable in our language.

For a Kripke-Hintikka structure $\mathcal{M} := \otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$ and a formula φ, φ is **true in** $\mathcal{M}$ (denotation – $\mathcal{M} \Vdash \varphi$) if $\forall a \in \mathcal{M}\ (\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-), a) \Vdash_V \varphi$. A formula φ is *satisfiable* in $\mathcal{PTL}_\alpha$ iff there is a Kripke structure based on $\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$ where φ is true at some world and $||I|| \leq \alpha$.

Definition 1. *For a cardinal number $\alpha \leq \omega$, the branching time temporal logic $\mathcal{PTL}_\alpha$ is the set of all formulas (in the language of LTL) which are true in any Kripke/Hinttikka structure based on any frame $\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$ with $||I|| \leq \alpha$.*

First we would like to discuss how to get an algorithm for checking satisfiability in $\mathcal{PTL}_\alpha$. The $\mathcal{PTL}_\alpha$-Kripke structures are based on $\mathcal{Z}^-$ and $\mathcal{N}_i$ and are infinite, the language of $\mathcal{PTL}$ with **U** (until) and **S** (before) is enough expressible, so we cannot find easy answer to this question immediately. In [31]) we studied the satisfiability problem for the temporal Tomorrow/Yesterday logic **TYL**, which is based on all finite intervals of integer numbers, but has no operations **U** and **S**. In was established (Theorem 2 [31]) that for a satisfiable formula φ there is a model for φ of a size linear from φ. Also (cf. [32]) it is shown that the standard (multi-modal) temporal logic (with only $\Diamond^+$ and $\Diamond^-$ as temporal operations) based on $\mathcal{Z}$ is decidable w.r.t. admissibility in spite of it has no finite model property. So, our logics $\mathcal{PTL}_\alpha$ inherit the lack of the finite model property, which shows that the task is not so easy. The satisfiability algorithm for $\mathcal{PTL}_\alpha$ (and decidability of $\mathcal{PTL}_\alpha$) will follow from more strong and general result – decidability of $\mathcal{PTL}_\alpha$ w.r.t. admissible consecutions – which will be presented in the final part of this paper.

3 Admissibility, Preliminary Discussion

This section contains an introductory discussion of structural logical consequence and provides definitions and terminology essential for the main part of this paper. The basic problem we are dealing with is how to convey that a formula (a statement) is a logical consequence of a collection of formulas (statements). We will study logical consequence in terms of logical consecutions. A **consecution**, (or, synonymously, – a **rule, inference rule**) **c** is an expression

$$\mathbf{c} := \frac{\varphi_1(x_1,\ldots,x_n),\ldots,\varphi_m(x_1,\ldots,x_n)}{\psi(x_1,\ldots,x_n)},$$

where $\varphi_1(x_1,\ldots,x_n),\ldots,\varphi_m(x_1,\ldots,x_n)$ and $\psi(x_1,\ldots,x_n)$ are some formulas constructed out of letters $x_1,\ldots,x_n$. Letters $x_1,\ldots,x_n$ are called variables of **c**, for any consecution **c**, $Var(\mathbf{c}) := \{x_1,\ldots,x_n\}$.

The formula $\psi(x_1,\ldots,x_n)$ is the conclusion of **c**, formulas $\varphi_j(x_1,\ldots,x_n)$ are the premises of **c**. Consecutions are supposed to describe the logical consequences, an informal meaning of a consecution is: *the conclusion* logically follows *from the premises*. The questions what *logically follows* means is crucial and has no evident and unique answer. We consider and compare below some approaches. Let $\mathcal{F}$ be a Kripke frame with a valuation V of all variables from a consecution $\mathbf{c} := \varphi_1,\ldots\varphi_n/\psi$, i.e. $Dom(V) = Var(\mathbf{c})$.

Definition 2. *The consecution* **c** *is said to be* **valid** *in the Kripke structure* $\langle\mathcal{F},V\rangle$ *(we will use notation* $\langle\mathcal{F},V\rangle \Vdash \mathbf{c}$*, or* $\mathcal{F}\Vdash_V \mathbf{c}$*) if* $(\mathcal{F}\Vdash_V \bigwedge_{1\leq i\leq m}\varphi_i) \Rightarrow (\mathcal{F}\Vdash_V \psi)$. *Otherwise we say* **c** *is* **refuted** *in* $\mathcal{F}$*, or* **refuted in** $\mathcal{F}$ **by** V*, and write* $\mathcal{F}\nVdash_V \mathbf{c}$.

A consecution **c** is **valid** in a frame $\mathcal{F}$ (notation $\mathcal{F}\Vdash\mathbf{c}$) if, for any valuation V of $Var(\mathbf{c})$, $\mathcal{F}\Vdash_V\mathbf{c}$. A consecution $\mathbf{c} := \varphi_1,\ldots\varphi_n/\psi$ is *valid* in a logic $\mathcal{L}(\mathcal{K})$ generated by a class of frames $\mathcal{K}$ if $\forall\mathcal{F}\in\mathcal{K}(\mathcal{F}\Vdash\mathbf{c})$. For many logics $\mathcal{L}$, if **c** is valid for $\mathcal{L}$ w.r.t. a class $\mathcal{K}$ generating $\mathcal{L}$, **c** will be also valid in all $\mathcal{L}$-Kripke frames. Note that this definition of valid consecutions is equivalent to the notion of valid modal sequent from [15], where a theory of sequent-axiomatic classes is developed. Also the notion of valid consecutions can be reduced to validity of formulas in the extension of the language with universal modality (cf. Goranko and Passy, [11]). Based on these results, some relevant technique of validity for consecutions can be constructed. This is easy to accept that valid consecutions correctly describe logical consequence (since, in particular, $\mathcal{L}(\mathcal{K})$ is closed w.r.t. valid consecutions). But a question is whether it is reasonable to restrict ourselves by only such consecutions studying a given logic $\mathcal{L}(\mathcal{K})$. Another class of *correct* logical consecutions consists of so called derivable consecutions. Let a logic $\mathcal{L}$ with a fixed axiomatic system $Ax_{\mathcal{L}}$ and a consecution $\mathbf{c} := \varphi_1,\ldots,\varphi_n/\psi$ are given.

Definition 3. **c** *is said to be* derivable *if* $\varphi_1,\ldots,\varphi_n \vdash_{Ax_{\mathcal{L}}} \psi$ *(i.e. if we can derive* ψ *from* $\varphi_1,\ldots,\varphi_n$ *in the given axiomatic system* $Ax_{\mathcal{L}}$*).*

The derivable consecutions are, of course, correct, and logics are closed w.r.t. derivable consecutions. But, for a logic $\mathcal{L}$ with a given axiomatic system, it can happen, that a formula ψ is not derivable from the premises $\varphi_1, ..., \varphi_m$, but still the rule $\mathbf{c} := \varphi_1, ..., \varphi_m/\psi$ is correct: $\mathbf{c}$ derives $\mathcal{L}$-provable conclusions from $\mathcal{L}$-provable premises. Derivable consecution must be valid, the converse is not always true. For instance, if we will take the modal logic $S4$ with the axiomatic system consisting of all $S4$-theorems as axioms and no inference rules at all, the consecution $x/\Box x$ is evidently non-derivable but is valid. The third kind of *correct* logical consecutions are admissible consecutions proposed by Lorenzen (1955) [19]. Given a logic $\mathcal{L}$, $Form_{\mathcal{L}}$ is the set of all formulas in the language of $\mathcal{L}$.

Definition 4. *A consecution*

$$\mathbf{c} := \frac{\varphi(x_1, \ldots, x_n), \ldots, \varphi_m(x_1, \ldots, x_n)}{\psi(x_1, \ldots, x_n)},$$

is said to be **admissible** *for a logic* $\mathcal{L}$ *if,* $\forall \alpha_1 \in Form_{\mathcal{L}}, \ldots, \forall \alpha_n \in Form_{\mathcal{L}}$,

$$[\bigwedge_{1 \leq i \leq m} [\varphi_i(\alpha_1, \ldots, \alpha_n) \in \mathcal{L}]] \Longrightarrow [\psi(\alpha_1, \ldots, \alpha_n) \in \mathcal{L}].$$

Thus, for any admissible consecution, any instance into the premises converting all of them into theorems of $\mathcal{L}$ also turns the conclusion to be a theorem. It is *most strong type of structural logical consecutions:* a consecution $\mathbf{c}$ is admissible in $\mathcal{L}$ iff $\mathcal{L}$, as the set of its own theorems, is closed with respect to $\mathbf{c}$. It is evident that any valid consecution is admissible. The converse is, again, not always true. Perhaps, the earliest example of a consecution which is admissible in the intuitionistic propositional logic (IPC, in sequel) but not valid for IPC (consequently, non-derivable in any axiomatic system for IPC) is the Harrop's rule (1960, [12]):

$$r := \frac{\neg x \to y \vee z}{(\neg x \to y) \vee (\neg x \to z)}.$$

G.Mints (1976, [22]) discovered an another inference rule:

$$\frac{(x \to y) \to x \vee z}{((x \to y) \to x) \vee ((x \to y) \to z)},$$

which is invalid but admissible in IPC. The Lemmon-Scott rule (cf. [27])

$$\frac{\Box(\Box(\Box\Diamond\Box p \to \Box p) \to (\Box p \vee \Box\neg\Box p))}{\Box\Diamond\Box p \vee \Box\neg\Box p}$$

is admissible but invalid in modal logics $S4$, $S4.1$ and Grz.

Notice that, for any temporal linear logic $\mathcal{PTL}_\alpha$ there are consecutions which are invalid (in particular, they are not derivable rules for any possible axiomatic system for $\mathcal{PTL}_\alpha$ where postulated inference rules preserve the truth values of formulas in the Kripke frames $\otimes_{i \in I} \mathcal{N}_i(cz^-)$, which, nevertheless, are admissible. For instance, the consecutions

$$c_1 := \frac{\Box^+\Diamond^+ x \wedge \Box^+\Diamond^+\neg x}{y}, \quad c_2 := \frac{\Box^-\Diamond^- x \wedge \Box^-\Diamond^-\neg x}{y}$$

are admissible but invalid in $\mathcal{PTL}_\alpha$ (because the premises of these consecutions are non-unifiable in any $\mathcal{PTL}_\alpha$). To comment satisfiability, the connection of admissibility with satisfiability problem is evident: φ is satisfiable in a logic $\mathcal{L}$ iff $x \to x/\neg\varphi$ is not admissible for $\mathcal{L}$. Therefore we address the rest of our paper to study of consecutions admissible in $\mathcal{PTL}_\alpha$.

