

Lecture Notes in Artificial Intelligence

2507

Subseries of Lecture Notes in Computer Science

Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science

Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer

Berlin

Heidelberg

New York

Barcelona

Hong Kong

London

Milan

Paris

Tokyo

Guilherme Bittencourt Geber L. Ramalho (Eds.)

Advances in Artificial Intelligence

16th Brazilian Symposium on Artificial Intelligence, SBIA 2002
Porto de Galinhas/Recife, Brazil, November 11-14, 2002
Proceedings



Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Guilherme Bittencourt
Universidade Federal de Santa Catarina
Departamento de Automação e Sistemas
88040-900 Florianópolis, SC, Brazil
E-mail: gb@das.ufsc.br

Geber L. Ramalho
Universidade Federal de Pernambuco
Centro de Informática
Cx. Postal 7851, 50732-970 Recife, PE, Brazil
E-mail: glr@cin.ufpe.br

Cataloging-in-Publication Data applied for

Bibliographic information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliographie;
detailed bibliographic data is available in the Internet at <<http://dnd.ddb.de>>.

CR Subject Classification (1998): I.2, F.4.1, H.2.8

ISSN 0302-9743

ISBN 3-540-00124-7 Springer-Verlag Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

Springer-Verlag Berlin Heidelberg New York,
a member of BertelsmannSpringer Science+Business Media GmbH

<http://www.springer.de>

© Springer-Verlag Berlin Heidelberg 2002
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Da-TeX Gerd Blumenstein
Printed on acid-free paper SPIN: 10870871 06/3142 5 4 3 2 1 0

Preface

The biennial Brazilian Symposium on Artificial Intelligence (SBIA 2002) – of which this is the 16th event – is a meeting and discussion forum for artificial intelligence researchers and practitioners worldwide. SBIA is the leading conference in Brazil for the presentation of research and applications in artificial intelligence. The first SBIA was held in 1984, and since 1995 it has been an international conference, with papers written in English and an international program committee, which this year was composed of 45 researchers from 13 countries. SBIA 2002 was held in conjunction with the VII Brazilian Symposium on Neural Networks (SBRN 2002). SBRN 2002 focuses on neural networks and on other models of computational intelligence.

SBIA 2002, supported by the Brazilian Computer Society (SBC), was held in Porto de Galinhas/Recife, Brazil, 11–14 November 2002. The call for papers was very successful, resulting in 146 papers submitted from 18 countries. A total of 39 papers were accepted for publication in the proceedings. We would like to thank the SBIA 2002 sponsoring organizations, CNPq, Capes, and CESAR, and also all the authors who submitted papers. In particular, we would like to thank the program committee members and the additional referees for the difficult task of reviewing and commenting on the submitted papers. We are also very grateful to our colleagues who provided invaluable organizational support and to Richard van de Stadt, the author of the Cyberchair system, a free software under GNU General Public License, that supported all the review process and the preparation of the proceedings.

November 2002

Guilherme Bittencourt
Geber Ramalho

Organization

SBIA 2002 was held in conjunction with the VII Brazilian Symposium on Neural Networks (SBRN 2002). Both conferences were organized by AI research groups that belong to the Federal University of Pernambuco.

Chair

Geber Ramalho (UFPE, Brazil)

Steering Committee

Ana Teresa Martins (UFC, Brazil)
Guilherme Bittencourt (UFSC, Brazil)
Jaime Sichman (USP, Brazil)
Solange Rezende (USP, Brazil)

Organizing Committee

Jacques Robin (UFPE, Brazil)
Flávia Barros (UFPE, Brazil)
Francisco Carvalho (UFPE, Brazil)
Guilherme Bittencourt (UFSC, Brazil)
Patrícia Tedesco (UFPE, Brazil)
Solange Rezende (USP, Brazil)

Supporting Scientific Society

SBC Sociedade Brasileira de Computação

Program Committee

Guilherme Bittencourt (Chair)	Universidade Federal de Santa Catarina (Brazil)
Agnar Aamodt	Norwegian University of Science and Technology (Norway)
Alexis Drogoul	Université Paris VI (France)
Ana Lúcia Bazzan	Universidade Federal do Rio Grande do Sul (Brazil)
Ana Teresa Martins	Universidade Federal do Ceará (Brazil)
Andre Valente	Knowledge Systems Ventures (USA)
Carles Sierra	Institut d'Investigació en Intelligència Artificial (Spain)
Christian Lemaitre	Laboratorio Nacional de Informatica Avanzada (Mexico)
Cristiano Castelfranchi	Institute of Psychology of CNR (Italy)
Díbio Leandro Borges	PUC-PR (Brazil)
Donia Scott	University of Brighton (United Kingdom)
Eugênio Costa Oliveira	Universidade do Porto (Portugal)
Evandro de Barros Costa	Universidade Federal de Alagoas (Brazil)
Fábio Cozman	Universidade de São Paulo (Brazil)
Flávia Barros	Universidade Federal de Pernambuco (Brazil)
Francisco Carvalho	Universidade Federal de Pernambuco (Brazil)
Gabriel Pereira Lopes	Universidade Nova de Lisboa (Portugal)
Gabriela Henning	Universidad Nacional del Litoral (Argentina)
Geber Ramalho	Universidade Federal de Pernambuco (Brazil)
Gerhard Widmer	Austrian Research Institute for Artificial Intelligence (Austria)
Gerson Zaverucha	Universidade Federal do Rio de Janeiro (Brazil)
Helder Coelho	Universidade de Lisboa (Portugal)
Jacques Wainer	Universidade de Campinas (Brazil)
Jacques Robin	Universidade Federal de Pernambuco (Brazil)
Jacques Calmet	Universität Karlsruhe (Germany)
Jaime Sichman	Universidade de São Paulo (Brazil)
Kathy McKeown	Columbia University (USA)
Lluis Godo Lacasa	Artificial Intelligence Research Institute (Spain)
Luis Otávio Alvares	Universidade Federal do Rio Grande do Sul (Brazil)
Marcelo Ladeira	Universidade de Brasília (Brazil)
Maria Carolina Monard	Universidade de São Paulo (Brazil)
Michael Huhn	University of South Carolina (USA)
Nitin Indurkha	University of New South Wales (Australia)
Olivier Boissier	Ecole Nationale Supérieure des Mines de Saint-Etienne (France)
Pavel Brazdil	Universidade do Porto (Portugal)

VIII Organization

Pedro Paulo B. de Oliveira	Universidade Presbiteriana Mackenzie (Brazil)
Ramon Lopes de Mantaras	Institut d'Investigació en Intelligència Artificial (Spain)
Rosaria Conte	National Research Council (Italy)
Sandra Sandri	Instituto Nacional de Pesquisas Espaciais (Brazil)
Solange Rezende	Universidade de São Paulo (Brazil)
Stefano Cerri	LIRMM (France)
Tarcísio Pequeno	Universidade Federal do Ceará (Brazil)
Uma Garimella	AP State Council for Higher Education (India)
Vincent Corruble	LIP6, Université Paris VI (France)
Vera Lúcia Strube de Lima	PUC-RS (Brazil)

Sponsoring Organizations

The SBIA 2002 conference received financial support from the following institutions:

CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico
CAPES	Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CESAR	Centro de Estudos e Sistemas Avançados do Recife

Referees

Adam Kilgarriff	Germano C. Vasconcelos
Alipio Jorge	Gina M.B. Oliveira
Alneu de Andrade Lopes	Gustavo Alberto Giménez Lugo
Ana Maria Monteiro	Gustavo Enrique de A.P. Alves Batista
Ana Paula Rocha	Jaqueline Brigladori Pugliesi
Anna H.R. Costa	Joao Carlos Pereira da Silva
Augusto Cesar Pinto Loureiro da Costa	Joaquim Costa
Basilis Gidas	Jomi Fred Hübner
Carlos Soares	José Augusto Baranauskas
Caroline Varaschin Gasperin	João Luis Pinto
Dante Augusto Couto Barone	Kees van Deemter
Diogo Lucas	Kelly Christine C.S. Fernandes
Edson Augusto Melanda	Lucia Helena Machado Rino
Edward Hermann Haeusler	Luis Antunes
Fernando Carvalho	Luis Moniz
Fernando Gomide	Luis Torgo
Fernando de Carvalho Gomes	Mara Abel
Francisco Tavares	Marcelino Pequeno
Frederico Luiz Gonçalves de Freitas	Marco Aurelio C. Pacheco

Marcos Ferreira de Paula
 Maria Benedita Malheiro
 Mario Benevides
 Marta Mattoso
 Maurício Marengoni
 Maxime Morge
 Nicandro Cruz
 Nizam Omar
 Nuno Correia
 Nuno Marques
 Patricia Tedesco

Paulo Cortez
 Paulo Quaresma
 Pavel Petrovic
 Rafael H. Bordini
 Renata Vieira
 Rita A. Ribeiro
 Riverson Rios
 Rosa M. Vicari
 Sheila Veloso
 Teresa Bernarda Ludermir
 Tore Amble

Table of Contents

Theoretical and Logical Methods

On Special Functions and Theorem Proving in Logics for 'Generally'	1
<i>Sheila R. M. Veloso and Paulo A. S. Veloso</i>	
First-Order Contextual Reasoning	11
<i>Laurent Perrussel</i>	
Logics for Approximate Reasoning:	
Approximating Classical Logic "From Above"	21
<i>Marcelo Finger and Renata Wassermann</i>	
Attacking the Complexity of Prioritized Inference Preliminary Report	31
<i>Renata Wassermann and Samir Chopra</i>	
A New Approach to the Identification Problem	41
<i>Carlos Brito</i>	
Towards Default Reasoning through MAX-SAT	52
<i>Berilhes Borges Garcia and Samuel M. Brasil, Jr.</i>	

Autonomous Agents and Multi-agent Systems

Multiple Society Organisations and Social Opacity:	
When Agents Play the Role of Observers	63
<i>Nuno David, Jaime Simão Sichman, and Helder Coelho</i>	
Altruistic Agents in Dynamic Games	74
<i>Eduardo Camponogara</i>	
Towards a Methodology for Experiments with Autonomous Agents	85
<i>Luis Antunes and Helder Coelho</i>	
How Planning Becomes Improvisation? – A Constraint Based Approach for Director Agents in Improvisational Systems	97
<i>Márcia Cristina Moraes and Antônio Carlos da Rocha Costa</i>	
Extending the Computational Study of Social Norms with a Systematic Model of Emotions	108
<i>Ana L. C. Bazzan, Diana F. Adamatti, and Rafael H. Bordini</i>	
A Model for the Structural, Functional, and Deontic Specification of Organizations in Multiagent Systems	118
<i>Jomi Fred Hübner, Jaime Simão Sichman, and Olivier Boissier</i>	
The Queen Robots: Behaviour-Based Situated Robots	
Solving the N-Queens Puzzle	129
<i>Paulo Urbano, Luís Moniz, and Helder Coelho</i>	

The Conception of Agents as Part of a Social Model of Distance Learning	140
<i>João Luiz Jung, Patrícia Augustin Jaques, Adja Ferreira de Andrade, and Rosa Maria Vicari</i>	
Emotional Valence-Based Mechanisms and Agent Personality	152
<i>Eugénio Oliveira and Luís Sarmento</i>	
Simplifying Mobile Agent Development through Reactive Mobility by Failure	163
<i>Alejandro Zunino, Marcelo Campo, and Cristian Mateos</i>	
Dynamic Social Knowledge: The Timing Evidence	175
<i>Augusto Loureiro da Costa and Guilherme Bittencourt</i>	

Machine Learning

Empirical Studies of Neighborhood Shapes in the Massively Parallel Diffusion Model	185
<i>Sven E. Eklund</i>	
Ant-ViBRA: A Swarm Intelligence Approach to Learn Task Coordination	195
<i>Reinaldo A. C. Bianchi and Anna H. R. Costa</i>	
Automatic Text Summarization Using a Machine Learning Approach	205
<i>Joel Larocca Neto, Alex A. Freitas, and Celso A. A. Kaestner</i>	
Towards a Theory Revision Approach for the Vertical Fragmentation of Object Oriented Databases	216
<i>Flavia Cruz, Fernanda Baião, Marta Mattoso, and Gerson Zaverucha</i>	
Speeding up Recommender Systems with Meta-prototypes	227
<i>Byron Bezerra, Francisco de A.T. de Carvalho, Geber L. Ramalho, and Jean-Daniel Zucker</i>	
ActiveCP: A Method for Speeding up User Preferences Acquisition in Collaborative Filtering Systems	237
<i>Ivan R. Teixeira, Francisco de A.T. de Carvalho, Geber L. Ramalho, and Vincent Corruble</i>	
Making Recommendations for Groups Using Collaborative Filtering and Fuzzy Majority	248
<i>Sérgio R. de M. Queiroz, Francisco de A.T. de Carvalho, Geber L. Ramalho, and Vincent Corruble</i>	

Knowledge Discovery and Data Mining

Mining Comprehensible Rules from Data with an Ant Colony Algorithm ..	259
<i>Rafael S. Parpinelli, Heitor S. Lopes, and Alex A. Freitas</i>	

Learning in Fuzzy Boolean Networks – Rule Distinguishing Power	270
<i>José A.B. Tomé</i>	
Attribute Selection with a Multi-objective Genetic Algorithm	280
<i>Gisele L. Pappa, Alex A. Freitas, and Celso A.A. Kaestner</i>	
Applying the Process of Knowledge Discovery in Databases to Identify Analysis Patterns for Reuse in Geographic Database Design ...	291
<i>Carolina Silva, Cirano Iochpe, and Paulo Engel</i>	
Lithology Recognition by Neural Network Ensembles	302
<i>Rafael Valle dos Santos, Fredy Artola, Sérgio da Fontoura, and Marley Vellasco</i>	

Evolutionary Computation and Artificial Life

2-Opt Population Training for Minimization of Open Stack Problem	313
<i>Alexandre César Muniz de Oliveira and Luiz Antonio Nogueira Lorena</i>	
Grammar-Guided Genetic Programming and Automatically Defined Functions	324
<i>Ernesto Rodrigues and Aurora Pozo</i>	
An Evolutionary Behavior Tool for Reactive Multi-agent Systems	334
<i>Andre Zanki Cordenonsi and Luis Otavio Alvares</i>	
Controlling the Population Size in Genetic Programming	345
<i>Eduardo Spinosa and Aurora Pozo</i>	

Uncertainty

The Correspondence Problem under an Uncertainty Reasoning Approach	355
<i>José Demisio Simões da Silva and Paulo Ouverá Simoni</i>	
Random Generation of Bayesian Networks	366
<i>Jaime S. Ide and Fabio G. Cozman</i>	
Evidence Propagation in Credal Networks: An Exact Algorithm Based on Separately Specified Sets of Probability	376
<i>José Carlos F. da Rocha and Fabio G. Cozman</i>	
Restoring Consistency in Systems of Fuzzy Gradual Rules Using Similarity Relations	386
<i>Isabela Drummond, Lluís Godo, and Sandra Sandri</i>	

Natural Language Processing

Syntactic Analysis for Ellipsis Handling in Coordinated Clauses	397
<i>Ralph Moreira Maduro and Ariadne M. B. R. Carvalho</i>	
Assessment of Selection Restrictions Acquisition	407
<i>Alexandre Agustini, Pablo Gamallo, and Gabriel P. Lopes</i>	

Author Index	417

On Special Functions and Theorem Proving in Logics for 'Generally'

Sheila R. M. Veloso¹ and Paulo A. S. Veloso¹

¹ Inst. Matemática and PESC/COPPE, UFRJ, Praça Eugênio Jardim, 6/apt. 501,
22061-040 Rio de Janeiro, RJ, Brazil
{sheila,veloso}@cos.ufrj.br

Abstract. Logics for 'generally' are intended to express some vague notions, such as 'generally', 'most', 'several', etc., by means of the new generalized quantifier ∇ and to reason about assertions with ∇ (important issues in Logic and in Artificial Intelligence). We introduce the ideas of special functions: generic and coherent ones. Generic functions (akin to Skolem functions) enable elimination of ∇ and coherent functions reduce consequence to the classical case. These devices permit using proof procedures and theorem provers for classical first-order logic to reason about assertions involving 'generally'.

1 Introduction

In this paper we provide a framework for theorem proving in logics for 'generally' based on special functions, which permit using proof procedures and theorem provers for classical first-order logic to reason about assertions involving 'generally'.

Some logics for 'generally' were introduced for handling assertions with vague notions, such as 'generally', 'most', 'several' [1]¹. Their expressive power is quite convenient and they have sound and complete deductive systems. This, however, still leaves open the question of theorem proving, namely theorem provers for them. We will show that special functions (generic functions, which are similar to Skolem functions, and coherent functions) allow one to use existing theorem provers (for classical first-order logic) for this task. The development will concentrate on ultrafilter logic [2], but its the main lines can be adapted to some other logics for 'generally'.²

¹ These logics are related to variants of default logic (and to belief revision): they have various common applications, as indicated by benchmark examples. They are however quite different logical systems, both technically (our logics are monotonic and conservative extensions of classical logic, in sharp contrast to non-monotonic approaches) and in terms of intended interpretations (our approach caters to a positive view, in the sense of representing 'generally' explicitly, rather than interpreting it as "in the absence of information to the contrary").

² For instance, filter logic (for 'most') and upward closed logic (for 'several'). The expressive power of our generalized quantifiers paves the way for other possible applications where it may be helpful: e. g. expressing some fuzzy concepts [3, 4].

This paper is structured as follows. The remainder of this section provides some motivations for logics for 'generally' and a brief overview of the main ideas. In section 2 we briefly review some logics for 'generally'. In section 3 we introduce the ideas of generic functions and then internalize them. Coherent functions are introduced in section 4 to complete the reduction of ultrafilter reasoning to first-order reasoning. In section 5, we will put together our results (and indicate how to adapt them to other logics for 'generally') to provide a framework where reasoning with 'generally' reduces to first-order reasoning with coherent functions. Section 6 contains some concluding remarks about our approach.

We now briefly examine some motivations underlying logics for 'generally'.

Assertions and arguments involving some vague notions occur often, not only in ordinary language, but also in some branches of science, where "modifiers", such as 'generally', 'rarely', 'most', 'several', etc., occur. For instance, one often encounters assertions such as "Bodies 'generally' expand when heated", "Birds 'generally' fly" and "Metals 'rarely' are liquid under ordinary conditions". Somewhat vague terms such as 'likely', 'prone', etc., are frequently used in everyday language. More elaborate expressions involving 'propensity' are often used as well.³ Such notions may also be useful in reporting experimental set-ups and results. Qualitative reasoning about such notions often occurs in everyday life.⁴ The assertions "Whoever likes sports watches Sport channel" and "Boys 'generally' like sports" appear to lead to "Boys 'generally' watch Sport channel".

Considering a universe of birds, we can express within classical first-order logic assertions such as "All birds fly" (by $\forall v F(v)$) and "Some birds fly" (by $\exists v F(v)$). But, what about vague assertions like "Several (or most) birds fly"?

We wish to express such assertions and reason about them in a precise manner. Extensions of first-order logic with an operator ∇ and axioms to characterize the vague notion expressed by ∇ provide logics for reasoning about some vague notions [1]. So, one can express "Birds 'generally' fly" by $\nabla v F(v)$.⁵ In this paper, we show that we can reason about such "generalized" assertions entirely within first-order logic by means of special functions: generic and coherent ones, the former (akin to Skolem functions) enabling elimination of ∇ and the latter reducing consequence to the classical case. These devices permit translating assertions with 'generally' to first-order counterparts about which we can reason by classical means.

We now give a brief overview of these ideas indicating how special (generic and coherent) functions can be used for proving theorems in logics for 'generally'.

When we say that "Birds 'generally' fly" ($\nabla v F(v)$) we are meaning that the set of flying birds is an 'important' set of birds (in the sense of being representative). We may consider a 'generic' bird as one that exhibits exactly the properties that birds generally posses, thus representing birds in general. So, we take a new constant symbol c and express that c is generic (with respect to flying) by $\nabla v F(v) \leftrightarrow F(c)$.

³ For instance, a physician may say that a patient's genetic background indicates a certain 'propensity', which makes him or her 'prone' to some ailments.

⁴ For instance, a medical doctor prescribes a treatment to a patient considering this treatment as appropriate to a typical patient with such symptoms.

⁵ We are considering a universe of birds. If there are other animals, we use sorts (the birds form a subsort of the universe); relativization (e. g. $\nabla v (B(v) \rightarrow F(v))$) does not express the intended meaning, due to properties of ∇ and $\rightarrow$ [2, 7].

We can extend this idea to formulas with free variables. For instance, let $L(x,y)$, stand for " x is taller than y ". Then, when we say that "people generally are taller than y " ($\nabla x L(x,y)$) we mean that the set of people taller than y is an 'important' set of people. We may consider a generic person as one that has exactly the properties that people generally have (e. g. being taller than y). So, we take a new function symbol f whose intended meaning is to associate to y a generic person. The genericity of $f(y)$ (with respect to being taller) is expressed by $\nabla x L(x,y) \leftrightarrow L(f(y),y)$.

In general, the occurrences of ∇ in a formula can be recursively eliminated in favor of generic functions giving a first-order formula from which the original one can be recovered. For example, $\exists y \nabla x L(x,y)$ corresponds to $\exists y L(f(y),y)$ and $\nabla x \exists y L(x,y)$ corresponds to $\exists y L(c,y)$, while $\nabla y \nabla x L(x,y)$ corresponds to $\nabla y L(f(y),y)$, which corresponds to $L(f(c),c)$. Note that the elimination is applied recursively to the smaller generalized subformulas of the formula.

One can use these ideas to reduce reasoning in generalized logics to classical reasoning with new function symbols and axioms, as we will now illustrate.

Let $L(x,y)$ stand for " x loves y ". Then, $\forall x \nabla y L(x,y)$ expresses "everybody loves people in general", $\exists x \nabla y L(x,y)$ expresses "somebody loves people in general" and "people generally love each other" can be expressed by $\nabla x \nabla y L(x,y)$.

a) From $\forall x \nabla y L(x,y)$ we infer $\forall x \exists y L(x,y)$ {"everybody loves someone"}: transform $\forall x \nabla y L(x,y)$ into $\forall x L(x,f(x))$ and use first-order logic.

b) From $\forall x \nabla y L(x,y)$ we infer $\nabla x \nabla y L(x,y)$: transform $\forall x \nabla y L(x,y)$ into $\forall x L(x,f(x))$ and $\forall x \nabla y L(x,y)$ into $L(c,f(c))$ and use first-order logic.

c) From $\nabla y L(b,y)$ {"Bill loves people in general"} we infer $\exists x \nabla y L(x,y)$: transform $\nabla y L(b,y)$ into $L(b,c)$ and $\exists x \nabla y L(x,y)$ into $\exists x L(x,f(x))$, use first-order logic and the coherence axiom $\forall x [L(b,c) \leftrightarrow L(b,f(x))]$.

In the sequel we shall examine this procedure for reducing ultrafilter consequence to classical first-order with coherent function. To show that this reduction procedure is sound and complete, we will establish the following facts: the extension by generic axioms is conservative, the generic axioms yield the coherence axioms and the extension of a coherent first-order theory by generic axioms is conservative.

2 Logics for 'Generally'

Logics for 'generally' extend classical first-order logic [5, 6] by a generalized quantifier ∇ , whose intended interpretation is 'generally' [1, 7]. In this section we briefly review some of these logics: syntax, semantics and axiomatics, illustrating some features with emphasis on ultrafilter logic.

Given a signature σ , we let $L(\sigma)$ be the usual first-order language (with equality $\approx$) of signature σ .⁶ We will use $L^\nabla(\sigma)$ for the extension of $L(\sigma)$ by the new operator ∇ . The formulas of $L^\nabla(\sigma)$ are built by the usual formation rules and a new variable-binding formation rule giving *generalized formulas*: for each variable v , if φ is a formula in $L^\nabla(\sigma)$ then so is $\nabla v \varphi$. Other syntactic notions, such as substitution ($\varphi[v:=t]$ or $\varphi(t)$) and substitutable, can be easily adapted.

Examples illustrating the expressive power of ∇ appear in section 1.

⁶ It is convenient to have a fixed, though arbitrary, ordering for the variables. In each list of variables, they will be listed according to this fixed ordering.

The semantic interpretation for 'generally' is provided by enriching first-order structures with families of subsets and extending the definition of satisfaction to ∇ .

A *modulated structure* $\mathbf{A}^{\mathcal{K}} = (\mathbf{A}, \mathcal{K})$ for signature σ consists of a usual structure $\mathbf{A}$ for σ together with a *complex*: a family $\mathcal{K}$ of subsets of the universe A of $\mathbf{A}$. We extend the usual definition of *satisfaction* of a formula in a structure under assignment $\underline{a}$ to its (free) variables, by using the *extension* $\mathbf{A}^{\mathcal{K}}[\varphi(\underline{a}, z)] := \{b \in A : \mathbf{A}^{\mathcal{K}} \models \varphi(\underline{a}, z)[\underline{a}, b]\}$, as follows:

for a formula $\nabla z \varphi(\underline{a}, z)$, we define $\mathbf{A}^{\mathcal{K}} \models \nabla z \varphi(\underline{a}, z)[\underline{a}]$ iff $\mathbf{A}^{\mathcal{K}}[\varphi(\underline{a}, z)]$ is in $\mathcal{K}$.

Satisfaction of a formula hinges only on the realizations assigned to its symbols.⁷

Other semantic notions, such as reduct and model ($\mathbf{A}^{\mathcal{K}} \models \Gamma$), are as usual [5, 6].

An *ultrafilter structure* is a modulated structure $\mathbf{A}^{\mathcal{U}} = (\mathbf{A}, \mathcal{U})$ whose complex is an ultrafilter over its universe. Now, the notion of *ultrafilter consequence* is as expected: $\Gamma \models^{\mathcal{U}} \tau$ iff $\mathbf{A}^{\mathcal{U}} \models \tau$ for every ultrafilter model $\mathbf{A}^{\mathcal{U}} \models \Gamma$ (likewise for validity).⁸

We now formulate deductive systems for our logics of 'generally' by adding schemata to a calculus for classical first-order logic. To set up a deductive system $\vdash^{\mathcal{U}}$ for ultrafilter logic we take a sound and complete deductive calculus for classical first-order logic, with Modus Ponens (MP) as the sole inference rule (as in [6]), and extend its set A of axiom schemata by adding a set $\Phi^{\mathcal{U}}$ of new axiom schemata (coding properties of ultrafilters), to form $A^{\mathcal{U}} := A \cup \Phi^{\mathcal{U}}$. This set $\Phi^{\mathcal{U}}$ consists of all the universal generalizations of the following six schemata (where φ, ψ and θ are formulas of language $L^{\nabla}(\sigma)$):

$$\begin{array}{ll} [\nabla\exists]: \nabla z \varphi \rightarrow \exists z \varphi; & [\nabla\alpha]: \nabla z \varphi \rightarrow \nabla w \varphi [z := w], \text{ for a new variable } w; \\ [\forall\nabla]: \forall z \varphi \rightarrow \nabla z \varphi; & [\rightarrow\nabla]: \forall z (\psi \rightarrow \theta) \rightarrow (\nabla z \psi \rightarrow \nabla z \theta); \\ [\nabla\wedge]: (\nabla z \psi \wedge \nabla z \theta) \rightarrow \nabla z (\psi \wedge \theta); & [\neg\nabla]: \neg \nabla z \varphi \rightarrow \nabla z \neg \varphi. \end{array}$$

These schemata express properties of ultrafilters, with $[\nabla\alpha]$ covering alphabetic variants.⁹ Other usual deductive notions, such as (maximal) consistent sets, witnesses and conservative extension [5, 6], can be easily adapted.¹⁰

We have sound and complete deductive systems for our logics (e.g. $\models^{\mathcal{U}} = \vdash^{\mathcal{U}}$), which are proper conservative extensions of classical first-order logic [2, 7, 8].¹¹

⁷ So, satisfaction for first-order formulas (without ∇) does not depend on the complex.

⁸ Other classes of modulated structures have as complexes filters (for 'most') and upward closed families (for 'several'). The behavior of ∇ is intermediate between those of the classical $\forall$ and $\exists$. But, the behavior of iterated ∇ 's contrasts with the commutativities of each classical $\forall$ and $\exists$: the formula $\nabla y \nabla x L(x, y) \rightarrow \nabla x \nabla y L(x, y)$ fails to be valid.

⁹ Some schemata, such as $[\forall\nabla]$ and $[\rightarrow\nabla]$, are derivable from the others; an independent axiomatization consists of $[\nabla\exists]$, $[\nabla\alpha]$, $[\nabla\wedge]$ and $[\neg\nabla]$. For upward closed logic, we take $\Phi^{\mathcal{C}} := \{[\nabla\exists], [\nabla\alpha], [\forall\nabla], [\rightarrow\nabla]\}$, and for filter logic $\Phi^{\mathcal{F}} := \Phi^{\mathcal{C}} \cup \{[\nabla\wedge]\}$ [7].

¹⁰ Derivations are first-order derivations from the schemata. Hence, we have monotonicity, and substitutivity of equivalents. In ultrafilter logic, we also have prenex forms: each formula is equivalent to a prefix of quantifiers followed by a quantifier-free matrix [2].

¹¹ Soundness is clear and completeness can be established by adapting Henkin's familiar proof for classical first-order logic. It is not difficult to see that we have conservative extensions of classical logic. These extensions are proper, because some sentences, such as $\exists u \nabla z u \approx z$, cannot be expressed without ∇ [7, 8].

3 Generic Functions and Axioms

We will now introduce the ideas of generic functions and then internalize them, so as to reason about them.

3.1 Generic Objects and Functions in a Structure

We first examine generic objects and functions in a modulated structure.

Consider a modulated structure $\mathbf{A}^{\mathcal{K}} = (\mathbf{A}, \mathcal{K})$ for a signature σ . Given a generalized sentence $\nabla z \varphi(z)$, by a *generic element for* $\nabla z \varphi(z)$ we mean an element $a \in A$ such that $\mathbf{A}^{\mathcal{K}} \models \nabla z \varphi(z)$ iff $\mathbf{A}^{\mathcal{K}} \models \varphi(z) [a]$. A generic object provides decisive local tests for generalized assertions [2, 7].¹²

It is natural to extend this idea to generalized formulas with free variables. Given a generalized formula $\nabla z \varphi(\underline{u}, z)$ of $L^{\nabla}(\sigma)$ with list $\underline{u}$ of m free variables, a *generic function for* $\nabla z \varphi(\underline{u}, z)$ is an m -ary function $f : A^m \rightarrow A$, assigning to each m -tuple $\underline{a} \in A^m$ a generic element $f(\underline{a}) \in A$: $\mathbf{A}^{\mathcal{K}} \models \nabla z \varphi(\underline{u}, z) [\underline{a}]$ iff $\mathbf{A}^{\mathcal{K}} \models \varphi(\underline{u}, z) [\underline{a}, f(\underline{a})]$.¹³

3.2 Generic Axioms

We will now formulate the idea of generic functions by means of axioms.

Given a signature σ , consider for each $n \in \mathbf{N}$, a new n -ary function symbol f_n not in σ , and form the expansion $\sigma[F] := \sigma \cup F$, obtained by adding the set $F := \{f_n : n \in \mathbf{N}\}$ of new function symbols. In this expanded signature we can express ideas of generic functions by means of sentences.

Given a generalized formula $\nabla z \varphi$ of $L^{\nabla}(\sigma)$ with list $\underline{u}$ of m free variables, the *generic axiom* $\varpi[f_m \backslash \nabla z \varphi]$ for $\nabla z \varphi$ is the universal closure of the formula $\nabla z \varphi \leftrightarrow \varphi[z := f_m(\underline{u})]$ of $L^{\nabla}(\sigma[F])$. We also extend this idea to sets of formulas. Given a set Ψ of generalized formulas of $L^{\nabla}(\sigma)$, the *generic axiom schema* for set Ψ of formulas is the set $\varpi\{F \backslash \Psi\}$ consisting of the generic axioms for every generalized formula $\nabla z \varphi$ in Ψ . When Ψ is the set of all the (generalized) formulas of signature σ , we will use $\varpi[F \backslash \sigma^{\nabla}] := \{\varpi\{F \backslash L^{\nabla}(\sigma)\}\}$ for the *generic axiom schema* for $L^{\nabla}(\sigma)$.

These axioms enable the elimination of the new quantifier ∇ in favor of generic functions, as illustrated in section 1. In general, with the generic axiom schema $\varpi[F \backslash \sigma^{\nabla}]$ for $L^{\nabla}(\sigma)$ we can eliminate ∇ : we transform each formula φ of $L^{\nabla}(\sigma[F])$ to a formula $\langle \varphi \rangle_{\nabla}$ of $L(\sigma[F])$, by replacing (inside-out) each subformula $\nabla z \psi(u_1, \dots, u_m, z)$ of φ by $\psi[z := f_m(u_1, \dots, u_m)]$.¹⁴

¹² For instance, consider an ultrafilter structure $\mathbf{A}^{\mathcal{U}}$, representing a world of animals where "Animals generally are voracious" and "Animals generally do not fly": $\mathbf{A}^{\mathcal{U}} \models \nabla u \vee V(u)$ and $\mathbf{A}^{\mathcal{U}} \models \nabla u \neg F(u)$. Then, voracious animals are generic for general voracity and non-flying animals are generic with respect to generally not flying.

¹³ The previous case of generic element amounts to a generic nullary function.

¹⁴ We can define the ∇ -elimination function $\langle \_ \rangle_{\nabla} : L^{\nabla}(\sigma[F]) \rightarrow L(\sigma[F])$ recursively by:

$\langle \varphi \rangle_{\nabla} := \langle \theta \rangle_{\nabla}[z := f_m(u_1, \dots, u_m)]$, for φ of the form $\nabla z \theta(u_1, \dots, u_m, z)$;

Lemma. Each formula φ of $L^\nabla(\sigma)$ can be transformed to formula $\langle\varphi\rangle_\nabla$ in $L(\sigma[F])$, so that $\mathfrak{M}[F\sigma^\nabla] \vdash^u \varphi \leftrightarrow \langle\varphi\rangle_\nabla$.

Proof outline. By induction on the structure of formula φ .¹⁵

This results shows that the generic axiom schema reduces the generalized quantifier ∇ to generic functions: $\mathfrak{M}[F\sigma^\nabla] \cup \Gamma \vdash^u \varphi$ iff $\mathfrak{M}[F\sigma^\nabla] \cup \langle\Gamma\rangle_\nabla \vdash^u \langle\varphi\rangle_\nabla$.

3.3 Extension by Generic Axioms

We now wish to see that we can add generic axioms conservatively.

For this purpose, we will show that an ultrafilter structure has functions that are generic for a finite set of generalized formulas. Call a set F of functions *generic for set Ψ* of formulas iff each generalized formula in Ψ has a generic function in set F .

Lemma. An ultrafilter structure $\mathcal{A}^{\mathcal{U}}$ has generic functions for each finite set Φ of formulas.

Proof outline. The finite intersection of sets in an ultrafilter is nonempty.¹⁶

Proposition. Given a set Γ of sentences of $L^\nabla(\sigma)$, for each set Ψ of (generalized) formulas of $L^\nabla(\sigma)$, $\Gamma\{F\backslash\Psi\} := \Gamma \cup \mathfrak{M}\{F\backslash\Psi\}$ is a conservative extension of Γ : $\Gamma\{F\backslash\Psi\} \vdash^u \tau$ iff $\Gamma \vdash^u \tau$ for each sentence τ of $L^\nabla(\sigma)$.

Proof outline. The assertion follows from the preceding lemma.¹⁷

Thus, we can always conservatively extend a given theory Γ so as to reduce the ultrafilter quantifier to generic functions: $\Gamma \vdash^u \tau$ iff $\mathfrak{M}[F\sigma^\nabla] \cup \langle\Gamma\rangle_\nabla \vdash^u \langle\tau\rangle_\nabla$. But, notice that the reasoning with generic functions will still occur within ultrafilter logic, since it relies on the generic axiom schema. To reduce this reasoning

$\langle\varphi\rangle_\nabla := Qz\langle\theta\rangle_\nabla$, for φ of the form $Qz\theta$, Q being $\forall$ or $\exists$;
 $\langle\varphi\rangle_\nabla := \neg\langle\theta\rangle_\nabla$, for φ of the form $\neg\theta$;
 $\langle\varphi\rangle_\nabla := \langle\psi\rangle_\nabla * \langle\theta\rangle_\nabla$, for φ of the form $\psi * \theta$, for a binary connective $*$;
 $\langle\varphi\rangle_\nabla := \varphi$, for φ of $L(\sigma[F])$.

¹⁵ For φ of the form $\nabla z\theta(\underline{u},z)$, we have $\langle\theta\rangle_\nabla$ in $L(\sigma[F])$ so that $\mathfrak{M}[F\sigma^\nabla] \vdash^u \theta \leftrightarrow \langle\theta\rangle_\nabla$ (by inductive hypothesis); so $\mathfrak{M}[F\sigma^\nabla] \vdash^u \nabla z\theta \leftrightarrow \nabla z\langle\theta\rangle_\nabla$. Now, $\mathfrak{M}[f_{\underline{u}}]\nabla z\theta \in \mathfrak{M}[F\sigma^\nabla]$ is $\forall \underline{u} (\nabla z\theta \leftrightarrow \theta [z := f_{\underline{u}}(\underline{u})])$. We thus have $\mathfrak{M}[F\sigma^\nabla] \vdash^u \nabla z\theta \leftrightarrow \langle\theta\rangle_\nabla [z := f_{\underline{u}}(\underline{u})]$, with the formula $\langle\nabla z\theta\rangle_\nabla = \langle\theta\rangle_\nabla [z := f_{\underline{u}}(\underline{u})]$ in $L(\sigma[F])$.

¹⁶ Let m be the maximum number of free variables occurring in the generalized formulas of Φ . For $0 \leq n \leq m$, we define $f_n : A^n \rightarrow A$ at $\underline{a} \in A^n$ as follows. Consider the set Φ_n of generalized formulas of Φ with at most n free variables and split it into two depending on satisfaction: ${}^+\Phi_n$ and ${}^-\Phi_n$. Since $\mathcal{U}$ is an ultrafilter, the finite intersection of the extensions $\mathcal{A}^{\mathcal{U}}[\psi(\underline{a},z)]$, for $\nabla z\psi \in {}^+\Phi_n$, and $\mathcal{A}^{\mathcal{U}}[\neg\theta(\underline{a},z)]$, for $\nabla z\theta \in {}^-\Phi_n$, is in $\mathcal{U}$, thus being nonempty, and we can select some b in it to set $f_n(\underline{a}) := b$. By construction, these functions f_n , for $0 \leq n \leq m$, are generic for the finite set Φ of formulas.

¹⁷ The lemma yields expansion of models for finite sets of formulas, so conservativeness.

completely to first-order logic, we need to replace the generic axiom schema by purely first-order schemata. We will examine this in the sequel.

4 Coherent Functions

We will now show how to complete the reduction of ultrafilter reasoning to first-order reasoning within a theory of coherent functions. We will first introduce the idea of coherent functions and then formulate it by first-order sentences, to reason with them.

A motivation for coherence comes from the question of replacing the generic axiom schema by first-order schemata. Consider the ∇ -elimination of the ultrafilter schemata. We can see that, but for $[\nabla \wedge]$ (and $[\rightarrow \nabla]$), each instance of these schemata becomes logically valid. Coherence will provide a way to handle schema $[\nabla \wedge]$.

4.1 Coherent Functions and Axioms

Recall that $\sigma[F] = \sigma \cup F$ is the expansion of signature σ by the set $F = \{ f_n : n \in \mathbf{N} \}$ of new function symbols: a new n -ary function symbol f_n , for each $n \in \mathbf{N}$.

We will introduce the idea of coherent functions. Section 1 shows a simple example: $\forall x [L(b,c) \leftrightarrow L(b,f(x))]$ connecting functions with two distinct arities: nullary c and unary f . For a list x, y and z of variables and formula ϕ with list x and z of free variables, selecting variable z we form the coherent axiom $[\phi/z]_{x,y}$ as the universal sentence $\forall x \forall y (\phi[z := f_1(x)] \leftrightarrow \phi[z := f_2(x,y)])$ of $L(\sigma[F])$.

Consider a list $\underline{v}$ of $m+1$ variables. Given a formula ϕ of $L(\sigma[F])$ whose list $\underline{u}$ of free variables is a sublist of $\underline{v}$ with length $n+1$, select a variable z in sublist $\underline{u}$ and form the *coherent axiom* $[\phi/z]_{\underline{v}}$ for formula ϕ with respect to variable z and list $\underline{v}$ as the universal sentence $\forall \underline{v} (\phi[z := f_n(\underline{u})] \leftrightarrow \phi[z := f_m(\underline{v})])$ of $L(\sigma[F])$. The *coherence axiom schema* $\Omega[\sigma/F]$ for $L(\sigma[F])$ consists of the schemata $[\phi/z]_{\underline{v}}$, for all formulas ϕ of $L(\sigma[F])$. Now, given a set Σ of sentences of $L(\sigma[F])$, we shall say that the set F of function symbols is *coherent* in theory Σ iff $\Sigma \vdash \Omega[\sigma/F]$.

The next result shows that the generic axioms yield the coherence axioms. Given a list $\underline{v}$ of variables, with length m , we let $\mathfrak{w}[f_m \setminus L^\nabla(\sigma)(\underline{v})]$ be the set consisting of the generic axioms for all the generalized formulas of $L^\nabla(\sigma)$ with list $\underline{v}$ of free variables.

Proposition. Given a list $\underline{v}$ of m variables, with length m , and a formula $\nabla z \phi$ of $L^\nabla(\sigma)$ whose list $\underline{u}$ of free variables is a sublist of $\underline{v}$, with length n , the coherent axiom $[\phi/z]_{\underline{v}}$, namely $\forall \underline{v} (\phi[z := f_n(\underline{u})] \leftrightarrow \phi[z := f_m(\underline{v})])$, follows from $\mathfrak{w}[f_m \setminus L^\nabla(\sigma)(\underline{v})]$ and $\mathfrak{w}[f_n \setminus \nabla z \phi]$: $\mathfrak{w}[f_m \setminus L^\nabla(\sigma)(\underline{v})] \cup \{ \mathfrak{w}[f_n \setminus \nabla z \phi] \} \vdash^u [\phi/z]_{\underline{v}}$.

Proof outline. The assertion follows from substitutivity of equivalents.¹⁸

Thus, the generic axioms yield the coherence axioms: $\mathfrak{w}[F \setminus \sigma] \vdash^u \Omega[\sigma/F]$.

¹⁸ We use the equivalence between ϕ and $\phi \wedge \underline{v} \approx \underline{v}$, where $\underline{v} \approx \underline{v}$ is the conjunction of $v_i \approx v_i$, for each v_i in $\underline{v}$.

4.2 Extension by Coherence Axioms

We will now argue that coherent functions can be regarded as generic functions, in the sense that a first-order theory with coherent functions has a conservative extension where they are generic functions.

We will proceed as follows. We will first show that a first-order structure with coherent functions of each arity can be expanded to an ultrafilter structure where the functions are generic in $L^\nabla(\sigma)$.

Lemma. Given a set Σ of sentences of $L(\sigma[F])$ where the set $F = \{f_n : n \in \mathbf{N}\}$ of function symbols is coherent, each first-order model $\mathbf{A} \models \Sigma$ can be expanded to an ultrafilter structure $\mathbf{A}^{\mathcal{U}} = (\mathbf{A}, \mathcal{U})$ satisfying the generic axiom schema $\mathfrak{W}[F\sigma^\nabla]$.

Proof outline. We can produce an ultrafilter by means of the coherent functions.¹⁹

Proposition. Given a set Σ of sentences of $L(\sigma[F])$ where the set $F = \{f_n : n \in \mathbf{N}\}$ of function symbols is coherent, the extension $\Sigma\{F\sigma^\nabla\} := \Sigma \cup \mathfrak{W}[F\sigma^\nabla]$ is conservative: $\Sigma\{F\sigma^\nabla\} \vdash^{\mathcal{U}} \tau$ iff $\Sigma \vdash \tau$ for each sentence τ of $L(\sigma[F])$.

Proof outline. The preceding lemma yields expansion of models.

Thus, we can conservatively extend a first-order theory Σ with coherent functions symbols of each arity so that these symbols become generic: $\Sigma \cup \mathfrak{W}[F\sigma^\nabla] \vdash^{\mathcal{U}} \tau$ iff $\Sigma \vdash \langle \tau \rangle_\nabla$, for each sentence τ of $L^\nabla(\sigma)$. This reduces reasoning with generic functions within ultrafilter logic to reasoning with coherent functions within first-order logic.

5 A Framework for Reasoning with 'Generally'

We will now put together our results to show how we can provide a framework where reasoning with 'generally' reduces to first-order reasoning with coherent functions.

In general, the proof procedure reduces ultrafilter consequence to classical first-order derivability with coherent functions as follows: establishing $\Gamma \models^{\mathcal{U}} \tau$ amounts to showing that $\langle \Gamma \rangle_\nabla \cup \Omega[\sigma/F] \vdash \langle \tau \rangle_\nabla$.

To show that this reduction procedure is sound and complete, we have established the following facts for sets of sentences $\Gamma \subseteq L^\nabla(\sigma)$ and $\Sigma \subseteq L(\sigma)$.

¹⁹ Considering constants naming the elements of $\mathbf{A}$, form the set ${}_\nabla\Sigma$ of all extensions $\mathbf{A}[\varphi[\underline{u} := \underline{a}]] \subseteq \mathbf{A}$ such that $\mathbf{A} \models \varphi[\underline{u} := \underline{a}][z := f_{|\underline{u}|}(\underline{a})]$ for each formula φ of $L(\sigma[F])$ having list of free variables with $\underline{u}$ and z . This family ${}_\nabla\Sigma \subseteq \wp(\mathbf{A})$ has the finite intersection property (since $\emptyset \notin {}_\nabla\Sigma$ and coherence makes ${}_\nabla\Sigma$ closed under intersection), so it can be extended to an ultrafilter $\mathcal{U} \supseteq {}_\nabla\Sigma$. For each formula φ of $L(\sigma)$, $\mathbf{A}[\varphi[\underline{u} := \underline{a}]] \in {}_\nabla\Sigma$ iff $\mathbf{A}[\varphi[\underline{u} := \underline{a}]] \in \mathcal{U}$ (if $\mathbf{A}[\varphi[\underline{u} := \underline{a}]] \notin {}_\nabla\Sigma$ then $\mathbf{A}[\neg \varphi[\underline{u} := \underline{a}]] \in {}_\nabla\Sigma \subseteq \mathcal{U}$, whence $\mathbf{A}[\varphi[\underline{u} := \underline{a}]] \notin \mathcal{U}$). The ultrafilter structure $\mathbf{A}^{\mathcal{U}} = (\mathbf{A}, \mathcal{U})$ satisfies the generic axiom schema $\mathfrak{W}[F\sigma^\nabla]$: for each formula φ of $L^\nabla(\sigma)$ with m free variables, $\mathbf{A}^{\mathcal{U}} \models \mathfrak{W}[f_m \nabla z \varphi]$, by induction on the structure of formula φ of $L^\nabla(\sigma)$, the basis being by construction.

1. The extension by generic axioms is conservative: $\Gamma \leq \Gamma \cup \mathfrak{W}[F \setminus \sigma^\nabla]$.
2. The generic axioms yield the coherence axioms: $\mathfrak{W}[F \setminus \sigma^\nabla] \vdash^u \Omega[\sigma/F]$.
3. The extension of a coherent first-order theory by generic axioms is conservative: $\Sigma \leq \Sigma \cup \mathfrak{W}[F \setminus \sigma^\nabla]$ whenever $\Sigma \vdash \Omega[\sigma/F]$.

We will then have $\Gamma \cup \mathfrak{W}[F \setminus \sigma^\nabla]$ (equivalent to $\langle \Gamma \rangle_\nabla \cup \Omega[\sigma/F] \cup \mathfrak{W}[F \setminus \sigma^\nabla]$ by 2) as a common conservative extension of both Γ and $\langle \Gamma \rangle_\nabla \cup \Omega[\sigma/F]$.²⁰

$$\frac{\frac{\frac{L^\nabla(\sigma)}{\Gamma}}{\Lambda \mid} \quad \frac{\frac{L(\sigma[F])}{\Gamma_\nabla \cup \Omega[\sigma/F]}}{\Lambda \mid}}{\Gamma \cup \mathfrak{W}[F \setminus \sigma^\nabla]} \equiv \frac{\Gamma_\nabla \cup \Omega[\sigma/F] \cup \mathfrak{W}[F \setminus \sigma^\nabla]}{L^\nabla(\sigma[F])}$$

Fig. 1. Common conservative extension

In many practical cases (as in databases, for instance), we deal only with formulas with bounded depth of nested ∇ 's; then, it suffices to add a finite number of new special functions and axioms.

For an induction-like example, consider a sample of minerals and let $S(x,y)$ stand for "x is similar to y", $G(x)$ for "x is green" and e for a particular emerald. Assume that "minerals generally are similar to e": $\nabla z S(z,e)$ and that "e is green": $G(e)$. Also, suppose that similarity transfers colors: $\forall u \nabla z [S(z,u) \rightarrow (G(u) \rightarrow G(z))]$. Then, we can infer that minerals generally are green: $\nabla z G(z)$. In this case, Γ consists of $\nabla z S(z,e)$, $G(e)$ and $\forall u \nabla z [S(z,u) \rightarrow (G(u) \rightarrow G(z))]$, and we can reduce $\Gamma \vdash^u \nabla z G(z)$ to $\langle \Gamma \rangle_\nabla \cup \Omega[\sigma/\{f_0, f_1\}] \vdash \nabla z G(z)_{>\nabla}$, where $\langle \Gamma \rangle_\nabla$ consists of $S(f_0, e)$, $G(e)$ and $\forall u [S(f_1(u), u) \rightarrow (G(u) \rightarrow G(f_1(u)))]$, $\Omega[\sigma/\{f_0, f_1\}]$ has $\forall u [S(f_0, u) \leftrightarrow S(f_1(u), u)]$ and $\forall u [G(f_0) \leftrightarrow G(f_1(u))]$, with $\nabla z G(z)_{>\nabla}$ being $G(f_0)$.

We have concentrated on ultrafilter logic, but we can adapt the main lines of the development to other logics for 'generally'. In these cases, the generic functions will be more similar to Skolem functions (one for each formula) and coherence axioms (corresponding to translated schemata) will connect these functions.²¹

²⁰ Thus, soundness will follow from 1 and 2, while 3 will yield completeness.

²¹ We will have a generic function $f_{\nabla z \phi}$ for each generalized formula $\nabla z \phi$, with generic axiom of the form $\nabla z \phi(\underline{u}, z) \leftrightarrow \phi(\underline{u}, f_{\nabla z \phi}(\underline{u}))$. We will employ coherence axioms like $\forall z [\psi(\underline{u}, z) \rightarrow \theta(\underline{v}, z)] \rightarrow [\psi(\underline{u}, f_{\nabla z \psi}(\underline{u})) \rightarrow \theta(\underline{v}, f_{\nabla z \theta}(\underline{v}))]$, corresponding to the translation of schema $[\rightarrow \nabla]$, and similarly for $[\nabla \wedge]$ and $[\nabla \alpha]$ (the translations of the other schemata become valid formulas). This procedure works for any logic having schemata $[\nabla \exists]$ and $[\nabla \nabla]$, its correctness being simple to establish.

6 Conclusion

Logics for 'generally' were introduced for handling assertions with vague notions, such as 'generally', 'most', 'several'. To make possible automated theorem proving in these logics, we have introduced special functions, which reduce the situation to classical first-order logic. These special functions enable using any available classical proof procedure; so, there are many proof procedures and theorem provers at one's disposal. Their behavior may be affected by these special functions, so good strategies should take advantage of these functions. For instance, in the case of resolution [9], the unification procedure may incorporate the coherence axioms.²²

The main lines of the development (concentrated on ultrafilter logic) can be adapted to other logics for 'generally': with generic functions more similar to Skolem functions and coherence axioms connecting these functions.

Our framework is not meant as a competitor to non-monotonic logics, although it does solve monotonically various problems (e. g. generic reasoning) addressed by non-monotonic approaches.²³

As special functions enable using any available classical proof procedure, we expect to have paved the way for theorem proving in logics for 'generally'.²⁴

References

1. Grácio, M. C. G.: Lógicas Moduladas e Raciocínio sob Incerteza. D. Sc. dissertation, Unicamp, Campinas (1999).
2. Carnielli, W. A. and Veloso, P. A. S.: Ultrafilter Logic and Generic Reasoning. In: Gottlob, G., Leitsch, A. and Mundici, D. (eds.): Computational Logic and Proof Theory. Lecture Notes in Computer Science, Vol. 1289, Springer-Verlag, Berlin (1997) 34-53.
3. Zadeh, L. A.: Fuzzy Logic and Approximate Reasoning. *Synthese* 30 (1975) 407-428.
4. Turner, W.: Logics for Artificial Intelligence. Ellis Horwood, Chichester (1984).
5. Chang, C. C. and Keisler, H. J.: Model Theory. North-Holland, Amsterdam (1973).
6. Enderton, H. B.: A Mathematical Introduction to Logic. Academic Press, New York (1972).
7. Veloso, P. A. S. and Carnielli, W. A.: Logics for Qualitative Reasoning. CLE e-Prints 1 (2001) {<http://www.cle.unicamp.br/e-prints>}.
8. Veloso, P. A. S.: On 'almost all' and some presuppositions. In: Pereira, L. C. P. D. and Wrigley, M. B. (eds.): Logic, Language and Knowledge. Manuscript XXII (1999) 469-505.
9. Chang, C. L. and Lee, R. C. T.: Symbolic Logic and Mechanical Theorem Proving. Academic Press, New York (1973).
10. Rentería, C. J., Haeusler, E. H. and Veloso, P. A. S.: NUL- Natural Deduction for Ultrafilter Logic. In: Natural Deduction 2001, Rio de Janeiro (2001) {to appear as 'Dedução Natural para Lógica de Ultrafiltros', Res. Rept, PUC-Rio, Rio de Janeiro}.

²² Natural deduction for ultrafilter logic [10] can be regarded as using generic functions.

²³ We intend to investigate the applicability of our machinery to non-monotonic contexts (e. g. control of extensions and elimination of undesirable ones).

²⁴ The authors gratefully acknowledge partial financial support from the Brazilian National Research Council (CNPq): grants 522979/96-3 (to SRMV) and 301163/91-0 (to PASV).

First-Order Contextual Reasoning

Laurent Perrussel

IRIT/CERISS - Université Toulouse 1
Manufacture des Tabacs, 21 allée de Brienne
F-31042 Toulouse Cedex, France
`Laurent.Perrussel@univ-tlse1.fr`

Abstract. The objective of this paper is to develop a first order logic of contexts. Dealing with contexts in an explicit way has been initially proposed by J. McCarthy [16] as a means for handling generality in knowledge representation. For instance, knowledge may be distributed among multiple knowledge bases where each base represents a specific domain with its own vocabulary. To overcome this problem, contextual logics aim at defining mechanisms for explicitly stating the assumptions (i.e. the context) underlying a theory and also mechanisms for linking different contexts, such as lifting axioms for connecting one context to another one. However, integrating knowledge supposes the definition of inter-contextual links, based not only on relationships between contextual assertions, but also on relationships built upon contexts. In this paper, we introduce a quantificational modal-based logic of contexts where contexts are represented as explicit terms and may be quantified: we show how this framework is useful for defining first order properties over contexts.

1 Introduction

Nearly every assertion is based on underlying assumptions represented by a context. The explicit representation of contexts makes knowledge management easier by enabling representation of multiple micro-theories and links between them rather than a large theory (cf. [12, 16, 17]). Contextual theories are roughly composed of two components: contexts and assertions; and every assertion is stated in its context. Since the initial proposal of J. McCarthy [16], several proposals have been made for formalizing contextual reasoning:

- modal based propositional logic of contexts [4, 18, 19], first order logic of contexts [3, 9];
- propositional logics of contexts based on Labelled Deductive Systems (LDS) and fibred semantics [8];
- logics of contexts based on the situation theory [1];
- belief modelling as local reasoning and interaction/reification for representing lifting rules [10, 5];
- decision procedure for propositional logic of contexts [14, 15].

Among the applications of the different theories of contexts, let us mention: databases integration [7, 11, 9], huge knowledge base management [12], proof systems for multi-agent reasoning [2].

Propositional logic of contexts enables to represent inter-context relations in a restrictive way: relations are based on contextual truths, for instance "bridge" axioms may state that if ϕ holds in a first context then ψ holds in a second context. If contexts are denoted by terms, relations may be based on contextual assertions but also on first-order relations between contexts themselves. These new characteristics will be useful in numerous applications such as: knowledge bases federation, linguistics, distributed systems specification (such as multi-agent systems)... Our logic is based on modal predicate logic and the propositional and quantified logic of contexts introduced by [4, 3]. In [19], we have proposed a modal based logic of contexts. In this previous work, we were only considering the propositional case. We extend this proposal by considering the first-order case. As [16], we define a contextual modality: $\mathbf{ist}(\chi, \phi)$ which means that ϕ is true in context χ . We also root every statement in a sequence of contexts: $\langle \chi_1 \cdots \chi_n \rangle : \phi$ where the sequence represents nested contexts. As already proposed in [19], our logic defines rules for entering and exiting a context such that we can handle hierarchical knowledge. (see also [16]).

This paper is organized as follows: the next section presents our logic (proof theory and semantics). Section 3 illustrates our contribution with an example. Section 4 details similar contributions. In section 5 we draw some conclusions and discuss future work.

2 The Logic

Logic of contexts introduces a truth modality: $\mathbf{ist}(\chi, \phi)$, which has the following meaning: the formula ϕ is true in the context χ . Since formulae are always considered in a context or more generally in a sequence of contexts which represents nested contexts, formulae are always rooted in an initial sequence of contexts [4, 19]. For instance:

$$\langle US, Year2000 \rangle : \mathbf{ist}(California, governor(Davis))$$

asserts that in the nested contexts $\langle US, Year2000 \rangle$, the formula $governor(Davis)$ is true in the Californian context. The aim of first-order contextual logic is to represent assertions such as: $\sigma : \mathbf{ist}(\chi, \phi) \rightarrow \exists c \mathbf{ist}(c, \phi')$ where σ is a sequence of contexts, χ a context, c a variable ranging over contexts and ϕ and ϕ' are two formulae. Our logic of contexts extends the first order multi-sorted calculus. When we "enter" a context, we "forget" the contextual operator. Let us suppose the following formula $\chi : \mathbf{ist}(\chi', \phi)$; when we have entered the context χ' , we get $\chi.\chi' : \phi'$, i.e. the modality is omitted. At the opposite, leaving the context χ' consists of asserting $\mathbf{ist}(\chi', \phi)$ in the context χ (i.e. $\chi : \mathbf{ist}(\chi', \phi)$).

Since we want to define inter-contextual relations, symbols denoting predicates, constants and terms have to be considered as shared among the contexts. We

consider two sorts of terms: terms denoting contexts and terms denoting the objects of the domain. The first one is referred as the context sort and the second one as the domain sort. Variables may appear in the contextual truth modality and in sequence of contexts. Free variables are considered universally quantified, in particular in sequences of contexts. Consequently, quantification has to be also defined on the sequences.

2.1 Syntax

The first order logic of contexts is based on first order modal logic [6], propositional logic of contexts [4] and quantificational logic of contexts presented in [3]. The definition of the language is twofold: firstly we describe the syntax of an auxiliary language $\mathcal{L}_{aux}$. Secondly we associate to each $\mathcal{L}_{aux}$ -formula a sequence of contexts. Let $\mathcal{L}_{seq}$ be the resulting language.

Definition 1. Let $\mathcal{C}^S$ be a set of constants of the domain sort, $\mathcal{V}^S$ a set of variables of the domain sort, $\mathcal{C}^C$ a set of constants of the context sort and $\mathcal{V}^C$ a set of variables of the context sort. Let $\mathcal{PRE}\mathcal{D}$ be a set of predicate symbols; each predicate has an arity.

Firstly, we define the $\mathcal{L}_{aux}$ language. Definitions are based on the classical definitions of the first order calculus.

Definition 2 (Syntax of $\mathcal{L}_{aux}$). Let $\mathcal{T}^S$ be the set of terms of the domain sort: $\mathcal{T}^S = \mathcal{C}^S \cup \mathcal{V}^S$. Let $\mathcal{T}^C$ be the set of terms of the context sort: $\mathcal{T}^C = \mathcal{C}^C \cup \mathcal{V}^C$. The set of $\mathcal{L}_{aux}$ -formulae is defined with the following rules:

- if $t_1, t_2 \in \mathcal{T}^S$ then $t_1 = t_2 \in \mathcal{L}_{aux}$ and if $t_1, t_2 \in \mathcal{T}^C$ then $t_1 = t_2 \in \mathcal{L}_{aux}$;
- if $P \in \mathcal{PRE}\mathcal{D}$, n is the arity of P and $t_1 \dots t_n \in \mathcal{T}^S \cup \mathcal{T}^C$ then $P(t_1, \dots, t_n) \in \mathcal{L}_{aux}$;
- if $\phi \in \mathcal{L}_{aux}$ then $\neg\phi \in \mathcal{L}_{aux}$;
- if $\phi, \psi \in \mathcal{L}_{aux}$ then $\phi \rightarrow \psi, \phi \wedge \psi, \phi \vee \psi, \phi \leftrightarrow \psi \in \mathcal{L}_{aux}$;
- if $\chi \in \mathcal{T}^C$ and $\phi \in \mathcal{L}_{aux}$ then $\text{ist}(\chi, \phi) \in \mathcal{L}_{aux}$;
- if $v \in \mathcal{V}^C \cup \mathcal{V}^S$ and $\phi \in \mathcal{L}_{aux}$ then $\forall v(\phi)$ and $\exists v(\phi) \in \mathcal{L}_{aux}$.

Definition 3 (Sequences of contexts). Let Σ be the set of sequences defined as follows:

1. if $\chi \in \mathcal{T}^C$ then $\langle \chi \rangle \in \Sigma$;
2. if $\sigma = \langle \chi_1 \dots \chi_n \rangle \in \Sigma$ and $\chi \in \mathcal{T}^C$ then $\langle \chi_1 \dots \chi_n, \chi \rangle \in \Sigma$.

If $\sigma = \langle \chi_1 \dots \chi_n \rangle$ and $\sigma' = \langle \chi'_1 \dots \chi'_m \rangle$ are two sequences then $\sigma.\sigma'$ refers to the concatenation of the two sequences: $\langle \chi_1 \dots \chi_n, \chi'_1 \dots \chi'_m \rangle$. For simplicity, $\sigma.\chi$ represents the sequence $\sigma.\langle \chi \rangle$ ($\chi \in \mathcal{T}^C$). If common variables appear in sequences σ and σ' then every common variable should be renamed in a previous stage. Next the concatenation may be applied with the resulting sequences.

Definition 4 (Syntax of $\mathcal{L}_{seq}$). The set of $\mathcal{L}_{seq}$ -formulae is defined as follows:

- if $\phi \in \mathcal{L}_{aux}, \sigma \in \Sigma$ then $\sigma : \phi \in \mathcal{L}_{seq}$.
- if $\sigma : \phi \in \mathcal{L}_{seq}$ and $x \in \mathcal{V}^C$, then $\forall x(\sigma : \phi) \in \mathcal{L}_{seq}$.

2.2 Proof Theory

Since our logic extends the propositional logics of contexts, we still find here the inference rules for entering and exiting a context. Let $\vdash$ be the proof relation. Since we derive formulae in sequences of contexts, we say that a formula ϕ is σ -provable iff there is a proof of $\vdash \sigma : \phi$. A formula ϕ is σ -provable with respect to a set of $\mathcal{L}_{aux}$ -formulae T rooted in σ , $T \vdash \sigma : \phi$ iff there are formulae $\phi_1, \dots, \phi_n$ belonging to T such that $\vdash \sigma : \phi_1 \wedge \dots \wedge \phi_n \rightarrow \phi$. In the following definition, $\phi(x)$ refers to a formula where x is a free variable:

Definition 5 (Proof Theory). *The axiom schemas are:*

all tautologies of classical propositional logic

(AS-1) $\sigma : \quad \forall x \phi(x) \rightarrow \phi(t)$

(AS-2) $\sigma : \quad \forall x (\phi \rightarrow \psi) \rightarrow (\phi \rightarrow \forall x \psi)$

(AS-3) $\sigma : \quad \forall x (x = x)$

(AS-4) $\sigma : \quad (x = y) \rightarrow (\phi(x) \rightarrow \phi(y))$

(AS-5) $\sigma : \quad \forall x \text{ist}(\chi, \phi(x)) \rightarrow \text{ist}(\chi, \forall x \phi(x))$

(AS-K) $\sigma : \text{ist}(\chi, \phi \rightarrow \psi) \rightarrow (\text{ist}(\chi, \phi) \rightarrow \text{ist}(\chi, \psi))$

Since our language is two sorted, we have to define some constraints. Concerning (AS-1), if $x \in \mathcal{V}^C$ then $t \in \mathcal{T}^C$ and if $x \in \mathcal{V}^S$ then $t \in \mathcal{T}^S$. In both cases, t is free for x in ϕ . Concerning (AS-2), x is free in ψ and does not appear in ϕ . (AS-5) represents the Barcan schema. In (AS-5), x does not appear in χ . We have to adopt this schema since one of our desiderata is to define inter-contextual relations.

The inference rules are: *modus ponens (MP)*

$$(MP) \frac{\vdash \sigma : \phi \quad \vdash \sigma : \phi \rightarrow \psi}{\vdash \sigma : \psi}$$

the contextual inference rule for entering a context (CIR^{IN}) and for exiting a context (CIR^{OUT}):

$$(CIR^{IN}) \frac{\vdash \langle \chi_1 \dots \chi_n \rangle : \text{ist}(\chi, \phi)}{\vdash \langle \chi_1 \dots \chi_n, \chi \rangle : \phi} \quad (CIR^{OUT}) \frac{\vdash \langle \chi_1 \dots \chi_n, \chi \rangle : \phi}{\vdash \langle \chi_1 \dots \chi_n \rangle : \text{ist}(\chi, \phi)}$$

and the generalization rules

$$(G) \frac{\vdash \sigma : \phi}{\vdash \sigma : (\forall x) \phi} \quad (G') \frac{\vdash \sigma : \phi}{\vdash \forall x (\sigma : \phi)}$$

The inference rule (G') is needed to handle free variable in sequences. Let us mention that (G') could not be defined with (G) and (AS-5) since empty sequences are prohibited (every statement has to be considered in some context). We call a *CFO*-system an axiomatic system which includes these axiom schemas and inference rules.

2.3 Semantics

Our semantics is based on the possible worlds semantics. Let us assume $\sigma' = \sigma.\chi$. A formula $\sigma : \text{ist}(\chi, \phi)$ is true in a world w if and only if the formula $\sigma' : \phi$ is true for all the worlds w' which are accessible from R . Thus, we replace a world by a couple $\langle \text{sequence}, \text{world} \rangle$ that we call a *situation* s . To take into account these situations, let $S \subseteq \Sigma \times W$ be the set of situations. The relation R is a subset of $S \times S$. Now, we can define our interpretation function. A model M_{FO} is a tuple $\langle W, D_d, D_c, S, R, I \rangle$ which has the following definition:

Definition 6 (M_{FO}). *Let M_{FO} be a tuple $\langle W, D_d, D_c, S, R, I \rangle$ where:*

- W is a non empty set of worlds;
- D_d is a non empty set which is the universe of the discourse;
- D_c is a non empty set which is the universe of contexts;
- S is a set of situations ($S \subseteq W \times \Sigma_{D_c}$) such that Σ_{D_c} represents the set of sequences built upon the domain D_c ;
- R is an accessibility relation: $R \subseteq S \times S$;
- I is a tuple $\langle I_1, I_2, I_3 \rangle$ of interpretation functions:
 - I_1 is an assignment function such that for each $\chi \in \mathcal{C}^C$, $I_1(\chi) \in D_c$;
 - I_2 is an assignment function such that for each $c \in \mathcal{C}^S$, $I_2(c) \in D_d$;
 - I_3 is an interpretation function, such that, where P is n -place predicate, s is a situation; $I_3(P, s)$ is a subset of $(D_d \cup D_c)^n$.

The variable assignment is a couple of assignments: a first one for the variables of the context sort and a second one for the variables of the domain sort. Let v be the variable assignment such that $v = \langle v_1, v_2 \rangle$. v_1 is a function such that, for each variable $x \in \mathcal{V}^C$, $v_1(x) \in D_c$; v_2 is a function such that, for each variable $x \in \mathcal{V}^S$, $v_2(x) \in D_d$. An x -alternative v' of v is a variable assignment similar to v for every variable except x ($v'(x)$ respects the sort of x). $\llbracket t \rrbracket_{M,v}$ refers to the assignment of terms, such that $t \in \mathcal{T}^S \cup \mathcal{T}^C$, M is a M_{FO} -model and v is a variable assignment:

- $\llbracket \chi \rrbracket_{M,v} = I_1(\chi)$ if $\chi \in \mathcal{C}^C$; $\llbracket c \rrbracket_{M,v} = I_2(c)$ if $c \in \mathcal{C}^S$;
- $\llbracket x \rrbracket_{M,v} = v_1(x)$ if $x \in \mathcal{V}^C$; $\llbracket y \rrbracket_{M,v} = v_2(y)$ if $x \in \mathcal{V}^S$;
- $\llbracket \sigma \rrbracket_{M,v} = \langle \llbracket \chi_1 \rrbracket_{M,v}, \dots, \llbracket \chi_n \rrbracket_{M,v} \rangle$ if $\sigma = \langle \chi_1, \dots, \chi_n \rangle \in \Sigma$;

Contexts and variables are considered regardless of worlds and contexts and therefore they are treated in a *rigid* way. The relation $M_{FO}, w \models_v \sigma : \phi$ should be interpreted as following: the $\mathcal{L}_{aux}$ -formula ϕ is σ -satisfied by the model M_{FO} in the world w and for the assignment v .

Definition 7 (Semantics). *The satisfiability of an $\mathcal{L}_{seq}$ -formula $\sigma : \phi$ is defined as follows:*

- $M_{FO}, w \models_v \sigma : t_1 = t_2$ iff $\llbracket t_1 \rrbracket_{M_{FO},v} = \llbracket t_2 \rrbracket_{M_{FO},v}$;
- $M_{FO}, w \models_v \sigma : p(t_1, \dots, t_n)$ iff $\langle \llbracket t_1 \rrbracket_{M_{FO},v}, \dots, \llbracket t_n \rrbracket_{M_{FO},v} \rangle \in I_3(p, \langle w, \llbracket \sigma \rrbracket_{M_{FO},v} \rangle)$;
- $M_{FO}, w \models_v \sigma : \neg \phi$ iff $M_{FO}, w \not\models_v \sigma : \phi$;
- $M_{FO}, w \models_v \sigma : \phi \rightarrow \psi$ iff if $M_{FO}, w \models_v \sigma : \phi$ then $M_{FO}, w \models_v \sigma : \psi$;

- $M_{FO}, w \models_v \sigma : \exists x \phi$ iff there is an x -alternative v' , $M_{FO}, w \models_{v'} \sigma : \phi$;
- $M_{FO}, w \models_v \sigma : \forall x \phi$ iff for any x -alternative v' , $M_{FO}, w \models_{v'} \sigma : \phi$;
- $M_{FO}, w \models_v \forall x(\sigma : \phi)$ iff for any x -alternative v' , $M_{FO}, w \models_{v'} \sigma : \phi$;
- $M_{FO}, w \models_v \sigma : \mathbf{ist}(\chi, \phi)$ iff for any w' s.t. $(\langle w, \llbracket \sigma \rrbracket_{M,v} \rangle, \langle w', \llbracket \sigma.\chi \rrbracket_{M,v} \rangle) \in R$, we have $M_{FO}, w' \models_v \sigma.\chi : \phi$.

An $\mathcal{L}_{aux}$ -formula ϕ is σ -satisfied iff there exists a variable assignment v , a world w and an interpretation M_{FO} such that $M_{FO}, w \models_v \sigma : \phi$. An $\mathcal{L}_{aux}$ -formula ϕ is σ -valid in an interpretation M_{FO} and for variable assignment v iff for every world w , ϕ is σ -satisfied. An $\mathcal{L}_{aux}$ -formula ϕ is σ -valid iff ϕ is σ -valid in any interpretation and every assignment v , i.e. $\models \sigma : \phi$. We write $T \models \sigma : \phi$ (T is a set of $\mathcal{L}_{aux}$ -formulae) iff for all M_{FO}, v, w if $M_{FO}, w \models_v \sigma : T$ then $M_{FO}, w \models_v \sigma : \phi$. We constrain the relation R such that R is hyper-reflexive. This constraint reflects the inference rule CIR^{OUT} .

for every world $w \in W$, every sequence σ and every context χ , if $\langle \sigma, w \rangle \in S$ then $(\langle \sigma, w \rangle, \langle \sigma.\chi, w \rangle) \in R$.

Models whose relation R satisfies this constraint are called W_{FO} -models.

2.4 Soundness and Completeness

We close the description of our first order contextual logic with the classical result of soundness and completeness.

Theorem 1 (Soundness and Completeness). $\vdash \sigma : \phi \iff \models \sigma : \phi$

Proofs are based on [6, 20].

3 An Example

In the following example, we use contexts for describing different concepts: agent, city... Let us consider a multi-agent system which delivers information about weather, accomodation... The end user interacts with a special agent called the mediator agent (referred as M). The mediator agent interacts with agents supplying information about different cities. Let us focus on weather information in a city. Assume the predicate $CurrentTemp(x)$ which means the current temperature is x . Our aim is to infer the temperature in a city in the mediator agent knowledge base.

Note that the city is implicit when the current temperature is stated. The formula is in fact asserted in a *city* context (a variable of the context sort):

$$city : \exists x CurrentTemp(x)$$

Assume that the weather agents deliver information about the temperature with the predicate $temperature(y, z)$ where y is a city and z is the temperature in this city. In other words, if x represents a weather agent we write

$x : temperature(y, z)$. Every agent connects its own context with the *city* context with the lifting axiom $\forall y \forall z (temperature(y, z) \leftrightarrow \mathbf{ist}(y, CurrentTemp(z)))$. This $\mathcal{L}_{aux}$ -formula is stated in the context of an agent denoted by the variable x :

$$x : \forall y \forall z (temperature(y, z) \leftrightarrow \mathbf{ist}(y, CurrentTemp(z)))$$

For the mediator agent M , the lifting axiom holds for every agent x : weather agents may enter in the *city* context in order to derive the temperature:

$$\langle M.x \rangle : \forall y \forall z (temperature(y, z) \leftrightarrow \mathbf{ist}(y, CurrentTemp(z)))$$

Let us suppose the resource agent a and the city of San Francisco (SF). Firstly, we describe the current temperature:

$$\langle M.a.SF \rangle : currentTemp(15)$$

Secondly, by exiting the context of SF we get $\langle M.a \rangle : \mathbf{ist}(SF, currentTemp(15))$ (CIR^{OUT}) and thus, by modus ponens, we conclude:

$$\langle M.a \rangle : temperature(SF, 15)$$

By exiting the resource agent context, the mediator may then derive in its own context $\mathbf{ist}(a, temperature(SF, 15))$ in order to provide it to the final user.

4 Related Works

In this section, we consider the state of art for contextual knowledge representation. As previously mentioned, [16] and [4, 3] have been the primary source of inspiration for this work. Our logic is also closely related to first order multi-modal logics. At the end of the section we compare our logic to the quantificational logics of contexts and first order modal logic.

Before, we consider the main contributions in the contextual reasoning area. [18] presents a propositional logic of contexts based on propositional modal logic. The main difference with our logic concerns the rooting: statements are not considered as rooted in a sequence of contexts and thus contexts are viewed in a flat way. Consequently, notions such as "entering" or "exiting" a context have disappeared. In [13, 14], a context is a logical system and consequently contextual reasoning is considered as integrating different logics. [9] presents a first order logic (DFOL) for integrating distributed knowledge (among different bases). The main differences between this contribution and ours concern three main points. Firstly [9] distinguishes contextual knowledge representation and the definition of inter-contextual relations. These relations are described using specific inference rules named bridge rules and thus prevent from mixing contextual knowledge definition and inter-contextual relations definition. In other words, relations (also stated in a context) such as $\chi_R : \mathbf{ist}(\chi, \phi) \rightarrow \exists x \mathbf{ist}(x, \phi')$ could not be represented. This kind of statements may be useful for approximate reasoning: χ describes an "approximate context" (e.g. a section of introduction)

and $\exists \text{ist}(x, \phi')$ states that there is a more specific context which "describes" ϕ in more specific terms (ϕ') (e.g. a chapter). Another kind of inter-contextual relations which could not be represented by [9] is the equivalence of domain theories under special circumstances:

$$\chi_R : \text{circum} \rightarrow (\chi = \chi')$$

The basic idea is to state that if *circum* holds (i.e. a sufficient condition) then χ and χ' have to be considered as similar contexts. This may be useful when contexts represent distributed knowledge bases and *circum* represents a query: according to the query, the answer may be defined with respect to χ or χ' . Secondly, [9] proposes to take into account different vocabularies and domains. We do not assume this option in our framework since we want to define first order properties of contexts. The last difference concerns the notion of sequence of contexts: as in [18], it does not appear in [9].

4.1 Comparison to the First Order Logic of S. Buvac

In [3], S. Buvac presents a quantificational logic of contexts where every statement is rooted in context. However, nested contexts are not considered and thus, "entering in a context" has a different meaning: it has to be viewed as switching from one context to a second one. The rule for entering is defined as follows:

$$\frac{\chi : \text{ist}(\chi', \phi)}{\chi' : \phi}$$

S. Buvac justifies this definition by considering that every context looks the same regardless of the initial context. In our logic, we adopt a different approach: we tolerate nested contexts and variables in sequences of contexts. This characteristic allows to define hierarchical knowledge (as the example considered in the previous section) while it is impossible in the S. Buvac framework.

4.2 Comparison to First-Order Modal Logic

Quantified modal logics do not allow quantification of modal operators. Let us also note that local derivability and rules CIR^{IN} and CIR^{OUT} are concepts specific to contextual logics. However, if we consider a subset of $\mathcal{L}_{seq}$ -formulae, we may define mapping rules for translating a contextual statement in modals terms. Let us consider the statement $\sigma : \Phi$ such that:

- no variable (of the context sort) appears in the sequence σ ;
- for every sub-formula $\text{ist}(c, \varphi) \in \Phi$, c is a constant.

For every sequence σ , assume a modal operator $[\sigma]$. Let $\mathcal{L}_m$ be a first order modal language which includes the axiom schema K . For every formula $\sigma : \Phi$ s.t. $\sigma : \Phi$ respects the previous conditions, we consider a modal formula $f(\sigma, \Phi)$ s.t. if $\vdash \sigma : \Phi$ then $\vdash_{\mathcal{L}_m} f(\sigma, \Phi)$. The function f is defined as follows:

- $f(\sigma, P(\vec{x})) = P(\vec{x})$,

- $f(\sigma, \varphi_1 \rightarrow \varphi_2) = f(\sigma, \varphi_1) \rightarrow f(\sigma, \varphi_2)$ (respectively for $\wedge, \vee, \neg$),
- $f(\sigma, \mathbf{ist}(c, \varphi)) = [\sigma.c]f(\sigma.c, \varphi)$.

When no variable of the context appears in sequences and in $\mathbf{ist}$ statements, the logic $\mathcal{L}_{seq}$ may then be reduced to $\mathcal{L}_m$. For instance, the formula $c : \mathbf{ist}(c', \exists xp(x)) \rightarrow \mathbf{ist}(c'', p(y))$ is translated, with respect to f , as $[c.c']\exists xp(x) \rightarrow [c.c'']p(y)$. However, as we can see we are limited to a sub part of $\mathcal{L}_{seq}$ since we can not translate first order intercontextual relationships.

5 Conclusion

In this article, we have presented a logical formalism that can handle quantified contextual statements. Firstly we have defined some requirements for representing inter-contextual relations. Secondly, we have described our logic $\mathcal{L}_{seq}$ (proof theory and semantics) and we have stated the completeness and soundness of $\mathcal{L}_{seq}$ logic. Finally, after illustrating the interest of first order contextual statements for describing knowledge, we have compared $\mathcal{L}_{seq}$ with similar logics.

Since contexts are represented by terms in $\mathcal{L}_{seq}$, first order properties over contexts may be easily defined: quantified inter-contextual relations are represented in a simple way. Moreover, derivability and interpretation are contextual since we claim that every formula should be considered in some context (or a sequence of contexts). In other words, we have rejected the notion of "super-context". This characteristic distinguishes $\mathcal{L}_{seq}$ from the "classical" modal logics. This difference is represented in the axiomatic, by inference rules which allow to go in and out of a context and, in the model theory, by a specific accessibility relation and constraints.

Applications are numerous: federated knowledge bases, linguistics... Clearly more work needs to be done so as to define a more richer framework. For instance, $\mathcal{L}_{seq}$ do not consider multiple languages (and thus multiple universes of discourses) as [9]. This is necessary if, for instance, we want to describe systems which represent federation of heterogeneous knowledge bases. Another key point concerns context hierarchies described here as sequences of contexts: how can they be used for non-monotonic reasoning (as suggested in [16])? This last issue will probably lead us to characterize the contexts in terms of generalization and specialization and thus to give a definition to the concept of context.

References

- [1] V. Akman. The use of situation theory in context modeling. *Computational Intelligence*, 1997. 11
- [2] P. Bonzon. A reflective proof system for reasoning in context. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI'97)*, Providence, Rhodes Island, 1997. 12
- [3] S. Buvač. Quantificational Logic of Contexts. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996. 11, 12, 13, 17, 18
- [4] S. Buvač, V. Buvač, and I. A. Mason. Meta-Mathematics of Contexts. *Fundamenta Informaticae*, 23(3):263–301, 1995. 11, 12, 13, 17

- [5] A. Cimatti and L. Serafini. MultiAgent Reasoning With Belief Contexts II: Elaboration Tolerance. In *Proceedings of the first International Conference on Multi-Agent Systems (ICMAS-95)*, June 12–14, 1995. San Francisco, CA, USA, pages 57–64. AAAI Press / The MIT Press, 1995. [11](#)
- [6] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge*. MIT Press, 1995. [13](#), [16](#)
- [7] A. Farquhar, A. Dappert, R. Fikes, and W. Pratt. Integrating information sources using context logic. In *AAAI-95 Spring Symposium on Information Gathering from Distributed Heterogeneous Environments*, 1995. [12](#)
- [8] D. Gabbay and R. Nossum. Structured contexts with fibred semantics. In *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*, Rio de Janeiro, Brazil, February 4–6, pages 46–56, 1997. [11](#)
- [9] C. Ghidini and L. Serafini. A context-based logic for distributed knowledge representation and reasoning. In P. Bouquet, L. Serafini, P. Brézillon, M. Benerecetti, and F. Castellani, editors, *Proceedings of the Second International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT'99)*, Trento, Italy, September 1999, number 1688 in Lecture Notes In Computer Science, pages 159–172. Springer-Verlag, 1999. [11](#), [12](#), [17](#), [18](#), [19](#)
- [10] F. Giunchiglia, L. Serafini, E. Giunchiglia, and M. Frixione. Non-Omniscient Belief as Context-Based Reasoning. In *Proceedings IJCAI-93, 13th International Joint Conference on Artificial Intelligence*. Chambéry, France, 1993. [11](#)
- [11] C. Goh, S. Madnik, and M. Siegel. Ontologies, contexts and mediation: Representing and reasoning about semantic conflicts in heterogeneous and autonomous systems. Technical Report 2848, Sloan School of Management, 1996. also CISL Working Paper 95-04. [12](#)
- [12] R. V. Guha. *Contexts: A Formalization and Some Applications*. PhD thesis, Stanford University, 1991. [11](#), [12](#)
- [13] F. Massacci. A Bridge Between Modal Logics and Contextual Reasoning. In *IJCAI-95 International Workshop on Modeling Context in Knowledge Representation and Reasoning*, 1995. [17](#)
- [14] F. Massacci. Superficial tableaux for contextual reasoning. In S. Buvač, editor, *Proc. of the AAAI-95 Fall Symposium on "Formalizing Context"*, number FS-95-02 in AAAI Tech. Reports Series, pages 60–66. AAAI Press/The MIT Press, 1995. [11](#), [17](#)
- [15] F. Massacci. Contextual reasoning is NP-complete. In W. Clancey and D. Weld, editors, *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 621–626. AAAI Press/The MIT Press, 1996. [11](#)
- [16] J. McCarthy. Notes on formalizing context. In *Proceeding of the thirteen International Joint Conference on Artificial Intelligence, IJCAI'93, Chambéry, France*. Morgan Kaufmann Publishers, 1993. [11](#), [12](#), [17](#), [19](#)
- [17] J. McCarthy and S. Buvač. Formalizing Context: Expanded Notes. Technical Report STAN-CS-TN-94-13, Computer Science Dpt. - Stanford University, 1994. [11](#)
- [18] P. Nayak. Representing Multiple Theories. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1154–1160, Cambridge, MA-USA, 1994. AAAI Press/MIT Press. [11](#), [17](#), [18](#)
- [19] L. Perrussel. Contextual Reasoning. In H. Prade, editor, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI '98)*, August 23–28, 1998, Brighton UK, pages 366–367. John Wiley & Sons, Ltd, 1998. [11](#), [12](#)

- [20] L. Perrussel. *Un outillage Logique pour l'Ingénierie des Exigences Multi-Points de Vue*. PhD thesis, Université Toulouse 3, Toulouse, 1998. 16

Logics for Approximate Reasoning: Approximating Classical Logic “From Above”

Marcelo Finger and Renata Wassermann

Department of Computer Science
Institute of Mathematics and Statistics
University of São Paulo, Brazil
{mfinger,renata}@ime.usp.br

Abstract. Approximations are used for dealing with problems that are hard, usually NP-hard or coNP-hard. In this paper we describe the notion of approximating classical logic from above and from below, and concentrate in the first. We present the family $\mathbf{s}_1$ of logics, and show it performs approximation of classical logic from above. The family $\mathbf{s}_1$ can be used for disproving formulas (the SAT-problem) in a local way, concentrating only on the relevant part of a large set of formulas.

1 Introduction

Logic has been used in several areas of Artificial Intelligence as a tool for representing knowledge as well as a tool for problem solving. One of the main criticism to the use of logic as a tool for automatic problem solving refers to the computational complexity of logical problems. Even if we restrict ourselves to classical propositional logic, deciding whether a set of formulas logically implies a certain formula is a co-NP-complete problem [GJ79]. Another problem comes from the inadequacy of modelling real agents as logical beings. Ideal, logically omniscient agents know all the consequences of their beliefs. However, real agents are *limited* in their capabilities.

Cadoli and Schaerf have proposed the use of approximate entailment as a way of reaching at least partial results when solving a problem completely would be too expensive [SC95]. Their method consists in defining different logics for which satisfiability is easier to compute than classical logic and treat these logics as upper and lower bounds for the classical problem. In [SC95], these approximate logics are defined by means of valuation semantics and algorithms for testing satisfiability. The language they use is restricted to that of clauses, i.e., negation appears only in the scope of atoms and there is no implication.

The approximations are based on the idea of a context set S of atoms. The atoms in S are the only ones whose consistency is taken into account in the process of verifying whether a given formula is entailed by a set of formulas. As we increase the size of the context set S , we get closer to classical entailment, but the computational complexity also increases.

Cadoli and Schaerf proposed two systems, intending to approximate classical entailment from two ends. The S_3 family approximates classical logic from below, in the following sense. Let $\mathcal{P}$ be a set of propositions and $S^0 \subseteq S^1 \subseteq \dots \subseteq \mathcal{P}$; let $Th(L)$ indicate the set of theorems of a logic. Then:

$$Th(S_3(\emptyset)) \subseteq Th(S_3(S^0)) \subseteq Th(S_3(S^1)) \subseteq \dots \subseteq Th(S_3(\mathcal{P})) = Th(CL)$$

where CL is classical logic (in Section 3 this notion is extended to the entailment relation $\models$).

Approximating a classical logic from below is useful for efficient theorem proving. Conversely, approximating classical logic from above is useful for *disproving theorems*, which is the satisfiability (SAT) problem.

Unfortunately, Cadoli and Schaerf's other system, S_1 , does not approximate classical logic from above, as we will see in Section 3. In this paper, we study the family of logical entailments $\mathbf{s}_1$, which are approximations of classical logic from above. While S_1 only deals with formulas in negation normal form, $\mathbf{s}_1$ covers full propositional logic.

The family of logic $\mathbf{s}_1$ also tackles the problem of non-locality in S_1 , which implies that S_1 approximations do not concentrate on the *relevant formulas*. Discussions on locality are found in Section 5.

This paper proceeds as follows: in the next section, we briefly present Cadoli and Schaerf's work on approximate entailment. In Section 3 we present the notion of approximation that we are aiming at and show why Cadoli and Schaerf's system S_1 does not approximate classical logic from above. In Section 4 we present our system $\mathbf{s}_1$ and in Section 5 some examples of its behaviour.

Notation: Let $\mathcal{P}$ be a countable set of propositional letters. We concentrate on the classical propositional language $\mathcal{L}_C$ formed by the usual boolean connectives $\rightarrow$ (implication), $\wedge$ (conjunction), $\vee$ (disjunction) and $\neg$ (negation).

Throughout the paper, we use lowercase Latin letters to denote propositional letters, lowercase Greek letters to denote formulas, and uppercase letters (Greek or Latin) to denote sets of formulas. The letters S and s will denote sets of propositional letters.

Let $S \subset \mathcal{P}$ be a finite set of propositional letters. We abuse notation and write that, for any formula $\alpha \in \mathcal{L}_C$, $\alpha \in S$ if all its propositional letters are in S . A *propositional valuation* v_p is a function $v_p : \mathcal{P} \rightarrow \{0, 1\}$.

2 Approximate Entailment

We briefly present here the notion of *approximate entailment* and summarise the main results obtained in [SC95].

Schaerf and Cadoli define two approximations of classical entailment: $\models_S^1$ which is complete but not sound, and $\models_S^3$ which is classically sound but incomplete. These approximations are carried out over a set of atoms $S \subseteq \mathcal{P}$ which determines their closeness to classical entailment. In the trivial extreme of approximate entailment, i.e., when $S = \mathcal{P}$, classical entailment is obtained.

At the other extreme, when $S = \emptyset$, $\models_S^1$ holds for any two formulas (i.e., for all α, β , we have $\alpha \models_S^1 \beta$) and $\models_S^3$ corresponds to Levesque's logic for explicit beliefs [Lev84], which bears a connection to relevance logics such as those of Anderson and Belnap [AB75].

In an S_1 assignment, if $p \in S$, then p and $\neg p$ are given opposite truth values, while if $p \notin S$, both p and $\neg p$ get value 0. In an S_3 assignment, if $p \in S$, then p and $\neg p$ get opposite truth values, while if $p \notin S$, p and $\neg p$ do not both get 0, but may both get 1. The names S_1 and S_3 come from the possible truth assignments for literals outside S . If $p \notin S$, there is only one S_1 assignment for p and $\neg p$, the one which makes them both false. There are three possible S_3 assignments, the two classical ones, assigning p and $\neg p$ opposite truth values, and an extra one, making them both true. The set of formulas for which we are testing entailments is assumed to be in clausal form. Satisfiability, entailment, and validity are defined in the usual way.

The following examples illustrate the use of approximate entailment. Since $\models_S^3$ is sound but incomplete, it can be used to approximate $\models$, i.e., if for some S we have that $B \models_S^3 \alpha$, then $B \models \alpha$. On the other hand, $\models_S^1$ is unsound but complete, and can be used for approximating $\not\models$, i.e., if for some S we have that $B \not\models_S^1 \alpha$, then $B \not\models \alpha$.

Example 1. ([SC95]) We want to check whether $B \models \alpha$, where $\alpha = \neg \text{cow} \vee \text{molar-teeth}$ and

$$B = \{ \neg \text{cow} \vee \text{grass-eater}, \neg \text{dog} \vee \text{carnivore}, \\ \neg \text{grass-eater} \vee \neg \text{canine-teeth}, \neg \text{carnivore} \vee \text{mammal}, \\ \neg \text{mammal} \vee \text{canine-teeth} \vee \text{molar-teeth}, \\ \neg \text{grass-eater} \vee \text{mammal}, \neg \text{mammal} \vee \text{vertebrate}, \\ \neg \text{vertebrate} \vee \text{animal} \}.$$

Using the S_3 -semantic defined above, we can see that for $S = \{\text{grass-eater}, \text{mammal}, \text{canine-teeth}\}$, we have that $B \models_S^3 \alpha$, hence $B \models \alpha$.

Example 2. ([SC95]) We want to check whether $B \not\models \beta$, where $\beta = \neg \text{child} \vee \text{pensioner}$ and

$$B = \{ \neg \text{person} \vee \text{child} \vee \text{youngster} \vee \text{adult} \vee \text{senior}, \\ \neg \text{adult} \vee \text{student} \vee \text{worker} \vee \text{unemployed}, \\ \neg \text{pensioner} \vee \text{senior}, \neg \text{youngster} \vee \text{student} \vee \text{worker}, \\ \neg \text{senior} \vee \text{pensioner} \vee \text{worker}, \neg \text{pensioner} \vee \neg \text{student}, \\ \neg \text{student} \vee \text{child} \vee \text{youngster} \vee \text{adult}, \\ \neg \text{pensioner} \vee \neg \text{worker} \}.$$

Using the S_1 -semantic above, for $S = \{\text{child}, \text{worker}, \text{pensioner}\}$, we have that $B \not\models_S^1 \beta$, and hence $B \not\models \beta$.

Note that in both examples above, S is a small part of the language. Schaerf and Cadoli obtain the following results for approximate inference.

Theorem 1 ([SC95]). *There exists an algorithm for deciding if $B \models_S^3 \alpha$ and deciding $B \models_S^1 \alpha$ which runs in $O(|B| \cdot |\alpha| \cdot 2^{|S|})$ time.*

The result above depends on a polynomial time satisfiability algorithm for belief bases and formulas in clausal form alone. This result has been extended in [CS95] for formulas in negation normal form, but is not extendable to formulas in arbitrary forms [CS96].

3 The Notion of Approximation

The notion of approximation proposed by Cadoli and Schaerf can be described in the following way. Let $\models_S^3: 2^{\mathcal{L}} \times \mathcal{L}$ be the entailment relation in the logic $S_3(S)$, that is, the member of the family of logics S_3 determined by parameter S . Then, we had the following property. For

$$\emptyset \subseteq S' \subseteq S'' \subseteq \dots \subseteq S'^n \subseteq \mathcal{P}$$

we have that

$$\models_{\emptyset}^3 \subseteq \models_{S'}^3 \subseteq \dots \subseteq \models_{S'^n}^3 \subseteq \models_{\mathcal{P}}^3 = \models_{\text{CL}}$$

where $\models_{\text{CL}}$ is classical entailment, and hence this was justifiably called an *approximation of classical logic from below*.

A family of logics that approximates classical logic from below is useful for theorem proving. For in such case, if a $B \models_S^3 \alpha$ in logic $S_3(S)$, then we know that classically $B \models \alpha$. So if it is more efficient to do theorem proving in $S_3(S)$, we may prove some classical theorems at a “reduced cost” as theorem proving is a coNP-complete problem. If we fail to prove a theorem in $S_3(S)$, however, we don’t know its classical status; it may be provable in $S_3(S')$ for some $S' \supset S$, or it may be that classically $B \not\models \alpha$. The method for theorem proving in S_3 presented in [FW01] had the advantage of providing an incremental method of theorem proving; that is, if we failed to prove $B \models_S^3 \alpha$, a method was provided for incrementing S and continuing the proof without restarting the proof.

Besides the potential economy in theorem proving, logic $S_3(S)$, by means of its parameter S gives us a clear notion of what propositional symbols are *relevant* for the proof of $B \models \alpha$.

Similarly, we say that a family of parameterised logics $L(S)$ is an *approximation of classical logic from above* if we have:

$$\models_{\emptyset}^L \supseteq \models_{S'}^L \supseteq \dots \supseteq \models_{S'^n}^L \supseteq \models_{\mathcal{P}}^L = \models_{\text{CL}}$$

In a dual way, a family of logics that approximates classical logic from above is useful for *disproving* theorems. That is, if we show that $B \not\models_{\mathcal{L}}^L \alpha$ then we classically know that $B \not\models \alpha$, with the advantage of disproving a theorem at a reduced cost, for the problem in classical logic is the SAT-problem, and therefore NP-complete. Similarly, the parameter S gives us a clear notion of what propositional symbols are *relevant* for disproving a theorem (i.e. for satisfying its negation).

Unfortunately, S_1 does not approximate classical logic from above. In fact, if S_1 approximated classical logic from above, one would expect any classical

theorem to be a theorem of $S_1(S)$ for any S . However, the formula $p \vee \neg p$ is false unless $p \in S$ and hence the logic S_1 does not qualify for an approximation of classical logic from above.

Besides not being an approximation of classical logic from above, there is another limitation in the Cadoli and Schaerf approach which is common to both S_1 and S_3 : The system is restricted to $\rightarrow$ -free formulas and in negation normal form. For the case of S_3 , we have addressed this limitation in [FW01]. We are now going to address this limitation, while also trying to provide a logic that approximates classical logic from above.

Another problem of S_1 is that reasoning within S_1 is not local, at least one literal of each clause must be in S , as it was noted in [tTvH96]. This means that even clauses which are completely irrelevant for disproving the given formula will be examined.

In the next section, we present a system that approximates classical logic without suffering from these limitations.

4 The Family of Logics $\mathbf{s}_1$

The problem of creating a logic that approximates classical logic from above comes from the following fact. Any logic that is defined in terms of a binary valuation $v : \mathcal{L} \rightarrow \{0, 1\}$ that properly extends classical logic is inconsistent. This is very simple to see. If it is a proper extension of classical logic, it will contradict a classical validity. Since it is an extension of classical logic, from this contradiction any formula is derivable.

The way Cadoli and Schaerf avoided this problem was not to make its binary valuation a full extension of classical logic. Here, we take a different approach, for we want to construct an extension of classical entailment, and define a ternary valuation, that is, we define a valuation $v_s^1(\alpha) \subseteq \{0, 1\}$; later we show that $v_s^1(\alpha) \neq \emptyset$.

For that, consider the full language of classical logic based on a set of proposition symbols $\mathcal{P}$. We define the family of logics $\mathbf{s}_1(s)$, parameterised by the set $s \subseteq \mathcal{P}$. Let α be a formula and let $prop(\alpha)$ be the set of propositional symbols occurring in α . We say that $\alpha \in s$ iff $prop(\alpha) \subseteq s$.

Let v_p be a classical propositional valuation. Starting from v_p , we build an $\mathbf{s}_1$ -valuation $v_s^1 : \mathcal{L} \rightarrow 2^{\{0,1\}}$, by defining when $1 \in v_s^1(\alpha)$ and when $0 \in v_s^1(\alpha)$. This definition is parameterised by the set $s \subseteq \mathcal{P}$ in the following way. Initially, for propositional symbols, v_s^1 extends v_p :

$$\begin{aligned} 0 \in v_s^1(p) &\Leftrightarrow v_p(p) = 0 \\ 1 \in v_s^1(p) &\Leftrightarrow v_p(p) = 1 \text{ or } p \notin s \end{aligned}$$

That is, v_s^1 extends v_p but whenever we have an atom $p \notin s$, $1 \in v_s^1(p)$; if $p \notin s$ and $v_p(p) = 0$, we get $v_s^1(p) = \{0, 1\}$. The rest of the definition of v_s^1 proceeds in the same spirit, as follows:

$$\begin{aligned}
0 \in v_s^1(\neg\alpha) &\Leftrightarrow 1 \in v_s^1(\alpha) \\
0 \in v_s^1(\alpha \wedge \beta) &\Leftrightarrow 0 \in v_s^1(\alpha) \text{ or } 0 \in v_s^1(\beta) \\
0 \in v_s^1(\alpha \vee \beta) &\Leftrightarrow 0 \in v_s^1(\alpha) \text{ and } 0 \in v_s^1(\beta) \\
0 \in v_s^1(\alpha \rightarrow \beta) &\Leftrightarrow 1 \in v_s^1(\alpha) \text{ and } 0 \in v_s^1(\beta) \\
1 \in v_s^1(\neg\alpha) &\Leftrightarrow 0 \in v_s^1(\alpha) \quad \text{or } \neg\alpha / \mathfrak{E} \\
1 \in v_s^1(\alpha \wedge \beta) &\Leftrightarrow 1 \in v_s^1(\alpha) \text{ and } 1 \in v_s^1(\beta) \text{ or } \alpha \wedge \beta / \mathfrak{E} \\
1 \in v_s^1(\alpha \vee \beta) &\Leftrightarrow 1 \in v_s^1(\alpha) \text{ or } 1 \in v_s^1(\beta) \quad \text{or } \alpha \vee \beta / \mathfrak{E} \\
1 \in v_s^1(\alpha \rightarrow \beta) &\Leftrightarrow 0 \in v_s^1(\alpha) \text{ or } 1 \in v_s^1(\beta) \quad \text{or } \alpha \rightarrow \beta / \mathfrak{E}
\end{aligned}$$

We start pointing out two basic properties of v_s^1 , namely that is a ternary relation and that $1 \in v_s^1(\alpha)$ whenever $\alpha / \mathfrak{E}$.

Lemma 1. *Let α be any formula. Then*

- (a) $v_s^1(\alpha) / \neq \emptyset$.
- (b) If $\alpha / \mathfrak{E}$ then $1 \in v_s^1(\alpha)$.

Proof. Let α be any formula. Then:

- (a) First note that for any propositional symbol, $v_p(p) \in v_s^1(p)$, so $v_s^1(p) / \neq \emptyset$. Then a simple structural induction on α shows that $v_s^1(\alpha) / \neq \emptyset$.
- (b) Straight from the definition of v_s^1 . □

It is interesting to see that in one extreme, i.e., when $s = \emptyset$, s_1 -valuations trivialise, assigning the value 1 to every formula in the language. When $s = \mathcal{P}$, s_1 -valuations over the connectives correspond to Kleene's semantics for three valued logics [Kle38].

The next important property of v_s^1 is that it is an extension of classical logic in the following sense. Let v_s^1 be an s_1 -valuation; its *underlying propositional valuation*, v_p is given by

$$\begin{aligned}
v_p(p) &= 0, 0 \in v_s^1(p) \\
v_p(p) &= 1, 0 / \notin v_s^1(p)
\end{aligned}$$

as can be inspected from definition of v_s^1 . Also note that v_p and s uniquely define v_s^1 .

Lemma 2. *Let $v_c : \mathcal{L} \rightarrow \{0, 1\}$ be a classical binary valuation extending v_p . Then, for every formula α , $v_c(\alpha) \in v_s^1(\alpha)$.*

Proof. By structural induction on α . It suffices to note that the property is valid for $p \in \mathcal{P}$. Then a simple inspection of the definition of v_s^1 gives us the inductive cases. □

Just note that Lemma 2 implies Lemma 1(a). We can also say that if $\alpha \in s$, then v_s^1 behaves *classically* in the following sense.

Lemma 3. *Let v_p be a propositional valuation and let v_s^1 and v_c be, respectively, its $s_1(s)$ and classical extensions. If $\alpha \in s$, $v_s^1(\alpha) = \{v_c(\alpha)\}$.*

Proof. A simple inspection of the definition of v_s^1 shows that if $\alpha \in s$, v_s^1 behaves classically. $\square$

Finally, we compare $\mathbf{s}_1$ -valuations under expanding sets s .

Lemma 4. *Suppose $s \subseteq s'$ and let $v_s^1(\alpha)$ and $v_{s'}^1(\alpha)$ extend the same propositional valuation. Then $v_s^1(\alpha) \supseteq v_{s'}^1(\alpha)$.*

Proof. If $\alpha \in s$, $v_s^1(\alpha)$ and $v_{s'}^1(\alpha)$ behave classically. If $\alpha \notin s$, then $1 \in v_s^1(\alpha)$ and we have to analyse what happens when $0 \in v_{s'}^1(\alpha)$. By structural induction on α , we show that $0 \in v_s^1(\alpha)$.

For the base case, just note that v_s^1 and $v_{s'}^1$ have the same underlying propositional valuation.

Consider $0 \in v_{s'}^1(\neg\alpha)$, then $1 \in v_s^1(\alpha)$. Since $\alpha \notin s$, $1 \in v_s^1(\alpha)$, so $0 \in v_s^1(\neg\alpha)$.

Consider $0 \in v_{s'}^1(\alpha \rightarrow \beta)$, then $1 \in v_{s'}^1(\alpha)$ and $0 \in v_{s'}^1(\beta)$. By the induction hypothesis, $0 \in v_s^1(\beta)$. If $\alpha \notin s$, $1 \in v_s^1(\alpha)$ and we are done. If $\alpha \in s$, then also $\alpha \in s'$, $v_s^1(\alpha)$ and $v_{s'}^1(\alpha)$ behave classically and agree with each other, so $1 \in v_s^1(\alpha)$ and we are done.

The cases where $0 \in v_{s'}^1(\alpha \wedge \beta)$ and $0 \in v_{s'}^1(\alpha \vee \beta)$ are straightforward consequences of the induction hypothesis. $\square$

The next step is to define the notion of a $\mathbf{s}_1$ -entailment.

4.1 $\mathbf{s}_1$ -Entailment

The idea is to define an entailment relation for $\mathbf{s}_1$, $\models_s^1$, parameterised on the set $s \subseteq \mathcal{P}$ so as to extend for any s the classical entailment relation

$$B \models \alpha$$

To achieve that, we have to make valuations applying on the left handside of $\models_s^1$ to be *stricter* than classical valuations, and the valuations that apply to the right handside of $\models_s^1$ to be more *relaxed* than classical valuations, for every $s \subseteq \mathcal{P}$. This motivates the following definitions.

Definition 1. *Let $\alpha \in \mathcal{L}$ and let v_s^1 be a $\mathbf{s}_1$ -valuation. Then:*

- *If $v_s^1(\alpha) = \{1\}$ then we say that α is strictly satisfied by v_s^1 .*
- *If $1 \in v_s^1(\alpha)$ then we say that α is relaxedly satisfied by v_s^1 .*

That these definitions are the desired ones follows from the following.

Lemma 5. *Let $\alpha \in \mathcal{L}$. Then:*

- (a) *α is strictly satisfiable implies that α is classically satisfiable.*
- (b) *α is classically satisfiable implies that α is relaxedly satisfiable.*

Proof.

- (a) Consider v_s^1 such that $v_s^1(\alpha) = \{1\}$. Let v_p be its underlying propositional valuation and let v_c be a classical valuation that extends v_p . Since $0 \neq v_s^1(\alpha)$, by Lemma 2 we have that $v_c(\alpha) \neq 0$, so $v_c(\alpha) = 1$.
- (b) Consider a classical valuation v_c such that $v_c(\alpha) = 1$. Let v_p be its underlying propositional valuation. Then directly from Lemma 2, $1 \in v_s^1(\alpha)$. $\square$

We are now in a position to define the notion of s_1 -entailment.

Definition 2. We say that $\beta_1, \dots, \beta_m \models_s^1 \alpha$ iff all s_1 -valuation v_s^1 that strictly satisfies all β_i , $1 \leq i \leq m$, relaxedly satisfies α .

The following are important properties of s_1 -entailment.

Lemma 6.

- (a) $B \models_{\emptyset}^1 \alpha$, for every $\alpha \in \mathcal{L}$.
- (b) $\models_P^1 = \models_{CL}$.
- (c) If $s \subseteq s'$, $\models_s^1 \supseteq \models_{s'}^1$.

Proof.

- (a) By Lemma 1(b), $1 \in v_{\emptyset}^1(\alpha)$, for every $\alpha \in \mathcal{L}$.
- (b) By Lemma 3, v_P^1 is a classical valuation, and the notions of strict, relaxed and classical valuation coincide.
- (c) Suppose $s \subseteq s'$, $B \models_{s'}^1 \alpha$ but $B \not\models_s^1 \alpha$. Then exists v_s^1 such that $v_s^1(\beta_i) = \{1\}$, for all $\beta_i \in B$ but $v_s^1(\alpha) = \{0\}$. Let $v_{s'}^1$ be the s_1 -valuation generated by v_s^1 underlying propositional valuation. From Lemma 4 we have that $v_{s'}^1(\beta_i) = \{1\}$, for all $\beta_i \in B$. Since $B \models_{s'}^1 \alpha$, we have that $1 \in v_{s'}^1(\alpha)$. Again by Lemma 4 we get $1 \in v_s^1(\alpha)$, which contradicts $v_s^1(\alpha) = \{0\}$. So $B \models_s^1 \alpha$. $\square$

From what has been shown, it follows directly that this notion of entailment is the desired one.

Theorem 2. The family of s_1 -logics approximates classical entailment from above, that is:

$$\models_{\emptyset}^1 \supseteq \models_{s'}^1 \supseteq \dots \supseteq \models_{s'^n}^1 \supseteq \models_P^1 = \models_{CL}$$

Proof. Directly from Lemma 6. $\square$

It is interesting to point that if v_s^1 is a s_1 -valuation falsifying $B \models_s^1 \alpha$, we have a classical valuation v_c that falsifies $B \models \alpha$ built as an extension of the propositional valuation v_p such that $v_p(p) = 1 \Leftrightarrow v_s^1(p) = \{1\}$.

One interesting property that fails for s_1 -entailment is the *deduction theorem*. One half of it is still true, namely that

$$B \models_s^1 \alpha \Rightarrow \models_s^1 (\bigwedge B) \rightarrow \alpha$$

However, the converse is not true. Here is a counterexample. Suppose $q \neq \alpha$ and $p \in s$, so $q \rightarrow p \neq \alpha$. Then $\models_s^1 q \rightarrow p$; take a valuation that makes $v_s^1(q) = 1$ and $v_s^1(p) = 0$, hence $q \not\models_s^1 p$.

5 Examples

In this section, we examine some examples and compare $\mathbf{s}_1$ to Cadoli and Schaerf's S_1 . We have already seen that, unlike S_1 entailment, $\mathbf{s}_1$ entailment truly approximates classical entailment from above.

Let us have a look at what happens with Example 2 when we use $\mathbf{s}_1$ entailment:

Example 3 (Example 2 revisited).

We want to check whether $B \not\models \beta$, where $\beta = \neg\text{child} \vee \text{pensioner}$ and

$$B = \{ \neg\text{person} \vee \text{child} \vee \text{youngster} \vee \text{adult} \vee \text{senior}, \\ \neg\text{adult} \vee \text{student} \vee \text{worker} \vee \text{unemployed}, \\ \neg\text{pensioner} \vee \text{senior}, \neg\text{youngster} \vee \text{student} \vee \text{worker}, \\ \neg\text{senior} \vee \text{pensioner} \vee \text{worker}, \neg\text{pensioner} \vee \neg\text{student}, \\ \neg\text{student} \vee \text{child} \vee \text{youngster} \vee \text{adult}, \\ \neg\text{pensioner} \vee \neg\text{worker} \}.$$

It is not difficult to see that with $s = \{\text{child}, \text{pensioner}\}$, we can take a propositional valuation v_p such that $v_p(\text{pensioner}) = 0$ and $v_p(p) = 1$ for p any other propositional letter, such that the $\mathbf{s}_1$ -valuation obtained from v_p strictly satisfies every formula in B but does not relaxedly satisfy β . Hence, we have that $B \not\models_{\mathbf{s}} \beta$, and $B \not\models \beta$.

This example shows that we can obtain an answer to the question of whether $B \not\models \beta$ with a set s smaller than the set S needed for S_1 .

Another concern was the fact that S_1 did not allow for local reasoning. Consider the following example, borrowed from [CPW01]:

Example 4. The following represents beliefs about a young student, Hans.

$$B = \{ \text{student}, \text{student} \rightarrow \text{young}, \text{young} \rightarrow \neg\text{pensioner}, \\ \text{worker}, \text{worker} \rightarrow \neg\text{pensioner}, \\ \text{blue-eyes}, \text{likes-dancing}, \text{six-feet-tall} \}.$$

We want to know whether Hans is a pensioner.

We have seen that in order to use Cadoli and Schaerf's S_1 , we had to start with a set S containing at least one atom of each clause. This means that when we build S , we have to take into account even clauses which are completely irrelevant to the query, as `likes-dancing`.

In our system, formulas not in s will be automatically set to 1. If we have $s = \{\text{pensioner}\}$, a propositional valuation such that $v_p(\text{pensioner}) = 0$ and $v_p(p) = 1$ for p any other propositional letter, can be extended to an $\mathbf{s}_1$ -valuation that strictly satisfies B but does not relaxedly satisfy `pensioner`. Hence, $B \not\models_{\mathbf{s}} \text{pensioner}$.

It is not difficult to see that, unlike in Cadoli and Schaerf's S_1 and S_3 , the classical equivalences of the connectives hold in $\mathbf{s}_1$, which means that we do not have any gains in terms of the size of the set s using different equivalent forms of the same knowledge base.

6 Conclusions and Future Work

We have proposed a system for approximate entailment that can be used for approximating classical logic “from above”, in the sense that at each step, we prove less theorems, until we reach classical logic. The system proposed is based on a three-valued valuation and a different notion of entailment, where the logic on the right hand side of the entailment relation does not have to be the same as the logic on the left hand side. This sort of “hybrid” entailment relation has been proposed before in Quasi-Classical Logic [Hun00].

Future work includes the study of the formal relationship between our system and other three-valued semantics and the design of a tableaux proof method for the logic, following the line of [FW01].

References

- [AB75] A.R. Anderson and N.D. Belnap. *Entailment: The Logic of Relevance and Necessity*, Vol. 1. Princeton University Press, 1975. 23
- [CPW01] Samir Chopra, Rohit Parikh, and Renata Wassermann. Approximate belief revision. *Logic Journal of the IGPL*, 9(6):755-768, 2001. 29
- [CS95] Marco Cadoli and Marco Schaerf. Approximate inference in default logic and circumscription. *Fundamenta Informaticae*, 23:123-143, 1995. 24
- [CS96] Marco Cadoli and Marco Schaerf. The complexity of entailment in propositional multivalued logics. *Annals of Mathematics and Artificial Intelligence*, 18(1):29-50, 1996. 24
- [FW01] Marcelo Finger and Renata Wassermann. Tableaux for approximate reasoning. In Leopoldo Bertossi and Jan Chomicki, editors, *IJCAI-2001 Workshop on Inconsistency in Data and Knowledge*, pages 71-79, Seattle, August 6-10 2001. 24, 25, 30
- [GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979. 21
- [Hun00] A. Hunter. Reasoning with contradictory information in quasi-classical logic. *Journal of Logic and Computation*, 10(5):677-703, 2000. 30
- [Kle38] S. C. Kleene. On a notation for ordinal numbers. *Journal of Symbolic Logic*, 1938. 26
- [Lev84] Hector Levesque. A logic of implicit and explicit belief. In *Proceedings of AAAI-84*, 1984. 23
- [SC95] Marco Schaerf and Marco Cadoli. Tractable reasoning via approximation. *Artificial Intelligence*, 74(2):249-310, 1995. 21, 22, 23
- [tTvH96] Annette ten Teije and Frank van Harmelen. Computing approximate diagnoses by using approximate entailment. In *Proceedings of KR’96*, 1996. 25

Attacking the Complexity of Prioritized Inference Preliminary Report

Renata Wassermann¹ and Samir Chopra²

¹ Department of Computer Science, University of Sao Paulo
Sao Paulo, Brazil
`renata@ime.usp.br`

² School of Computer Science and Engineering
University of New South Wales
Sydney, NSW, Australia
`schopra@cse.unsw.edu.au`

Abstract. In the past twenty years, several theoretical models (and some implementations) for non-monotonic reasoning have been proposed. We present an analysis of a model for *prioritized inference*. We are interested in modeling resource-bounded agents, with limitations in memory, time, and logical ability. We list the computational bottlenecks of the model and suggest the use of some existent techniques to deal with the computational complexity. We also present an analysis of the tradeoff between formal properties and computational efficiency.

1 Introduction

We are often confronted with situations where we must reason in the absence of complete information, and draw conclusions that can be later retracted. You may conclude that it has rained after seeing the street wet. If later on you find out that someone has washed the street, you give up your previous conclusion. This kind of reasoning is *non-monotonic*, as the set of possible inferences does not grow monotonically upon addition of new information. Several formal systems have been proposed to model non-monotonic reasoning [Rei80, Moo88, McC80], but they ended up being computationally harder than classical logic. Prioritized inference [Bre94] assigns degrees of certainty to formulas. If two formulas contradict each other, the one with the highest degree “wins”. Non-monotonicity arises when one adds a formula that cancels some previous inference. Inference then is not as hard as in other formalisms for non-monotonic reasoning, but there is the addition burden of having to rank formulas. As we will see, for some applications, this ranking is already given with the problem.

In this paper, we analyze a particular proposal for prioritized inference – first presented in [CGP01] – which takes relevance into account in order to minimize the search space for a proof. Although intuitively appealing, the model uses some computationally expensive operations. We list the bottlenecks of the model and show how they can be dealt with. Every computational improvement in this model involves the loss of some formal properties. In each case, we make explicit

the tradeoff involved. Interestingly, the formal properties lost are not found in realistic agents that have to reason in real time. And, given enough time and memory, the system proposed here finds the “right” solution, i.e., the solution that would be found by the original proposal.

Notation: We assume a finite propositional language $\mathcal{L}$ built from a set of atoms $\mathbf{Atm} = \{p, q, r, \dots\}$ and equipped with the usual connectives and constants $\wedge, \vee, \rightarrow, \leftrightarrow, \neg, \perp, \top$. The symbol $\vdash$ denotes classical derivability; subscripts will denote alternative relations. A literal is either an atom or a negated atom. A clause is a disjunction of literals. Greek lowercase letters $\alpha, \beta, \dots$ stand for formulas. Uppercase letters $A, B, C, \dots, \Delta, \Gamma, \dots$ stand for sets of formulas. $\mathbf{Atm}(\alpha)$ is the set of atoms that occur in α , $\mathbf{Atm}_{\min}(\alpha)$ is the minimal set of atoms needed to express a formula logically equivalent to α ¹. If $\alpha = p \wedge (q \vee \neg q)$ then $\mathbf{Atm}(\alpha) = \{p, q\}$ while $\mathbf{Atm}_{\min}(\alpha) = \{p\}$, since $\alpha \equiv p$.

2 The Model

In this section, we present the formal model, introduced in [CGP01], that will serve as a base for the development of our computational model. As a motivation, we use the example of bank transactions. We have three sources of information: the system, the manager, and the client. Information coming from the manager is more reliable than from the system which is in turn more reliable than information coming from the client. In case of conflict, more recent pieces of information have preference over older ones.

18/04 client: Credit 5	08/06 manager: Offer credit card.
20/04 client: Debit 2	10/06 client: Debit 2
28/04 system: Balance 3	15/06 client: Debit 3
03/05 client: Credit 4	20/06 client: Debit 2
08/05 manager: Good client.	23/06 client: Debit 3
11/05 client: Credit 3	28/06 system: Balance -1
15/05 client: Change of address.	08/07 manager: Cancel credit card.
20/05 client: Debit 1	28/07 system: Balance -1
28/05 system: Balance 9	08/08 manager: Bad client.
05/06 client: Asks credit card.	...

If we want to know the client’s situation, we start from the most recent pieces of information. From the bank’s point of view, the client is now considered a bad one, even if the manager previously assessed him as a good client. And the client does not have a credit card, even if he had one before.

This is a typical example of day-to-day non-monotonic reasoning. The knowledge base is linearly ordered, therefore represented by a sequence. The use of a linear ordering can be interpreted in many ways. In applications as the one above, recent beliefs are more important than old ones, and the linear order represents recency. The linear ordering may also be a combination of several orderings (as in [Rya93]), representing, for example, the reliability of the source,

¹ Parikh has shown in [Par96] that the minimal set of atoms is unique.

recency, some measure of probability. In our example, even if the system stated that the client was good, the manager's last statement would "overwrite" it.

The main idea of the model is that when a query is made, the sequence is reordered, according to the relevance of the formulas to the query. We reduce the search space for a proof or refutation of the formula. Considering the bank example, suppose that the query is whether the client can get a loan and the system has rules such as "To get a loan, client must have a credit card", "To get a loan, client must be rated good", etc. The system should be able to collect information about the client having a credit card and being rated good or not. Irrelevant information (his address has changed) does not need to be considered.

In the rest of this section, we present the formal model that we will use.

Definition 1. *A belief sequence B is a linearly ordered multiset of formulas i.e., $B = \langle \beta_1, \dots, \beta_n \rangle$ where for any pair of beliefs β_i, β_j if $i < j$, $\beta_j \sqsubseteq \beta_i$.*

In what follows, whenever we refer to a belief sequence, we will assume an underlying linear ordering. The following relation of *relevance* is used:

Definition 2. *α, β are directly relevant if $\mathbf{Atm}_{\min}(\alpha) \cap \mathbf{Atm}_{\min}(\beta) \neq \emptyset$.*

Although we use the above (rather simplistic) notion throughout this paper, all subsequent definitions hold for any other notion of relevance.

Definition 3. *Given a belief sequence B , two formulas α, β are k -relevant w.r.t. B if $\exists \chi_1, \chi_2, \dots, \chi_k \in B$ such that: (i) α, χ_1 are directly relevant; (ii) χ_i, χ_{i+1} are directly relevant for $i = 1, \dots, k-1$; and (iii) χ_k, β are directly relevant. $\mathcal{R}_k(\alpha, \beta, B)$ indicates that α, β are at least k -relevant w.r.t. B . If $k = 0$, the formulas are directly relevant. Two formulas are irrelevant if they are not k -relevant for any finite k . We set $\text{rel}(\alpha, \beta, B)$ as the lowest k s.t. $\mathcal{R}_k(\alpha, \beta, B)$.*

Note that the degree of relevance of formulas depends on the belief sequence. New information is added to belief sequences by simply appending it at the end. *Prioritized inference* on a belief sequence B employs a consistent subset of B . Consider a formula γ expressed using only $\mathbf{Atm}_{\min}(\gamma)$. A maxiconsistent subset $\Gamma_{\langle B, k, \gamma \rangle}$ (of formulas k -relevant to γ) of B is constructed, regulated by the ordering $\prec$ that γ creates on B , reshuffling $\beta_1, \dots, \beta_n$ into $\delta_1, \dots, \delta_n$:

Definition 4. *Given a formula γ and a belief sequence B , $\beta, \beta' \in B$, $\beta \prec \beta'$ if either (a) $\text{rel}(\gamma, \beta, B) < \text{rel}(\gamma, \beta', B)$ (β is more relevant to γ than β'); or (b) β, β' are equally relevant ($\text{rel}(\gamma, \beta, B) = \text{rel}(\gamma, \beta', B)$) but $\beta' \sqsubseteq \beta$*

The new sequence is now ordered according to decreasing relevance, with lower indexed formulas being more relevant than those with higher indexes.

The $\delta_1, \dots, \delta_n$ are the $\beta_1, \dots, \beta_n$ under this order. In the definition below Γ is the set $\Gamma_{\langle B, k, \gamma \rangle}$ and k is a preselected level of relevance.

Definition 5. $\Gamma^0 = \emptyset$, and Γ^{i+1} is given by (a) Γ^i if either $\Gamma^i \vdash \neg \delta_{i+1}$ or if $\neg \mathcal{R}_k(\delta_{i+1}, \gamma, B)$; (b) $\Gamma^i \cup \{\delta_{i+1}\}$ otherwise. $\Gamma_{\langle B, k, \gamma \rangle} = \Gamma^n$.

Formulas are added to $\Gamma_{\langle B, k, \gamma \rangle}$ in order of their decreasing relevance to γ . The lower the level of relevance allowed (i.e., the higher the value of k), the larger the part of B considered. If $B = \langle p, \neg p, q, p \vee q \rangle, \gamma = p$, then $\Gamma_{\langle B, 0, p \rangle} = \{p \vee q, p\}$ and $\Gamma_{\langle B, 1, p \rangle} = \{p \vee q, p, q\}$. We define k -inference as:

Definition 6. $B \vdash_k \gamma$ iff $\Gamma_{\langle B, k, \gamma \rangle} \vdash \gamma$ ²

The inference operation defined above enables a query answering scheme. If $\Gamma_{\langle B, k, \gamma \rangle} \vdash \gamma$, the agent answers ‘yes’ and if $\Gamma_{\langle B, k, \gamma \rangle} \vdash \neg \gamma$ the agent answers ‘no’. Otherwise, the agent answers ‘no information’. Even if B is classically inconsistent, the agent is able answer consistently every consistent query. For example, suppose that besides relevance, a temporal ordering is used, as in [CGP01]. Consider $B = \langle p, \neg p \wedge \neg q \rangle$. $\neg p \wedge \neg q$ overrides p ($\Gamma_{\langle B, 0, p \rangle}$ is $\{\neg p \wedge \neg q\}$) and so $B \not\vdash p$. However, $B + (p \vee q) \vdash_0 p$, since newer information overrides $\neg p \wedge \neg q$ ($\Gamma_{\langle B + p \vee q, 0, p \rangle}$ is $\{p, p \vee q\}$); the latest information decreases the reliability of $\neg p \wedge \neg q$ and p regains its original standing. Conclusions sanctioned are dependent on whether new information arrives “in several pieces” or as a single formula. Receiving two pieces of information individually and together can have different effects on an epistemic state. If α and β are received separately, then α can stand without β . But if the *conjunction* $\alpha \wedge \beta$ is received, then undermining one will undermine both. Furthermore, new inputs can block previously possible derivations and provide a modeling for loss of belief in a proposition. Agents do not lose beliefs without a reason: to drop the belief that α is to add information that undermines α . Still, it is possible to lose α without acquiring $\neg \alpha$. Consider $B = p \wedge q$ and $B + (\neg p \vee \neg q)$. The new sequence no longer answers ‘yes’ to p but neither does it answer ‘yes’ to $\neg p$. $(\neg p \vee \neg q)$ has undermined $p \wedge q$ without actually making $\neg p$ derivable. Theories corresponding to positive answers to queries are defined as follows:

Definition 7. $C_k(B) = \{\gamma \mid B \vdash_k \gamma\}$; $C(B) = \{\gamma \mid \exists k, B \vdash_k \gamma\} = \bigcup C_k(B)$

With unlimited computational resources, $C(B)$ would be the desired extension of the belief sequence since $C_k(B) \subseteq C_{k+1}(B)$; a smaller k conserves computational resources. $\vdash_k$ is *monotonic* in k , the degree of relevance, and *non-monotonic* in expansions of a belief sequence.

3 Towards Implementation

The model presented above is intuitive and has interesting formal properties. We now analyze its sources of complexity and suggest techniques to handle these. There is no magic in what we suggest here: in each suggestion, we are sacrificing one property of the model for computational efficiency.

² The construction of $\Gamma_{\langle B, k, \gamma \rangle}$ is a particular case of *local maxichoice consolidation* [HW02]. Local consolidation is defined as first finding the relevant subset of the belief base and then making it consistent. Maxichoice consolidation selects a maximal consistent subset of the base. Definition 5 shows one way of selecting such a set, given an ordered belief base.

For any query γ , calculating $\mathbf{Atm}_{\min}(\gamma)$ is a co-**NP** complete problem [HR99] in the length of γ . Furthermore, the construction of the set Γ (the maxi-consistent set relevant to γ) requires checking the consistency of Γ_i at each step. The construction of Γ is structured so as to pick the most important formulas in any query operation, since even in the case that a large number of formulas are found to be k -relevant, only the ones highest in the linear ordering $\sqsubseteq$ will be retrieved first. Controlling k and using $\sqsubseteq$ to break ties, we can keep the set Γ small. In so doing, we trade the completeness of the search for efficiency.

The query scheme above first calculates the relevance relation for the entire sequence and then constructs the set Γ . Since each stage of the construction involves a consistency check, the complexity of the procedure is polynomial with an **NP** oracle, but only in the size of the set of relevant atoms variables which is small for a suitably small value of k – a smaller k implies using only the “most” relevant formulas. In checking for k -derivability, costs are reduced sharply when most formulas in the sequence are not k -relevant and the size of $\Gamma_{(B,k,\gamma)}$ is small. Relevance relations cut down the effort involved in these cases. In conclusion, while the basic model itself is quite tractable, we would like to go further. There are three computational bottlenecks: the calculation of the minimal set of atoms for each formula, the use of consistency checks in the query operation and the collection of relevant formulas in the reordered sequences. We now suggest techniques to attack each source of complexity. We do not claim to beat established complexity results, but we expect that on the average case, the heuristics suggested drastically reduce search spaces.

3.1 Prime Implicates in Relevance Tests

Our objective is to minimize the computational cost of calculating the minimal set of atoms for a formula and to use a tractable form of inference in the query scheme. For the latter we would like formulas to be internally represented as clauses. The calculation of prime implicates for formulas in the belief base – and for all queries – accomplishes both. A clause c is an *implicate* of a formula α iff $\alpha \vdash c$. A clause c is a *prime implicate* of α iff for all implicates c' of α such that $c' \vdash c$, it is the case that $c \vdash c'$. A set $\mathcal{D}$ of prime implicates is a *covering* of α iff for every clause c such that $\alpha \vdash c$, there exists $c' \in \mathcal{D}$ such that $c' \vdash c$. Let $\mathbf{Atm}(\mathcal{D})$ be the set of propositional atoms in $\mathcal{D}$. Then note that $\mathbf{Atm}(\mathcal{D}) = \mathbf{Atm}_{\min}(\alpha)$ and $\vdash \bigwedge \mathcal{D} \leftrightarrow \alpha$ [HR99].

In the case of addition of new information, we add a covering set of prime implicates for the new input to the belief sequence; the sequence then is a set of covering sets of prime implicates. Our motivation for using covering sets of prime implicates is that since computational effort is required in constructing $\mathbf{Atm}_{\min}(\alpha)$ we calculate it indirectly and amortize the cost over subsequent query operations. The tradeoff is that of balancing the time complexity of calculating $\mathbf{Atm}_{\min}$ versus the exponential blowup in size. However, the calculation of prime implicates has the desirable effect that it facilitates consistency checks. While we are stuck with certain baseline computational costs, we can reuse our efforts. Note that several theorem provers (e.g., [MW97]) first transform the

formulas into clausal form. We can avail ourselves of several algorithms for calculating prime implicates, including incremental ones [Mar95, MS96]. The use of prime implicates in this way has the theoretical advantage that we do not restrict the form of formulas in the belief sequence (which could be viewed as a restriction on expressivity) but instead, only use the clausal form at the time of querying. Of course, searching for relevant clauses depends now on the size of the belief sequence, which may be exponential in the number of original formulas entered. A possible solution is to maintain a table of atoms linked to the clauses where they occur. This will be explored in Section 3.3. Having transformed the belief sequence into clausal form, we can use approximate inference.

3.2 Approximate Inference

Cadoli and Schaerf [SC95] define two approximations of classical entailment: $\vdash_S^1$ which is complete but unsound and $\vdash_S^3$ which is sound and incomplete. These approximations are carried out over a set of atoms $S \subseteq \mathbf{Atm}$ which determines their closeness to classical entailment. When $S = \mathbf{Atm}$, classical entailment is obtained; when $S = \emptyset$, $\vdash_S^1$ holds for any two formulas and $\vdash_S^3$ corresponds to Levesque's logic for explicit beliefs [Lev84]. In an S_1 assignment, for an atom $p \in S$, $p, \neg p$ are given opposite truth values; if $p \notin S$, then $p, \neg p$ both get the value 0. In an S_3 assignment, if $p \in S$, then $p, \neg p$ get opposite truth values, while if $p \notin S$, $p, \neg p$ do not both get 0, but may both get 1. The belief base B is assumed to be in clausal form. Since $\vdash_S^3$ is sound but incomplete, it can be used to approximate $\vdash$, i.e., if for some S we have that $B \vdash_S^3 \alpha$, then $B \vdash \alpha$. On the other hand, since $\vdash_S^1$ is unsound but complete, it can be used for approximating $\not\vdash$, i.e., if for some S we have that $B \not\vdash_S^1 \alpha$, then $B \not\vdash \alpha$. The application of the non-standard truth assignments allows for a reduction in the size of the belief base to be checked for classical satisfiability.

Approximate inference has been successfully applied to model-based diagnosis [tTvH96] and belief revision [CPW01]. We propose that the inference relation $\vdash_k$ can be based on approximated inference instead of classical inference. That is, $\Gamma_{\langle B, k, \gamma \rangle} \not\vdash_S^1 \gamma \Rightarrow B \not\vdash_k \gamma$ and $\Gamma_{\langle B, k, \gamma \rangle} \vdash_S^3 \gamma \Rightarrow B \vdash_k \gamma$. Employing approximate inference relations as the background inference relation conforms to basic intuitions about querying. We obtain a tractable means for confirming disbelief in a proposition by employing the $\vdash_S^1$ relation and similarly for confirming belief by employing $\vdash_S^3$. The following is a view of querying operations:

1. For each epistemic input α , calculate a covering set of prime implicates $\mathcal{D}_\alpha$.
2. For each query γ , test whether $\mathbf{Atm}(\mathcal{D}_\alpha) \cap \mathbf{Atm}(\mathcal{D}_\gamma) \neq \emptyset$.
3. Reorder B based on relevance and ordering $\sqsubseteq$.
4. Construct maxiconsistent subset Γ .
5. Use approximate inference on Γ .

$\vdash_k$ is an inference relation that closely resembles $\vdash_S^3$: it is sound and incomplete and like $\vdash_S^3$ it is a language sensitive relation. In [CPW01] a heuristic for constructing the set S is given which is based on the notion of relevance amongst

formulas: given a query γ and a belief sequence B , we start with $S = \mathbf{Atm}_{\min}(\gamma)$ and proceed by adding relevant atoms. Under some conditions the two relations will be identical. Consider these cases: (i) The belief base is consistent; (ii) The base is inconsistent but the inconsistency is irrelevant to the query; (iii) The base is inconsistent and the inconsistency is in the set of formulas relevant to the query. In cases (i) and (ii), it is possible to give heuristics for finding the set S such that $\vdash_S^3$ coincides with $\vdash_k$:

$$S_0 = \mathbf{Atm}(\alpha); \quad S_{i+1} = S_i \cup \mathbf{Atm}(\{\beta \mid \beta \text{ directly relevant to } \alpha\})$$

Obviously, for any given k , if the set of k -relevant formulas for a query γ is consistent, then $B \vdash_k \gamma$ iff $B \vdash_{S_{k+1}}^3 \gamma$. In case (iii), the two sorts of inferences behave in a different way, since $\vdash_k$ resolves inconsistencies but $\vdash_S^3$ does not. Let $B = \langle p, \neg p \rangle$. Then for any k , $B \not\vdash_k p$, while for any S containing p , $B \vdash_S^3 p$. And for any k , $B \vdash_k \neg p$, while for any S not containing p , $B \not\vdash_S^3 \neg p$. Playing with the parameters k and S allows for fine tuning of the approximation process.

3.3 Structured Bases

To reduce the complexity of the theoretical model, we still have to optimize the collection of relevant formulas, organizing the knowledge base. [Was01] shows how to structure a belief base to find the subset of the base which is relevant to a belief change operation. The method described uses relevance relations between formulas of the belief base. Given a relevance relation $\mathcal{R}$, a belief base is represented as a graph where each node is a formula and with an edge between φ and ψ if and only if $\mathcal{R}(\varphi, \psi)$. The shorter the path between two formulas of the base, the more relevant they are. The connected components partition the graph into unrelated “topics” or “subjects”. Sentences in the same connected component are related, even if far apart. We now show, given the structure of a belief base, how to retrieve the set of formulas relevant to a given formula α :

Definition 8. [Was01] (a) The set of formulas of B which are relevant to α with degree i is given by: $\Delta^i(\alpha, B) = \{\varphi \in B \mid \text{rel}(\alpha, \varphi, B) = i\}$ for $i \geq 0$

(b) The set of formulas of B which are relevant to α up to degree n is given by: $\Delta^{\leq n}(\alpha, B) = \bigcup_{0 \leq i \leq n} \Delta^i(\alpha, B)$ for $n \geq 0$. We say that $\Delta^{<\omega}(\alpha, B) = \bigcup_{i \geq 0} \Delta^i(\alpha, B)$ is the set of formulas relevant to α .

In the notation of Definition 2, $\Delta^{\leq k}(\alpha, B)$ is the set of formulas k -relevant to α with respect to the set B , i.e., $\Delta^{\leq k}(\alpha, B) = \{\beta \mid \text{rel}(\alpha, \beta, B) \leq k\}$. The definition of the sets Δ is used to design an efficient algorithm for retrieval of the set of relevant formulas of a belief base. The method is an interruptible anytime method; whenever it is interrupted, it has retrieved the most relevant beliefs, and the longer it runs, the closer it gets to retrieving all the relevant beliefs (the maximal connected subgraph).³ This is a very desirable property for modeling agents that may not have enough time or memory to find all the related beliefs.

³ Cadoli and Schaerf’s approximate entailment is anytime: larger S give better approximations.

If there is no resource limitation, the method succeeds in retrieving a maximal connected subgraph. [Was01] presents a sketch of an algorithm that takes as input a formula α and a belief base and returns the set of formulas of the base that are relevant to α . As an anytime algorithm, it always returns the set of most relevant beliefs for α . It is a modification of the algorithm for breadth first search in [CLR90] and depending on how the structure is encoded, runs in linear time.

Structuring the formulas in the knowledge bases allows us to re-use relevance relations, which are computed only once.

3.4 The Model Revisited

To sum up, we review the process of building the knowledge base and querying it. First, adding new formulas to the base:

1. Convert formula into a set of prime implicates.
2. Link all formulas directly relevant to new input.

Then, querying the knowledge base:

1. Convert the query into a set of prime implicates.
2. The system looks for formulas that are k -relevant to the query.
3. The retrieved subset of k relevant formulas is ordered according to relevance. To break ties, the underlying linear order is used.
4. A maximal consistent subset Γ of the retrieved set is built starting with the most relevant formula. Checking consistency is facilitated by the use of prime implicates.
5. α can be inferred from the knowledge base if $\Gamma \vdash_S^3 \alpha$; α cannot be inferred from the knowledge base if $\Gamma \not\vdash_S^1 \alpha$; otherwise, we do not know and the set S must be enlarged.

4 Conclusions and Related Work

Implemented systems to deal with inconsistencies in knowledge bases, such as SNeBR [MS88] or systems based on ATMS [dK86] rely on sophisticated data structures to keep the relationship between different beliefs. Our approach differs from those in that we do not always obtain perfect results, but apply “quick and dirty” heuristics which eventually – with enough resources – will lead to the right answer. In this way, we avoid storing too much information. A few implemented systems for belief revision [Dix94, Wil97] are also based on prioritized bases. However, these orderings are hard to recompute when new information is added, since they depend on entailment relations between formulas. Here again, our use of approximate entailment relations provides computational gains since these models are based on classical theorem provers.

We have presented several solutions for reducing the computational costs of non-monotonic inference in the average case. If we want to look for a “complete”

answer, nothing is really gained in the worst case. However, our system allows for partial answers to be obtained with less resources. The logic is parametrized by the degree of relevance k and the context set S . According to the available resources, we can choose appropriate values for k and S .

The formal model presented in Section 2 has been implemented; future work includes applying the techniques described here in order to empirically measure the improvement.

Note that the modifications we suggest here are not just computational ‘tricks’ for better performance, but also reflect intuitive characteristics of realistic agents, such as only examining relevant data and deriving partial conclusions.

One issue not addressed thus far is the notion of ‘forgetting’. Agents with limited resources cannot be expected to store all information received during their lifetime. One way to solve the problem is to have a second temporal ordering which reflects the last time a formula was used. When the agent runs out of memory, formulas not used for a long time can be deleted (“forgotten”).⁴

References

- [AB75] A.R. Anderson and N.D. Belnap. *Entailment: The Logic of Relevance and Necessity, Vol. 1*. Princeton University Press, 1975.
- [Bre94] Gerhard Brewka. Adding priorities and specificity to default logic. In *European Workshop on Logics in Artificial Intelligence (JELIA’94)*, LNAI. Springer-Verlag, 1994. 31
- [CGP01] S. Chopra, K. Georgatos, and R. Parikh. Relevance sensitive non-monotonic inference on belief sequences. *Journal of Applied Non-Classical Logics*, 11(1-2), 2001. 31, 32, 34
- [CLR90] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, 1990. 38
- [CPW01] Samir Chopra, Rohit Parikh, and Renata Wassermann. Approximate belief revision. *Logic Journal of the IGPL*, 9(6):755–768, 2001. 36
- [Dix94] Simon E. Dixon. *Belief Revision: A Computational Approach*. PhD thesis, Basser Department of Computer Science, University of Sydney, 1994. 38
- [dK86] J. Kleer. An assumption-based truth maintenance system. *Art. Intelligence*, 28:127–162, 1986. 38
- [HR99] Andreas Herzig and Omar Rifi. Propositional belief base update and minimal change. *Artificial Intelligence*, 115(1):107–138, 1999. 35
- [HW02] S. O. Hansson and R. Wassermann. Local change. *Studia Logica*, 70(1):49–76, 2002. 34
- [Lev84] H. Levesque. A logic of implicit and explicit belief. In *Proceedings of AAAI-84*, 1984. 36
- [Mar95] Pierre Marquis. Knowledge compilation using theory prime implicates. In *Proceedings of IJCAI95*, pages 837–843, 1995. 36
- [McC80] John McCarthy. Circumscription. *Artificial Intelligence*, 13(1+2):27–39, 1980. 31
- [Moo88] R. C. Moore. Autoepistemic logic. In *Non-Standard Logics for Automated Reasoning*, pages 105–136. Academic Press, London, 1988. 31

⁴ This was suggested in [Was99].

- [MS88] J. Martins and S. Shapiro. A model for belief revision. *Art. Int.*, 35:25–79, 1988. 38
- [MS96] Pierre Marquis and S. Sadaoui. A new algorithm for computing theory prime implicate compilations. In *Proceedings of AAAI96*, pages 504–509, 1996. 36
- [MW97] W. McCune and L. Wos. Otter: The cade-13 competition incarnations. *Journal of Automated Reasoning*, 1997. 35
- [Par96] R. Parikh. Beliefs, belief revision and splitting languages. In *Proceedings Italic-96*, 1996. 32
- [Rei80] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13, 1980. 31
- [Rya93] Mark D. Ryan. Prioritizing preference relations. In *Proceedings of the First Imperial College, Department of Computing, Workshop on Theory and Formal Methods*, 1993. 32
- [SC95] Marco Schaerf and Marco Cadoli. Tractable reasoning via approximation. *Artificial Intelligence*, 74(2):249–310, 1995. 36
- [tTvH96] Annette ten Teije and Frank van Harmelen. Computing approximate diagnoses by using approximate entailment. In *Proceedings of KR'96*, pages 256–265, 1996. 36
- [Was99] R. Wassermann. Resource-bounded belief revision. *Erkenntnis*, 50:429–446, 1999. 39
- [Was01] Renata Wassermann. On structured belief bases. In Hans Rott and Mary-Anne Williams, editors, *Frontiers in Belief Revision*. Kluwer, 2001. 37, 38
- [Wil97] Mary-Anne Williams. Implementing belief revision. In G. Antoniou, editor, *Nonmonotonic Reasoning*. MIT Press, 1997. 38

A New Approach to the Identification Problem

Carlos Brito

Cognitive Systems Laboratory
Computer Science Department, University of California
Los Angeles, CA 90024

Abstract. The Identification problem concerns the assessment of direct causal effects from a combination of: (i) non-experimental data, and (ii) qualitative domain knowledge. Domain knowledge is encoded in the form of a directed acyclic graph (DAG), in which all interactions are assumed linear, and some variables are presumed to be unobserved. Traditional approaches to this problem are based on algebraic manipulations of the equations defining the model. In this paper, we propose a new approach to the problem which takes advantage of the graphical representation of the model.

1 Introduction

This paper explores the feasibility of inferring linear cause-effect relationships from various combinations of data and theoretical assumptions. The assumptions have the form of linear equations and can be represented by an acyclic causal diagram, which contains both arrows and bi-directed arcs [Pearl, 1995, Pearl, 2000]. Intuitively, the arrows represent the potential existence of direct causal relationships between the corresponding variables, and the bi-directed arcs represent spurious correlations due to unmeasured common causes. Our task is to decide whether the assumptions represented in the diagram are sufficient for assessing the strength of causal effects from non-experimental data and, if sufficiency is proven, to express the target causal effect in terms of estimable quantities.

This decision problem has been tackled in the past half century, primarily by econometricians and social scientists, under the rubric “The Identification Problem” [Fisher, 1966] – it is still unsolved. Certain restricted classes of models are nevertheless known to be identifiable, and these are often assumed by social scientists as a matter of convenience or convention [Wright, 1960; Duncan, 1975]. [McDonald, 1997] characterizes a hierarchy of three such classes: (1) no bidirected arcs, (2) bidirected arcs restricted to root variables, and (3) bidirected arcs restricted to variables that are not connected through directed paths (see Figure 1). The structural equations in all three classes are regressional, and the parameters can therefore be estimated uniquely using OLS techniques (Bollen [1989, pp.104]).

Traditional approaches to the Identification problem are based on algebraic manipulation of the equations defining the model. Examples of these methods include the rank and order conditions [Fisher, 1966], and the well-known method

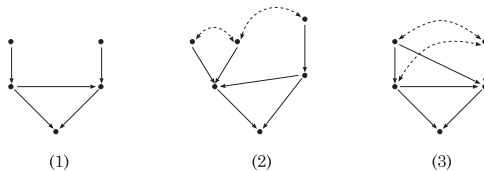


Fig. 1. McDonald’s regression hierarchy examples

of Instrumental Variables [Bowden and Turkington, 1984]. Most of these methods take advantage of conditional independence relations implied by the model to obtain the identification of specific causal effects. However, when the model is not rich in conditional independences, those methods are not much informative.

More recently, [Pearl, 1995] proposed a purely graphical condition for identification called Backdoor criterion. The backdoor criterion consists of a d-separation [Pearl, 1988] test applied to the causal diagram, and provides a sufficient condition for the identification of specific causal effects in the model.

In this paper we introduce a new approach to the identification problem based on Wright’s method of path coefficients [Wright, 1960]. Our method combines graphical information of the causal diagram with the manipulation of linear equations on the parameters of the model to obtain sufficient conditions for identification. The distinctive characteristic of our method is that it does not rely on the conditional independences implied by the model. As a consequence, the criteria obtained here can be successfully applied to prove the identification of models with few conditional independencies, while most existing methods would fail. This paper formalizes in a systematic approach ideas developed in [Brito and Pearl, 2002c] [Brito and Pearl, 2002b] [Brito and Pearl, 2002a].

The remainder of the paper is organized as follows. In section 2 we give a brief introduction to linear models and the identification problem, and review some useful graph definitions. In section 3 we describe our approach and present the main results of the paper. Section 4 provides some applications which illustrate the method. Finally, in section 5 we present our conclusions.

2 Background

2.1 Linear Models and Identification

An equation $Y = \beta X + e$ encodes two distinct assumptions: (1) the possible existence of (direct) causal influence of X on Y ; and, (2) the absence of causal influence on Y of any variable that does not appear on the right-hand side of the equation. The parameter β quantifies the (direct) causal effect of X on Y . That is, the equation claims that a unit increase in X would result in β units increase of Y , assuming that everything else remains the same. The variable e is called an “error” or “disturbance”; it represents unobserved background factors that the modeler decides to keep unexplained.

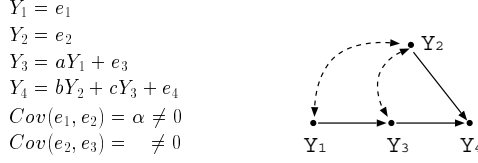


Fig. 2. A simple linear model and its causal diagram

A linear model for a set of random variables $\mathbf{Y} = \{Y_1, \dots, Y_N\}$ is formally defined by a set of equations of the form

$$Y_j = \sum_i c_{ji} Y_i + e_j, \quad j = 1, \dots, N$$

and a error variance/covariance matrix Ψ , i.e. $\Psi_{ij} = Cov(e_i, e_j)$. The error terms are assumed to have normal distribution with zero mean.

The equations and the pairs of error-terms (e_i, e_j) with non-zero correlation define the structure of the model. The model structure can be represented by a directed graph, called causal diagram, in which the set of nodes is defined by the variables $Y_1, \dots, Y_N$, and there is a directed edge from Y_i to Y_j if the coefficient of Y_i in the equation for Y_j is different from zero. Additionally, if error-terms e_i and e_j have non-zero correlation, we add a (dashed) bidirected edge between Y_i and Y_j . Figure 2 shows a model with the respective causal diagram.

The structural parameters of the model, denoted by θ , are the coefficients c_{ij} , and the non-zero entries of the error covariance matrix Ψ . In this work, we consider only recursive models, that is, $c_{ji} = 0$ for $i \geq j$.

2.2 Identification Almost Everywhere

Fixing the model structure and assigning values to the parameters θ , the model determines a unique covariance matrix Σ over the observed variables $\{Y_1, \dots, Y_N\}$, given by (see [Bollen, 1989], page 85)

$$\Sigma(\theta) = (I - C)^{-1} \Psi \left[(I - C)^{-1} \right]^T \quad (1)$$

where C is the matrix of coefficients c_{ji} .

Conversely, in the Identification problem, after fixing the structure of the model, one attempts to solve for θ in terms of the observed covariance Σ . This is not always possible. In some cases, no parametrization of the model could be compatible with a given Σ . In other cases, the structure of the model may permit several distinct solutions for the parameters. In these cases, the model is called *nonidentified*.

Sometimes, although the model is nonidentified, some parameters may be uniquely determined by the given assumptions and data. Whenever this is the case, the specific parameters are *identified*.

Finally, since the conditions we seek involve the structure of the model alone, and do not depend on the numerical values of parameters θ , we insist only on having identification almost everywhere, allowing few pathological exceptions. This concept is formalized as follows.

Let h denote the total number of parameters in a linear model. Then, each vector $\theta \in \Re^h$ defines a parametrization of the model. For each parametrization θ , the model generates a unique covariance matrix $\Sigma(\theta)$. Parameters θ are identified almost everywhere if

$$\Sigma(\theta) = \Sigma(\theta') \quad \text{implies} \quad \theta = \theta'$$

except when θ resides on a set of Lebesgue measure zero.

2.3 Graph Definitions

Definition 1. A path in a graph is a sequence of edges (directed or bidirected) such that each edge starts in the node ending the preceding edge. A directed path is a path composed only by directed edges, all oriented in the same direction. We say that node X is an ancestor of node Y if there is a directed path from X to Y . A path is said to be blocked if there is a node Z and a pair of consecutive edges in the path such that both edges are oriented toward Z (e.g., $\dots \rightarrow Z \leftarrow \dots$).

Let p be a path between nodes X and Y . We say that path p points to X if the edge of p incident to X is oriented toward it. Let Z be an intermediate variable of path p . We denote the subpath of p consisting of the edges between X and Z by $p[X \sim Z]$.

Definition 2. Define the depth of a node X in a DAG as the length (number of edges) of the longest directed path between X and any of its ancestors. Nodes with no ancestors have depth 0.

3 Our Approach

The approach presented here is based on Wright’s method of path coefficients [Wright, 1960]. We begin with a brief description of this method, and then show how to use graphical information of the structure of the model to obtain a powerful sufficient criteria for identification.

3.1 Wright’s Method of Path Coefficients

Wright’s method is based on the fundamental fact that the correlation of a pair of variables can be expressed as a polynomial on the parameters of the model. Formally, for variables X and Y in a recursive linear model, the correlation coefficient of X and Y , denoted σ_{XY} , is given by:

$$\sigma_{X,Y} = \sum_{\text{paths } p_l} T(p_l) \tag{2}$$

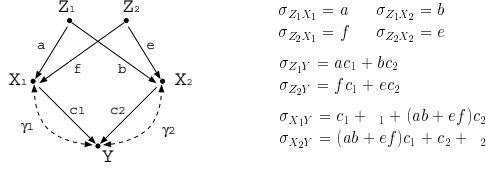


Fig. 3. Wright's equations

where term $T(p_i)$ represents the multiplication of the parameters of edges along path p_i , and the summation ranges over all unblocked paths between X and Y . For this equality to hold, the variables in the model must be standardized (variance equal to 1) and have zero mean. However, if this is not the case, a simple transformation can put the model in this form [Wright, 1960]. We refer to Eq.(2) as Wright's equation for X and Y . Figure 3 shows a simple model with the corresponding equations for some pairs of variables.

Wright's method for identification consists in forming Eq.(2) for each pair of variables in the model, and solving for the parameters in terms of the correlations. Whenever there is a unique solution for a parameter λ , this parameter is identified.

To illustrate the method, let us study the identification of the parameters in the model of Figure 3. Note that parameters a, b, e and f are directly obtained from the first equations. From the equations for $\rho_{Z_1,Y}$ and $\rho_{Z_2,Y}$ it follows that parameters c_1 and c_2 are identified if and only if $\text{Det} \begin{bmatrix} a & e \\ b & f \end{bmatrix} \neq 0$.

The set of Wright's equations summarizes all the statistical information encoded in the model. Therefore, any question about identification can be decided by studying the solutions for this system of equations. However, although this could be feasible for small models, it may become a very complex task in general. To the best of our knowledge, the work presented in this paper is the first attempt at a method to solve such equations in a systematic way.

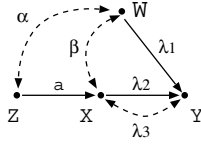
In the following, we show how to use the graphical representation of the model to partition the set of Wright's equations, such that identification results can be obtained by studying simple systems of linear equations.

3.2 Basic Linear Equations

Fix a variable Y in the model, and let $\text{depth}(Y) = k$. Let $\mathbf{X} = \{X_1, \dots, X_n\}$ be the set of variables at depth smaller than k which are connected to Y by an edge (directed, bidirected, or both). Define the set of edges with an arrowhead at Y as:

$$\text{Inc}(Y) = \{(X_i, Y) : X_i \in \mathbf{X}\}$$

Note that for some $X_i \in \mathbf{X}$ there may be more than one edge between X_i and Y (one directed and one bidirected). Thus, in general, $|\text{Inc}(Y)| \geq |\mathbf{X}|$. Let $\lambda_1, \dots, \lambda_m$, $m \geq n$, denote the parameters of the edges in $\text{Inc}(Y)$.



$$\begin{aligned}\sigma_{ZY} &= \alpha\lambda_1 + a\lambda_2 \\ \sigma_{XY} &= (a\alpha + \lambda_3)\lambda_1 + \lambda_2 + \lambda_3 \\ \sigma_{WY} &= \lambda_1 + (a\alpha + \lambda_3)\lambda_2\end{aligned}$$

Fig. 4. Wright's equations

Let Z be any variable at depth smaller than k . Then, Wright's equation for the pair (Z, Y) , is given by

$$\sigma_{Z,Y} = \sum_{\text{paths } p_l} T(p_l) \quad (3)$$

where each term $T(p_l)$ corresponds to an unblocked path between Z and Y . Next lemma proves an important property of such paths:

Lemma 1. *Let Y be a variable in a recursive model, and let Z be such that $\text{depth}(Z) < \text{depth}(Y)$. Then, any unblocked path between Z and Y must include exactly one edge from $\text{Inc}(Y)$.*

Proof. It is easy to see that no valid path between Z and Y can include more than one edge from $\text{Inc}(Y)$. Now, assume that p is an unblocked path between Z and Y , which does not contain any edge from $\text{Inc}(Y)$. Let W be the variable adjacent to Y in path p . Then, $\text{depth}(W) \geq \text{depth}(Y)$, and edge (W, Y) must point to W . Since p is an unblocked path, it follows that subpath $p[W \sim Z]$ is a directed path from W to Z . But this implies that $\text{depth}(Z) \geq \text{depth}(Y)$, which is a contradiction. $\square$

Lemma 1 allows to write Eq. (3) as

$$\sigma_{Z,Y} = \sum_{j=1}^m a_j \cdot \lambda_j \quad (4)$$

Thus, the correlation between Z and Y can be expressed as a linear function of the parameters $\lambda_1, \dots, \lambda_m$, with no constant term. Figure 4 shows an example of those equations for a simple model.

3.3 System of Equations and Linear Independence

Given the result in the previous section, it seems natural to consider the systems Φ_Y composed by the set of Wright's equations for each pair (Z, Y) , where $\text{depth}(Z) < \text{depth}(Y)$. From Eq. (4), we get that each Φ_Y is a system of linear equations on the parameters of the edges in $\text{Inc}(Y)$.

As before, for a fixed Y let $\lambda_1, \dots, \lambda_m$ be the parameters of the edges in $\text{Inc}(Y)$. Our goal is to obtain the identification of parameters $\lambda_1, \dots, \lambda_m$ by

solving the equations of Φ_Y . In the following, we study the conditions under which this is possible.

Clearly, a necessary condition is that Φ_Y must contain m linearly independent equations with respect to the λ_i 's. In general, Φ_Y may have more than m equations, but some of them could be linear combinations of others. This motivates the following definition:

Definition 3. *A set of variables $\mathbf{Z} = \{Z_1, \dots, Z_n\}$ is said to be an Auxiliary set with respect to Y if and only if:*

- (i) $\text{depth}(Z_i) < \text{depth}(Y)$, $i = 1, \dots, n$;
- (ii) *The subsystem $\Phi_{Y,\mathbf{Z}}$ consisting only of equations for the pairs (Z_i, Y) , with $Z_i \in \mathbf{Z}$, is linearly independent with respect to the λ_i 's.*

Next theorem states the first main result of this paper, providing a sufficient graphical condition for a set of variables $\mathbf{Z} = \{Z_1, \dots, Z_n\}$ to be an Auxiliary set with respect to Y . A sketch of the proof is given in the appendix.

Theorem 1. *Let $\mathbf{Z} = \{Z_1, \dots, Z_k\}$ be a set of variables such that $\text{depth}(Z_i) < \text{depth}(Y)$, $i = 1, \dots, n$. If there exist paths $p_1, \dots, p_n$ satisfying the conditions:*

- a) p_i is an unblocked path between Z_i and Y ;
- b) for $i < j$, variable Z_j does not appear in path p_i ;
- c) if paths p_i and p_j have a common variable V , different from Y , then both $p_i[V \sim Y]$ and $p_j[Z_j \sim V]$ point to V .

then $\mathbf{Z}$ is an Auxiliary set with respect to Y .

3.4 Identification

Unfortunately, the existence of an Auxiliary set $\mathbf{Z} = \{Z_1, \dots, Z_n\}$ with respect to Y , with $|\mathbf{Z}| = |\text{Inc}(Y)|$, does not guarantee that the parameters of the edges in $\text{Inc}(Y)$ are identified. This fact only implies that $\Phi_{Y,\mathbf{Z}}$ can be solved uniquely for the λ_i 's in terms of the coefficients of the equations in $\Phi_{Y,\mathbf{Z}}$. But these coefficients are polynomials on some other parameters of the model. If some of these parameters are non-identified, then the λ_i 's may not be identified, even if Φ_Y has enough independent equations.

Next theorem, however, provides a sufficient condition for the identification of the entire model. The proof easily follows by induction.

Theorem 2. *If for every variable Y in the model we can find an Auxiliary set $\mathbf{Z}$ such that $|\mathbf{Z}| = |\text{Inc}(Y)|$, then the model is completely identified.*

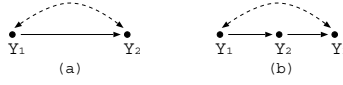


Fig. 5. (a) a "bow-pattern", and (b) a bow-free model

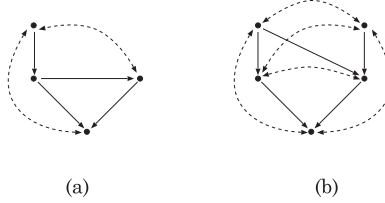


Fig. 6. More examples of bow-free models

4 Application

4.1 Bow-Free Models

A model is said to be bow-free if its associated causal diagram is free of any bow-pattern [Pearl, 2000] (see Figure 5). More precisely, in a bow-free model variables standing in direct causal relationship (i.e., variables connected by directed edges in the causal diagram) do not have correlated errors; no restriction is imposed on errors associated with indirect causes. Clearly, the class of bow-free models includes all the models in the regressional hierarchy of McDonald [McDonald, 1997].

Corollary 1. *Every bow-free model is completely identified.*

Proof: For an arbitrary variable Y in a bow-free model, let $\mathbf{X} = \{X_1, \dots, X_n\}$ be the set of variables at depth smaller than $\text{depth}(Y)$ which are connected to Y by an edge. Note that in a bow-free model there is at most one edge between each pair of variables. Then, it follows that $|\mathbf{X}| = |\text{Inc}(Y)|$.

Now, for $i = 1, \dots, n$, let p_i denote the unblocked path between X_i and Y consisting only of the edge $(X_i, Y) \in \text{Inc}(Y)$. Then, it is easy to see that paths $p_1, \dots, p_n$, witness that $\mathbf{X}$ is an Auxiliary set with respect to Y .

Therefore, every bow-free model is completely identified. $\square$

Figure 6 shows more examples of bow-free models.

4.2 Parent Condition

Here, we consider general recursive models. Let Y be a variable in the model, and again let $\mathbf{X} = \{X_1, \dots, X_n\}$ be the set of variables at depth smaller than $\text{depth}(Y)$ which are connected to Y by an edge. Define $\text{Bow}(Y)$ to be the subset of $\mathbf{X}$ consisting of variables which are connected to Y by both a directed and a bidirected edge. It is easy to see that $|\text{Inc}(Y)| = |\mathbf{X}| + |\text{Bow}(Y)|$.

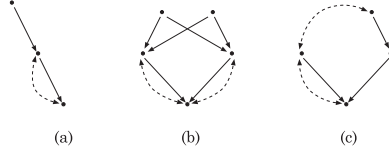


Fig. 7. Parent condition examples

We say that variable Y satisfies the parent condition if for each variable $X_{i_j} \in \text{Bow}(Y)$ we can find a unique Z_j such that:

1. $Z_j \notin \mathbf{X}$;
2. $\text{depth}(Z_j) < \text{depth}(Y)$;
3. there is an edge between Z_j and X_{i_j} oriented toward X_{i_j} .

Corollary 2. *If the parent condition holds for every variable in the model, then the model is completely identified.*

Proof: Let Y be an arbitrary variable, and let $\mathbf{Z} = \{Z_1, \dots, Z_l\}$ witness that the Parent condition holds for Y . Consider the set $\mathbf{W} = \mathbf{X} \cup \mathbf{Z}$. For each variable $W_i \in \mathbf{W}$, let path p_i be defined as follows:

- i) if $W_i \in \mathbf{X} - \text{Bow}(Y)$, then p_i consists of the edge connecting W_i and Y ;
- ii) if $W_i \in \text{Bow}(Y)$, then p_i consists of the bidirected edge between W_i and Y ;
- iii) if $W_i \in \mathbf{Z}$, then let X_{i_j} be the associated variable in $\text{Bow}(Y)$. Then, path p_i is given by the concatenation of the edge between W_i and X_{i_j} and the directed edge between X_{i_j} and Y .

It follows that paths $p_1, \dots, p_{n+l}$, witness that $\mathbf{W}$ is an Auxiliary set with respect to Y . Hence the result follows. $\square$

Figure 7 shows some examples of such models.

5 Discussion and Conclusion

In this paper, we have introduced a new approach to the Identification problem in linear models. Existing methods for the problem strongly rely on the conditional independencies implied by the model. Since our method does not make direct use of conditional independence, it may be successfully applied to models which are not rich in such features. For example, for the model in Figure (7b), methods like Instrumental Variables and Back-door criterion fail to prove identification, while our Parent condition is successful.

The technique presented here can prove the identification of a large class of models. It does not solve the Identification problem, though. Figure 8 shows a model in which our method fails. In this example, there is no Auxiliary set of size 2 for variable Y . However, manipulating the entire set of Wright's equations it is possible to show that every parameter in the model is identified. Currently, we are investigating the fundamental features of such models to obtain a complete method for Identification in recursive models.

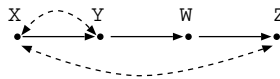


Fig. 8. A counter example to our method

Acknowledgment

This research was supported in parts by grants from ONR (MURI) and CNPq fellowship proc. 200201/99-9.

References

- [Bollen, 1989] K. A. Bollen. *Structural Equations with Latent Variables*. John Wiley, New York, 1989. 43
- [Bowden and Turkington, 1984] Roger J. Bowden and Darrell A. Turkington. *Instrumental Variables*. Cambridge University Press, Cambridge, MA, 1984. 42
- [Brito and Pearl, 2002a] C. Brito and J. Pearl. Generalized instrumental variables. In *submitted to Uncertainty in Artificial Intelligence*, 2002. 42
- [Brito and Pearl, 2002b] C. Brito and J. Pearl. A graphical criterion for the identification of causal effects in linear models. In *Proceedings of the AAAI Conference*, Edmonton, 2002. 42
- [Brito and Pearl, 2002c] C. Brito and J. Pearl. A new identification condition for recursive models with correlated errors. *To appear in Structural Equation Modelling*, 2002. 42
- [Duncan, 1975] O. D. Duncan. *Introduction to Structural Equation Models*. Academic Press, 1975.
- [Fisher, 1966] F. M. Fisher. *The Identification Problem in Econometrics*. McGraw-Hill, 1966. 41
- [McDonald, 1997] R. McDonald. Haldane’s lungs: A case study in path analysis. *Multiv. Behav. Res.*, pages 1–38, 1997. 41, 48
- [Okamoto, 1973] M. Okamoto. Distinctness of the eigenvalues of a quadratic form in a multivariate sample. *Annals of Stat.*, 1(4):763–765, July 1973. 51
- [Pearl, 1988] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA, 1988. 42
- [Pearl, 1995] J. Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–710, Dec 1995. 41, 42
- [Pearl, 2000] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge Press, New York, 2000. 41, 48
- [Wright, 1960] S. Wright. Path coefficients and path regressions: alternative or complementary concepts? *Biometrics*, pages 189–202, June 1960. 42, 44, 45

Appendix

Proof Sketch of Theorem 1:

Lemma 1 and condition (a) give that each path p_i contains exactly one edge from $Inc(Y)$. Condition (c) implies that, for $i \neq j$, paths p_i and p_j cannot share the same edge from $Inc(Y)$. Thus, w.l.o.g., we may assume that path p_i contains the edge from $Inc(Y)$ with parameter λ_i , $i = 1, \dots, k$.

The system of equations $\Phi_{Y,Z}$ can be expressed in matrix form as $Q\Lambda = \Sigma$, where $\Lambda = [\lambda_1 \ \lambda_2 \ \dots \ \lambda_m]'$ and $\Sigma = [\sigma_{Z_1 Y} \ \sigma_{Z_2 Y} \ \dots \ \sigma_{Z_k Y}]'$.

Let Q' be the submatrix corresponding to the first k columns of Q . It follows that $\det(Q') \neq 0$ implies that the equations in $\Phi_{Y,Z}$ are linearly independent with respect to the λ_i 's.

Now, the determinant of Q' is defined as the weighted sum, ranging over all permutations π of $\langle 1, 2, \dots, k \rangle$, of the product of entries of Q' selected by π , where the weights are either 1 or (-1) depending on the parity of each permutation.

Each diagonal entry q_{ii} of Q' contains the term given by the product of the parameters along the path p_i except for λ_i , which can be expressed as $T(p_i)/\lambda_i$. Then, it is easy to see that the term

$$T^* = \prod_{i=1}^k T(p_i)/\lambda_i$$

appears in the product of permutation $\pi = \langle 1, \dots, k \rangle$, which selects all diagonal entries of Q' .

The following two facts imply that T^* appears only once in the product of permutation $\pi = \langle 1, \dots, k \rangle$, and that it does not appear in the product of any other permutation π' :

1. there is no unblocked path p between Z_i and Y , different from p_i , including the edge with parameter λ_i , such that p is composed only by edges in $p_1, \dots, p_i$;
2. for $i > j$, there is no unblocked path p between Z_i and Y including the edge with parameter λ_j , such that p is composed only by edges in $p_1, \dots, p_i$.

Hence, term T^* is not cancelled out, and $\det(Q')$ is a non-trivial polynomial on the coefficients of Q' . Thus, $\det(Q')$ only vanishes on the zeros of a polynomial. However, [Okamoto, 1973] shows that this set has Lebesgue measure zero. $\square$

Towards Default Reasoning through MAX-SAT

Berilhes Borges Garcia and Samuel M. Brasil, Jr.

Universidade Federal do Espirito Santo - UFES

Department of Computer Science

Av. Fernando Ferrari, s/n - CT VII

29.060-970 - Vitoria - ES - Brazil

`berilhes@inf.ufes.br`

`smbrasil@tj.es.gov.br`

Abstract. We introduce a translation of a conditional logic semantics to a mathematical programming problem. A model of 0-1 programming is used to compute the logical consequences of a conditional knowledge base, according to a chosen default theory semantics. The key to understanding this model of mathematical programming is to regard the task of the entailment of plausible conclusions as isomorphic to an instance of weighted MAX-SAT problem. Hence, we describe the use of combinatorial optimization algorithms in the task of defeasible reasoning over conditional knowledge bases.

1 Introduction

Non-monotonic reasoning is a form of dealing with uncertainty usually found in common sense, and it is concerned with drawing conclusions from a set of rules which may have exceptions, and from a set of facts which is often incomplete. Researchers in Artificial Intelligence usually represent this type of common sense reasoning by means of a conditional knowledge base, to stay "closer" to standard deductive logics. A conditional knowledge base is a set of strict rules in classical logic and a set of defeasible rules. The former represents statements that must be satisfied, while the latter is used for expressing normal situation without inhibiting the existence of exceptions.

The properties of model-preference nonmonotonic logics have been discussed at length in the literature, and a number of semantics were presented. However, there exists few implementations for conditional logics, to our knowledge. In this paper we aim at showing how a default reasoning semantics can be implemented in a mathematical programming environment. For this task, we use a translation to a weighted MAX-SAT problem to compute the logical consequences from a conditional knowledge base, according to Pearl's System Z semantic ([10]). To understand this model of 0-1 programming one should regard the task of default reasoning as isomorphic to the task of solving the problem of weighted MAX-SAT.

There have been a few works on proving theorems in classical propositional logic using mathematical programming techniques, since Boole's seminal work (see [6] and [3]). Later, several researchers ([8], [7], [2]) have improved this approach. In this paper we demonstrated how mathematical programming can be used to implement a default theory.

The plan of this paper can be summarized as follows. Section (2) presents a briefly description of the Pearl's System Z [10]. The next section describes one of the possible translations for the satisfiability problem using an integer programming model. Then, we show how to determine the logical consequences of a conditional knowledge base in mathematical programming. Finally, in Section (5) we summarize the main results of the paper. The proofs are at <http://www.inf.ufes.br/~berilhes>.

2 The System Z Semantics

We assume that $\mathcal{P}$ is the propositional alphabet of a finite language $\mathcal{L}$, enriched with two symbols $\top$ and $\perp$, respectively, logical truthfulness and logical falsity. Propositional formulas are denoted by Greek letters $\alpha, \psi, \phi, \dots$ and built inductively by using the propositional letters existing in $\mathcal{P}$, and logical connectives. A Logical entailment is represented by $\models$. An interpretation of $\mathcal{L}$ is a function w of $\mathcal{P}$ to Boolean values $\{T, F\}$ and this function can be extended to propositions built from the alphabet $\mathcal{P}$ in the usual way, such that $w(\alpha \wedge \beta) = T$ iff $w(\alpha) = T$ and $w(\beta) = T$, etc. $\mathcal{W}$ represents the set of all possible interpretations. A model for a set of propositions H is an interpretation w such that $w(\alpha) = T$ for all $\alpha \in H$.

We represent the conditional knowledge base Δ by a pair (K_D, L) , where L is a finite set of strict conditionals, written as: $\alpha_i \Rightarrow \beta_i$, and K_D is a finite set of defaults, written as: $\alpha_i \rightsquigarrow \beta_i$. Both $\Rightarrow$ and $\rightsquigarrow$ are meta-connectives, where $\Rightarrow$ means definitely and $\rightsquigarrow$ means normally / typically, which can occur only as the main connective. We will refer to a rule on the conditional knowledge base Δ that can be either strict or defeasible as a *conditional* sentence δ . The conditional sentence with antecedent α_i and consequent β_i has the *material counterpart* by replacing the connective by the material implication¹ connective $\supset$, denoted by $\alpha_i \supset \beta_i$, and the material counterpart of Δ will be represented by Δ^* . An interpretation w is a model of δ , or satisfies the conditional δ , denoted by $w \models \delta$, iff $w \models \alpha \supset \beta$, i.e., iff w satisfies the material counterpart of δ . In the same way, w is a model of a set Δ of strict and defeasible rules, or w satisfies a set Δ , denoted by $w \models \Delta$, iff w satisfies each member of Δ . Furthermore, w falsifies a conditional δ iff $w \models \alpha \wedge \neg\beta$.

¹ In this paper we distinguish $\alpha \Rightarrow \beta$ and $\alpha \supset \beta$, where the former denotes generic knowledge and the later an item of evidence. For a complete discussion, see [5].

Example 1. Consider the following conditional knowledge base Δ , regarding a certain domain:

$$\Delta = \left\{ \begin{array}{l} \delta_1 : a \rightsquigarrow \neg f \\ \delta_2 : a \rightsquigarrow \neg fe \\ \delta_3 : b \rightsquigarrow f \\ \delta_4 : b \rightsquigarrow fe \\ \delta_5 : p \rightsquigarrow \neg f \\ \delta_6 : p \rightsquigarrow \neg fe \end{array} \right\} \cup \left\{ \begin{array}{l} \delta_7 : b \Rightarrow a \\ \delta_8 : p \Rightarrow b \end{array} \right\} \quad (1)$$

Rules δ_i represent the following information: (δ_1) animals (a) typically neither fly ($\neg f$), (δ_2) nor have feathers ($\neg fe$), (δ_3) birds (b) normally fly (f), (δ_4) and typically have feathers (fe), (δ_5) penguins (p) normally do not fly ($\neg f$) and (δ_6) typically have not feathers ($\neg fe$), (δ_7) birds (b) are definitely animals (a) and (δ_8) penguins (p) are definitely birds (b).

We use the specificity relations among defaults on a conditional knowledge base Δ to establish the preference relations between the possible interpretations. The determination of the specificity relations is made using system Z [10], which defines an unique partition of the set of defaults K_D , into ordered sets of mutually exclusive defaults $K_{D0}, K_{D1}, \dots, K_{Dn}$. The main concept used to determine this partitioning is the notion of tolerance. A default is tolerated by Δ if the antecedent and the consequent of this default are not in direct conflict with any inference sanctioned by Δ^* , where Δ^* is the material counterpart of Δ .

Definition 1 (Tolerance [10]). A conditional δ_i with antecedent α and consequent β is tolerated by a conditional knowledge base Δ iff there exists a w such that

$$w \models \{\alpha \wedge \beta\} \cup \Delta^*.$$

Using tolerance, Goldszmidt and Pearl [5] developed a syntactical test for consistency that generates the partition of a conditional knowledge base and, hence, the ranking among defaults.

Definition 2 (p -consistency of K_D [5]). K_D is p -consistent iff we can build an ordered partition of $K_D = (K_{D0}, \dots, K_{Dn})$ where:

1. for all $1 \leq i \leq n$, each $\delta \in K_{Di}$ is tolerated by $L \cup (K_D - \bigcup\{K_{Dj} | 0 \leq j < i\})$.
2. every conditional in L is tolerated by L .

The partition of K_D into $K_{D0}, \dots, K_{Dn}$ has the following property: Every default belonging to K_{Di} is tolerated by $L \cup \bigcup_{j=i}^n K_{Dj}$, where n is the number of the subsets in the partition.

It is important to note that there is no partition of the strict set L . A strict conditional can not be overruled but only defaults, what excludes the set of strict conditionals L from the partitioning. Now we shall define the strength of a conditional.

Definition 3 (Strenght of a Conditional). Let $Z(\delta_m)$ be the strength of the conditional δ_m , then

$$Z(\delta_m) = \begin{cases} i & \text{iff } \delta_m \in K_{Di} \\ \infty & \text{iff } \delta_m \in L \end{cases} \quad (2)$$

Definition 4 (Order relation among conditional defaults). A default δ_i has greater strength than δ_j iff $Z(\delta_j) < Z(\delta_i)$.

Example 2. The example (1) generates the following partition (2) of K_D :

$$K_{D0} = \{\delta_1, \delta_2\}; K_{D1} = \{\delta_3, \delta_4\}; K_{D2} = \{\delta_5, \delta_6\}$$

The conditionals δ_7 and δ_8 are not in the partitioning, because they are strict rules. A conditional knowledge base that falsifies a strict rule is inconsistent.

Example 3. $Z(\delta_1) = Z(\delta_2) = 0$, $Z(\delta_3) = Z(\delta_4) = 1$ and $Z(\delta_5) = Z(\delta_6) = 2$.

Definition 5 (Ranking). The ranking function $k(w)$ on interpretations induced by Z is defined as follows

$$k(w) = \begin{cases} \infty & \text{iff } w \not\models L \\ 0 & \text{iff } w \models L \cup K_D \\ 1 + \max_{\delta_m \in K_{Di}: w \not\models \delta_m} Z(\delta_m) & \text{otherwise} \end{cases} \quad (3)$$

Definition 6 (Minimal Interpretation). The interpretation w_p is minimal with respect to Z -ordering iff there exists no w_q such that $k(w_q) < k(w_p)$.

The conclusions entailed by Δ for any ranking k form a *consequence relation*.

Definition 7 (Consequence relation). A ranking k induces a consequence relation $\vdash_k$, denoted by $\Delta \vdash_k \alpha \rightsquigarrow \beta$, iff $k(\alpha \wedge \beta) < k(\alpha \wedge \neg \beta)$.

Thus, since Δ permits several ranking functions, the entailment should take into account the consequence relations induced by k wrt Δ .

Definition 8 (Z-entailment). A default $\delta_m : \alpha \rightsquigarrow \beta$ is Z -entailed by Δ , $\Delta \vdash_z \alpha \rightsquigarrow \beta$, iff $\Delta \vdash_k \alpha \rightsquigarrow \beta$ is in the consequence relation $\vdash_k$.

Now, we shall give a brief introduction to the weighted MAX-SAT problem.

3 Maximum Satisfiability in Mathematical Programming

The satisfiability problem (SAT) is a propositional logic problem, and its goal is to determine an assignment of truth values to propositional letters that makes a given conjunctive normal formula (CNF) satisfied or show that none exists; in other words, the goal of SAT is to find a true assignment that satisfies a given conjunctive normal form $\phi = c_1 \wedge c_2 \wedge \dots \wedge c_n$, where each c_i is a clause.

MAX-SAT problem is closely related to the SAT problem, and informally is defined as: Given a collection of clauses, we seek a true assignment that minimizes the number of falsified clauses. The *weighted MAX-SAT* problem is an instance of MAX-SAT that assigns a weight to each clause, and seeks a true assignment that minimizes the sum of weights of the unsatisfied clauses. Both problems (MAX-SAT and weighted MAX-SAT) are NP-hard problems.

Williams and Jeroslow ([12] and [8]) have shown some existing connections among classical propositional logic and integer programming. We shall briefly summarize one of the possible translations of a CNF into a set of linear constraints, to implement a nonmonotonic semantics using integer programming techniques.

A literal is a propositional letter a_i or the negation of a propositional letter $\neg a_i$. A clause is a disjunction of literals. A clause is *satisfied* by an interpretation iff at least one of the literals present in the clause has “true” value. A formula ϕ of the propositional language $\mathcal{L}$ is said to be in “conjunctive normal form” (CNF) if ϕ is the conjunction of clauses. Each formula ϕ has equivalent CNF formulas². The function $CNF(\alpha)$ returns a CNF formula that is equivalent to α . We assume that the function $CNF(.)$ maps α to only one CNF formula, although α may have several equivalent CNF formulas. A formula ϕ in CNF is said *satisfied* for an interpretation I iff all clauses in ϕ are satisfied. We denote by $\mathcal{H}_\phi$ the Herbrand base of formula CNF ϕ , i.e., the set of all literals existents on ϕ .

Definition 9. (Binary Variables) x is a binary variable if it can only has the integer values $\{0, 1\}$. Each binary variable is labeled with the literal which it is related.

Definition 10. (Binary Representation of a Formula) $\mathcal{H}_\phi$ is the base of Herbrand associated to the formula ϕ . $B(\mathcal{H}_\phi)$ represents the set of binary variables associated to ϕ and is formed by $P(\mathcal{H}_\phi) \cup N(\mathcal{H}_\phi)$, such that for each $a_i \in \mathcal{H}_\phi$ if a_i is a positive literal then x_{a_i} belongs to $P(\mathcal{H}_\phi)$, otherwise if a_i is a negative literal then x_{a_i} belongs to $N(\mathcal{H}_\phi)$.

Definition 11. (Binary Attribution) $B(\mathcal{H}_\phi) = \{x_{a_1}, \dots, x_{a_m}\}$ is the binary representation of $\mathcal{H}_\phi$. An attribution of binary variables is a mapping $s : B(\mathcal{H}_\phi) \rightarrow \{0, 1\}^m$.

A binary variable x_a represents the truth value of a .

We assume through all this paper that the language is finite; for this reason we can assume that the attribution of binary variables is well defined with respect to any formula ϕ , i.e., with respect to the binary representation of a formula ϕ .

Definition 12. (Linear Inequality Generated from a clause) Assume that c_i is a clause, in a propositional language $\mathcal{L}$, and that $B(\mathcal{H}_{c_i}) = P(\mathcal{H}_{c_i}) \cup$

² Two formulas α and β are equivalent iff $\alpha \models \beta$ and $\beta \models \alpha$.

$N(\mathcal{H}_{c_i})$ represents the set of binary variables associated to c_i (definition 10). $\lambda(c_i)$ is the linear inequality generated from $B(\mathcal{H}_{c_i})$, and it is defined as:

$$\sum_{x_{a_k} \in P(\mathcal{H}_{c_i})} x_{a_k} + \sum_{x_{a_k} \in N(\mathcal{H}_{c_i})} (1 - x_{a_k}) \geq 1 \quad (4)$$

We can extend the definition of linear inequality generated from a clause to a system of inequalities generated from a conjunctive normal formula ϕ .

Definition 13. ϕ is a CNF, $\mathcal{C}_\phi$ is the set of clauses in ϕ ; then the system of linear inequalities generated by ϕ , $sd(\phi)$, is:

$$sd(\phi) = \{\lambda(c_i) : \text{for all } c_i \in \mathcal{C}_\phi\} \quad (5)$$

Example 4. Consider the following conjunctive normal formula:

$$\phi = \underbrace{(a \vee b)}_{c_1} \wedge \underbrace{(\neg a \vee c \vee b)}_{c_2} \quad (6)$$

The set of clauses in ϕ is $\mathcal{C}_\phi = \{c_1, c_2\}$. The bases of Herbrand of the clauses c_1 and c_2 are, respectively, $\{a, b\}$ and $\{\neg a, c, b\}$. Thus $B(\mathcal{H}_{c_1}) = \{x_a, x_b\}$ and $B(\mathcal{H}_{c_2}) = N(\mathcal{H}_{c_2}) \cup P(\mathcal{H}_{c_2})$, such that $N(\mathcal{H}_{c_2}) = \{x_a\}$ and $P(\mathcal{H}_{c_2}) = \{x_c, x_b\}$. Therefore, the inequality system generated by ϕ , $sd(\phi)$, is:

$$\left\{ \begin{array}{l} \lambda(c_1) : x_a + x_b \geq 1 \\ \lambda(c_2) : x_c + x_b - x_a \geq 0 \end{array} \right\} \quad (7)$$

Definition 14. A binary attribution s satisfies a inequality system $sd(\phi)$ iff s does not falsify none constraint in $sd(\phi)$.

As previously noted, the propositional satisfiability problem for a CNF ϕ consists on finding an attribution of truth values to literals that are in ϕ which satisfies each clause in ϕ or showing that this attribution does not exist. Therefore, the propositional satisfiability problem consisting of finding a binary attribution that satisfies the inequality set $sd(\phi)$ [3].

MAX-SAT problem is primarily concerned with finding an attribution of truth values to literals in ϕ that falsifies the smaller set of inequalities $\lambda(c_i)$. The weighted MAX-SAT problem assigns a weight to each inequality $\lambda(c_i)$, i.e., a weight to each clause, and seeks an assignment that minimizes the sum of the weight of falsified clauses. So, to formulate the weighted MAX-SAT problem as an integer program we first define that $\lambda(c_i)$ is equal to the following inequality:

$$\sum_{x_{a_k} \in P(\mathcal{H}_{c_i})} x_{a_k} + \sum_{x_{a_k} \in N(\mathcal{H}_{c_i})} (1 - x_{a_k}) + t_i \geq 1 \quad (8)$$

Where t_i is an artificial binary variable created to represent each clause c_i that forms a CNF ϕ . Additionally w_i represents the weight associated to clause c_i .

So, the weighted MAX-SAT problem can be formulated as the integer program:

$$\begin{aligned} \text{Min} \quad & \sum_{\forall c_i \in \mathcal{C}_\phi} w_i t_i \\ \text{Subject to: } & \lambda(c_i) \text{ for all } c_i \in \mathcal{C}_\phi \end{aligned} \quad (9)$$

In the next section we shall demonstrate how a MAX-SAT problem can be used to compute the logical entailment of a conditional knowledge base.

4 Z-Entailment through MAX-SAT

In this section, a translation ζ of a conditional knowledge base Δ to a weighted MAX-SAT problem is proposed, showing the interrelationship among the solutions for this problem and the set of Z -entailed consequences of Δ .

Informally, we can explain the translation proposed in this paper by understanding each conditional as a CNF, where each default sentence has a related specific weight (cost). The strict sentences does not have a weight (cost), because they can not be falsified. The objective function is to minimize the total cost of the sum of the not satisfied CNF formulas. Note that according to the Z -entailment, falsifying a default δ_m that belongs to the partition K_{Di} means that all defaults belonging to partitions K_{Dj} , for $j \leq i$, that is, all defaults at least so normal as δ_m , are not considered in the determination of the logical consequences of the conditional knowledge base.

By δ_m^* we represent the material counterpart of the conditional $\delta_m \in \Delta$. Moreover, by $CNF(\delta_m^*)$ we denote the CNF formula equivalent to δ_m^* . In addition, $B(\mathcal{H}_{ci}) = P(\mathcal{H}_{ci}) \cup N(\mathcal{H}_{ci})$, definition 10, represents the set of binary variables associated to the clause $c_i \in CNF(\delta_m^*)$.

Definition 15. [Artificial Variables of a Partition] t_i is a new binary variable associated to the partition K_{Di} , that is, if a conditional knowledge base Δ has m partitions then there exists m variables t_i 's, one for each partition.

If the conditional δ_m is a defeasible rule $\alpha \rightsquigarrow \beta \in K_{Di}$, then each clause $c_i \in CNF(\delta_m^*)$ will yield the following linear inequality:

$$\lambda(c_m) : \sum_{x_{a_k} \in P(\mathcal{H}_{c_m})} x_{a_k} + \sum_{x_{a_k} \in N(\mathcal{H}_{c_m})} (1 - x_{a_k}) + t_i \geq 1 \quad (10)$$

If the conditional δ_m is a strict rule $\alpha \Rightarrow \beta$, then each clause $c_i \in CNF(\delta_m^*)$ will generate the following linear inequality:

$$\lambda'(c_m) : \sum_{x_{a_k} \in P(\mathcal{H}_{c_m})} x_{a_k} + \sum_{x_{a_k} \in N(\mathcal{H}_{c_m})} (1 - x_{a_k}) \geq 1 \quad (11)$$

The main difference among defaults and strict conditionals lies on the fact that we do not have a weight (cost) associated with strict rules, so we do not

use the new binary variable t_m in the inequality linear system attributed to defeasible sentences.

Therefore, the following linear inequalities system ($sd(\delta_m)$) will be generated:

$$sd(\delta_m) = \begin{cases} \lambda(c_i) : \forall c_i \in CNF(\delta_m^*) \text{ iff } \delta_m \in K_D \\ \lambda'(c_i) : \forall c_i \in CNF(\delta_m^*) \text{ iff } \delta_m \in L \end{cases} \quad (12)$$

A ζ -translation of Δ is defined as the union of the linear inequalities system generated by the translation of each conditional sentence belonging to Δ .

Definition 16. (ζ -translation of Δ) The ζ -translation of Δ will be equal to the following linear inequalities system, $sd(\Delta)$:

$$sd(\Delta) = \{sd(\delta_i) \mid \forall \delta_i \in \Delta\} \quad (13)$$

If a default from a partition K_{Di} is falsified, then Z -entailment demands that all defaults from the partition K_{Di} and from the partitions of lower levels do not be considered in the determination of the logical consequences of a conditional knowledge base. To certify this feature we inserted the constraints $t_i \geq t_{i+1}$, for $i = 0, \dots, m-1$. Hence, the system of inequalities generated from Δ given by example (1) was enlarged by the following constraints $pd(\Delta) = \{t_0 \geq t_1; t_1 \geq t_2\}$.

Example 5. The ζ -translation of Δ given by example (1) is equal to the following system of linear inequalities $sd(\Delta)$:

$$sd(\Delta) = \left\{ \begin{array}{l} t_0 - x_f - x_a \geq -1 \\ t_0 - x_{fe} - x_a \geq -1 \\ t_1 + x_f - x_b \geq 0 \\ t_1 + x_{fe} - x_b \geq 0 \\ t_2 - x_f - x_p \geq -1 \\ t_2 - x_{fe} - x_p \geq -1 \end{array} \right\} \cup \left\{ \begin{array}{l} x_a - x_b \geq 0 \\ x_b - x_p \geq 0 \end{array} \right\} \cup \left\{ \begin{array}{l} t_0 \geq t_1 \\ t_1 \geq t_2 \end{array} \right\} \quad (14)$$

We shall define now an important element in formulating the weighted MAX-SAT problem: the appropriate weight of each clause in K_D . The underlying idea is that a more specific default, i.e., those that belong to the higher order partition, should have priority over a less specific one. Hence, falsifying a more specific default will result in a higher cost. Since the clauses generated from defaults belonging to higher partitions might have a higher cost, the system will prefer to falsifying a clause from a less specific default than a more specific one.

Definition 17. (Cost Attribution) A cost attribution $f(\cdot)$ to the default conditional knowledge base K_D is a mapping of each partition of K_D to $\mathbb{R}^+$.

Definition 18. (Admissible Cost Attribution) A cost attribution f is admissible wrt K_D iff $\sum_{i=0}^{j-1} f(K_{Di}) < f(K_{Dj})$, for $j \in \{1, \dots, m\}$.

This condition certifies that violating a less specific default than a more specific one is preferred.

Definition 19. (*MAX-SAT* (Δ)) Given a p -consistent Δ with m partitions enumerated as $\{K_{D0}, K_{D1}, \dots, K_{Dm}\}$ and an admissible cost attribution $f(\cdot)$, we denote by $\text{MAX-SAT}(\Delta)$ the following combinatorial optimization problem:

$$\text{Min } \sum_{i=0}^m f(K_{Di}) \quad t(K_{Di}) \quad (15)$$

Subject to $sd(\Delta) \cup pd(\Delta)$

Now, the main point consists of finding a relation, if there exists one, among the solutions of the MAX-SAT problem described and the minimal interpretations of Δ .