4 Main Results

Our approach is based on semantic description of consecutions admissible via special n-characterizing Kripke structures, whose wrapping algebras are free algebras from the varieties corresponding to logics.

Definition 5. *Given a logic $\mathcal{L}$ and a Kripke structure $\mathcal{M}$ with a valuation defined for a set of letters $p_1, \ldots, p_k$. $\mathcal{M}$ is said to be* ***k*-characterizing** *for $\mathcal{L}$ if the following holds. For any formula $\varphi(p_1, \ldots, p_k)$ built using letters $p_1, \ldots, p_k$, $\varphi(p_1, \ldots, p_k) \in \mathcal{L}$ iff $\mathcal{M} \Vdash \varphi(p_1, \ldots, p_k)$.*

Recall that a Kripke structure $\mathcal{M}$ is said to be **definable** if any state $a \in \mathcal{M}$ is definable in $\mathcal{M}$, i.e. there is a formula ϕ_a which is true in $\mathcal{M}$ only at the element a. Given a Kripke structure $\mathcal{M} := \langle \mathcal{F}, V \rangle$ based upon a Kripke frame $\mathcal{F}$, and a new valuation V_1 in $\mathcal{F}$ of a set of propositional letters q_i. The valuation V_1 is **definable** in $\mathcal{M}$ if, for any propositional letter q_i, $V_1(q_i) = V(\phi_i)$ for some formula ϕ_i.

Lemma 1. *(cf., for instance, [27]) A consecution* **c** *is not admissible in a logic $\mathcal{L}$ iff, for any sequence $Ch_{\mathcal{L}}(k)$, $k \in N$, of k-characterizing models for $\mathcal{L}$, for some k, the frame of $Ch_{\mathcal{L}}(k)$ refutes* **c** *by a certain definable in $Ch_{\mathcal{L}}(k)$ valuation.*

To use this lemma we need a sequence of k-characterizing models for $\mathcal{PTL}_\alpha$. Take the all Kripke frames $\otimes_{i \in I} \mathcal{N}_i(\mathcal{Z}^-)$, $||I|| \leq \alpha$, with all possible valuations of letters p_1, …, p_k. Consider the all resulting Kripke models $\mathcal{M}_j, j \in J$ and the disjoint union $\bigsqcup_{j \in J} \mathcal{M}_j$ of all such non-isomorphic Kripke models, denote this disjoint union by $Ch_k(\mathcal{PTL}_\alpha)$. Semantics definition of $\mathcal{PTL}_\alpha$ implies

Lemma 2. *$Ch_k(\mathcal{PTL}_\alpha)$ is k-characterizing for $\mathcal{PTL}_\alpha$.*

Notice that any Kripke structure $Ch_k(\mathcal{PTL}_\alpha)$ fails to be definable. Therefore we cannot directly implement technique evolved in [27,31] to describe consecutions admissible in $\mathcal{PTL}_\alpha$, we need some more subtle, sophisticated technique.

As before, a useful tool we need is reduction of consecutions to equivalent reduced normal forms. A consecution $\mathbf{c}$ is said to have the *reduced normal form* if $\mathbf{c} = \varepsilon_c/x_1$ where

$$\varepsilon_c := \bigvee_{1\leq j\leq m} (\bigwedge_{1\leq i,k\leq n, i\neq k} [x_i^{t(j,i,0)} \wedge (\mathbf{N}x_i)^{t(j,i,1)} \wedge (\mathbf{N^{-1}}x_i)^{t(j,i,2)} \wedge$$

$$(x_i \mathbf{U} x_k)^{t(j,i,k,3)} \wedge (x_i \mathbf{S} x_k)^{t(j,i,k,4)}]),$$

and x_s are certain variables, $t(j,i,z), t(j,i,k,z) \in \{0,1\}$ and, for any formula α above, $\alpha^0 := \alpha$, $\alpha^1 := \neg\alpha$.

Definition 6. *Given a consecution c_{nf} in the reduced normal form, c_{nf} is said to be a normal reduced form for a consecution $\mathbf{c}$ iff, for any temporal logic $\mathcal{L}$, $\mathbf{c}$ is admissible in $\mathcal{L}$ iff $\mathbf{c_{nf}}$ is.*

Based on proofs of Lemma 3.1.3 and Theorem 3.1.11 from [27], by similar technique, we obtain

Theorem 1. *There exists an algorithm running in (single) exponential time, which, for any given consecution $\mathbf{c}$, constructs its normal reduced form $\mathbf{c_{nf}}$.*

To describe our algorithm distinguishing consecutions admissible in $\mathcal{PTL}_\alpha$, we need the following special finite Kripke structures.

For any natural numbers n, k and m, where $k > 1$, $n > 3$ and $m > n+1$, the Kripke structure $C[-k,0] \oplus_{0,1} C[1,n] \oplus_{n,n+1} C[n+1,m]$ is based on all integer numbers from the interval $[-k,m]$, and the relations $Next$ and $Prev$ on $[-k,m]$ are as follows: $\forall t \in [-k, m-1](Next(t) := t+1)$; $Next(m) := n+1$; $\forall t \in [-k+1, m](Prev(t) := t-1)$; $Prev(-k) := 0$. So, relations $Next$ and $Prev$ remain to be functions though they already are not mutually converse for all worlds of the structure. Therefore, this structure will not form $\mathcal{PTL}_\alpha$-frame for any possible interpretation of the accessibility relation R. Standard accessibility relation R (as in models $\otimes_{i\in I}\mathcal{N}_i(\mathcal{Z}^-)$ before) is not defined now at all. For a family of finite Kripke frames

$$\mathcal{F}_i := C[-k,0] \oplus_{0,1_i} [1_i, n_i] \oplus_{n_i,n_i+1} C[n_i+1, m_i], \quad i \in I,$$

with the structure specified above, and chosen words $v_i \in (1_i, n_i)$, where all v_i are equal, the frame

$$\otimes_{i\in I}\mathcal{F}_i{}^v := \langle \bigcup |\mathcal{F}_i|^v, Next, Prev\rangle$$

is the disjoint union of frames $\mathcal{F}_i$ with

(i) agglutinate worlds v_i: $v_i = v$ for all i,

(ii) agglutinate components $[-k,0]$ and

(iii) agglutinate components $[1_i, v_i]$.

In the frame $\otimes_{i\in I}\mathcal{F}_i^v$ the accessibility relation R is not defined. But for any formula φ in the language of $\mathcal{PTL}_\alpha$ and any valuation V of letters of φ in $\otimes_{i\in I}\mathcal{F}_i^v$

we can define truth values of φ in the frame $\otimes_{i\in I}\mathcal{F}_i^v$ w.r.t. V using inductive steps modified as follows:

$$(\otimes_{i\in I}\mathcal{F}_i^v, a) \Vdash \mathbf{N}\varphi \Leftrightarrow \forall b[\ a\ Next\ b \Rightarrow (\otimes_{i\in I}\mathcal{F}_i^v, b) \Vdash \varphi];$$

$$(\otimes_{i\in I}\mathcal{F}_i^v, a) \Vdash \mathbf{N^{-1}}\varphi \Leftrightarrow \forall b[\ a\ Prev\ b \Rightarrow (\otimes_{i\in I}\mathcal{F}_i^v, b) \Vdash \varphi];$$

$$(\otimes_{i\in I}\mathcal{F}_i^v, a) \Vdash \varphi\mathbf{U}\psi \Leftrightarrow \exists t \in N[[(\otimes_{i\in I}\mathcal{F}_i^v, Next^t(a)) \Vdash \psi]\wedge$$

$$\forall t_1 \in N[(0 \leq t_1 < t)\Rightarrow(\otimes_{i\in I}\mathcal{F}_i^v, Next^{t_1}(a)) \Vdash \varphi]];$$

$$(\otimes_{i\in I}\mathcal{F}_i^v, a) \Vdash \varphi\mathbf{S}\psi \Leftrightarrow \exists t \in N[[(\otimes_{i\in I}\mathcal{F}_i^v, Prev^t(a)) \Vdash \psi]\wedge$$

$$\forall t_1 \in N[(0 \leq t_1 < t)\Rightarrow(\otimes_{i\in I}\mathcal{F}_i^v, Prev^{t_1}(a)) \Vdash \varphi]].$$

Here we assume $Next(v)$ to be the next to v world, $Prev(v)$ is the previous to v world, $Next^{t_1}(a)$ ($Prev^{t_1}(a)$) is a world at the path leading from a to $Next^t(a)$ (from a back to $Prev^t(a)$). Also we will use notation aRb if $a = b$ or there is a path in $\otimes_{i\in I}\mathcal{F}_i^v$ by $Next$ leading from a to b.

For any consecution $\mathbf{c_{nf}}$ in normal reduced form, $Pr(c_{nf}) = \{\varphi_i \mid i \in I\}$ is the set of all disjunctive members of the premise of $\mathbf{c_{nf}}$. $Sub(c_{nf})$ is the set of all subformulas of $\mathbf{c_{nf}}$. For any Kripke frame $\mathcal{F}$ and any valuation V of the set of propositional letters of a formula φ, the expression $(\mathcal{F} \Vdash_V \varphi)$ is the abbreviation for $\forall a \in \mathcal{F}((\mathcal{F}, a) \Vdash_V \varphi)$.

Lemma 3. *For any frame $\mathcal{F}$ with a valuation V, where $\mathcal{F} \Vdash_V \bigvee Pr(c_{nf})$, for any $a \in \mathcal{M}$, there is a unique disjunct φ_i from $Pr(c_{nf})$ such that $(\mathcal{F}, a) \Vdash_V \varphi_i$.*

In the sequel we will denote this unique disjunct by $D_{\mathcal{F}}^{c_{nf},V}(a)$.