Definition 20. (*Set of Clauses from* Δ) By $\mathcal{C}(\Delta)$ we denote the set of clauses associated to Δ , defined as follows:

$$\mathcal{C}(\Delta) = \{c_i | c_i \in \text{CNF}(\delta_m^*) \text{ for all } \delta_m \in \Delta\} \quad (16)$$

Definition 21. (*Binary variables of* Δ) $B(\Delta)$ represents the set of binary variables associated to Δ , and it is defined as:

$$B(\Delta) = \{x_a | x_a \in B(\mathcal{H}_{c_i}), \text{ for all } c_i \in \mathcal{C}(\Delta)\} \quad (17)$$

Definition 22. (*Variables Attribution*) Let w be an interpretation of Δ . The attribution of binary variables generated by w , that we denote by s_w , is defined as:

$$\begin{array}{ll} \text{for all } x_a \in B(\Delta): & \text{for all variables } t_i: \\ s_w(x_a) = \begin{cases} 1 & \text{if } a \text{ is true in } w \\ 0 & \text{otherwise} \end{cases} & s_w(t_i) = \begin{cases} 1 & \text{if a default } \delta_i \text{ is falsified by } w \\ 0 & \text{otherwise} \end{cases} \end{array} \quad (18)$$

From an attribution of binary variables s_w generated by an interpretation w we can easily generated a solution for $\text{MAX-SAT}(\Delta)$:

Definition 23. (*Interpretation generated by a solution*) If u is a solution for a $\text{MAX-SAT}(\Delta)$ problem, then the interpretation generated by u , represented by w_u , is achieved adjusting the truth value of each literal a present in Δ to true iff $x_a = 1$ in u and adjusting the truth value of all others literals to false.

Now, we can establish one of the main results of this paper. Informally, the following theorem affirms that exist a correspondence one-to-one among the solution of a weighted MAX-SAT and the minimal interpretations of Δ wrt Z -system semantic.

Theorem 1. u is an optimal solution to the $\text{MAX-SAT}(\Delta)$ problem (19) iff exist a minimal interpretation m wrt a Z -system semantic of Δ , such that $s_m = u$ and $w_u = m$.

Next we define the notion of consensual applied to a default $\delta_i : \alpha \rightsquigarrow \beta$ wrt the MAX-SAT problem(Δ).

Definition 24. A default $\delta_m : \alpha \rightsquigarrow \beta$ is consensual wrt MAX-SAT(Δ) iff the cost of any optimal solution³ of MAX-SAT(Δ) $\cup \{x_\alpha = 1, x_\beta = 1\}$ is smaller than the cost of any optimal solution of MAX-SAT(Δ) $\cup \{x_\alpha = 1, x_\beta = 0\}$. If both costs are equal then the default $\delta_n : \alpha \rightsquigarrow \beta$ is undecidable wrt Δ .

We now shall introduce the main result of this paper. Informally this result says that Δ Z-entails a default $\delta_i : \alpha \rightsquigarrow \beta$ iff this default is consensual wrt combinatorial optimization problems resulting from the ζ -translation of $\Delta = (K_D, L)$.

Theorem 2. A default $\delta_i : \alpha \rightsquigarrow \beta$ is Z-entailed by Δ iff $\delta_i : \alpha \rightsquigarrow \beta$ is consensual wrt MAX-SAT(Δ) problem.

The following algorithm can determine if a default is Z-entailed by $\Delta = (K_D, L)$.

Algorithm 3 *Input:* $\Delta = (K_D, L)$ and a default $\delta_m : \alpha \rightsquigarrow \beta$; *Output:* Yes or No.

1. Build the optimization problems MAX-SAT(Δ) $\cup sd'(\delta_m)$ and MAX-SAT(Δ) $\cup sd'(\delta_n)$, where $\delta_n : \alpha \rightsquigarrow \neg\beta$.
2. Solve these problems using one of the algorithms for integer programming. Let c and c' be the cost of optimal solutions of the first and second problems, respectively.
3. If $c < c'$ then return yes else return no.

5 Discussion

Related Research. The literature have already proposed an instantiation of a default theory as an Integer Programming, but all of them regarding to logic programming. A reference is the work of Bell et al. [1], Kagan [9] and Simons [11]. However, our framework introduces a different translation, since we do not use logic programming but rather a conditional logic based on Z-System semantics. We also use a different inequality system, based on *weighted* MAX-SAT, to allow ranking among defaults of a conditional knowledge base.

Conclusion. We have shown how the inference problem over conditional knowledge bases can be treated as a combinatorial optimization problem through weighted MAX-SAT model. For each conditional knowledge base a family of weighted MAX-SAT problems is defined in a way that there exists a one-to-one relation among the optimal solutions of each one of these problems and the minimal models obtained by System Z. At first sight, rewriting inference problems

³ Note that all optimal solution for a combinatorial optimization problem has the same cost.

in conditional knowledge bases as combinatorial optimization problems may be look like trying to make a hard solving problem even more difficult. However, some issues make us believe that this approach deserves attention. Eiter and Lukasiewicz [4] have shown that the inference problem over conditional knowledge bases under various semantics is in general intractable. We believe that many integer programs can be solved once a special mathematical structure has been detected in the model, extending, therefore, the tractable classes of problems. Heuristics and approximation algorithms to MAX-SAT problem can be used to by-pass this obstacle.

Acknowledgements

We would like to thank the valuable comments provided by anonymous referees.

References

- [1] Colin Bell, Anil Nerode, Raymond T. Ng, and V. S. Subrahmanian. Mixed integer programming methods for computing nonmonotonic deductive databases. *Journal of the ACM*, 41(6):1178–1215, 1994. 61
- [2] Colin Bell, Anil Nerode, Raymond T. Ng, and V.S. Subrahmanian. Implementing deductive databases by mixed integer programming. *ACM Transactions on Database Systems*, 21(2):238–269, 1996. 53
- [3] V. Chandru and J.N. Hooker. *Optimization Methods for Logical Inference*. Series in Discrete Mathematics and Optimization. John Wiley & Sons, Inc., 1999. 53, 57
- [4] Thomas Eiter and Thomas Lukasiewicz. Default reasoning from conditional knowledge bases: Complexity and tractable cases. *Artificial Intelligence*, 124(2):169–241, 2000. 62
- [5] Moises Goldszmidt and Judea Pearl. On the consistency of defeasible databases. *Artificial Intelligence*, 52(2):121–149, 1991. 53, 54
- [6] T. Hailperin. *Boole’s Logic and Probability: A Critical Exposition from Standpoint of Contemporary Algebra and Probability Theory*. North Holland, Amsterdam, 1976. 53
- [7] John N. Hooker. A quantitative approach to logical inference. *Decision Support Systems*, 4:45–69, 1988. 53
- [8] Robert G. Jeroslow. *Logic-Based Decision Support. Mixed Integer Model Formulation*. Elsevier, Amsterdam, 1988. 53, 56
- [9] Vadim Kagan, Anil Nerode, and V.S. Subrahmanian. Computing definite logic programs by partial instantiation. *Annals of Pure and Applied Logic*, 67(1-3):161–182, 1994. 61
- [10] J. Pearl. System Z: A natural ordering of defaults with tractable applications to nonmonotonic reasoning. In Rohit Parikh, editor, *TARK: Theoretical Aspects of Reasoning about Knowledge*, pages 121–136. Morgan Kaufmann, 1990. 52, 53, 54
- [11] P. Simons. Towards constraint satisfaction through logic programs and the stable model semantics. Research report A47, Helsinki University of Technology, 1997. 61
- [12] H.P. Williams. Fourier-Motzkin elimination extension to integer programming problems. *Journal of Combinatorial Theory (A)*, 21:118–123, 1976. 56

Multiple Society Organisations and Social Opacity: When Agents Play the Role of Observers

Nuno David^{1,2,*}, Jaime Simão Sichman^{2,†} and Helder Coelho³

¹ Department of Information Science and Technology, ISCTE/DCTI
Lisbon, Portugal

Nuno.David@iscte.pt
<http://www.iscte.pt/~nmcd>

² Intelligent Techniques Laboratory, University of São Paulo, Brazil
Jaime.Sichman@poli.usp.br
<http://www.pcs.usp.br/~jaime>

³ Department of Informatics, University of Lisbon, Portugal
hcoelho@di.fc.ul.pt
<http://www.di.fc.ul.pt/~hcoelho>

Abstract. Organisational models in MAS usually position agents as plain actors-observers within environments shared by multiple agents and organisational structures at different levels of granularity. In this article, we propose that the agents' capacity to observe environments with heterogeneous models of other agents and societies can be enhanced if agents are positioned as socially opaque observers to other agents and organisational structures. To this end, we show that the delegation of the observation role to an artificial agent is facilitated with organisational models that circumscribe multiple opaque spaces of interaction at the same level of abstraction. In the context of the SimCog project [9], we exemplify how our model can be applied to artificial observation of multi-agent-based simulations.

1 Introduction

The architecture of a multi-agent system (MAS) can naturally be seen as a computational organisation. The organisational description of a multi-agent system is useful to specify and improve the modularity and efficiency of the system, since the organisation constraints the agents' individual behaviours towards the system goals. To this end, several organisational abstractions have been proposed as methodological tools to analyse, design and simulate MAS societies. Meanwhile, while most research lines tend to commonly use the concept of *society* as an influential organisational metaphor to specify MAS (see [6]), this concept is rarely understood as an explicit structural and

* Partially supported by FCT/PRAXIS XXI, Portugal, grant number BD/21595/99.

† Partially supported by CNPq, Brazil, grant number 301041/95-4.

relational entity. Rather than explicit entities, societies are often implicitly defined in terms of inclusiveness of multiple agents and other organisational structures, like communication languages, groups or coalitions (e.g.[3]). This tendency comes from the general conceptual idea of conceiving autonomous agents as mere *internal actors* of societies, as opposed to the possibility of conceiving them as external, *neutral observers*, creators, or even autonomous designers of one or multiple societies. Societies are then conceived as closed, possibly infinite, mutually opaque social spaces, with an omnipresent opaque observer in the person of the human designer. Whereas a few models in the literature have explicitly defined multiple societies (e.g.[3][7]), the concept of society in such models is still reducible to one of a group, where agents are viewed simultaneously as actors and non-neutral observers in a given society. Also in works with reactive agents [4] or simulation with cognitive agents [5], where the stress is given to emergent organisational structures, the role of opaque observation is not explicitly assigned to agents, but exclusively and implicitly defined in the person of the human designer.

Nevertheless, in the real world, we have the ability to create explicit organisational structures and reason about them, like other agents, institutions or even new societies (e.g., artificial agent societies). Similarly, the artificial agent's ability to build topologies of multiple societies can be very powerful. In some environments, especially in environments with cognitive agents, an important factor in the system dynamics is the agents' beliefs and social reasoning mechanisms about other agents and the environment. The agents' skill to create and observe societies dynamically, possibly within a same or different level of abstraction than their own society, corresponds to the ability to instantiate and observe *given* models of other agents and societies in the world, allowing agents to reason autonomously about the heterogeneity of different models of societies at various levels of observation. This capacity is especially important in MAS models specified to observe and inspect results of simulations that involve other self-motivated agent societies.

The problem of "agentified" autonomous design and observation is partially the problem of delegating the human's observer role to the artificial agent. When an agent adopts the observer's role he should be able to create and observe dynamical aspects of organisational structures in other societies. In some situations, the observer agent must have the ability to look inside the other agents' minds. In others it will even be useful to give the agent the ability to pro-actively influence or change the organisational structure and cognitive representations of other agents in other societies. But while the observed agents and societies must be visible to the observer at various dimensions, the observer must be *socially opaque* to the observed agents. The model that we propose in this paper characterizes an organisation composed of multiple societies, where certain organisational configurations are able to dynamically manage different degrees of *social opacity* between these societies. A multiple society organisation is an environment in which the agents are themselves capable of creating explicit organizational structures, like other agents or societies. The problem of *social opacity* questions the conditions under which the control of cognitive information transfer between agents in different societies is possible.

This paper is organised as follows. In section 2 we will present our organisational model of multiple societies. In section 3 we will analyse two different organisational

abstractions that can be used to circumscribe opaque social spaces in our model. In section 4 we will present an application example related to multi-agent based simulations. Finally, in section 5, we will present some related work and conclusions.

2 One, Two, Three, Many Societies

2.1 Multiple Society Organisations

From an observer's point of view the concept of society encircles the vision of a common interaction space that allows agents to coexist and interact, generating the conditions for the explicit or emergent design of organizational structures. Since a society may contain any number of such structures our concept of society belongs to a higher level of abstraction than those structures. Yet, some of the social features of computational MAS must ultimately be specified by a minimal set of organizational structures. In this sense, the following consideration of a society as an explicit organizational entity is instrumental to generalize models of one to many societies.

Our Multi-Society Organisation (MSO) is based on four explicit organizational ingredients as follows: **(i)** A set AGT of agents – agents are active entities that are able to play a set of roles. **(ii)** A set ROL of roles – a role is an abstract function that may be exercised by agents, like different abilities, identifications or obligations. **(iii)** A set SOC of societies – a society is an interaction space that authorizes the playing of certain roles. An agent can enter a society and play a specific role if that role is *authorized* in that society. The partial functions $agtsoc: SOC \rightarrow P(AGT)$ and $rolsoc: SOC \rightarrow P(ROL)$ map a given society, respectively, on the set of agents that are *resident* in that society and the set of *authorized* roles in that society. **(iv)** A set RPY of role-players – we distinguish roles from role-players. Role-players are the actual entities through which agents act in the MSO. Each role-player is able to play a single role, but multiple role-players in the MSO can represent a same agent. For example, if the MSO is the planet earth and societies are nations, a possible situation for an agent with three role-players is to have a *Professor* role-player and a *Father* role-player in Portugal, and another *Professor* role-player in Brazil. In addition, every role-player holds a set of *delegable* roles that may be ascribed to other role-players upon its creation. We represent a role-player as a quadruple $rpy_i = (soc_i, agt_i, rol_i, R_i)$ composed by a society $soc_i \in SOC$, an agent $agt_i \in agtsoc(soc_i)$, a playing role $rol_i \in rolsoc(soc_i)$ and a set of *delegable* roles $R_i \in P(ROL)$. The partial function $delrol: RPY \rightarrow P(ROL)$ maps a given role-player on his set of delegable roles.

Definition 1. A MSO is a 7-tuple, $\langle AGT, ROL, SOC, agtsoc, rolsoc, RPY, delrol \rangle$, with components as above.

Agents interact in the MSO with others through *social events*, like message passing and creating other agents or societies. Agents can also be created on behalf of *external applications*. An external application (EA) is an entity capable of creating agents or societies in the MSO but that is not explicitly represented in the MSO, such as the agent launching shell. One may see EAs represented in a different level of abstraction

than the MSO. As a result, the transfer of information between agents can occur *explicitly* and internally in the MSO, through social events, or *implicitly* and externally to the MSO, via arbitrary interactions between agents, EAs, and again agents. For most of this paper we assume that *implicit* transfer of information does not take place. This is not always the case and we will refer to it when appropriate.

2.2 Social Dynamics

Agents and EAs can modify the state of the MSO along the time through *social events*. If a social event is on an agent's initiative, it must occur by means of his role-players. We call such role-player an *invoker* role-player. External applications originate social events when they wish to launch agents or societies in the MSO. Given a MSO in state k , the occurrence of social events will modify its state. We record the state of the MSO with a superscript like MSO^k . The invocation of a social event $MSO^k \rightarrow MSO^{k+1}$ depends on a set of pre-conditions. We define four social events $SE1, \dots, SE4$ as follows. The character $*$ next to a pre-condition denotes that the pre-condition is not applicable if the event is originated by EAs:

SE1: Society creation. Both role-players and EAs can invoke the creation of societies. Given a set of intended authorized roles, it may be the case that these roles are not yet defined in the MSO. The creation of a society that authorizes a set of roles R_j , will create a new society $soc_j \notin SOC^k$ and eventually a new set of roles in the MSO:

$$MSO^k \xrightarrow{SE1} MSO^{k+1} \mid agtsoc^{k+1}(soc_j) = \emptyset, rolsoc^{k+1}(soc_j) = R_j, \\ SOC^{k+1} = SOC^k \cup \{soc_j\}, ROL^{k+1} = ROL^k \cup R_j$$

SE2: Agent creation / SE3: Role-player creation. Agent creation refers to the instantiation of new agents in the MSO, invoked by other agents or EAs. When a new agent is instantiated, a new role-player must be created in some target society. However, if an agent is already instantiated, a similar social event is the creation of additional role-players, which cannot be invoked by EAs. This event occurs when agents want to be represented with multiple role-players in a same society or join additional societies with new role-players. In this paper we will only illustrate the specification of agent creation. We use the subscript i to refer to the creator agent and the subscript j to the new agent. If the social event is on an agent's initiative, consider its invoker role-player rpy_i . The creation of a new agent in a target society $soc_j \in SOC^k$, playing the target role rol_j , with delegable roles R_j , generates a new agent $agt_j \notin AGT^k$, a new role-player $rpy_j = (soc_j, agt_j, rol_j, R_j)$ and, possibly, a new set of roles in the MSO:

Pre-conditions: if **(c1)** $rol_j \in rolsoc^k(soc_j)$, the target role rol_j must be authorized in the target society soc_j ; **(c2*)** $rol_j \in delrol^k(rpy_i)$, the target role rol_j must be delegable by the invoker role-player rpy_i ; **(c3*)** $R_j \subseteq delrol^k(rpy_i)$, the target set of delegable roles R_j must be a subset of the invoker role-player rpy_i delegable roles.

$$MSO^k \xrightarrow{SE2} MSO^{k+1} \mid AGT^{k+1} = AGT^k \cup \{agt_j\}, ROL^{k+1} = ROL^k \cup R_j, \\ RPY^{k+1} = RPY^k \cup \{rpy_j\}, delrol^{k+1}(rpy_j) = R_j, agtsoc^{k+1}(soc_j) = agtsoc^k(soc_j) \cup \{agt_j\}$$

SE4: Message passing in a society. Only role-players can originate this social event, therefore excluding EAs. Message passing in the MSO does not alter its structure, but the sender and receiver role-players must operate in the same society.

The particularity of a MSO is the possibility of creating multiple societies in the same level of abstraction: an agent may be the creator of a society and also its member; and a member of the created society can be a member of the creator's society. In effect, while role-players can only communicate with each other if they share a same society, a same agent can act with multiple role-players across multiple societies. As a result, societies are not *opaque* relative to each other, in terms of information transfer between agents residing in different societies.

3 Social Spaces and Opacity

3.1 Visibility Dimensions

The set of social events and pre-conditions for its invocation determines the conditions to analyse the opacity between different societies. Opacity is also dependent on the organisational dynamics. Ultimately, if an agent ever resides in more than one society during his life cycle, opacity will depend on the agent's internal mechanisms, with respect to the transfer of information between its different roles-players. In general, we characterize the opacity of a society according to information transfer conditions from the *inside* to the *outside* of a society.

To begin with, we analyse the opacity of a society along three dimensions: **(i) Organisational visibility** – relative to the access, from the outside of a society, to organizational properties of the society in the MSO global context, like its physical location or shape. E.g., a valley that appears to be the environment of an unknown tribe in the Amazon may become identifiable by a satellite photograph, even though we may have no relevant information from inside the tribe. In our MSO this is inherently obtainable through the invocation of social events that create organizational structures, i.e., the identification of a society is always visible to its creator and can become known by others through message passing. **(ii) Openness** – relative to organisational conditions, prescribed by the MSO designer, or subjective conditions, prescribed by agents inside the society, restricting agents in the outside from entering the inside. These may vary extensively, for instance, according to some qualified institutional owner (an human or artificial agent), which decides if some given agent may or may not enter the society. In our MSO openness will ultimately depend on the level of convergence between the set of authorized roles in a society and the set of delegable roles accessible to each agent's role-player. **(iii) Behavioural and cognitive visibility** – relative to the access, from the outside of the society, to behaviours or cognitive representations of agents in the inside. *Behavioural visibility* concerns the observation of social events; for instance, a spy satellite may try to scout the transmission of messages between agents in a competitor country. *Cognitive visibility* refers to the observation of the agents' internal representations, such as its beliefs. In our MSO, behavioural and/or cognitive visibility implies the superposition of agents in the inside and the outside of a society. This is a necessary but not a sufficient condition. As we will soon show, other mechanisms must be designed to provide behavioural and cognitive visibility.

Notice the three dimensions are not independent from each other. Suppose we have an MSO with two societies and there is not a single agent residing simultaneously in both societies. The organizational and cognitive visibility of one society relative to agents in the other will vary according to the existence of a potential *bridge* agent in the latter able to join the former. In this sense, the concept of opacity is related to the problem of circumscribing the *internal* from the *external* environment of a society. The circumscription of an internal environment depends essentially on two factors: (1) objective organizational conditions associated with the dynamic structure of the MSO and independent from the agents' internal representations, like communication or role playing conditions, and (2) different internal representations emerging cognitively [1] within each member, relative to his own individual perception about the range of its social environment, like for instance dependence relations (e.g.[2]). Our interest is to fix circumscriptions along the first factor so as to control the range of circumscriptions based on the second factor. We classify the internal space of a society along two vectors: communication and role-playing conditions.

3.2 Communication Opacity

We define the internal communication space of a society according to communication conditions between agents that are resident and agents that are not resident in that society. Consider the *Plane Communication Space* (PCS) of a society. The PCS circumscribes role-players that are able to communicate directly with each other using message passing inside the society, that is, inside the society plane boundaries.

Plane Communication Space. The PCS of a society $\text{soc}_j \in \text{SOC}$ is the set of all role-players in that society: $\text{PCS}(\text{soc}_j) = \{(\text{soc}_i, \text{agt}_i, \text{rol}_i, R_i) \in \text{RPY} \mid \text{soc}_j = \text{soc}_i\}$

Agents playing roles inside a society may also play roles outside. The *Internal Communication Space* (ICS) of a society expands the PCS by including additional role-players in the outside if the corresponding agents have role-players in the inside.

Internal Communication Space. The ICS of a society $\text{soc}_j \in \text{SOC}$ is the set of all role-players in the MSO controlled by agents who are members of that society:

$$\text{ICS}(\text{soc}_j) = \{(\text{soc}_i, \text{agt}_i, \text{rol}_i, R_i) \in \text{RPY} \mid \text{agt}_i \in \text{agtsoc}(\text{soc}_j)\}$$

Pure Internal Communication Space. The ICS of a society $\text{soc}_j \in \text{SOC}^k$ in state k is *pure* if for any state i , with $i \leq k$, the ICS coincides with the PCS.

In figure 1a we represent a non-pure ICS relative to society soc_j . There are two societies – soc_j and soc_i – and three agents – A, B and C. Each point represents an agent role-player, for several points may represent an agent. E.g., the role-player $\langle \text{soc}_j, A, r1, \{r2, r3\} \rangle$ is the agent A in society soc_j playing role $r1$ with delegable roles $\{r2, r3\}$. Society soc_j authorizes roles $r1$ and $r2$, and society soc_i authorizes $r2$ and $r3$. The ICS is non-pure because agents A and B are playing roles in both societies.

If for some state an agent resides simultaneously in two societies, the ICS of either society will be circumscribed outside the boundaries of the PCS, encompassing role-players of both societies. On the contrary, the ICS of a society is *pure* if there was never an agent with role-players in that society that has ever had role-players in any

other society. Nevertheless, a *pure* ICS is not a sufficient condition to guarantee the opacity of a society, at least in terms of openness and organisational visibility. To this end a set of organizational conditions must be established in order to preclude agents outside the society to be able to identify it and eventually create new agents within it.

Consider a society and a set of resident agents, all created by an EA. Suppose that (1) the organizational conditions do not ever allow for role-players outside that society to create role-players in the inside, in other words, the society is closed; (2) the agents inside the society cannot join other societies according to their design specification. The first condition can be achieved if all authorized roles in the society are different from all delegable roles in the outside. Since no agent will ever reside simultaneously inside and outside the society, the corresponding ICS will be *pure* and opacity will not depend on cognitive information transfer through the agents' internal architectures. With these strict conditions the society organizational visibility will exclusively depend on *implicit* information transfer through the EAs. However, it is precisely the impossibility of explicit information transfer between the inside and the outside of a society that makes its range of practical applications limited, restricted to systems where agents are designed to co-operatively achieve a given set of goals.

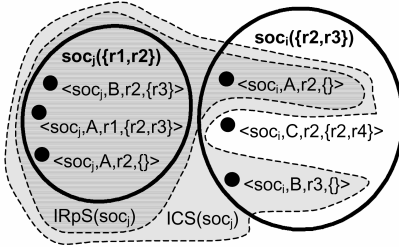


Fig. 1a. Non-pure ICS and non-pure IRpS

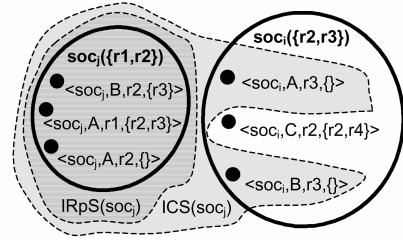


Fig. 1b. Non-pure ICS and pure IRpS

3.3 Role-Playing Opacity

Another way of circumscribing social spaces is to make use of role-playing conditions. The composition of communication and role-playing conditions allows an agent to play multiple roles simultaneously in the internal and external *pure* space of a society. The purpose of using role-playing conditions is to control social opacity through the agents' internal architectures. The *Internal Role-playing Space* (IRpS) of a society **subsets** the ICS by excluding role-players outside the society that do not have its playing roles authorized in the inside:

Internal Role-playing Space. The IRpS of a society $\text{soc}_i \in \text{SOC}$ is the set of all role-players in the corresponding ICS that have its playing roles authorized in that society:

$$\text{IRpS}(\text{soc}_i) = \{(\text{soc}_i, \text{agt}_i, \text{rol}_i, R_i) \in \text{ICS}(\text{soc}_i) \mid \text{rol}_i \in \text{rolsoc}(\text{soc}_i)\}$$

Pure Internal Role-playing Space. The IRpS of a society $\text{soc}_i \in \text{SOC}^k$ in state k is *pure* if for any state i , with $i \leq k$, the IRpS coincides with the PCS.

Figure 1a illustrates a non-pure IRpS relative to society soc_i . The IRpS is non-pure because agent A is playing role r2 in society soc_i , whereas role r2 is also authorized in

society soc_i . The difference between a *non-pure* and a *pure* IRpS is that in the first case an agent can play a same role inside and outside the society. The IRpS of a society stays *pure* if the agents with role-players in the inside do not have role-players in the outside whose roles are authorized in the inside. But differently from a *pure* ICS, opacity will now depend on the agents' internal mechanisms, with respect to the playing of different roles. Figure 1b illustrates a possible state for a *pure* IRpS.

The purpose of circumscribing role-playing spaces is to produce a flexible mechanism to design different organizational topologies of opaque and non-opaque *observation* spaces, according to role-playing conditions, that can be autonomously prescribed by the observer agent. Since the agents themselves can create other agents, roles and societies, the topology of social spaces may assume different configurations in a dynamic way. This means that the MSO itself can assume an emerging autonomous character from the human designer with respect to its own topology, as well as to its different points for opaque observation of social spaces.

4 MOSCA: An Opaque Organisation

The example that we illustrate in this section is motivated by the field of MAS simulations, especially simulation of cognitive agents (see [5,9]). In such simulations it is often the case that the simulated setting and the agents' behavioural rules or cognitive representations have to be observed or enforcedly modified during the simulation. The goal is to design a simulator based on MAS organisations, in the context of the Sim-Cog project [9]. With MOSCA (Meta-Organisation to Simulate Cognitive Agents), the simulation of MAS societies requires one MOSCA agent and two basic roles for *each* target agent intended as object of simulation: the *Control* and *Generic* roles.

The *Control* role is exclusively played within a society or set of societies (a region) called $S\_Control$, with a *pure* IRpS, whereby MOSCA agents co-operate for a common goal: to reproduce in a MAS distributed environment the behaviours of agents that are the real targets of simulation in a controllable way and **outside** the IRpS of the $S\_Control$ society. The set of societies outside the IRpS of $S\_Control$ is called the *Arena*. Hence, each MOSCA agent plays at least two roles expressing distinct behaviours: (i) the behaviour of a benevolent agent that cooperates with other MOSCA agents in the $S\_Control$ society, exclusively expressed through a *Control* role-player, in order to observe and maintain a consistent world state in the *Arena*, and (ii) a given arbitrary behaviour, exclusively expressed through a *Generic* role-player in the *Arena*, which is the effective target of simulation.

Besides reproducing the target agents' social events in the *Arena*, the MOSCA agents must respond to the users' requests throughout the simulation, such as observing social events or changing the targets' internal states. Owing to the distributed character of the environment, and to observation and intervention activities, each social event invoked by *Generic* role-players will imply a contingency set of social events invoked by *Control* role-players. Suppose the goal is to simulate a particular MAS organization, which we call the *target application*. The MSO is initially empty and MOSCA is an external application (EA). The simulation proceeds as follows:

Stage A. Launching MOSCA

(1) MOSCA loads the target application script that specifies the target agents, societies and delegable/authorized roles that must be launched to start the target application. We call *S_Arena* to the target society, *Generic* to every target role, and *generic-player* to any role-player playing the role *Generic*.

(2) Subsequently, MOSCA invokes the creation of a society called *S_Control* with a single authorized role called *Control*. As a result, the *S_Control* society will be liable to the playing of a single role. We use the name *control-player* to refer to role-players that play the *Control* role.

(3) MOSCA creates an agent called *Guardian* in the society *S_Control*. The *Guardian control-role* includes in his set of delegable roles the *Control* and *Generic* roles. The purpose of the *Guardian* is to coordinate the simulation with other MOSCA agents, while safeguarding the *S_Control* opacity to the *Arena*.

Stage B. Launching the Target Application

(4) The *Guardian* creates the target society *S_Arena* where the simulation will initially take place, with authorized role(s) *Generic*. Subsequently the *Guardian* creates a set of agents in the society *S_Control* that we call *Monitors*. Each *Monitor control-player* includes in his set of delegable roles the *Generic* but it does not include the *Control* role. This means that they are not able to create other *control-players*. Nevertheless, the *Monitors* are benevolent agents with a well-defined specification: to cooperate with the *Guardian* and other *Monitors* in order to reproduce in a controllable way the target application in the *Arena*.

(5) In the *S_Control* society, the *Guardian* notifies each *Monitor* about the target agents, delegable roles and the target society where the targets will be created.

Stage C. Running the Simulation

(6) At this point, the *Monitors* are ready to create and reproduce the target agents, expressing its social events through the society *S_Arena*, or any other society created during the simulation.

According to these conditions the IRpS of *S_Control* will be *pure*: since the *Control* role is not delegable to (and by) role-players outside *S_Control*, the target agents will never be able to join it. To attain social opacity, the computation of *control-roles* must be opaque to the computation of *generic-roles*, and this should be prescribed in the MOSCA agents' internal architectures. A point that should be stressed is that while the algorithm illustrates the creation of a single *S_Control* society, it can be easily generalisable to a set of mutually visible *S_Control* societies, i.e., an opaque region of several interacting *S_Control* societies. This is useful if one wants to distribute various points of observation according to the emergent topology of multiple societies in the *Arena*. Modularity and efficiency is the issue here. One can distribute different control societies according to different groups of target agents, associated with an independent logical or physical pattern of execution, like different simulation step algorithms (discrete time, event based) or efficiency patterns.

In figure 2 we illustrate an example with a control region that strictly follows a *mirror* topology: for every society created in the *Arena* another society and a corresponding *Guardian control-player* are created in the control region. Note that the target agents in the *Arena* can create recursively their own opaque observation spaces, but they will always be liable to observation in the control region.

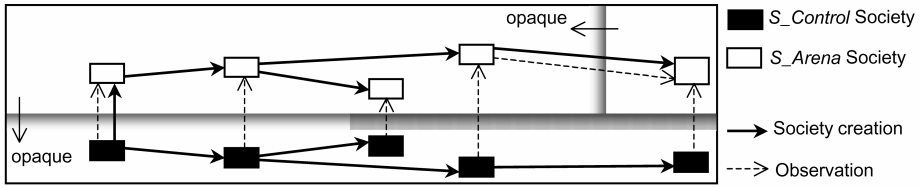


Fig. 2. Mirror control topology

5 Summary and Related Work

In this article we have proposed that the agents' capacity to observe heterogeneous models of other agents and societies can be enhanced if agents are positioned as socially opaque observers to other organisational structures. To this end, we have showed that the delegation of the human observer's role to an agent is facilitated with organisational models that instantiate multiple social spaces of interaction at the same level of abstraction. Nevertheless, we also showed that the right set of organisational conditions must be found if one wants to elect the agent as a socially opaque observer. We have exemplified how our model can be applied to the design of MAS simulators based on MAS organizations. Regarding this example, a related work that deserves special attention is the Swarm [10] simulation system. The Swarm model accommodates hierarchical modelling approaches in which agents are themselves composed of *swarms* (societies) of other agents in different levels of abstraction (e.g. bacteria composed of atoms). Each swarm (an agent) can observe agents in lower level swarms. However, visibility between agents of different swarms at the same level of abstraction is deliberately avoided and, consequently, agents in different swarms cannot interact explicitly. This is partly because the observer agent is represented within a different level of granularity from the observed agents. In contrast with the flexibility of our model, interaction between agents in different societies is therefore not transversal, since agents cannot create (and communicate with) other agents in other swarms at the same level of abstraction.

The idea of multiple social spaces has elsewhere been proposed in somewhat of a different approach [8], and it does not approach the problematic of social opacity. The authors speculate around the convenience of creating social spaces in the conceptualisation context of emergence and multiple viewpoints analysis. The usefulness of creating apprehensible micro-macro links in MAS is hypothesised, by giving the agents the means to become aware of their mutual interaction, and give birth to new types of agents and societies out of their collective activity. As the latter authors we believe that the answer to build interesting MAS is the creation of environments capable of showing spontaneous emergence along multiple levels of abstraction, while being compatible with the explicit design of organisational structures in order to actively observe, and eventually manipulate, such emergent structures at arbitrary levels of abstraction. The model that we have presented is a valuable and original starting point to that end. In the future we plan to investigate the problem of observation and social opacity in models with higher levels of organisational complexity.

References

1. Castelfranchi C., Simulating with Cognitive Agents: The Importance of Cognitive Emergence. In [5], pp.26-44, 1998.
2. David N., Sichman J.S. and Coelho H, Agent-Based Social Simulation with Coalitions in Social Reasoning. Multi-Agent-Based Simulation, Sp-Verlag, LNAI1979, p.245-265,2001.
3. Ferber J. and Gutknecht O., A Meta-model for the Analysis and Design of Organizations in MAS. Proc. of Int. Conf. in Multi-Agent Systems, IEEE Computer Society, 1998.
4. Ferber J., Reactive Distributed Artificial Intelligence: Principles and Applications. Foundations of DAI, G. O'Hare and Jennings N., editors, 1996.
5. Gilbert N., Sichman J.S. and Conte R. (eds), Multi-Agent Systems and Agent-Based Simulation, Springer Verlag, LNAI1534, 1998.
6. Huhns M. and Stephens L.M., Multiagent Systems and Societies of Agents. Multi-Agent Systems – A Modern Approach to AI, Weiss G. (editor), MIT press, 79-114, 1999.
7. Norbert G. and Philippe M., The Reorganization of Societies of Autonomous Agents. Multi-Agent Rationality, Proc. of MAAMAW'97, Sp. Verlag, LNAI1237, pp.98-111, 1997.
8. Servat D., Perrier E., Treuil J.P, and Drogoul A., When Agents Emerge from Agents: Introducing Scale Viewpoints in Multi-agent Simulations. In [5], pp.183-198, 1998.
9. SimCog, Simulation of Cognitive Agents. <http://www.lti.pcs.usp.br/SimCog/>.
10. Swarm, The Swarm Simulation System, <http://www.swarm.org/>.

Altruistic Agents in Dynamic Games

Eduardo Camponogara

Universidade Federal de Santa Catarina
Florianópolis SC 88040-900, Brasil

Abstract. The collective effort of the agents that operate distributed, dynamic networks can be viewed as a dynamic game. Having limited influence over the decisions, the agents react to one another's decisions by resolving their designated problems. Typically, these iterative processes arrive at attractors that can be far from the Pareto optimal decisions—those yielded by an ideal, centralized agent. Herein, the focus is on the development of augmentations for the problems of altruistic agents, which abandon competition to draw the iterative processes towards Pareto decisions. This paper elaborates on augmentations for unconstrained, but general problems and it proposes an algorithm for inferring optimal values for the parameters of the augmentations.

1 Motivation

A standard approach to coping with the complexity of a large, dynamic network divides the control task into a sizeable number of small and local, dynamic problems [5], [6]. A problem is small if it has far fewer variables and constraints than the whole of the network; it is local if its variables are confined to a neighborhood. The distributed agents, having limited authority over the variables in their neighborhoods, compete with their neighboring agents as they do the best for themselves in solving the problems entrusted to them. Thus, this standard approach reduces the operation of the network to a dynamic game among its distributed agents. But this reduction has a price: the iterative processes used by the agents to solve their problems often, if not always, reach decisions that are suboptimal. In typical networks, such as traffic systems and the power grid, the optimal decisions from the viewpoint of an agent can be far from the best if the entire network is accounted for—cascading failures in power systems and traffic jams caused by an inept handling of contingencies are dramatic instances of a suboptimal operation.

Above all, the game-theoretic view brings out two issues of concern: the convergence to and location of attractors [12]. For one thing, only the iterative processes that converge to attractors induce a stable operation of the dynamic network. For another, only the attractors that induce Pareto optimal decisions yield an optimal quality of services, which in principle can be obtained with an ideal, centralized agent. To improve the quality of services, automatic learning techniques could be embedded in the agents, allowing them to infer decision policies from past records that promote convergence to near-optimal attractors.

More specifically, two applications of these techniques are: the prediction of the reactions of an agent's neighbors, which would allow the agent to proceed asynchronously, and the recognition of decisions, perhaps counter-intuitive from the agent's viewpoint, that draw the attractor closer to the Pareto set.

The work reported here is a relevant step to improve convergence to attractors and their location. For dynamic games originating from the iterative solution of unconstrained optimization problems, the paper develops simple yet powerful augmentations for the problems so as to influence the issues of concern. These augmentations are called altruistic factors and the agents that implement them, altruistic agents. To compute the altruistic factors, an algorithm is designed to learn their values from the interactions among the agents, but the problems are further restricted to quadratic functions. Nevertheless, this work paves the ground for further developments. The rest of the paper elaborates on the aforementioned altruistic factors and the learning algorithm, providing an illustrative example.

2 Dynamic Games

The roots of game theory can be traced back to the pioneering work of Von Neumann, followed by the more rigorous formalization of Kuhn and the insightful notions of equilibrium by Nash [2]. In essence, the domain is concerned with the dynamic and outcomes of multi-player decision making, wherein each competitive player does the best for itself by influencing only a few of the variables, while its profit depends on the decisions of the others as well. Game theory has proven to be a powerful tool for analysis in economics [1], a means for modeling and synthesis of control strategies in robotics [8], and, more related to this work, a framework for understanding the interplay among the elements of multi-agent systems [4]. Although the borders between its branches are not clear-cut, a game is typically said to be "infinite" if the number of decisions available to at least one of its players is infinite, and "finite" otherwise. The game is said to be "dynamic" if the decisions of at least one of its players evolve in time, and "static" otherwise. Herein, the point of departure is an infinite, dynamic game arising from the solution of a set of problems $\{P_m\}$, one for each of the agents, which are of the following form:

$$\begin{aligned} P_m : \text{Minimize} \quad & f_m(x_m, y_m, \dot{x}_m, \dot{y}_m, t) \\ & x_m \\ \text{Subject to :} \quad & H_m(x_m, y_m, \dot{x}_m, \dot{y}_m, t) = 0 \\ & L_m(x_m, y_m, \dot{x}_m, \dot{y}_m, t) \leq 0 \end{aligned}$$

where: x_m is the vector with the decisions under control of the m -th agent; y_m is the vector with the decisions of the other agents; f_m is the agent's objective function; and, H_m and L_m are vector functions corresponding to the equality and inequality constraints.

In a competitive setting, as the agents react to one another's decisions by re-solving their problems, the aggregate of their decisions $x = (x_m, y_m)$ traces

a trajectory in decision space that, if convergent, arrives at a Nash equilibrium point. To present these concepts more formally, let $R_m(y_m, t)$ be the reaction set of agent- m at time t —the best decisions from the agent’s point of view—which is defined as:

$$R_m(y_m, t) = \{x_m : \underset{x_m}{\text{Argmin}} \quad f_m(x_m, y_m, \dot{x}_m, \dot{y}_m, t) \\ \text{Subject to : } H_m(x_m, y_m, \dot{x}_m, \dot{y}_m, t) = 0 \\ L_m(x_m, y_m, \dot{x}_m, \dot{y}_m, t) \leq 0\} .$$

An aggregate of the decisions x induces a Nash point if no rational, competitive agent- m has any incentive to deviate from its decisions x_m unilaterally—i.e., the agent will be worse off if it changes the values of x_m so long as the other agents stick to their decisions.

The above game-theoretic framework is of high generality and complexity, serving the purpose of modeling dynamic systems operated by autonomous agents. There are, of course, many issues of concern that seem difficult to be resolved in general, such as the feasibility of the agents’ problems over time and the convergence of their decisions to an attractor, leaving the challenge to address them on a case-by-case basis with numerical or analytical means.

3 Inducing Convergence to Attractors

Hereafter, the focus is on games consisting of unconstrained, time-invariant problems that are much simpler than those appearing in the general game-theoretic framework of the preceding section. There is merit despite the seemingly simplifications: with respect to constraints, the agents can approximate a constrained problem with a series of unconstrained subproblems, typically resorting to barrier and Lagrangean methods [10]; likewise, with respect to the time-dependency, the agents can solve a series of static approximations, in the same manner that model predictive control treats dynamic control problems [9].

This paper extends our preceding developments, which were confined to quadratic games [7], by assuming that the problem of agent- m is of the form:

$$P_m : \underset{x_m}{\text{Minimize}} \quad f_m(x_m, y_m)$$

where, as before, x_m is the vector with the decisions of the agent, y_m is the vector with the decisions of the others, and f_m is a continuously differentiable function expressing the agent’s objective.

Assumption 1. *The reaction set of each agent- m is obtained by nullifying the gradient of f_m with respect to x_m , i.e., $R_m(y_m) = \{x_m | \frac{\partial f_m}{\partial x_m} = 0\}$. Further, the agent’s reaction function G_m arises from the selection of one element from R_m , i.e., $x_m(k+1) = G_m(y_m(k))$ where G_m is a function such that $G_m(y_m) \in R_m(y_m)$.*

Definition 1. *The parallel, iterative process induced by the reactions of M agents is $G = [G_1, \dots, G_M]$, implying that $x(k+1) = G(x(k))$.*

Definition 2. *For the m -th agent, a vector $\alpha_m \in R^{\dim(x_m)}$, such that no entry of α_m is zero, is referred to as the agent's altruistic factors for convergence. The vector of all convergence factors is $\alpha = [\alpha_1, \dots, \alpha_M]$.*

Proposition 1. *If the m -th agent uses convergence factors from α_m to replace its objective function with $f'_m = f_m(D(\alpha_m)^{-1}x_m, y_m)$, then its reaction becomes $x_m(k+1) = D(\alpha_m)G_m(y_m(k))$, where $D(\alpha_m)$ is the diagonal matrix whose diagonal corresponds to the entries of α_m .*

Proof. With z_m as $D(\alpha_m)^{-1}x_m$, it follows from Assumption 1 that $z_m(k+1) = G_m(y_m(k)) \Rightarrow D(\alpha_m)^{-1}x_m(k+1) = G_m(y_m(k)) \Rightarrow x_m(k+1) = D(\alpha_m)G_m(y_m(k))$. $\square$

Proposition 2. *Let $\alpha = [\alpha_1, \dots, \alpha_M]$ be a vector with the altruistic factors of M agents. (The competitive agent- m sets $\alpha_m = 1$.) If the agents modify their problems as delineated in Proposition 1, then the resulting iterative process, $x(k+1) = D(\alpha)G(x(k))$, can be made more contractive if $\|D(\alpha)\|_\infty < 1$.*

Proof. The net effect of implementing the altruistic factors from α is the conversion of the original iterative process, $x(k+1) = G(x(k))$, into $x(k+1) = D(\alpha)G(x(k))$. Suppose that for some vector-norm $\|\cdot\|$ and scalar $\gamma \geq 0$, $\|G(x^a) - G(x^b)\| \leq \gamma\|x^a - x^b\|$ for all x^a, x^b . Thus, $\|D(\alpha)G(x^a) - D(\alpha)G(x^b)\| = \|D(\alpha)[G(x^a) - G(x^b)]\| \leq \|\text{Max}\{|\alpha_k| : k = 1, \dots, \dim(\alpha)\}\| \|G(x^a) - G(x^b)\| = \text{Max}\{|\alpha_k|\} \|G(x^a) - G(x^b)\| = \|D(\alpha)\|_\infty \|G(x^a) - G(x^b)\| \leq \|D(\alpha)\|_\infty \gamma \|x^a - x^b\|$ for all x^a, x^b and, therefore, the resulting iterative process is more contractive than the original process if $\|D(\alpha)\|_\infty < 1$. $\square$

One of the most fundamental results of iterative processes is that (synchronous) parallel iterations converge to a unique attractor, a fixed point x^* which satisfies the equation $x^* = G(x^*)$, if the operator G induces a contraction mapping for some vector-norm $\|\cdot\|$, that is, if $\|G(x^a) - G(x^b)\| \leq \gamma\|x^a - x^b\|$ for all x^a, x^b and some $0 \leq \gamma < 1$ [11]. In light of this fact and Proposition 2, the altruistic agents can promote convergence by picking values for their factors that induce $\|D(\alpha)\|_\infty < 1$. Although a contraction mapping cannot always be obtained if one or more agents remain competitive, examples can be easily conceived to illustrate that even in the presence of competition the altruistic agents can draw the decisions to attractors of an otherwise divergent game—a consequence of the conditions being sufficient, but not necessary for convergence. On a side note, asynchronous convergence to the unique attractor is guaranteed if the iterative process induces a contraction mapping for the l_∞ -vector-norm, $\|\cdot\|_\infty$, [3], thereby allowing each agent- m to use values of y_m not as recent as $y_m(k)$ in computing its reaction $x_m(k+1)$.

4 Relocating Attractors

Thus far, our developments have shown how altruistic agents can, for the overall good, improve convergence of iterative processes through simple modifications of their objectives. Not unlike these contributions, altruistic agents can alter their objective functions allowing them to drive the attractor nearer to the Pareto optimal set—the optimal solutions from the perspective of centralization¹.

Definition 3. For the m -th agent, a vector $\beta_m \in R^{\dim(x_m)}$ is referred to as the agent's altruistic factors for location. The vector with all of the location factors is $\beta = [\beta_1, \dots, \beta_M]$.

Proposition 3. If the m -th agent uses location factors from β_m to replace its objective function with $f'_m = f_m(x_m - \beta_m, y_m)$, then its reaction becomes $x_m(k+1) = G_m(y_m(k)) + \beta_m$.

Proof. By naming z_m as $(x_m - \beta_m)$, it follows from Assumption 1 that $z_m(k+1) = G_m(y_m(k)) \Rightarrow x_m(k+1) - \beta_m = G_m(y_m(k)) \Rightarrow x_m(k+1) = G_m(y_m(k)) + \beta_m$. $\square$

Proposition 4. Let $\beta = [\beta_1, \dots, \beta_M]$ be a vector with the altruistic factors for location of M agents. (The competitive agent- m sets $\beta_m = 0$.) If the agents modify their problems as delineated in Proposition 3, then the resulting iterative process inherits the same contraction properties of the original process, while the location of its attractor is influenced by the value of β , i.e., the solution x^* to the equation $x = G(x) + \beta$ defines an attractor.

Proof. The iterative process arising from the implementation of β factors, G' , is defined as $x(k+1) = G'(x(k)) = G(x(k)) + \beta$, where G is the original process. Clearly, an attractor for G' must solve the equation $x = G(x) + \beta$ and, therefore, it can be relocated by tweaking the values of β . With respect to the contraction properties, for some vector-norm $\|\cdot\|$ and points x^a, x^b , $\|G'(x^a) - G'(x^b)\| = \|G(x^a) + \beta - G(x^b) - \beta\| = \|G(x^a) - G(x^b)\|$, which implies that the contraction produced by G carries over to G' . $\square$

Fig. 1 depicts a dynamic game between two agents. The plot shows the contour lines of the agents' objective functions, their reaction curves (R_1 and R_2), and the set of Pareto points $\mathcal{P}$. Both the serial and parallel iterations recede from the Nash equilibrium point, $\mathcal{N}$, unless the agents begin at the Nash point, $(-4.33, -4.03)$. Agent-1 can, however, draw the decisions to an attractor (Nash point) if it chooses to be altruistic by setting its altruistic factor α_1 to $1/5$. If both agents behave altruistically, with agent-1 implementing altruistic factors for convergence as well as location and agent-2 implementing altruistic factors for location, the agents can place the attractor inside the Pareto optimal set.

¹ A solution x^a belongs to the Pareto optimal set if there does not exist another solution x^b such that $f_m(x^b) \leq f_m(x^a)$ for all m and $f_m(x^b) < f_m(x^a)$ for some m .

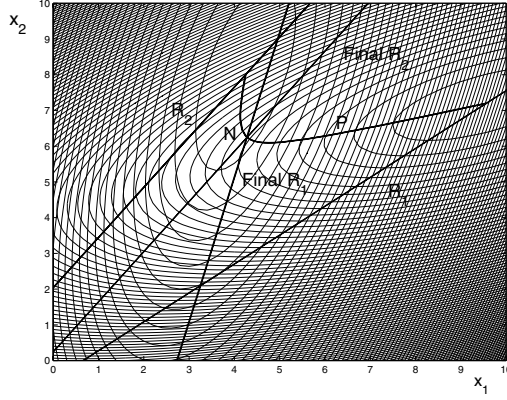


Fig. 1. The attractor obtained if both agents are altruistic, with agent-1 setting $\alpha_1 = 1/5$ and $\beta_1 = 2.6$ while agent-2 sets $\beta_2 = -1.8$. The location of the attractor (Nash point) intercepts the Pareto optimal set of the original game. The original Nash point was located at $x^a = (-4.33, -4.03)$, yielding $f_1(x^a) = 1,626$ and $f_2(x^a) = 1,701$. The final attractor is located at $x^b = (4.27, 6.24)$, yielding $f_1(x^b) = -968.36$ and $f_2(x^b) = -571.70$. The problems of the agents are:

$$\begin{aligned} P_1 : \text{Min}_{x_1} \quad & f_1 = 9.11215x_1^2 - 22.5402x_1x_2 + 35.88785x_2^2 - 11.9718x_1 - 301.2580x_2 \\ P_2 : \text{Min}_{x_2} \quad & f_2 = 47.00345x_1^2 - 22.4380x_1x_2 + 7.99655x_2^2 - 219.8309x_1 - 32.6516x_2 \end{aligned}$$

5 Inferring Altruistic Responses in Quadratic Games

Our ultimate goal is to have the agents implement some sort of altruistic response and, from their interactions, infer or learn optimal behavior. Though optimal behavior can be further elaborated, herein we define it as the one leading to convergence of the iterative process to an attractor that is in some sense as close as possible to the Pareto set, while the learning process does not incur excessive computational burden. Achieving this goal is a daunting, however necessary task to improve the quality of the services delivered by the agents that operate dynamic systems. In what follows, we report a step towards achieving this goal—specifically, for quadratic and convergent games, we deliver an algorithm that allows the altruistic agents to infer factors β that optimize an aggregate of the agents' objectives. In quadratic games, agent- m 's problem is of the form:

$$P_m : \text{Minimize}_{x_m} \quad \frac{1}{2}x^T A_m x + b_m^T x + c_m$$

where: x_m is a vector with the decision variables of the agent; $x = [x_1, \dots, x_M]$ is a vector with the decisions of all the agents; A_m is a symmetric and positive definite matrix; b_m is a vector; and, c_m is a scalar. By breaking up A_m into

sub-matrices and b_m into sub-vectors, P_m can be rewritten as:

$$P_m : \underset{x_m}{\text{Minimize}} \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M x_i^T A_{m,i,j} x_j + \sum_{i=1}^M b_{m,i}^T x_i + c_m .$$

In accordance with this notation, the agent- m 's iterative process becomes:

$$x_m(k+1) = G_m(y_m(k)) = -[A_{m,m,m}]^{-1} \left[\sum_{n \neq m} A_{m,m,n} x_n(k) + b_{m,m} \right] . \quad (1)$$

Putting together the agents' iterative processes, we can express the overall iterative process G as the solution to $Ax(k+1) = -Bx(k) - b$ for suitable A , B , and b ². The solution to this equation leads to the iterative process $x(k+1) = G(x(k)) = -A^{-1}[Bx(k) + b]$. In case agent- m is altruistic with respect to the location of the attractor, its problem takes on the following form after introducing the factors from β_m :

$$P'_m : \underset{x_m}{\text{Minimize}} \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M (x_i - \beta_{m,i})^T A_{m,i,j} (x_j - \beta_{m,j}) + \sum_{i=1}^M b_{m,i}^T (x_i - \beta_{m,i}) + c_m$$

where: $\beta_m = [\beta_{m,1}, \dots, \beta_{m,M}]$ is the vector with the altruistic factors of agent- m ; $\beta_{m,i} = 0$ for all $i \neq m$; and, the other variables and parameters are identical to their equivalent in P_m . Under altruism, the iterative process of the m -th agent arises from the solution of P'_m , becoming:

$$x_m(k+1) = G'_m(y_m(k)) = -[A_{m,m,m}]^{-1} \left[\sum_{n \neq m} A_{m,m,n} x_n(k) + b_{m,m} \right] + \beta_{m,m} . \quad (2)$$

As before, the overall iterative process G' of altruistic agents can be obtained from the solution of the equation $A[x(k+1) - \beta] = -Bx(k) - b$ where A , B , and b are identical to those appearing in the iterative process without altruism and $\beta = [\beta_{1,1}, \dots, \beta_{M,M}]$. The solution to this equation yields the following iterative process for altruistic agents:

$$x(k+1) = G'(x(k)) = -A^{-1}[Bx(k) + b] + \beta . \quad (3)$$

Assumption 2. $\|A^{-1}B\| < 1$ for some matrix-norm $\|\cdot\|$ induced by a vector-norm $\|\cdot\|$, which implies that G as well as G' induce contraction mappings.

5.1 Predicting the Location of the Attractor

By manipulating (3), the location of the attractor can be cast as a linear function of its original location and the elements of $\{\beta_{m,m}\}$ as follows:

$$\begin{aligned} x^*(\beta) &= -(I + A^{-1}B)^{-1}(A^{-1}b - \beta) = x^*(0) + Z\beta = \\ &= x^*(0) + Z_1\beta_{1,1} + \dots + Z_M\beta_{M,M} . \end{aligned} \quad (4)$$

² $A = [A_{1,1,1} \ 0 \ \dots \ 0; 0 \ A_{2,2,2} \ 0 \ \dots \ 0; \dots; 0 \ \dots \ 0 \ A_{M,M,M}]$, $b = [b_{1,1}; \dots; b_{M,M}]$, and $B = [0 \ A_{1,1,2} \ \dots \ A_{1,1,M}; A_{2,2,1} \ 0 \ A_{2,2,3} \ \dots \ A_{2,2,M}; \dots; A_{M,M,1} \ \dots \ A_{M,M,M-1} \ 0]$.

Remark 1. The matrix $(I + A^{-1}B)$ admits an inverse because $|||A^{-1}B||| < 1$.

Let $\Psi \subseteq \{1, \dots, M\}$ be the subset with the ids of the altruistic agents. These agents can organize themselves to in turn learn their individual influence on the location of the attractor, i.e., for each $m \in \Psi$, agent- m can tweak the values of $\beta_{m,m}$ so as to compute Z_m . Hereafter, $x^*(\beta)$ denotes the attractor if the agents implement altruistic factors from β , as prescribed by (2), (3), and (4). The procedure below lists the steps for altruistic agents to calculate their influence on the location of the attractor.

Procedure 5.1: Computing the elements of $\{Z_m : m \in \Psi\}$

-
- The agents coordinate among themselves to set $\beta_{m,m} = 0$ for each $m \in \Psi$.
 - Let $x^*(0)$ be the attractor without altruism.
 - The altruistic agents schedule themselves to run one at time, so that for each $m \in \Psi$, agent- m executes the steps below.
 - For $k = 1$ to $\dim(\beta_{m,m})$ do
 - Set $(\beta_{m,m})_k = 1$.
 - Allow the agents to iterate until they reach the attractor $x^*(\beta)$.
 - According to (4), the k -th column of Z_m is the vector $x^*(\beta) - x^*(0)$.
 - Set $(\beta_{m,m})_k = 0$.
 - At the end of the loop, Z_m is known.
-

5.2 Improving the Location of the Attractor

At this stage, the agents are in a position to coordinate their actions to draw the attractor "closer" to the Pareto set. The issue yet to be addressed is how one measures closeness to this set. Actually, the goal is to reach decisions that are Pareto optimal, but this may be unattainable in the presence of competitive agents, which leaves the possibility of reaching an attractor that induces lower overall cost. This, in turn, means that we need a criterion for establishing preference among different attractors. One way of improving the attractor's location consists in maximizing the minimum reduction over all agents' objectives, more formally the problem can be expressed as:

$$\text{OP}_a : \text{Max}_\beta \text{Min}\{f_m(x^*(0)) - f_m(x^*(\beta)) : m = 1, \dots, M\}.$$

Another way is the minimization of a weighted sum of the agents' objectives, which spells out the relative preferences among them, more formally:

$$\text{OP}_b : \text{Min}_\beta \sum_{m=1}^M w_m f_m(x^*(\beta)) \equiv \text{Min}_\beta \frac{1}{2} x^*(\beta)^T \mathcal{A} x^*(\beta) + \mathcal{B}^T x^*(\beta) + \mathcal{C}$$

where: w_m is a positive constant; $\mathcal{A}$ is a suitable matrix; $\mathcal{B}$ is a suitable vector; and, $\mathcal{C}$ is a suitable scalar. Here, the focus is on the distributed solution of the overall problem OP_b by the altruistic agents. In essence, the altruistic agents will solve OP_b indirectly: in turns, each agent- m computes $\beta_{m,m}$ to reduce the

objective of OP_b and then implements $\beta_{m,m}$ in its reaction function (2), thereby allowing the attractor to reach the improved location. More precisely, agent- m will tackle the following form of OP_b :

$$\text{OP}_m : \underset{\beta_{m,m}}{\text{Minimize}} \quad \frac{1}{2} x^*(\beta_{m,m})^T \mathcal{A} x^*(\beta_{m,m}) + \mathcal{B}^T x^*(\beta_{m,m}) + \mathcal{C}$$

where: $x^*(\beta_{m,m}) = x^*(\gamma_m) + Z_m \beta_{m,m}$; $\gamma_m = [\beta_{k,k} : k = 1, \dots, M \text{ and } k \neq m]$; and, $x^*(\gamma_m)$ is the attractor obtained by using the current value of γ_m and having $\beta_{m,m} = 0$. It is worth mentioning that the computational effort necessary to solve OP_m is equivalent to that of solving P_m .

Remark 2. Because $\mathcal{A}$ is positive definite and Z_m has full column rank, the Hessian matrix $Z_m^T \mathcal{A} Z_m$ of the objective function of OP_m is positive definite.

Procedure 5.2: Solving OP_b

-
- The altruistic agents use Procedure 5.1 to compute $\{Z_m : m \in \Psi\}$.
 - The agents take turns, in any sequence, to execute the steps below.
 - Let $m \in \Psi$ correspond to the agent of the turn.
 - Agent- m senses the value of $x^*(\beta)$ and calculates $x^*(\gamma_m)$ using Z_m and $\beta_{m,m}$.
 - The agent proceeds to solve OP_m , yielding a new value $\beta_{m,m}$.
 - The m -th agent implements the new value of $\beta_{m,m}$ in its iteration function, (2), allowing the agents to reach the improved attractor.
 - Agent- m transfers the turn to the next agent in the sequence.
-

Proposition 5. *If the quadratic game is convergent and the altruistic agents follow Procedure 5.2, then the attractor $x^*(\beta)$ converges to an optimal solution to OP_b .*

Proof. Because the game is convergent and the agents go through a phase of learning the elements of $\{Z_m\}$, equation (4) predicts perfectly the location of the attractor as a function of β (assuming that $Z_m = [0]$ and $\beta_{m,m} = 0$ if the m -th agent is competitive). Thus, OP_m is equivalent to OP_b but constrained to the variable $\beta_{m,m}$. Let h_b denote the objective function of OP_b and $\nabla h_b(x^*(\beta))$, the gradient of h_b at $x^*(\beta)$. Further, let h_m denote the objective function of OP_m and $\nabla h_m(x^*(\beta_{m,m}))$, its gradient. Notice that if $\nabla h_b \neq 0$, then there must be at least one agent- m such that $\nabla h_m(x^*(\beta_{m,m})) \neq 0$. Let agent- m be the first such agent to run. By obtaining an optimal solution to OP_m , agent- m actually yields a solution to OP_b that is not worse than the best solution obtained for OP_b by searching along the direction induced by $-\nabla h_m(x^*(\beta_{m,m}))$. Because $-\nabla h_m(x^*(\beta_{m,m}))$ induces an improving direction for OP_b , $[0, \dots, 0, -\nabla h_m, 0, \dots, 0]^T \nabla h_b = -\nabla h_m^T \nabla h_m < 0$, and because OP_m was solved to optimality, the Wolfe conditions are met and global convergence follows [10] (pp. 35–46). $\square$

6 Closing Remarks

The decomposition approach to operating a large, dynamic network breaks the control task into a number of problems that are dynamic, small and local, one for each of its distributed agents. To cope with the dynamic nature of these problems, the agents instantiate series of static approximations thereby allowing the use of standard optimization techniques. The end result is a series of dynamic games, whose dynamics is dictated by an iterative process arising from the agents' iterative search for the solutions to their problems. Convergence of the iterative processes, if attained, is typically to suboptimal attractors.

To that end, this paper has developed augmentations for the problems of altruistic agents aimed to promote convergence of their iterative processes to improved attractors. The paper has also delivered an algorithm for computing optimal values for the parameters of these augmentations. Although these augmentations are confined to unconstrained problems and the algorithm applicable only to quadratic functions, the developments herein can play a role as heuristics for more general games and they seem to be extendable in a number of ways. For one thing, our recent analyses indicate that a trust-region algorithm [10] can be designed to infer optimal altruistic factors in general, unconstrained games. For another, series of unconstrained games can, at least in principle, approximate games of higher complexity.

Acknowledgments

The research reported here was funded in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico, Brasil, under grant number 68.0122/01-0.

References

- [1] Aumann, R. J., Hart, S. (eds.): Handbook of Game Theory with Economic Applications. Vol. 1. North-Holland, Amsterdam (1992) 75
- [2] Basar, T., Olsder, G. J.: Dynamic Noncooperative Game Theory. Society for Industrial and Applied Mathematics, Philadelphia (1999) 75
- [3] Bertsekas, D. P.: Distributed Asynchronous Computation of Fixed Points. Mathematical Programming 27 (1983) 107–120 77
- [4] Bowling, M., Veloso, M. M.: Rational and Convergent Learning in Stochastic Games. Proc. of the 17th Int. Joint Conference on Artificial Intelligence (2001) 1021–1026 75
- [5] Camponogara, E.: Controlling Networks with Collaborative Nets. Doctoral Dissertation. ECE Department, Carnegie Mellon University, Pittsburgh (2000) 74
- [6] Camponogara, E., Jia, D., Krogh, B. H., Talukdar, S. N.: Distributed Model Predictive Control. IEEE Control Systems Magazine 22 (2002) 44–52 74
- [7] Camponogara, E., Talukdar, S. N., Zhou, H.: Improving Convergence to and Location of Attractors in Dynamic Games. Proceedings of the 5th Brazilian Symposium on Intelligent Automation, Canela (2001) 76

- [8] LaValle, S. M.: Robot Motion Planning: A Game-theoretic Foundation. *Algorithmica* 26 (2000) 430–465 75
- [9] Morari, M., Lee, J. H.: Model Predictive Control: Past, Present and Future. *Computers and Chemical Engineering* 23 (1999) 667–682 76
- [10] Nocedal, J., Wright, S. J.: Numerical Optimization. Springer, New York (1999) 76, 82, 83
- [11] Ortega, J. M., Rheinboldt, W. C.: Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York (1983) 77
- [12] Talukdar, S. N., Camponogara, E.: Network Control as a Distributed, Dynamic Game. Proceedings of the 34th Hawaii International Conference on System Sciences. IEEE Computer Society (2001) (*Best Paper Award in the Complex Systems Track*) 74

Towards a Methodology for Experiments with Autonomous Agents

Luis Antunes and Helder Coelho

Faculdade de Ciências, Universidade de Lisboa, Portugal
{xarax,hcoelho}@di.fc.ul.pt

Abstract. Experimental methodologies are harder to apply when self-motivated agents are involved, especially when the issue of decision gains its due relevance in their model. Traditional experimentation has to give way to exploratory simulation, to bring insights into the design issues, not only of the agents, but of the experiment as well. The role of its designer cannot be ignored, at the risk of achieving only obvious, predictable conclusions. We propose to bring the designer *into* the experiment. We use the findings of extensive experimentation to compare current experimental methodologies in what concerns evaluation.¹

1 Context

Agents can be seen as unwanting actors, but gain additional technological interest and use when they have their own motivations, and are left for autonomous labour. But no-one is completely assured that a program does the “right thing,” or all faulty behaviours are absent. If agents are to be used by someone, *trust* is the key issue. But, how can we trust an agent that pursues its own agenda to accomplish some goals of ours [3]?

Autonomy deals with the agents’ freedom of choice, and choice leads to the agents’ behaviour through specific phases in the decision process. Unlike BDI (beliefs-desires-intentions) models, where the stress is given on the technical issues dealing with the agents pro-attitudes (what can be achieved, how can it be done), in BVG (beliefs-values-goals) multi-dimensional models, the emphasis is given on choice machinery, through explicit preferences. Choice is about *which* goals to pursue (or, where do the goals come from), and how the agent *prefers* to pursue them (or, which options the agent wants to pick).

The central question is the evaluation of the quality of decision. If the agent aims at optimising this measure (which may be multi-dimensional), why does s/he not use it for the decision in the first place? And, should this measure be unidimensional, does it amount to a utility function (which would configure the “totitarian” view: maximising expected utility as the *sole* motivation of the agent)? This view, however discredited since the times of the foundation

¹ Longer version in Lindemann, Moldt, Paolucci and Yu, *International Workshop on Regulated Agent-Based Social Systems: Theory and Applications (RASTA'02)*, Universität Hamburg, FBI-HH-M-318/02, July 2002.

of artificial intelligence [13], still prevails in many approaches, even through economics or the social sciences (cf. [8]).

In this paper we readdress the issue of principled experimentation involving self-motivated agents. The sense of discomfort borne by reductionist approaches undermines the conclusions of the (ever-so-few) experiments carried out in the field. Hence, we issue a contribution for the synthesis of a method for systemic experimental integration.

In the next section, we summarise the choice framework we adopt, and state the problem of evaluating the results of the agents' decisions. In section 3, we briefly compare two experimental methodologies. We conclude that none completely solves the issue, and note the similarities between evaluation of the results by the designer, and adaptation by the agents. In section 4 we propose two answers for the issue of assessing experimental results. The combination of both approaches, bringing the designer's insights and conjectures into the setting of experiments, fits well into the notion of pursuing exploratory simulation. In the last two sections, we briefly present some experimental results, and finally conclude by exalting the advantages of explicitly connecting the experimenter's and the agents' evaluative dimensions.

2 Choice and Evaluation

The role of value as a mental attitude towards decision is to provide a reference framework to represent agent's preference during deliberation (the pondering of options candidate to contribute to a selected goal). In the BVG choice framework, the agent's system of values evolves as a consequence of the agent's assessment of the results of previous decisions. Decisions are evaluated against certain dimensions (that could be the same previously used for the decision or not), and this assessment is fed back into the agent's mind, by adapting the mechanisms associated with choice. This is another point that escapes the traditional utilitarian view, where the world (and so the agent) is static and known. BVG agents can adapt to an environment where everything changes, including the agent's own preferences (for instance as a result of interactions). This is especially important in a multi-agent environment, since the agents are autonomous, and so potentially sources of change and novelty.

The evaluation of the results of our evaluations becomes a central issue, and this question directly points to the difficulties in assessing the results of experiments. We would need meta-values to evaluate those results. But if those "higher values" exist (and so they are the important ones) why not use them for decision?

When tackling the issue of choice, the formulation of hypotheses and experimental predictions becomes delicate. If the designer tells the agent how to choose, how can he not know exactly how the agent will choose? To formulate experimental predictions and then evaluate to what extent they are fulfilled becomes a spurious game: it amounts to perform calculations about knowledge and reasons, and not to judge to what extent those reasons are the best reasons, and

correctly generate the choices. We return to technical reasons for behaviour, in detriment of the will and the preferences of the agent.

By situating the agent in an environment with other agents, autonomy becomes a key ingredient, to be used with care and balance. The duality of value sets becomes a necessity, as agents cannot access values at the macro level, made judiciously coincide with the designer values. The answer is the designer, and the problem is methodological. The update mechanism provides a way to put to test this liaison between agent and designer.

The designer's model of choice cannot be the model of perfect choice against which the whole world is to be evaluated. It is our strong conviction that the perfect choice does not exist. It is a model of choice to be compared to another one, by using criteria that in turn may not be perfect.

3 Experimental Methodologies

When Herbert Simon received his Turing award, back in 1973, he felt the need to postulate “artificial intelligence is an empirical science.” The duality science/engineering was always a mark of artificial intelligence, so that claim is neither empty nor innocent. Since that time, there has been an ever-increasing effort in artificial intelligence and computer science to experimentally validate the proclaimed results.

3.1 A Methodology for Principled Experimentation

Cohen's MAD (modelling, analysis and design) methodology [6] is further expanded in [7], where he states the fundamental question to link this methodology to the concept of experiment with self-motivated agents: “What are the criteria of good performance? Who defines these criteria?”

The answer to these questions is an invitation to consider rationality itself, and its criteria. The fact that rationality is situated most times imposes the adoption of *ad hoc* decision criteria. But the evaluation of the results of experiments is not intrinsically different from the evaluation the agents conduct of their own performance (and upon which they base their adaptation). In particular, there was always a designer defining both types of evaluation. So the question comes natural: why would the design of some component be “better” than the other (and support one “right thing”)? Most times there is no reason at all, and the designer uses the same criteria (the same “rationality”) either for the agent's adaptation or for the evaluation of its performance.

3.2 A Methodology from the Social Sciences

Computational simulation is methodologically appropriate when a social phenomenon is not directly accessible [11]. A new methodology can be synthesised, and designated “exploratory simulation” [8]. The prescriptive character (exploration) cannot be simplistically reduced to optimisation, such as the descriptive character is not a simple reproduction of the real social phenomena.

A recent methodology for computational simulation is the one proposed by Gilbert [10]. This is not far from MAD, but there are fundamental differences: in MAD there is no return to the original phenomenon, the emphasis is still on the system, and the confrontation of the model with reality is done once and for all, and represented by causal relations. All the validation is done at the level of the model, and the journey back to reality is done already in generalisation. In some way, that difference is acceptable, since the object of the disciplines is also different. But it is Cohen himself who asks for more realism in experimentation, and his methodology fails in that involvement with reality.

But, Is it possible to do better? Is the validation step in Gilbert’s methodology a realist one? Or can we only compare models with other models and never with reality? If our computational model produces results that are adequate to what is known about the real phenomenon, can we say that our model is validated, or does that depend on the source of knowledge about that phenomenon? Isn’t that knowledge obtained also from models? For instance, from results of questionnaires filled by a representative sample of the population – where is the real phenomenon here? Which of the models is then the correct one?

The answer could be in [14]: social sciences have an exploratory purpose, but also a predictive and even prescriptive one. Before we conduct simulations that allow predictions and prescriptions, it is necessary to understand the phenomena, and for that one uses exploratory simulation, the exploration of simulated (small) worlds. But when we do prediction, the real world gives the answer about the validity of the model.

Once collected the results of simulations, they have to be confronted with the phenomenon, for validation. But this confrontation is no more than analysis. With the model of the phenomenon to address and the model of the data to collect, we have again a simplification of the problem, and the question of interpretation returns. It certainly isn’t possible to suppress the role of the researcher, ultimate interpreter of all experiments, be it classical or simulation.

4 Two Answers

In this section we will present two different answers for the problem of analysing (and afterwards, generalising) the results of the experimentation, which we have already argued to have quite a strong connection to the problem of *improving* the agents performance as a result of evaluation of the previous choices.

The explicit consideration of the relevant evaluative dimensions in decision situations can arguably provide a bridge between the agent’s and the experiments designer’s mind. In a multi-dimensional choice model, the agent’s choice mechanisms are fed back with a set of multi-dimensional update values. These dimensions may or not be the same that were used to make the decision in the first place. If these dimensions should be different, we can identify the ones that were used for decision with the interests of the agent, and the ones used for update with the interests of the designer. And moreover, we have an explicit link between the two sets of interests. So, the designer is no longer left for purely

subjective guessing of what might be happening, confronted with the infinite regress of ever more challenging choices. S/he can explore the liaisons provided by this choice framework, and experiment with different sets of preferences (desired results), both of hers and of the agents.

4.1 Positivism: Means-Ends Analysis in a Layered Mind

We can postulate a positivist (optimistic) position by basing our ultimate evaluations on a pre-conceived ontology of such deemed relevant dimensions (or values). Having those as a top-level reference, the designer's efforts can concentrate on the appropriate models, techniques and mechanisms to achieve the best possible performance as measured along those dimensions.

It seems that all that remains is then optimisation along the desired dimensions, but even in that restrained view we have to acknowledge that it does not mean that all problems are now solved. Chess is a domain where information is perfect and the number of possibilities is limited, and even so it was not (will it ever be?) solved.

Alternatively, the designer can be interested in evaluating how the agents perform in the absence of the knowledge of what dimensions are to be optimised. In this case, several models can be used, and the links to the designer's mind can still be expressed in the terms described above.

The key idea is to approximate the states that the agent wishes to achieve to those that it believes are currently valid. This amounts to performing a complex form of means-ends analysis, in which the agent's sociality is an issue, but necessarily one in which the agent does not have any perception about the meta-values involved. Because that would reinstate the infinite regression problem.

The external evaluation problem can be represented in terms as complex as the experiment designer thinks appropriate. In a BDI-like logical approach, evaluation can be as simple as answering the question "were the desired states achieved or not?," or as complicated as the designer desires and the decision framework allows to represent. The choice mechanisms update becomes an important issue, for they are trusted to generate the desired approximation between the agent's performance (in whichever terms) and the desired one.

Interesting new architectural features recently introduced by Castelfranchi [4] can come to the aid of the task of unveiling these ultimate aims that justify behaviour. Castelfranchi acknowledges a problem for the theory of cognitive agents: "*how to reconcile the 'external' teleology of behaviour with the 'internal' teleology governing it; how to reconcile intentionality, deliberation, and planning with playing social functions and contributing to the social order.*" [4, page 6, original italics].

Castelfranchi defends *reinforcement* as a kind of internal natural selection, the selection of an item (e.g. a habit) directly within the entity, through the operation of some internal choice criterion. And so, Castelfranchi proposes the notion of *learning*, in particular, reinforcement learning in cognitive, deliberative agents. This could be realised in a hybrid layered architecture, but not one where reactive behaviours compete against a declarative component. The idea is to have

“a number of low-level (automatic, reactive, merely associative) mechanisms operate *upon* the layer of high cognitive representations” [4, page 22, original italics].

Damasio’s [9] somatic markers, and consequent mental reactions of attraction or repulsion, serve to constrain high level explicit mental representations. This mental architecture can do without the necessity of an infinite recursion of meta-levels, goals and meta-goals, decisions about preferences and decisions. In this meta-level layer there could be no explicit goals, but only simple procedures, functionally teleological automatisms.

In the context of our ontology of values, the notion of attraction/repulse could correspond to the top level of the hierarchy, that is, the ultimate value to satisfy. Optimisation of some function, manipulation and elaboration of symbolic representations (such as goals), pre-programmed (functional) reactivity to stimuli, are three faces of the same notion of ending up the regress of motivations (and so of evaluations over experiments). This regress of abstract motivations can only be stopped by grounding the ultimate reason for choice in concrete concepts, coming from *embodied* minds.

4.2 Relativism: Extended MAD, Exploratory Simulation

There are some problems in the application of MAD methodology to decision situations. MAD is heavily based on hypotheses formulation and predictions about systems behaviour, and posterior confrontation with experimental observations. An alternative could be conjectures-led exploratory simulation.

The issues raised by the application of MAD deal with meta-evaluation of behaviours (and so, of underlying models). We have proposed an extension to MAD that concerns correction between the diverse levels of specification (from informal descriptions to implemented systems, passing by intermediate levels of more or less formal specification). This extension is based on the realisation of the double role of the observer of a situation (which we could translate here into the role of the agent and that of the designer).

The central point is to evaluate the results of agent’s decisions. Since the agent is autonomous and has its own reasons for behaviour, how can the designer dispute its choices? A possible answer is that the designer is not interested in allowing the agent to use the best set of reasons. In this case what is being tested is not the agent, but what the designer thinks are the best reasons. The choice model to be tested is not the one of the agent, and the consequences may be dramatic in open societies.

In BVG, the feedback of such evaluative information can be explicitly used to alter the agents choice model, but also to model the mind of the designer. So, agents and designer can share the same terms in which the preferences can be expressed, and this eases up validation. The model of choice is not the perfect reference against which the world must be evaluated (such a model cannot exist), but just a model to be compared to another one, by using criteria that again might not be perfect.

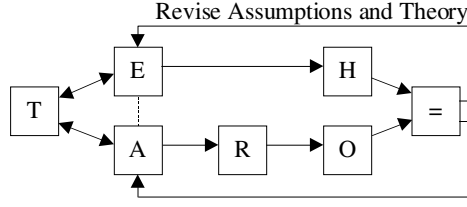


Fig. 1. Construction of theories. An existing theory (T) is translated in a set of assumptions (A) represented by a program and an explanation (E) that expresses the theory in terms of the program. The generation of hypotheses (H) from (E) and the comparison with observations (O) of runs (R) of the program allows both (A) and (E) to be revised. If finally (H) and (O) correspond, then (A), (E) and (H) can be fed back into a new revised theory (T) that can be applied to a real target (from [11])

This seems to amount to an infinite regress. If we provide a choice model of some designer, it is surely possible to replicate it in the choice model of an agent, given enough liberty degrees to allow the update mechanisms to act. But what does that tell us? Nothing we couldn't predict from the first instant, since it would suffice that the designer's model would be used *in* the agent. In truth, to establish a realist experiment, the designer's choice model would itself be subject to continuous evolution to represent his/her choices (since it is immersed in a complex dynamical world). And the agent's model, with its update mechanisms, would be "following" the other, as well as it could. But then, what about the designer's model, what does it evolve to follow? Which other choice model can this model be emulating, and how can it be represented?

Evaluation is harder for choice, for a number of reasons: choice is always situated and individual, and it is not prone to generalisations; it is not possible to establish criteria to compare choices that do not challenge the choice criteria themselves; the adaptation of the choice mechanisms to an evaluation criteria appears not as a test to its adaptation capabilities, but rather as a direct confrontation of the choices.

Who should tell if our choices are good or not, based on which criteria can s/he do it, why would we accept those criteria, and if we accept them and start making choices by them, how can we evaluate them afterwards? By transposing this argument to experimental methodology, we see the difficulty in its application, for the decisive step is compromised by this opposition between triviality (when we use the same criteria to choose and to evaluate choices) and infinite and inevitable regression (that we have just described).

Despite all this, the agent *cannot* be impotent, prevented from improving its choices. Certainly, human agents are not, since they keep choosing better (but not every time), learn from their mistakes, have better and better performances, not only in terms of some external opinion, but also according to their own. As a step forward, and out of this uncomfortable situation, we can also consider

that the agent has two different rationalities, one for choice, another for its evaluation and subsequent improvement. One possible reason for such a design could be the complexity of the improvement function be so demanding that its use for common choices would not be justified.

To inform this choice evaluation function, we can envisage three candidates: (i) a higher value, or some specialist's opinion, be it (ii) some individual, or (iii) some aggregate, representing a prototype or group.

The first, we have already described in detail in the previous subsection: some higher value, at a top position in a ontological hierarchy of value. In a context of social games of life and death, survival could be a good candidate for such a value. As would some more abstract dimension of goodness or righteousness of a decision. That is, the unjustifiable (or irreducible) sensation that, all added up, the right (good, just) option is evident to the decider, even if all calculations show otherwise. This position is close to that of *moral imperative*, or *duty*. But this debate over whether all decisions must come from the agents pursuing their own interest has to be left for further studies.

The second follows Simon's idea for the evaluation of choice models: choices are compared to those made by a human specialist. While we want to verify if choices are the same or not, this idea seems easy to implement. But if we want to argue that the artificial model chooses *better* than the reference human, we return to the problem of deciding what 'better' means.

The third candidate is some measure obtained from an aggregation of agents which are similar to the agent or behaviour we want to study. We so want to compare choices made by an agent based on some model, with choices made by some group to be studied (empirically, in principle). In this way we test realistic applications of the model, but assuming the principle that the decider agent represents in some way the group to be studied.

4.3 Combining the Two Approaches

A recent methodological approach can help us out here [12]. The phases of construction of theories are depicted in figure 1. However, we envisage several problems in the application of this methodology. Up front, the obvious difficulties in the translation from (T) to (E) and from (T) to (A), the subjectivity in the selection of the set of results (R) and corresponding observations (O), the formulation of hypotheses (H) from (E) (as Einstein said: "no path leads from the experience to the theory"). The site of the experimenter becomes again central, which only reinforces the need of defining common ground between him/her and the mental content of the agents in the simulation.

Thereafter, the picture (as its congeners in [12]) gives further emphasis to the traditional forms of experimentation. But Hales himself admits experimentation in artificial societies demands for new methods, different from traditional induction and deduction. Like Axelrod says: "Simulation is a third form of making science. (...) While induction can be used to discover patterns in data, and deduction can be used to find consequences of assumptions, the modelling of simulations can be used as an aid to intuition." [2, page 24]

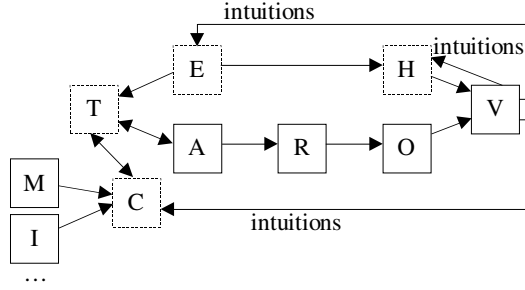


Fig. 2. Exploratory simulation. A theory (T) is being built from a set of conjectures (C), and in terms of the explanations (E) that it can generate, and hypotheses (H) it can produce. Conjectures (C) come out of the current state of the theory (T), and also out of metaphors (M) and intuitions (I) used by the designer. Results (V) of evaluating observations (O) of runs (R) of the program that represents assumptions (A) are used to generate new explanations (E), which allow the reformulation of the theory (T)

This is the line of reasoning already defended in [8]: to observe theoretical models running in an experimentation test bed, it is ‘exploratory simulation.’ The difficulties in concretising the verification process (=) in figure 1 are even more stressed in [5]: the goal of these simulation models is not to make predictions, but to obtain more knowledge and insight.

This amounts to radically changing the drawing of figure 1. The theory is not necessarily the starting point, and the construction of explanations can be made autonomously, as well as the formulation of hypotheses. Both can even result from the application of the model, instead of being used for its evaluation. According to Casti [5], model validation is done qualitatively, recurring to intuitions of human specialists. These can seldom predict what occurs in simulations, but they are experts at explaining the occurrences.

Figure 2 is inspired in the scheme of explanation discovery of [12], and results from the synthesis of the scheme for construction of theories of figure 1, and a model of simulations validation. The whole picture should be read at the light of [5], that is, the role of the experimenter and his/her intuition is ineluctable. Issues of translation, retroversion and their validation are important, and involve the experimenter. On the other hand, Hales’ (=) is substituted by an evaluation machinery (V), that can be designed around values. Here, the link between agents and experimenter can be enhanced by BVG choice framework.

One of the key points of the difference between figures 1 and 2 is the fact that theories, explanations and hypotheses are being constructed, and not only given and tested. Simulation is precisely the search for theories and hypotheses. These come from conjectures, through metaphors, intuitions, etc. Even evaluation needs intuitions from the designer to lead to new hypotheses and explanations. This process allows the agent’s choices to approximate the model that is

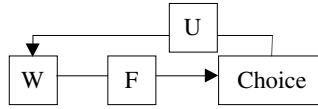


Fig. 3. Choice and update in the BVG architecture

provided as reference. Perhaps this model is not as accurate as it should be, but it can always be replaced by another, and the whole process of simulation can provide insights into what this other model should be.

The move from BDI to BVG was driven by a concern with choice. But to tune up the architecture, experimentation is called for. BVG is more adaptive to dynamic situations than BDI, and this presents new demands on the experimental methodology. In BVG (see figure 3), choice is based on the agent's values (W), and performed by a function F . F returns a real value that momentarily serialises the alternatives at the time of decision. The agent's system of values is updated by a function U that uses multidimensional assessments of the results of previous decisions. We can represent the designer's choice model by taking these latter dimensions as a new set of values, W' . Mechanisms F and U provide explicit means for drawing the link between the agent's (choosing) mind and the designer's experimental questions, thus transporting the designer into the (terms of the) experiment. This is accomplished by relating the backwards arrows in both figures (2 and 3). We superimpose the scheme of the agent on the scheme of the experiment.

5 Assessment of Experimental Results

This concern with experimental validation is an important keynote in the BVG architecture. Initially we reproduced (using Swarm) the results of Axelrod's "model of tributes," because of the simplicity of the underlying decision model [1]. Through principled exploration of the decision issues, we uncovered certain previously unidentified features of the model. But the rather rigid character of the decision problems would not allow the model to show its full worth.

In other experiments, agents selected from a pool of options, in order to satisfy some (value-characterised) goals. This introduced new issues in the architecture, such as non-transitivity in choice, the adoption of goals and of values, non-linear adaptation, the confront between adaptation based on one or multiple evaluations of the consequences of decisions. We provide some hints into the most interesting results we have found.

In a series of runs, we included in F a component that subverts transitivity in the choice function: the same option can rise different expectations (and decisions) in different agents. A new value was incorporated, to account for the effect of surprise that a particular value can raise, causing different evaluations (of attraction and of repulse).

The perils of subverting transitivity are serious. It amounts to withdrawing the golden rule of classical utility, that “all else being equal” we will prefer the better option. However, we sustain that it is not necessarily irrational (sometimes) not to do so. We have all done that in some circumstances. The results of the simulations concerning this effect of surprise were very encouraging. Moreover, the agent’s choices remained stable with this interference. The agent does not lose sense of what its preferences are, and what its rationality determines. It acts as if it allowed itself a break, in personal indulgence.

In other runs, we explored the role of values in regulating agent interactions, for instance, goal adoption. We found that when we increase the heterogeneity of the population in terms of values (of opposite sign, say), we note changes in the choices made, but neither radical, neither significant, and this is a surprising and interesting fact. The explanation is the “normalising” force of the multiple values and their diffusion. An agent with one or another different value still remains in the same world, sharing the same information, exchanging goals with the same agents. The social ends up imposing itself.

What is even more surprising is that this force is not so overwhelming that all agents would have exactly the same preferences. So many things are alike in the several agents, that only the richness of the model of decision, allied to their particular life stories, avoids that phenomenon.

The model of decision based on multiple values, with complex update rules, and rules for information exchange and goal adoption, presents a good support for decision making in a complex and dynamic world. It allows for a rich range of behaviours that escapes from directed and excessive optimisation (in terms of utilitarian rationality, it allows for “bad” decisions), but does not degenerate in pure randomness, or nonsense (irrationality). It also permits diversity of attitudes in the several agents, and adaptation of choices to a dynamic reality, and with (un)known information.

6 Conclusions

No prescribed methodology will ever be perfect for all situations. Our aim here is to draw attention to the role of the designer in any experiment, and also to the usually underaddressed issue of choice in the agent’s architecture. Having a value-based choice model at our hands as a means to consider self-motivated autonomous agents, these two ideas add up to provide a complete decision framework, where the designer is brought into the experiment, through the use of common terms with the deciding agents. This is a step away from reductionism, and towards a holistic attitude in agent experimentation.

References

- [1] Robert Axelrod. A model of the emergence of new political actors. In *Artificial Societies – The Computer Simulation of Social Life*. UCL Press, 1995. 94
- [2] Robert Axelrod. Advancing the art of simulation in the social sciences. In *Simulating Social Phenomena*, volume 456 of *LNEMS*. Springer, 1997. 92
- [3] Cristiano Castelfranchi. Guarantees for autonomy in cognitive agent architecture. In *Intelligent Agents: agent theories, architectures, and languages, Proc. of ATAL’94*, volume 890 of *LNAI*. Springer, 1995. 85
- [4] Cristiano Castelfranchi. The theory of social functions: challenges for computational social science and multi-agent learning. *Journal of Cognitive Systems Research*, 2, 2001. 89, 90
- [5] John L. Casti. Would-be business worlds. *Complexity*, 6(2), 2001. 93
- [6] Paul R. Cohen. A survey of the eighth national conf. on AI: Pulling together or pulling apart? *AI Magazine*, 12(1):16–41, 1991. 87
- [7] Paul R. Cohen. *Empirical Methods for AI*. The MIT Press, 1995. 87
- [8] Rosaria Conte and Nigel Gilbert. Introduction: computer simulation for social theory. In *Artificial Societies: the computer simulation of social life*. UCL Press, 1995. 86, 87, 93
- [9] António Damásio. *Descartes’ error*. Putnam’s sons, New York, 1994. 90
- [10] Nigel Gilbert. Models, processes and algorithms: Towards a simulation toolkit. In *Tools and Techniques for Social Science Simulation*. Physica-Verlag, 2000. 88
- [11] Nigel Gilbert and Jim Doran, editors. *Simulating Societies: the computer simulation of social phenomena*. UCL Press, London, 1994. 87
- [12] David Hales. *Tag Based Co-operation in Artificial Societies*. PhD thesis, Univ. Essex, 2001. 92, 93
- [13] Herbert A. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 69:99–118, Feb. 1955. 86
- [14] Klaus G. Troitzsch. Social science simulation – origins, prospects, purposes. In *Simulating Social Phenomena*, volume 456 of *LNEMS*. Springer, 1997. 88

How Planning Becomes Improvisation? – A Constraint Based Approach for Director Agents in Improvisational Systems

Márcia Cristina Moraes^{1,2} and Antônio Carlos da Rocha Costa^{1,3}

¹PPGC □ Universidade Federal do Rio Grande do Sul, Av. Bento Gonçalves 9500
Bloco IV, 91501-970, Porto Alegre, Brazil

mmoraes@inf.ufrgs.br

²FACIN □ Pontifícia Universidade Católica do Rio Grande do Sul, Av. Ipiranga
6681, Prédio 30, 90619-900, Porto Alegre, Brazil

mmoraes@inf.pucrs.br

³ESIN □ Universidade Católica de Pelotas, R. Felix da Cunha 412, 96010-000
Pelotas, Brazil

rocha@altas.ucpel.tche.br

Abstract. The aim of this paper is to explain how planning becomes improvisation for agents represented through animated characters that can interact with the user. Hayes-Roth and Doyle [10] proposed some changes in the view of intellectual skills traditionally studied as components of artificial intelligence. One of these changes is that planning becomes improvisation. They pointed out that like people in everyday life, animated characters rarely will have enough information, time, motivation, or control to plan and execute extended courses of behavior. Animated characters must improvise, engaging in flexible give-and-take interactions in the here-and-now. In this paper we present an approach to that change. We propose that planning can be understood as improvisation under external constraints. In order to show how this approach can be used, we present a multi-agent architecture for improvisational theater, focusing on the improvisational director's processes.

1 Introduction

According to Hayes-Roth and Doyle [10] animated characters may make use of many intellectual skills studied as components of artificial intelligence. But in those characters these skills have to be revised in order to make the intellectual capabilities broader, more flexible and more robust. Those authors suggest some changes for three traditional artificial intelligence components: planning becomes improvisation, learning becomes remembering and natural language processing becomes conversation. In this paper we are going to focus in one of those changes: planning becomes improvisation.

Planning is a classical area of study in Artificial Intelligence. In planning a traditional agent has to build and execute a complete course of action in order to complete a task. But animated characters, like people in everyday life, will rarely have enough information, time, motivation or control to plan and execute extended courses of behavior [10]. Brooks [7] argue that traditional Artificial Intelligence systems, which describe the world in terms of symbols (as typed, named individuals and their relationships), need more and more complexity in order to have and to maintain beliefs from partial views of a chaotic world. The world's observation is the best way to have this kind of beliefs because the world is always updated and has all the details that needed to be known. In other words, the agents' ability to plan in detail is limited by the complexity of the environment so it is better to have agents that use improvisation. Besides, as mentioned by Loyall [14] and Bates [6] to be believable characters have to make the user suspend his disbelief, but to do that, characters have to show coherent and not repetitive behaviors.

To understand how planning becomes improvisation in a complex and dynamic world, we propose that it is better to guide agents with abstract descriptions than enumerate all possible actions, so we see plans as agent's intentions. And these intentions are more specifically improvisations with constraint satisfaction. In this paper we present our ideas about intention, improvisation and constraint satisfaction and also a multi-agent architecture that uses those ideas to simulate the functions of one director and several actors in an improvisational performance. We focus on how improvisation as a constraint satisfaction process is used by the director to direct the actors.

This work is built on the authors' previous experiences with improvisational interface agents [16] [17] [18]. It advances our model of improvisational process and enhances our multi-agent system architecture by incorporating an improvisational director.

2 Two Approaches for Planning: Plan-as-Program and Plan-as-Intention

To show how planning becomes improvisation we have to consider that the real world is not static and previously known. Because they are dynamic, real environments may change while an agent is reasoning about how to achieve some goal, and these changes may undermine the assumptions upon which the agent's reasoning is based. Agents in real, dynamic environments need to be receptive to many potential goals, goals that do not typically arise in a neatly sequential fashion. Agents need to reason about their actions [21]. To do that they have to know when new facts and opportunities happened and they have to adapt their selves to the current situation.

Many authors [1] [2] [3] [4] [5] [26] have considered how agents can use the current situation in order to act. Their approaches are different from traditional planning and are related to improvisation. All authors agree that the approach of classical planning can't be applied to dynamic and complex environment.

From the point of view of planning there are two ways to see a plan. In classical planning a plan is viewed like a program and in an alternative approach a plan is viewed like an intention.

According to Pflieger and Hayes-Roth [22] in the plan-as-program view, a plan is an executable program consisting of primitive actions that an agent executes in order to act. Thus, planning is a type of automatic programming, and plan following simply consists of direct execution. The other view of plan is that a plan is a commitment to a goal that guides but does not uniquely determine the specific actions an agent executes. In this way the agent cannot directly execute its plans, but only can execute behaviors, each of which may be more or less consistent with its plans.

The plans as programs view has several limitations as inadequacy for algorithmically intractable problems; inadequacy for a world characterized by unpredictable events; requires that plans are too detailed and doesn't address the problem of relating the plan text to the concrete situation [4]. Plans as intentions overcome these limitations because a plan is a resource that guides, through abstract actions, what the agent has to do.

We are going to use the second approach of plans as intentions to show how planning becomes improvisation. To reach their intentions people can have some idea of what they have to do and this is related to several kinds of limitations and opportunities that we are going to call constraints. The agent has freedom to improvise his/her actions considering the constraints that are present at the moment. This vision indicates not only the agent's goals but also some set of possible behaviors to achieve those goals [4] [19] [20].

In our view an agent has some intention and this intention could be described like a script that gives some hints on what to do and how to do something to reach his/her intention. Those hints are abstract, describing general procedures to achieve an intention. The agent only can choose what to do and how to do when it is in some situation. An intention could be understood as a goal that is represented through a high-level script that is instantiated with concrete actions according to the current situation. This high-level script and the concrete actions are represented as improvisations that are accomplished through constraint satisfaction.

2.1 Intention Representation

The intentions representation is based on the production rules model. According to Stefik [25] each production rule has two parts, called the if-part and the then-part. The if part of a rule consists of conditions to be tested. If all the conditions in the if-part of a rule are true, the actions in the then-part of the rule are carried out. The then-part of these rules consist of actions to be taken. The fundamental difference between our representation and traditional production rules is that the actions in the then part are abstract behaviors, or high level actions, that are going to be transformed in primitive actions during the improvisation. We choose to use production rules model because we consider that it is the most appropriated representation for intentions considering the approach proposed by the Improvisational Theater.

3 Improvisation and Constraint Satisfaction

According to Frost and Yarrow [9] improvisation is the ability to use body, space, and all human resources to generate a physical expression coherent with an idea, a situation, a character or a text, doing that with spontaneity in response to immediate

stimuli from the environment and doing it considering surprise and without preconceptions.

Considering the definition of Frost and Yarrow, Chacra [8] also pointed out that improvisation and traditional theater are different poles of the same subject. The difference between these two poles is determined by degrees that make the theatrical presentation more or less formalized or improvised. If actors intend to use improvisation they explicitly are integrated in what is called Improvisational Theater. So, they don't prepare in advance all their actions and speeches, they consider the moment of spontaneity.

Agents that use improvisation have to consider all aspects described above for dramatic improvisation. This kind of agent, called improvisational agent is an animated character that has to adapt its behaviors, and possibly some goals, in order to act in a dynamic environment. We consider that the degrees pointed by Chacra [8] are constraints that will make possible improvisation through their satisfactions.

Several problems in Artificial Intelligence and other computer science areas can be viewed as special cases of the constraint satisfaction problem [13]. Constraints are a mathematical formalization of relationships that can occur among objects. For instance "near to" or "far from" are constraints that happen between objects in the real world. According to Marriott and Stuckey [15] the legal form and meaning of a constraint is specified by a constraint domain. In this way, constraints are written in a made up language with constants, functions and relations of restrictions. The constraint domain specifies the syntax of a restriction, that is, specifies the rules to create constraints in a certain domain. It details the constants, functions and restriction relations allowed, as well as how many arguments each function and relation should have and in which order they have to appear.

In our architecture there are two classes of constraints: restrictions of order and restrictions of behavior. In the restrictions of order class are all kinds of restrictions related to the order in which some content should be organized and presented. In the restrictions of behavior class are all kinds of constraints related to the process of selecting appropriate behaviors to perform. In the next sections we present our kinds of constraints, their domains and how they are applied in agents that improvise.

4 How Agents Are Going to Use This Kind of Improvisation

We are defining a multi-agent architecture based on Improvisational Theater that uses the ideas presented in the previous sections. Our architecture has one director and several actors that use improvisation instead of planning.

Each agent is organized around a meta-level architecture. The meta-level contains all processes related to the agents' cognitive capabilities and the level contains processes related to perception and action on the environment. In the director's case its environment is the several actors that it has to direct. The director has to interact with a human author to receive the knowledge to build a performance.

The main objective of the director, as in an improvised Theater, is to direct and to manage actors in an improvised way. In our case we say that our director is going to use improvised directions to do its job. On the other hand, the actors have to improvise their performance according to the directions received. These directions are

intentions and have some constraints that must be satisfied. To do its job the director also will have an intention that will describe its goals.

Both, the director and the actors are going to perform improvisation as constraint satisfaction process. In this way, the director is going to work with some constraints and the actors with others. This gives us two different kinds of improvisations. The first one is related to the processes involved in the director's activity and the second one is related to the improvisational performance of actors. Both processes are related and can be viewed as different levels of an improvisational performance. This kind of improvisation can be applied in several domains such as education, commercial web sites and entertainment.

The next sections show the roles of an improvisational director and describe how constraints are applied in two director's processes: knowledge acquisition and intentions building. With these two modules we can have an idea of how the director can obtain information and uses it to realize an improvised direction of its actors.

4.1 Director

According to Spolin [23] [24] improvisation is related to the intuitive and consequently to spontaneity. Spolin says that in the improvisational theater, the director and actors have to create an environment in which the intuitive can emerge and all of them are acting together to create an inspiring and creative experience. To do this he compares the process of improvisation with a game, where is a problem that must be solved considering unpredictable situations that occur in a dynamic environment.

It is important to say here that the notion of game and problem solving mentioned by Spolin is not the same one proposed by classical artificial intelligence. As Rich [22] explains classical artificial intelligence defined the problem of playing chess as a problem of moving around in a state space. By contrast, Spolin uses games as a way of interaction between people where no one knows what can happen and there aren't any rules to determine the game's course.

Viewing the improvisation as a problem to be solved considering the moment of spontaneity and involving us in a moving, changing world, Spolin [23] [24] explains the processes related to the director in the improvisational theater. The first process is that the director has to inform the problem to be solved to the actors. This is done informing scripts to the actors. But only the directions that leads to some action or dialog must be included in the scripts. The director has to give freedom for her/his actors, so they can perform spontaneously. The second process is that the director has to evaluate the actors after an acting problem has finished. Besides, the director can guide the actors when necessary. When some unexpected problem arrives the director can help the actors to find a solution for it.

Considering the roles that the director must play during an improvisational theater, we specify the four components of our director agent: knowledge acquisition, intentions building, evaluation and problem solving. Besides, we also describe below how the director coordinates these processes through its intention.

4.1.1 **Director’s Intention**

In order to coordinate these four components the director has its own intention. The schema of the director’s intention can be visualized in Fig. 1.

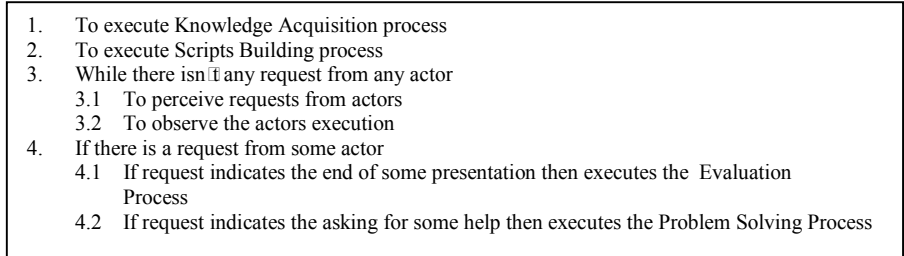


Fig. 1. The director’s intention

4.1.2 **Knowledge Acquisition**

In the knowledge acquisition process the author of a play has to give information about the play to the director. This information could be something like a sequence of contents that have to be presented and speeches related to the content. The sequence can be informed in complete, partial or any order at all. For instance the human author can say to the director that the actor has first to introduce himself saying some of the speeches, *"Hi! I’m Ana. I’m here to present Porto Alegre to you."* Or *"Hello! I’m Ana. And I’m here to talk to you about Porto Alegre."* Then the actor can choose between presenting the history of Porto Alegre City or presenting facts about Porto Alegre’s location. The human author also informs speeches related to these subjects. The last action is finishing the performance saying one of the speeches *"It was very nice to talk to you! See you another time"* or *"I hope you have enjoyed this presentation. Good bye"*. In the above example, the human author informed a partial order of the actions that an actor must perform. The human author has fixed the first and last actions and the actor has to choose the order of the intermediary actions.

In Fig. 2 we can see how the information follows in the sense of knowledge acquisition and intentions building.

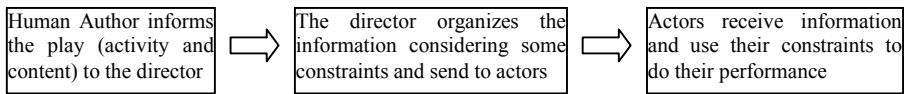


Fig. 2. Information follows in knowledge acquisition and intention building

It is important to notice that this is one of the ways in which the information follows. In other processes the actor can also send information to the director and the director to the human author.

As we mention in section 2.1 we can think about the components of traditional planning as something that agents can use to guide their course of action and not something that will be used to plan in advance their entire course of action. So the components precondition, action and effect can be seen as something involved in the organization of some kind of presentation or play. After receiving that information

from the human author the director organizes it as a partially ordered structure representing actions and use it to build the actors' dynamic intentions.

4.1.3 Intentions Building

In the author scripts building process the director has to use the knowledge about the play to build one problem, called here an intention, to each actor. As in the theater that intention is going to be informed as a script. That script will be a dynamic script because it doesn't dictate the actor's behavior and the actors are going to choose their performance in accordance with their environment in a certain moment during the execution. They are going to instantiate the script. In other words, the scripts provide classes of actions related to some activity and the actors have to choose which action to execute at each time, depending on the constraints related both to action and actor.

4.1.3.1 Director's Constraint to Build the Agent's Script

The director is also going to use improvisation to build an agent's script. The constraints that the director will follow are present in the constraint class named restrictions of order. This class is related to the ordering of content presentation. The kinds of constraints in this class are:

- Precondition □ preconditions related to the ordering of some execution.
- Effects □ what effects the execution of some activity will bring. The effect activation brings the satisfaction of a new precondition.
- Status of the script □ that indicates if the script is empty or not.

The constraint domain is composed by constants that indicates empty, not empty, none and end and constraint relations as equality ($=$), difference ($\neq$) and existence ($\exists$). Fig. 3 shows some examples of actions in the scripts building and their constraints.

Action	Example of Constraints
To look for activity whose precondition is none	status of the script is equal to empty
To store intermediate effect To store precedence in script To call behavior scheduler To store effect in script	status of the script is different from empty and there exists activity whose precedence is equal to the current effect
To attribute end to precedence	effect is equal to none

Fig. 3. Samples of actions and related constraints in the scripts building process

4.1.3.2 Director as Author

In our case the director is also an author, because the human author informs the activities with their partial or total precedence and content to the director. Then the director has to organize these activities in order to inform a specific actor what should be done. Sometimes the human author can leave open some order, for instance, there are three different places to talk about and there isn't any order among them. So the director can leave this order open to the actor or it can decide which order the talks should follow, thus completing the work of human author.

Besides guiding what the actor has to do the director can guide how the activity is going to be executed. The director also informs to the actor which classes of actions can be related to some content and the actor will choose, according with their restrictions, which action is the best to be executed in some moment.

Briefly, we are considering Hayes-Roth’s structures of personality and actions [11] [12]. So we have classes of actions that describe abstractly which actions could be executed. Each action is related to some personalities, moods, and verbal and physical behaviors. We are not going to discuss these structures here. The central idea is that the director informs classes of actions that can be executed in some situation and is the actor’s responsibility to choose which one will be executed. These classes of actions bring variation to the actors’s behaviors, in the sense that even when actors are in front of repetitive situations their moods and internal configurations will be different and so the behaviors.

4.1.3.3 Director Intentions Building Modules for Actors

Basically the director intention building is divided in two main modules. The first one is related to what the actors should do and is called activity scheduler. The second one is related to how the actors can perform their activities and is called behavior scheduler. Fig. 4 shows these two modules.

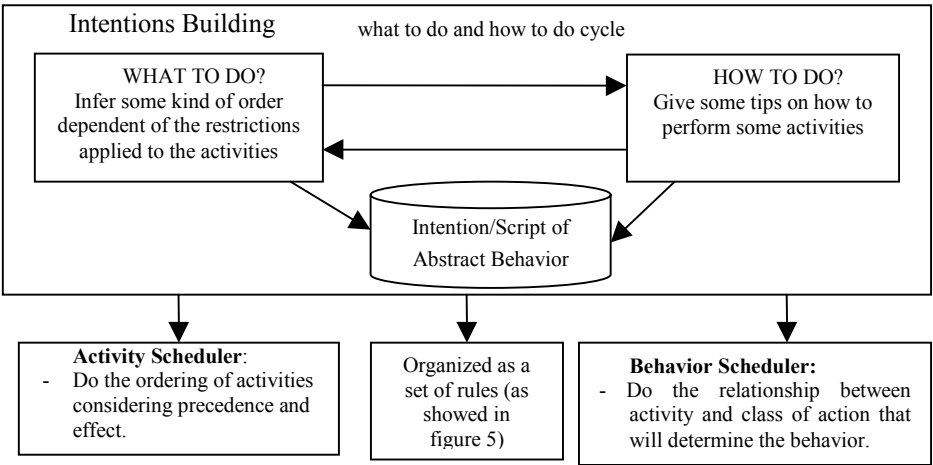


Fig. 4. Main modules in intentions building

The structure of an intention is showed in Fig. 5:

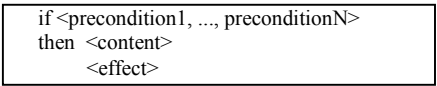


Fig. 5. Structure of an intention

<precondition> is the precondition of some activity. <content> is an indication that must be called an actor’s process to choose which content and specific action should be executed. The actor has to use its constraints to choose which content and

action to perform, because the director only informs some order and class of action to an actor. <effect> is the effect or effects related to the activity execution. The activation of some effect influences the satisfaction of one or more precondition.

In some cases, there could be more than one <precondition> that is satisfied in some moment. When an actor is executing its script and something like that occurs it will have to choose which is the best option according to its constraints. The actor has to improvise considering its constraints in a given situation. As we can see the intention or script of abstract behavior is an abstract description of what and how an actor is going to perform some activity. The director and the actors are working together to present some content for the user.

The activity scheduler³ and behavior scheduler³ algorithm can be visualized in Fig. 6 and Fig. 7.

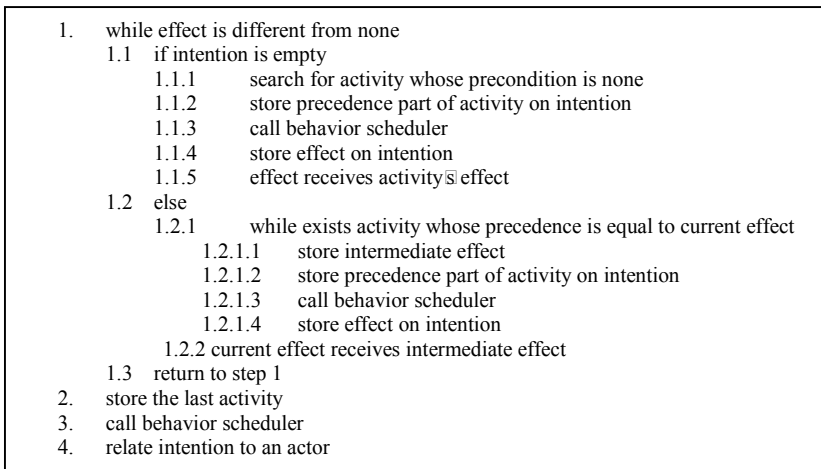


Fig. 6. Activity scheduler algorithm

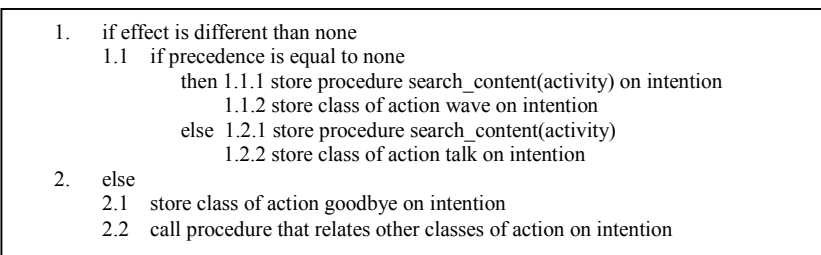


Fig. 7. Behavior scheduler³ algorithm

5 Conclusions

In this paper we present one approach for how planning becomes improvisation considering improvisation as a constraint satisfaction process. We have detailed this approach for two modules of a director agent. These two modules are responsible for

giving the instructions to the actors in the form of abstract actions that allow improvisation by the actors. These modules organization show how the director can make use of constraints to acquire knowledge and use it to build intentions to the actors that it directs. In this way the director executes part of what we call improvised direction. We are going to evaluate this approach using the criteria proposed by [9][23][24].

References

1. Agre, P. E.: The Dynamic Structure of Everyday Life. Phd Thesis, MIT Artificial Intelligence Laboratory, Technical Report 1085 (1988)
2. Agre, P. E.: Computation and Human Experience. Cambridge University Press (1997)
3. Agre, P. E., Chapman, D.: Pengi: An Implementation of a Theory of Activity. Sixth National Conference on Artificial Intelligence. Morgan Kaufmann Publishers. Vol. (1987) 268-272
4. Agre, P. E., Chapman, D.: What are plans for?. MIT A.I. Memo 1050. (1988)
5. Anderson, J. E.: Constraint Directed Improvisation for Everyday Activities. Phd Thesis University of Manitoba. (1995)
6. Bates, J.: The Nature of Characters in Interactive Worlds and The Oz Project. Carnegie Mellon University, Technical Report CMU-CS-92-200. (1992)
7. Brooks, R.: Elephants Don't Play Chess. Robotics and Autonomous Systems, Vol.6 (1990) 3-15
8. Chacra, S.: Natureza e Sentido da Improvis  o Teatral. Editora Perspectiva. (1983)
9. Frost, A., Yarrow, R.: Improvisation in Drama. MacMillan Education Ltd. (1990)
10. Hayes-Roth, B., Doyle, P.: Animated Characters. In: Autonomous Agents and Multi-Agent Systems, Vol. 1, Kluwer Academic Publishers, (1998) 195-230
11. Hayes-Roth, B., Rousseau, D.: A Social-Psychological Model for Synthetic Actors. Stanford University, Technical Report KSL 97-07. (1997)
12. Hayes-Roth, B., Rousseau, D.: Improvisational Synthetic Actors with Flexible Personalities. Stanford University, Technical Report KSL 97-10. (1997)
13. Kumar, V.: Algorithms for Constraint Satisfaction Problems: A Survey. AI Magazine. Spring 1992, (1992) 32-44
14. Loyall, B.: Believable Agents: Building Interactive Personalities. PhD Thesis Carnegie Mellon University, Technical Report CMU-CS-97-123. (1997)
15. Marriott, K., Stuckey, P. J.: Programming with Constraints: An Introduction. MIT Press, Cambridge Massachusetts (1998)
16. Moraes, M. C., Bertolotti, A. C., Costa, A. C. R.: Estudo e Avalia  o da Usabilidade de Agentes Improvisacionais de Interface. IV Workshop de Interfaces Homem-Computador. Brazil (2001)
17. Moraes, M. C., Bertolotti, A. C., Costa, A. C. R.: Evaluating Usability of SAGRES Virtual Museum Considering Ergonomic Aspects and Virtual Guides. 7th World Conference on Computers in Education: Networking the Learner. Denmark (2001)

18. Moraes, M. C., Bertoletti, A. C., Costa, A. C. R.: Virtual Guides to Assist Visitors in the SAGRES Virtual Museum. XIX Int. Conf. of Chilean Computer Science Society. (1999)
19. Pfleger, K., Hayes-Roth, B.: Using Abstract Plans to Guide Behavior. Stanford University, Technical Report KSL 98-02. (1998)
20. Pfleger, K., Hayes-Roth, B.: Plans Should Abstractly Describe Intended Behavior. In Alex Meystel, Jim Albus, and R. Quintero (eds.): Intelligent Systems: A Semiotic Perspective, Proceedings of the 1996 International Multidisciplinary Conference, Vol. 1 (1996) 29-34
21. Pollack, M. E.: The use of plans. Artificial Intelligence, Vol. 57. Elsevier Science Publishers (1992) 43-68
22. Rich, E.: Artificial Intelligence. McGraw-Hill Company: New York. (1983)
23. Spolin, V.: Improvisation for the Theater: A Handbook of Teaching and Directing Techniques. 1st edn. Northwestern University Press (1963)
24. Spolin, V.: Improvisation for the Theater. 3rd edn. Northwestern University Press (1999)
25. Stefik, M.: Introduction to Knowledge Systems. Morgan Kaufmann Publishers, Inc. San Francisco (1995)
26. Suchman, L. A.: Plans and Situated Actions: The problem of human machine communication. Cambridge: Cambridge University Press (1987)

Extending the Computational Study of Social Norms with a Systematic Model of Emotions

Ana L. C. Bazzan*, Diana F. Adamatti*, and Rafael H. Bordini

Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15064, 91501-970, Porto Alegre, Brazil
{bazzan,adamatti,bordini}@inf.ufrgs.br

Abstract. It is generally recognized that the use of emotions plays an important role in human interactions, for it leads to more flexible decision-making. In the present work, we extend the idea presented in a paper by Castelfranchi, Conte, and Paolucci, by employing a systematic and detailed model of emotion generation. A scenario is described in which agents that have various types of emotions make decisions regarding compliance with a norm. We compare our results with the ones achieved in previous simulations and we show that the use of emotions leads to a selective behavior which increases agent performance, considering that different types of emotions cause agents to have different acting priorities.

Keywords: Social norms, Emotions and personality, Multiagent-based simulation

1 Introduction

There are several arguments suggesting that emotion affects decision-making (see for instance [6] for a discussion on this issue). It is generally recognized that the benefits of humans having emotions encompass more flexible decision-making, as well as creativity. However, little work has focused on the investigation of interactions among social agents whose actions are somehow influenced by their current emotional setting.

Our overall goal is to create a framework to allow users to define the characteristics of a given interaction, the emotions agents can display, and how these affect their actions and interactions. In a previous paper [2], we have presented a prototype of such a framework using the Iterated Prisoner's Dilemma (IPD) scenario as a metaphor for interactions among agents. The present paper describes the use of a systematic model for generation of emotions applied to the scenario proposed in [4], in order to extend the study of the functions of social norms, such as the control of aggression among agents in a world where one's action influences the achievement of others' goals.

* Author partially supported by CNPq.

A review on the ideas motivating our work, namely, the specific scenario proposed by Castelfranchi *et al.* ([4],[5]) is presented in Section 2. The use of emotions in computing is given in Section 3. The proposed framework, its use in that scenario, and the results obtained by the modeling of agents with emotions are presented in Sections 4, 5, and 6, respectively. Section 7 concludes the paper and mentions future directions of the work.

2 The Scenario and Previous Results

The scenario proposed by Conte and Castelfranchi in [4] aimed at studying the effects of normative and non-normative strategies in the control of aggression among agents, and to explore the effects of the interaction between populations following different criteria for aggression control. The world as devised by the authors is a 10×10 square grid with randomly scattered food, in which agents can move in four directions only: up, down, left, and right. Various experiments were carried out (100 repetitions of a match consisting of 2000 time steps), in which the characteristics of agents were defined in different ways, as explained below. In all experiments, agents and food items are assigned locations at random. A cell cannot contain more than one object at a time, except when an agent is eating. At the beginning of each turn, every agent selects an action from its agenda according to the utility each will bring.

Eating is the most convenient choice for an agent. It begins at a given turn and may end two turns later if it is not interrupted by aggression. The eater's strength changes only when eating has been completed, that is, the eater's strength changes in a discrete way. When a food item has been consumed, it is immediately restored at a randomly chosen location in the grid. The second best choice of an agent is to move to a unoccupied grid position in which food has been seen. Agents can see food only within its territory, which consists of the four cells to which an agent can move in one step from its current location. The next choice is to move to a position where food has been smelt (if the agent does not see a food item, it can smell it within its extended neighborhood which consists of two steps in each direction from the agent's current location). Aggression is the next option available to an agent. If no food is available (either by seeing or smelling), an agent may attack an eating neighbor.

The outcome of an attack is determined by the agents' respective strengths (the stronger agent always wins). When the competitors are equally strong, the defender is the winner. The cost of aggression is equal to the cost of being attacked. Agents may be attacked by more than one agent at a time, in which case the victim's cost is multiplied by the number of aggressors. However, in this case only the strongest attacker earns the food item, while the others get nothing. Finally, the two last choices available to an agent are to move randomly (if no food is seen, smelt, and no attack is possible), and to pause (if even a random move is not possible).

Each match of 2000 time steps includes 50 agents and 25 food items with a nutritive value of 20 units each. Initially, agents' strength are set to 40 units.

During the match, agents have to pay the costs of their actions: 0 for pausing, 1 for moving to an adjacent cell, 4 for attacking or being attacked.

The main objective of the work on this scenario was the comparison of number of attacks and strength of agents when they follow social norms and when they act according to utilitarian rules. Three types of agents were proposed:

blind (B) whose aggression is constrained only by personal utility, with no reference to the eaters' strength. Blind agents attack eaters each time the cost of alternative action (as explained above) is higher. In other words, they are not aware of the eaters' strengths, nor of their own.

strategic (S) whose aggression is constrained by strategic reasoning. Strategic agents will only attack those eaters whose strength is not higher than their own. An eater's strength is perceptible one step away from the agent's current location.

normative (N) are the ones which follow a norm of precedence regarding agents that find food, thus becoming their owners. Each time agents or food are randomly allocated on the grid, the latter are assigned to the former when they happen to fall into the agents' territories. Owned food items are flagged and every player knows to whom they belong. Normative agents cannot attack possessors eating their own food.

From the results obtained in [4] and [5], normative strategies were found to reduce aggression, and also to afford the highest average strength and the lowest polarization of strength among the agents when compared to non-normative strategies.

Later on, the same scenario was used by Staller and Petta in [12], in order to investigate the interrelation between social norms and emotions. To this end, they adopted the same scenario of the study in [4], except for the simple action selection algorithm of the agents. In order to study the micro-level processes (as Staller and Petta put) underlying the interrelation between social norms and emotions, social simulations with more complex agents whose architecture includes emotion were conducted.

They conclude that emotions are crucial for the efficacy of norms, and that computational research has not yet paid adequate attention to this aspect, as we also pointed out in [3]. However, their claim is based on the idea of appraisal of concern-relevance only. The authors do not exactly use a model of emotions. Rather, they use some *ad hoc* variables measuring the intensity of states the agent is concerned with. For instance, depending on the average strength, the aggression level is modified.

In summary, in their approach, the act of obeying norms or not comes out because the intensities of aggression and strength were modified, not as a consequence of the type of emotions the agents have. This is the main motivation for our present work: we think their work can be improved by the use of a cognitive structure of emotions such as the one proposed in [10] and previously used by us in [2]. This may not only yield similar qualitative results, but also do so based on more sound grounds.

3 How Emotions Influence Decision and Behavior

The research on human emotions has a long tradition, both on a cognitive as well as on a physiological basis. See for instance the work in [8, 9] for the latter. However, we focus our work on the former, especially on the synergy between research in this field and decision-making, which, in turn, is relevant to many areas of artificial intelligence. In fact, a trend in the direction of agents displaying heterogeneous behaviors is reported in the literature (in quite distinct scenarios).

We do not attempt here to define what emotions are. As Picard [11] puts it, researchers in the area do not even agree on a definition. Rather, we concentrate on the cognitive and behavioral aspects of emotions as a computationally tractable model. This brings us to the need of stating the eliciting conditions for a particular emotion to arise, as well as the actions carried out as a consequence of it.

For our purposes, we find the so-called OCC theory by Ortony, Clore, and Collins [10] the most appropriate one. First, the authors are very concerned with issues dear to the Artificial Intelligence community; for instance, they believe that cooperative problem-solving systems must be able to reason about emotions. This is clearly an important research issue in Multi-Agent Systems as well. Second, it is a very pragmatic theory, based on grouping emotions by their eliciting conditions – events and their consequences, agents and their actions, or objects – which best suits a computational implementation.

The overall structure of the OCC model is based on emotion types or groups, which in their turn are based on how people perceive the world. They assume that there are three major perception aspects in the world: *events*, *agents*, and *objects*. Events are simply people’s construal about things that happened (not related to beliefs, nor necessarily with their possible causes). Objects are also a very straightforward level of perception. Finally, agents are both human and nonhuman beings, as well as inanimate objects (as explained next) or abstractions. In short, by focusing on events, objects, and agents, one is interested in their consequences, properties, and actions, respectively. In this model, another central idea is that emotions are valenced reactions; the intensity of the affective reactions determines whether or not they will be experienced as emotions. This points to the importance of framing the variables which determine the intensity of any reaction.

The structure of the OCC model based on types of emotions has three main branches, corresponding to the three ways people react to the world.

The first branch relates to emotions which are arising from aspects of **objects** such as liking, disliking, etc. This constitutes the single class in this branch, namely that called *attraction* which includes emotions such as love and hate.

The second branch relates to emotions which are consequences of **events**. Three classes appear here: *fortunes-of-others* (emotions *happy-for* and *gloating* or *Schadenfreude*); *prospect-based* (emotions *hope*, which can be either confirmed as *satisfaction* or disconfirmed as *disappointment*, and *fear*, which can be either confirmed as *fears-confirmed* or disconfirmed as *relief*); and *well-being* (emotions *joy* and *distress*).

The third branch is related to consequences of **agents**, namely the *attribution* class, comprising the following emotions: *pride* (person approves of self), *admiration* (person approves of other), *shame* (person disapproves of self), and *reproach* (person disapproves of other).

Finally, an additional class of emotions can be referred to as *compound*, since it focuses on both the action of an agent, and the resulting event and its consequences. This class is called *well-being/attribution compound*. It involves the emotions of gratification, remorse, gratitude, and anger.

Ortony *et al.* [10] recognize that this model is oversimplified, since in reality a person is likely to experience a mixture of emotions, especially when considering a situation from different perspectives at different moments. However, this co-occurrence would probably render the model computationally unfeasible. We believe that the model does have merits when one's goal is to conduct experiments on the effects of focusing on various aspects of an emotion-induced situation (as we do here), rather than to attempt at analyzing exactly what combinations or sequences of emotions could occur in given situations.

As for the intensity of emotions, which is important if one wants to implement a computational model, possibly relating certain variables to thresholds, Ortony *et al.* [10] distinguish between local and global variables affecting such intensity. Global variables affect all the types of emotions they have identified, and include: sense of reality (how much one believes the situation), proximity (how close one feels the situation), unexpectedness (how surprised one is by the situation), and arousal (how much one is aroused prior to the situation). On the other hand, local variables affect only specific groups of emotions. For example, the event-based emotions are affected by the desirability variable.

Some papers report previous usage of the OCC model. Elliott [7] has built the Affective Reasoner to map situations and agent state variables into a set of specific emotions, producing behaviors corresponding to them. Bates [1] has worked on micro-worlds that include moderately competent, emotional agents for the Oz Project.

4 The Proposed Framework

Our overall goal is to create a framework that allows users to define the characteristics of given interactions, the emotions agents can display, and how these affect their actions (hence those interactions). Such a framework is intended to be very general. That is, the user specifies the purpose of the simulation; the scenario for the interactions (which rules or norms agents follow when they meet); the environment (e.g., interactions happen among agents which belong to particular groups, agents are not attached to any group and meet randomly, interactions happen with respect to a spatial/geographical configuration); general parameters of the simulation (time frame, size of environment, etc.); the classification of any emotion that does not belong to the original OCC model, or the whole meaning of an emotion if it does not fit the model at all; and parameters related to each agent in the simulation (thresholds, types, etc.).

We base our framework on the OCC model for the reasons already explained. Additionally, this model can be translated into a rule-based system that generates cognitive-related emotions in an agent. We now explain how the rules look like in such a system. The IF part tests either the desirability (of a consequence of an event), or the praiseworthiness (of an action of an agent), or the appealingness (of an object). The THEN part sets the potential for generating an emotional state (e.g., a joyful state).

Let $A(p, o, t)$ be the appealingness of an object that a person p assigns to the object o at time t , $P_h(p, o, t)$ the potential to generate the state of hate, $G(vg_1, \dots, vg_n)$ a combination of global intensity variables, $I_h(p, o, t)$ the intensity of hate, $T_h(p, t)$ a threshold value, and f_h a function specific to hate. Then, a rule to generate a state of hate looks like:

IF $P_h(p, o, t) > T_h(p, t)$ THEN set $I_h(p, o, t) = P_h(p, o, t) - T_h(p, t)$
 ELSE set $I_h(p, o, t) = 0$

This rule is triggered by another one:

IF $A(p, o, t) > 0$ THEN set $P_h(p, o, t) = f_h(A(p, o, t), G)$

Ortony *et al.* [10] omit many of the details of implementation; a difficult issue might be to find appropriate functions for each emotion. It remains to be investigated whether general functions exist or whether they are domain-dependent. While we are studying these and other questions related to the implementation the OCC structure in a general framework, we are testing them on specific scenarios such as the one on the simulation of social norms.

5 Simulation of the Social Norms Scenario Using the OCC Model

The framework presented in the previous section was already used by us [2] in the simulation of a classic scenario concerning the Iterated Prisoner's Dilemma (IPD). It was shown that the use of emotions in such scenario increased the rate of cooperation.

We also maintain that the study of social norms is highly significant in the field of Multi-Agent Systems. Social norms form the basis of an approach to agent coordination (a central issues in MAS), facilitating decision-making by autonomous agents and avoiding unnecessary conflicts. The approach to agent coordination to which we refer is the one based on the social and cognitive sciences, rather than game theory. It is directly inspired by the role that social norms play in human societies. We argued in [3] that emotions are important in attaching agents to social norms, and that including emotions in agent architectures may help us come to a better understanding of how autonomous agents form and perpetuate conventions, which are essential for social behavior.

We have restricted the framework to agents displaying a single emotion (the predominant one), which is a consequence of the use of the OCC model. We start by identifying a set of emotions related to the scenario itself. As a first exercise, we have concentrated on typical emotions. Thus, agents may initially display

anger (a), joy (j), resentment (r), and pity (i). This way, almost all classes of the OCC model are represented within the set of emotions we use.

Let us now turn to the IF–THEN–ELSE rules derived for this specific scenario. All variables defined in Section 4 retain their meaning here. Beside those, we use below: $D(p, e, t)$ for the desirability that a person p assigns to event e at time t , $W(p, g, t)$ for the praiseworthiness that a person p assigns to agent g at time t , and $L(vl_1, \dots, vl_n)$ a combination of local intensity variables.

- Rules for joy:
 - IF $D(p, e, t) > 0$
 - THEN set $P_j(p, e, t) = f_j(D(p, e, t), G, L)$
 - function f_j returns value $T_j(p, t) + \epsilon$ (IF agent's strength $> average\_strength$)
 - IF $P_j(p, e, t) > T_j(p, t)$
 - THEN set $I_j(p, e, t) = P_j(p, e, t) - T_j(p, t)$
 - ELSE set $I_j(p, e, t) = 0$
- Rules for resentment:
 - IF $D(p, e, t) < 0$
 - THEN set $P_r(p, e, t) = f_r(D(p, e, t), G, L)$
 - function f_r returns value $T_r(p, t) + \epsilon$ (IF agent's strength = $average\_strength \pm \delta$ AND some agent is eating food which does not belong to it)
 - IF $P_r(p, e, t) > T_r(p, t)$
 - THEN set $I_r(p, e, t) = P_r(p, e, t) - T_r(p, t)$
 - ELSE set $I_r(p, e, t) = 0$
- Rules for pity:
 - IF $D(p, e, t) < 0$
 - THEN set $P_i(p, e, t) = f_i(D(p, e, t), G)$
 - function f_i returns value $T_i(p, t) + \epsilon$ (IF agent's strength = $average\_strength \pm \delta$ AND eater's strength $< average\_strength$)
 - IF $P_i(p, e, t) > T_i(p, t)$
 - THEN set $I_i(p, e, t) = P_i(p, e, t) - T_i(p, t)$
 - ELSE set $I_i(p, e, t) = 0$
- Rules for anger:
 - IF $(D(p, e, t) < 0 \text{ AND } W(p, g, t) < 0)$
 - THEN set $P_a(p, e, g, t) = f_a(D(p, e, t), W(p, g, t), G, L)$
 - function f_a returns value $T_a(p, t) + \epsilon$ (IF agent's suffered aggression $> average\_aggression$ OR agent's strength $< average\_strength$)
 - IF $P_a(p, e, g, t) > T_a(p, t)$
 - THEN set $I_a(p, e, g, t) = P_a(p, e, g, t) - T_a(p, t)$
 - ELSE set $I_a(p, e, g, t) = 0$

We now explain the rules. A joyful agent is defined as one whose strength is above the average (computed over all agents). There are two conditions under which an agent displays resentment: when its strength is close to the average (we can vary this threshold by changing the parameter δ), or when it perceives some agent eating other agents' food. The definition of a pitiful agent is as follows: its strength is close to the average, and it sees other agent(s) whose strength is

below the average. Finally, angry agents are those whose suffered aggression is higher than the average or whose strength is lower than the average strength.

Once fired, emotions have the following effects: joyful agents do not eat or attack (they only move at random); agents feeling resentment attack any agent eating others' food (regardless of strength); pitiful agents do not attack agents eating others' food if their strength is below the average; and angry agents never obey the norm: they attack any eating agent they perceive. Emotions are allowed to fire only after 200 steps of simulation, during which agents behave normatively.

6 Results and Comparison

Several comparisons can be made between the previous simulations of this scenario and the one we proposed here. Initially, we have replicated the experiments as proposed in [5]. We exclude the simulation of the strategic agents because they are of little significance for the comparison of different implementations of normative agents. Discrepancies can be explained by different interpretations of the scenario (an issue also reported in [12]).

Table 1 shows the results of our simulations. The first two lines show the replication of the results reported in [5] for blind and normative agents. The last line contains the results for the simulation with emotions. "Str." is the average strength over the 50 agents. "Dev." is the standard deviation regarding this average, and "Aggr." is the sum of aggressions suffered by the 50 agents. Each of these quantities is associated with a standard deviation ("dev.") computed over 100 repetitions of the simulation (for 2,000 time steps).

Table 1. The results of our simulations

Type	Str.	dev.	Dev.	dev.	Aggr.	dev.
blind	4135	191	3669	153	9299	449
normative	6757	29	132	20	2638	86
emotions	4307	223	173	72	2891	141

Next, an evaluation of the performance of our agents in the simulation can be made by comparing our results with those in [12]. Staller and Petta have reported a level of strength between 4624 and 5408 and a level of aggression between 3289 and 6902. In our experiments, the aggression decreased to 2891 (Table 1) since joyful agents never attack, and pitiful ones do this at a small rate. On the other hand, due to the design of the joyful agents (once they are satiated they neither attack nor eat), the strength is relatively low (4307). In fact, since in the beginning of the simulations there are many joyful agents (time steps 200 to 1200), the food items they "reject" account for the difference in strength.

Finally, the change in number of agents displaying each type of emotion within time is shown in Figure 1. It can be seen that in the beginning of the

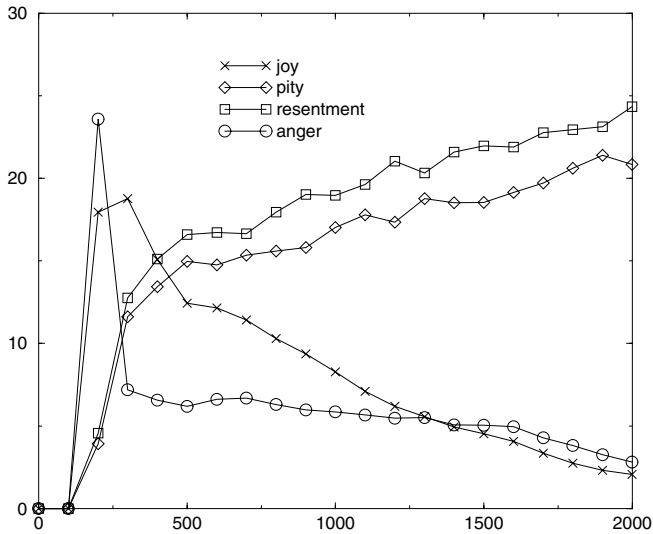


Fig. 1. Number of agents displaying each type of emotion within time

simulation (up to time step 200) there are only neutral agents because no emotion rule is allowed to fire. After this point, the numbers of angry and joyful agents are high, indicating a polarization of the group of agents. As time passes, resentment and pity increase as a reaction to this situation. This should be better understood in future extensions of the simulation, such as an accounting of the strength and aggression by type of agent.

7 Conclusion and Future Directions

The importance of emotions for human beings is that they yield more flexible decision-making. Staller and Petta have reached a similar conclusion (in [12], regarding the scenarios described in [4] and [5]). Their motivation was that agents do not spend all their time searching for food and attacking other agents. Following this argument, they were able to show that the performance of normative agents improves. We have shown that a similar study may be better conducted based on a formal theory of the cognitive structure of emotions such as that reported in [12]. Therefore, our aim in the present paper has been to carry this out and compare the results.

The present work contributes to the construction of a framework for simulating agents with emotions, by employing a scenario we regard as very important since it deals with social norms for agents. In order to implement that framework, we have first concentrated on finding a computationally tractable model that could account for the cognitive and behavioral aspects of emotions. For our purposes, we find the so-called OCC model [10] the most appropriate one, especially due to its pragmatism. To prove that the OCC model is suitable for

the social norm scenario, we have discussed its structure, as well as some issues which were missed in [10]. This paper also contributes to a deeper understanding of the OCC model regarding implementation details.

Our future plans include the construction of a general framework for simulating user-defined interactions. In order to achieve this, we are defining a series of primitives that can be combined by the users in constructing their own environments. These primitives comprise: the specification of the interactions, which actions to perform, when, and by whom, the wealth of agents, the types of emotions available (both those included in the OCC model and others), among other things. While no such domain-independent rules are available, the users are asked to construct the rules themselves, by entering the parameters for the primitives we have made available.

References

- [1] J. Bates. The role of emotion in believable agents. In *Communications of the ACM, Special Issue on Agents*, July, 1994. 112
- [2] A. L. Bazzan and R. H. Bordini. A framework for the simulation of agents with emotions: Report on experiments with the iterated prisoner's dilemma. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of The Fifth International Conference on Autonomous Agents (Agents 2001)*, 28 May – 1 June, Montreal, Canada, pages 292–299. ACM Press, 2001. 108, 110, 113
- [3] A. L. Bazzan, R. H. Bordini, and J. A. Campbell. Moral sentiments in multi-agent systems. In J. P. Müller, M. P. Singh, and A. S. Rao, editors, *Intelligent Agents V, Proceedings of ATAL-98*, number 1555 in LNAI, pages 113–131, Heidelberg, 1999. Springer-Verlag. 110, 113
- [4] C. Castelfranchi and R. Conte. Understanding the effects of norms in social groups through simulation. In G. N. Gilbert and R. Conte, editors, *Artificial Societies: the computer simulation of social life*, pages 252–267. UCL Press, London, 1995. 108, 109, 110, 116
- [5] C. Castelfranchi, R. Conte, and M. Paolucci. Normative reputation and the costs of compliance. *Journal of Artificial Societies and Social Simulation*, 1(3), 1998. <<http://www.soc.surrey.ac.uk/JASSS/1/3/3.html>>. 109, 110, 115, 116
- [6] A. Damasio. *Descartes's Error*. Avon, New York, 1994. 108
- [7] C. Elliott. Multi-media communication with emotion-driven believable agents. In *AAAI Spring Symposium on Believable Agents*, Stanford University in Palo Alto, California, March 21-23, 1994. 112
- [8] W. James. What is an emotion? *Mind*, 9:188–205, 1884. 111
- [9] W. James. *The Principles of Psychology*. Holt, New York, 1890. 111
- [10] A. Ortony, G. L. Clore, and A. Collins. *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge, UK, 1988. 110, 111, 112, 113, 116, 117
- [11] R. W. Picard. *Affective Computing*. The MIT Press, Cambridge, MA, 1997. 111
- [12] A. Staller and P. Petta. Introducing emotions into the computational study of social norms: A first evaluation. *Journal of Artificial Societies and Social Simulation*, 4(1), 2001. <<http://www.soc.surrey.ac.uk/JASSS/4/1/2.html>>. 110, 115, 116

A Model for the Structural, Functional, and Deontic Specification of Organizations in Multiagent Systems

Jomi Fred Hübner^{1*}, Jaime Simão Sichman^{1**}, and Olivier Boissier²

¹ LTI / EP / USP

Av. Prof. Luciano Gualberto, 158, trav. 3, 05508-900 São Paulo, SP

{jomi.hubner,jaime.sichman}@poli.usp.br

² SMA / SIMMO / ENSM.SE

158 Cours Fauriel

42023 Saint-Etienne Cedex, France

Olivier.Boissier@emse.fr

Abstract. A Multiagent System (MAS) that explicitly represents its organization normally focuses either on the functioning or the structure of this organization. However, addressing both aspects is a prolific approach when one wants to design or describe a MAS organization. The problem is to define these aspects in such a way that they can be both assembled in a single coherent specification. The $\mathcal{MOISE}^+$ model – described here through a soccer team example – intends to be a step in this direction since the organization is seen under three points of view: structural, functional, and deontic.

1 Introduction

The organizational specification of a Multiagent System (MAS) is useful to improve the efficiency of the system since the organization constrains the agents behaviors towards those that are socially intended: their global common purpose [8, 7]. Without some degree of organization, the agents' autonomy may lead the system to lose global congruence.

The models used to describe or project an organization are classically divided in two points of view: *agent* centered or *organization* centered [10]. While the former takes the agents as the engine for the organization formation, the latter sees the opposite direction: the organization exists *a priori* (defined by the designer or by the agents themselves) and the agents ought to follow it. In addition to this classification, we propose to group these organizational models in (i) those that stress the society's *global plans* (or tasks) [12, 11, 13] and (ii) those that have their focus on the society's *roles* [5, 6, 9]. The first group concern is the *functioning* of the organization, for instance, the specification of global

* Supported by FURB, Brazil; and CNPq, Brazil, grant 200695/01-0.

** Partially supported by CNPq, Brazil, grant 301041/95-4; and by CNPq/NSF PROTEM-CC MAPPEL project, grant 680033/99-8.

plans, policies to allocate tasks to agents, the coordination to execute a plan, and the quality (time consumption, resources usage, ...) of a plan. In this group, the global purposes are better achieved because the MAS has a kind of organizational memory where the best plans to achieve a global goal are stored. On the other hand, the second group deals with the specification of a more static aspect of the organization: its *structure*, i.e., the roles, the relations among them (e.g.: communication, authority), roles obligations and permissions, group of roles, etc. In these latter models, the global purpose is accomplished while the agents have to follow the obligations and permissions their roles entitle them. Thus we should state that organization models usually take into account either the functional (the first group) or structural (second group) dimension of the organization. However, in both groups the system may or may not have an explicit description of its organization that allows the organizational centered point of view.

The Fig. 1 briefly shows how an organization could explain or constrain the agents behavior in case we consider an organization as having both structural and functional dimensions. In this figure, it is supposed that a MAS has the purpose of maintaining its behavior in the set P , where P represents all behaviors which draw the MAS's global purposes. In the same figure, the set E represents all possible behaviors in the current environment. The organizational structure is formed, for example, by roles, groups, and links that constrain the agents behavior to those inside the set S , i.e., the set of possible behaviors ($E \cap S$) becomes closer to P . It is a matter of the agents, and not of the organization, to conduct their behaviors from a point in $((E \cap S) - P)$ to a point in P . In order to help the agents in this task, the functional dimension contains a set of global plans that has been proved efficient ways of turning the P behaviors active. For example, in a soccer team one can specify both the structure (defense group, attack group, each group with some roles) and the functioning of the team (e.g.: rehearsed plays, as a kind of predefined plans that was already validated).

If only the functional dimension is specified, the organization has nothing to "tell" to the agents when no plan can be performed (the set of possible behavior is outside the set F of the Fig. 1). Otherwise, if only the organizational structure is specified, the agents have to reason for a global plan every time they want to play together. Even with a smaller search space of possible plans, since the structure constrains the agents options, this may be a hard problem. Furthermore, the plans developed for a problem are lost, since there is no organizational memory to store these plans. Thus, in the context of some application domains, we hypothesize that if the organization model specifies both dimensions while

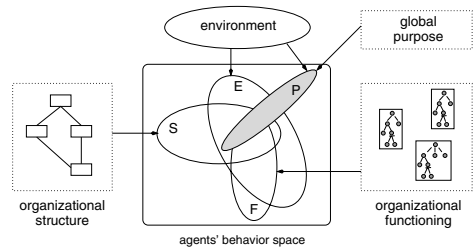


Fig. 1. The organization effects on a MAS

maintaining a suitable independence among them, then the MAS that follows such a model can be more effective in leading the group behavior to its purpose (Fig. 1). Another advantage of having both specifications is that the agents can reason about the others and their organization regarding these two dimensions in order to better interact with them (in the case, for example, of social reasoning).

A first attempt to join roles with plans is the MOISE (Model of Organization for multi-agent SystEms). The MOISE is structured along three levels: (i) the behaviors that an agent is responsible for when it adopts a role (*individual* level), (ii) the interconnections between roles (*social* level), and (iii) the aggregation of roles in large structures (*collective* level)[9]. The main shortcoming of MOISE, which motivates its extension, is the lack of the concept of an explicit global plan in the model and the strong dependence among the structure and the functioning.

This article sets out a proposal for an organizational model, called $\mathcal{MOISE}^+$, that considers the structure, the functioning, and the deontic relation among them to explain how a MAS organization collaborates for its purpose. The objective is an *organization centered model* where the first two dimensions can be specified almost *independently* of each other and after properly linked by the deontic dimension.

The organizational models that follow the organizational centered point of view (e.g., AALAADIN [5], MOISE [9]) usually are composed by two core notions: an Organizational Specification (OS) and an Organizational Entity (OE). An OE is a population of agents functioning under an OS. We can see an OE as an instance of an OS, i.e., agents playing roles defined in the OS (role instance), aggregated in groups instantiated from the OS groups, and behaving as normalized in the OS. Following this trend, a set of agents builds an OE by *adopting* an appropriate OS to easily achieve its purpose. An $\mathcal{MOISE}^+$ OS is formed by a Structural Specification (SS), a Functional Specification (FS), and a Deontic Specification (DS). Each of these specifications will be presented in the sequel.

2 Structural Specification

In $\mathcal{MOISE}^+$, as in MOISE, three main concepts, *roles*, *role relations*, and *groups*, are used to build, respectively, the individual, social, and collective structural levels of an organization. Furthermore, the MOISE original structural dimension is enriched with concepts such as inheritance, compatibility, cardinality, and sub-groups.

Individual level. The individual level is formed by the roles of the organization. A *role* means a set of constraints that an agent ought to follow when it accepts to enter a group playing that role. Following [2], these constraints are defined in two ways: in relation to other roles (in the collective structural level) and in a deontic relation to global plans (in the functional dimension).

In order to simplify the specification¹, like in object oriented (OO) terms, there is an *inheritance relation* among roles [6]. If a role ρ' inherits a role ρ (denoted by $\rho \sqsubset \rho'$), with $\rho \not\sqsupset \rho'$, ρ' receives some properties from ρ , and ρ' is a sub-role, or specialization, of ρ . In the definition of the role properties presented in the sequence, it will be precisely stated what one specialized role inherits from another role. For example, in the soccer domain, the attacker role has many properties of the player role ($\rho_{player} \sqsubset \rho_{attacker}$). It is also possible to state that a role specializes more than one role, i.e., a role can receive properties from more than one role. The set of all roles are denoted by $\mathcal{R}_{ss}$.

Following this OO inspiration, we can define an *abstract role* as a role that can not be played by any agent. It has just a specification purpose. The set of all abstract roles is denoted by $\mathcal{R}_{abs}$ ($\mathcal{R}_{abs} \subset \mathcal{R}_{ss}$). There is also a special abstract role ρ_{soc} where $\forall_{(\rho \in \mathcal{R}_{ss})} \rho_{soc} \sqsubset \rho$, trough the transitivity of $\sqsubset$, all other roles are specializations of it.

Social level. While the inheritance relation does not have a direct effect on the agents behavior, there are other kinds of relations among roles that directly constrain the agents. Those relations are called *links* [9] and are represented by the predicate $link(\rho_s, \rho_d, t)$ where ρ_s is the link source, ρ_d is the link destination, and $t \in \{acq, com, aut\}$ is the link type. In case the link type is *acq* (acquaintance), the agents playing the source role ρ_s are allowed to have a representation of the agents playing the destination role ρ_d (ρ_d agents, in short). In a communication link ($t = com$), the ρ_s agents are allowed to communicate with ρ_d agents. In a authority link ($t = aut$), the ρ_s agents are allowed to have authority on ρ_d agents, i.e., to control them. An authority link implies the existence of a communication link that implies the existence of an acquaintance link:

$$link(\rho_s, \rho_d, aut) \Rightarrow link(\rho_s, \rho_d, com) \quad (1)$$

$$link(\rho_s, \rho_d, com) \Rightarrow link(\rho_s, \rho_d, acq) \quad (2)$$

Regarding the inheritance relation, the links follow the rules:

$$(link(\rho_s, \rho_d, t) \wedge \rho_s \sqsubset \rho'_s) \Rightarrow link(\rho'_s, \rho_d, t) \quad (3)$$

$$(link(\rho_s, \rho_d, t) \wedge \rho_d \sqsubset \rho'_d) \Rightarrow link(\rho_s, \rho'_d, t) \quad (4)$$

For example, if the coach role has authority on the player role $link(\rho_{coach}, \rho_{player}, aut)$ and player has a sub-role ($\rho_{player} \sqsubset \rho_{attacker}$), by Eq. 4, a coach has also authority on attackers. Moreover, a coach is allowed to communicate with players (by Eq. 1) and it is allowed to represent the players (by Eq. 2).

Collective level. The links constrain the agents *after* they have accepted to play a role. However we should constrain the roles that an agent is allowed to play depending on the roles this agent is currently playing. This *compatibility*

¹ Although we will use the term “specification” in the sequence, the $\mathcal{MOISE}^+$ could also be used to “describe” an organization.

constraint $\rho_a \bowtie \rho_b$ states that the agents playing the role ρ_a are also allowed to play the role ρ_b (it is a reflexive and transitive relation). As an example, the team leader role is compatible with the back player role ($\rho_{leader} \bowtie \rho_{back}$). If it is not specified that two roles are compatible, by default they are not. Regarding the inheritance, this relation follows the rule

$$(\rho_a \bowtie \rho_b \wedge \rho_a / \neg \rho_b \wedge \rho_a \sqsubset \rho') \Rightarrow (\rho' \bowtie \rho_b) \quad (5)$$

Roles can only be played in the collective level, i.e., in a group already created in an OE. We will use the term “group” to mean the instantiated group in an OE and the term “group specification” to mean the group specified in an OS. Thus, a group must be created from a *group specification* represented by the tuple

$$gt =_{\text{def}} \langle \mathcal{R}, \mathcal{SG}, \mathcal{L}^{intra}, \mathcal{L}^{inter}, \mathcal{C}^{intra}, \mathcal{C}^{inter}, np, ng \rangle \quad (6)$$

where $\mathcal{R}$ is the set of not abstract roles that may be played in groups created from gt . Once there can be many group specifications, we write the identification of the group specification as subscript (e.g. $\mathcal{R}_{gt}$). The set of possible sub-groups of a group is denoted by $\mathcal{SG}$. If a group specification does not belong to any group specification $\mathcal{SG}$, it is a root group specification.

A group can have intra-group links $\mathcal{L}^{intra}$ and inter-group links $\mathcal{L}^{inter}$. The intra-group links state that an agent playing the link source role in a group gr is linked to all agents playing the destination role in the *same* group gr or in a gr sub-group. The inter-group links state that an agent playing the source role is linked to all agents playing the destination role despite the groups these agents belong to. For example, if there is a link $link(\rho_{student}, \rho_{teacher}, com) \in \mathcal{L}^{inter}$, then an agent α playing the role $\rho_{student}$ is allowed to communicate with the teacher(s) of the groups where it is a student and also with the teachers of any other group, even if α does not belong to these groups.

The roles compatibilities also have a scope. The intra-group compatibilities $\rho_a \bowtie \rho_b \in \mathcal{C}^{intra}$ state that an agent playing the role ρ_a in a group gr is allowed to also play the role ρ_b in the *same* group gr or in a gr sub-group. Otherwise, the inter-group compatibilities $\rho_a \bowtie \rho_b \in \mathcal{C}^{inter}$ state that an agent playing ρ_a in the group gr_1 is also allowed to play ρ_b in other group gr_2 ($gr_1 \neq gr_2$). For instance, an agent can be a teacher in a group and a student in another, but it can not be both in the same group, so it is an inter-group compatibility.

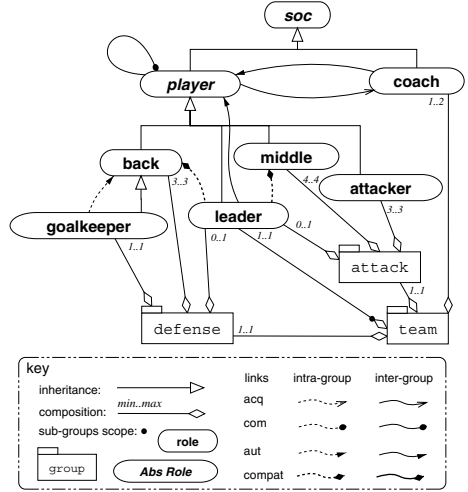


Fig. 2. Structure of a soccer team

Along with the compatibility, we state that a group is well formed if it respects both the role and sub-groups *cardinality*. The partial function $np_{gt} : \mathcal{R}_{gt} \rightarrow \mathbb{N} \times \mathbb{N}$ specifies the number (minimum, maximum) of roles that have to be played in the group, e.g., $np_{gt}(\rho_{coach}) = (1, 2)$ means that *gt* groups need at least one and no more than two coaches to be well formed. Analogously, the partial function $ng : \mathcal{SG}_{gt} \rightarrow \mathbb{N} \times \mathbb{N}$ specifies the sub-groups cardinality. By default, cardinality pairs are $(0, \infty)$.

For example, the defense soccer team group can be defined as

$$def = \langle \{\rho_{goalkeeper}, \rho_{back}, \rho_{leader}\}, \{\}, \{link(\rho_{goalkeeper}, \rho_{back}, aut)\}, \{\}, \{\rho_{leader} \bowtie \rho_{back}\}, \{\}, \{\rho_{goalkeeper} \vdash \neg(1, 1), \rho_{back} \vdash \neg(3, 3), \rho_{leader} \vdash \neg(0, 1)\}, \{\} \rangle$$

In this group specification (see Fig. 2), three roles are allowed and any defense group will be well formed if there is one, and only one, agent playing the role goalkeeper, exactly three agents playing backs, and, optionally, one agent playing the leader role. The goalkeeper has authority on the backs and the leader is allowed to be either a back or the goalkeeper, since $\rho_{back} \sqsubset \rho_{goalkeeper}$.

Using the recursive definition of group specification, we can specify a team as

$$team = \langle \{\rho_{coach}\}, \{def, att\}, \{\}, \{link(\rho_{player}, \rho_{player}), com\}, link(\rho_{leader}, \rho_{player}), aut, link(\rho_{player}, \rho_{coach}), acq, link(\rho_{coach}, \rho_{player}), aut\}, \{\}, \{\}, \{\rho_{leader} \vdash \neg(1, 1), \rho_{coach} \vdash \neg(1, 2)\}, \{def \vdash \neg(1, 1), att \vdash \neg(1, 1)\} \rangle$$

A *team* is well formed if it has one defense group, one attack group, one or two agents playing the coach role, one agent playing the leader role, and the two sub-groups are also well formed. The group *att* is specified only by the graphical notation presented in the Fig. 2. In this structure, the coach has authority on all players by an inter-group authority link. The players, in any group, can communicate with each other and are allowed to represent the coach. There must be a leader either in the defense or attack group. In the defense group, the leader can also be a back and in the attack group it can be a middle. The leader has authority on all players on all groups, since it has an inter-group authority link on the player role. In this group, an agent ought to belong to just one group because there is no inter-group compatibilities. However, notice that a role may belong to several group specifications (e.g., the leader).

Based on those definitions, the SS of a MAS organization is formed by a set of roles ($\mathcal{R}_{ss}$), a set of root group specifications (which may have their sub-groups, e.g. the group specification *team*), and the inheritance relation ($\sqsubset$) on $\mathcal{R}_{ss}$.

3 Functional Specification

The FS in $\mathcal{MOISE}^+$ is based on the concepts of missions (a set of global goals²) and global plans (the goals in a structure). These two concepts are assembled

² Regarding the terminology proposed in [3], these goals are collective goals and not social goals. Since we have taken an organizational centered approach, it is not possible to concept the social goal which depends on the agents internal mental state.

in a Social Scheme (SCH) which is essentially a goal decomposition tree where the root is the SCH goal and where the responsibilities for the sub-goals are distributed in missions (see Fig. 3 and Tab. 2 for an example). Each goal may be decomposed in sub-goals through plans which may use three operators:

- sequence “,”: the plan “ $g_2 = g_6, g_9$ ” means that the goal g_2 will be achieved if the goal g_6 is achieved and after that also the goal g_9 is achieved;
- choice “|”: the plan “ $g_9 = g_7 | g_8$ ” means that the goal g_9 will be achieved if one, and only one, of the goals g_7 or g_8 is achieved; and
- parallelism “||”: the plan “ $g_{10} = g_{13} || g_{14}$ ” means that the goal g_{10} will be achieved if both g_{13} and g_{14} are achieved, but they can be achieved in parallel.

It is also useful to add a certainty success degree in a plan. For example, considering the plan “ $g_2 = g_6, (g_7 | g_8)$ ”, there may be a environment where the achievement of g_6 followed by the achievement of g_7 or g_8 does not imply the achievement of g_2 . Usually the achievement of the plan right side implies the achievement of the plan goal g_2 , but in some contexts this may not happen. Thus, the plan has a success degree that is continually updated from its performance success. This value is denoted by a subscript on the =. For example, the plan “ $g_2 =_{0.85} g_6, (g_7 | g_8)$ ” achieves g_2 with 85% of certainty.

In a SCH, a *mission* is a set of coherent goals that an agent can commit to. For instance, in the SCH of the Fig. 3, the mission m_2 has two goals $\{g_{16}, g_{21}\}$, thus, the agent that accepts m_2 is committed to the goals g_{16} and g_{21} . More precisely, if an agent α accepts a mission m_i , it commits to all goals of m_i ($g_j \in m_i$) and α will try to achieve a g_j goal only when the precondition goal for g_j is already achieved. This precondition goal is inferred from the sequence operator (e.g.: the goal g_{16} of the Fig. 3 can be tried only after g_2 is already achieved; g_{21} can be tried only after g_{10} is achieved).

A *Social Scheme* is represented by a tuple $\langle \mathcal{G}, \mathcal{M}, \mathcal{P}, mo, nm \rangle$ where $\mathcal{G}$ is the set of global goal; $\mathcal{M}$ is the set of mission labels; $\mathcal{P}$ is the set of plans that builds the tree structure; $mo : \mathcal{M} \rightarrow \mathbb{P}(\mathcal{G})$ is a function that specifies the mission set of goals; and $nm : \mathcal{M} \leftrightarrow \mathbb{N} \times \mathbb{N}$ specifies the number (minimum, maximum) of agents that have to commit to each mission in order to say the SCH is well formed, by default, this pair is $(1, \infty)$, i.e., one or more agents can commit to the mission.

For example, a SCH to score a soccer-goal (sg) could be (see Fig. 3):

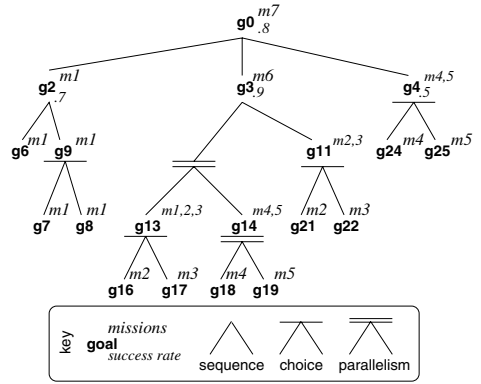


Fig. 3. An example of Social Scheme to score a soccer goal

$$sg = \langle \{g_0, \dots, g_{25}\}, \{m_1, \dots, m_7\}, \{ "g_0 =_{.8} g_2, g_3, g_4", "g_2 =_{.7} g_6, g_9)", \dots \}, \\ \{m_1 \vdash \rightarrow g_2, g_6, g_7, g_8, g_{13}\}, m_2 \vdash \rightarrow g_{13}, g_{16}, g_{11}, g_{24}\}, \dots, m_7 \vdash \rightarrow g_0 \}, \\ \{m_1 \vdash \neg(1, 4), m_2 \vdash \neg(1, 1), m_3 \vdash \neg(1, 1), \dots \} \rangle$$

This SCH is well formed if from one to four agents have committed to m_1 and one, and at most one, agent has committed to the other missions. The agent that will commit to the mission m_7 is the very agent that has the permission to create this SCH and to start its execution, since the m_7 is the sg root goal.

It is also possible to define a *preference order* among the missions. If the FS includes $m_1 \prec m_2$, then the mission m_1 has a social preference on the mission m_2 . If there is a moment when an agent is permitted to m_1 and also m_2 , it has to prioritize the execution of m_1 . Since m_1 and m_2 could belong to different SCHs, one can use this operator to specify the preferences among SCHs. For example, if m_1 is the root mission of the

Table 1. Goal descriptions of the Fig. 3.

goal	description
g_0	score a soccer-goal
g_2	the ball is in the middle field
g_3	the ball is in the attack field
g_4	the ball was kicked to the opponent's goal
g_6	a teammate has the ball in the defense field
g_7	the ball was passed to a left middle
g_8	the ball was passed to a right middle
g_9	the ball was passed to a middle
g_{11}	a middle passed the ball to an attacker
g_{13}	a middle has the ball
g_{14}	the attacker is in good position
g_{16}	a left middle has the ball
g_{17}	a right middle has the ball
g_{18}	a left attacker is in a good position
g_{19}	a right attacker is in a good position
g_{21}	a left middle passed the ball to a left attacker
g_{22}	a right middle passed the ball to a right attacker
g_{24}	a left attacker kicked the ball to the opponent's goal
g_{25}	a right attacker kicked the ball to the opponent's goal

SCH for an attack through one side of the field (sg) and m_2 is the root of other SCH for the substitution of a player, then $m_1 \prec m_2$ means that the sg must be prioritized.

To sum up, the FS is a set of several SCHs and mission preferences which describes how a MAS usually achieves its global goals, i.e., how these goals are decomposed by plans and distributed to the agents by missions. The FS evolve either by the MAS designer who specifies its expertise in a SCH form or by the agents themselves that store their (best) past solutions (as an enterprise does through its "procedures manual").

4 Deontic Specification

The FS and SS of a MAS, as described in Sec. 2 and Sec. 3, can be defined independently. However, our view of the organization effects on a MAS suggests

a kind of relation among them (Fig. 1). So in $\mathcal{MOISE}^+$ this relation is specified in the individual level as permissions and obligations of a role on a mission.

A permission $per(\rho, m, tc)$ states that an agent playing the role ρ is allowed to commit to the mission m , and tc is a time constraint on the permission, i.e., it specifies a set of periods during which this permission is valid, e.g.: every day/all hours, for Sundays/from 14h to 16h, for the first month day/all hours. In order to save space, the language for specifying the tc is not described here (it is based on the definitions presented in [1]). *Any* is a tc set that means “every day/all hours”. Furthermore, an obligation $obl(\rho, m, tc)$ states that an agent playing ρ ought to commit to m in the periods listed in tc . These two predicates have the following properties: if an agent is obligated to a mission it is also permitted to this mission; and deontic relations are inherited:

$$obl(\rho, m, tc) \Rightarrow per(\rho, m, tc) \quad (7)$$

$$obl(\rho, m, tc) \wedge \rho \sqsubset \rho' \Rightarrow obl(\rho', m, tc) \quad (8)$$

$$per(\rho, m, tc) \wedge \rho \sqsubset \rho' \Rightarrow per(\rho', m, tc) \quad (9)$$

For example, a team deontic specification could be:

$$\{per(\rho_{goalkeeper}, m_7, Any)\}, \{obl(\rho_{goalkeeper}, m_1, Any), \\ obl(\rho_{back}, m_1, Any), obl(\rho_{leader}, m_6, Any), obl(\rho_{middle}, m_2, Any), \\ obl(\rho_{middle}, m_3, Any), obl(\rho_{attacker}, m_4, Any), obl(\rho_{attacker}, m_5, Any)\}$$

In our example, the goalkeeper can decide that the SCH sg will be performed. The goalkeeper has this right due its permission for the sg mission root (Fig. 3). Once the SCH is created, other agents (playing $\rho_{back}, \rho_{leader}, \dots$) are obligated to participate in this SCH. These other agents ought to pursue their sg goals just in the moment allowed by this SCH. For instance, the middle agent α that accepts the mission m_2 will try to get the ball (g_{16}) only after the ball is in the middle field (g_2 was achieved).

The DS is thus a set of obligations and permissions for the agents, through roles, on SCH, through missions. In the context of the Fig. 1, the DS delimits the set $S \cap F$. Among the allowed behaviors (S), an agent would prefer a $S \cap F$ behavior because, for instance, this latter set gives it a kind of social power. If an agent starts a SCH (i.e., a place in $S \cap F$) it can force, by the DS, other agents to commit to this SCH missions. Notice that the set of all goal for an agent are not defined by the DS, only the relation of its roles to *global* goals are defined. The agents may also have their local, eventually social, goals, although this is not covered by the $\mathcal{MOISE}^+$.

Having an OS, a set of agents will instantiate it in order to form an OE which achieves their purpose. Once created, the OE history starts and runs by events like agent entrance or leaving, group creation, role adoption, SCH starting or finishing, mission commitment, etc. Despite the similarities with the object oriented area, there is not a “new Role()” command to create an agent for a role. In our point of view, the agents of a MAS are autonomous and decide to “follow” the rules stated by the OS. They are not created by/from the organization specification, they just accept to belong to groups playing roles. However, this

paper does not cover how an agent will (or won't) follow the organizational norms.

5 Conclusions

In this paper, we have presented a model for specifying a MAS organization along the structural and functional dimension, which are usually expressed separately in MAS organization models as we have stressed in the introduction. The main contribution of this model is the *independence* design of each one of these dimensions. Furthermore, it makes explicit the deontic relation which exists between them. We have used the MOISE^+ model to properly specify the three dimensions of a MAS organization in both a soccer domain, used as an example here, and in a B2B (business to business) domain, not presented here.

Comparing this proposal with the MOISE model [9], on which this work is based, the contributions in the structural dimension aim, on one hand, to facilitate the specification with the inclusion of an inheritance relation on the roles, and on the other hand, to verify if the structure is well formed, with the inclusion of the compatibility among roles and of a cardinality for roles and groups. Regarding the functional dimension, the main contributions are: the changes in the mission specification in order to express the relation among goals and their distribution through the inclusion of SCHs in the model; the inclusion of the preference among missions; and the inclusion of time in the deontic relations. Its functional specification is represented in a high abstraction level. Nevertheless, this specification could be specialized in a more detailed functional description already developed in the MAS area. For instance, a SCH could be detailed in a TEMS task description [4] without redefining the structural specification.

Even if an organization is useful for the achievement of a global purpose, as mentioned in the introduction, it can also make the MAS stiffer. Thus the system may lose one important property of the MAS approach, its flexibility. For example, if the environment changes, the current set of allowed organizational behaviors may not fit the global purpose anymore. In order to solve this problem, a reorganization process is mandatory. The MOISE^+ independence property was developed aiming to facilitate this process since we can change, for instance, the functioning dimension without changing the structure, only the deontic dimension needs to be adjusted. This trend will be part of our future work.

References

- [1] Thibault Carron and Olivier Boissier. Towards a temporal organizational structure language for dynamic multi-agent systems. In *Pre-Proceeding of the 10th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'2001)*, Annecy, 2001. 126
- [2] Cristiano Castelfranchi. Commitments: From individual intentions to groups and organizations. In Toru Ishida, editor, *Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS'96)*, pages 41–48. AAAI Press, 1996. 120

- [3] Cristiano Castelfranchi. Modeling social action for AI agents. *Artificial Intelligence*, (103):157–182, 1998. 123
- [4] Keith Decker and Victor Lesser. Task environment centered design of organizations. In *Proceedings of the AAAI Spring Symposium on Computational Organization Design*, 1994. 127
- [5] Jaques Ferber and Olivier Gutknecht. A meta-model for the analysis and design of organizations in multi-agents systems. In Yves Demazeau, editor, *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98)*, pages 128–135. IEEE Press, 1998. 118, 120
- [6] Mark S. Fox, Mihai Barbuceanu, Michael Gruninger, and Jinxin Lon. An organizational ontology for enterprise modeling. In Michael J. Prietula, Kathleen M. Carley, and Les Gasser, editors, *Simulating Organizations: Computational Models of Institutions and Groups*, chapter 7, pages 131–152. AAAI Press / MIT Press, Menlo Park, 1998. 118, 121
- [7] Francisco Garijo, Jorge J. Gómez-Sanz, Juan Pavón, and Philippe Massonet. Multi-agent system organization: An engineering perspective. In *Pre-Proceeding of the 10th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'2001)*, Annecy, 2001. 118
- [8] Les Gasser. Organizations in multi-agent systems. In *Pre-Proceeding of the 10th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'2001)*, Annecy, 2001. 118
- [9] Mahdi Hannoun, Olivier Boissier, Jaime Simão Sichman, and Claudette Sayetat. MOISE: An organizational model for multi-agent systems. In Maria Carolina Monard and Jaime Simão Sichman, editors, *Proceedings of the International Joint Conference, 7th Ibero-American Conference on AI, 15th Brazilian Symposium on AI (IBERAMIA/SBIA'2000)*, Atibaia, SP, Brazil, November 2000, LNAI 1952, pages 152–161, Berlin, 2000. Springer. 118, 120, 121, 127
- [10] Christian Lemaître and Cora B. Excelente. Multi-agent organization approach. In Francisco J. Garijo and Christian Lemaître, editors, *Proceedings of II Iberoamerican Workshop on DAI and MAS*, Toledo, Spain, 1998. 118
- [11] M. V. Nagendra Prasad, Keith Decker, Alan Garvey, and Victor Lesser. Exploring organizational design with TÆMS: A case study of distributed data processing. In Toru Ishida, editor, *Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS'96)*, pages 283–290. AAAI Press, 1996. 118
- [12] Young-pa So and Edmund H. Durfee. An organizational self-design model for organizational change. In *AAAI93 Workshop on AI and Theories of Groups and Organizations*, 1993. 118
- [13] Gerhard Weiß. Some studies in distributed machine learning and organizational design. Technical Report FKI-189-94, Institut für Informatik, Technische Universität München, 1994. 118

The Queen Robots: Behaviour-Based Situated Robots Solving the N-Queens Puzzle

Paulo Urbano, Luís Moniz, and Helder Coelho

Faculdade de Ciências da Universidade de Lisboa
Campo Grande 1749-016 Lisboa, Portugal
{pub,hal,hcoelho}@di.fc.ul.pt

Abstract. We study here the problem of solving the traditional n-queens puzzle by a group of homogeneous reactive robots. We have devised two general and decentralized behaviour-based algorithms that solve the puzzle for N mobile robots. They all make a depth-first search with backtracking –in the wild– guaranteeing –in principle– a solution. In the first one, there is a predefined precedence order in the group; each robot has local sensing (sonar), a GPS, and is able to communicate with the previous and next group elements. In the other algorithm, there is only local sensing ability and a GPS. There is neither a predefined group order nor any peer-to-peer communication between the robots. We have validated our algorithms in a simulation context.

1 Introduction

The n-queens puzzle is a standard example of the Deliberative Paradigm in Artificial Intelligence (AI). We have to find a way of disposing n chess queens on a board where there are no queens attacking each other. Solving this puzzle is considered an intelligent task and it is generally done by a reasoning process, operating on a symbolic internal model.

Recent research on autonomous agents tried to deal with the deficiencies of this paradigm for action-oriented tasks, such as its brittleness, inflexibility, no real time operation, dependence on well structured environments, and so on. Reactive robotics and Behaviour based robotics are new developed ideas on how autonomous agents should be organized in order to effectively cope with these type of tasks.

Behaviour based AI [1,5] was inspired by “the society of mind” of Minsky [6] where many small and relatively simple elements act in parallel each handling their own area of expertise. Intelligent behaviour arises from two sources: the interaction between multiple units running in parallel and the interaction between the agent and its environment. We use here the behaviour concept of Mataric [5] where a behaviour is a control law for reaching/maintaining a particular goal. In general, a behaviour is based on the sensory input but the notion of internal state can also be included. In fact, the

concept of behaviour is an abstraction for agent control, hiding the low-level details of control parameters, allowing task and goal specification in terms of high-level primitives. Attainment goals imply a terminal state: reaching a home region or rotating n degrees clockwise. In contrast, persistence goals are never attained but persist in time: avoiding obstacles is a good example of this type of goal. The reactive paradigm [2] requires that an agent respond directly to each situation without deliberation and planning. He has to find locally the necessary information in order to act. The situation affords the action. Another important concept is the fact that robots are embedded in the real world. They are spatially located entities, they have a body, and so they have to be facing some direction, having some objects in view. The idea is to take into account this inevitable fact in order to simplify cognitive tasks and the associated machinery.

We present two different kind of homogeneous and reactive robot groups, that are able to collectively solve the n -queens puzzle. Using only sonars and a GPS, they are able to collectively search externally for a solution, making a depth-first search “on the wild” and guaranteeing “in principle” a solution to the puzzle.

The second section describes the distributed depth-first search with backtracking that is behind our implementations. In section 3, we describe the simulation platform. The fourth section we present and discuss the implementations, using Player/Stage simulation system [3,7] and Aglets workbench [4]. Finally, we present our conclusions in section 5.

2 A Distributed Depth-First Search, in the World, with Back Tracking

Let’s consider we have four agents living in a world that includes a 4×4 grid. Their names are Shandy, Cossery, Hrabal and Erofeev, and this enumeration order corresponds to the group precedence order. Each agent has no cognitive capacities relying on perceptual, motor and communicative actions. An individual is able to detect others (attacks) on the precedent rows (top), along columns and diagonals of his current patch—this capacity is not cognitive but simply perceptual. Every agent, each one in the respective row, waits outside the board, until the precedent agent asks them to execute their individual behaviours, which are completely identical (see next figure).

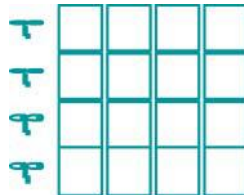


Fig. 1. Every agent is waiting for a message in order to explore its rows

What is the individual behaviour? It is very simple, each one explores their respective row, from left to right, in order to find a non-attacked patch. If the agent finds one good patch, he stops there, and just in case he is not the last group element, asks the next agent to look for a patch in the following row. Otherwise, when he does not find a non-attacked patch on the row and also when he is not the first agent, he will ask the precedent agent to look himself for a new patch, that is, to backtrack. The group stops when the last agent finds a good patch, solving the problem collectively, or in the worst case when the first agent explores completely the first row (no solution was found). Let us see how they do it.

As Shandy is the first agent, we have to give him a hand and ask him to begin his behaviour. Shandy finds immediately a non-attacked patch (the first one) and asks Cossery (the next one on the precedence order) to look for a good patch.

Cossery explores his row from left to right and stops on the third patch, asking Hrabal to go.

Hrabal will try to find a free patch but there is no free patch. When he arrives to the row end he will ask Cossery to find a new patch, that is, to backtrack, and Hrabal will start going towards his initial position.

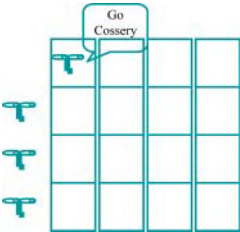


Fig. 2. Shandy found a free patch and signals Cossery to look also for a free patch

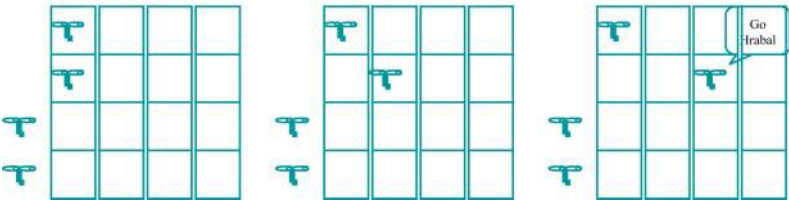


Fig. 3. Cossery found a free patch and signals Hrabal

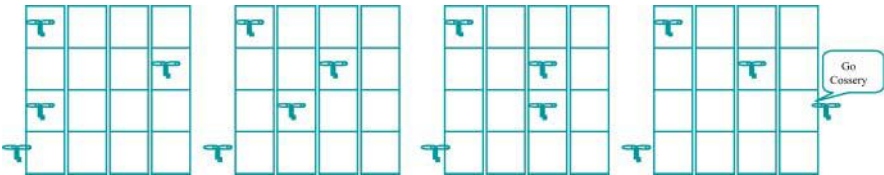


Fig. 4. Hrabal did not find any free patch he signals Cossery to find a new free patch

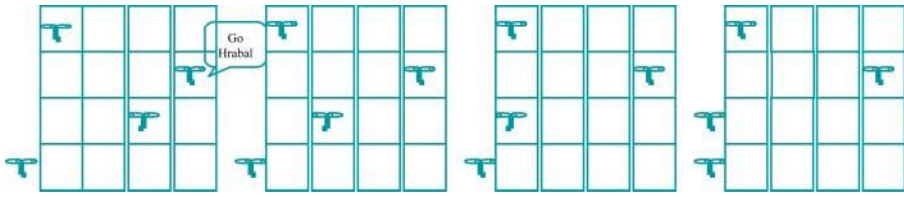


Fig. 5. Hrabal returns to his initial position while Cossery found a new free patch. He said to Hrabal to explore his row

Cossery will go on from left to right exploring his row and he finds another free patch. He sends a message to Hrabal: go. Meanwhile, Hrabal is returning to his initial position.

Now, Hrabal would remember he has already received a message from Cossery and would look for a safe patch and the group would go on exploring collectively the external problem space until finding a solution. We can see that the group is doing a depth-first search with a backtracking, not by using a reasoning process upon a symbolic state space, but is doing it in the world in a distributed fashion. Therefore, a solution to the problem is guaranteed in case it exists. There is an exhaustive board exploration from top to bottom and from right to the left. It is why the agents do not need to verify the bottom attacks. This algorithm does not depend on the number of agents, being well adapted to any board, solving the general n -queens puzzle.

It is important to notice that in general a backtracking process demands memory resources. In our algorithm, memory is not necessary because agents are exploring from left to right, which is coded in individual behaviours. This is due to the particularities of the puzzle structure, which our agents take into account. When an agent arrives to the row end, he has surely explored every patch on that same row and its time to send a message to the previous agent. We have to remark that we do not need two types of messages, just one: look for the next free patch. The agent that has just sent a message has now to wait for a forthcoming message from his partner while he repositions himself on the startup place. Now he will again explore his row, but for a new partner position.

Our agent has a body and he is always facing a certain direction: he is situated. Therefore, head movement corresponds to a kind of active perception. In order to verify that the patch underneath is not attacked he has to move his head towards the direction of the column and diagonals (North, North-west, North-east) and watch. We assume, due to body limitations, that he is not able to face the three directions at once, implying a sequence of three consecutive turns for testing if a patch is attacked. He will first watch the column, then the left diagonal and finally the right diagonal. But as soon as he detects another agent he will go to the next patch. Our algorithm depends on the agent body!

The agents' behaviour can be described by a finite-state-automata, where we associate actions with state transitions. We should stress that north direction corresponds to 0° and east to 90° (increasing clockwise). The finite-state machine diagram is depicted in figure 6. Let us describe the conditions and actions of the algorithm.

Conditions:

Attacked: Is the agent seeing another individual along the direction he is facing.

Inside: The agent is inside the board area?

Message: Did the agent receive a message?

Actions

Goto origin-x origin-y: Go to initial position.

Goto-next-cell: go forward along the row towards next cell. This implies to walk forward some distance, the cell length.

Sethead Dir: Turn body towards a certain direction.

Signal-previous: Send the message □GO□ to the previous agent, in case he exists.

Signal-next: Send the message □GO□ to the next agent, in case he exists.

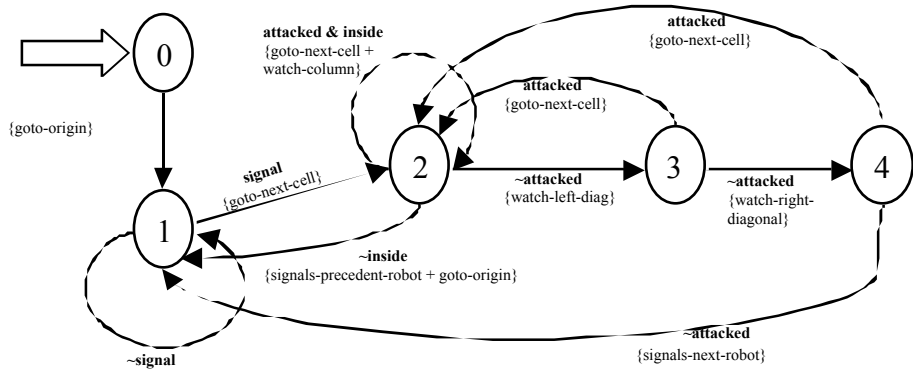


Fig. 6. Finite-state automaton diagram. The watch behaviours are dependent on the agent body□ they correspond to set heading towards some direction

3 Architecture: Player/Stage and Aglets

We have built a tool designed to aid the construction and management of simple behaviour based agents that control robots in a simulated environment. This framework is based on two different tools: the extended Aglets multiagent platform and the Player/Stage environment for robotics simulation. The framework links these two heterogeneous environments into a single platform, providing us with a tool for construct and an environment to experiment agents. The resulting testbed merges the Aglets platform features and the dynamic and unpredictable characteristics of the Player/Stage environment producing a tool capable of combining social and physical aspects of agents into a single experiment. In the next figure we present an architecture overview of the interaction of the Aglets platform and the Player/Stage environment

The Aglets framework consists in a set of Java class libraries on a top of a Java-based mobile agent framework. The system presented extends the original framework in order to provide a set of new capabilities to the agents and to the system designer.

The Player/Stage platform simulates a team of mobile robots moving and sensing in a two-dimensional environment. The robots behaviours are controlled by the Player component of the system. The Stage component provides a set of virtual devices to the Player, various sensors models, like a camera, a sonar and a laser, and actuators models, like motors and a gripper. It also controls the physical laws of robot interaction with each other and the obstacles. In our current environment only the sonar, laser and motors are usable.

This tool provides a controllable framework to test and experiment in a simple robotic environment.

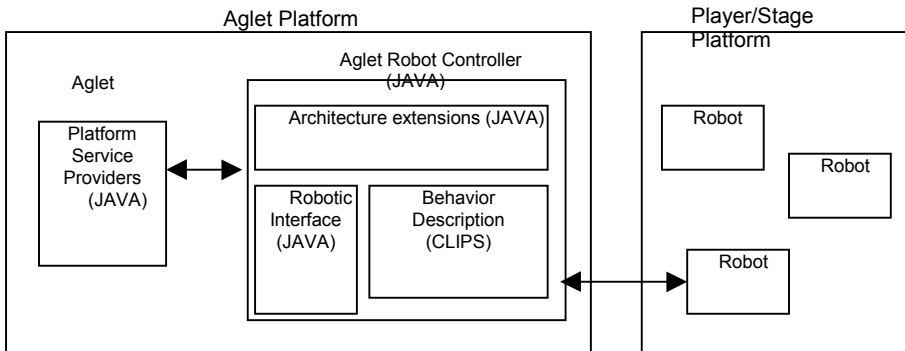


Fig 7. System overview

Integrating this tool with the Aglet platform, allowed us to add some new features to the environment and to associate an Aglet to manage each robot. This Aglet controls the robot behaviour using the Player interface (sensing and actuating), and it is capable of communicate with the other Aglets (robots) through the platform. This extension provides the robots with a communicating channel (peer to peer and broadcast) that provides them with complex message exchange capabilities. Additionally we add a GPS to the system, providing the robot with the knowledge of its absolute position in the environment.

We also associate a simple console command line and display to each Aglet. Through this console is possible to track the Aglet execution and communicate directly with it. We also add the possibility of add a special Aglet without a robot attached. This Aglet revealed itself useful to track the Aglet simulation and to communicate with the other agents, for instance, to broadcast a message to all of them.

To simplify the design of the robot behaviour we choose to describe it using CLIPS. The CLIPS language is a rule-based language with a forward chaining based reasoning engine. The user can define the robot behaviour in terms of first order logic rules, in the form of pairs conditions/actions. To support the feature we had to incorporate in our Aglets the Jess CLIPS interpreter engine.

4 The Queen Robots

We are going to discuss the implementations of the algorithm described in section 2, for the 4-queens puzzle, using the Aglets + Player/Stage simulation environment.

4.1 The Board

The sonar and GPS gives us values in the millimetre scale, so we have to draw a virtual world, a free space with no obstacles, where we can “imagine” our board. We consider that each patch is centred on a precise point in the board. A robot is considered inside a particular patch if he is near the patch centre. For example, in world with dimension 20000*20000, the board could be the square subpart of the world, from (10000,11000) to (13000 14000). Initially all the robots will be in their initial positions outside of the board. They will be in the column immediately to the left of this imaginary board. (The top robot initial position will be (10000,10000), the second robot initial position will be (10000,11000), and so on).

4.2 The Robot Body

In the next figure we have an image of the simulated robots we are working. They are equipped with 16 sonars and a GPS. Notice it is oriented 135 degrees.

As we may see in the robot body, the only sonars that we are going to use for attack detection will be the two laterals on the left side of the robot (indicated by the arrows). In order for the robot to detect attacks along the column and both diagonals he has to turn towards 0°, 45° and 135° in order to go along the line he will be orientated towards 90° and when he is returning to the initial position he will be heading 270°. The robot orientation in the figure allows him to detect attacks on the right diagonal.

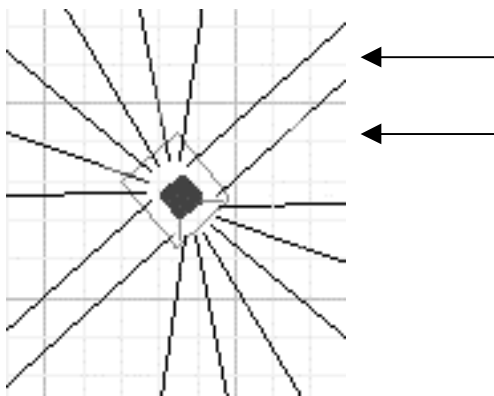


Fig. 8. The robot sonars

4.3 Situations/Behaviours

It is easy to implement the three conditions of the robot behaviour: (1) condition *attacked*, a robot is considered attacked if he detects a value on the two left lateral sensors which is less than the maximum sonar range, which means there is another robot in that direction. The robot does not need to know that the obstacle is a robot; the fact that there are no obstacles in the world simplifies perception abilities. (2) Condition *message*, each Aglet controlling the robot has a mailbox and it is trivial to verify if it has received a message. Finally (3) condition *inside*, each robot has a notion of the right end of the board and so when its GPS indicates that he is outside, the condition *inside* is considered false.

We have implemented behaviours, which correspond to the actions on the finite-state machine. The message services are already provided by our Aglets+player/stage platform. We have built several high-level behaviours that satisfy attainment goals, based on the Stage primitives (related with sonar, GPS and motor):

forward x: go forward x millimetres (a negative x means to go backwards). This behaviour has not an absolute precision. When the robot has covered a distance superior to x it stops.

goto x y: go to a particular patch. We can not have precision here either. When the robot is at a distance inferior to a certain small parameter (for example 80 mm) it stops

seth n: set heading towards the direction n. This behaviour is precise.

We have also what we can call a composite behaviour that corresponds to an ordered sequence of any number of behaviours. This way we can ask the robot, for example, to go forward 1000 mm, set the heading towards 0° and finally to go forward 700.

4.4 Robots with Peer-to-Peer Communication

We have run simulations with a group of four robots where we have fixed a precedence order between them. Each robot knows who is its precedent and next partners, if they exist. The first and last robots only have one partner. In general, the group is able to solve the n-queens puzzle but due to imprecision and noise sometimes the group does not converge towards a problem solution. For example, the robot can go out of the board too early after stopping only three times, also the robot can be displaced from a free patch centre, detecting obstacle which should not be detected for that patch. The robot stops around a certain point but small errors can be amplified, due to imprecision and noise as we said before.

4.5 Robots without Communication

In this second implementation, we tried to eliminate direct communication between robots. They do not know the id's of their precedent and next partners anymore. They will communicate by interfering with others, that is, by entering in the perceptual field of their partners – it is a kind of behavioural communication, a communicative act.

This time, if a robot has found a free patch it will go down into the next row in order to interfere with the next robot sonars. It will do it after waiting a fixed period of time that will be explained later. For this robot, this interference corresponds to the message go of the first implementation. To detect this behavioural signal the robots must be positioned facing south when they are in the initial position or when they are occupying a free patch the two situations robots are in when they receive a signal To be signalled is to detect an obstacle in the same two sonars as before.

In the next sequence of snapshots we see the four robots initially positioned, all facing south (180°); the first robot begins exploring its row and finds a free patch; at this time he goes down into the next row in order to call the second robot and returns to his free position; this last robot begins now explore its row.

Now, the second robot will find its free place and will go down signalling the third robot. This will try to find also a free patch, stopping around each patch centre, but there is none, so it will go up and signals the second robot to find a new free patch and return to its initial place. (Figure 9)

It could happen that a robot signals the next robot before this one arrives to its initial position, during the backtracking phase. Therefore, a robot after testing that a patch is not attacked it will wait some time before signalling the next robot. This waiting time, guarantees, in general, that the next robot will be positioned in the initial position and facing south in order to detect the interference. We built a new behaviour *wait t*, i.e., wait *t* seconds doing nothing.

We made several simulations and in the most part, the group achieved a solution, but, sometimes, due again to certain imprecisions and delays, the solution was not attained. For example, sometimes the robot takes a very long time to go to the start position and the robot on top has signalled him already. Thus the signal is lost and the group is not enough robust to recuperate.



Fig. 9. Initial global situation. The first robot finds a free patch and signals the second robot. In the final snapshot, the second robot is watching the left diagonal



Fig. 10. The third robot is exploring without success its row and when it goes out of the board, it goes up signalling the second robot. This one finds a new free patch while the third robot is going back to the start position

We have made a slight improvement on robot behaviour. When a robot executes its signalling ritual to the next robot it will do it several times until the other acknowledges it has received the message. So a robot in a free patch will be faced south and will go down and up signalling the next robot. So when he returns to its position he waits sometime with its right lateral sonars activated. The signalled robot, before starting row exploration, goes up to the previous row in order to interfere with the right sonars of his partner, acknowledging him that he has received the go message. This way we overcome most of the problems of the latter implementation.

5 Conclusions

We have presented a n-queens puzzle general distributed algorithm for real robots, using concepts and techniques derived from Behaviour-Based and Reactive AI. The notion of body plays also an important role in this algorithm: the attack detection is not cognitive, but only perceptual. Our main goal is to try to adapt to the real world, algorithms that are traditionally made on the cognitive level. The solution does not result from a reasoning process on a mental model. It is produced in a distributed way by very simple homogeneous artificial entities, embedded in the world. Our idea was not to compete in terms of efficiency with traditional algorithms, but rather study how we can manage the interaction between agents and the world in order to simplify choice and to diminish cognitive load.

We expect that the procedure we devised as an exhaustive collective search externally, without a symbolic space state, structuring reality and behaviour can be transferred to other more realistic situations. We think that our work can be a contribution towards mastering the design of real agents, which are not individually very complex, but can solve problems at the collective level in dynamic environments with incomplete information.

Using a platform where we mix the Aglets workbench and Player/Stage robot simulator, we have made two implementations of the algorithm. In the first one, robots are able to communicate directly with each other and in the second, robots rely only on perception.