Lemma 4. *If a consecution $\mathbf{c_{nf}}$ in the normal reduced form is not admissible in $\mathcal{PTL}_\alpha$ then, there are integer numbers $k, n_i, m_i \in \mathcal{Z}$, where $-k < -1, 3 < n_i, n_i + 1 < m_i$, $i \in I$, $||I|| \leq \alpha$ and there exists a finite Kripke structure $\mathcal{K}_{\mathbf{c_{nf}}} := \langle \mathcal{F}_{\mathbf{c_{nf}}}, V_1\rangle$ which refutes $\mathbf{c_{nf}}$ by a valuation V_1, where $\mathcal{F}_{\mathbf{c_{nf}}} := \otimes_{i\in I}\mathcal{F}_i^v$,*

(a) $\mathcal{F}_i := C[-k, 0] \oplus_{0,1_i} [1_i, n_i] \oplus_{n_i,n_i+1} C[n_i + 1, m_i]$;
(b) there exists an index i and $a \in (1_i, n_i)$ where $(\mathcal{F}_{\mathbf{c_{nf}}}, a) \nVdash_{V_1} x_1$;
(c) $\forall i \quad D_{\mathcal{F}_{\mathbf{c_{nf}}}}^{c_{nf},V_1}(-k_i) = D_{\mathcal{F}_{\mathbf{c_{nf}}}}^{c_{nf},V_1}(1_i)$; $\quad D_{\mathcal{F}_{\mathbf{c_{nf}}}}^{c_{nf},V_1}(n_i) = D_{\mathcal{F}_{\mathbf{c_{nf}}}}^{c_{nf},V_1}(m_i)$;
(d) k, n_i, m_i and $||I||$ are linearly computable from the size of $\mathbf{c_{nf}}$.

The 3-modal frame $\mathbf{1}$ is the frame based at the single element set $\{a_1\}$ where $a_1 Next\ a_1$, $a_1 Prev\ a_1$ and $a_1 \leq a_1$. We can define truth values of formulas in the language of $\mathcal{PTL}_\alpha$ in $\mathbf{1}$ in the evident way.

Lemma 5. *If a consecution* $\mathbf{c_{nf}}$ *in the normal reduced form is not admissible in* $\mathcal{PTL}_\alpha$ *then there exists a valuation* V_0 *of variables from* $\mathbf{c_{nf}}$ *in the frame* $\mathbf{1}$ *where*

$$\mathbf{1} \Vdash_{V_0} \bigvee\{\varphi_i \mid \varphi_i \in Pr(c_{nf})\}$$

Lemma 6. *If a consecution* $\mathbf{c_{nf}}$ *in normal reduced form satisfies the conclusions of Lemma 4 and Lemma 5 then* $\mathbf{c_{nf}}$ *is not admissible in* $\mathcal{PTL}_\alpha$.

Using Theorem 1, Lemmas 4, 5 and 6 we immediately obtain

Theorem 2. *Any logic* $\mathcal{PTL}_\alpha$ *is decidable w.r.t. admissible consecutions. There is an algorithm recognizing consecutions admissible in* $\mathcal{PTL}_\alpha$.

Notice that any formula φ is a theorem of $\mathcal{PTL}_\alpha$ iff $x \to x/\varphi$ is admissible in $\mathcal{PTL}_\alpha$. A formula φ is satisfiable in $\mathcal{PTL}_\alpha$ iff $x \to x/\neg\varphi$ is not admissible in $\mathcal{PTL}_\alpha$. Thus, as a simple consequence, this algorithm solves the satisfiability problem for $\mathcal{PTL}_\alpha$ and shows that $\mathcal{PTL}_\alpha$ itself is decidable. The complexity of the algorithm from Theorem 2 is exponential reduction of rules (consecutions) to reduced normal forms plus complexity of model checking in models of size linear in obtained reduced normal forms. Using presence of the definable universal modality in any logic $\mathcal{PTL}_\alpha$ (as $\Box^+\Box^-$) it is possible to reduce the verification of admissibility to the question about validity of inference rules (or just formulas) in frames generating $\mathcal{PTL}_\alpha$. But the algorithm (which we can derive from the one given in this paper for admissibility) will look very similar.

References

1. Barringer H, Fisher M, Gabbay D., Gough G. *Advances in Temporal Logic, Vol. 16 of Applied logic series, Kluwer Academic Publishers, Dordrecht, 1999.*
2. Carsten Fritz. Constructing Büchi Automaton from Linear Temporal Logic Using Simulation Relations for Alternating Büchi Automata . - *Lecture Notes in Computer Science (LNCS), 2759, 2003* , pp. 35 – 48.
3. Daniele M, Giunchiglia F, Vardi M. Improved Automata Generation for Linear Temporal Logic. - In book: (CAV'99), *International Conference on Computer-Aided Verification, Trento, Italy*, 1999.
4. Friedman H. One Hundred and Two Problems in Mathematical Logic.- *Journal of Symbolic Logic, Vol. 40, 1975, No. 3*, pp. 113 - 130.
5. Gabbay D. The Declarative Past and Imperative Future: Executable Temporal Logic for Interactive Systems. *In: B. Banieqbal, H. Barringer, and A. Pnueli, editors, Proceedings of the 1st Conference on Temporal Logic in Specification, volume 398 of LNCS* , 1987, pp. 409 – 448.
6. Gabbay D.M., Hodkinson I.M. and Reynolds M.A. Temporal Logic: - *Mathematical Foundations and Computational Aspects,* volume 1. Clarendon Press, Oxford, 1994.
7. Gabbay D.M., Hodkinson I.M. An axiomatisation of the temporal logic with Until and Since over the real numbers. - *Journal of Logic and Computation*, Vol. 1 (1990), pp. 229-260.
8. Ghilardi S. Unification in Intuitionistic logic. - *Journal of Symbolic Logic, Vol. 64, No. 2 (1999)* , pp. 859-880.

9. Goldblatt R. Logics of Time and Computation.- *CSLI Lecture Notes, No.7, 1992.*
10. Goldblatt R. Mathematical Modal Logic: A View of its Evolution. - *J. Applied Logic, Vol 1 (5-6), 2003,* pp. 309 - 392.
11. Goranko V., Passy S. Using the Universal Modality: Gains and Questions.- *J. Log. Comput., Vol 2(1), 1992,* pp. 5–30.
12. Harrop R. Concerning Formulas of the Types $A \rightarrow B \vee C$, $A \rightarrow \exists x B(x)$ in Intuitionistic Formal System.- *J. of Symbolic Logic, Vol. 25, 1960*, pp. 27-32.
13. Iemhoff R. On the admissible rules of Intuitionistic Propositional Logic. - *J.of Symbolic Logic Vol. 66, 2001*, pp. 281-294.
14. Jerábek E. Admissible Rules of Modal Logics. -*J. of Logic and Computation*, 2005, Vol. 15. pp. 411-431.
15. Kapron B.M. Modal Sequents and Definability, *J.of Symbolic Logic, Vol. 52(3), 1987,* pp. 756 - 765.
16. Lange M. A quick axiomatisation of LTL with past. *Mathematical Logic Quarterly, V, 51, No 1, 2005*, pp. 83 - 88.
17. Laroussinie F., Markey N. and Schnoebelen Ph. Temporal Logic with Forgettable Past. *In: Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS'02), Copenhagen, Denmark, 2002*, pp 383-392.
18. Lichtenstein O., Pnueli A. Propositional temporal logics: Decidability and completeness. *Logic Journal of the IGPL, 8(1), 2000,* pp. 55 – 85.
19. Lorenzen P. Einführung in die operative Logik und Mathematik. - *Berlin-Göttingen, Heidelberg, Springer-Verlag,* 1955.
20. Manna Z., Pnueli A. The Temporal Logic of Reactive and Concurrent Systems: Specification. - *Springer-Verlag, 1992.*
21. Manna Z., Sipma H. Alternating the Temporal Picture for Safety. - *In Proc. 27th Intl. Colloq. Aut. Lang. Prog.(ICALP 2000). LNCS 1853, Springer-Verlag*, pp. 429-450.
22. Mints G.E. Derivability of Admissible Rules.- *J. of Soviet Mathematics, V. 6, 1976, No. 4*, pp. 417 - 421.
23. Pnueli A. The Temporal Logic of Programs.- *In Proc. of the 18th Annual Symp. on Foundations of Computer Science, IEEE, 1977*, pp. 46 – 57.
24. Rybakov V.V. A Criterion for Admissibility of Rules in the Modal System $S4$ and the Intuitionistic Logic. - *Algebra and Logic, V.23 (1984), No 5,* pp. 369 - 384 (Engl. Translation).
25. Rybakov V.V. The Bases for Admissible Rules of Logics S4 and Int.- *Algebra and Logic, V.24, 1985, pp. 55 – 68* (English translation).
26. Rybakov V.V. Rules of Inference with Parameters for Intuitionistic logic.- *Journal of Symbolic Logic, Vol. 57, No. 3, 1992*, pp. 912 - 923.
27. Rybakov V.V. Admissible Logical Inference Rules. - *Studies in Logic and the Found. of Mathematics, Vol. 136, Elsevier Sci. Publ., North-Holland, New-York-Amsterdam*, 1997.
28. Rybakov V.V., Kiyatkin V.R., Oner T., On Finite Model Property For Admissible Rules. - *Mathematical Logic Quarterly, Vol.45, No 4, 1999*, pp. 505 –520.
29. Rybakov V.V. Terziler M., Rimazki V. Basis in Semi-Reduced Form for the Admissible Rules of the Intuitionistic Logic IPC. - *Mathematical Logic Quarterly, Vol.46, No. 2 (2000)*, pp. 207 - 218.
30. Rybakov V.V. Construction of an Explicit Basis for Rules Admissible in Modal System S4. - *Mathematical Logic Quarterly, Vol. 47, No. 4 (2001)*, pp. 441 – 451.
31. Rybakov V.V. Logical Consecutions in Intransitive Temporal Linear Logic of Finite Intervals. - *Journal of Logic Computation, (Oxford Press, London), Vol. 15 No. 5 (2005)*, pp. 633 – 657.