References

1. Arkin, Ronald C.: Behavior-Based Robotics. MIT Press (1998)
2. Brooks, R.: Intelligence Without Reason. A.I. Memo N°1293, MIT AI Laboratory (1990)
3. Gerkey, B., Stoy, K., Vaughan, R. T.: Player Robot Server, version 0.8c user manual, web reference <http://playerstage.sourceforge.net/doc/Player-manual-0.8d.pdf>
4. Lange, Danny B. and Oshima, Mitsuru: Programming & Deploying Mobile Agents with Java Aglets, Peachpit Press, (1998)

5. Mataric, M.: Interaction and Intelligent Behavior, Ph.D. Dissertation, Department of Electric Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge (1994)
6. Minsky, M.: The Society of Mind Simon and Schuster, New York (1986)
7. Vaughan, Richard T., "Stage: a multiple robot simulator" version 0.8c user manual, web reference <http://playerstage.sourceforge.net/doc/Stage-manual-0.8d.pdf>

The Conception of Agents as Part of a Social Model of Distance Learning

João Luiz Jung¹, Patrícia Augustin Jaques¹,
Adja Ferreira de Andrade^{2,3}, and Rosa Maria Vicari¹

¹ PPGC- Programa de Pós-
Graduação em Computação da Universidade Federal do Rio Grande do Sul -
Bloco IV, Campus do Vale
Av. Bento Gonçalves 9500, Porto Alegre, RS, Brasil
Fone: (51) 3316-6161
{jjung,pjaques,rosa}@inf.ufrgs.br

² PGIE- Programa de Pós-
Graduação em Informática na Educação
da Universidade Federal do Rio Grande do Sul.
Av. Paulo Gama, 110-sala 810- 8º andar (FACED)
Porto Alegre, RS, Brasil

³ FACIN-PUCRS - Pontifícia Universidade Católica do Rio Grande do Sul.
Prédio 30- Av. Ipiranga 6618, Porto Alegre, RS, Brasil
Fone: (51) 3320-35 58
adja@inf.pucrs.br

Abstract. This paper is part of a research project called "A Computational Model of Distance Learning based on the Socio-Interactionist Approach". This project is related to situated learning, i.e., in the conception of cognition as a social practice based on the use of language, symbols and signs. The objective is the construction of a Distance Learning environment, implemented as a multi-agent system composed of artificial and human agents, and inspired by Vygotsky's socio-interactionist theory. This paper aims at the conception of two of the agents from such architecture: the Semiotic and the Collaboration Agents. The Semiotic Agent is responsible for searching adequate instructional material in the database to be presented to the student. The Collaboration Agent is responsible for assisting the interaction among students in a collaborative communication tool and it will consider the cognitive, social and affective capabilities of the students, which becomes a more qualitative mechanism for learning.

Keywords: Intelligent Tutoring Systems, Distance Education, Socio-Interactionist Pedagogical Theories.

1 Introduction

The present work aims to present the conception of two agents (the Semiotic and Collaboration ones) that are modeled as part of the multi-agent architecture of the project "A Computational Model of Distance Learning Based on the Socio-Interactionist Approach".

The system proposed initially [1] [2] was formed by four classes of artificial agents □ the ZDP agent, the Mediating Agent, the Social Agent and the Semiotic Agent □ and the human agents (learners and tutors). The current system performed evolutions so that, now, it is composed by human agents (students and tutors) and by five classes of artificial agents: the *Diagnostic Agent* has the function of describing the cognitive diagnosis, modeling the group and suggesting pedagogical tactics; the *Mediating Agent* is an animated pedagogical agent responsible for the interface of the environment with the student and for applying (1) support tactics in accordance to student's cognitive profile (sent by the Diagnostic Agent) and (2) affective tactics in accordance to student's affective state (determined by the Mediating Agent); the *Collaboration Agent* is responsible for mediating/monitoring the interaction among students □ groups in synchronous tools of communication among the students (for example, chat); the *Social Agent* that should establish the integration of the society forming students □ groups for study and creating a Collaboration Agent for each formed group; and the *Semiotic Agent* responsible for the use of signs, concepts and language send to the Mediating Agent or Collaboration Agent and, consequently, presented to the student. Further details of the system may be found in [1], [2] and [9]. The tutoring system may function as an individual tutor, where the Mediating Agent presents pedagogical contents to the student in accordance to his/her profile and cognitive style, or as a facilitating system of collaboration, where the Collaboration Agent monitor and mediate the interaction among the students with collaborative tools. The architecture of the system can be viewed in the Fig. 1.

The social model implemented by the proposed system is strongly inspired by Vygotsky [18] [19]. One of the important concepts of the socio-interactionist theory of Vygotsky is that the relationship man-environment is a relationship mediated by symbolic systems, through instruments and signs. According to Vygotsky [18] [19], the signs are artificial incentives with the purpose of mnemonic aid; they work as middle ground for adaptation, driven by the individual's own control. The sign is guided internally. The function of an instrument is to serve as tool between the worker (in the case of this research, the student) and the object of his work, seeking help in some activity; these are guided externally.

To fulfill this function, the system is composed by an agent (the Semiotic Agent) that has the role to present the instruments and signs to the student as external stimulations. These signs and instruments (such as pictures, sounds, texts and others) compose the instructional material in the database, as we can see in the Fig. 1, represented by www, exercises and examples.

The presentation of this instructional material is based on Semiotic Engineering. According to the Semiotic Engineering [4], [15], [17], for the communication designer-user to be possible, it is necessary to consider that software applications (that comprehend interfaces) are signs, formed by signs, and that generate and interpret signs. The Semiotic Agent has the role of interface's designer. It decides which signs

will be used to present a determined subject to the student. In the Fig. 1, this pedagogical content will be presented to the student as a HTML page (HyperText Markup Language) that is sent, indirectly, to the Mediating Agent - a personal and animated tutor responsible for presenting the instructional material to the student. The Mediating Agent will also capture the student affective state for react in an appropriated way to develop a spirit state more positive to learning in the student.

In the Fig. 1, we can see that all information on user actions will be gathered by the Mediating Agent and sent to the Diagnostic Agent. The Diagnostic Agent updates the information in the student model and verifies, according to received data, if it is necessary to use a new educational tactic. In this case, it sends this information to the Mediating Agent. If this tactic is, for example, the presentation of an instructional content, the Mediating Agent makes a request to the Semiotic Agent.

The Diagnostic agent uses the concept of Zone of Proximal Development [19] to parameterize the cognitive diagnosis of the learner. The Diagnostic Agent has the role of modeling those skills of the group that are either in the "core"(learned) or in the ZPD - Zone of Proximal Development (need of support). The purpose is to support decisions on how to adapt the tutoring or choose the right level of coaching for the group.

When the Diagnostic Agent finds a deficiency in the student's learning and considers it would be interesting to perform an activity in group, it will make a request to the Social Agent. The Social Agent, in the Fig. 1, will create a Collaboration Agent and form a study group of students.

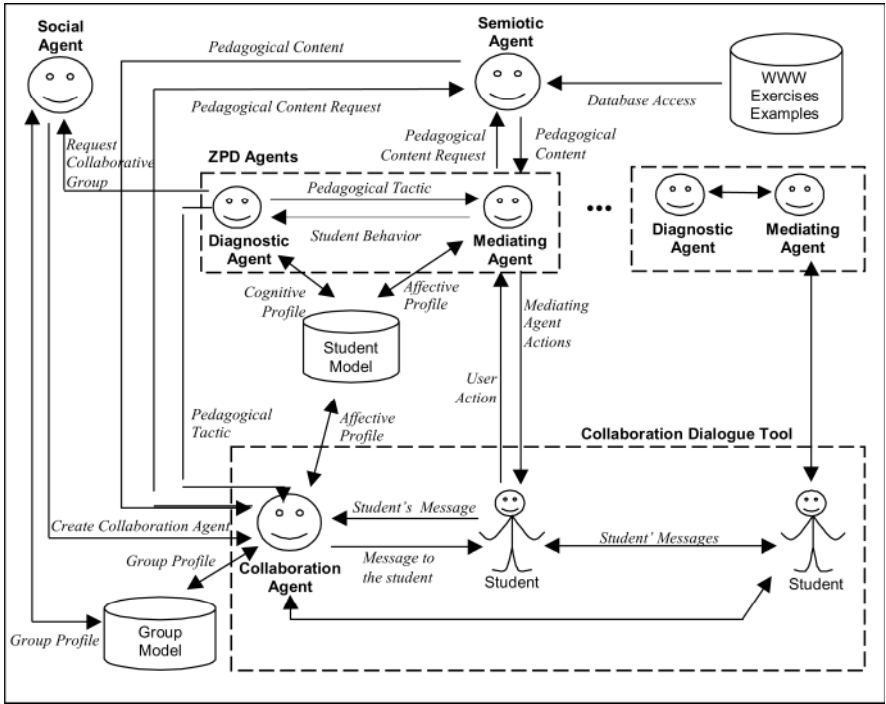


Fig. 1. A society of Social Agents for a Learning Environment

The Collaboration Agent, as we can see in the Fig. 1, is responsible for assisting the interaction among students in a virtual class within a collaborative communication tool, motivating them, correcting wrong concepts and providing new knowledge. This guiding agent will consider not only cognitive capabilities of students, but also social and affective characteristics, which becomes a more qualitative mechanism for collaboration among students and learning.

To implement the idea of social model of distance learning, this work presents a strong approach of communication among the agents which interact using KQML (Knowledge Query and Manipulation Language) performatives [5]. The architecture and further details about the system can be found in [2].

In the next section we describe the architecture and functionalities of the Semiotic Agent. In the section 3, we describe the Collaboration Agent. Finally, in section 4, we present some conclusions and some proposals of future work.

2 Semiotic Agent

The Semiotic Agent [11] looks for signs and instruments in the database, when requested by the Mediating Agent, to aid the student's cognitive activity, building dynamically the page to be introduced to the student and showing more specific contents as the student is going deeper in the detail of the subject. In this aim, the agent uses several signs, expressed in the most several ways, for example: the drawing, the writing (presenting the domain in form of paragraphs, examples, citations, tables, keywords, exercises), systems of numbers, illustrations and multimedia resources, propitiating, thus, the presentation of the instructional material conforms the teaching tactics specified by the Diagnostic Agent.

The Semiotic Agent is an agent that is inserted in a society of agents possessing the following properties [7] [20]:

- *autonomy*⁴, because it gets to act in the society for its own means and controlling its own actions;
- *social ability* interacting with other agents, as the Mediating Agent and Collaboration Agent;
- *reagent*, because it reacts to the incentives of solicitation of content of the Mediating Agent and Collaboration Agent;
- *continuity*, because it gets to stay continually in the society;
- *communicability*, because it exchanges messages with other agents (Mediating Agent and Collaboration Agent);
- *rationality*, although it is a weak "rationality", based on rules of decision, because it possesses the capacity to take decisions, in relation to which signs, or sequence of signs, it is better to present then as student's cognitive activity;
- *flexibility*, because it allows another agents' intervention (Mediating Agent and Collaboration Agent).

⁴ At this time, because there is only this agent, the autonomy degree it is not yet very well defined; as the other agents are being implemented, the degree of each agent's autonomy will be more visible and delimited.

To implement the idea of social model of distance learning, this work presents a strong approach of communication among the agents. The communication among the agents became a factor of great importance for the operation of the system. Detailed examples of messages exchanged among the agents can be seen in Jung [11].

2.1 Internal Architecture of Semiotic Agent

It can be observed that the Semiotic Agent, starting from the solicitation of incoming pedagogical content of the Collaboration Agent or Mediating Agent, verifies which are the tactics, preferences and the student’s level, seeking in the database which are the ideal signs to be used for the pedagogical content, generating dynamically a HTML page (as answer for the Mediating Agent) to be presented to the student. It can still send a message for the Collaboration Agent, in KQML, saying if the pattern found by the Collaboration Agent during the exchanges of messages among the students, it is part of certain content to be treated in the teaching-learning process [11].

The Fig. 2 shows the internal architecture of the Semiotic Agent.

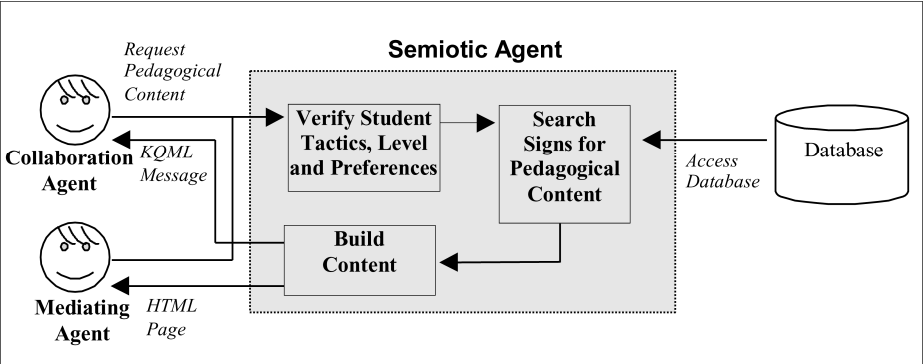


Fig. 2. Internal Architecture of Semiotic Agent [11]

2.2 Semiotic Agent and Semiotic Engineering

The Semiotic Agent has the role of interface’s designer. Its function is to decide which signs should be send for the Mediating Agent, given a certain situation, it means, depending on the teaching tactics specified by the Diagnostic Agent.

It is important to have a model to specify which signs will be used and how to present them to the user. In this research, we adopted the Message Specification Language of the Designer (MSLD) proposed by [12] and [13], whose objective is to support the formulation of the messages on the usability model.

Below, we show an example, using MSLD, of an instructional content introduced to the student. We can see that the rule of behaviour *Pedagogical_Content* (it will be explicated later at section 2.3), represented by action *Show_Content*, is composed by the junction of (1) information’s repetition of *Chapter*, *Section*, *Paragraphs*, *Html*, *Figure*, *Table*, *List*, *Example*, *Citation*, *Link*, *Keywords*, *Exercise*, followed by

information's repetition of *Reference*, and (2) *Previous* or *Next* options. Further details can be found in [10] and [11].

```

Command-Message Show_Content for Application-Function
Pedagogical_Content
Join{Sequence{Repeat{Join{ View Information-of Chapter
                           View Information-of Section
                           View Information-of Paragraphs
                           View Information-of Html
                           View Information-of Figure
                           View Information-of Table
                           View Information-of List
                           View Information-of Example
                           View Information-of Citation
                           View Information-of Link
                           View Information-of Keywords
                           Activate Show Command_Message Exercise}}
      Repeat{View Information-of Reference}}
Select{Activate Previous Application-Function
      Pedagogical_Content
      Activate Next Application-Function Pedagogical_Content}}

```

2.3 Semiotic Agent Implementation

The Semiotic Agent was implemented in Java, more specifically with the technology of servlets [21]. An environment was built, in Java, that allows to manage the whole instructional material (signs) stored in a database, where, later on, a XML file (eXtensible Markup Language) is generated with the content of each subject (see Fig. 3).

The Semiotic Agent, starting from the XML file, generates the instructional content (signs), according to the rules of behavior *User_Login*, *Pedagogical_Content* and *Requisition_Pedagogical_Content* defined by Jung [11]. Besides, it applies presentation styles (style sheets), through XSL (eXtensible Style Sheets), for formatting the exit, showing like this, in HTML, the same signs in a different way, depending on the level and the student's preference in subject.



Fig. 3. Interface of the Management Environment of the Instructional Material

All the actions of the Semiotic Agent are executed as a result of incentives generated through messages coming from the Mediating or Collaboration Agents. The rules of behaviour determine the course of the action that an agent should take from the beginning through the end of the agent's execution.

The rules of behaviour used in the implementation of the Semiotic Agent work in the following way [11]:

- **User_Login**: this rule happens when the Mediating Agent sends a message to the Semiotic Agent, informing that a student was connected to the system.

```

If user is registered Then
  it shows last pedagogical content accessed for the user
  it triggers the rule Pedagogical_Content
Else
  it registers the student
  it shows the first pedagogical content
  it triggers the rule Pedagogical_Content
End If

```
- **Pedagogical_Content**: the Semiotic Agent sends a message for the Mediating Agent as answer to the rule *Requisition_Pedagogical_Content* or *User_Login*.

```

If (operation = next) Or (operation = previous) Then
  it seeks in the database last content accessed for the user
  it accesses XML file
  it seeks ideal sign according to specified pedagogical
  tactics and user level
  If found ideal sign Then
    Show_Content exemplified in the section 2.2
    it applies XSL formatting exit according to level and user
    preference
    it sends message KQML for Mediating Agent with content in
    HTML
  Else
    it sends message KQML for Mediating Agent with empty content
  End If
Else If (operation = end) Then
  it keeps in the database the last action done by the student
End If

```
- **Requisition_Pedagogical_Content**: this rule is triggered by the Mediating Agent for the Semiotic Agent or of the Collaboration Agent to the Semiotic Agent. In the first case, according to the tactics defined by the Diagnostic Agent and the student's preference, the Mediating Agent will send a message to the Semiotic Agent requesting that the same generates a pedagogical content to be introduced to the student (it rules *Pedagogical_Content*). In the second case, the Collaboration Agent will request to the Semiotic Agent to verify certain pattern, found during interaction in the tool of collaboration, where this pattern is part of the pedagogical content that is discussed in the moment.

The illustration to proceed, presents the KQML performative sent by the Semiotic Agent for the Mediating Agent. For a complete explanation of the Distance Education Environment, with simulations of some messages KQML exchanged among the agents in the system, see [11].

The standardization of the signs (for example: chapter, section, paragraph, example, citation, lists) generates a cognitive pattern with the objective of facilitating the usability of the system and assistant the mnemonic process of the student's learning.

Table 1. KQML Message User Login

Parameter	Value
:performative	Tell
:sender	Mediating Agent
:receiver	Semiotic Agent
:ontology	user login
:in-reply-to	Mediating Agent
:reply-with	pedagogical content
:content	User Password Subject

3 Collaboration Agent

3.1 Definition of Collaboration Agent

Talk and discourse have long been seen as critical components in the learning process [14]. According to Vygotsky [19], the learning is frequently achieved through interactions supported by talk and that talk and language are frequently associated with the development of higher order learning.

Our system privileges the social interaction encouraging the students to interact in collaborative tools. In this way, the system has two agents with the ability to encourage the interaction among students: Social and Collaboration Agents. The Social Agent searches for peers that are capable of assisting a student in his/her learning process and creates a Collaboration Agent for mediating the interaction among the students. The Collaboration Agent will monitor and mediate the interaction between students in collaborative communication tools (for example, chat, discussion list and bulletin boards). It attends the students during the interactions, stimulating them when they look unmotivated, presenting new ideas and correcting wrong ones. In the Fig. 4, we show the internal architecture of the Collaboration Agent.

We can see in the Fig. 4, that during the interaction with the students in the collaborative tool, the Collaboration Agent interacts with the Diagnostic Agent to obtain new tactics to be used. In such a way, it must send the actions of the user, in this case, sent messages, so that the Diagnostic Agent decides which tactics must be carried out.

The Collaboration Agent interacts with the Semiotic Agent to get the pedagogical content (Fig. 4). For example, the Collaboration Agent can check, in accordance with statistical analyses of the students' message, which students presented incorrect ideas. As the interactions progress, the Diagnostic Agent can decide if a more difficult subject can be presented. In that case, the Collaboration Agent requests that the Semiotic Agent sends certain contents at a more difficult level.

The Collaboration Agent updates the affective model of the student (Fig. 4). It is responsible for obtaining the affective state of the student and updating the student model, in order to reply to the student with an appropriate emotional behaviour.

In collaborative learning, the group is an active entity; therefore, the system must contain information that refers to it as a whole. This information generates a group

model, which is constructed and stored by the Collaboration Agent, as showed in the Fig. 4.

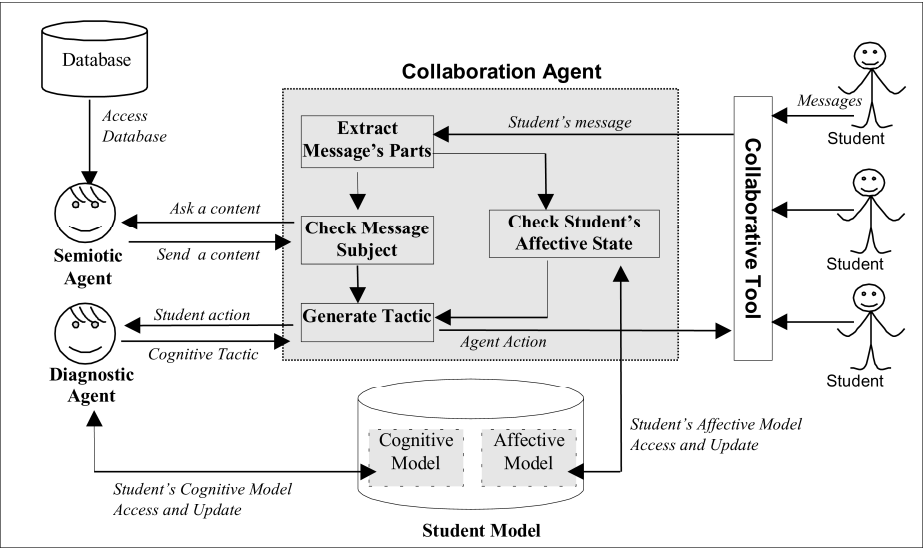


Fig. 4. The Internal Architecture of the Collaboration Agent

3.2 Collaboration Agent Implementation

Due to its social function to communicate with students, to promote and monitor the interaction among students it would be interesting for the Collaboration Agent to have an interface that would allow it to exploit students' social nature. In fact, one of our main concerns is to better exploit the social potential of the students to improve their learning, since studies demonstrate that people interacting with animated characters learn to interact with other humans [8]. Therefore, we chose to represent it as an animated character who has a personality and which interacts with the student through messages in natural language.

Thus, as in human social interactions, the Collaboration Agent must be able to show and perceive emotional responses. Learning is a comprehensive process which does not simply consists in the transmission and assimilation of contents. A tutor (in this case, the Collaboration Agent) must promote the student's emotional and affective development, enhancing his/her self-confidence and a positive mood, ideal to learning. The way in which emotional disturbances affect mental life has been discussed in the literature [6]. He recalls the well-known idea that depressed, bad-humoured and anxious students find greater difficulty in learning.

In order to interact with the student in an adequate way, the agent has to correctly interpret his/her emotions. In this way, we are studying with the aid of psychologists which affective states of the students the agent would consider and capture. Therefore, it is necessary for Collaboration Agent to have not only a student cognitive model, but also an affective one. We are going to use the student model proposed by [3], which considers the affective states such as effort, self-confidence and independence.

Still, it is necessary to have in mind the responsibility about the use of affective agent architecture for interaction with the user, especially in the education. Often we observe that agents have attitudes that are not suitable to students' mood (e.g., if an agent gets sad when the student could not carry out an exercise). This kind of attitude may generate a disturbed reaction in the student, making him/her more anxious and less self-confident. It is necessary to identify which behaviours are appropriate to promote a mood in the student that provides better learning conditions.

The Collaboration Agent will carry out the analysis of the student's dialogue based on statistical methods, such as pattern matching, message categorisation and information retrieval [16]. The messages will be generated in natural language, using dialogue models and frames.

4 Conclusions and Future Works

The use of agents in Intelligent Tutoring Systems (ITS) allows a better representation of the domain, with a larger possibility of application of pedagogical tactics that can aid in the learning process.

In this research, it tried to use the Semiotic Engineering through the formalism of MSLD for generation of signs, icons and symbols, representing the instructional material to be introduced to the student. The function of generation of the appropriate signs is responsibility of the Semiotic Agent, respecting like this, the important paper that it represents the mnemonic process of the student's learning through the signs, inspired by Vygotsky.

Besides, we tried to start the construction of a social model of distance learning, in that one of the involved agents has the role of designer and the role of meta-communicate the system's usability and functionality, generating the necessary signs for the teaching-learning process. The Semiotic Agent was implemented as part of the conception of collaborative learning in multi-agent system [9] obeying the negotiation, communication and learning properties.

When the systems function as a facilitating system of collaboration, the Collaboration Agent takes action. It monitors and mediates the interaction among the students in a collaborative dialogue tool, like a chat. In this case, it will collect and analyse the emotional data for react in an emotional way to promote a positive mood in the student, more ideal to learning. It must also present new ideas and correct wrong ones. In this aim, it will do a pre-analysis of the students' sentences and request to Semiotic Agent to verify if the sentences sent by the student are part of the discussed subject in the virtual collaborative class. It is possible due the model adopted to save the information in the database and by the form that they are manipulated by the Semiotic Agent.

The society of agents provides an environment that facilitates, through the social-interaction of the artificial and human agents (tutors and students) by talk and language, a process of teaching-learning inspired in the ideas defended by Vygotsky.

With the progress of implementation of the other agents, we can verify and to analyse the usability of the system, as well as to evaluate the results obtained with the use of the system.

As future works, the group intends to migrate the KQML messages communication among the agents for standard FIPA-ACL (Foundation for Intelligent Physical Agents – Agent Communication Language) [22], once it is stabilized and standardized, besides there is a movement in this direction in the KQML community [23].

References

1. Andrade, Adja; Jaques, Patrícia; Vicari, Rosa; Bordini, Rafael; Jung, João. Uma Proposta de Modelo Computacional de Aprendizagem – Distância Baseada na Concepção Sócio-Interacionista de Vygotsky. In: Workshop de Ambientes de Aprendizagem Baseados em Agentes; Simpósio Brasileiro de Informática na Educação, SBIE 2000, 11., 2000, Maceió, Brazil. Anais... Maceió UFAL, (2000).
2. Andrade, Adja; Jaques, Patrícia; Vicari, Rosa; Bordini, Rafael; Jung, João. A Computational Model of Distance Learning Based on Vygotsky's Socio-Cultural Approach. In: Mable Workshop (Multi-Agent Based Learning Environments), International Conference on Artificial Intelligence on Education, 10., 2001, Antonio, Texas. Proceedings... Texas: [s.n.], (2001).
3. Bercht, M.; Moissa, H.; Vicari, R.M. Identificação de fatores motivacionais e afetivos em um ambiente de ensino e aprendizagem. Simpósio Brasileiro de Informática na Educação, SBIE, 10., 1999, Curitiba, PR. Anais... Curitiba: UFRP, (1999). Poster.
4. Eco, U. Tratado geral de semiótica. São Paulo: Perspectiva, (1980). 282p. Original name: Trattato di semiotica generale, 1976.
5. Finin, Tim; Weber, Jay; Widerhold, Gio et al. DRAFT Specification of the KQML Agent-Communication Language: plus example agent policies and architectures. [S.l.]: The DARPA Knowledge sharing Initiative External Interfaces Working Group, (1993). Available online at <<http://logic.stanford.edu/papers/README.html>>.
6. Goleman, D. Emotional Intelligence. Objetiva, (1995).
7. Giraffa, Lucia Maria Martins. Uma arquitetura de tutor utilizando estados mentais. PhD Thesis in Computer Science – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil, (1999).
8. Huard, R. Character Mastery with the Improvisational Puppets Program. Technical Report (KSL-98-11) – Stanford University, (1998).
9. Jaques, P. A.; Andrade, A. F.; Jung, J. L.; Bordini, R. H.; Vicari, R. M. Using Pedagogical Agents to Support Collaborative Distance Learning. In: Conference in Computer Supported Collaborative Learning, CSCL, 2002, Boulder, Colorado, EUA. Proceedings... [S.l.:s.n.], (2002).
10. Jung, João; Jaques, Patrícia; Andrade, Adja; Bordini, Rafael; Vicari, Rosa. Um Agente Inteligente Baseado na Engenharia Semiótica Inserido em um Ambiente de Aprendizado – Distância. In: Workshop Sobre Fatores Humanos em Sistemas Computacionais, IHC, 4., 2001, Florianópolis, SC. Anais... Florianópolis: UFSC, (2001). Poster.
11. Jung, João Luiz. Concepção e Implementação de um Agente Semiótico como Parte de um Modelo Social de Aprendizagem a Distância. Master Dissertation in

- Computer Science □ Instituto de Inform tica, Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil, (2001).
12. Leite, J. C. Modelos e Formalismos para a Engenharia Sem tica de Interfaces de Usu rio. PhD Thesis in Computer Science - Departamento de Inform tica, Pontif cia Universidade Cat lica do Rio de Janeiro, Rio de Janeiro, Brazil, (1998).
13. Leite, J. C.; de Souza, C. S. Uma Linguagem de Especifica  o para a Engenharia Sem tica de Interfaces de Usu rio. In: Workshop Sobre Fatores Humanos em Sistemas Computacionais, IHC, 1999, Campinas, SP. Proceedings... Campinas: Instituto de Computa  o da UNICAMP, (1999).
14. Oliver, Ron; Omari, Arshad; Herrington, Jan. Exploring Students Interactions in Collaborative World Wide Web Learning Environments. In T. Muldner e T. Reeves, (Eds.), Educational Multimedia/Hypermedia and Telecommunications 1997. Charlottesville: AACE, (1997). pp 812-817.
15. Peirce, C. S. Sem tica. 3.ed. S o Paulo: Ed. Perspectiva, (2000). (Cole  o estudo, n. 46). Cole  o dos manuscritos de 1931-1958.
16. Soller, A. Supporting Social Interaction in an Intelligent Collaborative Learning System. Intelligence Journal of Artificial Intelligence in Education, 11. (2001).
17. Souza, C. S. de. The Semiotic Engineering of User Interface Languages. International Journal of Man-Machine Studies, [S.l.], v.39, p.753-773, (1993).
18. Vygotsky, L.S. Thought and Language. Cambridge, MA: MIT Press, (1962).
19. Vygotsky, L.S. Mind in Society. Cambridge, MA: Harvard University Press, (1978).
20. Wooldridge, M.; Jennings, N. Intelligent Agents: Theory and Practice. Knowledge Engineering Review, [S.l.], v.10, n.2, p.115-152, (1995). Available online at <<http://www.elec.qmw.ac.uk/dai/pubs>>.
21. Deitel, H. M.; Deitel, P. J. Java Como Programar. 3.ed. Porto Alegre: Bookman, (2001).
22. Foundation for Intelligent Physical Agents (FIPA) specifications homepage. Available online at <<http://www.fipa.org/specifications/index.html>>.
23. Jeon, Heecheol; Petrie, Charles; Cutkosky, Mark R. ACL-Based Agent Systems. In: IEEE Internet Computing Online. Available online at <<http://www.computer.org/internet/v4n2/w2agents.htm>>.

Emotional Valence-Based Mechanisms and Agent Personality

Eugénio Oliveira¹ and Luís Sarmiento^{1,2}

¹ NIADR - Faculdade de Engenharia Universidade do Porto
Rua Dr. Roberto Frias, s/n Lab. I 121, 4200-465 Porto, Portugal

² Escola das Artes □ Dep. Som e Imagem - Universidade Católica Portuguesa C.R.P.
Rua Diogo Botelho 1327, 4169-005 Porto, Portugal
eco@fe.up.pt
las@porto.ucp.pt

Abstract. Artificial Intelligence is once again emerging from a pragmatic cycle and entering a more ambitious and challenging stage of development. Although the study of emotion in the realm of Artificial Intelligence is not totally new (Simon, Minsky and Sloman and Croucher), much more attention has been recently devoted to this subject by several researchers (Picard, Velasquez, Wright). This renewed effort is being motivated by trends in neuroscience (Damasio, LeDoux) that are helping to clarify and to establish new connections between high level cognitive processes, such as memory and reasoning, and emotional processes. These recent studies point out the fundamental role of emotion in intelligent behavior and decision-making. This paper describes an on-going work that intends to develop a practical understanding of models backing those solutions and aims at their integration in Agent Architectures, having always in mind the enhancement of agents' deliberation capabilities in dynamic worlds.

1 Introduction

In the Artificial Intelligence field, the role of emotion in cognitive processing has been acknowledged since the late sixties by Herbert Simon [13]. Nevertheless, during the following 25 years, few were the researchers from the AI field that adventured themselves in the study of Emotion. Some notable exceptions are Marvin Minsky [7] and Aaron Sloman [14]. Recently, the work of the neuroscientist António Damásio [3] established a clear relationship between specific brain structures and emotional capabilities. Damásio's studies on his patients allowed the identification of specific brain regions (pre-frontal cortexes) that, whenever affected, would render the patient unable to respond to emotionally rich stimulus (e.g. violent or sexual content images). At the same time, these patients revealed significant difficulties in dealing with several real life situations, especially when confronted with the need to perform decisions either on a personal, social or professional level. However, in these cases, patients still keep their mathematics and speech skills intact, as well as their memory.

Their performance in IQ tests remains normal and most of the times their problem is unnoticeable. Damasio's results suggest that highly cognitive tasks such as risk assessment and decision-making are somehow related to emotional processing and that this relation is actually supported by neuronal structures. Evidence of a biological support for emotion-cognition relationship seems an extremely significant result, bringing some light over the original ideas of Simon, Minsky and Sloman.

Inspired by Damasio's work and following the work of several other researchers [11], [16], [17], [18], we started a project with the aim of endowing intelligent agents with the possibility to use an emotion-based mechanisms that strongly influence their own decision-making capabilities [15]. Furthermore, we are interested in studying how such emotion-based mechanisms can be manipulated and tuned to create different individual Agents based on the same Architecture. These Agents can be said to have distinct Personalities that could reveal themselves to be more advantageous for pursuing their own specific goals in specific environment conditions. The structure of this paper includes, besides this first introductory section, an introduction to the concept of Emotional Valence, which is at the core of our Architecture. We will then present our model of emotional mechanisms and establish its relationship with the other elements of the agent architecture. We will then try to show how emotional mechanisms, such as those we proposed, might be used to promote intelligent behavior. Finally we will present our current implementation of the architecture.

2 The Role of Emotion – Valence

Emotion is a highly complex multi-faceted phenomenon. Consequently, researchers from several fields have developed deep insights on the study of that concept. Thus, depending on the original field of the researchers the focus of the study could vary immensely. For an interesting survey about several emotion issues refer to [4]. From our perspective, as engineers and computer scientists, we are mostly interested in studying the functional aspects of emotional processes. Particularly, we aim to understand how emotional mechanisms can improve cognitive abilities, such as planning, learning and decision-making, for hardware and software Agents. We hope to develop more flexible Agent Architectures capable of dealing with highly complex, rapidly changing and uncertain environments. In a certain way, we are following the complementary direction of the work done by A. Ortony, A. Collins and G. Clore that lead to the well-known OCC model [10]. The OCC model is mainly focused on explaining "the contribution that cognition makes to emotion". The work presented in [10] discusses the cognitive processes that generate the appropriate conditions leading to given emotional states (*eliciting conditions*). We, on the other hand, seek to explore the functionality of emotional states to increase the performance of an Artificial Agent interacting with complex environments.

From a functional point of view there are several issues about emotion that we found useful to investigate and to work upon. The most fundamental functionality of emotion concerns state evaluation. In this context, Emotions can be regarded as a built-in mechanism able to provide *automatic and rapid evaluations of environment conditions together with its own internal state*. In particular, for a given Agent, with a defined set of goals and capabilities to change its environment, emotions are used to

identify the *valence of the environment and of its own capabilities*. We define *valence* as a subjective measure that relates the chances of an Agent being able to fulfill its goals given a particular environment situation, its internal state and its set capabilities. Valence may be positive if environment conditions and internal state of the agent are favorable to goal achievement, or negative otherwise. An important point to stress here about valence concept, is that agents evaluate environment not just ‘‘per se’’ but according to their own current goals and motivations. Based on the emotional capability we propose several other features that we believe may be advantageous for more sophisticated Agent Architectures: (1) Valence-based Long-Term Memory, (2) Emotional Alerts, (3) Action Tendency.

In the next sections we will address all these issues in detail. We will also address the possibility of exploring variations over several Emotional parameters. In fact, despite the same internal Emotional-based Architecture, we will demonstrate that two Agents may show different behavior in the same situation, reflecting the existence of two distinct Agent Personalities and two different past histories.

3 Emotional Valence-Based Mechanisms

As mentioned in the last section, Emotions provide an automatic and quick way of evaluating the environment and the internal state of the Agent in respect to its own goals. It is important to stress that this evaluation is twofold. Firstly, it reflects the outside environment conditions by providing a valence tag to the information gathered by the perception subsystems. For example, emotional mechanisms may alert to a particular outside situation that influences critically an agent goal (an extremely negative or positive valence situation) and, therefore, requires special treatment. In this context, emotional mechanisms will try to quickly answer questions such: ‘‘How good are environment conditions to my specific goal(s)?’’

Secondly, and also related with environment evaluation, Emotions do also reflect the fitness of the Agent to cope with specific environment states [5], [8], [9]. In particular, Emotions will valence Agent’s own action set, current plans and knowledge regarding their effect on goal achievement in a given environment. Up to a certain point, this process of internal evaluation can be regarded as a basic introspective activity. Emotional mechanisms will indirectly try to answer questions such as ‘‘How fit are these plans to help me achieving my goal(s)?’’ or ‘‘How useful has been my knowledge in my last decisions?’’

4 Valence Functions, Accumulators and Memory Thresholds

In this section we will introduce a model of the emotional mechanisms. As we have described above, these mechanisms should receive input from internal sources, I , as well external sources, E , and produce a valence measure, V , according to what we will call *Emotional Valence Function*, EVF . Emotional Valence Functions return the valence of the situation regarding a given goal, G : $V = EVF(I, E, G)$.

An Emotional Valence Function is supposed to be a fast mechanism and therefore it should easily computed. However, it is also possible conceive some higher level EVFs dealing with complex inputs, such social beliefs, as long as their computation does interfere in the ability of the Agent to respond in real-time to the environment. Emotional Valence Functions can be further decomposed in a Normalized Valence Function NEVF, whose values range from -1 to 1, and a Sensibility Factor S . Thus: $V = EVF_i(I, E, G) = S_i \times NEVF_i(I, E, G)$.

The Valence value returned by EVF is then used to update the agent internal state. For each EVF the agent keeps an *Emotional Accumulator* to which the Valence values are added. Emotional Accumulators exhibit a time dependent behavior. Their values decay with the passing of time, at a given *Decay Rate* (DA_i). This behavior is similar to the dynamics shown by emotions in people.

Emotional Accumulators are fundamental elements of the internal state of the agent, and have, as shown later, direct influence on all deliberative and reactive processes. Furthermore, let's assume that the valence measure and the sources of evaluation are then associated in order to form a Valence Vector: $\langle V, I, E, G \rangle$. Valence Vectors are stored in the working memory and made available for all Agent processes for further consideration. Valence Vectors may be stored afterwards in long-term memory by a dedicated process that selects specific vectors according to their relevance. For each EVF_k , let us define MT_k as the Memory Threshold level. Then a specific Valence Vector $\langle V_j, I, E, G \rangle$ is selected to be stored in long-term memory if: $|V_j| = |EVF_k(I, E, G)| > MT_k$.

Valence Vectors that have higher valence magnitude than the corresponding MT_k , can be seen as particularly relevant and should be stored for later processing while others may be simply discarded. We will explore this issue in the following section.

In summary, for each of its goals (explicit or implicit), an Agent will have one Emotional Valence Function, the corresponding Emotional Accumulators and Memory Thresholds. Figure 2 tries to depict what we have just described.

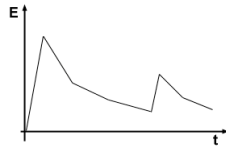


Fig. 1. Profile of an Emotional Accumulator. The rises of the curve represent (positive) updates from EVF. The value of the Accumulator decreases at a given decay each time slot

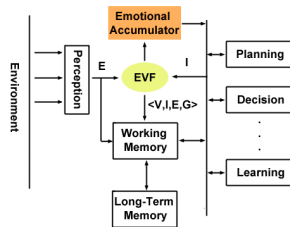


Fig. 2. Emotional Valence Function and its relationship within Agent Architecture

5 Valence-Based Long-Term Memory

By combining all the Valence Vectors that are being produced during its interaction with the environment, an Agent is able to create contextual memory maps of its past experiences. As we have seen before, Emotional Valence Functions and Memory Threshold levels allow the Agent to select which data is worth storing in long-term memory. Having in mind the purpose of Emotional Valence Function, we can say that highly valenced data is related either with good goal achievement perspectives (positive valence) or dangerous threats to specific goals (negative valence). Therefore, this selection process retains only the information that is considered particularly valuable to the goals of an Agent, while discarding less relevant, although probably much more abundant, information. Additionally, valenced long-term memory may help the search for pre-existing plans and facts. Long-term memory may be indexed by valence and then searched in an informed way. Contextually relevant information may be automatically transferred to working memory where more complex processing can be performed. For example, when facing a situation with a given calculated valence, all information coherent with that valence assessment can be decisive. Plans and facts used in situations with similar valence present a high probability of being reused or excluded according to the result (either positive or negative) they have achieved previously. Thus, the search for appropriate behaviors over a knowledge base can be pruned right from the beginning.

There is a certain similarity between the mechanism we have just described and Case Based Reasoning, although some important differences can be noted. Besides being much more simple than the overall CBR cycle [1], Valence-based Memory uses an Agent centered measure to choose which cases are to be retained: Valence. Thus, cases retained depend much more on the current performance of the Agent than on certain metrics defined during the design stage. Moreover, Valence-based Memory is not intended to store extensively all possible cases, which would be an inappropriate procedure considering the real-time demands of target environments and the memory limitations of Agents. As more recent Valence Vectors are computed, older or less significant ones may be "forgotten" so that the stored knowledge can be refreshed.

We will continue to work on this particular subject in order to develop a deeper understanding.

6 Emotion-Driven Agent Behaviors

One important feature of emotional processes is the immediate and intuitive recognition of critical situations, which are supposed to be reflected by strong valence assessments and high Accumulator levels. These emotional evaluations may generate new motives to change current behavior or even change Agent capabilities.

Alerts: Emotional Valence mechanisms may be useful in detecting situations that may interfere (positively or not) in goal achievement, alerting the Agent's internal processes for relevant events that may demand attention [8], [9]. When facing such events and situations, which in complex environments may not always be easily identified or expressed by beliefs, emotional alerts should drive the agent to focus on

important data. This alert and focusing will motivate the agent to eventually start classification or pattern recognition procedures and then search for appropriate actions. Emotional Accumulators, for example, may help the agent to detect situations that, although their instant valence is not particularly relevant, remain active for long periods.

Tendencies: However, more than just alerting and starting other processes, emotional mechanisms may also directly contribute to Agent response, by creating a specific internal context. As our own body and senses get prepared by the effect of fear to respond effectively (quickly or not) to a possible harmful situation (in this case, the goal is keeping physical integrity), Agents behavior at deliberative and reactive layers may also suffer similar alterations. Thus, emotion can be regarded as a mechanism capable of creating *action tendencies*. For example, emotions may be responsible for plan pre-selection, by offering deliberative layers a set of “experience tested” plans or rules. Although this first selection may eventually leave out the best choice, it does also contribute to reducing the work of deliberative layers that are then able to respond much more promptly, a condition that is usually essential in survival. In this sense, emotions contribute positively to the notion of Bounded Rational Agent [12] by allowing the Agent to behave as well as possible given its limited resources and complex environment conditions.

Moods: Emotional interference in action guidance can also be done in larger time spans. If we consider slower effect emotions that reflect themselves not in immediate actions but in new goals adoption we may be able to devise a long-term adaptation mechanism. These slower effect emotions, which remain active for longer time periods, may be seen as *moods* and their influence upon Agents is made at a higher level, namely in goal adoption. They also can be of great help in filtering current possible options selecting those which are in agreement with long term policies.

7 Testing Ideas

We are currently developing software simulations based on the platform RealTimeBattle (RTB), which is available at <http://realtimebattle.sourceforge.net>. This platform provides a simulated real-time environment where softbots fight for survival in dynamic scenarios. RTB allows the developer to program their own softbots in C/C++, as well as to create custom 2D scenarios. Simple physical properties (air resistance, friction, material hardness) are also implemented to enrich the simulation. Softbots perception is basically a set of radar events from which they can detect walls, other softbots, shots and randomly distributed energy sources and mines. Softbots can accelerate, break, rotate and shoot in a given direction. RealTimeBattle assures that Softbots have limited processor time but it demands real time response from the softbots. In this way RealTimeBattle platform seems appropriate to test some of the ideas described before. Our current Emotional Agent Architecture comprises 3 different layers. Each layer provides a set of capabilities that can be used by upper layers. Each layer has also included a set of simple Emotional Valence Functions and Accumulators are intended to reflect the success of the Agent in achieving an explicit or implicit goal.

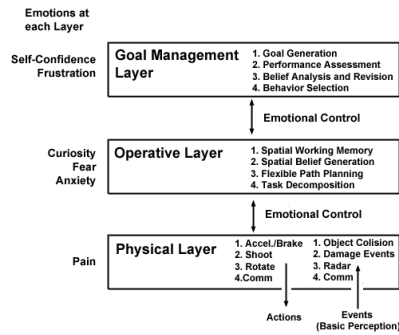


Fig. 3. Layered architecture. Emotional parameters generated at each layer are shown in the left side

The bottom layer is the Physical Layer and is highly domain dependent. In the case of RealTimeBattle it includes the softbot sensing capabilities and all the low-level action and communication mechanisms. At this level, the robot is capable of providing simple reactive responses to the environment. For, example, the softbot can shoot a close mine, without any further consideration. In the physical layer we have included one Emotional Valence Function and the corresponding Accumulator whose objective is to measure the aggressiveness of the environment. Our purpose is mimicking the function of pain in animals. Pain is deeply and directly related with the goal of survival and physical health. Thus, damaging events, as shoot and mine collisions, are internally reflected by high values of the $\square\text{Pain}\square\text{EVF}$ and increases in the $\square\text{Pain}\square\text{Accumulator}$. At the physical layer level these values will be reflected in some internal parameters such as the power used by the softbot when shooting or the speed of its moves. Thus, for each action Ac_j we have a set of preconditions P which include EVF and Accumulator (Acc) values: $P(Ac_j) = \{\{EVF\}, \{Acc\}\}$. The action itself is also function of a given set of EVFs and Accumulators: $Ac_j(\{EVF\}, \{Acc\})$. At upper layers other effects are also felt, but usually in an indirect way, as we will see.

The next layer, the Operative Layer, is responsible for more complex capabilities. It receives sensor data from the Physical Layer, and analyses it in order to construct a map representation of the environment and also to track the position of other robots, mines and cookies. This layer also keeps a record of the location where it was inflicted pain. The Operative Layer also provides path-planning capabilities. The softbot is capable of planning its path to a given destination given its knowledge of the environment collected through sensing. One particular issue about this planning capability is the possibility of controlling two different parameters: the number of steps in the path and their length. This allows the softbot to choose between simple and quicker plans or elaborate, but eventually slower, plans. These parameters will be subject to the influence of another EVF/Accumulator that represents an emotion similar to Anxiety. High the values of the $\square\text{Anxiety}\square\text{Accumulator}$ will result in the creation of shorter plan that allow the agent to respond promptly to a given situation. The EVF of the $\square\text{Anxiety}\square\text{accumulator}$ uses several input parameters, in which are included the value of other Emotional Accumulators such as $\square\text{Fear}\square$, $\square\text{Curiosity}\square$ and $\square\text{Pain}\square$. Since $\square\text{Fear}\square$, $\square\text{Curiosity}\square$ and $\square\text{Pain}\square$ depend themselves on several other

dynamic, time-varying, parameters (see Figure 4), it can be seen that the resulting structure is very complex and would be difficult to implement in the form of IF-THEN rules.

The upper layer, called Goal Management Layer (see figure 3), is still under implementation. Its purpose is to manage all agent high-level behaviors. It is responsible for generating goals and sub-goals and tracking their execution. The goal generation process will also be dependent on the value of EVF and Accumulators. For example, the Accumulator "Curiosity" may contribute to the generation of a goal such as "Explore Surroundings". This will then motivate an exploring behavior with several lower-level operative actions (look around, move to an unknown point in map). At this level we propose two different emotional dimensions related with global performance: "Self-Confidence" and "Frustration". "Self-Confidence" should increase when the softbot is regularly achieving its goals. It will be reflected in the way softbot deals with difficult situation, such as those that are related with high levels of "Fear". High levels of "Self-Confidence" will make the softbot adopt a more active behavior, such as attacking or hunting other robots. On the other hand, low levels of "Self-Confidence" accumulator will promote behaviors such as running away or hiding from enemies. Note that this behavior appears to provide a natural form of adaptation, increasing chances of survival. On the other side, "Frustration" should reflect the inadequacy of current behaviors to achieve given goals. It will indicate the softbot that a change of behavior or goal is needed. At this layer emotional mechanisms are essentially related with introspective activities.

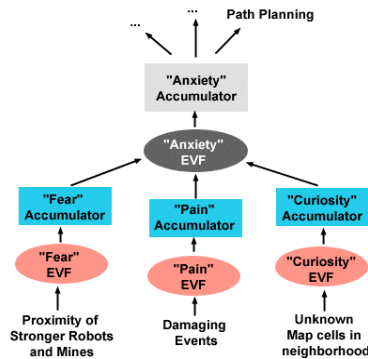


Fig. 4. Relationship between different emotional parameters from different layers

8 Agent Personality and Evolutionary Agent Design

Within the same Emotion-based Architecture, that includes specific EVFs, Emotional Accumulators and Memory Thresholds there are several different parameter variations that can be seen as distinct Agent Personalities. Although "Agent Personality" is certainly a difficult concept to define precisely, we may say that it is what distinguishes similar Agents (i.e. with the same Architecture) regarding their patterns of behavior. In this perspective, assuming that, for example, we change the intensity of how Emotional Accumulators interact with each other, we can expect the

overall behavior of the Agent to change because of their intense relations with all the Agent processes. In our Architecture the intensity of those interactions is ultimately controlled by the EVF . Therefore, EVF parameters may be considered, in a rather simplified way, as part of the Personality of the Agent. An Agent Ag_i personality, indirectly governing an Agent Ag_i behavior, can then be described as the complete set of its Emotional Valence Functions and corresponding Accumulators and Memory Thresholds: $Personality_i = \{EVF_{k,i}, Ac_{k,i}, MT_{k,i} \text{ for all } Em_k \in \{Em\} \text{ (the set of emotions)}\}$ and $Ag_i \in \{Ag\}$ (the set of Agents).

Let explore, for example, the Sensibility of the EVF_k . Despite the similarity of their overall internal structure, two Agents, Ag_r and Ag_s will tend to behave differently if they have different Sensibilities factors regarding the corresponding EVF : $Ag_r : EVF_{rk} = Sr_k * NEVF_k$ and $Ag_s : EVF_{rs} = Srs * NEVF_k$.

Higher sensibilities will naturally motivate the Agent to respond more quickly to a given environment stimulus. The Agent should therefore, in these cases, look more nervous and will probably change its behavior more abruptly.

We can broaden the concept of Agent Personality by also manipulating the Decay Rate of Emotional Accumulators (refer to Figure 1). Decay Rates are related with behavior stability. Slower decay rates will increase the stability of Agent  internal state, making it less dependent from environment changes. Agents with slow Decay Rates will be influenced by environment stimulus for longer periods. On the other hand faster decay rates will make the Agent surpass environment stimulus quickly.

These possibilities suggest an opportunity for tuning emotional parameters for better agent performance. Since each individual Agent has a particular set of emotional parameters, which comprise the Sensibility Factors (S) of EVF , the Decay Rate of Emotional Accumulator (DA) and Memory Threshold Levels (MT), we may admit there exists a specific combination of these parameters that optimizes Agent performance in a given environment. This combination would be the Optimal Agent Personality. For a given Emotion-based Architecture, we shall define the Personality Set Domain (PSD) as being the set of all possible combinations of Sensibility factors, Decay rates and Memory Thresholds:

$$PSD = \{S_1\} \times \{S_2\} \times \dots \times \{S_n\} \times \{DA_1\} \times \{DA_2\} \times \dots \times \{DA_n\} \times \{MT_1\} \times \{MT_2\} \times \dots \times \{MT_n\}.$$

Therefore, the PSD includes every possible Agent Personality within a specific Emotion-based Architecture. Finding the Optimal Agent Personality for a specific environment can be seen as a search problem over the PSD space. From a system designer point of view this suggests an evolutionary approach to the development of Emotion-based Agents, releasing to burden of finding the best Sensibility Factors, Decay Rates or even Memory Thresholds manually. The designer would perform a search for the optimal Agent Personality by varying these parameters around some reasonable initial values over several rounds of simulations. The parameters that yield the best Agents in respect to a certain performance criteria in a specific environment would then be selected as the Optimal Agent Personality for that environment.

Note that this search process does not reduce the ability of an Agent to cope with environmental changes. It is mainly a design method to help the developer automatically tune some or the available parameters. To cope with environment changes that happen during the  lifespan  of an Agent, the proposed Architecture includes other mechanisms located at the  Goal Management Layer . Namely, both

□Frustration□ and □Self-Confidence□ emotional mechanisms try to regulate the behavior of an Agent in order to promote adaptation (e.g. belief revision).

9 Conclusions

In this paper we have proposed an Emotional based Agent Architecture intended for Agents that operate in complex and real-time environments. Particularly we have been mainly concentrated on Emotional Valence Functions, which are mechanism that make possible for an agent to perform a fast evaluation of external and internal states regarding its chances of achieving its own goals. We have also showed how emotion-based processes could be used to direct deliberative agent processes, such as decision-making and planning. We have also discussed the possibility of exploring variations on such emotional mechanisms and its relation with the concept of Agent Personality.

References

1. Aamodt, E. Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications. IOS Press, Vol. 7:1, pp. 39-59.
2. L. Custódio, R. Ventura, C. Pinto-Ferreira. Artificial Emotions and Emotion-Based Control Systems. Proc. 7th IEEE Int. Conf. Emerging Technologies and Factory Automation. 1999.
3. Damásio. Descartes Error - Emotion, Reason and the Human Brain. 1994.
4. P. Ekman and R. Davidson (Editors). The Nature of Emotion. □ Fundamental Questions. Oxford University Press, 1994.
5. Nico Frijda. Emotions are functional, most of the time. In P. Ekman and R. Davidson (Editors). The Nature of Emotion. □ Fundamental Questions. Oxford University Press, 1994.
6. J. LeDoux. The Emotional Brain: the Mysterious Underpinnings of Emotional Life. 1996.
7. M. Minsky. The Society of Mind. First Touchstone Edition. 1988.
8. D. Moffat, N. Frijda. Functional Models of Emotion. In G. Hatano, N. Okada and H. Tanabe (Ed.), 13th Toyota Conference-Affective-Minds, pgs 169-181. Elsevier, Amsterdam, 2000.
9. D. Moffat. Rationalities of Emotion. To appear.
10. Ortony, G.L. Clore, A. Collins. The Cognitive Structure of Emotions. New York: Cambridge University Press.
11. R. Picard. Affective Computing. The MIT Press, 1997
12. S. Russel, P. Norvig. Artificial Intelligence: A Modern Approach . Prentice Hall. 1995.
13. H. Simon. Motivational and emotional controls of cognition. Psychological Rev., 74. 1967.

14. Sloman and M. Croucher. Why Robots will have emotions. In Proc. 7th Int. Joint Conference on AI, Vancouver 1981.
15. Sloman. Beyond Shallow Models of Emotion. In "Cognitive Processing", Vol. I, 2001.
16. J. Vel squez. Modeling Emotion-Based decision making. In Dolores Ca anero (Editor), Emotional and Intelligent: The Tangled Knot of Cognition, pages 164-169, 1998.
17. R. Ventura. Emotion-Based Agents. Msc. Thesis. Instituto Superior T cnico, Lisboa, Portugal, 2000.
18. Wright. Emotional Agents. Ph.D. thesis, School of Computer Science, The University of Birmingham, 1997 (<http://www.cs.bham.ac.uk/research/cogaff/>)

Simplifying Mobile Agent Development through Reactive Mobility by Failure

Alejandro Zunino, Marcelo Campo, and Cristian Mateos

ISISTAN Research Institute - UNICEN University
Campus Universitario (B7001BBO), Tandil, Bs. As., Argentina
{azunino, mcampo, cmateos}@exa.unicen.edu.ar

Abstract. Nowadays, Java-based platforms are the most common proposals for building mobile agent systems using web technology. However, the weak mobility model they use, the lack of adequate support for supporting inference and reasoning, added to the inherent complexity of developing location aware software, impose strong limitations for developing mobile intelligent agent systems. In this article we present *MoviLog*, a platform for building Prolog-based mobile agents with a strong mobility model. *MoviLog* is an extension of *JavaLog*, an integration of Java and Prolog, that allows a user to take advantage of the best features of the programming paradigms they represent. *MoviLog* provides logic modules, called *Brainlets*, which are able to migrate among different web sites, either proactively or reactively, to use the available knowledge in order to find a solution. The most interesting feature introduced by *MoviLog* is the reactive mobility by failure (RMF) mechanism. This mechanism acts when a specially declared Prolog predicate fails, by transparently moving a *Brainlet* to another host which has declared the same predicate to try to satisfy the current goal.

1 Introduction

A mobile agent is a computer program which represents a user in a computer network and is capable of migrating autonomously between hosts to perform some computation on behalf of the user [9]. Such a capability is particularly interesting when an agent makes sporadic use of a valuable shared resource. But also, efficiency can be improved by moving agents to a host to query a large database, as well as, response time and availability would improve when performing interactions over network links subject to long delays or interruptions [5].

Intelligent agents have been traditionally considered as systems possessing several dimensions of attributes. For example, [2] described intelligent agents in terms of a three dimensional space defined by *agency* (the degree of autonomy and authority vested in the agent), *intelligence* (the degree of reasoning and learned behavior) and *mobility* (the degree to which agents themselves travel through the network).

Based on these views it is possible to consider a mobile agent as composed of two separate and orthogonal behaviors: stationary behavior and mobile behavior;

the first one is concerned with the tasks performed by an agent on a specific place of the network, and the second one is in charge of making decisions about mobility.

Clearly, mobile agents, as autonomous entities fully aware of their location, have to be able to reason why, when and where to migrate in order to better use available network resources. Thus in addition to stationary behavior, whose development is recognized for being challenging and highly complex [14], mobile agent developers have to provide mechanisms to decide an agent's itinerary. Therefore, though agents' location awareness may be very beneficial, it also adds further complexity to the development of intelligent mobile agents, specially with respect of stationary applications [9, 10].

Most mobile agents rely on a *move* operation which is invoked when an agent wants to move to a remote site. Recent platforms support more elaborate abstractions that reduce the development effort. For example, Aglets [7] and Ajanta [13] support itineraries and meetings among agents. Despite these advances, the developer is repeatedly faced with the three *www*-questions of mobile agents: *why*, *when* and *where* to migrate.

This paper presents a new platform for mobile agents named *MoviLog* that uses stationary intelligent agents to assist the developer on managing mobility. *MoviLog* aims at reducing the development effort of mobile agents by automating decisions on why, when and where to migrate. *MoviLog* is an extension of the *JavaLog* framework [1] which implements an extensible integration between Java and Prolog

MoviLog provides mobility enabling mobile logic-based agents, called *Brainlets*, to migrate between hosts following a strong mobility model. Besides extending Prolog with operators to implement proactive mobility, the most interesting aspect of *MoviLog* is the incorporation of the notion of reactive mobility by failure (RMF). This mechanism acts when a specially declared Prolog predicate fails, by transparently moving a *Brainlet* to another host which has declared the same predicate to try to satisfy the current goal.

The article is structured as follows. The following section briefly describes the *JavaLog* framework. Section 3 introduces the *MoviLog* platform and examples of proactive and reactive mobility. Section 4 presents some experimental evaluations. Section 5 discusses the most relevant related work. Finally, in Section 6 concluding remarks and future works are presented.

2 The *JavaLog* Framework

JavaLog is multi-paradigm language that integrates Java and Prolog implemented in Java [1]. The *JavaLog* support is based on an extensible Prolog interpreter designed as a framework. This means that the basic Prolog engine can be extended to accommodate different extensions, such as multi-threading or modal logic operators, for example.

JavaLog defines the module (a list of Prolog clauses) as its basic concept of manipulation. In this sense, both objects and methods from the object-oriented

paradigm are considered as modules encapsulating data and behavior, respectively. The elements manipulated by the logic paradigm are also mapped to modules. JavaLog also provides two algebraic operators to combine logic modules into a single agent.

Each agent encapsulates a complex object called *brain*. This object is an instance of an extended Prolog interpreter implemented in Java, which enables developers to use objects within logic clauses, as well as to embed logic modules within Java code. In this way, each agent is an instance of a class that can define part of its methods in Java and part in Prolog. The definition of a class can include several logic modules defined within methods as well as referenced by instance variables.

The JavaLog language defines some interaction constraints between object-oriented and logic modules. These interaction constraints are classified by referring, communication and composition constraints. Referring constraints specify the composition limits of different modules. Communication constraints specify the role of objects in logic modules and the role of logic variables in methods. Composition constraints specify how logic modules can be combined, expressing also the composition of the knowledge base when a query is executed.

The following example involves customer agents capable to select and buy different articles based on users' preferences. A *CustomerAgent* class defines the behavior of customers whose preferences are expressed through a logic module received as a parameter. The *CustomerAgent* class is implemented in the following way:

```
public class CustomerAgent {
    private PILogicModule userPreferences;
    public CustomerAgent(PILogicModule prefs) { userPreferences = prefs; }
    public boolean buyArticle(Article anArticle) {
        userPreferences.enable(); type = anArticle.type; ...
        if(?-preference((#anArticle#,[#type#,#brand#,#model#,#price#]))
            buy(anArticle); userPreferences.disable(); }
    ... }
}
```

The example defines a variable named *userPreferences*, which references a logic module including a user's preferences. When the agent needs to decide whether to buy a given article, user's preferences are analyzed. The *buyArticle* method first enables the *userPreferences* logic module to be queried. In this way, the knowledge included in that module is added to the agent knowledge. Then, an embedded Prolog query is used to test if it is acceptable to buy the article. To evaluate *preference(Type, [Brand, Model, Price])*, *userPreferences* clauses are used. The query contains a Java variable enclosed into *#*. This mark allows us to use Java objects inside a Prolog clause. In the query, *send* is used to send a message to a Java object from a Prolog program. For instance, *send(#anArticle#, brand, Brand)* in Prolog is equivalent to *Brand = anArticle.brand()* in Java. Finally, the *buyArticle* method disables the *userPreferences* logic module. This operation deletes the logic module from the active database of the agent.

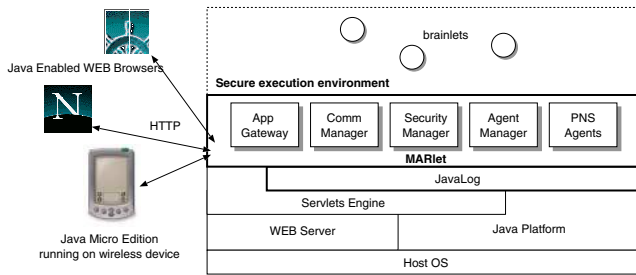


Fig. 1. MoviLog Web Server Extensions

To create a customer agent a logic module with the user preferences must be provided placed between `{{` and `}}` in the initialization:

```
CustomerAgent anAgent = new CustomerAgent( {{
    preference(car,[ford, Model, Price]) :- Model i 1998, Price i 200000.
    preference(motorcycle,[yamaha, Model, Price]) :-
    Model i= 1998, Price i 9000. }});
```

3 The MoviLog Platform

MoviLog is an extension of the JavaLog framework to support mobile agents on the web. MoviLog implements a strong mobility model for a special type of logic modules, called *Brainlets*. The MoviLog inference engine is able to process several concurrent threads and to restart the execution of an incoming Brainlet at the point where it migrated, either pro-active or reactively, in the origin host.

In order to enable mobility across sites, each web server belonging to a MoviLog network must be extended with MARlet (Mobile Agent Resource). A MARlet extends the Java servlets support encapsulating the MoviLog inference engine and providing services to access it (Fig. 1). In this way, a MARlet represents a web dock for Brainlets. Additionally, a MARlet is able to provide intelligent services under request, such as adding and deleting logic modules, activating and deactivating logic modules, and performing logic queries. In this sense, a MARlet can also be used to provide inferential services to legacy web applications or agents.

From the mobility point of view, MoviLog provides support to implement Brainlets with typical pro-active capabilities, but more interesting yet, it implements a mechanism for transparent reactive mobility by failure (RMF). This support is based on a number of stationary agents distributed across the network, called Protocol Name Servers (PNS). These agents provide a intelligent mechanism to automatically migrate Brainlets based on their resource requirements. Further details on this will be explained in Section 3.2.

3.1 Proactive Strong Mobility

The *moveTo* built-in predicate allows a Brainlet to autonomously migrate to another host. Before transport, MoviLog in the local host serializes the Brainlet and its state - i.e. its knowledge base and code, current goal to satisfy, instantiated variables, choice points, etc. Then, it sends the serialized form to its counterpart on the destination host. Upon receipt of an agent, MoviLog in the remote host reconstructs the Brainlet and the objects it refers to, and then it resumes its execution. Eventually, after performing some computation, the Brainlet can return to the originating host calling the *return* predicate.

The following example presents a simple Brainlet for e-commerce, which has the goal of finding and buying a given article in the network according a user's preferences. The *buy* clause looks for offers available in the different sites, selects the best and calls a generic predicate to buy the article (this process is not relevant here). The *lookForOffers* predicate implements the process of moving around through a number of sites looking for the available offers for the article (we assume that we get the first offer). If there is no offer in the current site, the Brainlet goes to the next one in the list.

```
Brainlet CustomerBrainlet = {
  sites([www.offers.com,www.freemarket.com,...]).
  preference(car,[ford, Model, Price]) :- Model = 1998, Price = 60000.
  preference(tv,[sony, Model, Price]) :- Model = 21in, Price = 1500.
  lookForOffers(A,[],[ ]). lookForOffers(A,[S— R], [O—RO], [O—ROff]):-
    moveTo(S), article( A, Offer, Email), O= (S,Offer,Email),
    lookForOffers(A, R, RO,ROff).
  lookForOffers(A,[S— R], [O—RO], [O— Roff]):- lookForOffers(A, R, RO,ROff).
  buy(Art):-
    sites(Sites), lookForOffers(Art, Sites,R,Offers), selectBest(Offers, (S,O,E)),
    moveTo(S), buy_article(O,E), return.
  ?- buy(#Art).
}
```

Although proactive mobility provides a powerful tool to take advantage of network resources, in the case of Prolog, it also adds an extra complexity due to its procedural nature. That is, mobile Prolog programs can not necessarily be built in the declarative way as a normal Prolog program is, forcing to implement solutions that depend on the mobility aspect. Particularly, when the mobile behavior depends on the failure or not of a given predicate solutions tend to be more complicated. This fact led us to develop a complementary mobility mechanism, called *reactive mobility by failure*.

3.2 Reactive Mobility by Failure

The MoviLog platform provides a new form of mobility called Reactive Mobility by Failure (RMF) which aims at reducing the effort of developing mobile agents by automatising some decisions about mobility. RMF is based on the assumption that mobility is orthogonal to the rest of attributes that an agent may possess

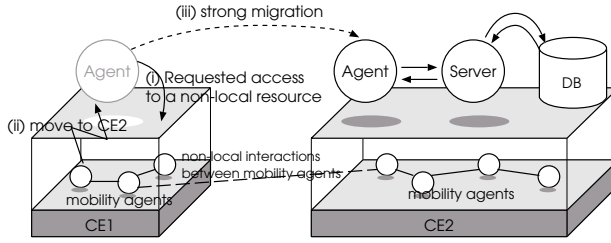


Fig. 2. Reactive Mobility by Failure

(intelligence, agency, etc) [2]. Under this assumption it is possible to think of a separation between these two functionalities or concerns at the implementation level [4]. RFM exploits this separation by allowing the programmer to focus his efforts on the stationary functionality, and delegating mobility issues on a distributed multi-agent system that is part of the MoviLog platform, as depicted in Fig 2.

RMF is a mechanism that, when a certain predicate fails, transparently moves a Brainlet to another site having definitions for such a predicate and continues the normal execution to try to find a solution. The implementation of this mechanism requires the MoviLog inference engine to know where to send the Brainlet. For this, MoviLog extends the normal definition of a logic module with protocol sections, which define predicates that can be shared across the network.

Protocol definitions create the notion of a *virtual database* distributed among several web sites. When a Brainlet defines a given protocol predicate in a MARlet h_n , MoviLog informs the PNS agents, which in turn inform the rest of registered MARlets that the new protocol is available in h_n . In this way, the database of a Brainlet can be defined as a set $D = \{D_L, D_R\}$, where D_L is the local database and D_R is a list of clauses stored in a remote MARlet with the same protocol clause as the current goal g . Now, in order to probe g the interpreter has to try with all the clauses $c \in D_L$ such that the head of c unifies with g . If none of those lead to probe g , then it is necessary to try to probe g from one of the non-local clauses in D_R . To achieve this, MoviLog transfers the running Brainlet to one of the hosts in D_R by using the same mechanism used for implementing proactive mobility. Once at the remote site, the execution continues trying to probe the goal. However, if the interpreter at the remote site fails to probe g , it continues with the next host in D_R . When no more possibilities are left, the Brainlet is moved to its origin.

The following code shows the implementation of the customer agent combining both mobility mechanisms. As can be noted, the solution using RMF looks much like a common Prolog program. This solution collects, through backtracking, the matching articles from the database until no more articles are left. The article protocol makes the Brainlet to try all the sites offering the same protocol before return to the origin site to collect (by using *findall*) all the offers in the

local database of the Brainlet. Once the best offer is selected the Brainlet proactively moves to the site offering that article to buy it. Certainly, this solution is simpler than the one using just proactive mobility.

PROTOCOLS

article(A, Offer, Email).

CLAUSES

preference(car, [ford, Model, Price]) :- Model $\geq$ 1998, Price $\geq$ 20000.

preference(tv, [sony, Model, Price]) :- Model = 21in, Price $\geq$ 1500.

lookForOffers(A, [O—RO], [O—Roff]) :- article(A, Offer, Email),
thisSite(ThisSite), assert(offer(ThisSite, Offer, Email)), fail.

lookForOffers(A, _, Offers) :- !, findAll(offer(S,O,E)), Offers).

buy(Art) :- lookForOffers(Art,R,Offers), selectBest(Offers,(S,O,E)),

moveTo(S), buy_article(O, E), return.

...

?- buy(Art).

Evaluation Algorithm The implementation of RMF can be understood by considering a classical Prolog interpreter with a stack S , a database D , and a goal g . Each entry of S contains a reference to the clause c being evaluated, a reference to the term of c that is being proved, a reference to the preceding clause and a list of variables and their values in the preceding clause to be able to backtrack. MoviLog extends this structure by adding information about the distributed evaluation mechanism. The idea is to keep a history of visited MARlets and possibilities for satisfying a given goal within a MARlet.

To better understand these ideas, let us give a more precise description of the evaluation mechanism. Let $s = \langle c, t_i, V, H, L \rangle$ be an element of the stack, where $c = h : -t_1, t_2, \dots, t_n$ is the clause being evaluated, t_i is the term of c being evaluated, V is a set of variable substitutions (ex. $X = 1, X = Z$) and $H = \langle H_t, H_v, P \rangle$, where H_t is a list of MARlets not visited, H_v is a list of MARlets visited and P is a list of candidate clauses at a given MARlet that match the protocol clause of c ; and L is a list of clauses with the same name and arity as t_i (candidate clauses at the local database).

The interpreter has two states: *call* and *redo*. When the interpreter is in state *call*, it tries to probe a goal. On the other hand, in state *redo* it tries to search for alternative ways of evaluating a goal after the failure of a previous attempt. Given a goal $? - t_1, t_2, \dots, t_n, S = \{ \}$ and *state* = *call*,

1. If *state* == *call*

(a) the interpreter pushes into the stack $\langle t_1, t_2, \dots, t_n, t_i, V = \{ \}, \langle H_t = \langle \rangle, H_v = \langle \rangle, PH_t = \langle \rangle \rangle$. For each term t_i in turn MoviLog performs the following steps:

i. If the MARlet is visited for the first time, the interpreter searches into the local database for clauses with the same name and arity as t_i . This result is stored into P (a list of clauses c_j at the current MARlet). Otherwise, P is updated with the clauses available at the current MARlet.

- ii. Then, the more general unifier (MGU) for t_i and the head of c_j is calculated. If there is not such an unifier for a given c_j , then c_j is removed from P . Otherwise, the substitutions for t_i and the head of c_j are stored into V . At this point, the algorithm tries to probe c_j by jumping to 1). If every t_i is successfully proved, then the algorithm returns *true*.
 - iii. If there is not a clause c_j such as there is a more general unifier for t_i and the head of c_j , the interpreter queries a PNS for a list of MARlets offering the same protocol clause as t_i . This is stored into H_t . Then, the BrainLet is moved to the first MARlet h_d in H_t . The current MARlet is removed from H_v to avoid visit it again.
 - iv. If H_v is empty then *state* = *redo*
- 2. Else
 - (a) This point of the execution is reached when the evaluation of a goal fails at the current MARlet. The step ii) of the algorithm selected a c_j from the local database for proving t_i . This selection was the source of the failure. Therefore, MoviLog simply restores the clause by reversing the effects of applying the substitutions in V , selects another clause c_j , sets *state* = *call* and jumps to i).
 - (b) If there are no more choices left in P , this implies that it is not possible to prove t_i from the local database. Therefore the top of the stack is popped and the algorithm returns false. This may require migrating the BrainLet to its origin.

Distributed Backtracking and Consistency Issues The RMF mobility model generates several tradeoffs related the standard Prolog execution semantics. Backtracking is one of them. When a Brainlet moves around several places, many backtracking points can be left untried, and the question is how the backtracking mechanism should proceed. The solution adopted by MoviLog at the current version resides in the PNS agents. These agents provide a sequential view of the multiple choice points that is used by the routing mechanism to go through the distributed execution tree.

Also the evaluation of MoviLog code in a distributed manner may lead to inconsistencies. For example, MARlets can enter or leave the system, may alter their protocol clauses or modify their databases. At this moment, MoviLog defines a policy that consists on updating the local view of a BrainLet when it arrives to a host. This involves automatically querying the PNS agents to obtain a list of MARlets implementing a given protocol clause and querying the current MARlet in order to obtain a list of clauses matching the protocol clause being evaluated.

4 Experimental Results

In this section we report the results obtained with an application implemented by using MoviLog, μ Code [8] (a Java-based framework for mobile agents) and Jinni [12] (a Prolog-based language with support for strong mobility).

The application consists of a number of customer agents that are able to select and buy articles offered by sellers based on users' preferences. Both, customers

and sellers reside in different hosts of a network. In this example, customers are ordered to buy books that has to satisfy a number of preferences such as price, author, subject, etc.

The implementation of the application with MoviLog using RMF was straightforward (39 lines of code). On the other hand, to develop the application by using μ Code we had to provide support for representing and managing users' preferences. As a result the total size of the application was 22605 lines of code. Finally, the Jinni implementation was easier, although not as easy as with MoviLog, due to the necessity of managing agents' code and data closure by hand. The size of the source code in this case was 353 lines. It is worth noting that MoviLog provides powerful abstractions for rapidly developing intelligent and mobile agents. On the other hand, the others platforms are more general, thus their usage for building intelligent agents require more effort.

We tested the implementations on three Pentium III 850 Mhz with 128 MB RAM, running Linux and Sun JDK 1.3.1. To compare the performance of the implementations we distributed a database containing books in the three computers. We ran the agents with a database of 1 KB, 600 KB and 1.6 GB. For each database we ran two test cases varying the user's preferences in order to verify the influence of the number of matched books (state that an agent has to move) on the total running time. On each respective test case the user's preferences matched 0 and 5 books (1 KB database), 3 and 1024 books (600 KB database, 4004 books), and 2 and 1263 (1.6 GB, 11135 books approx.). We ran each test case 5 times and measured the running time. Fig. 3 (right) shows the average running time as a function of the size of the database and the number of products found. In all the cases, the standard deviation is less than 5%.

On a second battery of tests we measured the network traffic generated by the agents using the complete database (1.6 GB, 11135 books approx.) distributed across three hosts. Fig. 3 (left) shows the network traffic measured in packets versus the number of books that matched the user's preferences. From the figure we can conclude that MoviLog and its RMF do not affect negatively neither the performance nor the network traffic, while considerably reducing the development effort. The next section discusses previous work related to MoviLog.

5 Related Work

At present, Java is the most commonly used language for the development of mobile agent applications. Aglets [7], Ajanta [13] and μ Code [8] are examples of Java-based mobile agent systems. These systems provide a weak mobility model, forcing a less elegant and more difficult to maintain programming style [10]. Recent works such as NOMADS [11] and WASP [3] extended the Java Virtual Machine (JVM) to support strong mobility. Despite the advantages of strong mobility, these extended JVM do not share some well known features of the standard JVM, such as its ubiquity, portability and compatibility across different platforms.

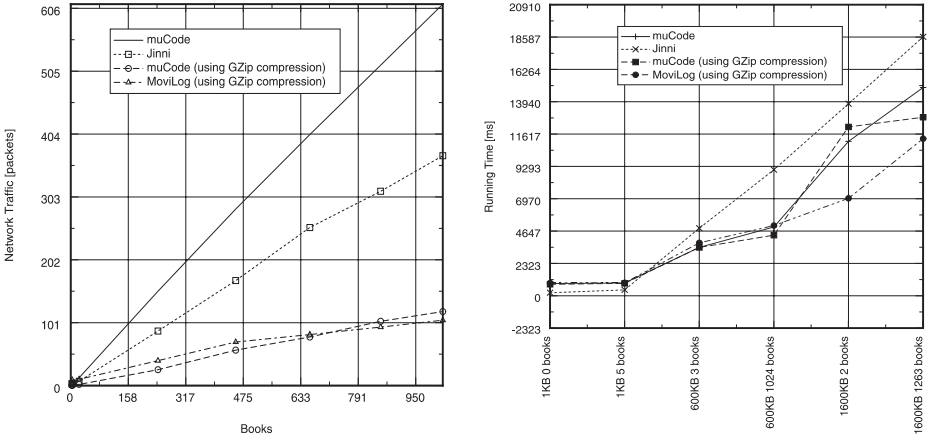


Fig. 3. Performance Comparisons

The logic programming paradigm represents an appropriate alternative to manage agents' mental attitudes. Examples of languages based on it are Jinni [12] and Mozart/Oz [6]. Jinni [12] is based on a limited subset of Prolog. Jinni supports strong mobility. However, the language lacks adequate support for mobile agents since its notion of code and data closure is limited to the currently executing goal. As a consequence developers have to program mechanisms for saving and restoring an agent's code and data. Mozart [6] is a multi-paradigm language combining objects, functions and constraint logic programming based on a subset of Prolog. Though the language provides some facilities such as distributed scope and communication channels that are useful for developing distributed applications, it only provides rudimentary support for mobile agents. Despite this shortcoming, Mozart offers a clean and easy syntax for developing distributed applications with little effort.

The main differences between MovILog and other platforms are its support for RMF, which reduces development effort by automatizing some decisions about mobility, and its multi-paradigm syntax, which provides mechanisms for developing intelligent agents with knowledge representation and reasoning capabilities. MovILog reduces and simplifies the effort of mobile agent development, while being as fast as any Java-based platform.

6 Conclusions

Intelligent mobile agents represent one of the most challenging research areas due to the different factors and technologies involved in its development. Strong mobility and inference mechanisms are, undoubtedly, two important features that an effective platform should provide. MovILog represents a step forward

in that direction. The main contribution of our work is the *reactive mobility by failure* concept. It enables the development of agents using common Prolog programming style, making in it easier thus for Prolog programmers. This concept, combined with proactive mobility mechanisms, also provides a powerful tool for developing intelligent Internet agents. At the moment, MoviLog is an academic prototype, which has shown an acceptable performance. Further research is needed about this topic, as well as, the potential consistency problems that can arise in more complex applications. However, these aspects also open exciting research challenges that can lead to more powerful platforms to build agent systems.

References

- [1] A. Amandi, A. Zunino, and R. Iturregui. Multi-paradigm languages supporting multi-agent development. In *Multi-Agent System Engineering, MAAMAW'99*, volume 1647 of *LNAI*, pages 128–139. Springer-Verlag, June 1999. 164
- [2] Jeffrey M. Bradshaw. *Software Agents*. AAAI Press, Menlo Park, USA, 1997. 163, 168
- [3] S. Fünfroeken and F. Mattern. Mobile Agents as an Architectural Concept for Internet-based Distributed Applications - The WASP Project Approach. In *Proceedings of KiVS'99*, 1999. 171
- [4] A. Garcia, C. Chavez, O. Silva, V. Silva, and C. Lucena. Promoting Advanced Separation of Concerns in Intra-Agent and Inter-Agent Software Engineering. In *Workshop on Advanced Separation of Concerns in Object-Oriented Systems (ASoC) at OOPSLA'2001*, 2001. 168
- [5] R.S. Gray, G. Cybenko, D. Kotz, and D. Rus. Mobile agents: Motivations and state of the art. In Jeffrey Bradshaw, editor, *Handbook of Agent Technology*. AAAI/MIT Press, 2001. 163
- [6] S. Haridi, P. Van Roy, and G. Smolka. An overview of the design of Distributed Oz. In *Proceedings of the Second International Symposium on Parallel Symbolic Computation*, 1997. 172
- [7] D.B. Lange and M. Oshima. *Programming and Deploying Mobile Agents with Java Aglets*. Addison-Wesley, Reading, MA, USA, September 1998. 164, 171
- [8] G.P. Picco. μ Code: A Lightweight and Flexible Mobile Code Toolkit. In *Proceedings of the 2nd International Workshop on Mobile Agents*, pages 160–171, 1998. 170, 171
- [9] G.P. Picco, A. Carzaniga, and G. Vigna. Designing distributed applications with mobile code paradigms. In R. Taylor, editor, *Proceedings of the 19th ICSE*, pages 22–32, 1997. 163, 164
- [10] A. Rodriguez Silva, A. Romao, D. Deugo, and M. Mira da Silva. Towards a Reference Model for Surveying Mobile Agent Systems. *Autonomous Agents and Multi-Agent Systems*, 4(3):187–231, September 2001. 164, 171
- [11] N. Suri, J.M. Bradshaw, M.R. Breedy, P.T. Groth, G.A. Hill, R. Jeffers, and T.S. Mitrovich. An Overview of the NOMADS Mobile Agent System. In *6th ECOOP Workshop on Mobile Object Systems: Operating System Support, Security and Programming Languages*, 2000. 171
- [12] Paul Tarau. Jinni: a lightweight java-based logic engine for internet programming. In *Proceedings of JICSLP'98 Implementation of LP languages Workshop*, June 1998. 170, 172

- [13] A.R. Tripathi, N.M. Karnik, T. Ahmed, R.D. Singh, A. Prakash, V. Kakani, M.K. Vora, and M. Pathak. Design of the Ajanta System for Mobile Agent Programming. *Journal of Systems and Software*, 2002. to appear. 164, 171
- [14] M. Wooldridge and N.R. Jennings. Pitfalls of agent-oriented development. In *Proceedings of the 2nd International Conference on Autonomous Agents*, pages 385–391, May 9–13 1998. 164

Dynamic Social Knowledge: The Timing Evidence

Augusto Loureiro da Costa¹ and Guilherme Bittencourt²

¹ Núcleo de Pesquisa
em Redes de Computadores, Universidade Salvador
40 171 100 Salvador - Ba, Brazil
tel. +55 71 203 2684
loureiro@unifacs.br

² Departamento de Automação e Sistemas
Universidade Federal de Santa Catarina
88040-900 - Florianópolis - SC, Brazil
tel. +55 48 331 9202
gb@lcmi.ufsc.br

Abstract. A comparative evaluation among the Contract Net Protocol, the Coalition Based on Dependence* and the Dynamic Social Knowledge cooperation strategies is presented in this paper. This evaluation uses the experimental results from a soft real-time application extracted from the robot soccer problem and focuses on the cooperation convergence time and on the amount of exchanged messages. Also one new concept called Plan Set is presented in this paper.

Keywords: Cognitive Multi-Agent, Multi-Agent Cooperation.

1 Introduction

Autonomous agents have a high degree of self determination, they can decide by themselves, when and under which conditions an action should be performed. There are many cases in which autonomous agents have to interact with other agents to achieve common goals, e.g., if it wants to perform an action to which the needed skills are not available in that agent, or if there is an interdependence among the agents. This interaction is done in the sense to find another agent to participate in the agent's actions, to modify a set of planned actions, or to achieve an agreement about join actions. Once that this agent do not have a direct control under the others, it is necessary to use a cooperation strategy to join others autonomous agent to perform a given cooperative action.

Several cooperation strategies have been proposed to support *Multi-Agent Systems* (MAS), most of them support a single method of cooperation, defined by the set of allowed negotiation steps. Two of the most used cooperation strategies are the *Contract Net Protocol (CNP)* [11] and the *Coalition Based on Dependence (CBD)* [9]. This last strategy gave rise to various works on negotiation strategies

to join agents into a coalition, e.g., *Service-Oriented Negotiation Model between Autonomous Agents* [10].

In a previous paper, a cooperation strategy called *Dynamic Social Knowledge (DSK)* [3] was proposed, that shares some features of both, Contract Net Protocol and Coalition Based on Dependence. It also introduces some new concepts and makes intensive use of the rule-based representation. The most important contributions of this strategy can be seen in *Open Autonomous Cognitive MAS* [8], with real-time restrictions, that accept the best effort approach. In this kind of agent community, the number of agents able to cooperate and the environment features can change dynamically. An analytical comparative evaluation among the Contract Net Protocol, the Coalition Based on Dependence* and the Dynamic Social Knowledge cooperation strategies keeping the evaluation focus on the number of exchanged messages in the cooperation process was presented in [5]. That evaluation pointed to Dynamic Social Knowledge as a good balance between the amount of social knowledge used to drive the cooperation strategy and the amount of interaction cycles involved in the cooperation process. On the other hand that evaluation do not cover computational effort to implement each one of the cooperation strategies neither the real-time response.

A new comparative evaluation among the CNP, CBD* and DSK cooperation strategies, keeping the focus on the experimental results from a multi-agent system implementation of each one of the mentioned cooperation strategy and comparing the convergence time and the amount of exchanged messages, is presented in this paper. This new evaluation covers the missing aspects of the previous one, allowing the evaluation of the computational effort associated with the implementations of the strategies and their real-time response.

Section 2 briefly describes a situation extracted from a robot team training section in the Soccer Server Simulator, used as the environment, and the respective environmental condition adopted for this evaluation. The multi-agent systems implementation for the evaluated cooperation strategies is presented in Section 3. The next three Sections – 4, 5 and 6 – present, respectively, the implementations of the CNP, CBD* and DSK cooperation strategies in a C++ Object Oriented library. This library is aimed to help multi-agent systems implementations under soft real-time restrictions and using the best effort approach [2] and is called Expert-Coop++. Section 7 presents the obtained results of the comparative evaluation. Finally, conclusions and future work are discussed in Section 8.

2 The Environment

The Soccer Server Simulator was chosen as the experimental environment and one consideration about agent communication was done: the agents were allowed to communicate, using peer-to-peer mode, with out limitation using Inet Socket Domain. A situation extracted from the robot soccer problem was chosen to evaluate the cooperation convergence time and amount of message exchanged by the agents. The Soccer Server simulator was driven to the state A in t_0 ,

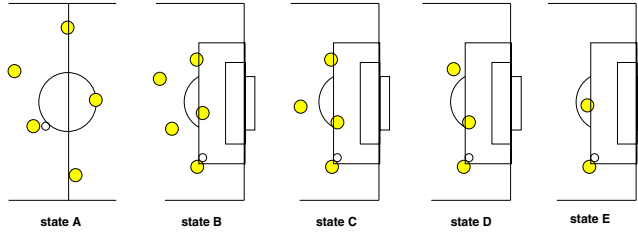


Fig. 1. A situation chosen from the robot soccer problem

depicted in figure 1, where the team has the ball control and the chosen goal is to perform a right hand side attack play-set.

Starting at *state A*, five possible states (*B*, *C*, *D*, *E*), satisfies the desired goal (see 1). The agents involved in the cooperation process know the plans that can drive the game to states (*B*, *C*, *D*, *E*). Depending on the available player to perform the desired goal, the agents will interact trying to converge to a shared plan able to drive the game to one of the states (*B*, *C*, *D*, *E*). The optimal situation, happens when the agents are able to execute a plan able to drive the game to *state A*. On the other hand the worst situation happens when the agents are able to execute just the plan able to drive the game to *state E*. For this evaluation, the optimal and the worst situations are taken in account.

- **Case 1:** this goal can be achieved by the optimal plan, involving five robots, to drive the game to *state B* in $t_0 + \Delta_1 t$. One player should drive the ball through the field right hand side, two others players should move to the main area entrance, and two more players should approach the penalty area.
- **Case 2:** Just the last plan is feasible to achieve this goal, involving two robots, to drive the game to *state E* in $t_0 + \Delta_2 t$. One player driving the ball through the field right hand side and another player in the penalty area entrance.

3 Mult-agent Systems Implementation

A five players robot team was implemented for each one of the evaluated cooperation CNP, CBD* or DSK. These robot teams, *CNP-Team*, *CBD*-team* and *DSK-Team* respectively, are multi-agent system, the agent architecture is the Concurrent Autonomous Agent [4], available in Expert-Coop++ library.

The Concurrent Autonomous Agent is Based on the Generic Model for Cognitive Agents [1], it implements an autonomous agent architecture with three decision levels, *Reactive*, *Instinctive* and *Cognitive*, according to a concurrent approach. Each decision level is implemented in a process: *Interface*, *Coordinator* and *Expert*.

The Interface process implements a collection of reactive behaviors already available in the Expert-Coop++ library for the Soccer Server Simulation players. The Coordinator process implements the Instinctive level, responsible for

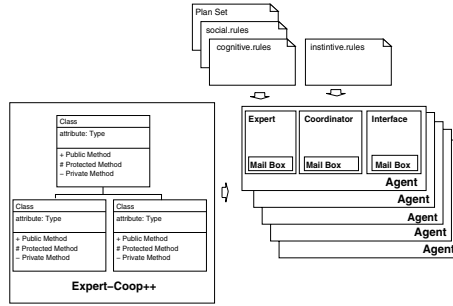


Fig. 2. Agents implementation

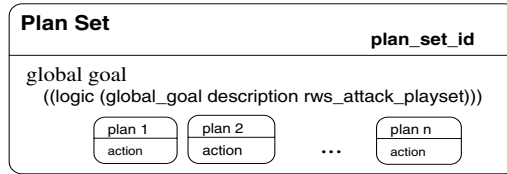


Fig. 3. Planset for global goal right wing side attack playset

recognition of the current world state, for choosing the adequate behavior for the current world state and the local goal and also responsible for updating the symbolic information used by the cognitive level. The Coordinator process encapsulates a single cycle knowledge-based system and a rules file to provide the guide lines is required.

The cognitive level, implemented in the Expert process, encapsulates two knowledge bases, local base and social base, which requires a rule file for each base, and one inference engine. The local knowledge base is responsible for handling the symbolic information sent from the Instinctive level and generating local goals according to this symbolic information and the global goals provided by the social base. The social knowledge needed by an agent to take part in a cooperation process is provided by the social base available. The social base introduces a new data structure handled by the social base called *Plan Set*.

- **Definition 1:** A *Plan Set* P_i is a data structure that contain: a string identification *plan – set-id*, a global goal g_i , and list of plans $p_1, p_2, \dots, p_w$ that can satisfy g_i , ranked according to the optimality criteria where p_1 is the optimal plan to achieve g_i .

A Plan Set list also needs to be supplied to the agent in a file handled by the Expert process. In this implementation one Plan Set P_i shown in figure 3 was supplied to the agents.

The five cooperative actions which integrate the optimal plan p_1 are represented by the following logic pattern:

Plan 1 ((logic (global_goal description rws_attack_playset))) actions ((logic (drive_ball_rws agent XXX-Team_7))) ((logic (main_area_position agent XXX-Team_9))) ((logic (main_area_position agent XXX-Team_11))) ((logic (cover_position agent XXX-Team_8))) ((logic (cover_position agent XXX-Team_10)))	Plan 4 ((logic (global_goal description rws_attack_playset))) actions ((logic (drive_ball_rws agent XXX-Team_Y))) ((logic (main_area_position agent XXX-Team_Z)))

Fig. 4. The optimal plan p_1 and the critical plan p_4 for g_i

The optimal plan p_1 is the first one stored in plan list head, and can drive the game to the desire state B in $t_0 + \Delta_1 t$, *Case 1* (see figure 4). On the other hand there is a plan p_4 , stored in the last position of the plan list, that allow g_i to be achieved driving the game to state E in $t_0 + \Delta_2$. This plan have just two cooperative actions (see figure 4).

Other alternative plans to achieve g_i according the available agents and their skills p_2 and p_3 which allow the agent society to achieve intermediate states C in $t_0 + \Delta_3 t$ and state D in $t_0 + \Delta_4 t$, complete the plan list, stored in P_i

The social base contains: the current global goal g_a , the potential agents to integrate the cooperation process, the active plan which allows the society to achieve g_a , the active *Plan Set* which contains all possible plans to achieve g_a , a list of *Plan Set*, loaded from a file, defines which global goals the agent is able to open and manage a cooperation process, and finally the cooperation strategies CNP, CBD* and DSK.

4 CNP Implementation in Expert-Coop++

The CNP implementation in Expert-Coop++ library is a social base called *CNP_Base*, which provides the follow contents: *Plan Set list*, *plan list*, *contract list*, *active plan*, *awarded contract list* and *active contract*. It also provides a method which implements the CNP cooperation strategy. The cooperation process begins with a global goal request from local base to social base, this request leads the agent to assume the *manager* role in CNP for this cooperation process. It is considered for convergence time measurement at instant t_0 . The agent, broadcasts to potential agents, $a_1, a_2, \dots, a_n$, the requested global goal g_i , selects the Plan Set P_i , related to g_i , the optimal plan $p_1 \in P_i$ becomes the active plan, and for each cooperative action in p_1 a contract c_i is open and stored in a contract list. The first contract in the contract list becomes active and it is broadcasted to potential agents $a_1, a_2, \dots, a_n$. This active contract is kept open until:

1. A satisfactory proposal pr_i have been received. Then c_i is awarded and the agent a_i who sent pr_i notified. The awarded contract c_i is stored in the awarded contract list, and the next contract from contact list became active.
2. All of agents have already replied the contact c_i and none of the received proposals satisfies the active contact. In this case the active plan fails, it is aborted and the next plan from the plan list becomes the active plan.

The convergence is achieved when all contracts from the same plan p_i are awarded. On the other hand when all plans in P_i fails the convergence can not be achieved.

5 CBD* Implementation in Expert-Coop++

Looking forward to attempt to some assumption presented in high dynamical environment, a slighted modification of Coalition Based on Dependence [9, 6] called CBD* [5] was adopted in this work. The CBD* implementation in Expert-Coop++ library is a social base called *CBD*_Base*, which provides the same *CNP_Base*: *Plan Set list*, *plan list*, *contract list*, *active plan*, *awarded contract list*, *active contract*, *potential partner list*. But the cooperation method implements CBD* cooperation strategy. Like in CNP the cooperation process begins with a global goal request leading this agent to assume *active agent* role in CBD* for this cooperation process. The active agent broadcasts to potential partners, the agents $a_1, a_2, \dots, a_n$, the requested global goal g_i , selects the Plan Set P_i , related to g_i , the optimal plan $p_1 \in P_i$ becomes the active plan, and for each cooperative action in p_1 a contract c_i is open and stored in a contract list.

Once received a goal broadcast, the potential partners, the agents $a_1, a_2, \dots, a_n$, broadcast their impressions about the global goal g_i expressing their availability and interest in g_i . The agents which express their availability and interests about g_i will be included in the potential partner list $a_1, a_2, \dots, a_p$.

Then, the first contract in the contract list becomes active the active agent tries a coalition with the agents included in the potential partner list $a_1, a_2, \dots, a_p$. At first, the active agent tries the coalition with the agent pointed by the active plan. If this agent refuses or it is not in the potential partner list, the coalition is tried with other agent. The negotiation cycle begins trying to allocate the contracts from the contract list to the potential partners. During negotiation cycle the following situations can happen:

1. the coalition for the active contract c_i succeeds, then c_i is closed, the partner agent is notified. The awarded contract c_i is stored in the awarded contract list, and the next contract from contract list becomes active.
2. All of potential partners $a_1, a_2, \dots, a_p$ have refused the coalition for c_i leading the active plan to fail. In this case the active plan fails, it is aborted and the next plan from plan list became the active plan.