32. Rybakov V.V. Logical Consecutions in Discrete Linear Temporal Logic. *Journal of Symbolic Logic, V.70, No 4 (2005)*, pp. 1137 – 1149.
33. Rybakov V.V. Branching Time Logic $\mathcal{PTL}_\alpha$ with Operations Until and Since Based on Bundles of Integer Numbers, Logical Consecutions, Deciding Algorithms, 2006, submitted
34. Sistla A.P. and Clarke E.M. The Complexity of Propositional Linear Temporal Logic. *Journal of the ACM, 32(3), 1985, pp. 733 – 749.*
35. Thomason S.K. Semantic Analysis of Tense Logic.- *Journal of Symbolic Logic, Vol. 37, No. 1 (1972).*
36. van Benthem J. The Logic of Time. - *Reidel, Dordrecht, Synthese Library, Vol. 156*, 1983.
37. Vardi M. An automata-theoretic approach to linear temporal logic.- In *Proceedings of the Banff Workshop on Knowledge Acquisition (1994), (Banff'94).*
38. Vardi M. Reasoning about the past with two-way automata. *In: Proc. 25th Int. Coll. on Automata, Languages, and Programming, Vol. 1443 of LNCS, 1998*, pp. 628–641.

Total Public Announcements

David Steiner and Thomas Studer

Universität Bern, Institut für Informatik und angewandte Mathematik,
Neubrückstrasse 10, CH-3012 Bern
steiner@iam.unibe.ch, tstuder@iam.unibe.ch
http://www.iam.unibe.ch/til/staff/

Abstract. We present a dynamic epistemic logic for knowledge change of rational agents. Existing approaches only deal with partial public announcements, that means an announcement may lead to an inconsistent state. We introduce an extension of the multi-modal logic $\mathsf{S5}_n$ featuring total public announcements where an update cannot result in an inconsistency. We also study total public announcements in the context of common knowledge and relativized common knowledge.

1 Introduction

At the end of the eighties, Plaza published the famous article about logics of public communications [1]. In the sequel, the theory of knowledge change caused by incoming information has been further developed by many authors. We confine ourselves to mentioning just a few typical articles: Baltag et al. [2,3], van Benthem et al. [4,5,6,7], van Ditmarsch et al. [8,9,10,11], as well as Renne [12].

The language for logics of public announcements is the language of standard multi-modal logic augmented with announcement operators $[\alpha]$ for every formula α. The expression $[\alpha]\beta$ then stands for *after every announcement of α, it holds that β*. In the classical setting, *announcement* has to be read as *truthful announcement.* Therefore, announcements are partial: that is

$$\neg[\alpha]\bot \tag{1}$$

is not valid (see [10], Proposition 4.11).

We propose a system in which announcements are *total*, that is (1) holds. Therefore, in our system announcements need not be truthful; they can be true or false. As usual, a true announcement will lead to an update of an agent's epistemic state. However, a false announcement will not lead to an inconsistent epistemic state, it will automatically be ignored by the agents. That is, after a false announcement, an agent will have the same epistemic state as before the announcement. Because (1) holds in our system, we call it *consistency preserving.* A system for consistency preserving belief change (instead of knowledge change) is studied in [13].

A property we keep from the classical setting is

$$p \to [\alpha]p. \tag{2}$$

That means, an announcement does not change atomic facts. We call a system that satisfies (2) *atomic preserving*.

The paper is organized as follows. In Section 2, we present an axiomatization of a system for total public announcements which satisfies both (1) and (2). We also propose a Kripke semantics for our system and show soundness and completeness of our axiomatization. In Section 3, we extend our logic with common knowledge operators. We show that agents can achieve common knowledge by receiving an announcement. For this system, the completeness proof cannot make use of a translation to a language without announcements (which is possible in the case without common knowledge). Completeness of total public announcements with common knowledge is established via the notion of maximal consistent sets. Recently, relativized common knowledge has received much attention in the context of logics for public announcements [7,5,6]. We introduce a system of relativized common knowledge and total public announcements in Section 4. Again, we establish soundness and completeness of our system. It follows from the proof that the addition of total public announcements to relativized common knowledge does not increase the expressive power of the language. We conclude this paper with some discussion in Section 5.

2 A System for Total Public Announcements

We introduce the language $\mathcal{L}_n^{\mathsf{A}}$ of multi-modal logic with dynamic-style operators for public announcements.

Given a natural number $n \geq 1$, we fix the set $\mathcal{A} = \{1, \ldots, n\}$ of n rational agents. Further, we take a countable non-empty set $\mathcal{P}$ of propositions, denoted by $p, q, \ldots$, possibly with subscripts. The set of $\mathcal{L}_n^{\mathsf{A}}$ formulas is defined by the following grammar ($p \in \mathcal{P}$, $i \in \mathcal{A}$),

$$\alpha, \beta, \ldots \ ::= \ p \ | \ \neg\alpha \ | \ \alpha \wedge \beta \ | \ K_i\alpha \ | \ [\alpha]\beta.$$

The formula $K_i\alpha$ stands for *the agent i knows* α, the formula $[\alpha]\beta$ means β *holds after the public announcement of* α. The $\mathcal{L}_0$ formulas are the propositional formulas, the $\mathcal{L}_n$ formulas are the modal formulas without announcement operators.

For all formulas α, β of $\mathcal{L}_n^{\mathsf{A}}$, we define $\alpha \vee \beta$, $\alpha \to \beta$, and $\alpha \leftrightarrow \beta$ as usual. Further, we let

$$\top := p_0 \vee \neg p_0 \qquad \text{and} \qquad \bot := p_0 \wedge \neg p_0$$

for some fixed $p_0 \in \mathcal{P}$. Iterated announcements $[\alpha]^k\beta$ are defined by induction on k. We set

$$[\alpha]^0\beta := \beta \qquad \text{and} \qquad [\alpha]^{k+1}\beta := [\alpha][\alpha]^k\beta.$$

The following system for total public announcements in the context of belief change can be obtained from Gerbrandy and Groeneveld [14]. It results from deleting edges in a Kripke structure.

(PT) Every instance of a propositional tautology,
(K) $K_i(\alpha \to \beta) \to (K_i\alpha \to K_i\beta)$,
(4) $K_i\alpha \to K_iK_i\alpha$,
(5) $\neg K_i\alpha \to K_i\neg K_i\alpha$,
(A1) $[\alpha]p \leftrightarrow p$,
(A2) $[\alpha](\beta \to \gamma) \to ([\alpha]\beta \to [\alpha]\gamma)$,
(A3) $[\alpha]\neg\beta \leftrightarrow \neg[\alpha]\beta$,
(A4.B) $[\alpha]K_i\beta \leftrightarrow K_i(\alpha \to [\alpha]\beta)$,

$$(\mathsf{MP})\frac{\alpha \qquad \alpha \to \beta}{\beta}, \qquad (\mathsf{NEC.1})\frac{\alpha}{K_i\alpha}, \qquad (\mathsf{NEC.2})\frac{\alpha}{[\beta]\alpha}.$$

To obtain a system for knowledge change, we cannot just add the knowledge axiom $K_i\alpha \to \alpha$ since that would lead to an inconsistency. The theory $\mathsf{S5}_n^{\mathsf{A}}$ is obtained by changing axiom (A4.B) to (A4) and adding axioms (T) and (A5).

(T) $K_i\alpha \to \alpha$,
(A4) $\alpha \to ([\alpha]K_i\beta \leftrightarrow K_i(\alpha \to [\alpha]\beta))$,
(A5) $\neg\alpha \to ([\alpha]\beta \leftrightarrow \beta)$.

The instances of axiom (A5) of the form

$$\neg\alpha \to ([\alpha]K_i\beta \leftrightarrow K_i\beta) \tag{3}$$

can be seen as a necessary companion to the knowledge axiom (T), because a false formula can never be known by an agent, thus it can never be learned. Note that we could formulate the system $\mathsf{S5}_n^{\mathsf{A}}$ with (3) instead of (A5). Then (A5) would be provable in the resulting system. However, later we will consider an extension of $\mathsf{S5}_n^{\mathsf{A}}$ by common knowledge operators. There, things get much simpler if (A5) is already included as an axiom.

By replacing (T) with the (D)-axiom $\neg K_i\bot$, the agents can decide whether to accept an incoming formula or to reject it. For a detailed treatment of this approach, see [13].

Our system is consistency preserving as well as atomic preserving. We have the following lemma.

Lemma 1. *For all $\mathcal{L}_n^{\mathsf{A}}$ formulas α and all propositions $p \in P$ we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \neg[\alpha]\bot \qquad \textit{and} \qquad \mathsf{S5}_n^{\mathsf{A}} \vdash p \to [\alpha]p.$$

Observe, that the public announcement operator is self-dual due to axiom (A3). This means we do not have to distinct the statements 'β holds after every (truthful) public announcement of α' and 'β holds after some (truthful) public announcement of α'. In our setting, there is only one public announcement of a formula. It can be truthful or not.

We will now state some properties which will be helpful for the rest of the paper.

Lemma 2. *For all $\mathcal{L}_n^{\mathsf{A}}$ formulas α, β, and γ we have that $\mathsf{S5}_n^{\mathsf{A}}$ proves*

$$\begin{aligned}
[\alpha](\beta \wedge \gamma) &\leftrightarrow ([\alpha]\beta \wedge [\alpha]\gamma),\\
[\alpha](\beta \vee \gamma) &\leftrightarrow ([\alpha]\beta \vee [\alpha]\gamma),\\
[\alpha]K_i\beta &\leftrightarrow (\neg\alpha \wedge K_i\beta) \vee (\alpha \wedge K_i(\alpha \rightarrow [\alpha]\beta)).
\end{aligned}$$

We now give semantics to our logic of total public announcements. Since we deal with an extension of $\mathsf{S5}_n$, we will only need Kripke structures where the accessibility relations are equivalence relations.