In CBD*, when a new plan becomes active, the agent first checks whether this new plan has cooperative actions that have already been stored in the awarded contract list. In affirmative case, the new contracts will be open just for that cooperative action in awarded contract list. The convergence is achieved when all contracts from the same plan p_i are awarded. On the other hand when all plans in P_i fail, the convergence can not be achieved.

6 DSK Implementation in Expert-Coop++

The DSK implementation in Expert-Coop++ library is a social base called *DSK_Base*, which provides the follow contents: *Plan Set list*, *Plan Set*, *Contract Frame list*, *plan list*, *primer plan* to stored the optimal plan and *awarded contract list*. It also provides an inference engine, a rule base which rules are automatically generated from the Plan Set file, the Dynamic Social Knowledge Base *dsk_base*, and the method that implements the DSK cooperation strategy.

The cooperation process begins with a global goal request like CNP and CBD*. The agent, broadcasts to potential partners, the agents $a_1, a_2, \dots, a_n$, the requested global goal g_i , selects the Plan Set P_i , related to g_i . Then a Contract Frame C_i is created for goal g_i which contains a *contract-set* with all of different actions presented in the Plan Set P_i . It means that, for a cooperative action that appears more than once, just one contract is open and any time that it appears again the contract multiplicity is increased. Once created C_i all contracts stored in the *contract-set* will be announced together and the agent begins to receive proposals until:

- **Direct Award:** all the contracts that belongs to the optimal plan have received satisfactory proposal. Then, all of these contacts are awarded leading the cooperation process to the convergence.
- **Built DSK base:** the agent has received proposals to all contract, but the Direct Award is not possible. In this situation all the already received proposals are used to built a *DSK base* that together with the rule base and the inference engine will decide which plan $p_i \in P_i$ will be performed leading the cooperation process to the convergence without extra messages exchange, or none of $p_i \in P_i$ can be performed the cooperation fails.

7 Results and Comparative Evaluation

The implemented multi-agent systems *CNP-Team*, *CBD*-team* and *DSK-Team* were connected to the Soccer Server Simulator and the game state driven to *state A* in t_0 (described in section 2) for both cases considered in section 2. A Athlon 700 Mhz with 196 MB RAM was used for this experiment and the time between the global goal broadcast and the convergence message was measured by the agent that have open the cooperation process using the Ansi C function *clock()*. The experimental results shown in this section are expressed in miliseconds and the instant 0 consists in the instant which a cooperation request message was treated by the agent social base. A convergence message, broadcasted by the agent responsible for the cooperation process management was introduced to explicit the convergence instant.

The message exchanged by the multi-agent systems *CNP-Team*, *CBD*-team* and *DSK-Team* submitted to *Case 1* and *Case 2* during the cooperation process are shown in figure 5 and figure 6, respectively.

For Case 1 the best convergence time was presented by CBD*, 60 ms followed by DSK 100 ms and finally CNP 130 ms. An important point is that the

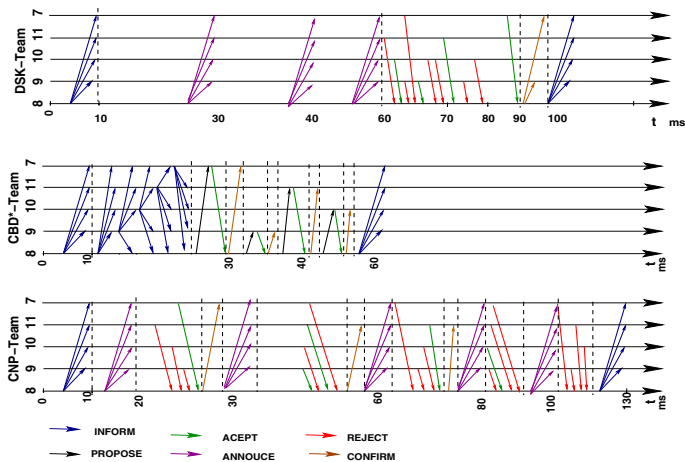


Fig. 5. Messages exchanged in case 1 by CNP-Team, CBD*-Team and DSK-Team

CBD-team* presented the considered optimal convergence process, when the first coalition tries to succeed for all cooperative action mentioned in [5]. But for *CNP-Team* and *DSK-Team* the considered optimal convergence process, when for all contacts the first received proposal is satisfactory also mentioned in [5], do not happen, because both at CNP and at DSK it is not possible to control the order of the arriving messages. For Case 2 the best convergence time was presented by DSK *130 ms* while CBD* presented *150 ms* convergence time and CNP *160 ms*.

The time convergence for mult-agent systems *CNP-Team*, *CBD*-team* and *DSK-Team* submitted to *Case 1* and *Case 2* described in section 2 are presented in table 7.

The smaller amount of messages exchanged during the cooperation process was presented by *CBD** 32 messages, followed by *DSK* 34 messages and finally *CNP* 49 messages. Once more, it is important to remember that the considered optimal convergence process do not happen for *DSK* and *CNP*. For Case 2 the smaller amount of messages exchanged during the cooperation process was presented *DSK* 34 messages, followed by *CBD** 32 messages and finally *CNP* 66 messages. Another important point is that the *DSK* presented the same amount

Table 1. Convergence time, in milleseconds (ms)

Cooperation Strategy	Case 1	Case 2
CNP	130	160
CBD*	60	150
DSK	100	130

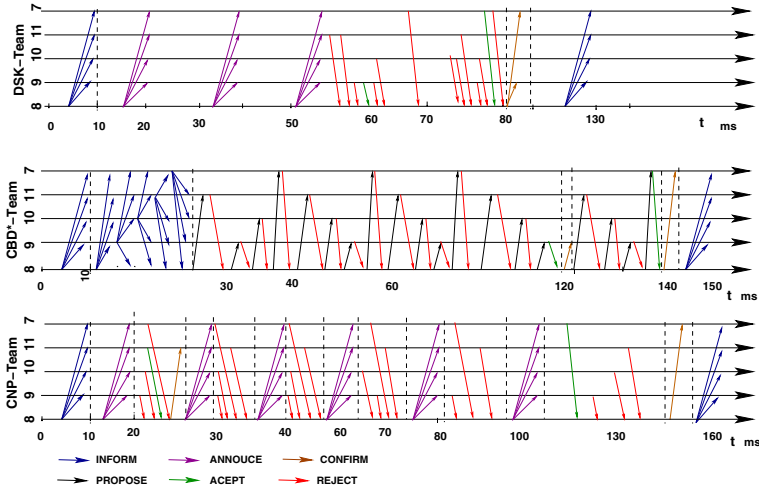


Fig. 6. Messages exchanged in case 2 by CNP-Team, CBD*-Team and DSK-Team

Table 2. Amount of exchanged messages

Cooperation Strategy	Case 1	Case 2
CNP	49	66
CBD*	32	64
DSK	34	34

of message in Case 1 and Case 2. It means that in *DSK* when the Direct Award is not available the convergence can be achieved without extra communication, as mentioned in section 6. This experiment was repeated several times but no changes were verified on convergence time, just the received messages have presented different ranks.

The time stamp mechanism [7] available in Expert-Coop++ was not used in this evaluation. It would allow a total ranking event, but, it would require that all agents receive all messages, this time stamp assumption, would give a considered advantage for CNP and DSK cooperation strategies.

8 Conclusions

The experimental results from the multi-agent system *CNP-Team*, *CBD*-team* and *DSK-Team* implemented using the Expert-Coop++ library, point to Dynamic Social Knowledge cooperation strategy as a good balance between the amount of exchanged messages to drive the cooperation strategy and the convergence time. The use of rule-based inference combined with simultaneous announcements of alternative ways to perform an agent action, allows the agent society to converge faster to a plan, avoiding a sequential search for a feasible

plan which is crucial to reduce both the amount of exchanged messages and the cooperation convergence time. The computational cost to implement DSK cooperation strategies did not cause a significative load, allowing the *DSK-Team* to present short convergences times.

The CBD*-Team kept the advantage to assure that if the optimal solution is available, it will present both the smaller convergence time and smaller amount of exchanged messages. On the other hand, it can not be assured by CNP neither by DSK because the sequence of received messages can not be controled.

The Dynamic Social Knowledge cooperation strategy intend to use in a next future in some real-time multi-agent systems implementation, like collective robotics, internet search and Urban traffic control.

References

- [1] G. Bittencourt. In the quest of the missing link. In *Proceedings of IJCAI 15, Nagoya, Japan, August 23-29*, pages 310–315. Morgan Kaufmann (ISBN 1-55860-480-4), 1997. 177
- [2] A. Burns and A. Wellings. *Real-Time Systems and Programming Languages*. Addison-Wesley, 1997. Secound Edition. 176
- [3] A.L. da Costa and G. Bittencourt. Dynamic social knowledge: A cooperation strategie for cognitive multi-agent systems. *Third International Conference on Multi-Agent Systems (ICMAS'98)*, pages 415–416, Paris, France, July 2-7 1998. IEEE Computer Society. 176
- [4] A.L. da Costa and G. Bittencourt. From a concurrent architecture to a concurrent autonomous agents architecture. *IJCAI'99, Third International Workshop in RoboCup*, pages 85–90, Sweden, Stockholm, July 31 - August 1999. IJCAI Press. 177
- [5] A.L. da Costa and G. Bittencourt. Dynamic social knowledge: A comparative evaluation. *Intenational Join Conference IBREAMIA'2000 / SBIA'2000.*, pages 176–185, Brazil, Atibaia - SP, November 19 - 22 2000. Spring-Verlag, Lecture Notes in Artificial Inteligence, vol. 1952 - Best Paper Track Award. 176, 180, 182
- [6] M. Ito and J.S. Sichman. Dependence based coalition and contract net: A comparative analysis. *Intenational Join Conference IBREAMIA'2000 / SBIA'2000.*, pages 106–115, Brazil, Atibaia - SP, November 19 - 22 2000. Spring-Verlag, Lecture Notes in Artificial Inteligence, vol. 1952. 180
- [7] P. Jalote. *Fault Tolerance in Distributed System*. PTR Prentice Hall, Englewood Cliffs, New Jersey, 1994. 183
- [8] J.S. Sichman. A model for the decision phase of autonomous belief revesion in open multi-agent system. *Journal of Brazilian Computer Society*, 3(1):40–50, March 1996. ISSN 0104-6500. 176
- [9] J.S. Sichman. Depint: Dependence-based coalition formation in an open multi-agent scario. *Journal of Artificial Societies and Social Simulation*, 1:<http://www.soc.surrey.ac.uk/JASS/1/2/3.html>, March 1998. 175, 180
- [10] C. Sierra, Faratin. P., and N.R. Jennings. A service-oriented negotiation model between autonomous agents. In *8th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW-97) International Conference on Computational Linguistic (COLIG)*, pages 15–35, 1997. Ronneby, Sweden. 176

- [11] R. G. Smith. The contract net protocol:high-level communication and control in a distributed problem solving. *IEEE Transactions on Computers*, 29(12):1104–1113, December 1980. 175

Empirical Studies of Neighborhood Shapes in the Massively Parallel Diffusion Model

Sven E. Eklund

Computer Science Department
Dalarna University, SWEDEN
sven.eklund@ieee.org

Abstract. In this paper we empirically determine the settings of the most important parameters for the parallel diffusion model. These parameters are the selection algorithm, the neighbourhood shape and the neighbourhood size.

1 Background

Genetic Algorithms (GA) and Genetic Programming (GP) are groups of stochastic search algorithms which were discovered during the 1960's, inspired by evolutionary biology. Over the past decades GA and GP have proven to work well on a variety of problems with little a-priori information about the search space. However, in order for them to solve hard, human-competitive problems, like those suggested in [12], they require vast amounts of computer power, sometimes involving more than 10^{15-17} operations.

2 Parallel GA

It is a well-known fact that the genetic algorithm is inherently parallel, a fact that could be used to speed up the calculations of GP. The basic algorithm by Holland [10] is very parallel, but also has a frequent need for communication and is based on centralized control, which is not desirable in a parallel implementation.

An *efficient* architecture for GP should of course be optimized for the calculations and communication involved in the algorithm. However, it has also to be *flexible* enough to work efficiently with a variety of applications, which have different function sets. Also, the architecture should be *scalable* so that larger and harder problems can be addressed with more computing hardware.

By distributing independent parts of the genetic algorithm to several processing elements which work in parallel, it is possible to speedup the calculations. Traditionally, the parallel models have been categorized by the method by which the population is handled. The choice between a global and a distributed population is basically a decision on selection pressure, since smaller populations result in faster

(sometimes premature) convergence. However, the choice also has a major effect on the communication need of the algorithm.

2.1 The Farming Model

With a global population the algorithm has direct access to all the individuals in the population, either by a global memory or by some type of communication topology, which connects several distributed memories. This parallel model is often referred to as the farmer-model or the master-slave-model [4]. A central unit, a farmer or master, controls the selection of individuals from the global population and is assisted by workers or slaves that perform the evaluation of the individuals.

This model has been reported to scale badly when the number of processing elements grow, due to the communication overhead of the algorithm [1], [2]. This is however heavily dependent on the ratio between communication time and computation time.

By dividing the population into more independent subpopulations, two alternative parallel models can be identified. Based on the size and number of subpopulations, they are referred to as coarse-grained or fine-grained distributed population models. When dealing with very large populations, which are common in hard, human-competitive problems, these models are better suited since their overall communication capacity scale better with growing population size.

2.2 The Island Model

The coarse-grained, distributed population model, also known as the island model, consists of a number of subpopulations or “demes” that evolve rather independently of each other. With some migration frequency they exchange individuals between each other over a communication topology.

The island model is a very popular parallel model, mainly because it is very easy to implement on a local network with standard workstations (cluster). A major drawback of the island model is that it modifies the basic genetic algorithm and introduces new parameters, for instance the migration policy and the network topology. Today, there exists little or no theory on how to adjust those parameters [5]. Also, a system based on the island model is physically quite large, a fact that exclude many applications.

2.3 The Diffusion Model

The fine-grained distributed population model, often referred to as the diffusion model, cellular GA or massively parallel GA, distributes its individuals evenly over a topology of processing elements or *nodes*. It can be interpreted as a global population laid out on a structure of processing elements, where the spatial distribution of individuals defines the subpopulations. The subpopulations overlap so that every processing node, and its individuals, belongs to several subpopulations, which makes the communication implicit and continuous and enables fit individuals to “diffuse” throughout the population in contrast to the explicit migration of the island model. Selection and genetic operations are only performed within these local neighborhoods [3].

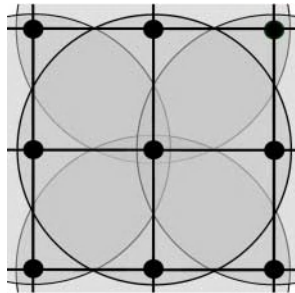


Fig. 1. Part of a 2D topology with a five node neighbourhood

The diffusion model is well suited for VLSI implementation since the nodes are simple, regular and mainly use local communication. Since every node has its own local communication links over the selected topology (1D, 2D, hypercube, etc), the communication bandwidth of the system can be made to scale nicely with a growing number of nodes. Further, the nodes operate synchronously in a SIMD-like manner and have small, distributed memories, which also make the diffusion model suitable for implementation in VLSI.

3 First Implementation

In [7] and [8] it has been reported on a hardware implementation of the diffusion model, capable of evolving more than 20.000 generations per second.

In this first implementation of the architecture every node held two individuals. We used GP as representation, i.e. every individual is a program which, when executed, creates a solution. In this low-level hardware implementation we used linear machine code as program representation. During fitness calculations this code was executed by a CPU embedded in each and every node. The implementation used an X-net topology on a toroidal grid where each neighborhood consisted of nine nodes. For more details on this architecture, please refer to [7] and [8].

Evaluations of this first implementation showed that a better performance-per-gate ratio could be achieved if two CPU:s were implemented in every node (one per individual).

This first implementation was also evaluated at a higher, application level to see how well it worked with real applications. These simulations proved that the algorithm, the GP representation and the structure as a whole worked for real applications. For more details on these high-level simulations, please see [9].

4 Simulations

As mentioned above, the diffusion model does not have to set the explicit migration parameters as in the island model. However, it still has some important parameters

that need to be determined in order to optimize performance. In [3] some parameter settings are suggested, but that system used a traditional GA-representation.

It is the main objectives of these simulations to determine the settings of the most important parameters for the diffusion model. These parameters are; *the selection algorithm, the neighborhood shape and the neighborhood size.*

4.1 Applications

During the simulations three different regression problems were used as test problems; the De Jong test-suite function #1 (1), the classic Rosenbrock function (2) and a function suggested by Nordin (3) [13].

$$f(x_1, x_2, x_3) = \sum_{j=1}^3 x_j^2 \quad (1)$$

$$f(x_1, x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 \quad (2)$$

$$f(x_1, x_2) = 5(x_1^4 + x_2^3) - 3x_1^2 \quad (3)$$

The functions were resampled in 10 random points every 10 generations and the sum of absolute error in function estimation in these points was used as raw fitness measure. The parameters of Table 1 were used throughout all the experiments.

Table 1. Parameter setup

Parameter	Value
Crossover frequency	70 %
Crossover type	2-point
Mutation frequency	30 %
Maximum code length	64 words
Function set :	ADD W, Fi SUB W, Fi MUL W, Fi MOV W, Fi MOV Fi, W MOV const, W
Registers	W, F0-F3
Constants	0 .. 31
Maximum number of generations	10,000
Number of runs	100

4.2 Experiment 1: Selection Algorithm

Traditional selection algorithms such as roulette or ranking selection require global fitness or ranking averages to be calculated, distributed and maintained. This often introduces a communication bottleneck which will limit the performance of a parallel

model. By using a local selection pool this communication problem can be dealt with, however it also means that the basic algorithm is changed. On exception to this is tournament selection which is not dependent on such calculations or communication.

This experiment compared binary tournament selection, roulette selection and ranking selection both with and without local elitism (i.e. only update node with new individual if it performs better than the old one). The population size was 4096 nodes and a one-dimensional neighborhood of 13 nodes was used (1D ring topology). The three selection algorithms only used the sub-population (i.e. the neighborhood) as local selection pool.

4.3 Experiment 2: Neighborhood Shape

When connecting the diffusion nodes to their neighbors one can choose from many different topologies and neighborhood shapes. Which topology or shape should be used? How would randomly selected neighborhoods perform compared with structured, symmetrical ones?

This experiment compared one-dimensional, two-dimensional and random neighborhoods of sizes 5, 9 and 13 nodes. The geometries of the neighborhoods are illustrated in figure 2 (except for the random ones). The white nodes are outside of the neighborhood and the black node is the center node that is being updated. Please note that the structured topologies are closed, making the 1D structure a ring and the 2D structure a toroidal.

The random neighborhood used a randomly selected offset pattern or offset vector to define its neighborhood. This offset vector was kept fixed throughout the run.

Binary tournament with local elitism was used as selection algorithm and population sizes of 1024, 4096, 9216 and 16384 nodes were used in this setup.

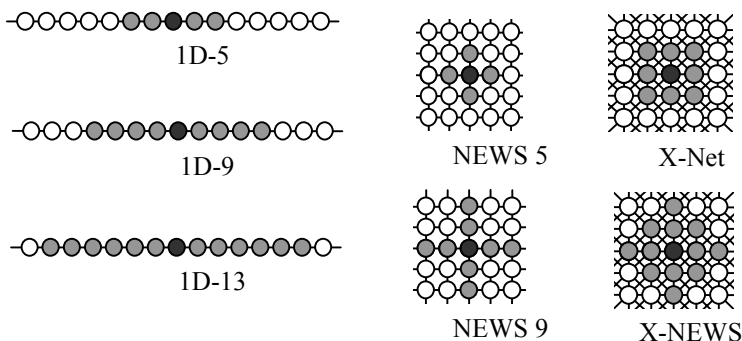


Fig. 2. Neighborhoods used in experiment 2

4.4 Experiment 3: Neighborhood Size

Between the global population and the single-individual-neighborhood there is a spectrum of neighborhood sizes to choose from. Is there an optimal neighborhood size that balances the genetic algorithm between exploration and exploitation? Is the ratio between population size and neighborhood size important?

This experiment required a new set of configurations where selection algorithm and topology were fixed (Binary tournament with local elitism, one dimensional topology). The population size was then varied between 1024, 4096, 9216 and 16384 nodes and the size of the one dimensional neighborhood was varied between 5, 9, 20, 40, 100, 200, 500 and 1000 nodes.

5 Results

5.1 Experiment 1: Selection

From table 2 it is evident that some kind of elitism is needed for the algorithm to work. Without the local elitism the linear ranking selection outperforms both roulette and binary tournament, both of which rarely find the correct solution.

Introducing local elitism makes all the difference for tournament and roulette selection. They are now comparable to the ranking selection, but without its overhead.

These results confirm the more theoretical results by Sarma and DeJong [6].

Table 2. Selection - No. of generations until correct solution found

No local elitism	Tournament	Roulette	Ranking	Average
De Jong	9658	10000	234	6631
Rosenbrock	10000	10000	596	6865
Nordin	10000	9005	410	6472
Average	9886	9668	413	6656

Local elitism	Tournament	Roulette	Ranking	Average
De Jong	203	226	155	195
Rosenbrock	721	1166	648	845
Nordin	792	832	496	707
Average	572	741	433	582

5.2 Experiment 2: Neighborhood Shape

The first obvious and not so surprising observation from table 3 is that larger populations will give fewer generations before convergence occurs. This however, will only translate to shorter wall clock time if parallel hardware is used.

Second, table 3 indicates that a randomly selected neighborhood is outperformed by a structured one (if population sufficiently large and neighborhood not too small). This is probably due to the fact that the symmetry and collaboration between neighboring sub-populations is lost. If node i has node j as neighbor it is not certain that node j has node i as neighbor with the randomly selected neighborhood (the neighborhood shapes in the first column are described in figure 2).

Table 3. Shape – No. of generations

1024 nodes	De Jong	Rosenbrock	Nordin	Average
1D-5	638	4018	2532	2396
NEWS5	460	5521	4632	3538
Random5	1408	5139	7163	4570
1D-9	533	2627	3559	2240
NEWS9	825	5561	4649	3678
X-Net9	654	4710	5076	3480
Random9	945	6466	7844	5085
1D-13	460	5303	4201	3321
X-NEWS	1478	5512	7413	4801
Random13	642	5248	6611	4167
Average	804	5011	5368	3728

4096 nodes	De Jong	Rosenbrock	Nordin	Average
1D-5	358	1458	980	932
NEWS5	202	2598	2091	1630
Random5	213	2662	3269	2048
1D-9	217	1046	512	592
NEWS9	362	2040	1891	1431
X-Net9	136	2111	2233	1493
Random9	359	4193	4009	2854
1D-13	203	721	792	572
X-NEWS	105	2493	4135	2244
Random13	205	3749	5781	3245
Average	236	2307	2569	1704

9216 nodes	De Jong	Rosenbrock	Nordin	Average
1D-5	240	916	716	624
NEWS5	117	434	964	505
Random5	111	373	1915	800
1D-9	185	778	603	522
NEWS9	83	803	1179	688
X-Net9	119	718	658	498
Random9	123	1382	1954	1153
1D-13	152	603	530	428
X-NEWS	113	945	669	576
Random13	138	2346	4124	2203
Average	138	930	1331	800

16384 nodes	De Jong	Rosenbrock	Nordin	Average
1D-5	208	701	515	475
NEWS5	97	350	352	266
Random5	73	283	310	222
1D-9	163	604	422	396
NEWS9	72	416	528	339
X-Net9	92	305	351	249
Random9	83	1718	2465	1422
1D-13	109	482	366	319
X-NEWS	76	317	437	277
Random13	127	361	1823	770
Average	110	554	757	474

It is also indicated by table 3 that a one dimensional neighborhood is better than a two dimensional neighborhood (with the same number of nodes) in smaller populations. Increasing the population size will reduce the difference and in some cases with the largest populations in this experiment, the situation is reversed.

Last, for the neighborhood sizes tested (5, 9 and 13 nodes) it looks like larger 1D neighborhoods will make the algorithm converge in fewer generations.

5.3 Experiment 3: Neighborhood Size

Experiment 2 suggested that larger 1D neighborhoods could be beneficial. In table 4 the average number of generations is reported as a function of both population size and neighborhood size. As can be seen in table 4, this is true up to a certain neighborhood size. Beyond that, an increase in the average number of generations can be seen when the neighborhood grows (lowest number of generations for each column highlighted).

Table 4. Neighborhood Size

De Jong	1024	4096	9216	16384
5	638	358	240	199
9	533	217	185	149
20	431	163	118	92
40	621	152	105	82
100	687	128	85	76
200	663	163	80	67
500	987	260	82	70
1000	1121	366	166	76

Nordin	1024	4096	9216	16384
5	2532	980	716	515
9	3559	512	603	422
20	4957	1205	400	357
40	5138	1159	384	327
100	6054	1612	1329	372
200	7556	2042	1277	609
500	5668	4610	2726	949
1000	7331	4301	3562	3235

Rosenbrock	1024	4096	9216	16384
5	4018	1458	916	701
9	2627	1046	778	604
20	5437	853	628	453
40	5208	1947	432	372
100	6063	1682	672	352
200	5362	2676	479	350
500	6842	3186	1657	270
1000	7176	4366	3016	657

6 Conclusions

Given the regression applications mentioned above we conclude the following: Using local elitism during selection in the diffusion model one can choose a selection algorithm that, for instance, is easy to implement in hardware without losing any performance.

The optimal neighborhood size is dependent on the total population size. If the ratio between neighborhood size and population size is too small the performance will decrease seriously.

Choosing the best neighborhood shape also seems dependent on the total population size, even if the neighborhood size is kept constant. A higher dimensional shape will spread fit individuals faster than a lower dimensional shape and will therefore require a larger population.

References

1. Abramson, D., & Abela, J., "A Parallel Genetic Algorithm for Solving the School Timetabling Problem", In Proceedings of the Fifteenth Australian Computer Science Conference (ACSC-15), Volume 14, pp 1-11, 1992.
2. Abramson, D., Mills, G., & Perkins, S., "Parallelization of a Genetic Algorithm for the Computation of Efficient Train Schedules", Proceedings of the 1993 Parallel Computing and Transputers Conference, pp 139-149, 1993.
3. Baluja, S., "A Massively Distributed Parallel Genetic Algorithm (mdpGA)", CMU-CS-92-166R, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1992.
4. Cantú-Paz, E., "Designing Efficient Master-Slave Parallel Genetic Algorithms", IlliGAL Report No. 97004, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1997.
5. Cantú-Paz, E., "A Survey of Parallel Genetic Algorithms", Department of Computer Science, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, 1998.
6. DeJong, K., Sarma, J., "On Decentralizing Selection Algorithms", Proceedings of the 6:th International Conference on Genetic Algorithms, pp 17-23. Morgan Kaufmann, 1995.

7. Eklund, S., "A Massively Parallel GP Architecture", EuroGen2001, Athens, 2001.
8. Eklund, S., "A Massively Parallel Architecture for Linear Machine Code Genetic Programming", ICES 2001, Tokyo, 2001.
9. Eklund, S., "A Massively Parallel GP Engine in VLSI ", Congress on Evolutionary Computing, CEC2002, Honolulu, 2002.
10. Holland, J. H., "Adaptation in Natural and Artificial Systems", The University of Michigan Press, Ann Harbor, 1975.
11. Koza, J., "Genetic Programming: On the Programming of Computers by Means of Natural Selection", MIT Press, Cambridge, MA, 1992.
12. Koza, J., Bennett III, F., Shipman, J., Stiffelman, O., "Building a Parallel Computer System for \$18,000 that Performs a Half Peta-Flop per Day", Proceedings of the Genetic and Evolutionary Computation Conference, pp 1484-1490, 1999.
13. Nordin, P., Hoffmann, F., Francone, F., Brameier, M., Banzhaf, W., "AIM-GP and Parallelism", Proceedings of the Congress on Evolutionary Computation, pp 1059-1066, 1999.

Ant-ViBRA: A Swarm Intelligence Approach to Learn Task Coordination

Reinaldo A. C. Bianchi and Anna H. R. Costa

Laboratório de Técnicas Inteligentes - LTI/PCS
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, 158. 05508-900 São Paulo - SP, Brazil
{reinaldo.bianchi,anna.reali}@poli.usp.br
<http://www.lti.pcs.usp.br/>

Abstract. In this work we propose the Ant-ViBRA system, which uses a Swarm Intelligence Algorithm that combines a Reinforcement Learning (RL) approach with Heuristic Search in order to coordinate agent actions in a Multi Agent System. The goal of Ant-ViBRA is to create plans that minimize the execution time of assembly tasks. To achieve this goal, a swarm algorithm called the Ant Colony System algorithm (ACS) was modified to be able to cope with planning when several agents are involved in a combinatorial optimization problem where interleaved execution is needed. Aiming at the reduction of the learning time, Ant-ViBRA uses *a priori* domain knowledge to decompose the assembly problem into subtasks and to define the relationship between actions and states based on the interactions among subtasks. Ant-ViBRA was applied to the domain of visually guided assembly tasks performed by a manipulator working in an assembly cell. Results acquired using Ant-ViBRA are encouraging and show that the combination of RL, Heuristic Search and the use of explicit domain knowledge presents better results than any of the techniques alone.

1 Introduction

In the last years the use of Swarm Intelligence for solving several kinds of problems has attracted an increasing attention of the AI community [1, 2, 3]. It is an approach that studies the emergence of collective intelligence in groups of simple agents, and emphasizes the flexibility, robustness, distributedness, autonomy and direct or indirect interactions among agents.

As a promising way of designing intelligent systems, researchers are applying this technique to solve problems such as: communication networks, combinatorial optimization, robotics, on-line learning to achieve robot coordination, adaptative task allocation and data clustering.

The purpose of this work is to use a Swarm Algorithm that combines Reinforcement Learning (RL) approach with Heuristic Search to:

- coordinate agent actions in a Multi Agent System (MAS) used in an assembly domain, creating plans that minimize the execution time, by reducing the number of movements executed by a robotic manipulator.

- reduce the learning time of each new plan.
- adapt to new domain configurations.

To be able to learn the best assembly plan in the shortest possible time a well known Swarm Algorithm – the Ant Colony System (ACS) Algorithm [4] – was adapted to be able to cope with planning when several agents are involved. The ACS algorithm is a learning algorithm based on the metaphor of ant colonies and was initially proposed to solve the Traveling Salesman Problem (TSP), where several ants are allowed to travel between cities, and the path of the ant that have the shortest length is reinforced. The ACS is a combination of distributed algorithms and Q-Learning [11], a well known RL algorithm. It is considered one of the faster algorithms to solve TSP problems [4] and has been successfully applied to several optimization problems, such as Asymmetric TSPs, Network and Vehicle Routing and Graph Coloring.

Aiming at the reduction of the learning time, we also propose the use of *a priori* domain knowledge to decompose the assembly problem into subtasks and to define the relationship between actions and states based on the interactions among subtasks.

The remainder of this paper is organized as follows. Section 2 reviews some key concepts concerning Swarm Intelligence algorithms and section 3 presents the ACS algorithm. Section 4 describes the assembly task domain used in the experiments. Section 5 describes the proposed approach to solve the assembly problem and section 6 presents the experimental setup, the experiments performed in the simulated domain and the results obtained. Finally, Section 7 summarizes some important points learned from this research and outlines future work.

2 Swarm Intelligence

Based on the social insect metaphor for solving problems, Swarm Intelligence has become an exciting topic to researchers in the last years. [1, 2, 3].

The most common Swarm Methods are based on the observation of ant colonies behavior. In this methods, a set of simple agents, called ants, cooperate to find good solutions to combinatorial optimization problems.

Swarm Intelligence can be viewed as a major new paradigm in control and optimization, and it can be compared to the Artificial Neural Network (ANN) paradigm. “An ant colony is a ‘connectionist’ system, that is, one in which individual units are connected to each other according to a certain pattern” [2]. Some differences that can be noted between ANNs and Swarm Algorithms are [2]: the mobility of the units, which can be a mobile robot or a Softbot moving on the Internet; the dynamic nature of the connectivity pattern; the use of feedback from the environment as a medium of co-ordination and communication; and the use of pheromone – ants that discover new paths leave traces, which informs the other ants whether the path is a good one or not – which facilitates the design of distributed optimization systems.

Researchers are applying Swarm Intelligence techniques in the most varied fields, from automation systems to the management of production processes. Some of them are:

- Routing problems [10]: using the Swarm Intelligence paradigm it is possible to feed artificial ants to communications networks, so that they can identify congested nodes. For example, if an ant has been delayed a long time because it went through a highly congested part of the network, it will update the corresponding routing-table entries with a warning. The use of Ant Algorithms in communication networks or vehicle routing and logistics problems is now called Ant Colony Routing – ACR .
- Combinatorial optimization problems such as the Travelling Salesman Problem [4], and the Quadratic Assignment Problem [7]: techniques to solve these problems were inspired by food retrieval in ants and are called Ant Colony Optimization – ACO.
- In several problems involving robotics, on-line learning to achieve robot coordination and transport [5], Adaptive Task Allocation [6].
- Data Clustering.

In the next section we describe the Ant Colony System Algorithm (ACS), which is an algorithm of the ACO class. ACS is the basis of our proposal.

3 The Ant Colony System Algorithm

The ACS Algorithm is a Swarm Intelligence algorithm proposed by Dorigo and Gambardella [4] for combinatorial optimization based on the observation of ant colonies behavior. It has been applied to various combinatorial optimization problems like the symmetric and asymmetric traveling salesman problems (TSP and ATSP respectively), and the quadratic assignment problem [7]. The ACS can be interpreted as a particular kind of distributed reinforcement learning (RL) technique, in particular a distributed approach applied to Q-learning [11]. In the remaining of this section TSP is used to describe the algorithm.

The most important concept of the ACS is the $\tau(r, s)$, called *pheromone*, which is a positive real value associated to the edge (r, s) in a graph. It is the ACS counterpart of Q-learning Q-values, and indicates how useful it is to move to the city s when in city r . $\tau(r, s)$ values are updated at run time by the artificial ants. The pheromone acts as a memory, allowing the ants to cooperate indirectly.

Another important value is the *heuristic* $\eta(r, s)$ associated to edge (r, s) . It represents an heuristic evaluation of which moves are better. In the TSP $\eta(r, s)$ is the inverse of the distance δ from r to s , $\delta(r, s)$.

An agent k positioned in the city r moves to city s using the following rule, called state transition rule [4]:

$$s = \begin{cases} \arg \max_{u \in J_k(r)} \tau(r, u) \cdot \eta(r, u)^\beta & \text{if } q \leq q_0 \\ S & \text{otherwise} \end{cases} \quad (1)$$

where:

- β is a parameter which weighs the relative importance of the learned pheromone and the heuristic distance values ($\beta > 0$).
- $J_k(r)$ is the list of cities still to be visited by the ant k , where r is the current city. This list is used to constrain agents to visit cities only once.
- q is a value chosen randomly with uniform probability in $[0,1]$ and q_0 ($0 \leq q_0 \leq 1$) is a parameter that defines the exploitation/exploration rate: the higher q_0 the smaller the probability to make a random choice.
- S is a random variable selected according to a probability distribution given by:

$$p_k(r, s) = \begin{cases} \frac{[\tau(r, u)] \cdot [\eta(r, u)]^\beta}{\sum_{u \in J_k(r)} [\tau(r, u)] \cdot [\eta(r, u)]^\beta} & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

This transition rule is meant to favor transition using edges with a large amount of pheromone and which are short.

In order to learn the pheromone values, the ants in ACS update the values of $\tau(r, s)$ in two situations: the local update step and the global update step.

The ACS local updating rule is applied at each step of the construction of the solution, while the ants visit edges and change their pheromone levels using the following rule:

$$\tau(r, s) \leftarrow (1 - \rho) \cdot \tau(r, s) + \rho \cdot \Delta\tau(r, s) \quad (3)$$

where $0 < \rho < 1$ is a parameter, the learning step.

The term $\Delta\tau(r, s)$ can be defined as: $\Delta\tau(r, s) = \gamma \cdot \max_{z \in J_k(s)} \tau(s, z)$. Using this equation the local update rule becomes similar to the Q-learning update, being composed of a reinforcement term and the discounted evaluation of the next state (with γ as a discount factor). The only difference is that the set of available actions in state s , (the set $J_k(s)$) is a function of the previous history of agent k . When the ACS uses this update it is called Ant-Q.

Once the ants have completed the tour, the pheromone level τ is updated by the following global update rule:

$$\tau(r, s) \leftarrow (1 - \alpha)\tau(r, s) + \alpha \cdot \Delta\tau(r, s) \quad (4)$$

where α is the pheromone decay parameter (similar to the discount factor in Q-Learning) and $\Delta\tau(r, s)$ is a delayed reinforcement, usually the inverse of the length of the best tour. The delayed reinforcement is given only to the tour done by the best agent – only the edges belonging to the best tour will receive more pheromones (reinforcement).

The pheromone updating formulas intends to place a greater amount of pheromone on the shortest tours, achieving this by simulating the addition of new pheromone deposited by ants and evaporation.

In short, the system works as follows: after the ants are positioned in initial cities, each ant builds a tour. During the construction of the tour, the local

updating rule is applied and modifies the pheromone level of the edges. When the ants have finished their tours, the global updating rule is applied, modifying again the pheromone levels. This cycle is repeated until no improvement is obtained or a fixed number of iterations were reached. The ACS algorithm is presented below.

The ACS algorithm (in the TSP Problem)

```
Initialize the pheromone table, the ants and the list of cities.
Loop /* an Ant Colony iteration */
  Put each ant at a~starting city.
  Loop /* an ant iteration */
    Chose next city using equation (1).
    Update list Jk of yet to be visited cities for ant k.
    Apply local update to pheromones using equation (3).
  Until (ants have a~complete tour).
Apply global pheromone update using equation (4).
Until (Final Condition is reached).
```

In this work, we propose to use a modified version of the ACS Algorithm in the assembly domain, which is described in the next section.

4 The Application Domain

The assembly domain can be characterized as a complex and reactive planning task, where agents have to generate and execute plans, to coordinate its activities to achieve a common goal, and to perform online resource allocation. The difficulty in the execution of the assembly task rests on possessing adequate image processing and understanding capabilities and appropriately dealing with interruptions and human interactions with the configuration of the work table. This domain has been the subject of previous work [8, 9] in a flexible assembly cell.

In the assembly task, given a number of parts arriving on the table (from a conveyor belt, for example), the goal is to select pieces from the table, clean and pack them. The pieces can have sharp edges as molded metal or plastic objects usually presents during their manufacturing process. To clean a piece means to remove these unwanted edges or other objects that obstruct packing. In this way, there is no need to clean all the pieces before packing them, but only the ones that will be packed and are not clean. In this work, pieces to be packed (and eventually cleaned) are named tenons and the desired place to pack (and eventually clean) are called mortises.

While the main task is being executed, unexpected human interactions can happen. A human can change the table configuration by adding (or removing) new parts to it. In order to avoid collisions, both the cleaning and packing tasks can have their execution interrupted until the work area is free of collision contingencies.

The assembly domain is a typical case of a task that can be decomposed into a set of independent tasks: packing (if a tenon on the table is clean, pick it up with the manipulator and put it on a free mortise); cleaning (if a tenon or mortise have sharp edges, clean it before packing) and collision avoidance. One of the problems to be solved when a task is decomposed into several tasks is how to coordinate the task allocation process in the system.

One possible solution to this problem is to use a fixed, predefined authority structure. Once established that one agent has precedence over another one, the system will always behave in the same way, no matter if it results in an inefficient performance. This solution was adopted in ViBRA - Vision Based Reactive Architecture [8]. The ViBRA architecture proposes that a system can be viewed as a society of Autonomous Agents (AAs), each of them depicting a problem-solving behavior due to its specific competence, and collaborating with each other in order to orchestrate the process of achieving its goals. ViBRA is organized with authority structures and rules of behavior. However, this solution have several drawbacks, e.g., in a real application, if an unwanted object is not preventing a packing action, it is not necessary to perform a previous cleaning action, and the ViBRA authority structure doesn't observe this.

Another solution to the task allocation problem is to use a Reinforcement Learning Algorithm to learn assembly plan, taking into account the packing and the cleaning tasks and thus selecting the best order in which this agents should perform their actions, based on the table configuration perceived by the vision system. This solution was adopted in the L-ViBRA [9], where a control agent using the Q-Learning algorithm was introduced in the agent society. The use of the Q-Learning algorithm in L-ViBRA resulted in a system that was able to create the optimized assembly plan needed, but that was not fast enough in producing these plans. Every time the workspace configuration is changed, the system must learn a new assembly plan. This way, a high performance learning algorithm is needed.

As this routing problem can be modeled as a combinatorial TSP Problem, a new system – the Ant-ViBRA – is proposed by adapting the ACS algorithm to cope with different sub-tasks, and using it to plan the route that minimizes the total amount of displacement done by the manipulator during its movements to perform the assembly task.

The next section describes the proposed adaptation of the ACS Algorithm to the assembly domain.

5 The Ant-ViBRA System

To be able to cope with a combinatorial optimization problem where interleaved execution is needed, the ACS algorithm was modified by introducing: (i) several pheromone tables, one for each operation that the system can perform, and; (ii) an extended $J_k(s, a)$ list, corresponding to the pair state/action that can be applied in the next transition.

A *priori* domain knowledge is intensively used in order to decompose the assembly problem into subtasks, and to define possible interactions among subtasks. Subtasks are related to assembly actions, which can only be applied to different (disjunct) sets of states of the assembly domain.

The assembly task is decomposed into three independent subtasks: packing, cleaning and collision avoidance. Since collision avoidance is an extremely reactive task, its precedence over cleaning and assembly tasks is preserved. This way, only interactions among packing and cleaning are considered. The packing subtask is performed by a sequence of two actions – *Pick-Up* followed by *Put-Down* – and the cleaning subtask applies the action *Clean*. Actions and relations among them are:

- *Pick-Up*: to pick up a tenon. After this operation only the *Put-Down* operation can be used.
- *Put-Down*: to put down a tenon over a free mortise. In the domain, the manipulator never puts down a piece in a place that is not a free mortise. After this operation both *Pick-Up* and *Clean* can be used.
- *Clean*: to clean a tenon or a mortise, removing unwanted material to the trash can and maintaining the manipulator stopped over it. After this operation both *Pick-Up* and *Clean* can be used.

The use of knowledge about the conditions under which every action can be applied reduces the learning time, since it makes explicit which part of the state space must be analyzed before making a state transition.

In the Ant-ViBRA, the pheromone value space is decomposed into three subspaces, each one related to an action, reducing the search space. The pheromone space is discretized in “actual position” (of the manipulator) and “next position” for each action. The assembly workspace configuration perceived by the vision system defines the position of all objects and also the dimensions of the pheromone tables.

The pheromone table corresponding to the *Pick-Up* action has entries “actual position” corresponding to the position of the trash can and of all the mortises, and entries “next position” corresponding to the position of all tenons. This means that to perform a pick-up, the manipulator is initially over a mortise (or the trash can) and will pick up a tenon in another place of the workspace.

In a similar way, the pheromone table corresponding to the *Put-Down* action has entries “actual position” corresponding to the position of the tenons and entries “next position” corresponding to the position of all the mortises. The pheromone table corresponding to the *Clean* action has entries “actual position” corresponding to the position of the trash can and of all the mortises, and entries “next position” corresponding to the position of all tenons and all mortises.

The $J_k(s, a)$ list is an extension of the $J_k(r)$ list described in the ACS. The difference is that the ACS $J_k(r)$ list was used to record the cities to be visited, assuming that the only action possible was to move from city r to one of the cities in the list.

To be able to deal with several actions, the $J_k(s, a)$ list records pairs (*state/actions*), which represent possible actions to be performed at each state.

The Ant-ViBRA algorithm is similar to that presented in the last section, with the following modifications:

- Initialization takes care of several pheromone tables, the ants and the $J_k(s, a)$ list of possible actions to be performed at every state.
- Instead of directly choosing the next state by using the state transition rule (equation 1), the next state is chosen among the possible operations, using the $J_k(s, a)$ list and equation (1).
- The local update is applied to pheromone table of the executed operation.
- When cleaning operations are performed the computation of the distance δ takes into account the distance from the actual position of the manipulator to the tenon or mortise to be cleaned, added by the distance to the trash can.
- At each iteration the list $J_K(s, a)$ is updated, pairs of (*state/actions*) already performed are removed, and new possible pairs (*state/actions*) are added.

The next section presents experiments and results of the implemented system.

6 Experimental Description and Results

Ant-ViBRA was tested in a simulated domain, which is represented by a discrete workspace where each cell in this grid presents one of the following six configurations: one tenon, one mortise, only trash, one tenon with trash on it, one mortise with trash on it, one tenon packed on one mortise, or a free cell.

Experiments were performed considering different numbers of workspace cells, learning successfully action policies in each experiment under the assembly task domain. In order to illustrate the results we present three examples. In all of them, the goal is to find a sequence in which assembly actions should be performed in order to minimize the distance traveled by the manipulator grip during the execution of the assembly task. One iteration finishes when there is no more piece left to be packed, and the learning process stops when the result becomes stable or a maximum number of iterations is reached.

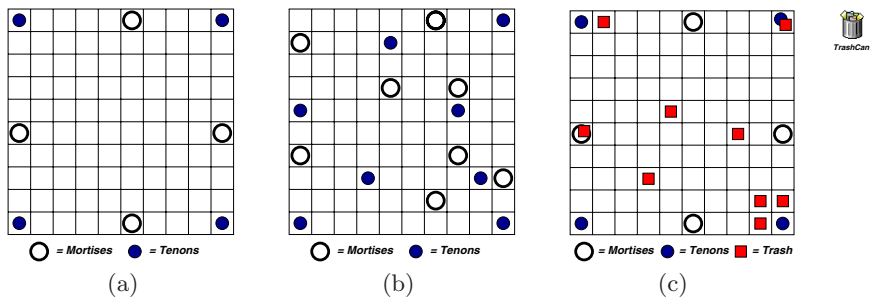


Fig. 1. Configuration of example 1 to 3 (from left to right)

In the first example (figure 1-a) there are initially 4 pieces and 4 tenons on the border of a 10x10 grid. Since there is no trash, the operations that can be performed are to pick up a tenon or put it down over a mortise. The initial (and final) position of the manipulator is over the tenon located at (1,1).

In this example, the modified ACS algorithm took 844 iterations to converge to the optimal solution, which is 36 (the total distance between pieces and tenons). The same problem took 5787 steps to achieve the same result using the Q-learning algorithm. This shows that the combination of both reinforcement learning and heuristics yields good results.

The second example (figure 1-b) is similar to the first one, but now there are 8 tenons and 8 mortises spread in a random disposition on the grid. The initial position of the manipulator is over the tenon located at (10,1). The result (see figure 2-b) is also better than that performed by the Q-learning algorithm.

Finally, example 3 (figure 1-c) presents a configuration where the system must clean some pieces before performing the packing task. The tenons and mortises are on the same position as example 1, but there are trashes that must be removed over the tenon in the position (1, 10) and over the mortise (6, 1). The initial position of the manipulator is over the tenon located at (1,1). The operations are pick up, put down and clean. The clean action moves the manipulator over the position to be cleaned, picks the undesired object and puts it on the trash can, located at position (1, 11). Again, we can see in the result shown in figure 2-c that the modified ACS presents the best result.

In the 3 examples above the parameters used were the same: the local update rule used was the Ant-Q rule (equation 3); the exploitation/exploration rate is 0.9; the learning step ρ is set at 0.1; the discount factor α is 0.3; the maximum number of iterations allowed was set to 10000 and the results were built during 25 epochs.

The system was implemented on a AMD K6-II-500MHz, with 256 MB RAM memory, using Linux and GNU gcc. The time to run each iteration is less than 0.5 seconds for examples 1 and 3. Increasing the number of pieces require an increasing iteration time in the learning algorithms.

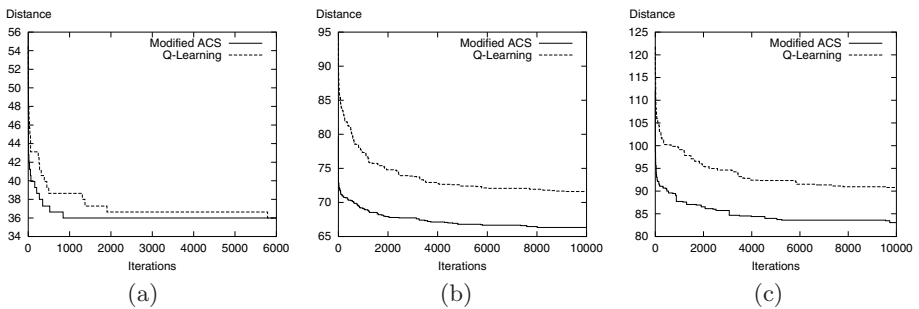


Fig. 2. Result of the Modified ACS for examples 1 to 3 (from left to right)

7 Conclusion

From the experiments carried out we conclude that the combination of Reinforcement Learning, Heuristic Search and explicit domain information about states and actions to minimize the search space used in the proposed algorithm presents better results than any of the techniques alone.

The results obtained show that the Ant-ViBRA was able to minimize the task execution time (or the total distance traveled by the manipulator) in several configurations. Besides that, the learning time was also reduced when compared to other RL techniques.

Future works include the implementation of this architecture in a Flexible Assembly Cell with a robotic manipulator, the extension of the system to control teams of mobile robots performing foraging tasks, and the exploration of new forms of composing the experience of each ant to update the pheromone table after each iteration.

Acknowledgements

This research was conducted under the NSF/CNPq-ProTeM CC Project MAP-PEL (grant no. 68003399-8) and FAPESP Project AACROM (grant no. 2001/14588-2).

References

- [1] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, New York, 1999. 195, 196
- [2] E. Bonabeau, M. Dorigo, and G. Theraulaz. Inspiration for optimization from social insect behaviour. *Nature* 406 [6791], 2000. 195, 196
- [3] M. Dorigo. Ant algorithms and swarm intelligence. *Proceedings of the Seventeen International Joint Conference on Artificial Intelligence, Tutorial MP-1*, 2001. 195, 196
- [4] M. Dorigo and L. M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1), 1997. 196, 197
- [5] C.R. Kube and H. Zhang. Collective robotics: from social insects to robots. *Adaptive Behavior*, 2:189–218, 1994. 197
- [6] C.R. Kube and H. Zhang. Task modelling in collective robotics. *Autonomous Robots*, 4:53–72, 1997. 197
- [7] V. Maniezzo, M. Dorigo, and A. Coloni. Algodesk: an experimental comparison of eight evolutionary heuristics applied to the qap problem. *European Journal of Operational Research*, 81:188–204, 1995. 197
- [8] A. H. Reali-Costa, L. N. Barros, and R. A. C. Bianchi. Integrating purposive vision with deliberative and reactive planning: An engineering support on robotics applications. *Journal of the Brazilian Computer Society*, 4(3):52–60, April 1998. 199, 200
- [9] A. H. Reali-Costa and R. A. C. Bianchi. L-vibra: Learning in the vibra architecture. *Lecture Notes in Artificial Intelligence*, 1952:280–289, 2000. 199, 200

- [10] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz. Ant-based load balancing in telecommunications networks. *Adapt. Behav.*, 5:169–207, 1997. 197
- [11] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD Thesis, University of Cambridge, 1989. 196, 197

Automatic Text Summarization Using a Machine Learning Approach

Joel Larocca Neto, Alex A. Freitas, and Celso A. A. Kaestner

Pontifical Catholic University of Parana (PUCPR)
Rua Imaculada Conceicao, 1155
Curitiba - PR. 80.215-901. BRAZIL
{joel,alex,kaestner}@ppgia.pucpr.br
<http://www.ppgia.pucpr.br/~alex>

Abstract. In this paper we address the automatic summarization task. Recent research works on extractive-summary generation employ some heuristics, but few works indicate how to select the relevant features. We will present a summarization procedure based on the application of trainable Machine Learning algorithms which employs a set of features extracted directly from the original text. These features are of two kinds: statistical – based on the frequency of some elements in the text; and linguistic – extracted from a simplified argumentative structure of the text. We also present some computational results obtained with the application of our summarizer to some well known text databases, and we compare these results to some baseline summarization procedures.

1 Introduction

Automatic text processing is a research field that is currently extremely active. One important task in this field is *automatic summarization*, which consists of reducing the size of a text while preserving its information content [9], [21]. A *summarizer* is a system that produces a condensed representation of its input's for user consumption [12].

Summary construction is, in general, a complex task which ideally would involve deep natural language processing capacities [15]. In order to simplify the problem, current research is focused on extractive-summary generation [21]. An extractive summary is simply a subset of the sentences of the original text. These summaries do not guarantee a good narrative coherence, but they can conveniently represent an approximate content of the text for relevance judgement.

A summary can be employed in an *indicative* way – as a pointer to some parts of the original document, or in an *informative* way – to cover all relevant information of the text [12]. In both cases the most important advantage of using a summary is its reduced reading time. Summary generation by an automatic procedure has also other advantages: (i) the size of the summary can be controlled; (ii) its content is

determinist; and (iii) the link between a text element in the summary and its position in the original text can be easily established.

In our work we deal with an automatic trainable summarization procedure based on the application of machine learning techniques. Projects involving extractive summary generation have shown that the success of this task depends strongly on the use of heuristics [5], [7]; unfortunately few indicatives are given of how to choose the relevant features for this task. We will employ here statistical and linguistic features, extracted directly and automatically from the original text.

The rest of the paper is organized as follows: section 2 presents a brief review of the text summarization task; in section 3 we describe in detail our proposal, discussing the employed set of features and the general framework of trainable summarizer; in section 4 we relate the computational results obtained with the application of our proposal to a reference document collection; and finally, in section 5 we present some conclusions and outline some envisaged research work.

2 A Review of Text Summarization

An automatic summarization process can be divided into three steps [21]: (1) in the *preprocessing step* a structured representation of the original text is obtained; (2) in the *processing step* an algorithm must transform the text structure into a summary structure; and (3) in the *generation step* the final summary is obtained from the summary structure.

The methods of summarization can be classified, in terms of the level in the linguistic space, in two broad groups [12]: (a) shallow approaches, which are restricted to the syntactic level of representation and try to extract salient parts of the text in a convenient way; and (b) deeper approaches, which assume a semantics level of representation of the original text and involve linguistic processing at some level.

In the first approach the aim of the preprocessing step is to reduce the dimensionality of the representation space, and it normally includes: (i) *stop-word* elimination – common words with no semantics and which do not aggregate relevant information to the task (e.g., “the”, “a”) are eliminated; (ii) *case folding*: consists of converting all the characters to the same kind of letter case - either upper case or lower case; (iii) *stemming*: syntactically-similar words, such as plurals, verbal variations, etc. are considered similar; the purpose of this procedure is to obtain the stem or radix of each word, which emphasize its semantics.

A frequently employed text model is the vectorial model [20]. After the preprocessing step each text element – a sentence in the case of text summarization – is considered as a N -dimensional vector. So it is possible to use some metric in this space to measure similarity between text elements. The most employed metric is the cosine measure, defined as $\cos \theta = (\langle x, y \rangle) / (|x| \cdot |y|)$ for vectors x and y , where $(\langle, \rangle)$ indicates the scalar product, and $|x|$ indicates the module of x . Therefore maximum similarity corresponds to $\cos \theta = 1$, whereas $\cos \theta = 0$ indicates total discrepancy between the text elements.

The evaluation of the quality of a generated summary is a key point in summarization research. A detailed evaluation of summarizers was made at the

TIPSTER Text Summarization Evaluation Conference (SUMMAC) [10], as part of an effort to standardize summarization test procedures. In this case a reference summary collection was provided by human judges, allowing a direct comparison of the performance of the systems that participated in the conference. The human effort to elaborate such summaries, however, is huge. Another reported problem is that even in the case of human judges, there is low concordance: only 46 % according to Mitra [15]; and more importantly: the summaries produced by the same human judge in different dates have an agreement of only 55 % [19].

The idea of a “reference summary” is important, because if we consider its existence we can objectively evaluate the performance of automatic summary generation procedures using the classical Information Retrieval (IR) *precision* and *recall* measures. In this case a sentence will be called *correct* if it belongs to the reference summary. As usual, *precision* is the ratio of the number of selected correct sentences over the total number of selected sentences, and *recall* is the ratio of the number of selected correct sentences over the total number of correct sentences. In the case of fixed-length summaries the two measures are identical, since the sizes of the reference and the automatically obtained extractive summaries are identical.

Mani and Bloedorn [11] proposed an automatic procedure to generate reference summaries: if each original text contains an author-provided summary, the corresponding size- K reference extractive summary consists of the K most similar sentences to the author-provided summary, according to the cosine measure. Using this approach it is easy to obtain reference summaries, even for big document collections.

A Machine Learning (ML) approach can be envisaged if we have a collection of documents and their corresponding reference extractive summaries. A trainable summarizer can be obtained by the application of a classical (trainable) machine learning algorithm in the collection of documents and its summaries. In this case the sentences of each document are modeled as vectors of features extracted from the text. The summarization task can be seen as a two-class classification problem, where a sentence is labeled as “correct” if it belongs to the extractive reference summary, or as “incorrect” otherwise. The trainable summarizer is expected to “learn” the patterns which lead to the summaries, by identifying relevant feature values which are most correlated with the classes “correct” or “incorrect”. When a new document is given to the system, the “learned” patterns are used to classify each sentence of that document into either a “correct” or “incorrect” sentence, producing an extractive summary. A crucial issue in this framework is how to obtain the relevant set of features; the next section treats this point in more detail.

3 A Trainable Summarizer Using a ML Approach

We concentrate our presentation in two main points: (1) the set of employed features; and (2) the framework defined for the trainable summarizer, including the employed classifiers.

A large variety of features can be found in the text-summarization literature. In our proposal we employ the following set of features:

(a) Mean-TF-ISF. Since the seminal work of Luhn [9], text processing tasks frequently use features based on IR measures [5], [7], [23]. In the context of IR, some very important measures are term frequency (TF) and term frequency $\times$ inverse document frequency (TF-IDF) [20]. In text summarization we can employ the same idea: in this case we have a single document d , and we have to select a set of relevant sentences to be included in the extractive summary out of all sentences in d . Hence, the notion of a collection of documents in IR can be replaced by the notion of a single document in text summarization. Analogously the notion of document – an element of a collection of documents – in IR, corresponds to the notion of sentence – an element of a document – in summarization. This new measure will be called term frequency $\times$ inverse sentence frequency, and denoted $\text{TF-ISF}(w,s)$ [8]. The final used feature is calculated as the mean value of the TF-ISF measure for all the words of each sentence.

(b) Sentence Length. This feature is employed to penalize sentences that are too short, since these sentences are not expected to belong to the summary [7]. We use the normalized length of the sentence, which is the ratio of the number of words occurring in the sentence over the number of words occurring in the longest sentence of the document.

(c) Sentence Position. This feature can involve several items, such as the position of a sentence in the document as a whole, its the position in a section, in a paragraph, etc., and has presented good results in several research projects [5], [7], [8], [11], [23]. We use here the percentile of the sentence position in the document, as proposed by Nevill-Manning [16]; the final value is normalized to take on values between 0 and 1.

(d) Similarity to Title. According to the vectorial model, this feature is obtained by using the title of the document as a “query” against all the sentences of the document; then the similarity of the document’s title and each sentence is computed by the cosine similarity measure [20].

(e) Similarity to Keywords. This feature is obtained analogously to the previous one, considering the similarity between the set of keywords of the document and each sentence which compose the document, according to the cosine similarity.

For the next two features we employ the concept of text cohesion. Its basic principle is that sentences with higher degree of cohesion are more relevant and should be selected to be included in the summary [1], [4], [11], [15].

(f) Sentence-to-Sentence Cohesion. This feature is obtained as follows: for each sentence s we first compute the similarity between s and each other sentence s' of the document; then we add up those similarity values, obtaining the raw value of this feature for s ; the process is repeated for all sentences. The normalized value (in the range [0, 1]) of this feature for a sentence s is obtained by computing the ratio of the raw feature value for s over the largest raw feature value among all sentences in the document. Values closer to 1.0 indicate sentences with larger cohesion.

(g) Sentence-to-Centroid Cohesion. This feature is obtained for a sentence s as follows: first, we compute the vector representing the centroid of the document, which is the arithmetic average over the corresponding coordinate values of all the sentences of the document; then we compute the similarity between the centroid and each sentence, obtaining the raw value of this feature for each sentence. The normalized value in the range [0, 1] for s is obtained by computing the ratio of the raw feature value over the largest raw feature value among all sentences in the

document. Sentences with feature values closer to 1.0 have a larger degree of cohesion with respect to the centroid of the document, and so are supposed to better represent the basic ideas of the document.

For the next features an approximate argumentative structure of the text is employed. It is a consensus that the generation and analysis of the complete rethorical structure of a text would be impossible at the current state of the art in text processing. In spite of this, some methods based on a surface structure of the text have been used to obtain good-quality summaries [23], [24]. To obtain this approximate structure we first apply to the text an agglomerative clustering algorithm. The basic idea of this procedure is that similar sentences must be grouped together, in a bottom-up fashion, based on their lexical similarity. As result a hierarchical tree is produced, whose root represents the entire document. This tree is binary, since at each step two clusters are grouped. Five features are extracted from this tree, as follows:

(h) Depth in the tree. This feature for a sentence s is the depth of s in the tree.

(i) Referring position in a given level of the tree (positions 1, 2, 3, and 4). We first identify the path from the root of the tree to the node containing s , for the first four depth levels. For each depth level, a feature is assigned, according to the direction to be taken in order to follow the path from the root to s ; since the argumentative tree is binary, the possible values for each position are: left, right and none, the latter indicates that s is in a tree node having a depth lower than four.

(j) Indicator of main concepts. This is a binary feature, indicating whether or not a sentence captures the main concepts of the document. These main concepts are obtained by assuming that most of relevant words are nouns. Hence, for each sentence, we identify its nouns using a part-of-speech software [3]. For each noun we then compute the number of sentences in which it occurs. The fifteen nouns with largest occurrence are selected as being the main concepts of the text. Finally, for each sentence the value of this feature is considered “true” if the sentence contains at least one of those nouns, and “false” otherwise.

(k) Occurrence of proper names. The motivation for this feature is that the occurrence of proper names, referring to people and places, are clues that a sentence is relevant for the summary. This is considered here as a binary feature, indicating whether a sentence s contains (value “true”) at least one proper name or not (value “false”). Proper names were detected by a part-of-speech tagger [3].

(l) Occurrence of anaphors. We consider that anaphors indicate the presence of non-essential information in a text: if a sentence contains an anaphor, its information content is covered by the related sentence. The detection of anaphors was performed in a way similar to the one proposed by Strzalkowski [22]: we determine whether or not certain words, which characterize an anaphor, occur in the first six words of a sentence. This is also a binary feature, taking on the value “true” if the sentence contains at least one anaphor, and “false” otherwise.

(m) Occurrence of non-essential information. We consider that some words are indicators of non-essential information. These words are speech markers such as “because”, “furthermore”, and “additionally”, and typically occur in the beginning of a sentence. This is also a binary feature, taking on the value “true” if the sentence contains at least one of these discourse markers, and “false” otherwise.

The ML-based trainable summarization framework consists of the following steps:

1. We apply some standard preprocessing information retrieval methods to each document, namely stop-word removal, case folding and stemming. We have employed the stemming algorithm proposed by Porter [17].
2. All the sentences are converted to its vectorial representation [20].
3. We compute the set of features described in the previous subsection. Continuous features are discretized: we adopt a simple “class-blind” method, which consists of separating the original values into equal-width intervals. We did some experiments with different discretization methods, but surprisingly the selected method, although simple, has produced better results in our experiments.
4. A ML trainable algorithm is employed; we employ two classical algorithms, namely C4.5 [18] and Naive Bayes [14]. As usual in the ML literature, we employ these algorithms trained on a training set and evaluated on a separate test set.

The framework assumes, of course, that each document in the collection has a reference extractive summary. The “correct” sentences belonging to the automatically produced extractive summary are labeled as “positive” in classification/data mining terminology, whereas the remaining sentences are labeled as “negative”. In our experiments the extractive summaries for each document were automatically obtained, by using an author-provided non-extractive summary, as explained in section 2.

4 Computational Results

As previously mentioned, we have used two very well-known ML classification algorithms, namely Naive Bayes [14] and C4.5 [18]. The former is a Bayesian classifier which assumes that the features are independent from each other. Despite this unrealistic assumption, the method presents good results in many cases, and it has been successfully used in many text mining projects. C4.5 is a decision-tree algorithm that is frequently employed for comparison purposes with other classification algorithms, particularly in the data mining and ML communities.

We did two series of experiments: in the first one, we employed automatically-produced extractive summaries; in the second one, manually-produced summaries were employed. In all the experiments we have used a document collection available in the TIPSTER document base [6]. This collection consists of texts published in several magazines about computers, hardware, software, etc., which have sizes varying from 2 Kbytes to 64 Kbytes. Due to our framework, we used only documents which have an author-provided summary, and a set of keywords. The whole TIPSTER document base contained 33,658 documents with these characteristics. A subset of these documents was randomly selected for the experiments to be reported in this section.

In the first experiment, using automatically-generated reference extractive summaries, we employed four text-summarization methods, as follows:

- (a) Our proposal (features as described in section 3) using C4.5 as the classifier;
- (b) Our proposal using Naive Bayes as the classifier.

- (c) First Sentences (used as a baseline summarizer): this method selects the first n sentences of the document, where n is determined by the desired compression rate, defined as the ratio of summary length to source length [12], [21]. Although very simple, this procedure provides a relatively strong baseline for the performance of any text-summarization method [2].
- (d) Word Summarizer (WS): Microsoft's WS is a text summarizer which is part of Microsoft Word, and it has been used for comparison with other summarization methods by several authors [1], [13]. This method uses non-documented techniques to perform an "almost extractive" summary from a text, with the summary size specified by the user.

The WS has some characteristics that are different from the previous methods: the specified summary size refers to the number of characters to be extracted, and some sentences can be modified by WS. In our experiments due to these characteristics a direct comparison between WS and the other methods is not completely fair: (i) the summaries generated by WS can contain a few more or a few less sentences than the summaries produced by the other methods; (ii) in some cases it will not be possible to compute an exact match between a sentence selected by WS and an original sentence; in these cases we ignore the corresponding sentences.

It is important to note that only our proposal is based on a ML trainable summarizer; the two remaining methods are not trainable, and were used mainly as baseline for result comparison.

The document collection used in this experiment consisted of 200 documents, partitioned into disjoint training and test sets with 100 documents each. The training set contained 25 documents of 11 Kbytes, 25 documents of 12 Kbytes, 25 documents of 16 Kbytes, and 25 documents of 31 Kbytes. The average number of sentences per document is 129.5, since there are in total 12,950 sentences in the training set. The test set contained 25 documents of 10 Kbytes, 25 documents of 13 Kbytes, 25 documents of 15 Kbytes, and 25 documents of 28 Kbytes. The average number of sentences per document is 118.6, since there are in total 11,860 sentences in the test set.

Table 1 reports the results obtained by the four summarizers. We consider compression rates of 10 % and 20 %. The performance is expressed in terms of precision / recall values, expressed in percentage (%), and the corresponding standard deviations are indicated after the " $\pm$ " symbol. The best obtained results are shown in boldface.

Table 1. Results for training and test sets composed by automatically-produced summaries

Summarizer	Compression rate: 10%		Compression rate: 20%	
	Precision	Recall	Precision	Recall
Trainable-C4.5	22.36 $\pm$ 1.48		34.68 $\pm$ 1.01	
Trainable-Bayes	40.47 $\pm$ 1.99		51.43 $\pm$ 1.47	
First-Sentences	23.95 $\pm$ 1.60		32.03 $\pm$ 1.36	
Word-Summarizer	26.13 $\pm$ 1.21	34.44 $\pm$ 1.56	38.80 $\pm$ 1.14	43.67 $\pm$ 1.30

Table 2. Results for training set composed by automatically-produced summaries and test set composed by manually-produced summaries

Summarizer	Compression rate: 10%		Compression rate: 20%	
	Precision	Recall	Precision	Recall
Trainable-C4.5	24.38 ± 2.84		31.73 ± 2.41	
Trainable-Bayes	26.14 ± 3.32		37.50 ± 2.29	
First-Sentences	18.78 ± 2.54		28.01 ± 2.08	
Word-Summarizer	14.23 ± 2.17	17.24 ± 2.56	24.79 ± 2.22	27.56 ± 2.41

We can draw the following conclusions from this experiment: (1) the values of precision and recall for all the methods are significantly higher with the rate of 20% than with the compression rate of 10%; this is a expected result, since the larger the compression rate, the larger the number of sentences to be selected for the summary, and then the larger the probability that a sentence selected by a summarizer matches with a sentence belonging to the extractive summary; (2) the best results were obtained by our trainable summarizer with Naive Bayes classifier for both compression rates; using the same features, but with the C4.5 as classifier, the obtained results were poor: the results are similar to the First-Sentences and Word Summarizer baselines.

The latter result offers us an interesting lesson: most research projects on trainable summarizers focus on the proposal of new features for classification, trying to produce more and more elaborate statistics-based or linguistics-based features, but they usually employ a single classifier in the experiments. Normally “conventional” classifiers are used. Our results indicate that researchers should concentrate their attention in the study of more elaborate classifiers, tailored for the text-summarization task, or at least evaluate and select the best classifier among the conventional ones already available.

In the second experiment we employ in the test step summaries manually produced by a human judge. We emphasize that in the training phase of our proposal we have used the same database of automatically-generated summaries employed in the previous experiment.

The test database was composed of 30 documents, selected at random from the original document base. The manual reference summaries were produced by a human judge – a professional English teacher with many years of experience –specially hired for this task. For the compression rates of 10 % and 20% the same four summarizers of the first experiment were compared. The obtained results are presented in Table 2.

Here again the best results were obtained by our proposal using the Naive Bayes algorithm as classifier. Similar to the previous experiment, results for 20% of compression were superior to the results produced with 10% of compression.

In order to verify the consistency between the two experiments we have compared the manually-produced summaries and the automatically-produced ones. We considered here the manually-produced summaries as a reference, and we calculated the precision and recall for the automatically produced summaries of the same

documents. Obtained results are presented in Table 3. These results are consistent with the ones presented by Mitra [15], and indicate that the degree of dissimilarity between a manually-produced summary and an automatically-produced summary in our experiments is comparable to the dissimilarity between two summaries produced by different human judges.

Table 3. Comparison between automatically-produced and manually-produced summaries

	Precision / Recall
Compression rate: 10%	30.79 ± 3.96
Compression rate: 20%	42.98 ± 2.42

5 Conclusions and Future Research

In this work we have explored the framework of using a ML approach to produce trainable text summarizers, in a way which was proposed a few years ago by Kupiec [7]. We have chosen this research direction because it allows us to measure the results of a text summarization algorithm in an objective way, similar to the standard evaluation of classification algorithms found in the ML literature. This avoids the problem of subjective evaluation of the quality of a summary, which is a central issue in the text summarization research.

We have performed an extensive investigation of that framework. In our proposal we employ a trainable summarizer that uses a large variety of features, some of them employing statistics-oriented procedures and others using linguistics-oriented ones. For the classification task we have used two different well known classification algorithms, namely the Naive Bayes algorithm and the C4.5 decision tree algorithm. Hence, it was possible to analyze the performance of two different text-summarization procedures. The performance of these procedures was compared with the performance of two non-trainable, baseline methods.

We did basically two kind of experiments: in the first one we considered automatically-produced summaries for both the training and test phases; in the second experiment we used automatically-produced summaries for training and manually-produced summaries for testing. In general the trainable method using Naive Bayes classifier significantly outperformed all the baseline methods.

An interesting finding of our experiments was that the choice of the classifier (Naive Bayes versus C4.5) strongly influenced the performance of the trainable summarizer. We intend to focus mainly on the development of a new or extended classification algorithm tailored for text summarization in our future research work.

References

1. Barzilay, R. ; Elhadad, M. Using Lexical Chains for Text Summarization. In Mani, I.; Maybury, M. T. (eds.). In *Proceedings of the ACL/EACL-97 Workshop on Intelligent Scalable Text Summarization*, Association of Computational Linguistics (1997)

2. Brandow, R.; Mitze, K., Rau, L. Automatic condensation of electronic publications by sentence selection. *Information Processing and Management* 31(5) (1994) 675-685
3. Brill, E. A simple rule-based part-of-speech tagger. In *Proceedings of the Third Conference on Applied Comp. Linguistics*. Assoc. for Computational Linguistics (1992)
4. Carbonell, J. G.; Goldstein, J. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR-98* (1998)
5. Edmundson, H. P. New methods in automatic extracting. *Journal of the Association for Computing Machinery* 16 (2) (1969) 264-285
6. Harman, D. Data Preparation. In Merchant, R. (ed.). *The Proceedings of the TIPSTER Text Program Phase I*. Morgan Kaufmann Publishing Co. (1994)
7. Kupiec, J. ; Pedersen, J. O.; Chen, F. A trainable document summarizer. In *Proceedings of the 18th ACM-SIGIR Conference, Association of Computing Machinery* (1995) 68-73
8. Larocca Neto, J.; Santos, A. D.; Kaestner, C.A.; Freitas, A.A.. Document clustering and text summarization. *Proc. of 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining (PADD-2000)* London: The Practical Application Company (2000) 41-55
9. Luhn, H. The automatic creation of literature abstracts. *IBM Journal of Research and Development* 2(92) (1958) 159-165
10. Mani, I.; House, D.; Klein, G.; Hirschman, L.; Obrsl, L.; Firmin, T.; Chrzanowski, M.; Sundheim, B. *The TIPSTER SUMMAC Text Summarization Evaluation*. MITRE Technical Report MTR 98W0000138. The MITRE Corporation (1998)
11. Mani, I.; Bloedorn, E. Machine Learning of Generic and User-Focused Summarization. In *Proceedings of the Fifteenth National Conference on AI (AAAI-98)* (1998) 821-826
12. Mani, I. *Automatic Summarization*. J.Benamins Publ. Co. Amsterdam Philadelphia (2001)
13. Marcu, D. Discourse trees are good indicators of importance in text. In Mani., I.; Maybury, M. (eds.). *Adv. in Automatic Text Summarization*. The MIT Press (1999) 123-136
14. Mitchell, T. *Machine Learning*. McGraw-Hill (1997)
15. Mitra, M.; Singhal, A.; Buckley, C. Automatic text summarization by paragraph extraction. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*. Madrid (1997)
16. Nevill-Manning, C. G. ; Witten, I. H. Paynter, G. W. et al. KEA: Practical Automatic Keyphrase Extraction. *ACM DL 1999* (1999) 254-255
17. Porter, M.F. An algorithm for suffix stripping. *Program* 14, 130-137. 1980. Reprinted in: Sparck-Jones, K.; Willet, P. (eds.) *Readings in Information Retrieval*. Morgan Kaufmann (1997) 313-316
18. Quinlan, J. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Sao Mateo California (1992)
19. Rath, G. J. ; Resnick A. ; Savvage R. The formation of abstracts by the selection of sentences. *American Documentation* 12 (2) (1961) 139-141

20. Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management* 24, 513-523. 1988. Reprinted in: Sparck-Jones, K.; Willet, P. (eds.) *Readings in I.Retrieval*. Morgan Kaufmann (1997) 323-328
21. Sparck-Jones, K. Automatic summarizing: factors and directions. In Mani, I.; Maybury, M. *Advances in Automatic Text Summarization*. The MIT Press (1999) 1-12
22. Strzalkowski, T.; Stein, G.; Wang, J.; Wise, B. A Robust Practical Text Summarizer. In Mani, I.; Maybury, M. (eds.), *Adv. in Autom. Text Summarization*. The MIT Press (1999)
23. Teufel, S.; Moens, M. Argumentative classification of extracted sentences as a first step towards flexible abstracting. In Mani, I.; Maybury M. (eds.). *Advances in automatic text summarization*. The MIT Press (1999)
24. Yaari, Y. Segmentation of Expository Texts by Hierarchical Agglomerative Clustering. *Technical Report*, Bar-Ilan University Israel (1997)

Towards a Theory Revision Approach for the Vertical Fragmentation of Object Oriented Databases

Flavia Cruz, Fernanda Baião, Marta Mattoso, and Gerson Zaverucha

Department of Computer Science - COPPE/UFRJ
PO Box: 68511, Rio de Janeiro, RJ, Brazil, 21945-970
Telephone: +55+21+590-2552
Fax: +55+21+290-6626
{fcruz,baiao,marta,gerson}@cos.ufrj.br

Abstract. The performance of applications on Object Oriented Database Management Systems (OODBs) is strongly affected by Distribution Design, which reduces irrelevant data accessed by applications and data exchange among sites. In an OO environment, the Distributed Design is a complex task, and an open research problem. In this work, we present a knowledge-based approach for the vertical fragmentation phase of the distributed design of object-oriented databases. In this approach, we show a Prolog implementation of a vertical fragmentation algorithm, and describe how it can be used as background knowledge for a knowledge discovery/revision process through Inductive Logic Programming (ILP). The objective of the work is to extend our framework proposed to handle the class fragmentation problem, showing the viability of automatically improving the vertical fragmentation algorithm to produce more efficient fragmentation schemas, using a theory revision system. We do not intend to propose the best vertical fragmentation algorithm. We concentrate here on the process of revising a vertical fragmentation algorithm through knowledge discovery techniques, rather than only obtaining a final optimal algorithm.

1 Introduction

Distributed and parallel processing may improve performance for applications that manipulate large volumes of data. This is addressed by removing irrelevant data accessed by queries and transactions and by reducing the data exchange among sites [9], which are the two main goals of the Distributed Design of Databases. The fragmentation phase of the Distributed Design is the process of clustering in fragments the information accessed simultaneously by applications, and is known to be an NP-hard problem[12]. Therefore, heuristic-based algorithms have been proposed in the literature to handle the problem in an efficient manner.

This work addresses the vertical fragmentation of classes, by providing an alternative way of automatically modifying an existing algorithm for the problem. This approach uses a rule-based implementation of the algorithm from Navathe and Ra[3] as background knowledge when trying to discover a new revised algorithm through the use of a machine learning technique: Inductive Logic Programming (ILP) [22,13]. The revised algorithm may reflect important issues to the class fragmentation problem that may be implicit, that is, not yet discovered by any of the proposed distributed design algorithms in the literature. In the present knowledge-based approach, we represent the vertical fragmentation algorithm (VFA) as a set of rules (Prolog clauses) and perform a fine-tuning of it, thus discovering a new set of rules that will represent the revised algorithm. This new set of rules will represent a revised vertical fragmentation algorithm that will propose optimal (or near to optimal) vertical class fragments. In other words, we intend to perform Data Mining considering available database schema and data access information as a test bed to produce optimal vertical class fragments as an output.

The organization of this work is as follows: the next Section presents some definitions from the literature regarding the Distributed Design task, identifies some difficulties which motivated the use of a knowledge based approach for this problem and shows the state of the art in the Distributed Design research area and in the ILP field. The vertical fragmentation algorithm is described in Section 3. Section 4 discusses the use of ILP to revise the VFA and describe its Prolog implementation. Finally, Section 5 concludes this paper.

2 Background and Related Work

Distributed Design (DD) involves making decisions on the fragmentation and placement of data across the sites of a computer network [12]. In a top down approach, the distributed design has two phases: fragmentation and allocation. The fragmentation phase is the process of clustering in fragments the information accessed simultaneously by applications, and the allocation phase is the process of distributing the generated fragments over the database system sites. To fragment a class, it is possible to use two basic techniques: horizontal (primary or derived) fragmentation [12] and vertical fragmentation. In an object oriented (OO) environment, vertical fragmentation breaks the class logical structure (its attributes and methods) and distributes them across the fragments, which will logically contain the same objects, but with different structures. The vertical fragmentation favors the class extension access and the use of class attributes and methods, by removing irrelevant data accessed by operations.

The DDOODB is known to be an NP-hard problem [12]. In the object model, additional issues contribute to increasing the difficulty of the task and turn it into an even more complex problem. Therefore, heuristic-based algorithms have been proposed in the literature to handle the problem in an efficient manner [1, 3, 4, 5, 6, 8, 10]. Additionally, some researchers have been working on applying machine learning techniques [11] to solve database research problems. For example, Blockeel and De Raedt [14, 19] presented an approach for the inductive design of deductive databases, based on the database instances to define some intentional predicates. Getoor *et al.*

[17, 18] use relational bayesian networks to estimate query selectivity in a query processor and to predict the structure of relational databases, respectively. Previous work of our group [1,2,24,25,26] presented a framework to handle the class fragmentation problem, including a theory revision module that automatically improves the choice between horizontal and/or vertical fragmentation techniques. In this work, we extend these ideas and propose a theory revision approach to automatically improve a VFA.

3 The Vertical Fragmentation Algorithm

This Section presents an overview of the whole fragmentation process of OODBs illustrated in Figure 1, and describes the algorithm for the Vertical Fragmentation Phase of the Class Fragmentation Process that was proposed in [1,24].

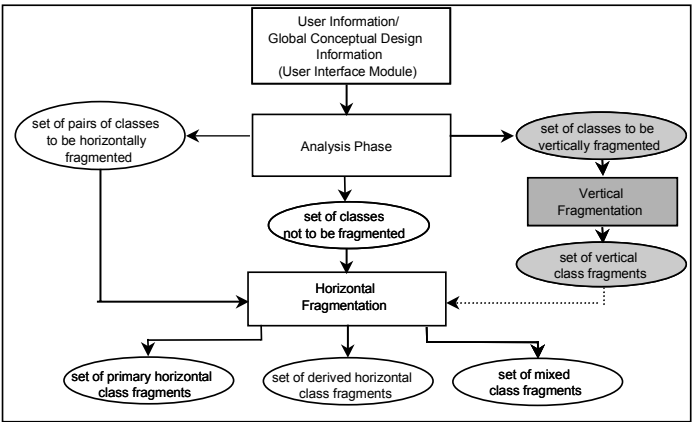


Fig. 1. A framework for class fragmentation in Distributed Design of OODBs

The Analysis Phase algorithm considers issues about the database structure and user operations to decide on the fragmentation strategy (horizontal and/or vertical) for each class in the schema. Information considered in this phase includes class and operation characteristics can be found in [1,24]. Additionally, for the purpose of this work it is necessary to know which class elements are accessed by each operation.

The algorithm used for the vertical fragmentation, which is the focus of this work, is an extension of the Graphical Algorithm proposed in [3] and [4] to handle OO issues such as methods such as methods, and is executed for each class assigned to be vertically fragmented by the analysis phase (lines 1-4 of Fig. 2).

Building the element affinity matrix (lines 5-9 of Fig. 2). It builds the affinity matrix between the elements of the class. The elements of the class represent the matrix dimensions. Each value $M(e_i, e_j)$ in the element affinity matrix represents the sum of the frequencies of operations that accesses elements e_i and e_j simultaneously.

Building the element affinity graph (lines 10-22 of Fig. 2). It implements a graph-based approach to group elements in cycles, and map them to the vertical fragments of

the class. The algorithm forms cycles in which each node will have high affinity to the nodes within its cycle, but low affinity to the nodes of other cycles.

The overall idea of the affinity graph construction is as follows. Each graph node represents an element of the class, and graph links between the nodes are inserted one at a time by selecting (from the element affinity matrix) the highest value $M(e_i, e_j)$ that was not previously selected (lines 10–12 of Fig. 2). Let e_i be one of the graph extremities (one edge incident to it), and e_j the new node to be inserted in the graph by selecting edge (e_i, e_j) . If the inclusion of edge (e_i, e_j) forms a cycle in the graph (line 13 of Fig. 2), then we test if this cycle can be an affinity cycle[20]. Affinity cycles are then considered as fragment candidates (line 16 of Fig. 2). On the other hand, if the inclusion of edge (e_i, e_j) does not form a cycle in the graph (line 17 of Fig. 2) and there is a fragment candidate already (line 18 of Fig. 2), then we test if the affinity cycle representing this candidate fragment can be extended by edge (e_i, e_j) [20]. If the affinity cycle cannot be extended (line 20 of Fig. 2), then the candidate fragment is considered a graph fragment (lines 21 and 22 of Fig. 2).

After building the element affinity graph, each vertical fragment of the class is defined by a projection on the elements of the correspondent graph fragment. An additional fragment must be further defined for the elements that were not used by any operation (line 23 of Fig. 2). This additional fragment is required because it reduces the number of class fragments (by grouping less frequently used elements in a unique fragment rather than defining distinct fragments for each of these elements), and eliminates the overhead of managing more vertical fragments of a class for less used data.

4 Theory Revision in the Vertical Fragmentation of OODBs

The heuristic-based vertical fragmentation algorithm presented in section 3 produced good performance results as shown in [1]. However, it would be very interesting to continue to improve these results, by discovering new heuristics for the vertical fragmentation problem, and incorporate them on the algorithm. Nevertheless, this would require a detailed analysis of each new experimental performance result from the literature, and manual modifications on the algorithm. Additionally, the formalization of new heuristics from the experiments, while maintaining previous heuristics consistent, proved to be an increasingly difficult task.

Therefore, this section proposes a knowledge-based approach for improving the VFA with theory revision [13,23,27]. We extend the ideas proposed in [1], where the authors show the effectiveness of this knowledge-based approach in improving a previous version of an existing analysis algorithm with an experiment with the 007 benchmark [7]. In that work, the theory revision process automatically modified the previous version of the analysis algorithm in order to produce a new version of it, which obtained a fragmentation schema with better performance.

```

function VerticalFragmentation ( Cv: set of classes to be vertically fragmented, Oproj: the set of projection operations )
returns Fv : set of vertical class fragments
begin
  for each Ck that is in Cv do (1)
    M = BuildElementAffinityMatrix (Ck, Oproj) (2)
    fragmentsOfCk = BuildAndPartitionElementAffinityGraph (Ck, M) (3)
    Fv += fragmentsOfCk (4)
  end for
  return Fv
end

function BuildElementAffinityMatrix ( Ck: class to be vertically fragmented, Oproj: the set of projection operations and
their execution frequencies )
returns M : element affinity matrix of Ck
begin
  for each Oi that is in Oproj do (5)
    for each element ei of Ck that is accessed by Oi do (6)
      for each element ej of Ck that is accessed by Oi do (7)
        freqOi = execution frequency of Oi
        if M(ei, ej) is not null then M(ei, ej) += freqOi (8)
        else create M(ei, ej), set M(ei, ej) = freqOi (9)
      end if
    end for
  end for
  return M
end

function BuildAndPartitionElementAffinityGraph ( Ck: class to be vertically fragmented, M : element affinity matrix of Ck )
returns fragmentsOfCk : set of vertical fragments of Ck
begin
  N = empty set of nodes; A = empty set of links ; G = (N, A)
  N += any element of Ck
  while there is an element of Ck that is not in N do (10)
    M(ei, ej) = highest element from M such that ei is a graph extremity and ej is the new node to be inserted (11)
    a := link between ei and ej
    N += ei; N += ej (12)
    if M(ei, ej) forms a cycle in the graph G then (13)
      let cp be this cycle (14)
      if cp can be an affinity cycle then (15)
        mark cp as a fragment candidate (16)
      end if
    else (17)
      if there is a fragment candidate then (18)
        let cf be this candidate (19)
        if cf cannot be extended then (20)
          mark cf as a fragment (21)
          A += a
          fragmentsOfCk += cf (22)
        end if
      end if
    end while
    cf += elements in Ck and not in any fragment in fragmentsOfCk (23)
  return fragmentsOfCk
end

```

Fig. 2: Algorithm for the vertical fragmentation of a class

The final goal of our work is then to automatically incorporate in the VFA the changes required to obtain better fragmentation schemas that may be found through additional experiments, and therefore automatically reflect the new heuristics implicit on these new results. This revision process will then represent a “fine-tuning” of our initial set of rules, thus discovering a new set of rules that will represent the revised algorithm.

Some researchers have been working on applying machine learning techniques to solve database research problems (see Sect. 2). However, considering the vertical class fragmentation as an application for theory revision is a novel approach in the area.

The idea of using knowledge-based neural networks (NN) to revise our background knowledge was first considered. There are, in the literature, many approaches for using NN in a theory revision process using propositional rules, such as KBANN [15]

and CIL2P [16]. However, due to the existence of function symbols in our analysis algorithm (such as lists) that could not be expressed through propositional rules, we needed a more expressive language, such as first-order Horn clauses. Since first-order theory refinement in NN is still an open research problem[28], we decided to work with another machine learning technique - Inductive Logic Programming (ILP) [13,22].

According to Mitchel [11], the process of ILP can be viewed as automatically inferring Prolog programs from examples and, possibly, from background knowledge. In [11], it has been pointed out that machine learning algorithms that use background knowledge, thus combining inductive with analytical mechanisms, obtain the benefits of both approaches: better generalization accuracy, smaller number of required training examples and explanation capability.

When the theory being revised contains variables, the rules are called first-order Horn clauses. Because sets of first order Horn clauses can be interpreted as programs in the logic programming language Prolog, the theory revision process for learning them may be called Inductive Logic Programming.

The theory revision is responsible for automatically changing the initial algorithm (which is called initial theory, or background knowledge) in such a way that it produces new results presented to the process. The result of the revision process is the revised algorithm. The theory revision task can be specified as the problem of finding a minimal modification of an initial theory that correctly classifies a set of training examples.

The resulting algorithm performance will depend not only on the quality of the background knowledge, but also on the quality of the examples considered in the training phase, as in conventional Machine Learning algorithms. Therefore, we need a set of validated vertical class fragmentation schema with good performance. However, such a set of optimal fragmentation schema is not easily found in the literature, since it is private information from companies. We then decided to work on some scenarios used as simple examples in the literature. We are also working on the generation of examples through the Branch & Bound(B&B) VF module under development.

This module represents an approach of exhaustive search for the best vertical class fragments for a given set of classes. The B&B module searches for an optimal solution in the space of potentially good vertical class fragments for a class and outputs its result to the distribution designer. Since the B&B algorithm searches over a large (yet not complete) hypotheses space, its execution cost is very high. To handle this, the B&B algorithm tries to bound its search for the best vertical class fragments by using a query processing cost function during the evaluation of each class in the hypotheses space. This cost function, defined in [29], is responsible for estimating the execution cost of queries on top of a class being evaluated. The B&B module then discards all the vertical class fragments with an estimate cost higher than the cost of the vertical class fragments output from the heuristic VFA[1] implemented in Prolog. Since the cost function is incremental, through the heuristic cost we can bound several alternatives at an early stage. Finally, the result from the B&B module, as well as the vertical class fragments discarded during the search, may generate examples (positive or negative, respectively) to the VF theory revision module.

4.1 A Prolog Implementation of the Vertical Fragmentation Algorithm

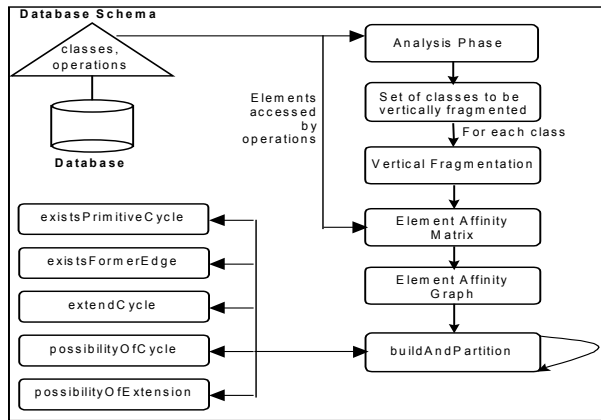


Fig. 3. The overall structure of our Prolog implementation for the vertical class fragmentation

```

findPartition(FirstNode,N,ListPartition):-
    retractall(partition(_)),
    retractall(cut(_,_,_)),
    retractall(candidateNewEdge(_,_,_)),
    retractall(removedProvisorilyEdge(_,_,_)),
    assert(cut(0,0,0)),
    createListNewEdge(ListCandidateNewEdge),
    createCandidateNewEdge(ListCandidateNewEdge),
    buildAndPartition([FirstNode],FirstNode,
        0,0,0,0,0,0,[],[],[],N,FirstNode),
    findall(P,partition(P),ListPartition_dup),
    list_to_set(ListPartition_dup, ListPartition),
    retractall(partition(_)).
  
```

Fig. 4. The starting point of VFA as a set of Prolog clausesProlog implementation

The set of heuristics implemented by the VFA may be further improved by executing a theory revision process using inductive logic programming (ILP) [22,27]. This process was initially proposed by [1] as Theory REvisionN on the Design of Distributed Databases (TREND3). In the present work, the improvement process may be carried out by providing two input parameters to the revision process: the VFA (representing the initial theory) and a vertical class fragmentation schema with previously known performance (representing a set of examples). The VFA will then be automatically modified by a theory revision system (called FORTE [23]) to produce a revised theory. The revised theory will represent an improved VFA that will be able to produce the schema given as input parameter, and this revised algorithm will then substitute the original one.

The overall structure of our set of rules is on Fig. 3. We have implemented our VFA as a set of Prolog clauses and used the example given in [3] as a test case. Some

of these clauses are illustrated in Fig. 4 and Fig. 5. This set of rules constitutes a very good starting point for the ILP process to obtain the revised algorithm. The predicate `buildAndPartition` is recursively called until all the nodes are considered to form the class fragments, and will be the target predicate for the revision process, that is, the one to be revised.

Our set of training examples is being derived from several works in the literature. We are extracting from each selected work two sets of facts, one representing the initial database schema and another representing the desired fragmentation schema. It is important to notice that, as we did not have as many available examples in the literature as it would be desired, the background knowledge will play a major role in the ILP learning process.

```

buildAndPartition(Tree, Begin, End, CycleNode,
NodeCompletingEdge, WeightCompletingEdge,
NodeFormerEdge, WeightFormerEdge, PrimitiveCycle,
CandidatePartition, Partition, N, Node) :-
    selectNewEdge(NewNode, BiggestEdge, NodeConnected,
        Begin, End),
    adjustLimits(NewNode, Begin, End, NewBegin, NewEnd,
        NodeConnected),
    refreshTree(Tree, NewNode, NewBegin, NewEnd, NewTree),
    notExistsPrimitiveCycle(NewNode, NodeConnected,
        CycleNode, Tree, Begin, End, NewPrimitiveCycle,
        NewNodeCompletingEdge, NewWeightCompletingEdge),
    CandidatePartition==[],
    removeCandidateNewEdge(NewNode, NodeConnected,
        BiggestEdge),
    T is N-1,
    buildAndPartition(NewTree, NewBegin, NewEnd, CycleNode,
        NodeCompletingEdge, WeightCompletingEdge,
        NodeFormerEdge, WeightFormerEdge, PrimitiveCycle,
        CandidatePartition, Partition, T, NewNode), !.

```

Fig. 5. One of the rules to build the graph and find the partition

5 Conclusion

In this paper, we have presented a knowledge-based approach to the vertical class fragmentation problem during the Distributed Design of Object Oriented Databases. Our developed VFA was implemented as a set of rules and is used as background knowledge when trying to discover a new revised algorithm through the use of Theory Revision [27]. In our approach, we will perform a fine-tuning of our initial algorithm (represented as a set of Prolog rules), thus discovering a new set of (Prolog) rules that will represent the revised algorithm. This new set of rules will represent a revised VFA that will propose optimal (or near to optimal) vertical class fragmentation schemas with improved performance.

We have presented the main ideas embedded in this novel approach for refining the VFA using a machine learning method - Inductive Logic Programming (ILP). This approach performs a knowledge discovery/revision process using our set of rules as

background knowledge. The objective of the work is to discover ($\square$ learn $\square$) new heuristics to be considered in the vertical fragmentation process.

Our main objective was to show the viability of performing a revision process in order to obtain a better VFA. We do not intend to obtain the best VFA ever possible. We concentrate here on the process of revising a DDOODB algorithm through Knowledge Discovery techniques, rather than on a final product.