Definition 1. An n-Kripke structure $\mathsf{K} = (S, R_1, \ldots, R_n, V)$ is an $(n+2)$-tuple, where $S \neq \emptyset$ is a set of states, $R_i \subseteq S^2$ for all $i \in \mathcal{A}$, and $V : \mathcal{P} \mapsto Pow(S)$ is a valuation function.

Since n is fixed, we will drop it and use only the term Kripke structure. The set S is called the universe of K, denoted by $|\mathsf{K}|$. In the sequel, we will write $\mathcal{K}_n^{eq}$ for the class of all Kripke structures with equivalence relations exclusively.

We will now define the validity of an $\mathcal{L}_n^{\mathsf{A}}$ formula in a state s of a Kripke structure K. The crucial point in this definition is the case of $[\alpha]\beta$, where we simultaneously define an operation on K, depending on α and s.

Definition 2. Let $\mathsf{K} = (S, R_1, \ldots, R_n, V)$ be an arbitrary Kripke structure and $s \in S$ be given. The validity of $\mathcal{L}_n^{\mathsf{A}}$ formulas in the Kripke-world K, s is inductively defined as follows.

$$\begin{aligned}
\mathsf{K}, s \models p &\text{ iff } s \in V(p),\\
\mathsf{K}, s \models \neg\alpha &\text{ iff } \mathsf{K}, s \not\models \alpha,\\
\mathsf{K}, s \models \alpha \wedge \beta &\text{ iff } \mathsf{K}, s \models \alpha \text{ and } \mathsf{K}, s \models \beta,\\
\mathsf{K}, s \models K_i\alpha &\text{ iff } \text{for every } t \in S,\ sR_it \Rightarrow \mathsf{K}, t \models \alpha,\\
\mathsf{K}, s \models [\alpha]\beta &\text{ iff } \mathsf{K}^{\alpha,s}, s \models \beta,
\end{aligned}$$

where for given $\alpha \in \mathcal{L}_n^{\mathsf{A}}$ and $s \in |\mathsf{K}|$, the Kripke structure $\mathsf{K}^{\alpha,s}$ is simultaneously defined by

$$\begin{aligned}
\mathsf{K}^{\alpha,s} &:= (S, R_1^{\alpha,s}, \ldots, R_n^{\alpha,s}, V),\\
R_i^{\alpha,s} &:= \begin{cases} R_i \cap \{(u,v) \in S^2 \mid \mathsf{K}, u \models \alpha \text{ iff } \mathsf{K}, \mathrm{v} \models \alpha\} & \text{if } \mathsf{K}, s \models \alpha,\\ R_i & \text{if } \mathsf{K}, s \not\models \alpha. \end{cases}
\end{aligned}$$

We say that an $\mathcal{L}_n^{\mathsf{A}}$ formula α is valid in the Kripke structure K ($\mathsf{K} \models \alpha$), iff for all $s \in S$, $\mathsf{K}, s \models \alpha$. The formula α is valid with respect to $\mathcal{K}_n^{eq}$ ($\mathcal{K}_n^{eq} \models \alpha$), iff for all $\mathsf{K} \in \mathcal{K}_n^{eq}$, $\mathsf{K} \models \alpha$. Further, we say that α is satisfiable in $\mathcal{K}_n^{eq}$, iff there is a $\mathsf{K} \in \mathcal{K}_n^{eq}$ and an $s \in |\mathsf{K}|$, such that $\mathsf{K}, s \models \alpha$.

If all R_i in K are equivalence relations, then $\mathsf{K}^{\alpha,s}$ belongs to $\mathcal{K}_n^{eq}$, as is stated in the next lemma. It is an immediate consequence of the definition of $R_i^{\alpha,s}$.

Lemma 3. *For all Kripke structures K, all $\mathcal{L}_n^{\mathsf{A}}$ formulas α, and all $s \in |\mathsf{K}|$ we have*

$$\mathsf{K} \in \mathcal{K}_n^{eq} \quad \Rightarrow \quad \mathsf{K}^{\alpha,s} \in \mathcal{K}_n^{eq}.$$

Example 1 (Wise men puzzle). Alice, Bob, and Charlie wear a hat and cannot see its color. But they can see, of course, the color of the others' hats. There are only two red and three blue hats, and every person knows that. The Kripke structure K, which represents this situation is shown in Figure 1 (reflexivity and symmetry of the relations R_A, R_B, and R_C are self-evident).

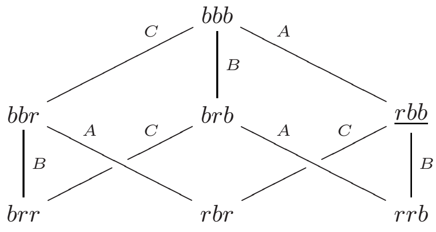


Fig. 1. The initial structure

In the state *rbb*, Alice wears a red hat, whereas Bob and Charlie both wear a blue hat. Now, Alice publicly announces, that she does not know the color of her hat, which is true. After that, Bob announces the same true fact. This results in the Kripke structure

$$(\mathsf{K}^{\neg K_A r_A \wedge \neg K_A b_A, rbb})^{\neg K_B r_B \wedge \neg K_B b_B, rbb},$$

which is illustrated in Figure 2.

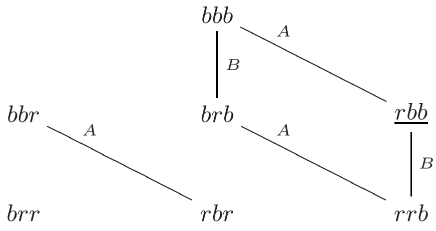


Fig. 2. The situation after two announcements

Now Charlie knows that he is wearing a blue hat. Observe, that for a false announcement, e.g. b_A in the state *rbb*, we have that $\mathsf{K}^{b_A, rbb} = \mathsf{K}$.

Soundness of our system can be proved in the usual way.

Lemma 4. *The system* $\mathsf{S5}_n^{\mathsf{A}}$ *is sound with respect to* $\mathcal{K}_n^{eq}$*, i.e. for all* $\mathcal{L}_n^{\mathsf{A}}$ *formulas* α *we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \quad \Rightarrow \quad \mathcal{K}_n^{eq} \models \alpha.$$

Proof. The proof is by induction on the length of the derivation. In the base case, soundness of the axiom $(\mathsf{A4})$ is proved as follows. Let $\mathsf{K} \in \mathcal{K}_n^{eq}$, $s \in |\mathsf{K}|$, and $i \in \mathcal{A}$ be given and assume that $\mathsf{K}, s \models \alpha$. Then we have

$$\begin{aligned}
\mathsf{K}, s \models [\alpha]K_i\beta \ &\text{iff}\ \ \mathsf{K}^{\alpha,s}, s \models K_i\beta \\
&\text{iff}\ \ \text{for all } t \in S,\ sR_i^{\alpha,s}t \ \Rightarrow\ \mathsf{K}^{\alpha,s}, t \models \beta \\
&\text{iff}\ \ \text{for all } t \in S,\ sR_it \text{ and } \mathsf{K}, t \models \alpha \ \Rightarrow\ \mathsf{K}, t \models [\alpha]\beta \\
&\text{iff}\ \ \mathsf{K}, s \models K_i(\alpha \to [\alpha]\beta).
\end{aligned}$$

In the induction step, soundness of the rule $(\mathsf{NEC.2})$ immediately follows from Lemma 3. □

Completeness of $\mathsf{S5}_n^{\mathsf{A}}$ can be proved via a translation from $\mathcal{L}_n^{\mathsf{A}}$ to $\mathcal{L}_n$ since the two languages have the same expressive strength. As a preparation, we will define a translation from $\{[\alpha]\beta \mid \alpha, \beta \in \mathcal{L}_n\}$ to $\mathcal{L}_n$.

Definition 3. The function h from $\{[\alpha]\beta \mid \alpha, \beta \in \mathcal{L}_n\}$ to $\mathcal{L}_n$ is inductively defined by

$$\begin{aligned}
h([\alpha]p) &:= p, \\
h([\alpha]\neg\beta) &:= \neg h([\alpha]\beta), \\
h([\alpha](\beta \wedge \gamma)) &:= h([\alpha]\beta) \wedge h([\alpha]\gamma), \\
h([\alpha]K_i\beta) &:= (\neg\alpha \wedge K_i\beta) \vee (\alpha \wedge K_i(\alpha \to h([\alpha]\beta))).
\end{aligned}$$

Of course, h eliminates the announcement operator. Its definition leads to two important properties which we state in the following lemma.

Lemma 5. *For all $\mathcal{L}_n^{\mathsf{A}}$ formulas α, β and all $\mathcal{L}_n$ formulas φ, ψ we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash [\varphi]\psi \leftrightarrow h([\varphi]\psi) \qquad \textit{and} \qquad \mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \leftrightarrow \beta \ \Rightarrow\ \mathsf{S5}_n^{\mathsf{A}} \vdash [\alpha]\psi \leftrightarrow [\beta]\psi.$$

We are now able to define our translation t, which eliminates the announcement operator in every $\mathcal{L}_n^{\mathsf{A}}$ formula.

Definition 4. The translation t from $\mathcal{L}_n^{\mathsf{A}}$ to $\mathcal{L}_n$ is inductively defined by

$$\begin{aligned}
t(p) &:= p, \\
t(\neg\alpha) &:= \neg t(\alpha), \\
t(\alpha \wedge \beta) &:= t(\alpha) \wedge t(\beta), \\
t(K_i\alpha) &:= K_i t(\alpha), \\
t([\alpha]\beta) &:= h([t(\alpha)]t(\beta)).
\end{aligned}$$

It is obvious that for every $\mathcal{L}_n^{\mathsf{A}}$ formula α, its translation $t(\alpha)$ is a formula of $\mathcal{L}_n$. In addition, we can prove the equivalence of α and $t(\alpha)$ in $\mathsf{S5}_n^{\mathsf{A}}$.

Lemma 6. *For all $\mathcal{L}_n^{\mathsf{A}}$ formulas α we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \leftrightarrow t(\alpha).$$

Lemma 6 is very helpful for proofs by induction on arbitrary $\mathcal{L}_n^{\mathsf{A}}$ formulas. Making use of it, one can easily show the following property.

Corollary 1. *For all $\mathcal{L}_n^{\mathsf{A}}$ formulas α, β, and γ we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \leftrightarrow \beta \quad \Rightarrow \quad \mathsf{S5}_n^{\mathsf{A}} \vdash [\alpha]\gamma \leftrightarrow [\beta]\gamma.$$

As another consequence of Lemma 6 we get the following equivalence concerning consecutive announcement operators.

Lemma 7. *For all $\mathcal{L}_n^{\mathsf{A}}$ formulas α, β, and γ we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \wedge [\alpha]\beta \rightarrow ([\alpha][\beta]\gamma \leftrightarrow [\alpha \wedge [\alpha]\beta]\gamma).$$

Making use of Lemma 6, we can easily show completeness of $\mathsf{S5}_n^{\mathsf{A}}$.

Lemma 8. *The system $\mathsf{S5}_n^{\mathsf{A}}$ is complete with respect to $\mathcal{K}_n^{eq}$, i.e. for all $\mathcal{L}_n^{\mathsf{A}}$ formulas α we have*

$$\mathcal{K}_n^{eq} \models \alpha \quad \Rightarrow \quad \mathsf{S5}_n^{\mathsf{A}} \vdash \alpha.$$

Proof. Assuming $\mathcal{K}_n^{eq} \models \alpha$, we get $\mathcal{K}_n^{eq} \models t(\alpha)$ by soundness and Lemma 6. Due to completeness of $\mathsf{S5}_n$, we have $\mathsf{S5}_n \vdash t(\alpha)$, which yields $\mathsf{S5}_n^{\mathsf{A}} \vdash t(\alpha)$ because $\mathsf{S5}_n$ is contained in $\mathsf{S5}_n^{\mathsf{A}}$. Now, we get $\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha$ by Lemma 6. $\square$

In a next step, we define *announcement-resistant* $\mathcal{L}_n^{\mathsf{A}}$ formulas. This notion is inspired by – but different than – the notion of successful formulas, see [9]. A formula α is successful if $[\alpha]\alpha$ is valid. However, in our setting not even propositions would be successful formulas. As an alternative, we introduce the class of announcement-resistant formulas.

Definition 5. An $\mathcal{L}_n^{\mathsf{A}}$ formula α is called announcement-resistant, if for all $\mathcal{L}_n^{\mathsf{A}}$ formulas β we have

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \rightarrow [\beta]\alpha.$$

Observe, that $\mathsf{S5}_n^{\mathsf{A}}$ proves $\alpha \rightarrow [\beta]^k\alpha$ for all $k \geq 0$, if α is announcement-resistant. There are many announcement-resistant formulas, as the following lemma shows.

Lemma 9.

1. *All $\mathcal{L}_0$ formulas as well as all provable $\mathcal{L}_n^{\mathsf{A}}$ formulas are announcement-resistant.*
2. *If α and β are announcement-resistant, then so also are the formulas $\alpha \wedge \beta$, $\alpha \vee \beta$, and $K_i\alpha$.*

By the previous lemma, we know that $K_i\alpha$ is announcement-resistant for all $\mathcal{L}_0$ formulas α. That means knowledge in propositional formulas can never be contracted by public announcements. We can therefore say that the logic of total public announcements formalizes expansion for propositional knowledge.

As we have seen in Example 1, agents can really expand their knowledge due to announcements. The next lemma shows that they learn true announcement-resistant formulas by one single announcement.

Lemma 10. *Let α be an announcement-resistant $\mathcal{L}_n^{\mathsf{A}}$ formula. Then for all $k \geq 1$, all $m \geq 0$, and all $i_1, \ldots, i_m \in \mathcal{A}$ we have*

$$\mathsf{S5}_n^{\mathsf{A}} \vdash \alpha \rightarrow [\alpha]^k K_{i_m} \ldots K_{i_1}\alpha.$$

3 Incorporating Common Knowledge

Lemma 10 even shows that agents can acquire so-called common knowledge. In this section, we will extend our logic of total public announcements by common knowledge operators. To this aim we have to formalize the notion of mutual knowledge. For every non-empty group $G \subseteq \mathcal{A}$ of agents, the formula $\mathsf{E}^k_G\alpha$ is inductively defined by

$$\mathsf{E}^0_G\alpha := \alpha \qquad \text{and} \qquad \mathsf{E}^{k+1}_G\alpha := \bigwedge_{i\in G}(K_i\mathsf{E}^k_G\alpha).$$

We simply write $\mathsf{E}_G\alpha$ for $\mathsf{E}^1_G\alpha$ to express that *everybody in G knows α*. The following property holds.

Lemma 11. *For all $\mathcal{L}^{\mathsf{A}}_n$ formulas α and β we have*

$$\mathsf{S5}^{\mathsf{A}}_n \vdash \alpha \to ([\alpha]\mathsf{E}_G\beta \leftrightarrow \mathsf{E}_G(\alpha \to [\alpha]\beta)).$$

The language $\mathcal{L}^{\mathsf{C,A}}_n$ of common knowledge and public announcements is the language $\mathcal{L}^{\mathsf{A}}_n$ expanded by the common knowledge operator C_G for every non-empty group $G \subseteq \mathcal{A}$ of agents. To define the validity of $\mathcal{L}^{\mathsf{C,A}}_n$ formulas in a Kripke-world K, s, we add the following clause to Definition 2.

$$\mathsf{K}, s \models \mathsf{C}_G\alpha \quad \text{iff} \quad \text{for every } t \in |\mathsf{K}|,\ s(R_G)^\star t \Rightarrow \mathsf{K}, t \models \alpha,$$

where $(R_G)^\star$ denotes the transitive closure of $R_G := \bigcup\{R_i \mid i \in G\}$, see Fagin et al. [15]. It is an easy exercise to show that

$$\mathsf{K}, s \models \mathsf{C}_G\alpha \quad \text{iff} \quad \text{for every } t, \text{ there is a } G\text{-path from s to t} \Rightarrow \mathsf{K}, t \models \alpha.$$

The system $\mathsf{S5}^{\mathsf{C,A}}_n$ is defined as extension of $\mathsf{S5}^{\mathsf{A}}_n$ by an additional announcement axiom, the axioms and rules for common knowledge, as well as a rule for common knowledge after an announcement.

Definition 6. The theory $\mathsf{S5}^{\mathsf{C,A}}_n$ is defined to be $\mathsf{S5}^{\mathsf{A}}_n$ augmented by the following axioms and rules.

(A6) $\alpha \wedge [\alpha]\beta \to ([\alpha][\beta]\gamma \leftrightarrow [\alpha \wedge [\alpha]\beta]\gamma)$,
(C) $\mathsf{C}_G\alpha \to \mathsf{E}_G(\alpha \wedge \mathsf{C}_G\alpha)$,

$$(\mathsf{IND.1})\frac{\alpha \to \mathsf{E}_G(\alpha \wedge \beta)}{\alpha \to \mathsf{C}_G\beta}, \qquad (\mathsf{IND.2})\frac{\alpha \to [\beta]\gamma \qquad \alpha \wedge \beta \to \mathsf{E}_G(\beta \to \alpha)}{\alpha \wedge \beta \to [\beta]\mathsf{C}_G\gamma}.$$

Axiom (A6) is provable in $\mathsf{S5}^{\mathsf{A}}_n$, see Lemma 7. However, this is not the case for $\mathsf{S5}^{\mathsf{C,A}}_n$ since there is no translation available that eliminates the announcement operators. The rule (IND.2) is a slight modification of the action rule from Baltag et al. in [2,3].

To show soundness and completeness, we will need the following notion of a G^α-path.

Definition 7. Let $\alpha \in \mathcal{L}_n^{\mathsf{C,A}}$, $\mathsf{K} \in \mathcal{K}_n^{eq}$, $\emptyset \neq G \subseteq \mathcal{A}$, and $s, t \in |\mathsf{K}|$. Then we say that there is a G^α-path from s to t, if there are states $u_1, u_2, \ldots, u_k \in |\mathsf{K}|$ s.t. $sR_Gu_1R_Gu_2R_G \ldots R_Gu_k$, $u_k = t$, and for all j, $1 \leq j \leq k \Rightarrow \mathsf{K}, u_j \models \alpha$.

It is easy to see that $\mathsf{K}, s \models \alpha$ implies

$$\mathsf{K}, s \models [\alpha]\mathsf{C}_G\beta \quad \text{iff}$$
$$\text{for every } t \text{, there is a } G^\alpha\text{-path from } s \text{ to } t \Rightarrow \mathsf{K}, t \models [\alpha]\beta.$$

Lemma 12. *The system* $\mathsf{S5}_n^{\mathsf{C,A}}$ *is sound with respect to* $\mathcal{K}_n^{eq}$*, i.e. for all* $\mathcal{L}_n^{\mathsf{C,A}}$ *formulas* α *we have*

$$\mathsf{S5}_n^{\mathsf{C,A}} \vdash \alpha \quad \Rightarrow \quad \mathcal{K}_n^{eq} \models \alpha.$$

Proof. The proof is by induction on the length of the derivation. In the induction step, soundness of the rule (IND.2) is proved as follows. Suppose, the formula $\alpha \wedge \beta \rightarrow [\beta]\mathsf{C}_G\gamma$ has been derived with the rule (IND.2). Then, by induction hypothesis, we know that the formulas (a) $\alpha \rightarrow [\beta]\gamma$ and (b) $\alpha \wedge \beta \rightarrow \mathsf{E}_G(\beta \rightarrow \alpha)$ are valid. Now, take any Kripke-world K, s such that $\mathsf{K}, s \models \alpha \wedge \beta$. Using (b) we get that $\mathsf{K}, t \models \alpha$ in every world $t \in |\mathsf{K}|$ which is reachable on a G^β-path from s. But then, by (a), we have that $\mathsf{K}, t \models [\beta]\gamma$ in every t reachable on a G^β-path from s. This implies, by definition of the announcement semantics, that $\mathsf{K}, s \models [\beta]\mathsf{C}_G\gamma$, and we are done. □

Lemma 13. *For all* $\mathcal{L}_n^{\mathsf{C,A}}$ *formulas* α*,* β*, and* γ *we have*

$$\mathsf{S5}_n^{\mathsf{C,A}} \vdash \alpha \wedge \mathsf{C}_G(\alpha \rightarrow [\alpha]\beta) \rightarrow [\alpha]\mathsf{C}_G\beta,$$
$$\mathsf{S5}_n^{\mathsf{C,A}} \vdash \alpha \leftrightarrow \beta \;\Rightarrow\; \mathsf{S5}_n^{\mathsf{C,A}} \vdash [\alpha]\gamma \leftrightarrow [\beta]\gamma.$$

Since the formula $\alpha \wedge [\alpha]\mathsf{C}_G\beta \rightarrow \mathsf{C}_G(\alpha \rightarrow [\alpha]\beta)$ is not valid (compare with (A4)), there is no translation available that could be used to give an easy completeness proof of $\mathsf{S5}_n^{\mathsf{C,A}}$. Thus we have to employ maximal consistent sets to show completeness. Our argument is the same as the one presented in [15] for the logic of common knowledge except that we have more cases in the truth lemma. We start by defining the closure $cl(\alpha)$ of a formula α.