Although we have addressed the problem of class fragmentation in the DDOODB context, an important future work is the use of the same inductive learning approach in other phases of the Distributed Design (such as the allocation phase), as well as in the Database Design itself, and use other data models (relational or deductive). Also, the resulting fragmentation schema obtained from our revised algorithm may be applied to fragment the database that will be used in [21], which proposes the use of distributed databases in order to scale-up data mining algorithms.

References

1. Bai $\square$, F.: A Methodology and Algorithms for the Design of Distributed Databases using Theory Revision. DSc Thesis, Technical Report ES-547/01, COPPE, Federal University of Rio de Janeiro, Rio de Janeiro, Brazil (2002)
2. Bai $\square$, F., Mattoso, M., Zaverucha, G.: A Knowledge-Based Perspective of the Distributed Design of Object Oriented Databases. Proc. Int. Conf. on Data Mining 1998. WIT Press, Rio de Janeiro, Brazil (1998) 383-400
3. Navathe, S., Ra, M.: Vertical Partitioning for Database Design: A Graphical Algorithm. Proc. of the 1989 ACM SIGMOD. Portland, Oregon (1989) 440-450
4. Navathe, S., Ceri, S., Wiederhold, G., Dou, J.: Vertical Partitioning Algorithms for Database Design. ACM Trans. Database Systems, Vol. 9(4) (1984) 680-710
5. Ezeife, C., Barker, K.: Distributed Object Based Design: Vertical Fragmentation of Classes. Int. J. of Distribute and Parallel Databases, Vol. 6(4) (1998) 317-350
6. Bellatreche, L., Simonet, A., Simonet, M.: Vertical Fragmentation in Distributed Object Database Systems with Complex Attributes and Methods. 7th International Workshop on Database and Expert Systems Applications, Zurich, Switzerland (1996) 15-21
7. Carey, M., De Witt, D., Naughton, J.: The 007 Benchmark.. In: Proc. of 1993 ACM SIGMOD, Washington DC (1993) 12-21
8. Chen, Y., Su, S.: Implementation and Evaluation of Parallel Query Processing Algorithms and Data Partitioning Heuristics in Object Oriented Databases. Distributed and Parallel Databases, Vol. 4(2) (1996) 107-142
9. Karlapalem, K., Navathe, S., Morsi, M.: Issues in Distribution Design of Object-Oriented Databases. In: $\square$ zsu, M. et. al (eds): Distributed Object Management, Morgan Kaufman Publishers (1994)
10. Malinowski, E.: Fragmentation Techniques for Distributed Object-Oriented Databases. MSc. Thesis, University of Florida (1996)
11. Mitchell, T.: Machine Learning. McGraw-Hill Companies Inc. (1997)

12. □zsu, M., Valduriez, P.: Principles of Distributed Database Systems. 2nd edn. Prentice-Hall, New Jersey (1999)
13. Lavrac, N., Džreroski, S.: Inductive Logic Programming: Techniques and Applications, Ellis Horwood (1994)
14. Blockeel, H., de Raedt, L.: Inductive Database Design. In: Proceedings of the International Symposium on Methodologies for Intelligent Systems (ISMIS'96). Lecture Notes in Artificial Intelligence, Vol. 1079. Springer-Verlag (1996) 376-385
15. Towell, G., Shavlik, J.: Knowledge-Based Artificial Neural Networks. Artificial Intelligence, 70 (1-2) (1994) 119-165
16. Garcez, A. S., Zaverucha, G.: The Connectionist Inductive Learning and Logic Programming System. Applied Intelligence Journal, Vol. 11(1) (1999) 59-77
17. Getoor, L., Taskar, B., Koller, D.: Selectivity Estimation using Probabilistic Models. In: Proc. of the 2001 ACM SIGMOD. Santa Barbara, California, USA (2001) 461-472
18. Getoor, L., Friedman, N., Koller, D., Taskar, B.: Probabilistic Models of Relational Structure. In: Proc. of the Int. Conf. on Machine Learning, Williamstown, MA (2001)
19. Blockeel, H., De Raedt, L.: IsIdd: an Interactive System for Inductive Database Design. Applied Artificial Intelligence 12(5) (1998) 385-420
20. Navathe, S., Karlapalem, K., Ra, M.: A Mixed Fragmentation Methodology for Initial Distributed Database Design. J. of Computer and Software Engineering, Vol. 3(4) (1995)
21. Provost, F., Hennessy, D.: Scaling-Up: Distributed Machine Learning with Cooperation. In: Proceedings of AAAI . AAAI Press, Portland, Oregon (1996) 74-79
22. Muggleton, S., De Raedt, L.: Inductive logic programming: Theory and methods. Journal of Logic Programming, Vol. 19(20) (1994) 629-679
23. Richards, B., Mooney, R.: Refinement of First-Order Horn-Clause Domain Theories. Machine Learning, Vol. 19(2) (1995) 95-131
24. Bai□, F., Mattoso, M., Zaverucha, G.: A Distribution Design Methodology for Object DBMS. Submitted in Aug 2000; revised manuscript sent in Nov 2001 to International Journal of Distributed and Parallel Databases. Kluwer Academic Publishers (2001)
25. Bai□, F., Mattoso, M., Zaverucha, G.: Towards an Inductive Design of Distributed Object Oriented Databases. In: Proc. of the Third IFCIS Conference on Cooperative Information Systems (CoopIS98). IEEE CS Press, New York, USA, Ago (1998) 88-197
26. Bai□, F., Mattoso, M., Zaverucha, G.: Horizontal Fragmentation in Object DBMS: New Issues and Performance Evaluation. In: Proc. of the 19th IEEE Int. Performance, Computing and Communications Conf.. IEEE CS Press, Phoenix (2000) 108-114
27. Wrobel, S.: First Order Theory Refinement. In: L. De Raedt (ed.): Advances in Inductive Logic Programming. IOS Press, Amsterdam (1996)

28. Basilio, R., Zaverucha, G., Barbosa, V.: Learning Logic Programs with Neural Networks. 11th Int. Conf. on Inductive Logic Programming (ILP). Lectures Notes in Artificial Intelligence, Vol. 2157. Springer-Verlag, Strasbourg, France (2001) 15-26
29. Ruberg, G.: A Cost Model for Query Processing in Distributed Object Databases, MSc Thesis, COPPE, Federal University of Rio de Janeiro, Brazil (in portuguese) (2001)

Speeding up Recommender Systems with Meta-prototypes

Byron Bezerra¹, Francisco de A.T. de Carvalho¹, Geber L. Ramalho¹,
and Jean-Daniel Zucker²

¹ Centro de Informatica - CIn / UFPE,
Av. Prof. Luiz Freire, s/n - Cidade Universitaria, CEP 52011-030 Recife - PE, Brazil
{bldb,fatc,glr}@cin.ufpe.br

² PeleIA – LIP6 – Universite Paris VI,
4, Place Jussieu, 75232 Paris, France
{Jean-Daniel.Zucker}@lip6.fr

Abstract. Recommender Systems use Information Filtering techniques to manage user preferences and provide the user with options, which will present greater possibility to satisfy them. Among these techniques, *Content Based Filtering* recommend new items by comparing them with a *user profile*, usually expressed as a set of items given by the user. This comparison is often performed using the *k*-NN method, which presents *efficiency* problems as the user profile grows. This paper presents an approach where each user profile is modeled by a meta-prototype and the comparison between an item and a profile is based on a suitable matching function. We show experimentally that our approach clearly outperforms the k-NN method while they presenting equal or even better prediction accuracy. The meta-prototype approach performs slightly worse than *kd-tree* speed up method but it exhibits a significant gain in prediction accuracy.

1 Introduction

Information systems which filter in relevant information for a given user based on his/her profile are known as Recommender Systems. Such systems may use two sort of information filtering techniques for this purpose: the *Content Based Filtering* (CBF) and the *Collaborative Filtering*. Both techniques have been presenting good results and, since they are complementary [1], they tend to be used together [2, 3].

CBF recommends new items by comparing them with a user profile, usually expressed as a set of items given by the user (e.g., the set of books bought by the user in a online bookstore). This comparison is often performed using the k-NN method [4], which presents efficiency problems as the user profile grows. This problem becomes significant in web-systems, which may have millions of users. Techniques such as *kd-trees* can reduce the time required to find the nearest neighbor(s) of an input vector but suffer a reduction of the *prediction accuracy* [5].

This paper introduces a novel approach of CBF, the *Meta-Prototypes* (MP), which improves the speed of Recommender Systems and the prediction accuracy compared to the *kNN* method. Moreover, the meta-prototype approach performs slightly worse than *kd-tree* speed up method but it exhibits a significant gain in prediction accuracy. This method had been developed in the framework of the Symbolic Data Analysis [10] and had been firstly successfully applied in classification of a special kind of SAR simulated image [6]. In this sense, our work has the main contribution of, for the first time, adapt the *MP* technique to the recommendation domain improving efficiency without degrading accuracy.

In next Section, we describe briefly the state of the art in CBF speed up issue. In Section 3 and 4, we present respectively some concepts of symbolic data analysis and meta-prototype approach. In Section 5 we describe the adaptations we needed to introduce in order to cope with the recommendation domain. Then we present the results in the case study domain: the recommendation of movies. Finally, we draw some conclusions and point out future research directions.

2 Speeding Up Content-Based Filtering

The idea behind all variants of CBF is to suggest items that are similar to those items that the user has liked in the past. The notion of user profile used in this work is a set of examples of items associated with their classes. It is in fact the notion of user profile in extension. Particularly, in the movie domain the user profile is a set of movies with their respective grades.

In the *kNN* [7], the exemplars are original instances of the training set (items of the user profile). These systems use a distance function to determine how close a new input vector y is to each stored instance, and use the nearest instance(s) to predict the output class of y . Probably one of the main problems in the *kNN* method is its low efficiency in dealing with thousands of users and items. In fact, every item in a user profile needs to be compared with every item in the query set to make sure it is good or not for this user [4].

Three main approaches can be used to speed up of exemplar-based training algorithms, such as *kNN*. The first one is to modify the original instances (items of the user profile) using a new representation [7, 13]. This is the case of the *RISE* method [9]. Unfortunately, this method is not able to take into account multi-valued nominal attributes, which are common in describing items (e.g. the *cast attribute* of a movie). The second one is to reduce the set of original instances [8]. A good representative of this approach is the *Drop* method [8]. It is possible to reduce the instance set with *Drop* following by a change in representation of the reduced set using Meta-Prototypes. Therefore, it is a good work to do this experiment in order to evaluate the contribution of both methods.

The third approach consists of indexing the training instances. This is the case of *k-d trees* [5]. We will compare our approach with this one in Section 6.

3 Symbolic Data Analysis (SDA)

Symbolic data are more complex than usual data as they contain internal variation and they are structured. They come from many sources, for example in summarizing huge relational databases or as expert knowledge. The need to introduce new tools to analyze symbolic data is increasing and it is why SDA has been introduced [10].

SDA is a new domain in the knowledge discovery, related to multivariate analysis, pattern recognition, data bases and artificial intelligence. SDA provides suitable tools to work with higher-level data described by multi-valued variables where the entries of a data table are sets of categories, intervals or probability distributions, related by rules and taxonomies. So, SDA methods generalize classical exploratory data analysis methods, like factorial techniques, decision tree, discrimination, neuronal methods, multidimensional scaling, clustering and conceptual lattices.

In classical data analysis, the input is a data table where the rows are the descriptions of the individuals, and the columns are the variables. One cell of such data table contains a single quantitative or categorical value. However, sometimes in the real world the information recorded is too complex to be described by usual data. That is why different kinds of symbolic variables and symbolic data have been introduced [10]. For example, an interval variable takes, for an object, an interval of its domain, whereas a categorical multi-valued variable takes, for an object, a subset of its domain. A modal variable takes, for an object, a non-negative measure (a frequency or a probability distribution or a system of weights) defined on its support (set of values included in the variable domain).

A symbolic description of an item is a vector whose descriptors are symbolic variables. In the approach explained in the next section, the user profile is a vector whose descriptors are modal symbolic variables. The comparison between a user profile and an item to be recommended is accomplished by a suitable matching function. This approach has been applied successfully on image recognition [6].

4 Meta-prototype Approach

The CBF based on meta-prototypes has two points to consider: i) every instance is represented by a modal symbolic description and, ii) the user profile is represented by one or more modal symbolic objects or, equivalently, meta-prototypes. The point *i* is the pre-processing phase and the step *ii* has two sub-tasks: first, the items of the user profile is represented by modal symbolic descriptions (pre-processing phase) and second, this descriptions are aggregated by a generalization step.

4.1 Pre-processing

Each instance is described as a vector of attributes. This description may include several kinds of attributes: single valued qualitative (nominal or ordinal), multi-valued qualitative (ordered or not) and textual.

Table 1. Attributes in movie domain

Attribute	Type	Example
Genre	Multi-valued qualitative	Drama
Country	Nominal single valued qualitative	EUA
Director	Nominal single valued qualitative	Steven Spielberg
Cast	Multi-valued qualitative	Tom Hanks, David Morse, Bonnie Hunt, Michael Clarke
Year	Ordinal single valued qualitative	1999
Description	Textual	The USA government offers one million dollars for some information about a dangerous terrorist.

The aim of the pre-processing step is to represent each item as a modal symbolic description, i.e., a vector of vectors of couples (value, weight). The items are the input of the learning step.

The couples of (value, weight) are formed according to the type of the descriptors:

1. if the descriptor is single valued or multi-valued qualitative or single valued quantitative discrete, each value is weighted by the inverse of the cardinal of the set of values from its domain taken by an individual;
2. if the descriptor is textual some *Information Retrieval* methods are applicable, such as *Centroid* and *TFIDF* [11]. The centroid is a technique used to extract a weight system of a text through the main important words of a text. By this way it can be thought as a multi-valued qualitative attribute like the *cast* one. The TFIDF technique includes the relevance of each word in the whole documents base with the importance of the word in a single text document.

The modal symbolic description of the movie in table 1, for the attributes *Cast* and *Description*, is shown in table 2.

Table 2. Modal symbolic description of the example of table 1.

Attribute	Movie' Modal Symbolic Description (x)
Cast	(0.25 Tom Hanks, 0.25 David Morse, 0.25 Bonnie Hunt, 0.25 Michael Clarke,
Description	(0.125 USA, 0.125 government, 0.125 offers, 0.125 million, 0.125 dollars, 0.125 information, 0.125 dangerous, 0.125 terrorist

4.2 Generalization

This step aims to represent each user profile as a modal symbolic object (*Meta-Prototype*). The symbolic description of each user profile is a generalization of the modal symbolic description of its segments.

The meta-prototype representing the user profile is also a vector of vectors of couples (value, weight). The values, which are present on the description of at least an item already evaluated by the user, are also present in the user profile description (meta-prototype). The corresponding weight is the average of the weights of the same value presenting in the item descriptions. Suppose there are two movies in the user profile where the *Cast* attribute is presented in table 3. Table 4 shows the simplified MP of the user profile exemplified in the table 3.

Table 3. Examples of movies evaluated by some user

Attribute	Movie 1	Movie 2
Cast	Tom Hanks, Michael Clarke, James Cromwell, Ben Kingsley, Ralph Fiennes	Caroline Goodall, Jonathan Sagall, Liam Neeson, Michael Clarke

Table 4. The meta-prototype concerning with the user profile exemplified in the table 3

Attribute	User Meta-Prototype (u)
Cast	((0.2 Tom Hanks, (0.2+0.25) Michael Clarke, 0.2 James Cromwell, 0.2 Ben Kingsley, 0.2 Ralph Fiennes, 0.25 Caroline Goodall, 0.25 Jonathan Sagall, 0.25 Liam Neeson)*0.5)

4.3 Comparing an Item with a User Profile

The recommendation of an item to a user is based on a matching function, which compares the symbolic description of the item with the symbolic description of the user. The matching function measures the difference in contents, by a context dependent component, and in position, by a context free component, between an item and an user descriptions.

Let $x = (x_1, \dots, x_p)$ and $u = (u_1, \dots, u_p)$ be the modal symbolic description of an item and the meta-prototype of a user, respectively, where $x_j = \{(x_{j1}, w_{j1}), \dots, (x_{jk(j)}, w_{jk(j)})\}$, $u_j = \{(u_{j1}, w_{j1}), \dots, (u_{jm(j)}, w_{jm(j)})\}$, $j = 1, \dots, p$. $k(j)$ and $m(j)$ are the number of categories of domain O_j of variable y_j , present in item and user descriptions, respectively. The comparison between the item x and the user u is accomplished by the following matching function:

$$\phi(x, u) = \sum_{j=1}^p (\phi_{cf}(x_j, u_j) + \phi_{cd}(x_j, u_j)) \quad (1) \text{ The matching function.}$$

The *context free* component of the matching function ϕ_{cf} is defined as,

$$\phi_{cf}(x_j, u_j) = \frac{|\bar{X}_j \cap \bar{U}_j \cap (X_j \oplus U_j)|}{|X_j \oplus U_j|} \quad (2) \text{ The context free component of the matching function.}$$

where $X_j = \{x_{j1}, \dots, x_{jk(j)}\}$, $U_j = \{u_{j1}, \dots, u_{jk(j)}\}$ ($\bar{X}_j$ and $\bar{U}_j$ are the complementary of sets X_j and U_j). If domain O_j is ordered, let $x_{jB} = \min X_j$, $x_{jT} = \max X_j$, $u_{jB} = \min U_j$ and $u_{jT} = \max U_j$. The join $X_j \oplus U_j$ [12] is defined as:

$$X_j \oplus U_j = \begin{cases} X_j \cup U_j, & \text{if domain } O_j \text{ is non ordered} \\ \{\min(x_{jB}, u_{jB}), \dots, \max(x_{jT}, u_{jT})\} \end{cases} \quad (3) \text{ The join operator.}$$

The *context dependent* component of the matching function ϕ_{cd} is defined as,

$$\phi_{cd}(x_j, u_j) = \frac{1}{2} \left[\sum_{k/x_k \in X_j \cap \bar{U}_j} w_k + \sum_{m/u_m \in \bar{X}_j \cap U_j} \right] \quad (4) \text{ The context dependent component of the matching function.}$$

The meta-prototype does not have to be created again if a new item is evaluated.

5 Meta-prototype in the Recommendation Domain

In this section we discuss some improvements of this model, well adapted to the recommendation domain.

5.1 Two Meta-prototypes

In general, the recommendation systems acquires the satisfaction of the user concerning with a suggestion by getting his/her evaluation. Therefore, the recommendation domain has some additional information, which has not yet been considered, such as the user evaluations. So, how to use the “negative” (e.g., the movies which got a grade 1 or 2) evaluations of the user? We have reflected about this problem and decided to use the negative user evaluations to construct a brand new meta-prototype that incorporates them. Therefore, the user profile is represented by two MP: a positive meta-prototype (u^+) and a negative meta-prototype (u^-). An item with grade 1 and 2 goes in u^- , and an item with grade 4 and 5 goes in u^+ . There are three choices for items with grade 3: i) they must not be added in any meta-prototype because the user has no opinion about the movie (don’t care); ii) they must be added in u^- and iii) they will be added in u^+ . The decision about that depends on experimental analysis.

The matching function of equation 3 becomes:

$$\Phi(x, u) = \frac{\phi(x, u^+) + (1 - \phi(x, u^-))}{2} \quad (5) \text{ The matching function } \Phi \text{ considering two MP, where } \phi \text{ is defined in equation 3.}$$

5.2 Replication

The user grades have other hypothesis, which may be stronger than the later discussed in section 5.1. It is clear that a grade 5 means very good whereas a grade 4 is just good, and a grade 1 means very bad whereas a grade 2 is not bad enough. One way to model this behavior in our approach is: i) items with grade 5 has a higher proportion than the items with grade 4 in $\mathbf{u}^+$, and, equivalently, ii) items with grade 1 has a higher proportion than the items with grade 2 in $\mathbf{u}^-$. In any case, an item with grade 3 must not be replicated since it is in the average grade.

5.3 Refinements

The problems discussed in Sections 5.1 and 5.2 suggested some previous experiments in order to refine our model, before attempt the main experiments. Because it is not the scope of this paper we just present the conclusions of this previous experiments. The first conclusion is that two MP, as described in section 5.1, improve the prediction accuracy if compared with the original MP approach. Additionally, the replication (Section 5.2) improves the prediction accuracy. Finally, the results showed the items with grade 3 degrade the prediction accuracy in any case. Then items with this grade will be ignored.

The refinements of our model is summarized as: i) items with grade 1 are added 3 times in $\mathbf{u}^-$; ii) items with grade 2 are added twice in $\mathbf{u}^-$; iii) items with grade 4 are added twice in $\mathbf{u}^+$; and iv) items with grade 5 are added 3 times in $\mathbf{u}^+$. After these refinements, the MP method showed a prediction accuracy as good as kNN one.

6 Experiments and Results

The following experiments are based on a subset of EachMovie database [14] consisting of 22.867 users and 1.572.965 numeric ratings between 1 to 5 (1:very bad, 2:bad, 3:reasonable, 4:good, 5:very good) for 638 movies. The original database from EachMovie has no description of movies and for this reason it would not be possible to test CBF on the whole base. So, the original movie table was matched with a second database of movies with the complete description of movies in Portuguese idiom.

6.1 Results and Discussion

The aim of the experiments of this section is to compare the prediction accuracy and the speed of the *kNN*, *k-d Tree* and *MP* methods. For all experiments it was considered the following settings: i) the kNN and k-d Tree with 5 or 11 nearest neighbors; ii) the prediction accuracy was measured according to *Breese*¹ criterion, which is very ap-

¹ The *Breese* criterion measures the utility of a sorted list produced by a recommendation system for a particular user. The main advantage of this criterion for real systems is that the estimated utility takes into account the user generally consumes only the first items in the sorted list. See [15] for details.

propritate for this subject in Recommender Systems; and iii) the speed was measured by the average time spent for produce the suggestions in seconds.

In each experiment, 50 users with at least 300 evaluations were randomly chosen. For each user, it was chosen from the evaluated items: i) 200 items for the query set and ii) 100 distinct items for the training set. Moreover, the number of items (m) of the training set was varied with $m \in \{5,10,20,40,60,80,100\}$. Finally, it was compared for each user the speed and prediction accuracy of the recommendation through the query set. The figures 1 and 2 show the results of this experiment.

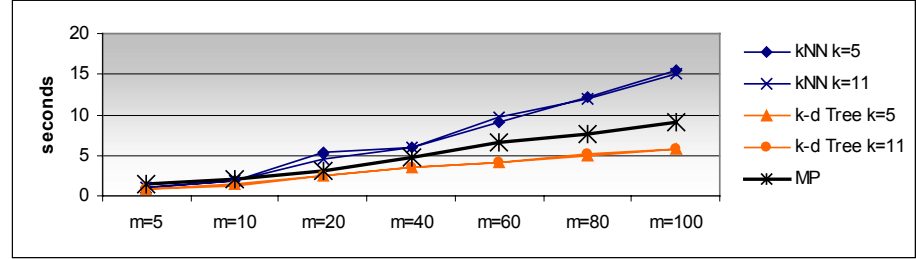


Fig. 1. The speed results

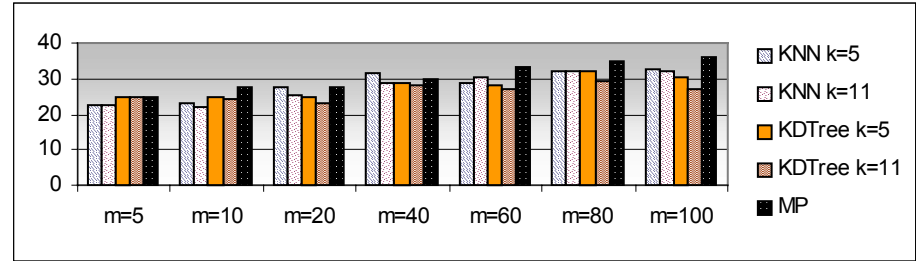


Fig. 2. The prediction accuracy results according to Breese

The figure 2 indicates that MP method shows the best prediction accuracy among the evaluated methods. The figure 1 shows that m higher than 60 the response time of kNN is higher than 10 seconds, which maybe considered a bad behavior. The figure 1 also shows that MP performance is lower than k-d Tree whereas his response time is not bad enough as the kNN one. However, in a real recommender system the prediction accuracy is as critical as the response time. Therefore, the MP method is very useful for these systems, because it gets the best prediction accuracy with a good response time, which takes twice as much time than k-d Tree method.

Nevertheless, if the one favors the prediction accuracy instead of response time, we note that the MP and k-d Tree methods are closer concerning the response time. In order to support this conclusion, we considered figures 3 and 4, which were inferred from the results presented in figures 1 and 2. According to figure 3, the value of 29 for Breese prediction accuracy is achieved with 80 items concerning with k-d Tree ($k=5$) whereas it is achieved just with a half of these items, e.g. 40 items, for MP approach. As another example, if the recommendation system requires a prediction accuracy of

27, then it is sufficient to use a training set with 10 items for MP and spent just about 2 seconds for generating the recommendations. Nevertheless, for a prediction accuracy of 27, it is needed 40 items which implies about 3.8 seconds for the same task with k-d Tree. According these two examples, it seems that the difference in response time between both methods showed in figure 1 disappear if the system goal is to furnish recommendations with a fixed level of accuracy.

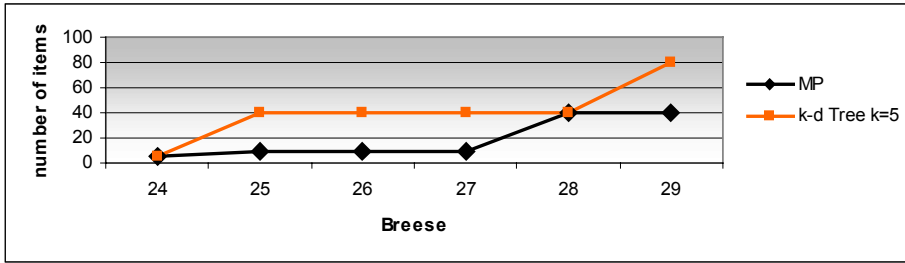


Fig. 3. The relation of the number of items in the training set versus the prediction accuracy according to Breese criterion for MP and k-d Tree ($k=5$)

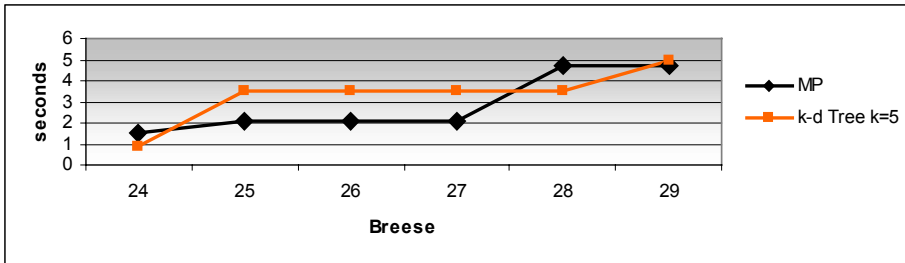


Fig. 4. The relation of the speed in seconds versus the prediction accuracy according to Breese criterion for MP and k-d Tree ($k=5$)

7 Conclusions

CBF techniques, such as kNN, which is commonly used in Recommender Systems, suffer from speed problems. There are some works proposing a solution for this problem, but, among those applicable to the domain, none of them improves the speed without degrading the prediction accuracy. The MP method fulfills this requirement.

In the future, we plan to apply the MP approach to other domains where different approaches have been successfully used. We also intend to use techniques such as the Drop method [8] before applying MP modeling in order to assess its impact. Finally, we will include the analysis of storage gain using MP, since we think that it can provide a significant reduction, which is not the case of techniques such as kd-trees.

Acknowledgements

This paper is supported by grants from the joint project Smart-Es (COFECUB-France and CAPES-Brazil) as well as by grants from CNPq-Brazil.

References

- [1] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining Content-Based and Collaborative Filters in an Online Newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*, August 19 1999.
- [2] Joshua Alspector, Aleksander Kolcz, and Nachimuthu Karunanithi. Comparing Feature-Based and Clique-Based User Models for Movie Selection. In *Proceedings of the Third ACM Conference on Digital Libraries*, pages 11-18, 1998.
- [3] Smyth, B. & Cotter, P. (1999) Surfing the Digital Wave: Generating Personalised TV Listings using Collaborative, Case-Based Recommendation. *Proceedings of the 3rd International Conference on Case-Based Reasoning*, Munich, Germany, 561-571.
- [4] Arya S.: Nearest Neighbor Searching and Applications, Ph.D. thesis, University of Maryland, College Park, MD, 1995.
- [5] Bentley J., "Multidimensional binary search trees used for associative searching", *Communications of the ACM*, Vol.18, pp. 509-517, 1975.
- [6] De Carvalho, F.A.T., Souza, R.M.C.M. and Verde, R. (submitted): A Modal Symbolic Pattern Classifier.
- [7] Cover, T. M., and P. E. Hart (1967). Nearest Neighbor Classifiers. *IEEE Transactions on Computers*, 23-11, November, 1974, pp. 1179-1184.
- [8] D. R. Wilson and T. R. Martinez. Reduction techniques for exemplar-based learning algorithms. *Machine Learning*, 38(3):257-268, 2000.
- [9] Domingos, Pedro (1995). Rule Induction and Instance-Based Learning: A Unified Approach. to appear in *The 1995 International Joint Conference on Artificial Intelligence (IJCAI-95)*.
- [10] Bock, H. H. and Diday, E. (2000): *Analysis of Symbolic Data*. Springer, Heidelberg.
- [11] Baeza, Y. and Ribeiro, N. *Modern Information Retrieval*.
- [12] Ichino, M. and Yaguchi, H. (1994): Generalized Minkowsky Metrics for Mixed Feature Type Data Analysis. *IEEE Transactions system, Man and Cybernetics*, 24, 698-708
- [13] Verde, R., De Carvalho, F.A.T. and Lechevallier, Y. (2000): A Dynamical Clustering Algorithm for Symbolic Data, in *25th Annual Conference of the Germany Classification Society*, Munich (Germany), 59-72
- [14] McJones, P. (1997). EachMovie collaborative filtering data set. DEC Systems Research Center. <http://www.research.digital.com/SRC/eachmovie/>
- [15] Herlocker, Jonathan Lee. Understanding and Improving Automated Collaborative Filtering Systems, cp 3.

ActiveCP: A Method for Speeding up User Preferences Acquisition in Collaborative Filtering Systems

Ivan R. Teixeira¹, Francisco de A. T. de Carvalho¹,
Geber L. Ramalho¹, and Vincent Corruble²

¹Centro de Informática - CIn/UFPE - Cx. Postal 7851
50732-970, Recife, Brazil

{irt,fatc,glr}@cin.ufpe.br

²Laboratoire d'Informatique de Paris VI- LIP6 – 4
Place Jussieu, 75232, Paris, France
Vincent.Corruble@lip6.fr

Abstract. Recommender Systems enhance user access to relevant items {information, product} by using techniques, such as collaborative and content-based filtering, to select items according to the users personal preferences. Despite the success perspective, the acquisition of these preferences is usually the bottleneck for the practical use of this systems. Active learning approach could be used to minimize the number of requests for user evaluations but the available techniques cannot be applied to collaborative filtering in a straightforward manner. In this paper we propose an original active learning method, named *ActiveCP*, applied to KNN-based Collaborative Filtering. We explore the concepts of item's *controversy* and *popularity* within a given community of users to select the more informative items to be evaluated by a target user. The experiments testifies that *ActiveCP* allows the system to learn fast about each user preference, decreasing the required number of evaluations while keeping the precision of the recommendations.

1 Introduction

Recommender Systems are a recent innovation that intends to enhance user's access to relevant and high quality recommendation by automating the process by which recommendations are formed and delivered [[6]]. They collect information about users' preferences and use Information Filtering techniques to make use of the knowledge in their information base so as to provide the user with a personalized recommendation.

Two common approaches used for information filtering are content-based filtering and collaborative filtering [[4]]. Content-based filtering relies on the intuition that people seek items (e.g., books, CDs, films) with a content similar to what they have liked in the past. Collaborative Filtering is based on human evaluations shared within

a community. It is common to ask for recommendations from friends or colleagues about relevant subjects and to base our choice on others' evaluations. Also one would give more credit to recommendations obtained from people that are known to have similar taste. Automated Collaborative Filtering applies to virtual communities of users that share opinions about a subject. It aims at enhancing person to person recommendations by adding scalability and anonymity to the process.

One important issue in Recommender Systems concerns the method by which the user provides preferences. Many systems work based on explicit evaluation [[6]][[11]], where the user provides direct information to the system by stating his preference for items, usually by giving some form of rating. However providing explicit evaluation may become a tedious task for the user, as the number of evaluations necessary for the machine to learn sufficiently about his interests can be too large.

Active learning techniques [[10]] could be used to minimize the number of requests for evaluations presented to users without reducing the systems ability to propose good recommendations. Most of these techniques can be straightforwardly applied to content-based filtering [[7],[13]], in which the user profile is induced from items *content* description (e.g. film's director, actors, etc.) and the similarity between items can be inferred. Unfortunately, this is not the case of collaborative filtering, in which the learning process relies only on the ratings given to the items. In fact, there is no sense in assessing similarity between items in collaborative filtering, since the very concept of CF relies in the similarities between users. Indeed, one of the advantages of this technique is exactly that it is not necessary to describe items' content in order to recommend them. This difficulty probably explains the fact that, to our knowledge, active learning techniques have not been applied yet to Recommender Systems based on Collaborative Filtering.

In this paper we propose an original active learning method, named *ActiveCP*, to Collaborative Filtering. We explore the concepts of item's *controversy* and *popularity* in a given users community to heuristically (1) minimize the quantity of user evaluations required to reach a target quality of recommendation, or (2) maximize the quality of recommendations for a fixed quantity of user evaluations. The method has been tested with success compared to a random item selection policy, which is analogous to how user preferences are obtained in current Recommender Systems.

In next section we discuss the need for active learning in Recommender Systems. In section 3 we describe the KNN-CF, a common filtering algorithm on Recommender Systems, which we based on to develop our active learning method. In section 4 we discuss how to select informative items for user evaluation. In section 5 we describe the experiments with the selection of items with our selection method *ActiveCP*. We finish presenting conclusions and future works.

2 Active Learning in Recommender Systems

Information filtering usually applies learning methods to learn from examples of user preferences. The system's task is to learn about each user's preferences so as to be able to predict future evaluation of unseen items. Most Recommender System require their users to manually express their opinion about the items. In this case, producing

examples is a costly process and it is desirable to reduce the number of training examples while maintaining the quality of future predictions. Instead of presenting items to be evaluated in an undefined order, the idea is to require user evaluations over specific items, those that would assist the system to learn more relevant aspects about his profile.

Machine Learning researchers have proposed a framework for selecting or designing more relevant and informative training examples for application domains where producing examples is a costly process or for those where it is desirable to use a reduced number of training examples without losing the quality of the classifiers. *Active learning* is the paradigm where the learning algorithm has some control on the inputs on which it trains. Various active learning algorithms have been developed to speed up learning in classification algorithms such as Neural Network [[5]], rule induction [[9]] and KNN [[7],[10]].

Active learning algorithms are divided in two subfields: *membership queries* and *selective sampling*. In membership queries it is possible for the algorithm to construct artificial examples and then ask for its classification. The problem with this methodology is the possibility of the algorithm coming up with badly constructed or not meaningful examples [[10]]. Selective sampling is a more restrictive approach than membership queries. It consists in, from a set of examples whose classification is unknown, selecting the next example that will be classified by the supervisor. The task in a selective sampling algorithm is to select a set of examples for which their classification are very suitable to form a consistent hypothesis rapidly and with fewer classifications as possible. In the case of Recommender Systems, the selective sample approach should naturally be used, since the creation of artificial items is complex and may be meaningless.

3 Collaborative Filtering Algorithm

Collaborative filtering was built up on the assumption that a good way to filter information is to find other people with similar interests, and use their evaluations on items to predict its interest for a target user. This filtering technique soon became very popular due to its advantages over content based filtering, such as the ability to filter information based on aspects determined by humans and to provide diversified recommendations [[1]]. Based on this paradigm, many algorithms were developed intending to automate the process of identifying like-minded users and performing cross recommendations of information. For this purpose very different approaches have been used, such as Neural Networks [[2]], rule-inducting [[4]] and Bayesian networks [[3]]. Aside all these different approaches, the KNN-based Collaborative Filtering (KNN-CF) is the methodology that had most acceptance due to its simplicity and efficiency on predicting users evaluation [[8]].

In the KNN-CF methodology, the prediction of an item evaluation for a target user is computed based on the item evaluations of other users similar to him/her. The similarity between two users is based on the ratings given to the items they evaluated in common. A common approach to compute users' similarity is using *Pearson correlation coefficient* [[12],[2]]. Herlocker et. al. [[8]] have made an empirical analysis of similarity metrics used for CF to conclude that Pearson correlations had

the best results on predictions accuracy. It is also suggested that the similarity measured by this coefficient should be weighted by the number of shared evaluated items, in order to avoid high similarity between users who have few items in common [[8]].

Once the similarities of all system users with the target user are computed, the most similar ones that evaluated the item to be predicted are selected to form the target user *prediction neighborhood* for this item. The evaluations of the neighbors for the target item are combined: the contribution of each neighbor on the prediction is weighted by his similarity with the target user. The prediction $p_{a,i}$ of item i for user a is computed as follows:

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n [(r_{u,i} - \bar{r}_u) w_{a,u}]}{\sum_{u=1}^n w_{a,u}} \quad (1)$$

where $\bar{r}_a$ is the mean rating for target user a and n is the size of the prediction neighborhood (number of neighbors that evaluated item i). For each user u on the prediction neighborhood the difference between his/her mean rating $\bar{r}_u$ (for all evaluated items) and rate $r_{u,i}$ for item i is weighted by his/her similarity $w_{a,u}$ with target user to compute final prediction. Equation (1) is a general formula used in several KNN-CF [[8]][[2]].

Remark that the number of items a target user evaluates influences the determination of his/her neighborhood. The more items a user evaluates, the more precise will be the similarity measurements between users, and consequently the more precise will be the predictions based on the neighborhood.

In our work, we have used the framework for KNN-CF using: the Pearson correlations as proposed in [[8]], a prediction neighborhood of size 40 and final predictions computed as showed in equation (1).

4 Active Learning for KNN-based Collaborative Filtering

As stated in the introduction, our aim is to maximize the utility of the evaluations provided by the user so as to optimize the quality of the system's recommendation. The question we face is how to obtain a measure that reflects the informative value of an evaluation in order to ascertain the taste of the user. The task can be seen as sampling: given a user, a list of objects, and evaluations of these objects by a number of previous user, which sublist of objects, once evaluated by a new user U , would optimally inform our system on U 's taste? Even under the simplifying assumption that U is able, if required, to evaluate any object of the set, this task is overly complex. So in the following, we will take a greedy approach, which assumes that there is an existing sublist of objects already evaluated by U , and the task is reduced to the question of finding one optimal item to add to the sublist for evaluation.

Now that the problem is better stated, we can try to find measures for assessing the informative value of an item evaluation to a target user.

4.1 Selecting Examples in KNN-based Filtering

In the context of instance-based learning techniques, such as KNN, some active learning methods have been proposed [[7]] and other methods, originally designed for reducing the number of training examples in instance-based learning, could also be adapted to perform selective sampling [[13]].

The problem is that these methods rely on the similarity between the examples (items) in order to explore notions such as *border points* and *center points*. Center points are training examples located inside clusters of points within the same classification, while border points are located in between clusters of different classifications. In the case of content-based filtering, this is not a problem since the items have a description (typically in an attribute-value form). However, collaborative filtering, as discussed in the introduction, relies only on the ratings given to the items and there is no sense in assessing similarity between items in this context, as discussed in the introduction. In other words, in the context of KNN-CF, the similarity is measured between users and not between the items we wish to select.

It will be then necessary to adapt some general active learning notions to provide a KNN-CF Recommender System with selective sampling capabilities. In other words, item selection criteria, not based on inter-items similarity, must be found to apply an active learning strategy to KNN-CF. In next section, we propose two selection criteria: controversy and popularity.

4.2 Controversy

An intuitive criterion that could be used in sampling is the item's controversy. Items loved or hated by everybody are likely to have low informative value to model the taste of a new user, since this user is statistically likely to be of the same opinion as the vast majority of the other users. Oppositely, an item for which users have expressed a wide range of evaluations (from extremely positive to highly negative) is probably more informative. In fact, knowing a new user's appreciation of this controversial item will help the system to identify more precisely his/her neighborhood, i.e. the users that are more similar to him/her.

The controversy of an item in a KNN-CF is analogous to the notion of border points [[13]], discussed in Section 4.1. Indeed, in both cases an example is considered informative because it helps the system to better discriminate groups of examples with opposing characteristics.

Various functions can be used to measure the controversy of an item based on previous evaluations of a set of users. The most natural one is the *variance* of the distribution of notes given to the item. Indeed this is the measure typically used to evaluate the dispersion of a distribution, which corresponds to our intuitive idea of controversy. After some preliminary tests, we have adopted the variance as the method for determining item's controversy. However, further reflections have been necessary due to the fact that, since the variance normalizes the dispersion by the number of samples, it neglects the fact that not all items have the same number of evaluations. For instance, given an item, the variance measure produces the same value whether two or hundred of users have evaluated it with equally opposing opinions. In this bizarre situation, even if the controversy *intensity* is the same, we

could say that the controversy *width* is different. In this sense, the controversy intensity measures the distribution of evaluation notes, whereas the controversy width depends on how many people have evaluated the controversial item.

In order to solve this problem, instead of using all users, we have decided to measure the controversy of an item using a fixed number of users, selected among those that have given an evaluation to this item. By fixing the number of users we guarantee that the width of the controversy computed for the items will be the same, focusing the measure only on the intensity of the controversy. Though fixing the number of users required to measure the controversy of an item seems a strong imposition, we consider it as a first approach to solve the problem of width and intensity of the controversy measure. This is a general problem in CF that requires further efforts to advance towards an ideal solution.

4.3 Popularity

Item's popularity indicates the number of evaluations made for it: the more the users evaluate an item, the more popular it is. As in KNN-CF the similarity between two users is measured considering the items they evaluated in common, we have hypothesized that the popularity of an item could be relevant to determine the neighborhood of a target user.

In fact, when a target user evaluates an item also evaluated by another user, there is more available information to measure similarity between these two users. Consequently the greater the number of users that evaluate item i , the greater will be the information about the similarity of the target user with other users of the system. In this sense, a target user should evaluate first popular items, since evaluating an unpopular item would result in much less information gain.

5 Empirical Evaluation

In this section we show some experiments intended to validate the controversy and popularity criteria for selective sampling in KNN-CF. We have used *Eachmovie database* (<http://research.compaq.com/SRC/eachmovie/>), which consists of 72,916 users that evaluated 1,628 movies on a 5 level score interval (1,2,3,4 and 5). For this work we have selected randomly a subset of 10,000 users in order to speed up experimental tests without losing generality of results.

5.1 Metrics

We have applied two metrics in our evaluations: ROC, appropriate to decision support recommendations and Breese, appropriate for ranked lists recommendations. These prediction accuracy measures are suggested by the Recommender Systems literature [[3],[8]]. To use ROC we considered items evaluated by users with 1, 2 and 3 as not relevant, and 4 and 5 as relevant as suggested in [[8]]. To use Breese we considered the value 3 as the central score, i.e. the score that indicates users neutral preference

and 5 as our half-life, i.e. the 5th item is the item the user will have 50% chance of viewing in a list of recommendation, both values being suggested in [[3]].

5.2 Experiments Organization

In our experiment the task of the system is to select one item at a time until the number of items evaluated reaches a determined size. For our experiments, from the users set, we randomly selected 1,000 users with at least 100 evaluations, and randomly select 100 of the items he evaluated. For each user, the items selected are divided in 5 sets of 20 items each to provide a 5-fold cross-validation. The whole process is described in the following algorithm:

```

Input          U[1..5]: user original items subsets to be
                    selected
                    n: number of items to select
Output         A: prediction accuracy
UserSelectionTest(U[1..5] , n)
1.  For i= 1 to 5
2.    Assign SelectionSet S <-  $\emptyset$  , TestSet T <-  $\emptyset$  ,
        UserEvaluationsSet E <-  $\emptyset$ 
3.    T <- U[i] //a given subset of U
4.    S <- the other 4 subsets of U
5.    While |E| < n
6.      E <- SelectItem(S,E)
7.      P <- Predict(T,E)
8.      a[i]<- Accuracy(P,T)
9.  Return the average accuracy of a[i] , i = 1...5

```

The prediction accuracy concerning the selection of n items for a user is computed by the method *UserSelectionTest* using the array $U[i]$ of his 5 items sets. The method *SelectItem(S,E)* selects one item from the set S that is not contained in the set E and adds it to E until the required number of n items is reached. For the selection of items of method *SelectItem()* we use:

1. **Random selection** is the base line for evaluating selection methods, since it approximates the way user evaluations are acquired in current KNN-CF Recommender Systems and we have found no other works on active learning in CF that can be compared to our results. The performance of random selection was averaged over the result of 10 repetitions of the test procedure.
2. **Selective Sampling selection** for simulating active learning according to the two criteria we have introduced in previous section.

The method *Predict(T,E)* implements the KNN-CF algorithm presented in section 3. It uses the evaluations of the items in the set E to generate predictions P of the user evaluation of items in the set T . The method *Accuracy(P,T)* measures the accuracy of system predictions P for the items in the set T using the metrics ROC or Breese. The output A for each user is the mean of the accuracy for this user in the cross validation. The performance of an selection method will be the mean accuracy of the 1,000 users tested using the method *UserSelectionSet()*.

5.2.1 Selecting with Controversy and Popularity

We have started the experiments applying the controversy, measured by the variance, as a selection criterion for selective sampling. As discussed in Section 4.2, to deal with controversy width variations, we use a fixed set of users (100, 500, 1000 and 5000 users). The best prediction accuracy has been achieved using a set of 1000 users randomly selected from entire user base. Regarding item popularity, we have directly applied the measure suggested in Section 4.3, i.e., the number of users that evaluated (regardless the given rate) a given item.

In our experiments, the predictions based on the set of items selected according to controversy or popularity presented better results than the predictions based on items selected randomly. However, the difference is not large enough to prove that any of these selection criteria is significantly superior to random selection (see Figure 1).

5.3 ActiveCP

Through in our experiments we observed that the controversy and the popularity of an item are somehow orthogonal within the community of users: controversial item may be either popular or not, as well as an item for whose quality there is a consensus. Thus we decided to combine these two selection criteria into a single criterion named CP.

In order to combine these criteria, for each item two values are associated indicating its order of preference according to the controversy and popularity criteria. For an item i ranked in position p_i in a list with n items, a value v_i associated to this item is calculated as follows:

$$v_i = \frac{p_i}{1-n} - \frac{n}{1-n} \quad (2)$$

The equation (2) maps the position p_i of i in the rank to a value $v_i \in [0,1]$, in a way that the first element of the rank ($p_i = 1$) is mapped to $v_i = 1$, and the last item in the rank ($p_i = n$) is mapped to $v_i = 0$. The values vc_i and vp_i , corresponding to the ranking according to controversy and popularity respectively, are combined in a single value CP_i as follows:

$$CP_i = w_c vc_i + w_p vp_i \quad (3)$$

Where w_c and w_p are weights given for controversy and popularity respectively.

We have tried several values for the weights w_c and w_p changing the importance of controversy and popularity on the selection of items. The best result was found using what we called CP 5, a criterion that combine controversy and popularity with weights w_c and w_p equal to 0.5 (i.e., controversy and popularity have equal strength). Results for other tests are suppressed due to space shortage. The figure 1 shows that the prediction accuracy using the evaluations of items selected by the ActiveCP with CP5 is superior to the accuracy of all other selection methods tested. Table 1 compares the prediction accuracy using the items selected randomly and selected according to ActiveCP with CP5. Table 1 shows the result of a significance test (p-value) at a level $\alpha = 0,1$, which means that the hypothesis that the two accuracy values (random and active) are equivalent can be reject with at least 90% of certitude.

According to this statistical test, the performance of our selection method ActiveCP is significantly superior to the performance of random selection for all numbers of selected items.

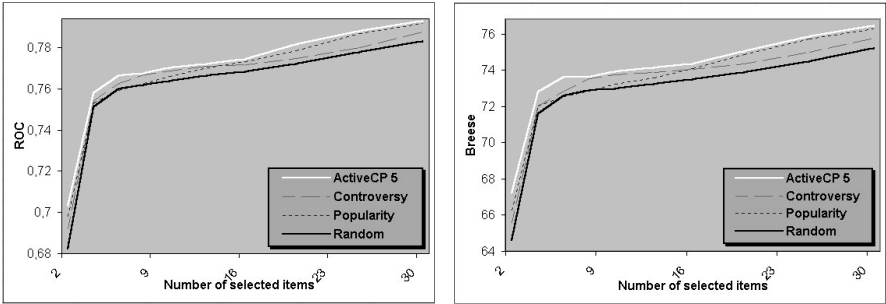


Fig. 1. Prediction accuracy using the evaluations of item selected according to different criteria

Another interesting way to view the advantage of the ActiveCP method is showed in figure 2. There, it is plotted the number of evaluations a user would have to perform when items are selected with ActiveCP to obtain the same performance of a number of random selected items. This figure shows that the evaluation of items selected by the method ActiveCP allows the system to reach the same prediction accuracy of random selection with a much lower number of evaluations.

Table 1. Result of significance test (p-value) for the hypothesis of superiority of ActiveCP 5 over random selection according to ROC and Breese metrics

Number of items	2	4	6	8	10	13	16	20	25	30
Roc	0,000	0,057	0,066	0,097	0,074	0,096	0,082	0,016	0,010	0,012
Breese	0,000	0,007	0,016	0,060	0,023	0,028	0,035	0,006	0,001	0,003

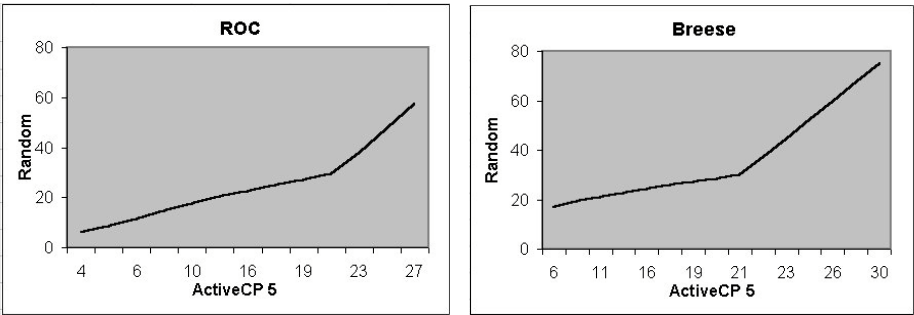


Fig. 2. Number of evaluations a user must perform to obtain the same performance when the items evaluated are selected randomly and by ActiveCP 5. Results for ROC and Breese

6 Conclusions

The acquisition of users' preferences through explicit evaluations of items is an important issue in recommender systems. In this paper we introduced an original active learning method (ActiveCP) well adapted to recommender systems employing KNN-based collaborative filtering, which is the most currently used filtering technique. ActiveCP combines the notions of controversy and popularity to select the items that are probably the most informative, reducing the number of required evaluations while keeping the precision of the recommendations. It was shown that ActiveCP is able to improve the learning about users preferences when compared to a random item selection policy, which is analogous to how user preferences are obtained in current Recommender Systems.

So far, we have measured controversy and popularity globally, i.e. considering the entire set of users. In a near future, we plan to apply these two criteria to the neighborhood of each target user. In other words, we intend to measure controversy and popularity among set of users closest to the target user. We suppose that knowing whether an item is controversial or popular within a user neighborhood will be more informative than in the whole user base, since it is the user neighborhood that is used to predict his/her evaluations on new items.

Acknowledgements

This paper is supported by grants from the joint project Smart-Es (COFECUB-France CAPES-Brazil) as well as by grants from CNPq-Brazil.

References

- [1] Balabanovic, M., Shoham, Y.: *Fab: Content-Based, Collaborative Recommendation*. Communications of the ACM, 40:3 (1997) 66-72.
- [2] Billsus, D., Pazzani, M. J.: *Learning Collaborative Information Filters*. Fifteenth International Conference on Machine Learning. (1998) 46-54.
- [3] Breese, J. S., Heckerman, D., Kadie, C.: *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence. (1998) 43-52.
- [4] Cohen, W. W., Basu, C., Hirsh, H.: *Recommendation as Classification: Using Social and Content-Based Information in Recommendation*. Proceedings of the AAAI-98. (1998) 714-720.
- [5] Cohn, D. A., Atlas, L., Lander, R.: *Improving generalization with active learning*. Machine Learning, 15:2. (1994). 201-221.
- [6] Cotter, P, & Smyth, B.: *PTV: Intelligent Personalised TV Guides*. Proceedings of the 12th Innovative Applications of Artificial Intelligence Conference. (2000). 957-964.
- [7] Hasenjager, M., Ritter, H.: *Active Learning with Local Models*. Neural Processing Letters, 7:2 (1998) 107-117.

- [8] Herlocker, J. L., Konstan, J. A., Borchers, A., Riedl, J.: An Algorithmic framework for performing collaborative filtering. Proceedings of the 1999 Conference on Research and Development in Information Retrieval. (1999).
- [9] Lewis, D. D., Catlett, J.: Heterogeneous uncertainty sampling for supervised learning. *Machine Learning*. (1994). 148-156.
- [10] Lindenbaum, M., Markovitch, S., Rusakov, D.: Selective Sampling for Nearest Neighbor Classifiers. *AAAI IAAI 99*. (1999) 366-371.
- [11] Perny, P., Zucker, J. D.: Preference-based Search and Machine Learning for Collaborative Filtering: the “Filme-Conseil” Movie Recommender System. *I3 Journal in Information Engineering Sciences*, 1:1. (2002).
- [12] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An Open Architecture fo Collaborative Filtering of Netnews. Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work. (1994) 175-186.
- [13] Wilson, D. R., Martinez T. R.: Reduction techniques for exemplar-based learning algorithms. *Machine Learning*, 38:3 (2000) 257-268.

Making Recommendations for Groups Using Collaborative Filtering and Fuzzy Majority

S rgio R. de M. Queiroz¹, Francisco de A. T. de Carvalho¹,
Geber L. Ramalho¹, and Vincent Corruble²

¹ Centro de Inform tica - Cin/UFPE - Cx. Postal 7851
CEP 50732-970, Recife, Brazil

{srmq, fatc, glr}@cin.ufpe.br

²Laboratoire d'Informatique de Paris VI - LIP6 □4
Place Jussieu, 75232, Paris, France
vincent.corruble@lip6.fr

Abstract. In recent years, recommender systems have achieved a great success. Popular sites like Amazon.com and CDNow give thousands of recommendations every day. However, although many activities are carried out in groups, like going to the theater with friends, these systems are focused on recommending items for individual users. This brings out the need of systems capable of performing recommendations for groups of people, a domain that has received little attention in the literature. In this article we introduce an investigation of automatic group recommendations, making connections with problems considered in social choice and psychology. Then we suggest a novel method of making recommendations for groups, based on existing techniques of collaborative filtering and classification of alternatives using fuzzy majority. Finally we experimentally evaluate the proposed method to see its behavior under groups of different sizes and degrees of homogeneity.

1 Introduction

The amount of information available is increasing at a speed that far exceeds our ability to manage it. People can choose from dozens of TV channels, thousands of movies, millions of books and Internet documents. Therefore, we are in a world of endless choice, how can we choose from a enormous universe of items with varying quality? When people have to make a choice without knowledge of the alternatives, a common approach is to rely on the recommendations of trusted individuals.

Computer *recommender systems* have appeared in the 1990s to automatize the process of recommendation. Today, popular sites like *Amazon.com* and *CDNow* have recommendation areas where, in addition to access expert information, users can rate and receive personalized suggestions about the items on sale.

One of the most successfully technologies used by these systems has been *collaborative filtering* [[1], [5], [10]]. Collaborative filtering works by building a database of item preferences for users. A user seeking a recommendation, say Lee, is matched against the database to find his *neighbors*, which are other users who have showed similar preferences to Lee in the past. Items that these neighbors have liked are suggested to Lee, as he will probably like them too.

These systems, however, are focused so far on recommending items for a sole user at a time. Nevertheless, many activities are carried out in groups (e.g. watching TV at home, going to the movies with friends, listening to the radio in a car during a family travel). Even some traditionally solitary activities (like web browsing) are sometimes performed in groups. WebTV have estimated that the average number of people who were watching during a browsing session with its service was two, indicating that in this case multi-user browsing was the norm rather than the exception [[7]]. Therefore, in these cases, a suggestion has to satisfy the group as a whole, not just a single member. This scenario brings out the need of recommender systems capable of making adequate recommendations for groups.

In this article we introduce an investigation of automatic group recommendations, making connections with problems considered in social choice and psychology. Then we suggest a novel method of making recommendations for groups, based on existing techniques of collaborative filtering and classification of alternatives using fuzzy majority. Finally we experimentally evaluate the proposed method to see its behavior under groups of different sizes and degrees of homogeneity.

2 The Problem

The problem of making collaborative filtering recommendations for groups is to suggest (new) items that the group will like, given that we have a set of historical preferences of the members of this group as well as preferences of other individual users (not in this group). This suggestion will ideally be the best possible for this group. Therefore, two central issues are raised: What is the best suggestion for a group? How to reach this suggestion?

Groups of people can have very different characteristics: they can be big or small; made of people with similar or antagonistic ideas. Consequently, when trying to answer these questions, it is also important to see how a recommendation technique behaves under these variables.

3 Related Work

The concept of making recommendations for groups has received little attention in the literature of recommender systems. Hill et al. [[5]] had as one of the design goals of their "virtual community" that recommendations and evaluations should be for sets of people not just individuals. Nevertheless, they did not delve into the difficulties involving the achievement of good recommendation for groups (i.e., the two fundamental questions cited in Section 2).

Let's Browse [[7]] is a collaborative web browsing agent that uses a content-based approach to recommend web pages to a group of people. A profile that consists of a list of weighted keywords is pre-built automatically for each user, employing a breadth-first search (with constrained depth) starting at the user's homepage. The group profile is a simple linear combination of each user's profile. Pages linked from the current visualized page are recommended if they match the group profile above a threshold. Therefore, *Let's Browse* is a content-based recommender system for groups, with a fixed recommendation strategy.

Although little about this topic has been studied in the literature of recommender systems, group decision-making has been an important research topic in the social sciences. Besides psychological and sociological approaches to how people make decisions as a collective (see e.g. [[8], [11]]), an interdisciplinary area called social choice theory (see e.g. [1]) has this study as one of its prime interests.

3.1 Results from Social Choice

In a seminal work concerning social choice [[1]], Arrow identified a set of reasonable characteristics that a function f (termed a social welfare function) that assigns a global preference relation from individual orderings should have: unrestricted domain, independence of irrelevant alternatives, Pareto condition and non-dictatorship. The function f has unrestricted domain if and only if (iff, for short) it is defined for every n -tuple of individual orderings (i.e. the entire Cartesian product set). It is independent of irrelevant alternatives iff the social preference relation over any two alternatives x and y depends on the individual preference relations regarding x and y only. The Pareto condition is satisfied iff $x P_i y$, for all individuals i , implies that $x R y$, where R means the weak ordering relation "is preferred or indifferent" and P_i denotes the asymmetric part of R (strict preference). Non-dictatorship, finally, is satisfied when there is no individual i such that $x R y$ iff $x R_i y$, for all x and y in X .

Arrow proved that it is impossible for any social welfare function to achieve all these characteristics simultaneously. Therefore *there is no ideal method* to aggregate individual preferences to achieve a group preference ordering.

3.2 Results from Social Psychology

Social Psychology, one area of Psychology, has been studying the problem of group decision making for years. One of its main concerns is to understand how do individual-level characteristics combine to create group-level products.

The Social Decision Scheme (SDS) model is a widely used approach in small group research to tackle this problem. It involves three central considerations: the distribution of the group members' preferences, the rule that combines these preferences (decision scheme), and the means of testing the adequacy of the decision schemes in predicting a sample of observed group decisions (model testing) [[6]].

- **The distribution of preferences:** the general SDS model assumes that each group member, and subsequently each group, selects one of n mutually exclusive and exhaustive alternatives. For a group having r individual members, their distribution among the n alternatives can be summarized by $(r_1, r_2, \dots, r_n)$, where r_j indicates the

number of group members who favor the j th alternative. Note that group members are indistinguishable but response alternatives are distinguishable in this expression. Extended versions of the SDS, like the SDS-Q [6], permit distinctions between individuals, as well as responses of non-discrete nature.

- **Decision schemes:** a social decision scheme is a rule or procedure that combines (usually in algebraic fashion) the various individual preferences (represented by the group distribution of preferences) into a single group decision. Decision schemes can be constructed to represent a variety of different social processes hypothesized to underlie group decision making.
- **Model testing:** an important concern is the comparison of the various plausible decision schemes through a model testing procedure. The results reached using the proposed decision scheme are compared to the observed (real) group responses. If the two results do not differ significantly, the proposed social decision scheme can be considered as a plausible description of the decision process used by the group.

Empirical experiments showed that the adequacy of a social decision scheme is dependent on the characteristics of the group members (e.g. willingness to argue, previous knowledge) and the type of problem in question. For example, it was observed a leniency bias in jury decision, which suggests that acquittal is easier to defend than conviction. On the other hand, in problem solving or collective recall, correct options frequently win with only one or two supporters in the group, particularly when correct members are confident in their choice [[11]]. A plethora of different social decision schemes have received empirical support under distinct circumstances. For a partial list of social decision schemes, see [[6]].

4 Using Fuzzy Majority to Obtain Group Recommendations

As we have seen in the previous section, there is no ideal method to achieve a group preference ordering from individual preferences. Moreover, psychological results show that the method people use to reach a group decision from individual preferences is complex, not easily predictable and subject to the influence of many variables. Therefore, a good method to make recommendations for groups should be flexible, easily parameterized (possibly by the users) and with human meaning (so that the user can understand what s/he is parameterizing). To achieve these goals, we used a two step process:

1. Predict the grades of the items for each group member using collaborative filtering. To predict the grade of item i to user a ($p_{a,i}$) we used Pearson correlation to weight user similarity, and computed a final prediction by performing a weighted average of deviations from the neighbor's mean:

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u=1}^n w_{a,u}}, \text{ where } w_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) \times (r_{u,i} - \bar{r}_u)}{\sigma_a \times \sigma_u}. \quad (1)$$

$\bar{r}_a$ is the average grade of user a , $r_{u,i}$ is the grade of user u for item i , n is the number of neighbors and $w_{a,u}$ is the similarity weight between the user and neighbor u as defined by the Pearson correlation coefficient. All computations are done only for items that both users evaluated.

This is the method used in GroupLens [[10]], except that we restricted the neighborhood size to the 40 best neighbors. According to experiments made in [[4]], the quality of the prediction decreases when larger neighborhoods are used.

2. According to the predicted grades, we derive a preference ordering of alternatives for each user. The set of these orderings is the input for the method of classifying alternatives based on fuzzy majority we use. The fuzzy majority provides a framework flexible and with "human consistency" to the choice process, because of the use of fuzzy linguistic quantifiers that represent the human discourse.

We used the classification method of alternatives proposed by Chiclana et al in [[2]]. We now give a brief description of the method. For a full explanation, see [[2]].

This procedure follows two steps before achieving a decision: aggregation and exploitation. The aggregation phase defines an outranking relation which indicates the global preference between every pair of alternatives, taking into consideration the different points of view. Exploitation transforms the global information about the alternatives into a global ranking of them, supplying a selection set of alternatives.

4.1 Aggregation: the Collective Fuzzy Preference Relation

For every individual preference ordering we derive a preference ordering relation P^k where p_{ij}^k reflects the pairwise preference ordering between the alternatives x_i and x_j for the individual e_k , $p_{ij}^k \in \{0, 1\}$. It takes the value 1 if x_i is preferred and 0 in the other case. Therefore, we have a set of individual binary preference relations: $\{P^1, \dots, P^m\}$. From this set of relations we derive the collective fuzzy preference relation, P^c . Each value, $p_{ij}^c \in [0, 1]$, represents the preference of alternative x_i over alternative x_j according to the majority individuals' opinions. Traditionally, the majority is defined as a threshold number of individuals. *Fuzzy majority* is a soft majority concept expressed by a fuzzy quantifier, which is manipulated via a fuzzy logic based calculus of linguistically quantified propositions (such as "most", "at least half", "as many as possible") [[2]]. Then the value of each p_{ij}^c is computed using an Ordered Weight Averaging (OWA) operator, as the aggregation operator of information. The OWA operator reflects the fuzzy majority calculating its weighting vector by means of a fuzzy quantifier. Therefore, the collective fuzzy preference relation is obtained as follows:

$$p_{ij}^c = \phi_Q(p_{ij}^1, \dots, p_{ij}^m), \quad (2)$$

where Q is the fuzzy quantifier used to compute the weights of the OWA operator, ϕ_Q .

4.2 Exploitation: Choosing the Alternatives

At this point, in order to select the alternatives "best" acceptable to the group of individuals as a whole, two quantifier guided choice degrees of alternatives based on

the concept of fuzzy majority are used: a dominance degree and a non dominance degree. Both are based on the use of the OWA operator.

4.2.1 Choice Degrees of Alternatives

Concretely, the two following quantifier guided choice degrees are used:

1. *Quantifier Guided Dominance Degree*: for the alternative, x_i , we compute the quantifier-guided dominance degree, $QGDD_i$, used to quantify the dominance that this alternative has over all the others in fuzzy majority sense as follows:

$$QGDD_i = \phi_Q(p_{ij}^c, j = 1, \dots, n, j \neq i). \quad (3)$$

2. *Quantifier Guided Non Dominance Degree*: the quantifier guided non dominance degree, $QGNDD_i$ is computed according to the following expression:

$$QGNDD_i = \phi_Q(1 - p_{ji}^s, j = 1, \dots, n, j \neq i), \text{ where } p_{ji}^s = \max\{p_{ji}^c - p_{ij}^c, 0\} \quad (4)$$

p_{ji}^s represents the degree to which x_i is strictly dominated by x_j . In this context, $QGNDD_i$ gives the degree in which each alternative is not dominated by a fuzzy majority of the remaining alternatives.

4.2.2 Selection Policy

We used the $QGDD$ to rank the alternatives (that is, the best alternative will be the one with the highest $QGDD$ whereas the worst will have the lowest $QGDD$). To break ties between alternatives, we used the $QGNDD$.

5 Experimental Evaluation

In this section we evaluate the influence of group size and degree of homogeneity on the results of the recommendation method using fuzzy majority. Different strategies for the fuzzy majority were used (linguistic quantifier applied in the aggregation and exploitation phases).

5.1 The EachMovie Dataset

The experiments were conducted using data from the EachMovie collaborative filtering service. The EachMovie service ran for 18 months (until September, 1997) as part of a project at *Compaq Systems Research Center*¹. In that period, 72,916 users provided 2,811,983 evaluations for 1,628 different movies. The grades were recorded in a 6-level numeric scale (0.0, 0.2, 0.4, 0.8, 1.0). The dataset is available for non-commercial use, and can be obtained from Compaq Computer Corporation [[3]].

Although 72,916 evaluations were available, the experiments were restricted to users that have evaluated at least 150 movies (2,551 users). This restriction was adopted to allow an intersection (of evaluated movies) of reasonable size between

¹ At that time, *Digital Equipment Corporation (DEC) Systems Research Center*.

each pair of users, so that more credit can be given to the comparisons that will decide the homogeneity degree of a group.

5.2 Data Preparation: the Formation of Groups

To conduct the experiments, it was necessary the existence of groups of users with varying sizes and homogeneity degrees. In the *EachMovie* dataset there isn't any notion of groups, so before running the experiments, it was needed to build them.

Four group sizes were defined: 3, 6, 12 and 24 individuals. We believe that this range of sizes includes the majority of scenarios where recommendation for groups can be used. For the degree of homogeneity factor, 3 levels were used: *high*, *medium* and *low* homogeneity. In our context, the groups don't need to be a *partition* of the set of users, i.e. the same user can be in more than one different group. The next sections describe the methodology used to build the groups.

5.2.1 Obtaining a Dissimilarity Matrix

The first step in the group definition was to build a dissimilarity matrix for the users. That is, a matrix m of size $n \times n$ (n is the number of users) where each m_{ij} contains the dissimilarity value between users i and j . To obtain this matrix, the dissimilarity of each user against all the others was calculated. As the dissimilarity is symmetric it is only necessary to calculate one diagonal (upper or lower).

The dissimilarities between users will be subsequently used to construct the groups with the three desired homogeneity degrees. To obtain the dissimilarity between two users, we follow these steps:

1. Calculate the Pearson correlation coefficient between the users. In our context, the correlation coefficient can be interpreted as a similarity measure, with values between -1 (smallest similarity) and +1 (largest similarity).
2. The correlation coefficient value is transformed into a dissimilarity value, that assumes values between 0 (smallest dissimilarity) and 1 (largest dissimilarity). This transformation is defined as: $\text{dissim}(i, j) = 1 - (w_{ij} + 1)/2$.

For the experiment, the movies from *EachMovie* were randomly separated in two sets of equal sizes. Only the user's evaluations which refer to elements of the first set (called *profile set*) were used to obtain the correlations between the users when calculating the dissimilarity matrix. The evaluations that refer to movies of the other set (called *test set*) were not used at this stage. The rationale behind this procedure is that the movies from the *test set* will be the ones that can be recommended for the groups. That is, it will be assumed that the members of the group did not know them previously. Therefore, it shall not be permitted that they affect the determination of the group's homogeneity degree. As we will see next, the group's homogeneity degree is dependent of the dissimilarity matrix between pair of users.

5.2.2 Group Formation

- **High homogeneity groups:** we wanted to obtain 100 groups with high homogeneity degree for each of the pre-defined sizes (3, 6, 12, 24). To achieve this, we first randomly generated 100 groups of 200 users each. Then the

hierarchical clustering algorithm *divisive analysis* (diana) was run, resulting in 100 agglomerative trees (the 100 groups of 200 random users were generated because it was too expensive to run the clustering algorithm in the whole set of users, moreover, it would be much more difficult to extract 100 groups from a single huge agglomerative tree than to extract one group from each tree). From each tree we extracted one group of each desired size. To extract a group of size n we used the elements of the "lowest" branch of the tree that had at least n elements. If the number of elements of this branch was larger than n we tested all combinations of size n and selected the one with lowest total dissimilarity (sum of all dissimilarities between the n users). However for groups of size 24, the number of combinations was too big. So instead of calculating them, we used a heuristic method, selecting the n users which have the lowest sum of dissimilarities in the branch (sum of dissimilarities between the user in question and all other users in the branch).

- **Low homogeneity groups:** to select a group of size n with low homogeneity from one of the groups with 200 users, we first calculated for each user its sum of dissimilarities (between this user and all the other 199). The n elements selected were the ones with the n largest sum of dissimilarities.
- **Medium homogeneity groups:** The dissimilarity between users can be seen as a random variable. We observed that it had a quite normal distribution. To select a group of size n with medium homogeneity degree, n elements were randomly selected from the total population of users. To avoid surprises due to randomness, after a group was generated, a test to compare a single mean (the one of the extracted group) to a specified value (the mean of the population) was done (using $\alpha = 0.05$) to see if the mean of the group didn't differ from the population mean.

5.3 Experimental Methodology

For each of the 1200 generated groups (4 sizes $\times$ 3 homogeneities $\times$ 100 repetitions) recommendations were generated using the fuzzy method. We used 4 fuzzy strategies: "As many as possible" + "As many as possible" (strategy 1); "As many as possible" + "At least half" (strategy 2); "At least half" + "Most" (strategy 3); and "Most" + "As many as possible" (strategy 4).

To evaluate the behavior of the fuzzy strategies with the various sizes and degrees of homogeneity of the groups, a metric is needed to represent this behavior. As we have a set of rankings as the input and a ranking as the output of the fuzzy process, a ranking correlation method was considered one appropriate method. We used the Kendall's rank correlation coefficient (τ). Given two ranks, this coefficient supplies a value between -1 (complete disagreement) and +1 (complete agreement). For each generated recommendation, we calculated τ between the final ranking generated for the group and the users' individual rankings (generated from the collaborative filtering process). Then we calculated the average τ , $\bar{\tau}$ for the recommendation.

The objective of the experiment was to evaluate how $\bar{\tau}$ is affected by the variation on the size and homogeneity degree of the groups. That is, verify if there were significant differences in $\bar{\tau}$ for the various sizes and degrees of homogeneity.

We did separated analyses of variance for each factor (*one-way anova*). To evaluate the influence of the homogeneity degree, we did the analyses of variance with the strategy and the size of the group fixed (and the degree of homogeneity as the factor, with 3 levels: high, medium and low). That way we had 16 analyses (4 strategies $\times$ 4 group sizes). To evaluate the influence of the group size in $\bar{r}$, we fixed the strategy and the degree of homogeneity (having this way the group size as the factor, with 4 levels: 3, 6, 12, 24). So in this case we did 12 analyses (4 strategies $\times$ 3 degrees of homogeneity).

6 Results and Discussion

Figure 1 shows the results of the analyses of variance. Fig. 1-a summarizes the results of the analyses when we have the homogeneity degree as the factor, and strategy and group size fixed. The solid line depicts the difference of means between every pair of factor levels, that is for each fuzzy strategy and group size we have 3 points, corresponding to the absolute value of the differences: [high homogeneity] [medium homogeneity] [high homogeneity] [low homogeneity] and [medium homogeneity] [low homogeneity], respectively (herein after the word [homogeneity] will be omitted when referring to these levels, for brevity). Fig. 1-b summarizes the results when we have the group size as the factor, and the strategy and homogeneity degree fixed. In this case we have 6 points for each fuzzy strategy and homogeneity degree, corresponding to the differences 03 [06, 03 [12, 03 [24, 06 [12, 06 [24 and 12 [24, respectively. In both figures, the dotted lines show the *least significant difference* (LSD) thresholds for 5% and 1% for the significance level [[9]]. That means that when the value of the difference is above both thresholds it is significant at 1% level, when it lies between the two it is significant at 5% and when it is below both threshold lines it is not significant at 5% of probability.

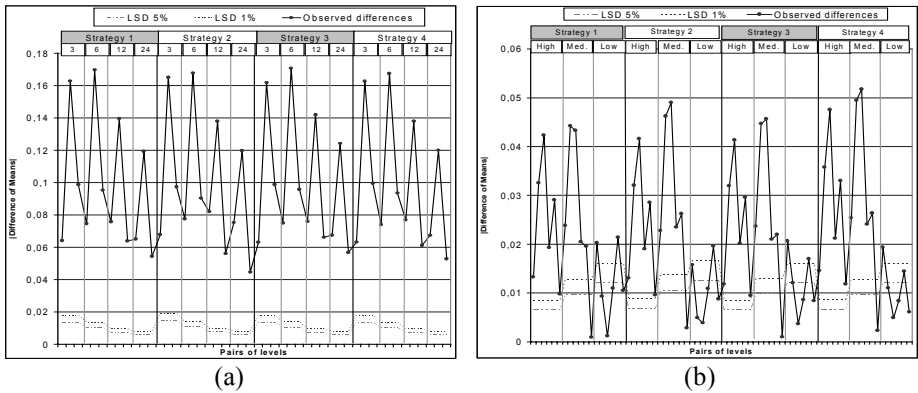


Fig. 1. Results of Analyses of Variance

The analyses of variance showed that the degree of homogeneity of the group was very important on the behavior of the recommendation strategies. In all strategies and

for all group sizes the difference of the $\bar{\tau}$ was extremely significant between the levels of homogeneity. The p -value was in all cases less than the smallest value detectable by the statistical software (2.2×10^{-16}). In all cases the averages of the levels differed significantly with a probability level $< 1\%$. Besides, we had: high average $>$ medium average $>$ low average, i.e. the compatibility degree between the group recommendation and the individual preferences was proportional to the group's homogeneity degree. All these facts were to be expected if the strategies were coherent, empirically indicating that the method maintains its plausibility.

The group size, however, didn't show a big importance on the behavior of the recommendation methods. The R^2 value of the analyses (which loosely indicates the fraction of the variability that is "explained" by the factor studied) was always small (always less than 0.4, in some cases as low as 0.02), showing that the model wasn't very adequate to explain the variation of $\bar{\tau}$. Many differences of means were not significant, highly homogeneous groups being the only case where all the differences were significant. This result indicates that this method could be used to recommend for groups of varying sizes, with similar performance (as measured by the $\bar{\tau}$).

Acknowledgements

This paper is supported by grants from the joint project Smart-Es (COFECUB-France and CAPES-Brazil) as well as by grants from CNPq-Brazil.

References

- [1] Arrow, K. J. *Social Choice and Individual Values*, Wiley, New York, 2nd edition, 1963.
- [2] Chiclana, F., Herrera, F., Herrera-Viedma, E., Poyatos, M. C. A classification method of alternatives for multiple preference ordering criteria based on fuzzy majority. *Journal of Fuzzy Mathematics*, 4(4):801-813, December 1996.
- [3] Compaq Systems Research Center. Eachmovie collaborative filtering data set. <http://www.research.compaq.com/SRC/eachmovie>, 2001.
- [4] Herlocker, J. L., Konstan, J. A., Borchers A., Riedl, J. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 230-237, 1999.
- [5] Hill, W., Stead, L., Rosenstein, M., Furnas, G. Recommending and evaluating choices in a virtual community of use. In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, vol. 1 of *Papers: Using the Information of Others*, 194-201, 1995.
- [6] Hinsz, V. B. Group decision making with responses of a quantitative nature: The theory of social decision schemes for quantities. *Organizational Behavior and Human Decision Processes*, 80(1):28-49, October 1999.
- [7] Lieberman, H., Dyke N. W. V., Vivacqua, A. S. Let's browse: A collaborative web browsing agent. In *Proceedings of the 1999 International Conference on*

- Intelligent User Interfaces*, Collaborative Filtering and Collaborative Interfaces, 65-68, 1999.
- [8] Mohammed, S., Ringseis, E. Cognitive diversity and consensus in group decision making: The role of inputs, processes, and outcomes. *Organizational Behavior and Human Decision Processes*, 85(2):310-335, July 2001.
 - [9] Montgomery, D. C. *Design and Analysis of Experiments*. Wiley, NY, 4th ed., 1997.
 - [10] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, 175-186, 1994.
 - [11] Stasser, G. A primer of social decision scheme theory: Models of group influence, competitive model-testing, and prospective modeling. *Organizational Behavior and Human Decision Processes*, 80(1):3-20, October 1999.

Mining Comprehensible Rules from Data with an Ant Colony Algorithm

Rafael S. Parpinelli¹, Heitor S. Lopes¹, and Alex A. Freitas²

¹ CEFET-PR, CPGEI

Av. Sete de Setembro, 3165, Curitiba - PR, 80230-901, Brazil
{rsparpin,hslopes}@cpgei.cefetpr.br

² PUC-PR, PPGIA-CCET

R. Imaculada Conceição, 1155, Curitiba - PR, 80215-901, Brazil
alex@ppgia.pucpr.br

Abstract. This work describes an algorithm for data mining called Ant-Miner (Ant Colony-based Data Miner). The goal of Ant-Miner is to extract classification rules from data. The algorithm is inspired by both research on the behavior of real ant colonies and some data mining concepts and principles. We compare the performance of Ant-Miner with CN2, a well-known data mining algorithm for classification, in six public domain data sets. The results provide evidence that: (a) Ant-Miner is competitive with CN2 with respect to predictive accuracy; and (b) The rule lists discovered by Ant-Miner are considerably simpler (smaller) than those discovered by CN2.

1 Introduction

In essence, the goal of data mining is to extract knowledge from data. We emphasize that in data mining – unlike for example in classical statistics – the goal is to discover knowledge that is not only accurate but also comprehensible for the user [8], [9], [10]. Comprehensibility is important whenever discovered knowledge will be used for supporting a decision made by a human user. After all, if discovered knowledge is not comprehensible for the user, he/she will not be able to interpret and validate it. In this case, probably the user will not trust enough the discovered knowledge to use it for decision making. This can lead to wrong decisions.

In this paper we describe an Ant Colony-based data mining algorithm for the classification task of data mining. In this task the goal is to assign each case (object, record, or instance) to one class, out of a set of predefined classes, based on the values of some attributes (called predictor attributes) for the case. For the classification task, the discovered knowledge is often represented in the form of *IF <conditions> THEN <class>* rules (which are further discussed in Section 3).

To the best of our knowledge the use of Ant Colony Optimization (ACO) algorithms [5] for discovering classification rules, in the context of data mining, is a research area still unexplored. Actually, the only Ant Colony-based algorithm

developed for data mining that we are aware of is an algorithm for clustering [13], which is, of course, a data mining task very different from the classification task addressed in this paper.

We believe the development of ACO algorithms for data mining is a promising research area, due to the following reason. ACO algorithms involve simple agents (ants) that cooperate with one another to achieve an emergent, unified behavior for the system as a whole, producing a robust system capable of finding high-quality solutions for problems with a large search space. In the context of rule discovery, an ACO algorithm has the ability to perform a flexible, robust search for a good combination of terms (logical conditions) involving values of the predictor attributes.

2 Ant Colony Optimization

An Ant Colony Optimization algorithm (ACO) is essentially a system based on agents which simulate the natural behavior of ants, including mechanisms of cooperation and adaptation. This new heuristic was proposed in [6] in order to solve combinatorial optimization problems and it has been shown to be both robust and versatile – in the sense that it has been successfully applied to a range of different combinatorial optimization problems [7]. In passing we mention that recently there has been a growing interest in developing rule-discovery algorithms based on other kinds of bio-inspired algorithms – mainly evolutionary algorithms [10]. ACO algorithms are based on the following ideas:

- Each path followed by an ant is associated with a candidate solution for a given problem;
- When an ant follows a path, the amount of pheromone (a chemical substance used in real ant colonies) deposited on that path is proportional to the quality of the corresponding candidate solution for the target problem;
- When an ant has to choose between two or more paths, the path(s) with a larger amount of pheromone (i.e., the path(s) that were more frequently chosen by other ants in the past) have a greater probability of being chosen by the ant.

As a result, the ants eventually converge to a short path, hopefully the optimum or a near-optimum solution for the target problem.

In essence, the design of an ACO algorithm involves the specification of [1]:

- An appropriate representation of the problem, which allows the ants to incrementally construct/modify solutions through the use of a probabilistic transition rule, based on the amount of pheromone in the trail and on a local, problem-dependent heuristic;
- A method to enforce the construction of valid solutions, that is, solutions that are legal in the real-world situation corresponding to the problem definition;
- A problem-dependent heuristic function (η) that measures the quality of items that can be added to the current partial solution;
- A rule for pheromone updating, which specifies how to modify the pheromone trail (τ);

- A probabilistic transition rule based on the value of the heuristic function (η) and on the pheromone trail (τ) that is used to iteratively construct a solution.

3 Ant-Miner: A New ACO Algorithm for Data Mining

In an ACO algorithm each ant incrementally constructs/modifies a solution for the target problem. In our case the target problem is to discover classification rules. As mentioned in the introduction, each classification rule has the form: *IF* $\langle term1 \text{ AND } term2 \text{ AND } \dots \rangle$ *THEN* $\langle class \rangle$. Each term is a triple $\langle attribute, operator, value \rangle$, where *value* is a value belonging to the domain of *attribute*. The operator element in the triple is a relational operator. The current version of Ant-Miner copes only with categorical attributes, so that the operator element in the triple is always “=”. Continuous (real-valued) attributes are discretized in a preprocessing step.

A high-level description of Ant-Miner is shown in Algorithm I.

```

TrainingSet = {all training cases};
DiscoveredRuleList = []; /* rule list is initialized with an empty list */
WHILE (TrainingSet > Max_Uncovered_Cases)
    i = 1; /* ant index */
    j = 1; /* convergence test index */
    Initialize all trails with the same amount of pheromone;
    REPEAT
        Anti starts with an empty rule and incrementally constructs a
        classification rule Ri, by adding one term at a time to the current rule;
        Prune rule Ri;
        Update the pheromone of all trails, by increasing pheromone in the trail
        followed by Anti (proportional to the quality of Ri) and decreasing
        pheromone in the other trails (simulating pheromone evaporation);
        IF (Ri is equal to Ri-1) /* update convergence test */
            THEN j = j + 1;
            ELSE j = 1;
        END IF
        i = i + 1;
    UNTIL (i ≥ No_of_Ants) OR (j ≥ No_Rules_Converg)
    Choose the best rule Rbest among all rules Ri constructed by all the ants;
    Add rule Rbest to DiscoveredRuleList;
    TrainingSet = TrainingSet - {set of cases correctly covered by Rbest};
END WHILE

```

Algorithm 1. A High-Level Description of Ant-Miner

Ant-Miner follows a sequential covering approach to discover a list of classification rules covering all or almost all training cases. At first, the list of discovered rules is empty and the training set consists of all training cases. Each iteration of the WHILE loop of Algorithm I, corresponding to a number of executions of the REPEAT-UNTIL loop, discovers one classification rule. This rule is added to

the list of discovered rules, and the training cases that are correctly covered by this rule (i.e., cases satisfying the rule antecedent and having the class predicted by the rule consequent) are removed from the training set. This process is iteratively performed while the number of uncovered training cases is greater than a user-specified threshold, called *Max_Uncovered_Cases*. Each iteration of the REPEAT-UNTIL loop of Algorithm I consists of three steps, comprising rule construction, rule pruning, and pheromone updating, discussed in the following.

3.1 Rule Construction

For the rule construction step, the Ant_i starts with an empty rule, that is, a rule with no term in its antecedent, and adds one term at a time to its current partial rule. The current partial rule constructed by an ant corresponds to the current partial path followed by that ant. Similarly, the choice of a term to be added to the current partial rule corresponds to the choice of the direction in which the current path will be extended. The choice of the term to be added to the current partial rule depends on both a problem-dependent heuristic function (η) and on the amount of pheromone (τ) associated with each term. Ant_i keeps adding one-term-at-a-time to its current partial rule until one of the following two stopping criteria is met:

- Any term to be added to the rule would make the rule cover a number of cases smaller than a user-specified threshold, called *Min_cases_per_rule* (minimum number of cases covered per rule). This enforces at least a certain degree of generality in the discovered rules, helping to avoid an overfitting to the training data; or
- All attributes have already been used by the ant, so that there is no more attributes to be added to the rule antecedent. Notice that each attribute can occur only once in each rule, to avoid invalid rules such as “IF (Sex = male) AND (Sex = female)....”.

This process is repeated until one of the two following conditions is met:

- The number of constructed rules is equal to or greater than the user-specified threshold *No_of_Ants*;
- The current Ant_i has constructed a rule that is exactly the same as the rule constructed by the previous *No_Rules_Converg* - 1 ants, where *No_Rules_Converg* stands for the number of rules used to test convergence of the ants, this is, whether the ants converged to a single rule (path).

Let $term_{ij}$ be a rule condition of the form $A_i = V_{ij}$, where A_i is the i -th attribute and V_{ij} is the j -th value of the domain of A_i . The probability that $term_{ij}$ is chosen to be added to the current partial rule is given by Equation (1):

$$P_{ij} = \frac{\eta_{ij} \cdot \tau_{ij}(t)}{\sum_{i=1}^a x_i \cdot \sum_{j=1}^{b_i} (\eta_{ij} \cdot \tau_{ij}(t))} \quad (1)$$

where:

- η_{ij} is the value of a problem-dependent heuristic function for $term_{ij}$;
- $\tau_{ij}(t)$ is the amount of pheromone associated with $term_{ij}$ at time t , corresponding to the amount of pheromone currently available in the position i,j of the path being followed by the current ant;
- a is the total number of attributes;
- x_i is set to 1 if the attribute A_i was not yet used by the current ant, or to 0 otherwise;
- b_i is the number of values in the domain of the i -th attribute.

Once the rule antecedent is completed, the system chooses the rule consequent (i.e., the predicted class) that maximizes the quality of the rule. This is done by assigning to the rule consequent the majority class among the cases covered by the rule.

3.2 Heuristic Function

For each $term_{ij}$ that can be added to the current rule, Ant-Miner computes the value η_{ij} of a heuristic function that is an estimate of the quality of this term, with respect to its ability to improve the predictive accuracy of the rule. This heuristic function is based on Information Theory [4]. More precisely, the value of η_{ij} for $term_{ij}$ involves a measure of the entropy (or amount of information) associated with that term. For each $term_{ij}$ its entropy is

$$H(W|A_i = V_{ij}) = - \sum_{w=1}^k (P(w|A_i = V_{ij}) \cdot \log_2 P(w|A_i = V_{ij})) \quad (2)$$

where:

- W is the class attribute (i.e., the attribute whose domain consists of the classes to be predicted);
- k is the number of classes;
- $P(w|A_i=V_{ij})$ is the empirical probability of observing class w conditional on having observed $A_i=V_{ij}$.

The higher the value of $H(W|A_i=V_{ij})$, the more uniformly distributed the classes are, and so the smaller the probability of the current ant choosing $term_{ij}$ to be added to its partial rule. It is desirable to normalize the value of the heuristic function to facilitate its use in Equation (1), combining both this function and the amount of pheromone. In order to implement this normalization, it is used the fact that the value of $H(W|A_i=V_{ij})$ varies in the range $0 \leq H(W|A_i=V_{ij}) \leq \log_2 k$, where k is the number of classes. Therefore, the proposed normalized, information-theoretic heuristic function is:

$$\eta_{ij} = \frac{\log_2 k - H(W|A_i = V_{ij})}{\sum_{i=1}^a x_i \cdot \sum_{j=1}^{b_i} (\log_2 k - H(W|A_i = V_{ij}))} \quad (3)$$

where:

- a is the total number of attributes;
- x_i is set to 1 if the attribute A_i was not yet used by the current ant or to 0 otherwise;
- b_i is the number of values in the domain of the i -th attribute.

Hence, the higher the value of η_{ij} , the more relevant for classification the $term_{ij}$ is, and so the higher its probability of being chosen.

In the above heuristic function there are just two minor caveats. First, if the value V_{ij} of attribute A_i does not occur in the training set then $H(W|A_i=V_{ij})$ is set to its maximum value of $\log_2 k$. This corresponds to assigning to $term_{ij}$ the lowest possible predictive power. Second, if all the cases belong to the same class then $H(W|A_i=V_{ij})$ is set to 0. This corresponds to assigning to $term_{ij}$ the highest possible predictive power.

3.3 Rule Pruning

Rule pruning is a commonplace technique in data mining [2]. The main goal of rule pruning is to remove irrelevant terms that might have been unduly included in the rule. Rule pruning potentially increases the predictive power of the rule, helping to avoid its overfitting to the training data. Another motivation for rule pruning is that it improves the simplicity of the rule, since a shorter rule is usually easier to be understood by the user than a longer one.

As soon as the current ant completes the construction of its rule, the rule pruning procedure is called. The basic idea is to iteratively remove one-term-at-a-time from the rule while this process improves the quality of the rule. More precisely, in the first iteration one starts with the full rule. Then it is tentatively tried to remove each of the terms of the rule – each one in turn – and the quality of the resulting rule is computed using a given rule-quality function (defined by Equation (5)). It should be noted that this step might involve replacing the class in the rule consequent, since the majority class in the cases covered by the pruned rule can be different from the majority class in the cases covered by the original rule. The term whose removal most improves the quality of the rule is effectively removed from it, completing the first iteration. In the next iteration it is removed again the term whose removal most improves the quality of the rule, and so on. This process is repeated until the rule has just one term or until there is no term whose removal will improve the quality of the rule.

3.4 Pheromone Updating

At each iteration of the WHILE loop of Algorithm I all $term_{ij}$, $\forall i, j$, are initialized with the same amount of pheromone, so that when the first ant starts its search, all paths have the same amount of pheromone. The initial amount of pheromone deposited at each path position is inversely proportional to the number of values of all attributes, and is defined by Equation (4):

$$\tau_{ij}(t = 0) = \frac{1}{\sum_{i=1}^a b_i} \quad (4)$$

where:

- a is the total number of attributes,
- b_i is the number of possible values that can be taken on by attribute A_i .

The value returned by this equation is normalized to facilitate its use in Equation (1). The amount of pheromone associated with each $term_{ij}$ occurring in the rule found by the ant (after pruning) is increased in proportion to the quality – Q – of that rule. and is computed by the formula: $Q = sensitivity \times specificity$ [12], defined as:

$$Q = \frac{TP}{TP + FN} \cdot \frac{TN}{FP + TN} \quad (5)$$

where: TP (true positives) is the number of cases covered by the rule that have the class predicted by the rule; FP (false positives) is the number of cases covered by the rule that have a class different from the class predicted by the rule; FN (false negatives) is the number of cases that are not covered by the rule but that have the class predicted by the rule; and TN (true negatives) is the number of cases that are not covered by the rule and that do not have the class predicted by the rule.

Q 's value is within the range $0 \leq Q \leq 1$ and, the larger the value of Q , the higher the quality of the rule. Pheromone updating for a $term_{ij}$ is performed according to Equation (6), for all terms $term_{ij}$ that occur in the rule.

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \tau_{ij}(t) \cdot Q, \forall i, j \in R \quad (6)$$

where R is the set of terms occurring in the rule constructed by the ant at time t .

In Ant-Miner, the pheromone evaporation is implemented in a somewhat indirect way. More precisely, the effect of pheromone evaporation for unused terms is achieved by normalizing the value of each pheromone τ_{ij} . This normalization is performed by dividing the value of each τ_{ij} by the summation of all τ_{ij} , $\forall i, j$.

In order to classify a new test case, unseen during training, the discovered rules are applied in the order they were discovered (recall that discovered rules are kept in an ordered list). The first rule that covers the new case is applied – this is, the case is assigned the class predicted by that rule's consequent. It is possible that no rule of the list covers the new case. In this situation the new case is classified by a default rule that simply predicts the majority class in the set of uncovered training cases, that is, the set of cases that are not covered by any discovered rule.

4 Computational Results and Discussion

4.1 Data Sets, Discretization Method and Parameters Values Used in the Experiments

The performance of Ant-Miner was evaluated using six public-domain data sets from the UCI (University of California at Irvine) repository¹.

¹ Available in the Internet in the address: <http://www.ics.uci.edu/~mllearn/MLRepository.html>.

The main characteristics of the data sets used in our experiment are summarized in Table 1. The first column of this table identifies the data set, and the other columns indicate, respectively, the number of cases, number of categorical attributes, number of continuous attributes, and number of classes of the data set.

Table 1. Data Sets Used in the Experiments

Data Set	#cases	#categ. attrib.	#contin. attrib.	#classes
Ljubljana breast cancer	282	9	-	2
Wisconsin breast cancer	683	-	9	2
tic-tac-toe	958	9	-	2
dermatology	366	33	1	6
hepatitis	155	13	6	2
Cleveland heart disease	303	8	5	5

As mentioned earlier, Ant-Miner discovers rules referring only to categorical attributes. Therefore, continuous attributes have to be discretized in a preprocessing step. This discretization was performed by the C4.5-Disc discretization method [11].

In all the experiments Ant-Miner’s parameters values were set to: *No_of_ants* = 3000; *Min_cases_per_rule* = 10; *Max_uncovered_cases* = 10 and *No_Rules_Converg* = 10.

We have made no serious attempt to optimize the setting of these parameters; however, they have produced quite good results, as will be shown later. In the same way, CN2’s parameters were not optimized as well, in order to make the comparison between the two algorithms more fair.

4.2 Comparing Ant-Miner with CN2

We have evaluated the performance of Ant-Miner by comparing it with CN2 [3], a well-known classification-rule discovery algorithm. In essence, CN2 searches for a rule list in an incremental fashion. It discovers one rule at a time. Each time it discovers a rule it adds that rule to the end of the list of discovered rules, removes the cases covered by that rule from the training set and calls again the procedure to discover another rule for the remaining training cases. Notice that this strategy is also used by Ant-Miner. In addition, both Ant-Miner and CN2 construct a rule by starting with an empty rule and incrementally add one term at a time to the rule.