Definition 8. For all $\mathcal{L}_n^{\mathsf{C,A}}$ formulas α, $sub^+(\alpha)$ is the smallest set which satisfies the following conditions.

1. $\alpha \in sub^+(\alpha)$.
2. If $\neg\beta \in sub^+(\alpha)$, then $\beta \in sub^+(\alpha)$.
3. If $\beta \wedge \gamma \in sub^+(\alpha)$, then $\beta, \gamma \in sub^+(\alpha)$.
4. If $K_i\beta \in sub^+(\alpha)$, then $\beta \in sub^+(\alpha)$.
5. If $\mathsf{C}_G\beta \in sub^+(\alpha)$, then $\mathsf{E}_G\beta, \mathsf{E}_G\mathsf{C}_G\beta \in sub^+(\alpha)$.
6. If $[\beta]p \in sub^+(\alpha)$, then $\beta, p \in sub^+(\alpha)$.
7. If $[\beta]\neg\gamma \in sub^+(\alpha)$, then $[\beta]\gamma, \neg\gamma \in sub^+(\alpha)$.
8. If $[\beta](\gamma \wedge \delta) \in sub^+(\alpha)$, then $[\beta]\gamma, [\beta]\delta, \gamma \wedge \delta \in sub^+(\alpha)$.
9. If $[\beta]K_i\gamma \in sub^+(\alpha)$, then $K_i\gamma, K_i(\beta \rightarrow [\beta]\gamma) \in sub^+(\alpha)$.
10. If $[\beta]\mathsf{C}_G\gamma \in sub^+(\alpha)$, then $\mathsf{C}_G\gamma, \mathsf{E}_G(\beta \rightarrow [\beta]\mathsf{C}_G\gamma) \in sub^+(\alpha)$.
11. If $[\beta][\gamma]\delta \in sub^+(\alpha)$, then $[\beta]\delta, [\gamma]\delta, [\beta \wedge [\beta]\gamma]\delta \in sub^+(\alpha)$.

The closure of α is defined by $cl(\alpha) := sub^+(\alpha) \cup \{\neg\beta \mid \beta \in sub^+(\alpha)\}$.

Observe, that for $\alpha = [\beta]\gamma$, we immediately get both $sub^+(\beta) \subseteq sub^+(\alpha)$ and $sub^+(\gamma) \subseteq sub^+(\alpha)$.

Lemma 14. *For every $\mathcal{L}_n^{\mathsf{C,A}}$ formula α, the set $cl(\alpha)$ is finite.*

Note that $cl(\alpha)$ is not closed under complements: for a given $\beta \in cl(\alpha)$ we need not have $\neg\beta \in cl(\alpha)$. However, there is always a formula $\sim\!\beta \in cl(\alpha)$, which is equivalent to $\neg\beta$.

Definition 9. For every $\mathcal{L}_n^{\mathsf{C,A}}$ formula α, the canonical structure of α is defined by $\mathsf{K}_\alpha = (con(\alpha), R_1, \ldots, R_n, V)$, where

$$\begin{aligned} con(\alpha) &:= \{U \cap cl(\alpha) \mid U \text{ is a maximal } \mathsf{S5}_n^{\mathsf{C,A}}\text{-consistent set}\}, \\ R_i &:= \{(X, Y) \mid X/K_i = Y/K_i\}, \\ V(p) &:= \{X \mid p \in X\}, \end{aligned}$$

and X/K_i denotes the set $\{\beta \mid K_i\beta \in X\}$.

For the notion of a maximal consistent set with respect to a theory, see for instance [15]. Observe, that K_α is in $\mathcal{K}_n^{eq}$ for all $\mathcal{L}_n^{\mathsf{C,A}}$ formulas α.

Lemma 15 (Truth Lemma). *Let α be an arbitrary $\mathcal{L}_n^{\mathsf{C,A}}$ formula and K_α be its canonical structure. Then we have for all $\beta \in cl(\alpha)$ and all $X \in con(\alpha)$,*

$$\beta \in X \quad \Leftrightarrow \quad \mathsf{K}_\alpha, X \models \beta.$$

Proof. We prove this lemma by induction on β, and we omit the cases where β does not begin with an announcement operator because they are standard. Hence assume β begins with an announcement operator. We show by side induction on δ that for any announcement operator $[\gamma]$ the claim holds for $\beta = [\gamma]\delta$. The base case is immediate, we have $[\gamma]p \in X$ iff $\mathsf{K}_\alpha, X \models [\gamma]p$ by axiom $(\mathsf{A1})$ and Definition 9. The cases $\beta = [\gamma]\neg\varphi$ and $\beta = [\gamma](\varphi \wedge \psi)$ are also straightforward.

Now let $\beta = [\gamma]K_i\varphi$. The interesting step is the case $\mathsf{K}_\alpha, X \models \gamma$ in the direction from right to left. First, one can show that the set

$$Y := \{K_i\xi \mid K_i\xi \in X\} \cup \{\neg K_i\xi \mid \neg K_i\xi \in X\} \cup \{\gamma, \neg[\gamma]\varphi\}$$

is inconsistent, using the induction hypothesis for γ and $[\gamma]\varphi$. Using the axioms (T), $(\mathsf{4})$, and $(\mathsf{5})$, we can prove the existence of a set

$$Z := \{K_i\xi_1, \ldots, K_i\xi_k, \neg K_i\xi_{k+1}, \ldots, \neg K_i\xi_l\} \subseteq Y$$

s.t. the set $Z \cup \{\neg K_i(\gamma \to [\gamma]\varphi)\}$ is inconsistent. Since $Z \subseteq X$ and $X \in con(\alpha)$, we have $K_i(\gamma \to [\gamma]\varphi) \in X$. Using axiom $(\mathsf{A4})$ and again the induction hypothesis for γ we get that $[\gamma]K_i\varphi \in X$.

For $\beta = [\gamma]\mathsf{C}_G\varphi$, the case $\mathsf{K}_\alpha, X \models \gamma$ in the direction from right to left is the challenging part of the proof. For every $Y \in con(\alpha)$ and for the set $\mathcal{B} := \{Z \in con(\alpha) \mid \mathsf{K}_\alpha, Z \models [\gamma]\mathsf{C}_G\varphi\}$, we define the formulas

$$\psi_Y := \bigwedge_{\xi \in Y} \xi \qquad \text{and} \qquad \chi_{\mathcal{B}} := \bigvee_{Y \in \mathcal{B}} \psi_Y .$$

It is not hard to show, that the formulas $\chi_{\mathcal{B}} \to [\gamma]\varphi$ and $\chi_{\mathcal{B}} \wedge \gamma \to \mathsf{E}_G(\gamma \to \chi_{\mathcal{B}})$ are both derivable. Applying the rule $(\mathsf{IND.2})$ we get $\vdash \chi_{\mathcal{B}} \wedge \gamma \to [\gamma]\mathsf{C}_G\varphi$. Since $X \in \mathcal{B}$ by assumption, and $\gamma \in X$ by induction hypothesis, we have $\vdash \psi_X \to [\gamma]\mathsf{C}_G\varphi$. Hence, we know that $[\gamma]\mathsf{C}_G\varphi \in X$.

The last case of our induction is $\beta = [\gamma][\varphi]\psi$. Here we distinct the three cases $\sim\gamma \in X$, $\sim[\gamma]\varphi \in X$, and $\gamma, [\gamma]\varphi \in X$. All three cases are straightforward using the axioms $(\mathsf{A5})$ and $(\mathsf{A6})$, respectively. Observe, that we need the second induction hypothesis in the third case. □

Due to the Truth Lemma, we know that every $\mathsf{S5}_n^{\mathsf{C,A}}$-consistent formula is satisfiable in $\mathcal{K}_n^{eq}$, thus we have completeness.

Theorem 1. *The system $\mathsf{S5}_n^{\mathsf{C,A}}$ is complete with respect to $\mathcal{K}_n^{eq}$, i.e. for all $\mathcal{L}_n^{\mathsf{C,A}}$ formulas α we have*

$$\mathcal{K}_n^{eq} \models \alpha \quad \Rightarrow \quad \mathsf{S5}_n^{\mathsf{C,A}} \vdash \alpha.$$

As an immediate consequence of our completeness proof, we get the finite model property and that the satisfiability problem is decidable.

We now extend the definition of announcement-resistance to the richer language. An $\mathcal{L}_n^{\mathsf{C,A}}$ formula α is called announcement-resistant, if

$$\mathsf{S5}_n^{\mathsf{C,A}} \vdash \alpha \to [\beta]\alpha$$

for every $\mathcal{L}_n^{\mathsf{C,A}}$ formula β.

It is still true, that all $\mathcal{L}_0$ formulas as well as all provable $\mathcal{L}_n^{\mathsf{C,A}}$ formulas are announcement-resistant. In addition, if α and β are announcement-resistant, then so also are the formulas $\alpha \wedge \beta$, $\alpha \vee \beta$, $K_i\alpha$, and $\mathsf{C}_G\alpha$.

Lemma 16. *Let α be an announcement-resistant $\mathcal{L}_n^{\mathsf{C,A}}$ formula and $G \subseteq \mathcal{A}$ be a non-empty group of agents. Then for all $k \geq 1$ we have*

$$\mathsf{S5}_n^{\mathsf{C,A}} \vdash \alpha \to [\alpha]^k \mathsf{C}_G\alpha.$$

4 Relativized Common Knowledge

In this section, we will study operators for relativized common knowledge. We will present a sound and complete axiomatization for the logic of relativized common knowledge augmented by total public announcements.

The notion of relativized common knowledge is a generalized version of common knowledge. The language of $\mathcal{L}_n^{\mathsf{RC,A}}$ formulas is the language $\mathcal{L}_n^{\mathsf{A}}$ extended by the binary operator RC_G for every non-empty group $G \subseteq \mathcal{A}$ of agents. The validity for relativized common knowledge is defined by

$$\begin{aligned} &\mathsf{K}, s \models \mathsf{RC}_G(\alpha, \beta) \quad \text{iff} \\ &\qquad\qquad \text{for every } t\text{, there is a } G^\alpha\text{-path from } s \text{ to } t \Rightarrow \mathsf{K}, t \models \beta. \end{aligned}$$

It is obvious that common knowledge can be defined by means of relativized common knowledge via $\mathsf{C}_G\alpha := \mathsf{RC}_G(\top, \alpha)$. It can be shown that relativized common knowledge is strictly more expressive than common knowledge, see van Benthem et al. [5,6] for a proof. For instance, the until operator of temporal logic can be defined using the relativized common knowledge operator.

Definition 10. The theory $\mathsf{S5}_n^{\mathsf{RC,A}}$ is defined to be $\mathsf{S5}_n^{\mathsf{A}}$ augmented by the following axioms and rules.

$$\begin{array}{ll} (\mathsf{A7}) & \alpha \to ([\alpha]\mathsf{RC}_G(\beta,\gamma) \leftrightarrow \mathsf{RC}_G(\alpha \wedge [\alpha]\beta, [\alpha]\gamma)), \\ (\mathsf{RC}) & \mathsf{RC}_G(\alpha,\beta) \to \mathsf{E}_G(\alpha \to \beta \wedge \mathsf{RC}_G(\alpha,\beta)), \end{array}$$

$$(\mathsf{IND.3})\frac{\alpha \to \mathsf{E}_G(\beta \to \alpha \wedge \gamma)}{\alpha \to \mathsf{RC}_G(\beta,\gamma)}.$$

Lemma 17. *For all $\mathcal{L}_n^{\mathsf{RC,A}}$ formulas α, β, and γ we have*

$$\begin{array}{l} \mathsf{S5}_n^{\mathsf{RC,A}} \vdash \mathsf{RC}_G(\alpha,\beta) \to (\alpha \to \beta), \\ \mathsf{S5}_n^{\mathsf{RC,A}} \vdash \mathsf{RC}_G(\alpha,\beta \to \gamma) \to (\mathsf{RC}_G(\alpha,\beta) \to \mathsf{RC}_G(\alpha,\gamma)), \\ \mathsf{S5}_n^{\mathsf{RC,A}} \vdash \mathsf{E}_G(\alpha \to \beta \wedge \mathsf{RC}_G(\alpha,\beta)) \to \mathsf{RC}_G(\alpha,\beta), \\ \mathsf{S5}_n^{\mathsf{RC,A}} \vdash \alpha \ \Rightarrow\ \mathsf{S5}_n^{\mathsf{RC,A}} \vdash \mathsf{RC}_G(\beta,\alpha). \end{array}$$

Theorem 2. *The system $\mathsf{S5}_n^{\mathsf{RC,A}}$ is sound and complete with respect to $\mathcal{K}_n^{eq}$, i.e. for all $\mathcal{L}_n^{\mathsf{RC,A}}$ formulas α we have*

$$\mathsf{S5}_n^{\mathsf{RC,A}} \vdash \alpha \quad \Leftrightarrow \quad \mathcal{K}_n^{eq} \models \alpha.$$

Proof. Soundness is proved as usual. The completeness proof for relativized common knowledge without announcement operators is similar to the proof of Theorem 1. The full system $\mathsf{S5}_n^{\mathsf{RC,A}}$ can now be treated by providing a translation which eliminates the announcements. Such a translation can easily be defined: simply observe that the axioms (A5) and (A7) imply

$$\mathsf{S5}_n^{\mathsf{RC,A}} \vdash [\alpha]\mathsf{RC}_G(\beta,\gamma) \leftrightarrow (\neg\alpha \wedge \mathsf{RC}_G(\beta,\gamma)) \vee (\alpha \wedge \mathsf{RC}_G(\alpha \wedge [\alpha]\beta, [\alpha]\gamma))$$

(compare with the case for $[\alpha]K_i\beta$ in Lemma 2). Making use of that translation and of the completeness for the system without announcements, we can show completeness of $\mathsf{S5}_n^{\mathsf{RC,A}}$ as in Lemma 8. □

5 Discussion

In the classical setting, public announcements are considered to be truthful. Thus, a specific announcement cannot happen in all possible worlds. If an announcement α is false in a given world, then the corresponding update action $[\alpha]$ cannot be performed, formally $[\alpha]\bot$ holds in that world.

We have presented a semantics, and corresponding formal systems, in which an announcement can happen in every possible world. Thus, announcements are total, that is the formula $\neg[\alpha]\bot$ is valid.

If an announcement is true, then an agent's knowledge gets updated; if it is false, then the agent's knowledge remains unchanged. In the classical setting, a false announcement cannot happen. In our approach it can happen but does not change an agent's knowledge. Thus we have a semantics which internalizes the idea of doing nothing when a false announcement is made.

This is a step towards a system in which an agent's knowledge and beliefs are considered. In such a setting, every announcement has to be performed since it may change an agent's beliefs even if its knowledge will not be affected. Thus it is necessary that announcements are total if we talk about both knowledge and beliefs of an agent.

In the present approach we use an alethic criterion (namely wether an announcement is true or false) to decide wether an agent's knowledge should be updated or not. Of course, one may argue about such a criterion. However, since an agent's knowledge has to be true by the knowledge axiom (T), it should only be updated by announcements that are indeed true. Therefore, unlike in the case of belief updates, it is not possible to use an epistemic criterion. The approach from [13], for instance, would lead to false knowledge by already accepting an announcement if it is consistent with the previous knowledge.

There are already languages available in which our form of announcements can be expressed. However, they are very expressive: non-deterministic choice, truthful public announcements, and PDL-like test operators [2] are needed in order to model our system. We have presented a lean and direct syntax and semantics for total public announcements which leads to new concepts such as announcement-resistant formulas. An interesting open question is to give a syntactic characterization of this class of formulas. As already mentioned, further work will also be done to investigate systems for updating knowledge and beliefs of an agent.

Acknowledgement

We would like to thank the anonymous referees for the long list of comments which helped to improve the paper.

References

1. Plaza, J.A.: Logics of public communications. In Emrich, M., Pfeifer, M., Hadzikadic, M., Ras, Z., eds.: Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems. (1989) 201–216
2. Baltag, A., Moss, L.S.: Logics for epistemic programs. Synthese **139**(2) (2004) 165–224
3. Baltag, A., Moss, L.S., Solecki, S.: The logic of public announcements, common knowledge, and private suspicions. In: TARK '98: Proceedings of the 7th conference on Theoretical aspects of rationality and knowledge, Morgan Kaufmann Publishers (1998) 43–56
4. van Benthem, J.: One is a lonely number: on the logic of communication. In Chatzidakis, Z., Koepke, P., Pohlers, W., eds.: Logic Colloquium '02. Number 27 in Lecture Notes in Logic, ASL and A. K. Peters (2006) 96–129

5. van Benthem, J., van Eijck, J., Kooi, B.: Common knowledge in update logics. In: TARK '05: Proceedings of the 10th conference on Theoretical aspects of rationality and knowledge, National University of Singapore (2005) 253–261
6. van Benthem, J., van Eijck, J., Kooi, B.: Logics of communication and change. Information and Computation **204**(11) (2006) 1620–1662
7. van Benthem, J., Kooi, B.: Reduction axioms for epistemic actions. In Schmidt, R., Pratt-Hartmann, I., Reynolds, M., Wansing, H., eds.: AiML '04: Proceedings of Advances in Modal Logic 5. Number UMCS-04-9-1 in Technical Report Series, University of Manchester (2004) 197–211
8. van Ditmarsch, H.: Knowledge games. PhD thesis, University of Groningen (2000) ILLC Dissertation Series 2000-06.
9. van Ditmarsch, H.: The russian cards problem. Studia Logica **75**(4) (2003) 31–62
10. van Ditmarsch, H., van der Hoek, W., Kooi, B.: Dynamic epistemic logic. Volume 337 of Synthese Library. Springer (2007)
11. van Ditmarsch, H., Kooi, B.: The secret of my success. Synthese **151**(2) (2005) 202–232
12. Renne, B.: Bisimulation and public announcements in logics of evidence-based knowledge. In Artemov, S., Parikh, R., eds.: ESSLLI '06: Proceedings of the European Summer School in Logic, Language and Information, Workshop on Rationality and Knowledge, Association for Logic, Language and Information (2006) 112–123
13. Steiner, D.: A system for consistency preserving belief change. In Artemov, S., Parikh, R., eds.: ESSLLI '06: Proceedings of the European Summer School in Logic, Language and Information, Workshop on Rationality and Knowledge, Association for Logic, Language and Information (2006) 133–144
14. Gerbrandy, J., Groeneveld, W.: Reasoning about information change. Journal of Logic, Language and Information **6**(2) (1997) 147–169
15. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning about Knowledge. MIT Press (1995)

Author Index