However, the rule construction procedure is very different in the two algorithms. CN2 uses a beam search to construct a rule.

In CN2 there is no mechanism to allow the quality of a discovered rule to be used as a feedback for constructing other rules. This feedback (using the mechanism of pheromone) is the major characteristic of ACO algorithms, and can be considered the main difference between Ant-Miner and CN2. In addition, Ant-Miner performs a stochastic search, whereas CN2 performs a deterministic search. In data mining and machine learning terminology, one can say that both algorithms have the same representation bias (since they both discover an ordered rule list), but different search (or preference) biases.

The comparison was carried out across two criteria, namely the predictive accuracy of the discovered rule lists and their simplicity. Predictive accuracy was measured by a well-known 10-fold cross-validation procedure [14].

All the results were obtained using a Pentium II PC with clock rate of 333 MHz and 128 MB of main memory. Ant-Miner was developed in C language and it took about the same processing time as CN2 (on the order of seconds for each data set) to obtain the results. The results comparing the accuracy rate of Ant-Miner and CN2 are reported in Table 2. The numbers right after the “ $\pm$ ” symbol are the standard deviations of the corresponding accuracy rates.

Table 2. Accuracy Rate of Ant-Miner vs CN2

Data Set	Ant-Miner's Accuracy rate (%)	CN2's Accuracy rate (%)
Ljubljana breast cancer	75.28 $\pm$ 2.24	67.69 $\pm$ 3.59
Wisconsin breast cancer	96.04 $\pm$ 0.93	94.88 $\pm$ 0.88
tic-tac-toe	73.04 $\pm$ 2.53	97.38 $\pm$ 0.52
dermatology	94.29 $\pm$ 1.20	90.38 $\pm$ 1.66
hepatitis	90.00 $\pm$ 3.11	90.00 $\pm$ 2.50
Cleveland heart disease	59.67 $\pm$ 2.50	57.48 $\pm$ 1.78

Concerning classification accuracy, Ant-Miner obtained results somewhat better than CN2 in four of the six data sets, whereas CN2 obtained a result much better than Ant-Miner in the Tic-tac-toe data set. In one data set both algorithms obtained the same accuracy rate. Therefore, overall one can say that the two algorithms are roughly competitive in terms of accuracy rate, even though the superiority of CN2 in the tic-tac-toe is more significant than the superiority of Ant-Miner in four data sets.

We now turn to the results concerning the simplicity of the discovered rule list. This simplicity was measured, as usual in the literature, by the number of discovered rules and the average number of terms (conditions) per rule. The results comparing the simplicity of the rule lists discovered by Ant-Miner and by CN2 are reported in Table 3.

Table 3. Simplicity of Rule Lists Discovered by Ant-Miner vs CN2

Data Set	No. of rules		No. of terms / No. of rules	
	Ant-Miner	CN2	Ant-Miner	CN2
Ljubljana breast cancer	7.10 $\pm$ 0.31	55.40 $\pm$ 2.07	1.28	2.21
Wisconsin breast cancer	6.20 $\pm$ 0.25	18.60 $\pm$ 0.45	1.97	2.39
Tic-tac-toe	8.50 $\pm$ 0.62	39.70 $\pm$ 2.52	1.18	2.90
Dermatology	7.30 $\pm$ 0.15	18.50 $\pm$ 0.47	3.16	2.47
Hepatitis	3.40 $\pm$ 0.16	7.20 $\pm$ 0.25	2.41	1.58
Cleveland heart disease	9.50 $\pm$ 0.92	42.40 $\pm$ 0.71	1.71	2.79

Concerning the simplicity of discovered rules, overall Ant-Miner discovered rule lists much simpler (smaller) than the rule lists discovered by CN2.

This seems a good trade-off, since in many data mining applications the simplicity of a rule list/set tends to be even more important than its accuracy rate.

5 Conclusions and Future Work

We have compared the performances of Ant-Miner and the well-known CN2 algorithm in six public domain data sets. The results showed that, concerning predictive accuracy, Ant-Miner obtained somewhat better results in four data sets, whereas CN2 obtained a considerably better result in one data set. In the other data set both algorithms obtained the same predictive accuracy. Therefore, overall one can say that Ant-Miner is roughly competitive with CN2 with respect to predictive accuracy.

On the other hand, Ant-Miner has consistently found much simpler (smaller) rule lists than CN2. Therefore, Ant-Miner seems particularly advantageous when it is important to minimize the number of discovered rules and rule terms (conditions), in order to improve the comprehensibility of the discovered knowledge. It can be argued that this point is important in many (probably most) data mining applications, where discovered knowledge will be shown to a human user as a support for intelligent decision making, as discussed in the introduction.

Two important directions for future research are as follows. First, it would be interesting to extend Ant-Miner to cope with continuous attributes as well, rather than requiring that this kind of attribute be discretized in a preprocessing step. Second, it would be interesting to investigate the performance of other kinds of heuristic function and pheromone updating strategy.

References

- [1] E. Bonabeau, M. Dorigo and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. New York, NJ: Oxford University Press, 1999.
- [2] L. A. Brewlow and D. W. Aha, "Simplifying decision trees: a survey," *The Knowledge Engineering Review*, vol. 12, no. 1, pp. 1-40, 1997.
- [3] P. Clark and T. Niblett, "The CN2 induction algorithm," *Machine Learning*, vol. 3, pp. 261-283, 1989.
- [4] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, New York: John Wiley & Sons, 1991.
- [5] M. Dorigo, A. Colormi and V. Maniezzo, "The ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 26, no. 1, pp. 1-13, 1996.
- [6] M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic," In: *New Ideas in Optimization*, D. Corne, M. Dorigo and F. Glover Eds. London, UK: McGraw Hill, pp. 11-32, 1999.
- [7] M. Dorigo, G. Di Caro and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial Life*, vol. 5, no. 2, pp. 137-172, 1999.

- [8] U. M. Fayyad, G. Piatetsky-Shapiro and P. Smyth, "From data mining to knowledge discovery: an overview," In: *Advances in Knowledge Discovery & Data Mining*, U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy Eds. Cambridge: AAAI/MIT, pp. 1-34, 1996.
- [9] A. Freitas and S. H. Lavington, *Mining Very Large Databases with Parallel Processing*, London: Kluwer, 1998.
- [10] A. Freitas, *Data Mining and Knowledge Discovery with Evolutionary Algorithms*, (Forthcoming book) Heidelberg: Springer-Verlag, 2002.
- [11] R. Kohavi and M. Sahami, "Error-based and entropy-based discretization of continuous features," In: *Proceedings of the 2nd International Conference Knowledge Discovery and Data Mining*. Menlo Park, CA: AAAI Press, pp. 114-119, 1996.
- [12] H. S. Lopes, M. S. Coutinho and W. C. Lima, "An evolutionary approach to simulate cognitive feedback learning in medical domain," In: *Genetic Algorithms and Fuzzy Logic Systems: Soft Computing Perspectives*, E. Sanchez, T. Shibata and L.A. Zadeh Eds. Singapore: World Scientific, pp. 193-207, 1998.
- [13] N. Monmarche, "On data clustering with artificial ants," In: *Data Mining with Evolutionary Algorithms*, Research Directions – Papers from the AAAI Workshop, Technical Report WS-99-06, A.A. Freitas Ed. Menlo Park: AAAI Press, pp. 23-26, 1999.
- [14] S. M. Weiss and C. A. Kulikowski, *Computer Systems That Learn*, San Francisco, CA: Morgan Kaufmann, 1991.

Learning in Fuzzy Boolean Networks – Rule Distinguishing Power

José A.B. Tomé

INESC,
Rua Alves Redol nº9, 1000 Lisboa Portugal
jose.tome@inesc.pt

Abstract. Fuzzy Boolean Networks are Boolean networks with nature like characteristics, such as organization of neurons on cards or areas, random individual connections, structured meshes of links between cards. They also share with natural systems some interesting properties: relative noise immunity, capability of approximate reasoning and learning from sets of experiments. An overview of the processes involved in reasoning supported on an hardware architecture are presented, as well as how Hebbian-Grossberg learning can be achieved. An interesting problem related with these nets is the number of different rules that they are able to capture from experiments without cross interferences, that is, their rule capacity. This work establishes a lower bound for this number, proving that it depends on the number of inputs per consequent neurons and its relation to consequent granularity. An application for traffic problems is also provided.

1 Introduction

The known capabilities of Fuzzy Systems to explain system behavior through qualitative rules may strongly benefit the global performance of neural networks, adding new capabilities to their classical possibilities of experimental learning variable relations. This can be achieved by fuzzification of the neural net components [2,3,6,7,8]. Another form of cooperation between the two paradigms is to build a Fuzzy System with components that are neural nets. To the usual inference and explanation properties of the Fuzzy System are thus added the learning capabilities of neural networks [5,7,11]. This synergy needs not to be achieved through this kind of “adding components” from the two paradigms. Instead they may be embedded together in a common structure, as is the case of author’s work presented elsewhere [9, 10]. In that work it is presented a Boolean neural network where variable (or concept) values are represented by the activation level of neurons on associated neural areas. These networks present other interesting similarities with natural neural systems, including an intrinsic immunity to noise in individual neurons or connections (since the density of neural activation is not disturbed by individual errors), fire/do not fire individual neuron operations, random

connections between neurons and structured macro connections between neural areas or cards. An emergent property of such networks is their fuzzy reasoning capabilities (qualitative rules implementation), despite none fuzzy concept was placed at the micro (neural) level. Moreover, the model is also an Universal Approximator [10]. These structures are also capable to automatically adapt to the granularity of the input variables and to decide about the relevance of input variables to a given problem. Concerning their learning capabilities, the Fuzzy Boolean Nets can learn from sets of experiments in a non-supervised way, using elements of binary memory embedded on the neuron internal structure.

An interesting question related with such networks is their capacity in memorizing different rules (fuzzy rules) , or the achievable granularity of antecedent and consequent variables. In this work it is shown how such questions may be answered and a lower bound for the consequent granularity (and thus for the rule capacity) is established.

2 Architecture

The complete description of the network architecture and the deduction of its reasoning and learning characteristics may be found in [9], but a brief summary follows.

Neurons are aggregated in areas (one area per variable) and connections are established between the outputs of antecedent neurons (those on antecedent areas) to the inputs of the neurons on the consequent area. The model here considered postulates that each consequent neuron is an $N.m$ input neuron, where N is the number of antecedents and m the number of inputs coming from the same antecedent area. Each input I_{kn} ($k=1,N; n=1,m$) is connected to a randomly chosen neuron output from antecedent area k . As an interpretation, one can say that each consequent neuron "observes" each of the antecedent areas through a sample of m binary values. Moreover, each of these samples is taken as a simple count on the number of activated inputs, since there is no reason to differentiate between two different samples with the same number of activated binary variables (the same number of "ones").

Consider d_{ij} the detection of i_j , that is the Boolean function which takes value "1" if and only if there are i activated inputs coming from antecedent j . Similarly, $d(i_1, \dots, i_N)$ is considered the joint detection of i_1 activated inputs from antecedent 1,..., and i_N activated inputs from antecedent N . Each single neuron is designed in order to implement the following Boolean function:

$$\bigvee_{i_1=0}^m \dots \bigvee_{i_N=0}^m d(i_1, \dots, i_N) \text{ AND } ff(i_1, \dots, i_N), \quad \text{with} \quad d(i_1, \dots, i_N) = \bigwedge_{j=1}^N d_{ij}$$

The term $ff(i_1, \dots, i_N)$ represents the memory of the neuron regarding that particular input count configuration and it will be established during a training phase. Each one of these terms (ff) is associated with a flip-flop in a possible hardware neuron implementation. A possible hardware implementation is given in fig. 1.

Define **activation ratio** of any area as the ratio of activated neurons and the total number of neurons of that area. This is the same as the probability of a randomly chosen neuron in that area to be activated. Let p_j represent the activation ratio of antecedent area j and $pr(i_1, \dots, i_N)$ the probability of $ff(i_1, \dots, i_N)$ to be activated. Then, since for a randomly chosen neuron, $\mathbf{V}$, one and only one of the $d(i_1, \dots, i_N)$ is activated, it follows that the activation ratio of the consequent area of $\mathbf{V}$ becomes:

$$\sum_{i_1=0}^m \dots \sum_{i_N=0}^m \prod_{j=1}^N \binom{m}{k_{ij}} p_j^{k_{ij}} \cdot (1-p_j)^{m-k_{ij}} \cdot pr(i_1, \dots, i_N)$$

or simply:

$$\sum_{i_1=0}^m \dots \sum_{i_N=0}^m P^a r(i_1, \dots, i_N) \cdot pr(i_1, \dots, i_N) \quad (1)$$

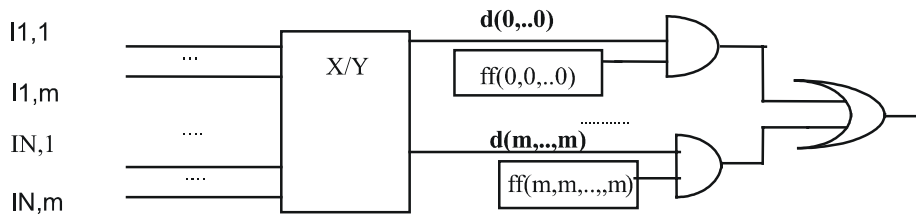


Fig. 1. Neuron architecture

Using the algebraic product and the bounded sum for t-norm and t-conorm respectively, it follows that microscopic neural operations as defined above emerge, at the macroscopic or network level, as fuzzy qualitative reasoning. To this purpose the equations above may be interpreted as follows:

Input variables, the activation ratios p_j , are fuzzified through binomial membership functions of the form $\binom{m}{k_{ij}} p_i^{k_{ij}} \cdot (1-p_i)^{m-k_{ij}}$. The evaluation of the expression for a given p_j represents the membership degree of p_j in that fuzzy set.

The product of the terms, the $\prod_{j=1}^N \binom{m}{k_{ij}} p_j^{k_{ij}} \cdot (1-p_j)^{m-k_{ij}} \cdot pr(i_1, \dots, i_N)$, repre-

sents the fuzzy intersection of the antecedents ($i=1, N$), by definition of the above t-norm. Considering the consequent fuzzy sets as singletons (amplitude "1") at the consequent UD values $pr(i_1, \dots, i_N)$, it follows that the equations represent the defuzzification by the Center of Area method.

One may conclude that the network implements a set of production rules of the type:

IF A is A1 AND B is B1 AND $\square$ $\square$. THEN C is C1

where A and B are Antecedent variables, C is the Consequent variable, $A_1, B_1, \dots$ are linguistic terms of fuzzy sets defined by the count samples on m inputs and C_1 is a fuzzy set (singleton) at the consequent defined by the probability of the flip-flop $ff(I_1)$ to be at $\square$. This probability is set during the learning process, being I_1 the N element vector of the above counts.

3 Learning

It is the setting of logical values at the neuron flip flops that establishes the learning phase of the network. Macroscopically, this turns to be the setting of the $pr(k_1, \dots, k_N)$ probabilities in the above expression, (1), of the internal flip-flops. During this learning phase the network is activated (both in antecedent and consequent areas) by a collection of experiments and for each experiment a particular input configuration is presented to each consequent neuron. This configuration addresses one and only one internal flip-flop of each neuron. Updating of each flip-flop value depends on its selection (or not) and on the logic value of the consequent neurone. This may be considered an Hebbian type of learning [4] if pre and post-synaptic activities are, in the present model, given by the activation ratios: p_j for antecedent area j and p_{out} for the consequent area. For each neuron, the $m+1$ different counts are the meaningful parameters to take into account for pre synaptic activity of one antecedent. Thus, in a given experiment, the correlation between posterior synapse activity (p_{out}) and pre synaptic activity -the probability of a given $d(i_1, \dots, i_N)$ to be activated- can be represented by the probability of the different flip-flops to be activated. In practical terms, for each teaching experiment and for each consequent neuron, the state of flip-flop $ff(i_1, \dots, i_N)$ is determined by, and only by, the Boolean values of decoder output $d(i_1, \dots, i_N)$ and of the output neuron state considered.

Considering then the $pr(k_1, \dots, k_N)$, in expression (1), as the synaptic strengths, one may have different learning types, depending on how they are updated (for simplicity the indexes are omitted in what follows). Here, one is considering the interesting case when non-selected flip-flops maintain their state and selected flip-flops take the value of consequent neuron, which corresponds to a kind of Grossberg based learning. It corresponds to the following updating equation (where indexes are not represented and p is used in place of pr, for simplicity), and where P^a is the probability of activating, in the experiment, the decoder output associated with p:

$$p(t+1) - p(t) = P^a \cdot (p_{out} - p(t)) \quad (2)$$

First, it is quite easy to see that the network converges to the taught rule, if every experiment teaches the same rule, say P_{out} as the consequent activation ratio for the given antecedent P^a .

Considering any initial p different from P_{out} , it will converge to P_{out} with experiments teaching the same rule (that is, with P_{out} as the consequent activation ratios and

the same P^a). To prove this, take $p(t+2)-p(t+1)$ and consider a consequent activation ratio of P_{out} for all experiments:

$$p(t+2)-p(t+1) = P^a \cdot (P_{out} - p(t) - P^a \cdot (P_{out} - p(t))) = (p(t+1)-p(t)) \cdot (P^a)^2 \cdot (P_{out}-p(t))$$

It is a simple matter to verify that:

$$|p(t+2)-p(t+1)| < |p(t+1)-p(t)| \text{ for any } t$$

Thus it may be concluded that with a set of coherent experiments -teaching the same rule- the net converges. It will establish the p with the same value as P_{out} , and in each experiment it will approach that value P_{out} proportionally to the distance between the present value of p and P_{out} itself, that is, with approaching zero decreasing steps.

A more difficult problem is the study of convergence in such a network, if a set of different rules is taught, not just one. And what about the number of different rules it can accommodate without interference, that is, cross learning? Obviously such a problem is equivalent to the one of studying the granularity of variables, in particular the consequent, since antecedents are limited by m .

Consider then a sequence of experiments, each one teaching a different rule. In such a case it is necessary to consider not only P^a , but the probabilities P_k^j , of activating any generic rule k antecedent part (on a neuron, each corresponds to different flip-flops), when teaching rule j with consequent P_{outj} on a given experiment.

Learning rule i , on time step t , when rule t is being taught, is given by the activation probability of the corresponding internal flip-flops: $p_i(t+1) = p_i(t) + P_i^t (P_{out} - p_i(t))$. This is known as the flywheel equation and its solution, [1], is well known when P_i^t is constant with t . The solution for $p_i(t)$ is P_{out}^t , if $1/P_i^t$ is large. However, since P_i^t varies with t , this can not be applied directly and a study must be done on the influence of varying P_i^t and P_{out}^t with time. Suppose a finite set of $R+1$ consequent singleton positions on the consequent UD, and assume an equal number of rules. This implies that a single rule is considered for each consequent fuzzy set. This is not a restriction, since it will be proved in what follows that, if there is a number of different antecedent rules with the same consequent, the system learns properly, as was the case for a single rule for each consequent singleton.

Consider this set of $R+1$ rules and assume (for commodity) that any rule k , where $k \in \{0, 1, 2, \dots, R\}$, is taught at time steps $k, k+R+1, \dots, k+r \cdot (R+1)$. Focusing on the learning of rule i one obtains, by the successive application of the equation above, and after a complete cycle of teaching each rule once:

$$p_i(i+R+1) = p_i(i) \cdot (1 - P_i^i) + \alpha \cdot (P_{out}^i - p_i(i)) \cdot (1 - P_i^i) + P_i^i P_{out}^i$$

where:

$$p_i(t) = [p_i(i) \ p_i(i) \ p_i(i) \ \dots \ p_i(i)]$$

$$\alpha = [(P_i^{i+1}) \ (P_i^{i+2}) \ (P_i^{i-1}) \ \dots \ (P_i^{i+1} \cdot P_i^{i+2}) \ \dots \ (-1)^R (P_i^{i+1} \cdot P_i^{i+2} \cdot \dots \cdot P_i^{i-1})]$$

$\mathbf{P}_{out}^i = [P_{out}^{i+1} \quad P_{out}^{i+2} \quad \dots \quad P_{out}^{i-1}]$ and the dimension of these vectors being $R + \binom{R}{2} + \binom{R}{3} + \dots + \binom{R}{R}$.

If the learning of any generic rule i is efficient, this means that $p_i(i+r.(R+1))/P_{out}^i$ approximates 1 with r (meaning that p_i equals what is being taught, without interference of other rules). Dividing both members of the equation by P_{out}^i one obtains a linear equation relating the efficiency of the rule i before and after a complete cycle of teaching every rule:

$$p_i(i+R+1)/P_{out}^i = p_i(i) \cdot (1 - \Sigma\alpha) \cdot (1 - P_i^i) / P_{out}^i + \alpha \cdot \mathbf{P}_{out}^i{}^T \cdot (1 - P_i^i) / P_{out}^i + P_i^i$$

and where $\Sigma\alpha$ means the sum of every element of vector α . Since $(1 - \Sigma\alpha) \cdot (1 - P_i^i) < 1$ there is a limit point for $p_i(i+R+1)/P_{out}^i = p_i(i)/P_{out}^i$, giving the solution, $p_i(t)$, when enough time steps have been passed:

$$p_i(t) = \alpha \cdot \mathbf{P}_{out}^i{}^T \cdot (1 - P_i^i) / (P_i^i + \Sigma\alpha - \Sigma\alpha \cdot P_i^i) + P_i^i \cdot P_{out}^i / (P_i^i + \Sigma\alpha - \Sigma\alpha \cdot P_i^i)$$

Considering a worst case where every rule distinct from i is taught the same consequent P_j , expression above becomes:

$$p_i(t) = P_j \cdot \Sigma\alpha \cdot (1 - P_i^i) / (P_i^i + \Sigma\alpha - \Sigma\alpha \cdot P_i^i) + P_i^i \cdot P_{out}^i / (P_i^i + \Sigma\alpha - \Sigma\alpha \cdot P_i^i)$$

In the case of every different antecedent rule having the same consequent ($P_j = P_{out}^i$) the expression above just reduces to $p_i(t) = P_{out}^i$, and the system learns as expected. For the more general case of different rules with different consequents it is easily seen that the condition for $p_i(t) = P_{out}^i$ (first parcel of second member tends to zero and second parcel to P_{out}^i) is:

$$P_i^i \gg \Sigma\alpha \cdot (1 - P_i^i) \quad (3)$$

Considering the behavior when m is large, the P_i^i and P_j^j are multidimensional Gaussians, calculated on the point defined by rule i antecedent. If generic rule i has A antecedents, its antecedent part can be characterized by an A -dimensional vector $\mathbf{i} = (i_1, i_2, \dots, i_A)$. Then it follows:

$$\begin{aligned} P_i^i &= N(i_1, \sqrt{i_1 \cdot (1 - i_1 / m)}) \cdot N(i_2, \sqrt{i_2 \cdot (1 - i_2 / m)}) \dots N(i_A, \sqrt{i_A \cdot (1 - i_A / m)}) \\ &= \prod_{k=1}^A N(k, \sqrt{k \cdot (1 - k / m)}) \\ \Sigma\alpha &= \sum_{p \in R \setminus i} \left(\prod_{q=1}^A N(k_p^q, \sqrt{k_p^q \cdot (1 - k_p^q / m)}) \right) + [\text{higher order terms}] \end{aligned}$$

and where $R \setminus i$ is the set of rules with exception of rule i and each of them being defined by vector $\mathbf{k}_p = (k_p^1, k_p^2, \dots, k_p^A)$.

Since the [higher order terms] give a negative contribution to the sum one may discard them, when condition (3) is investigated. Using a polynomial approximation [12] for the Gaussian distributions one obtains:

$$P_i^i = Z(0)^A \cdot \prod_{i=1}^{iA} \left(\frac{i(m-i)}{m} \right)^{-1/2} = 0.401^A \cdot \prod_{i=1}^{iA} \left(\frac{i(m-i)}{m} \right)^{-1/2}$$

$$\Sigma \alpha = \sum_{p \in R} \left(\prod_{q=1}^A (k_p^q \cdot (m - k_p^q) / m)^{-1/2} \cdot \left\langle \text{POL6} \left[k_p^q \cdot (m - k_p^q) / m \right]^{-1/2} \cdot (i_q - k_p^q) \right\rangle^{-1} \right),$$

with POL6 representing a 6 degree polynomial. If one is interested on the capacity of the system to learn different rules, it is necessary and sufficient to prove that any two consecutive antecedent parts are able to distinguish between any two consequents.

This worst case is when $(i_q - k_p^q)$ represents the difference between two consecutive counts (antecedent parts), that is m/R , where $R+1$ is the total number of rules. In order to satisfy inequality (3) it is sufficient guarantee the relation between the minimum value for the first member and the maximum for the second. As $\left(\frac{i(m-i)}{m} \right)^{-1/2}$ has a minimum for $i=m/2$, with value $2/m^{1/2}$, the minimum for the first member is $0.401^A \cdot (2/m^{1/2})^A$. The second member is $R \cdot (\Upsilon \cdot \Upsilon^{-6} \cdot (m/R)^{-6})^A$, with $\Upsilon = (k_p^q \cdot (m - k_p^q) / m)^{-1/2}$ varying between $2/m^{1/2}$ and $R^{1/2}/m^{1/2}$. The maximum value for the second member of (3) is then : $R \cdot 2^{-5A} / ((m^{1/2})^{-5A}) \cdot (m/R)^{-6A}$ and inequality (3) becomes $0.401^A \cdot (2/m^{1/2})^A >> 2^{-5A} \cdot R^{6A+1} \cdot m^{-7/2A}$, that is: $m^{-1/2A} >> 0.039^A \cdot m^{(6A+1) \cdot X - 7/2A}$. From this, making $R=m^X$ and taking the limit situation when m increases, one extracts: $X < 1/2$ that is $R < m^{1/2}$.

As R represents the consequent granularity, i.e., the number of distinguishable (for effective learning) consequent singletons, if one considers different rules those with different consequent parts (that is, of type: IF (A is A1 AND B is B1 AND ...) OR (A is A2 AND B is B2 AND...) OR... THEN Z is Zi), the following sufficient condition has been obtained :

It is a sufficient condition for Boolean neural nets to learn any set of rules, that the number of rules increases with the square root of m.

4 Traffic Application

There are real applications of this work, despite its main objective is simply to present a possible model for explanation of reasoning and learning in natural neural networks. Thus, one could use it in any context where each neuron can be associated with an agent that partially observes local samples of the real world. Different experiments could be simply different observations by the set of agents (neurons).

Consider, as a possible very simple example, that it is pretended to establish a qualitative base rule, capable of relating traffic conditions on three one lane main roads entering a town with the traffic on an internal one lane street of the same town. This is, clearly, a three antecedent one consequent problem. The town administration could admit a number of persons - the agents - and separate them by the three roads and the street to observe the traffic - N persons per street or road. Each one of them positioned on the roads (antecedents) should note down if there is a car passing in

front of him at times $t_1, t_2, \dots, t_m$, count that number and record it, and repeat the experiment on a previously accorded schedule defined by a set of $t_1 \leq$ and where intervals between t_j and t_{j+1} are previously defined (X experiments which could be distributed along the day or days). As for those positioned on the street (consequent), each one should note down if there is a car passing in front of him or not at times t_1 . This should be a completely decentralized set of experiments (one experiment considered to be the set of the N observations both for the consequent and antecedents), very easy to implement and where it is not asked to anybody to make, only by himself and dependent of subjective errors, any global assessment of the traffic conditions. If it is admitted that the time slots (t_m, t_1) are short enough to consider that the traffic conditions do not vary during each one of them and that people are randomly distributed along the roads one gets the conditions to use the net. At the end of the set of experiments, the data of each experiment - N recorded values $(C_i, i=1, N)$ of the street observers together with 3 random chosen person records ($A1_j, A2_k, A3_l$), one from each road $\square$ is used to teach the net. The process is repeated for every experiment.

A simulation of this example has been carried out, where 40 neurons per antecedent and consequent have been use and a set of 4 cycles of teaching experiments was used. In each cycle traffic has been generated on the roads and street using three traffic situations both for antecedents and consequent. For the antecedents, and for each experiment, it has been used a binomial distribution to generate the number of active antecedent neurons on each set of 8 neurons ($m=8$), with the following partition of activated neurons: $S0 \equiv \{0,1,2\}$; $S1 \equiv \{3,4,5\}$ and $S3 \equiv \{6,7,8\}$. The traffic conditions were randomly generated using as individual probabilities, for the binomial, $p=0.1$; $p=0.5$ and $p=0.9$, which can be labeled as **Low**, **Medium** and **High** respectively. For the consequent random traffic has been originated with three different probabilities: 0.15, 0.5 and 0.85, which may also be labeled as **Low**, **Medium** and **High**, respectively. Using these settings, each cycle taught the following set of traffic conditions: **LLL_L**; **LLM_L**; **LLH_L**; **LML_L**; **LMM_L**; **LMH_L**; **LHL_L**; **LHM_L**; **LHH_L**; **MLL_M**; **MML_M**; **MLH_L**; **MMH_L**; **MHL_M**; **MHM_M**; **MHH_M**; **HLL_H**; **HML_H**; **HHL_H**; **HLH_M**; **HMH_M** and **HHH_M**, where the first three letters indicate the traffic conditions of the three antecedents and the fourth the consequent traffic condition. As it is seen there is a set of rules not taught (**MMM_?**, for example), in order to evaluate the behavior of the net in those conditions, and for that purpose a third state (not taught) was used on the simulation for each flip-flop. The results, expressing the activation ratio of the internal flip-flops for each antecedent rule, which is the same as the learnt consequent activation ratio for that antecedent rule, can be seen on figures 2(a), 2(b) and 2(c), respectively for traffic conditions on Road1 $\square$ Low $\square$ $\square$ Medium $\square$ and $\square$ High $\square$ It may be noticed that the network has apprehended the taught rules and illustrates the fact that some traffic conditions are much better defined (more $\square$ crisp $\square$) than others. It is easily noticed that, for traffic conditions not taught, the ratio of the third state (not taught) is much higher and, in such cases, the activation ratios for 1 $\square$ and 0 $\square$ are influenced by neighbor taught rules. As a conclusion, the network has been able to learn the set of rules presented (not explicitly) during the teaching experiments.

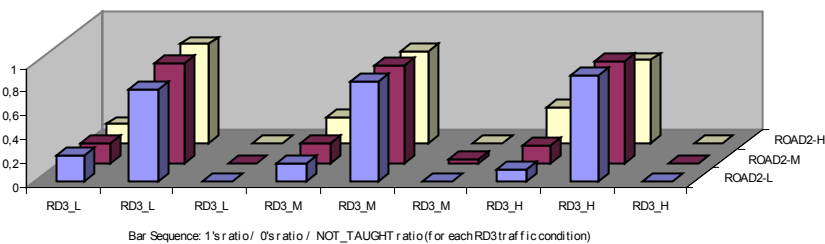


Fig 2(a) - Traffic Low on Road1

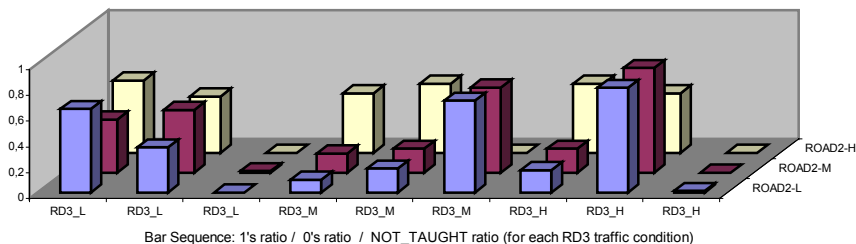


Fig 2(b)-Traffic Medium on Road1

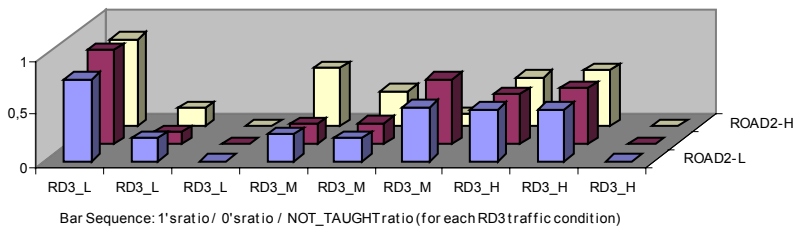


Fig 2(c)-Traffic High on Road1

5 Conclusions

A class of neural nets which functionality seems to be more similar to natural nets than the classic neural nets has been presented elsewhere[9]. In this class of networks variables or concepts are associated with different neural areas, meshes of links are established from areas to other areas depending on antecedent-consequent dependence, individual links are randomly established between neurons of those areas, individual neurons have only a two state space dimension (fire/do not fire) and the notion of "amplitude" of each variable is given by a natural activation ratio (that is the fraction

of activated neurons on a given area). The net is also totally digital (binary); there are no weights. Robustness is an intrinsic property of such nets: any number of neurons or connections may be deleted or corrupted, provided the remaining neurons are enough to define accurately the activation ratios

Moreover these nets are capable of reasoning, implementing qualitative rules. It appears that fuzzy reasoning is a natural emergent property of these networks. Mechanisms for non-supervised learning of fuzzy rules from real experiments use Hebbian like concepts. It has been shown that the learning process converges, not only for one single rule but also for a repetitive sequence of different rules. In such a case it has been concluded that, in the limit, the system is able to learn a number of rules compatible with a consequent granularity increasing with the square root of m , the number of inputs per neuron and per antecedent.

References

- [1] Goldberg, S. Introduction to Difference Equations, Dover Publications, NY. (1958).
- [2] Gupta, M. and QI "On fuzzy neurone models" Proc. Int. Joint Conf. Neural Networks, volII, 431-436, Seattle. (1991).
- [3] Hayashi, Y., E. Czogala and J.Buckley "Fuzzy neural controller" Proc. IEEE Int. Conf. Fuzzy Systems, 197-202, San Diego. (1992).
- [4] Hebb, D. The Organization of Behaviour: A Neuropsychological Theory. John Wiley & Sons. (1949).
- [5] Horikawa, S., T. Furuhashi and Y. Uchikawa "On fuzzy modelling using fuzzy neural networks with the back propagation algorithm." IEEE Transactions Neural Networks 3(5): 801-806. (1992).
- [6] Keller, J.M. and D.J.Hunt "Incorporating fuzzy membership functions into the perceptron algorithm." IEEE Transactions Pattern Anal. Mach. Intell., PAMI-7(6):693-699. (1985).
- [7] Lin, Chin-Ten and Lee, C.S. A Neuro-Fuzzy Synergism to Intelligent Systems. New Jersey : Prentice Hall. (1996).
- [8] Pedrycz, W. "Fuzzy neural networks with reference neurones as pattern classifiers" IEEE Trans. Neural Networks 3(5):770-775. (1992).
- [9] Tom, J.A.. "Neural Activation ratio based Fuzzy Reasoning." Proc. IEEE World Congress on Computational Intelligence, Anchorage, May 1988, pp 1217-1222. (1998, a).
- [10] Tom, J.A. "Counting Boolean Networks are Universal Approximators" Proc. of 1988 Conference of NAFIPS, Florida, August 1988, pp 212-216. (1998, b)
- [11] Yager, R. "OWA neurones: A new class of fuzzy neurones". Proc. Int. Joint Conference Neural Networks, vol I, 226-231, Baltimore. (1992).
- [12] Handbook of Mathematical, Scientific and Engineering Formulas, Tables, Functions, Graphs, Transforms. Research and Education Association, New Jersey.

Attribute Selection with a Multi-objective Genetic Algorithm

Gisele L. Pappa, Alex A. Freitas, and Celso A.A. Kaestner

Pontifícia Universidade Católica do Paraná (PUCPR),
Postgraduated Program in Applied Computer Science,
Rua Imaculada Conceição, 1155, Curitiba - PR, 80215-901, Brazil
{gilpappa, alex, kaestner}@ppgia.pucpr.br
<http://www.ppgia.pucpr.br/~alex>

Abstract. In this paper we address the problem of multi-objective attribute selection in data mining. We propose a multi-objective genetic algorithm (GA) based on the wrapper approach to discover the best subset of attributes for a given classification algorithm, namely C4.5, a well-known decision-tree algorithm. The two objectives to be minimized are the error rate and the size of the tree produced by C4.5. The proposed GA is a multi-objective method in the sense that it discovers a set of non-dominated solutions (attribute subsets), according to the concept of Pareto dominance.

1 Introduction

The amount of data stored in real-world databases grows much faster than our ability to process them, so that the challenge is to extract useful knowledge from data. This is the core idea of the field of data mining, where the goal is to discover, from real-world data sets, knowledge that is accurate, comprehensible, and useful for the user.

The complexity and large size of real-world databases have motivated researchers and practitioners to reduce the data dimensionality, for data mining purposes. This dimensionality reduction can be done in two main directions. First, one can select a random sample of the available records, reducing the number of records to be mined. Second, one can select a subset of the available attributes [10], [11], which can also reduce the number of records to be mined. The latter is the direction followed in this paper.

Attribute selection is important because most real-world databases were collected for purposes other than data mining [8]. Hence, a real-world database can have many attributes that are irrelevant for data mining, so that by discarding the irrelevant attributes one can actually improve the performance of a data mining algorithm. In addition, providing a data mining algorithm with a subset of attributes reduces the computational time taken by that algorithm, by comparison with using the entire set of available attributes. As a result, attribute selection is an active research area in data mining.

The goal of attribute selection is to discover a subset of attributes that are relevant for the target data mining task. In this paper we address the task of classification, where the goal is to predict the class of an example (a record) based on the values of the predictor attributes for that example. In the context of this task, in general two important objectives of attribute selection are to minimize the error rate of the classification algorithm and the complexity of the knowledge discovered by that algorithm. These are the two objectives to be minimized in this work.

Note that attribute selection, like many other data mining problems, involve the [simultaneous] optimization of more than one objective. However, such a simultaneous optimization is not always possible. The objectives to be optimized can be conflicting with one another, and they normally are non-commensurable [i.e., they measure different aspects of the target problem].

In order to solve problems involving more than one objective, recently there has been a growing amount of research in the area of multi-objective optimization [3]. The basic idea is to return to the user, as the result of the problem-solving algorithm, a set of optimal solutions (rather than a single solution) by taking both objectives into account, without *a priori* assigning greater priority to one objective or the other. The ultimate choice about which solution should be used in practice is left to the user, which can use his/her background knowledge and experience to choose the [best] solution for his/her needs *a posteriori*, among all the returned optimal solutions. The motivation for multi-objective optimization will be discussed in more detail in section 3.

Casting the attribute selection problem as a multi-objective optimization problem, this paper proposes a multi-objective genetic algorithm (GA) for attribute selection in the classification task of data mining. The paradigm of GA was chosen for the development of our attribute selection method mainly for the following reasons. First, GAs are a robust search method, capable of effectively exploring large search spaces, which is usually the case in attribute selection. Note that the size of the search space in attribute selection is 2^M , where M is the number of attributes [i.e., the size of the search space grows exponentially with the number of attributes]. Second, unlike many search algorithms which perform a local, greedy search, GAs perform a global search [5]. In the context of data mining, this global search means that GAs tend to cope better with attribute interaction than greedy search methods [6], [7]. Finally, it is important to notice that multi-objective optimization requires a problem-solving algorithm that is capable of considering a set of optimal solutions at each iteration, and this requirement is naturally satisfied by GAs, which work with a population of individuals, or candidate solutions [3].

The remainder of this paper is organized as follows. Section 2 reviews the main concepts of attribute selection. Section 3 discusses multi-objective optimization. Section 4 describes the proposed multi-objective GA for attribute selection. Section 5 reports computational results. Finally section 6 presents the conclusions of this work.

2 Attribute Selection

Attribute selection is one of the main preprocessing tasks for the application of a data mining algorithm [10]. As mentioned in the Introduction, the general goal of attribute

selection is to select a subset of attributes that are relevant for the target data mining task, out of all available attributes. In the classification task, which is the task addressed in this work, an attribute is deemed relevant if it is useful for discriminating examples belonging to different classes.

More specific goals of attribute selection are as follows:

- Improving the performance of a data mining algorithm with respect to several criteria, such as reducing the classification error rate and/or complexity (size) of discovered knowledge and reducing the processing time of the data mining algorithm;
- Removing noisy and/or irrelevant attributes, reducing the dimensionality of the data (which not only helps to improve the performance of the data mining algorithm, but also saves storage space, in the case of very large data sets).

There are many methods that can be used for attribute selection. They can be characterized mainly with respect to the search strategy used to explore the space of candidate attribute subsets and with respect to the evaluation function used to measure the quality of a candidate attribute subset.

With respect to the search strategy, two well-known methods are forward sequential selection (FSS) and backward sequential selection (BSS) [10]. In essence, FSS starts with an empty set of selected attributes and it adds one attribute at a time to that set until a stopping criterion is met — e.g., until the quality of the current set of selected attributes cannot be improved by adding another attribute to that set. BSS follows the opposite strategy. It starts with the full set of original attributes, and it removes one attribute at a time from that set until a stopping criterion is met. In both methods, the attribute chosen to be added or removed at each step is the one maximizing the value of some evaluation function. Hence, both are greedy methods, working with one attribute at a time, and therefore having the drawback of being sensitive to attribute interaction.

With respect to the evaluation function, attribute selection methods can be classified into the filter approach or the wrapper approach. This classification is independent of the search strategy used by the attribute selection method. It depends on whether or not the evaluation function uses the target data mining algorithm (which will be eventually applied to the ultimate set of selected attributes) to evaluate the quality of a candidate attribute subset. In the filter approach the attribute selection method does not use the data mining algorithm, whereas in the wrapper approach the attribute selection method uses the data mining algorithm to evaluate the quality of a candidate attribute subset. Note that the data mining algorithm is used as a black box.

The wrapper approach tends to obtain a predictive accuracy better than the filter approach, since it finds an attribute subset “customized” for a given data mining algorithm. However, the wrapper approach is considerably more computationally expensive than the filter approach, since the former requires many runs of the data mining algorithm.

3 Multi-objective Optimization

Many real-world problems involve the optimization of multiple objectives. However, the majority of methods used to solve these problems avoids the complexities associated with multi-objective optimization. As a result, many methods have been proposed to convert multi-objective problems into single objective problems [3]. Some of them can be found in [9], [14].

Once several conversion methods are available, it has been forgotten that, in reality, a single objective optimization problem is a degenerated case of a multi-objective optimization problem, and that there are crucial differences between the two kinds of problems. The main difference concerns the desired number of optimal solutions. In single objective optimization one usually wants to discover a single optimal solution. By contrast, assuming that the different objectives to be optimized represent conflicting goals (such as improving the quality of a product and reducing its cost), in multi-objective optimization the optimization of each objective corresponds to an optimal solution. Therefore, in multi-objective optimization one usually wants to discover several optimal solutions, taking all objectives into account, without assigning greater priority to one objective or the other. The ultimate choice about which solution should be used in practice is left to the user, which can use his/her background knowledge and experience to choose the "best" solution for her needs, among all returned optimal solutions.

In other words, in a multi-objective optimization framework the user has the advantage of being able to choose the solution representing the best trade-off between conflicting objectives *a posteriori*, after examining a set of high-quality solutions returned by the multi-objective problem-solving algorithm. Intuitively, this is better than forcing the user to choose a trade-off between conflicting goals *a priori*, which is what is done when a multi-objective optimization problem is transformed into a single-objective one.

In multi-objective optimization, in order to take all the objectives into account as a whole during the search for optimal solutions, one uses the concept of Pareto dominance, as follows. A given solution x_1 dominates another solution x_2 if and only if:

- 1 Solution x_1 is not worse than solution x_2 in any of the objectives;
- 2 Solution x_1 is strictly better than solution x_2 in at least one of the objectives.

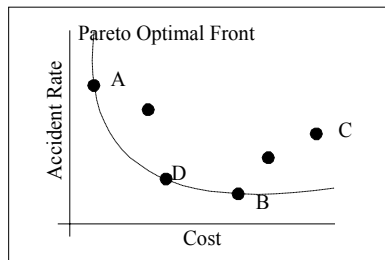


Fig. 1: Example of Pareto dominance in a two-objective problem [2]

The solutions that are not dominated by any other solution are considered Pareto-optimal solutions. Figure 1 shows a set of possible solutions for a hypothetical problem with two objectives to be minimized, namely accident rate and cost. Note that solution A has a small cost but a large accident rate. Solution B has a large cost but a small accident rate. Assuming that minimizing both objectives is important, one cannot say that solution A is better than B, nor vice-versa. In addition, solution D cannot be considered better than A or B. The three solutions A, B, and D are Pareto-optimal solutions: none of them is dominated by other solutions. These solutions are included in the Pareto front, represented by the dotted line in Figure 1. Note that solution C is not a Pareto-optimal solution, since it is dominated, for instance, by solution B (which is better than C with respect to both objectives).

4 The Proposed Multi-objective GA for Attribute Selection

A genetic algorithm (GA) is a search algorithm inspired by the principle of natural selection. The basic idea is to evolve a population of individuals, where each individual is a candidate solution to a given problem. Each individual is evaluated by a fitness function, which measures the quality of its corresponding solution. At each generation (iteration) the fittest (the best) individuals of the current population survive and produce offspring resembling them, so that the population gradually contains fitter and fitter individuals □ i.e., better and better candidate solutions to the underlying problem. For a comprehensive review of GAs in general the reader is referred to [12], [5]. For a comprehensive review of GAs applied to data mining the reader is referred to [7].

This work proposes a multi-objective genetic algorithm (GA) for attribute selection. As mentioned in the Introduction, our motivation for developing a GA for attribute selection, in a multi-objective optimization framework, was that: (a) GAs are a robust search method, capable of effectively exploring the large search spaces often associated with attribute selection problems; (b) GAs perform a global search [7], [4], so that they tend to cope better with attribute interaction than greedy search methods [6], [7], which is also an important advantage in attribute selection; and (c) GAs already work with a population of candidate solutions, which makes them naturally suitable for multi-objective problem solving [3], where the search algorithm is required to consider a set of optimal solutions at each iteration.

The goal of the proposed GA is to find a subset of relevant attributes that leads to a reduction in both the classification error rate and the complexity (size) of the rule set discovered by a data mining algorithm (improving the comprehensibility of discovered knowledge).

In this paper the data mining algorithm is C4.5 [15], a very well-known decision tree algorithm. The proposed GA follows the wrapper approach, evaluating the quality of a candidate attribute subset by using the target classification algorithm (C4.5). Hence, the fitness function of the GA is based on the error rate and on the size of the decision tree built by C4.5. These two criteria (objectives) are to be minimized according to the concept of Pareto dominance. The main aspects of the proposed GA are described in the next subsections.

4.1 Individual Encoding

In the proposed GA, each individual represents a candidate subset of selected attributes, out of all original attributes. Each individual consists of M genes, where M is the number of original attributes in the data being mined. Each gene can take on the value 1 or 0, indicating that the corresponding attribute occurs or not (respectively) in the candidate subset of selected attributes.

4.2 Fitness Function

The fitness (evaluation) function measures the quality of a candidate attribute subset represented by an individual. Following the principle of multi-objective optimization, the fitness of an individual consists of two quality measures: (a) the error rate of C4.5; and (b) the size of the decision tree built by C4.5. Both (a) and (b) are computed by running C4.5 with the individual's attribute subset only, and by using a hold-out method to estimate C4.5's error rate, as follows. First, the training data is partitioned into two mutually-exclusive data subsets, the building subset and the validation subset. Then we run C4.5 using as its training set only the examples (records) in the building subset. Once the decision tree has been built, it is used to classify examples in the validation set. The two components of the fitness vector are then the error rate in the validation set and the size (number of nodes) of the tree built by C4.5.

4.3 Selection Method and Genetic Operators

At each generation (iteration) of the GA, the selection of individuals to reproduce is performed as follows. First the GA selects all the non-dominated individuals (the Pareto front) of the current population. These non-dominated individuals are passed unaltered to the next generation by elitism [1]. Elitism is a common procedure in GAs, and it has the advantage of avoiding that good individuals disappear from the population due to the stochastic nature of selection.

Let N be the total population size (which is fixed for all generations, as usual in GAs), and let N_{elit} be the number of individuals reproduced by elitism. Then the other $N - N_{elit}$ individuals to reproduce are chosen by performing $N - N_{elit}$ times a tournament selection procedure [12], as follows. First, the GA randomly picks k individuals from the current population, where k is the tournament size, a user-specified parameter which was set to 2 in all our experiments. Then the GA compares the fitness values of the two individuals playing the tournament and selects as the winner the one with the best fitness values.

The selection of the best individual is based on the concept of Pareto dominance, taking into account the two objectives to be minimized (error rate and decision tree size). Given two individuals I_1 and I_2 playing a tournament, there are two possible situations. The first one is that one of the individuals dominates the other. In this case the former is selected as the winner of the tournament.

The second situation is that none of the individuals dominates the other. In this case, as a tie-breaking criterion, we compute an additional measure of quality for each individual by taking both objectives into account. Following the principle of Pareto dominance, care must be taken to avoid that this tie-breaking criterion assigns greater

priority to any of the objectives. Hence, we propose the following tie-breaking criterion. For each of the two individuals I_i , $i=1,2$, playing a tournament, the GA computes X_i as the number of individuals in the current population that are dominated by I_i , and Y_i as the number of individuals in the current population that dominate I_i . Then the GA selects as the winner of the tournament the individual I_i with the largest value of the formula: $X_i - Y_i$. Finally, if I_1 and I_2 have the same value of the formula $X_i - Y_i$ (which is rarely the case), the tournament winner is simply chosen at random.

Individuals selected by tournament selection undergo the action of two standard genetic operators, crossover and mutation, in order to create new offspring [12]. In essence, crossover consists of swapping genes (bits, in our individual encoding) between two individuals, whereas mutation replaces the value of a gene with a new randomly-generated value. In our individual encoding, where each gene is a bit, mutation consists simply of flipping the value of a bit. These operators are applied with user-specified probabilities. In all our experiments the probabilities of crossover and mutation were set to 80% and 1%, respectively, which are relatively common values in the literature.

The population size N was set to 100 individuals, which evolve for 50 generations. These values were used in all our experiments. The pseudocode of the GA is shown, at a high level of abstraction, in Algorithm 1. (Note that this pseudocode abstracts away details such as the fact that crossover and mutation are applied with user-defined probabilities. It shows only an overview of the flow of processing of the GA.)

```
Create initial population
FOR EACH generation DO
  FOR EACH Individual DO
    Run C4.5 with attribute subset represented by the individual
    Compute multi-objective fitness // error rate and tree size
  END FOR
  Add non-dominated individuals to next generation's population
  FOR i ← 1 to (N - Nelite)/2 DO
    Perform tournament selection twice, to select two parent
    individuals,  $P_1$  and  $P_2$ 
    Perform crossover of  $P_1$  and  $P_2$ , producing children  $C_1$  and  $C_2$ 
    Perform mutation on  $C_1$  and  $C_2$ 
    Add  $C_1$  and  $C_2$  to the next generation's population
  END FOR
END FOR
Compute fitness of the individuals of the last generation
Return all non-dominated individuals of the last generation
```

Algorithm 1. Pseudocode of the proposed multi-objective GA

5 Computational Results

We have performed experiments with six public-domain, real-world data sets obtained from the UCI (University of California at Irvine) data set repository [13]. The number of examples, attributes and classes of these data sets is shown in Table 1.

Table 1. Main characteristics of the data sets used in the experiments

Data Set	# examples	# attributes	# classes
Dermatology	366	36	6
Vehicle	846	18	4
Promoters	106	57	2
Ionosphere	351	34	2
Crx	690	15	2
Arrhythmia	452	269	16

All the experiments were performed by using a well-known 10-fold stratified cross-validation procedure, as follows. For each data set, all the available data is divided into 10 mutually-exclusive and exhaustive partitions having approximately the same size. In addition, each partition has approximately the same class distribution (stratified cross validation). Then the GA and C4.5 are run 10 times. In the i -th run of the algorithms, $i=1,\dots,10$, the i -th partition is used as the test set and the other 9 partitions are used as the training set. All results reported in this paper refer to average results in the test set over the 10 iterations of the cross-validation procedure.

At each iteration of the cross-validation procedure, a GA run is performed as follows. For each individual of the GA, out of the 9 partitions used as the training set, 8 partitions (the building subset mentioned in subsection 4.2) are used by C4.5 to build a decision tree, and the remaining partition (the validation subset mentioned in subsection 4.2) is used to compute the error rate of C4.5. We emphasize that the examples in the test set are never used during the evolution of the GA.

Finally, for each iteration of the cross-validation procedure, once the GA run is over we compare the performance of C4.5 using all the original attributes with the performance of C4.5 using only the attributes selected by the GA. In both runs of C4.5, the decision tree is built using the entire training set (9 partitions), and then we measure C4.5's error rate in the test set.

Therefore, the GA can be considered successful to the extent that the attributes subsets selected by it lead to a reduction in the error rate and size of the tree built by C4.5, by comparison with the use of all original attributes.

There is a final point concerning the evaluation of the solutions returned by the GA. It should be noted that, as explained before, the solution for a multi-objective optimization problem consists of all non-dominated solutions (the Pareto front). Hence, each run of the GA outputs the set of all non-dominated solutions (attribute subsets) present in the last generation's population. In a real-world application, it would be left to the user the final choice of the solution to be used in practice. However, in our research-oriented work, involving public-domain data sets, no user was available. Hence, in order to evaluate the quality of the non-dominated attribute subsets found by the GA in an automatic, data-driven manner — as usual in the majority of the data mining and machine learning literature — we measure the error rate and the size of the decision tree built by C4.5 using each of the non-dominated attribute subsets returned by the GA. The ultimate results associated with the attributes selected by the GA, which are the results reported in the following, are the corresponding arithmetic average over all non-dominated solutions returned by the GA.

The results of our experiments are reported in Table 2. The first column indicates the name of the data set. The second and third columns indicate the error rate obtained by C4.5 using only the attributes selected by the GA and using all original attributes, respectively. The fourth and fifth columns indicate the size of the decision tree built by C4.5 using only the attributes selected by the GA and using all original attributes, respectively. In each cell of the table, the value before the $\pm$ symbol is the average result over the 10 iterations of the cross-validation procedure, and the value after the $\pm$ symbol is the corresponding standard deviation. In addition, in the second and fourth columns the values of a given cell are shown in bold when the corresponding result in that cell is significantly better than the result in the third and fifth columns, respectively. A result is considered significantly better than another when the corresponding intervals, taking into account the standard deviations, do not overlap.

As shown in Table 2, the error rate associated with the attributes selected by the GA is better than the one associated with all attributes in three data sets, and the difference between the two error rates is significant in one data set. In the other three data sets, although the error rate associated with the attributes selected by the GA is somewhat worse than the one associated with all attributes, the differences between the two error rates are not significant $\square$ i.e., the corresponding intervals (taking into account the standard deviations) overlap.

Table 2. Computational Results with 10-fold stratified cross validation

Data Set	Error Rate (%)		Decision Tree Size	
	C4.5 + GA	C4.5 alone	C4.5 + GA	C4.5 alone
Dermatology	5.5 $\pm$ 1.46	4.2 $\pm$ 0.96	14.8 $\pm$ 1.08	17.1 $\pm$ 0.34
Vehicle	29.9 $\pm$ 0.70	29.6 $\pm$ 1.15	151.9 $\pm$ 8.32	181 $\pm$ 3.24
Promoters	14.1 $\pm$ 4.02	21.2 $\pm$ 3.05	16.8 $\pm$ 1.32	17.6 $\pm$ 0.99
Ionosphere	10.2 $\pm$ 1.16	8.5 $\pm$ 1.20	20.8 $\pm$ 1.62	24 $\pm$ 1.2
Crx	14.4 $\pm$ 1.38	16.3 $\pm$ 1.2	8.6 $\pm$ 0.71	69.4 $\pm$ 2.72
Arrhythmia	31.6 $\pm$ 2.6	32.0 $\pm$ 2.36	64.1 $\pm$ 2.3	75.4 $\pm$ 1.7

In Table 2 we also note that the tree size associated with the attributes selected by the GA is better than the one associated with all attributes in all the six data sets, and the difference is significant in five data sets.

In summary, the use of the GA has led to a significant reduction in the size of the trees built by C4.5 in five data sets, without significantly increasing C4.5's error rate in any data set $\square$ and even significantly reducing C4.5's error rate in one data set.

One disadvantage of the use of the GA is that it is computationally expensive. In the two largest data sets used in our experiments, Vehicle (with the largest number of examples) and Arrhythmia (with the largest number of attributes, viz. 269), a single run of the GA took about 25 minutes and 5 hours and 15 minutes, respectively; whereas a single run of C4.5 took less than one minute and one and half minute, respectively. (The results were obtained in a Pentium-IV PC with clock rate of 1.7 GHz and 512 Mb of RAM.) We believe the increase in computational time associated with the GA is a relatively small price to pay for its associated increase in the comprehensibility of discovered knowledge. Data mining is typically an off-line task, and it is well-known that in general the time spent on running a data mining algorithm is a small fraction

(less than 20%) of the total time spent with the entire knowledge discovery (KD) process. Hence, in many applications, even if a data mining algorithm is run for several days, this is acceptable, at least in the sense that it is not the bottleneck of the KD process. In any case, if necessary the computational time associated with the GA can be greatly reduced by using parallel processing techniques, since GAs can be easily parallelized [7].

6 Conclusions and Future Work

In this paper we have proposed a multi-objective genetic algorithm (GA) for attribute selection in the classification task of data mining. The goal of the GA is to select a subset of attributes that minimizes both the error rate and the size of the decision tree built by C4.5. The latter objective involves a commonplace measure of simplicity (or comprehensibility) in the data mining and machine learning literature. The smaller the size of a decision tree, the simpler it is, and so the more comprehensible to the user it tends to be. We emphasize that, in data mining, maximizing comprehensibility tends to be at least as important as minimizing error rate [7], [15]. In order to minimize the two objectives at the same time, the GA uses the concept of Pareto dominance, so that each GA run returns, as its output, the set of all non-dominated solutions found during the search.

We have done experiments with six data sets, comparing the error rate and the size of the decision tree built by C4.5 in two cases: using only the attributes selected by the GA and using all attributes. The results of these experiments have shown that, on average over all non-dominated solutions (attribute subsets) returned by the GA, the use of the GA as an attribute selection method has led to: (a) a significant reduction of the size of the tree built by C4.5 in five out of the six data sets; and (b) a significant reduction of C4.5's error rate in one data set. There was no case where the use of the GA as an attribute selection method has led to an error rate or tree size significantly worse than the ones associated with the use of all attributes.

With respect to future research, we have noted that in some cases the GA population converges very fast, possibly corresponding to a premature convergence of the population. We are currently investigating the use of a niching method to promote greater population diversity, in order to reduce this premature convergence problem.

References

- [1] Bhattacharyya, S., Evolutionary Algorithms in Data mining: Multi-Objective Performance Modeling for Direct Marketing. In: Proc KDD-2000, ACM Press (2000) 465-471
- [2] Deb, K, Multi-Objective Evolutionary Algorithms: Introducing Bias Among Pareto-Optimal Solutions. Kanpur Genetic Algorithms Laboratory Report n° 99002, India (1999)
- [3] Deb, K., Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, England (2001)

- [4] Fidelis, M.V., Lopes, H.S., Freitas, A.A., Discovering Comprehensible Classification Rules with a Genetic Algorithm. In: Proc. Congress on Evolutionary Computation (2000)
- [5] Goldberg, D. E., Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company (1989)
- [6] Freitas, A. A.; Understanding the Crucial Role of Attribute Interaction in Data Mining. In: Artificial Intelligence Review 16, Kluwer Academic Publishers (2001) 177-199
- [7] Freitas, A.A., Data Mining and Knowledge Discovery with Evolutionary Algorithms (forthcoming book). Springer-Verlag (2002)
- [8] Holsheimer, M., Siebes, A., Data Mining □ The Search for Knowledge in Databases. Report CS-R9406, Amsterdam: CWI (1991)
- [9] Ishibuchi, H., Nakashima, T., Multi-objective Pattern and Feature Selection by a Genetic Algorithm. In: Proc. Genetic and Evolutionary Computation Conf. (GECCO□2000), Morgan Kaufmann (2000) 1069-1076
- [10] Liu, H.; Motoda, H., Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers (1998)
- [11] Martn-Bautista, M. J., Vila, M. A., A Survey of Genetic Feature Selection in Mining Issues. In: Proc. IEEE Conference on Evolutionary Computation, Washington (1999) 1314-1321.
- [12] Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag (1996)
- [13] Murphy, P.M., Aha, D.W., UCI Repository of Machine Learning databases. [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of information and Computer Science (1994)
- [14] Rozsypal, A., Kubat, M., Using Genetic Algorithm to Reduce the Size of a Nearest-Neighbor Classifier and Select Relevant Attributes. Proc. Int. Conf. Machine Learning (ICML-2001), Morgan Kauf. (2001)
- [15] Quinlan, J.R., C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)

Applying the Process of Knowledge Discovery in Databases to Identify Analysis Patterns for Reuse in Geographic Database Design

Carolina Silva, Cirano Iochpe, and Paulo Engel

Instituto de Informática da UFRGS

Av. Bento Gonçalves, no.9500, bloco 4. Agronomia. 91.501-970. Porto Alegre, RS.
{carolina,ciochpe,engel}@inf.ufrgs.br

Abstract. Little support has been offered by geographic information systems (GIS) suppliers to reduce the complexity of geographic database (GDB) design. Design specialists [1] suggest that naive designers try to reuse at least parts of already existent, successful database schemes to reduce the effort that has to be invested in new projects. This so-called analysis patterns approach [2], [3] has a widespread acceptance in the area of software engineering. Although very promising, the use of analysis patterns in GDB design is yet very restrict. The main problem is the lack of a well known as well as globally accepted set of patterns for database design. This paper proposes the identification of analysis patterns on the basis of the Process of Knowledge Discovery in Databases (KDD). KDD supports the processing of a huge volume of database schemas and can help reducing the dependency on the subjective analysis of human specialists.

1 Introduction

Geographic information systems (GIS) enable users to store as well as manipulate and analyze georeferenced, spatio-temporal data [4]. They are mainly used as decision-support tools.

One of the most important subsystems of a GIS is its geographic database (GDB). Although the database literature prescribes a well-known and robust methodology for database design, most of the nowadays-existing GDB have not been designed properly. This is due to the fact that GDB are mainly developed by professionals with no knowledge of this methodology such as architects, cartographers, and environment engineers. Unfortunately, in many cases this has led to an incorrect as well as inefficient geographic database.

One possible solution to the above mentioned problem would be to enable non-skilled designers to reuse best practices of skilled database designers. This is possible through the concept of *analysis patterns* [5]. Patterns are generalizations of accepted solutions to a specific class of problems. In the case of GDB design, patterns are fre-

quently applied as database conceptual sub schemas [3]. The use of analysis patterns to support GDB design is not yet as popular as it could be, mostly because of the lack of a well known as well as globally accepted set of patterns for database design. Two main reasons for that are the need to compare a vast number of database schemas and the dependency on the knowledge of design specialists. This so-called *pattern mining* process [6] has not been automatized yet.

As to reduce dependence on human specialists as well as to be able to process a greater number of database schemas, this paper proposes the use of the process of knowledge discovery in databases (KDD) [6] to help identifying analysis pattern candidates in a set of GDB conceptual schemas.

The KDD process is structured as a sequence of processing steps. Our proposal concerns how to execute these steps in order to use KDD efficiently for the identification of analysis patterns.

First, the criteria used to select a data mining (DM) technique as well as the application requirements are discussed. Then, based on the selected technique, the paper explains how to organize and store GDB schemas that will be used as data input to the mining process. Finally, the paper discusses some post-processing activities necessary to better interpret the out coming rules.

2 Data Models, Database Schemas and Analysis Patterns

A data model is a set of concepts and a set of rules of how to relate them in order to formally describe the structure of a database [7]. Usual constructing concepts are *data types*, *relationships*, and *integrity rules*.

Data models exist in different abstraction levels. A *conceptual data model* enables a very high level representation of the reality and its resulting *database conceptual schemas* are independent from database management system (DBMS) or DBMS technology. The Entity-Relationship Model [8] is an example of a popular conceptual data model.

A specific representation of reality that can be built from the constructs of a data model is called a *database schema*. Thus, database schemas relying on a conceptual, object-oriented data model as, for instance, the UML [9] combine occurrences of constructs such as *classes*, *associations* between *classes*, *class attributes*, and *packages* (i.e., encapsulated sub-schemas). Examples of schema construction rules in this context are both that two *classes* can be related only through an *association* between them, and that each *class attribute* characterizes exclusively one specific *class*.

In [10], a comparison of various conceptual data models for GDB design is presented. Despite some differences among them, a set of common as well as basic constructs for GDB conceptual design can be identified.

From the experience of designing databases, for either geographic or non-geographic applications, data engineers develop the necessary skills for identifying sub-schemas, which can be reused for different applications. Relying on their capacity of perception as well as their abstraction ability, design specialists can both identify and describe analysis patterns for further reuse by other designers. These so-called patterns are descriptions of usual as well as already tested solutions of recurrent problems [2].

An analysis pattern is any reusable part of a requirements analysis specification, which in turn is the result of some information system design process [5]. It can be described either in a high level of abstraction as in [2] or for a specific application domain as in [3]. Actually, most analysis patterns for GDB design are described in a low level of abstraction as sub-schemas that can be directly applied in the development of other database schemas of related applications.

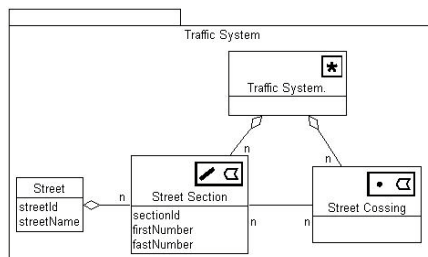


Fig. 1. Example of an analysis pattern, based on *GeoFrame* [11], an object-oriented framework for GDB design

Mainly unskilled designers find it very useful to apply analysis patterns instead of starting a new design process from scratch. Therefore, patterns can enhance design productivity as well as contribute to reduce errors in database design. Since most of GDB designers are professionals of areas such as cartography, architecture, and geography, they are usually not acquainted even with well-known database design practices. Especially for this type of designers, analysis patterns can be of great use.

Figure 1 shows an analysis pattern proposed in [3] to be reused in GDB design of urban applications. It represents an alternative for modeling traffic systems of urban areas. It is useful because all over the world, the main characteristics of traffic systems remain the same. Streets and street crossings compose the system. Streets are usually subdivided into street sections or segments in order to facilitate traffic management and control.

3 Selection of a DM Technique Suited for the Application

The selection of a data mining technique for any application depends on various factors such as both the volume and the format of the input data as well as the main goal of the mining process to be executed [12]. Some comparison criteria for data mining techniques are described in [13]. Relying on them, this section discusses the selection of a technique that can be used in order to identify analysis pattern candidates for GDB design from a set of database conceptual schemas.

On the basis of the comparison of data mining techniques shown in table 1, we selected the technique of *induction of association rules* [14]. The subsections below explain in more detail the selection criteria relying on which we made this decision.

3.1 Degree of Dependency of Human Specialists

The study of different data mining techniques showed that they all depend on decisions to be taken by human specialists on the application domain. Though, this dependency can be more or less strong. Since one of the goals of our research was to offer an alternative to pattern identification as much independent as possible from data engineering specialists, this comparison criterion became very important to us.

3.2 The Type of the Task

Data mining can be applied to support various types of tasks as, for instance, classification, clustering and identification of associations among items. In order to achieve success in a specific knowledge discovery process, one must choose a technique that supports exactly the type of task required by the application.

To acquire knowledge about analysis pattern candidates for GDB design, one must identify those sets of sub-schemas that occur most frequently in a set of GDB conceptual schemas. Thus, the application requires a task of identification of associations among items such as *classes*, *class attributes*, *associations* between *classes*, and *packages*.

3.3 Structure as Well as Semantics of the Output Data

There exist data mining techniques that can present the results of a same task (e.g., classification) in two different ways (e.g., as a list of rules and as a graphic). On the other hand, there are techniques that use the same presentation (e.g., list of rules) for the results of different tasks (e.g., classification and induction of association rules). In this latter case, result formats are similar but result interpretations are different.

For the application we are discussing, the selected technique should either provide its results directly in the form of GDB sub-schemas or at least allow the inference of sub-schemas in a post-processing phase.

3.4 Easy Interpretation of Results

Some data-mining techniques generate accurate results of great precision, but their interpretation requires more complex algorithms. We gave more importance to the criterion of easy interpretation of results than to that of result accuracy and precision.

3.5 Unlimited Volume of Input Data

The GIS community is trying to establish data model standards in order to facilitate the exchange of GDB schemas as well as geographic data through the Internet. So, one can infer that a large number of schemas will be available for processing in the near future. Therefore, the data mining technique to be selected should not impose any limit to the volume of input data.

3.6 Incrementally Computable Results

Since the number of existing as well as available GDB schemas is not known, one can expect that they will not be processed all at once. Thus, this application requires a data mining technique whose results can be incrementally estimated by repeatedly executing the same algorithm with different input data. For our application, this characteristic is much more important than performance in terms of processing time.

Table 1. DM techniques comparison based on the proposed selection criteria

Technique	Decision trees	Induction of association rules	Backprop. Neural network	Combinatorial Neural Model	Näive-Bayes
Criterion					
Degree of dependency of specialists	Intermediate	Low	Intermediate	Intermediate	Intermediate
Task	Classification/Associations identification	Associations identification	Classification	Classification/Associations identification	Classification/Associations identification
Output data representation	Graphics and rules	Rules	Mapping	Graphics and rules	Rules
Results comprehension facilities	Easy	Easy	Difficult	Easy	Easy
Incrementally computable results	No	Yes	No	Yes	No
Volume of input data	Great	Intermediate	Great	Intermediate	Great

4 The Induction of Association Rules and Its Typical Applications

The selected technique aims at identifying existent relationships among database representations of real world items [14]. It is typically applied to solve the problem of identifying a set of products with the highest probability of being bought together by customers in either a shopping store or a supermarket [14].

The input database is composed by a set of *buying transactions*, each one presenting a list of shopping items. An algorithm based on this technique processes data in order to identify existent relationships between groups of items. The main goal is to determine which items have high probability of being bought together.

Identified relations are expressed as rules. Each rule is composed by both an *antecedent* (A) and a *consequent* (C) part. A as well as C are sets of items and the rule $A \rightarrow C$ must be interpreted as A implies C . Associated with each rule, there are two statistical information that indicate its relevance: the *support* (S) and the *confidence* (CF). The prior indicates the percentage of all transactions that have the set (AUC) in the list of items. The latter expresses the quotient of S divided by the number of transactions that have at least A in their list.

5 Database Schemas as Input for Data Mining

In order to apply the data mining technique of *induction of association rules* to the problem of identifying analysis pattern candidates in a set of GDB conceptual schemas, the latter must be represented as a set of transactions. Each schema is represented as a transaction whose items are its sub-schemas. Depending on the aggregation level at which pattern candidates should be identified (e.g., sub-schema of both *classes* and *associations*, sub-schema of only one *class*) the process of dividing original database schemas should be applied recursively. The set of all items to be considered is composed by sub-schemas belonging to all transactions in the input database.

In order to restrict the complexity of what follows, in this paper we consider only sub-schemas with either only one *class*, or two *classes* associated with one another or one of the prior alternatives encapsulated by a *package*. We follow the definition as well as the notation established by the UML [9].

For our purpose, in order to create a transaction, a GDB schema must be recursively subdivided into sub-schemas up to the level of isolated classes. From all the resulting sub-schemas we need to consider only *packages*, isolated *classes*, and pairs of associated *classes* as the items of the transaction being created. Sub-schemas containing more than two associated classes can be inferred at post-processing by analyzing groups of rules with same items in *antecedent*.

By processing transaction items that are GDB sub-schemas, a data-mining algorithm for induction of *association rules* can produce rules such as the one shown in figure 2. That rule indicates that 60% of the database schemas processed contain both the *class Street* and the *class Street Section*. It also shows that, in the same 60% of cases, these *classes* are related to one another by an *aggregation association* from the prior to the latter class. The *confidence (CF)* points out that all transactions that present these *classes* also have the *aggregation* of them as another item.

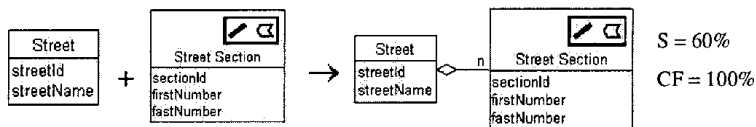


Fig. 2. Rule example

By executing a data mining algorithm based on the *induction of association rules* for an input database composed of GDB schemas as transactions and sub-schemas as their items, a set of rules is produced each one of them presenting database sub-schemas in both the *antecedent* and the *consequent*. Some examples of possible rules that can be generated from an input of database schemas are given below where *P* stands for *package*, *C* stands for *class*, $R(C_i, C_j)$ represents an *association* between *classes* C_i and C_j , and $P_k: C_i$ indicates that C_i belongs to the *package* P_k . As already explained above, *S* and *CF* are the rule's *support* and *confidence*, respectively.

$$\begin{aligned}
&Ci \rightarrow Pk: Ci; S \in CF \\
&Ci + Cj \rightarrow R(Ci, Cj); S \in CF \\
&Pk: Ci + Pk: Cj \rightarrow R(Ci, Cj); S \in CF \\
&Ci + Pk: Cj \rightarrow R(Ci, Cj); S \in CF \\
&Pk: Ci + Ph: Cj \rightarrow Ci; S \in CF \\
&R(Ci, Cj) \rightarrow R(Cj, Ck); S \in CF
\end{aligned}$$

6 Post-Processing of Associative Rules to Find Analysis Pattern Candidates

The simple rules generated by the data-mining algorithm are not capable of directly expressing all possible pattern candidates. Moreover, even for a small input database the algorithm can deliver a huge number of rules. Some of them lead to the same pattern candidate and another group indicate relationships that are not important for the inference of schema patterns. In order to both interpret and evaluate the out coming rules of the mining process, a post-processing step must be carried out.

Due to the complexity of possible schema structures, post-processing cannot rely only upon both *support* and *confidence* data associated with each rule. Therefore, a specific post-processing algorithm is being developed to execute the following tasks:

- selection of those associative rules that can really contribute to the identification of pattern candidates;
- inference of new rules on the basis of sets of related associative rules.

Since it is not yet clear that we have already identified all meaningful types of rules that can be used to infer analysis pattern candidates, we believe the post-processing algorithm might have to be extended in the future to capture more meaning from the data mining results.

6.1 Selecting Meaningful Rules

The first step to identify the meaningful rules is to understand output data. In order to do that, the principles used by algorithms that implement the chosen technique were studied. This study showed that many rules could be discarded because others have more useful information. Moreover, the schema decomposition in sub-schemas with different aggregation levels causes that a rule can be composed by more than one sub-schema composed by the same construct. For example, a rule like $Pi: Ck + Ck + Cj \rightarrow R(Ck, Cj)$ presents *class Ck* twice in the antecedent. This kind of rule can also be discarded.

Analyzing the results, it was possible to see that a sub set of output data is enough to identify the patterns candidates and that the remaining data can be consider trash, in this application. Some types of rules have already been identified as of interest. Though, due to space limitations only a few of them will be discussed in this paper. The graphic elements used in the examples below are introduced in figure 3.

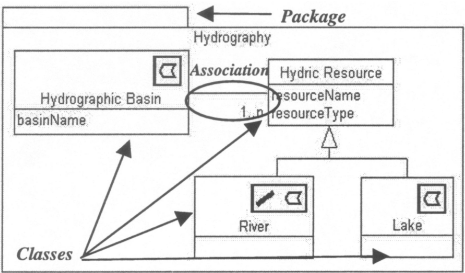


Fig. 3. Example of GDB schema

a) $C_j \rightarrow P_k: C_j; S \in CF$

There exists a probability that the schema $P_k: C_j$ (*consequent*) be an analysis pattern.

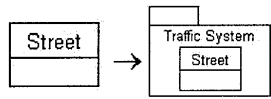


Fig. 4. Example of $C_j \rightarrow P_k: C_j$ rule

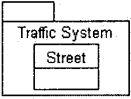


Fig. 5. Pattern inferred from rule showed in fig.4

b) $P_k: C_i + P_k: C_j \rightarrow R(C_i, C_j); S; CF.$

From this type of rule, it can be inferred that the *association* in the *consequent* is an element of *package* P_k shown in the *antecedent*. Furthermore, there is a probability that the sub-schema composed by both *package* and *association* be a pattern.

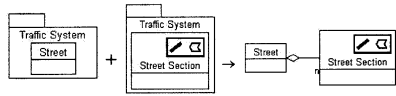


Fig. 6. Example of this type of rule

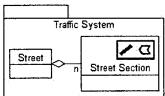


Fig. 7. Pattern inferred from rule in fig.6

6.2 Inferring New Rules from Combination of Existing Ones

Suppose a rule $R1$ shows a set of classes $CL=\{C_i \mid i=1,...,n\}$ as its *antecedent* and an association $R1(C_x, C_y)$, with $C_x \neq C_y$ and $C_x, C_y \in CL$ as its *consequent*. Suppose further that there exists another rule $R2$ with the same *antecedent* as well as the same S and CF as $R1$. If $R2$ has an association $R2(C_y, C_w)$ as its *consequent* and C_w is also an element of CL . Then, another rule $R3$ can be inferred with the following structure: $CL \rightarrow R(C_x, C_y)+R(C_y,C_w);S;CF$. Furthermore, there is a probability that $R3$'s *consequent* be an analysis pattern.

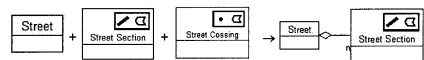


Fig. 8. Rule like $R1$



Fig. 9. Rule like $R2$

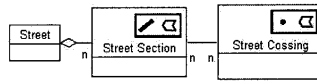


Fig. 10. Pattern inferred from the aggregation of rules showed in figs. 8 and 9

7 Validating the Proposal with a Prototype

In this section, we present the results of two KDD processes that were conducted according to our proposal.

The processing step was executed with the IBM Intelligent Miner for Data [15]. This product has been chosen due to its easy-to-use interface. A computer program was developed in Pascal to generate a set of GDB schemas, some of those composed of sub-schemas that are really existing analysis patterns. Another program was developed that can automatically discard some types of out coming association rules as part of the post-processing step. The rest of the post-processing tasks are still being done by hand.

In a first execution of the KDD process, 200 schemas were created and used as input data to the DM tool. Only classes and class associations composed each schema. 740 rules were generated as the result of data mining. By applying the post-processing techniques, more than half of them were discarded. At the end, the process has pointed out a set of pattern candidates that exactly matched the really existing ones present in the input database.

A second instance of the same KDD process was executed with an input database containing only 10 schemas. Though, these schemas were composed by packages besides classes and class associations. The data-mining tool identified more than 350.000 rules. During post-processing, only 15.245 of them have been considered of interest. Due to the large number of rules and the lack of implemented algorithms that automatize some of the already identified filtering procedures, we could not further reduce the set of rules that were considered meaningful.

8 Conclusions and Future Work

As the executed KDD processes demonstrated, the technique of *induction of associative rules* is appropriate to the identification of analysis patterns for database design. Furthermore, some developed post-processing practices have proven to be effective. Finally, the way the input data was organized, with schemas as transactions and sub-schemas as items, was adequate and allowed for the generation of meaningful rules for the application.

Though, there is yet a lot of work to do in order to come to a complete solution. Concerning the data preparation step, also other types of data model constructs as, for instance, *class attributes* should be represented as input data. In the case of the post-processing step, the following aspects should be worked in more details:

- selection of other types of meaningful rules;
- implementation of a ranking of rules according to some set of criteria defined by human specialist (e.g., patterns with a package are more important than those without it);
- presentation of pattern candidates identified during post-processing in a graphic way (e.g., using UML graphical notation).

References

1. BUSCHMANN, F. Pattern-oriented software architecture: a system of patterns. Chichester : John Wiley, 1996.
2. FOWLER, M. A Survey of Object-Oriented Analysis and Design Methods. 10th European Conference on Object-Oriented Programming, July 8-12, 1996. Linz, Áustria.
3. LISBOA F., J; IOCHPE, C.; BORGES, K. A. V. Padrões de Análise para reutilização de Esquemas de Dados de SIG em Aplicações de Gestão Urbana. XXVII Conferencia Latinoamericana de Informática. Anais... Venezuela, sep/2001.
4. BURROUGH, P. A.; MCDONNELL, R. A. Principles of Geographical Information Systems. Great Britain: Oxford university Press, 1997.
5. ROBERTSON, S.; STRUNCH, K. Reusing the products of analysis. Proceedings of International Workshop on Software Reusability. Lucca, Italy, 1993.
6. FAYYAD, U. M.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From Data Mining to Knowledge Discovery in Databases. AI Magazine, v.17, n.3, p.37-54, fall 1996.
7. HEUSER, C. A.. Projeto de Banco de Dados. Série Livros Didáticos N°4. Instituto de Informática da UFRGS. Sagra Luzzato. 4ª Edição. Porto Alegre, RS. 2001.
8. CHEN, P. P. The Entity-Relationship Model: Toward a Unified View of Data. ACM Transactions on Database Systems, v. 1, n.1, pp. 9-36, 1976.
9. ERIKSSON, H.-E.; PENKER, M. UML Toolkit. John Wiley & Sons: 1998.
10. BASSALO, G. H. M. Integração de modelos conceituais para sistemas de informação geográfica voltada à preparação de esquemas de bancos de dados geográficos para utilização em ferramentas de descoberta de conhecimento. Porto Alegre: PPGC da UFRGS, 2001 (Trabalho Individual).
11. ROCHA, L. V.; EDELWEISS, N; IOCHPE, C. GeoFrame-T: a Temporal Conceptual Framework for Data Modeling. In. INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHICAL INFORMATION SYSTEMS, 2001, Atlanta , Atlanta ACM Press, 2001.
12. GOEBEL, M.; GRUENWALD, L. A Survey of Data Mining and Knowledge Discovery Software Tools. In SIGKDD Explorations, v.1, n.1, pp.20-32, june/1999.

13. SILVA, C. M. S. Estudo de técnicas para suporte à geração de catálogos de padrões de análise para projeto de bancos de dados geográficos. Porto Alegre: PPGC da UFRGS, 2001 (Trabalho Individual).
14. AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules. In 20TH VLDB CONFERENCE, 20., 1994, Santiago. Proceedings. Hove: Morgan Kaufmann, 1994.
15. IBM. DB2 Intelligent Miner for Data. Available in <http://www-3.ibm.com/software/data/iminer/fordata/library.html>, accessed in 04/11/2002.

Lithology Recognition by Neural Network Ensembles

Rafael Valle dos Santos¹, Fredy Artola¹, Sérgio da Fontoura¹, Marley Vellasco²

¹GTEP / PUC-Rio □ Grupo de Tecnologia em Engenharia de Petróleo, Pontifícia
Universidade Católica do Rio de Janeiro

{rvsantos, fontoura, artola}@civ.puc-rio.br

²DEE / PUC-Rio □ ICA: Laboratório de Inteligência Computacional Aplicada
marley@ele.puc-rio.br

Abstract. This paper investigates the advantages of methods based on Neural Network Classifier Ensembles - sets of neural networks working in a cooperative way to achieve a consensus decision- in the solution of the lithology recognition problem, a common task found in the petroleum exploration field. Classifier ensembles (Committees) are developed here in two stages: first, by applying procedures for creating complementary networks, i.e., networks that are individually accurate but cause distinct misclassifications; second, by applying a combining method to those networks outputs. Among the procedures for creating committee members, the Driven Pattern Replication (*DPR*) was chosen for the experiments, along with the *ARC-X4* technique. With respect to the available combining methods, *Averaging* and *Fuzzy Integrals* were selected. All these choices were based on previous work in the field. This paper proves the effectiveness of applying ensembles in the recognition of geological facies and suggests algorithms that might be successfully applied to others classification problems.

1 Introduction

The solution of the lithology recognition problem has many potential utilities in the petroleum field. Among the main interests, lies the construction of vertical and lateral distribution of different kind of lithology, which leads to drilling path optimization.

J. H. Doveton [1] has formally presented a solution for this pattern classification problem by means of an Artificial Intelligence system, using Neural Networks. Since then, other researchers have also applied this approach [2], [3], [4], where single neural networks are used to classify lithologic facies. This work, on the other hand, addresses the problem by using the powerful approach of Neural Networks Classifier Ensembles.

Classifiers ensembles (Committees) have been investigated [5] in order to improve the performance of pattern recognition systems. They produce a consensus decision that is potentially more accurate than individual classifiers. This strategy is particularly useful when the available classifiers □ the committee members □ are

individually efficient and commit errors in different regions of the feature space. In such situations the classifiers are said to be complementary.

Many approaches for the design of committee members as well as for the combination strategy of the individual outputs have been proposed [6], [7], [8], [9], [10], [11]. In particular, [12] compares twelve combinations of procedures, providing some guidance for the design of *Multilayer Perceptron* (MLP) committees. According to the conclusions presented in [12], *Driven Pattern Replication* (DPR) seems to be the best choice among the methodologies for forming committee members when the ratio between the number of available training patterns and the dimension of the input space is large. When this condition is not satisfied, *ARC-x4* is the best option, particularly because it allows for more networks than pattern classes, which may compensate for the reduced sample size. With respect to the combining strategies, *Fuzzy Integrals* had the best performance for a large training set. For a small training set, *Averaging* achieved the best recognition rates.

Based on the exposed, the Driven Pattern Replication and the *ARC-X4* technique were chosen for the member creation experiments in this paper, while simple *Average* and Fuzzy Integrals were selected as the combining methods. As in [12] the neural networks used were MLPs running *Back-Propagation* (BP).

The next section describes the lithology recognition problem. Section 3 briefly presents the proposed procedures for the creation of complementary neural networks and the combination methods evaluated here. Section 4 describes the experiments in details, while section 5 brings the performance results obtained and a short discussion. The final conclusions are presented in section 6.

2 Lithology Recognition

The characterization of the vertical and lateral distribution of the diverse types of lithology in a petroleum field plays relevant role in the processes of thickness and rock quality mapping. Usually, these variations are estimated during wellbore drilling programs, directly from the analysis and interpretation of some types of well logs; they constitute what is called lithologic records. Normally, these records are used in the control and correlation during the operations of well logging.

This work presents some results concerning an automatic way to form lithology records, based on the construction of neural network ensemble systems, aiming to map an efficient relation between physical response (well logs) and lithology (Shale, Sand and so on). By means of this AI approach, it is possible to predict lithologic distributions over diverse locations in the petroleum field, locations where the only existing information comes from well logs. These results can be used in the preparation of spatial lithology distribution maps, especially of the lithologies associated with reservoir rocks levels.

3 MLP Ensembles

A committee of MLP neural networks is composed of independent classifiers that are designed to be later integrated in an ensemble. The committee is formed with the objective of achieving complementary performance in different regions of the feature space.

3.1 Forming Members with Driven Pattern Replication (DPR)

The number of available training patterns in a database, N , is given by

$$N = \sum_{i=1}^M n_i \quad (1)$$

where M is the number of pattern classes present in the application, and n_i (for $1 \leq i \leq M$) is the number of training patterns belonging to class i . As discussed in [12], the *Driven Pattern Replication (DPR)* method creates one expert neural network for each class in the training set; the M specialized neural networks are then combined. To build an expert network for class k , the n_k available training patterns belonging to that class are replicated by an integer factor $\gamma > 1$, so that the resulting training set will have a total of $N + (\gamma - 1) n_k$ patterns. Therefore, in each epoch, the training patterns not belonging to class k are presented to the network only once, while the patterns belonging to class k are presented γ times.

3.2 Forming Members with ARC-X4

The *ARC-x4* method, as suggested in [13], assigns sampling probabilities for each pattern of the original training set and then performs an iterative pattern selection algorithm. In each new iteration, a new training set is sampled and a new neural network is trained with the current selected patterns. The selection probabilities of misclassified patterns are increased for the next iteration, based on an empirical relationship that takes into account the number of times each pattern has been wrongly classified until the present iteration.

3.3 Combining Members by Average

When combining by average the output of the committee is simply given by the average of the corresponding outputs of its members.

3.4 Combining Members by Fuzzy Integrals

As in evidence theory [14], the combination of classifiers using Fuzzy Integrals relies on some measure relative to the pair classifier/class (e_k / c). In this technique such measures are called *Fuzzy Measures*.

A fuzzy measure is defined as a function that assigns a value in the $[0,1]$ interval to each crisp set of the universal set [15]. In the context of classifier combination, a

fuzzy measure expresses the level of competence of a classifier in assigning a pattern to a particular class.

A fuzzy integral [16] is a non-linear operation defined over the concept of fuzzy measure. In the framework of combining classifiers this can be explained as follows. Let $L = \{1, 2, \dots, M\}$ be the set of labels (classes) and $\varepsilon = \{e_1, e_2, \dots, e_K\}$ the set of available classifiers. A set of $K \times M$ fuzzy measures $g_c(e_i)$ is calculated, for c varying between 1 and M and i varying from 1 to K , denoting the competence of each classifier e_i in relation to each class c . These measures can be estimated by an expert or through an analysis of the training set (section 3.4.1 shows how competence may be computed). Fuzzy integrals are computed pattern by pattern, class by class, using mathematical relations considering competences and classifiers outputs. A pattern $\mathbf{x}$ will be assigned to the class with the highest value for the fuzzy integral; this class is selected as the response of the committee.

There are many interpretations for fuzzy integrals; they may be understood here as a methodology to rate the agreement between the response of an entity and its competence in doing so.

3.4.1 Estimating Competence

Competence of a classifier e_k in relation to a class i is estimated in this work by a ratio known as *local classification performance* [11], defined as:

$$g_i(e_k) = \frac{o_{ii}}{o_{ii} + \sum_{j, j \neq i} o_{ij} + \sum_{j, j \neq i} o_{ji}} \quad (2)$$

where o_{ij} is the number of patterns (observations) from class i assigned by the classifier e_k to the class j .

4 Experiments Description

Concerning well based lithology recognition, there are two main options for classification input data $\square$ *log information* or *seismic traces*. This work uses the first option, applying a *single hidden layer* MLP architecture for all its experiments.

Each input *tuple* (observation) corresponds to *four* log registers - GAMMA RAY, SONIC, DENSITY and RESISTIVITY $\square$ plus the observation's DEPTH, totalizing *five* attributes. The network outputs are *binary*, and equals the number of identified classes for the problem at hand. Each classifier is trained so that if output j is $\square$ on $\square$ the others are $\square$ off $\square$ and the observation is said to belong to class j .

4.1 The Original Data Set

The experiments were carried out over data from an offshore Brazilian well, located in its northeast coast. The raw data consists of 3330 observations, ranging from 130 to 3500m in depth. Each observation is assigned to one of eight classes, known as: SAND (1), CLAY (2), SANDSTONE (3), SHALE+LIMESTONE (4), SAND+LIMESTONE (5), SHALE (6), MARL (7) and SILTSTONE (8) (besides each class name is its numeric label).

4.2 Selected Subsets

The original data set was organized in two (non-exclusive) groups: *whole well* (all observations) and *reservoir* (observations with depth $\geq 2500\text{m}$). The whole well data set (3330 observations) has the following class distribution:

Table 1. *Whole well* class distribution

Label	1	2	3	4	5	6	7	8
#Observ.	57	44	626	119	471	1924	24	55
Total %	1.71	1.32	18.80	3.57	14.14	57.78	1.02	1.65

The reservoir class distribution is as follows:

Table 2. *Reservoir* class distribution

Label	3	4	6	7	8
#Observ.	563	52	276	14	55
Total %	58.65	5.42	28.75	1.46	5.73

It can be noticed that some classes are not present at the reservoir portion of the well. The practical implication here is that a 5-class recognition problem takes place, instead of an 8-class one. It must also be noticed that the observations are not equally distributed among the classes □ in both cases, some classes have much more assigned observations than the others do. Datasets with such structure can be called *non-stratified* datasets [17].

For each group of experiments, the following TESTING and TRAINING sets were chosen:

Table 3. TRAINING and TEST sets for *whole well*

Label	1	2	3	4	5	6	7	8
#Observ. (TRAIN)	38	29	230	79	230	230	23	37
Total	896							
#Observ. (TEST)	19	15	230	40	230	230	11	18
Total	793							

Table 4. TRAINING and TEST sets for *reservoir*

Label	3	4	6	7	8
#Observ. (TRAIN)	90	35	90	9	37
Total	261				
#Observ. (TEST)	90	17	90	5	18
Total	220				

In both cases, 2/3 of the total observations per classes were grouped for training, given the constraint that the rate between the number of patterns in the most populated class and the number of patterns in the less populated class, was, at most,

10 (that is why some classes are limited to 230 or 90 observations). The lasting 1/3 of the points were set aside for test purposes, following the same constraint. This constraint was applied to keep a balance between the numbers of points per class used for training and for testing, as some classes have a very small amount of assigned points. The ≤ 10 times constraint is a way to limit the "lack of harmony" between classes and, at the same time, a way to reduce computational load, as the number of processing patterns is reduced.

Following some guidelines presented in [12], in both case studies—*whole well* and *reservoir*—the number of hidden processors was set to 10. For each case, a proper "reference" network was created, i.e., a network that serves as a starting point from where all the experiments in the study case are carried out. The reference network's performance is used for comparison with each subsequent experiment.

As the reference networks are used as a starting point for all the experiments in the case studies, all *whole well* experiments have the same initial weights, the same happening with the *reservoir* experiments. This condition intends to provide fair comparison between achieved results. In this paper, the initial weights and biases were chosen using cross-validation over the training sets, which were split in two parts—50% for *estimation*, 50% for *validation* [18]. The number of epochs was fixed at 1000 (one thousand), a number that showed to be sufficient for convergence during the reference networks training.

As the methods implemented in this paper require changing the original training set, the training sets from Table 3 and Table 4 will be referred as *reference training sets*, respectively TR1a and TR1b.

Finally, it should be observed that in all the experiments, every input information were normalized to *standard scores* [19], where each observation equals itself less the sample average, divided by the sample standard deviation.

5 Results

The results obtained over the testing sets will be shown as *percentage average hits* (correct classifications) *per class* (AHC) and *percentage average hits considering all observations* (AHA). Classification *rejections* were not allowed.

5.1 Whole Well

The reference network created for this group of experiments, trained only with the reference training set TR1a, gave the following results:

Table 5. Results from TR1a

AHA	85,50
AHC	53,49

Before applying ensembles techniques, the size of the reference training set was equalized by replicating patterns according to class demand. This resulted in a second training set (TR2a):

Table 6. TR2a training set (TR1a Equalized)

Label	1	2	3	4	5	6	7	8
#Observ. (TRAIN)	228	232	230	237	230	230	230	222
Replication Factor	6	8	1	3	1	1	10	6
Total	1839							

After appropriated training, the following results were obtained:

Table 7. Results from TR2a

AHA	50,82
AHC	70,98

It is important to notice that with this single equalization step the AHA percentage has lowered around 30% and the AHC percentage has raised more than 17%. This is a sign of a possible existing trade-off between *global* and *local* classification, concerning this case study.

After the new training set was obtained, the ensemble methods were analyzed. The DPR application over TR2a, using, $\gamma=5$ (value chosen from the results in [12]) formed 8 new training sets. In this way, 8 neural networks were trained to be combined. The achieved DPR ensemble results are the following:

Table 8. Results from the DPR Ensemble (AVERAGING Combination)

AHA	74,02
AHC	80,12

Table 9. Results from the DPR Ensemble (FUZZY INTEGRALS Combination)

AHA	79,57
AHC	84,39

For the Fuzzy Integrals combination, the *competences* (section 3.4.1) were taken from the training set performance.

The ARC-X4 method applied to TR2a allows the user to form ensembles with as many networks as he or she wishes. As highlighted in [12], Fuzzy Integrals tends to be prohibitive in terms of processing time when the numbers of ensemble members increases beyond 15. For this reason, ensembles combined via ARC-X4 were only combined by averaging. Several numbers of networks were assessed, issuing the following results:

Table 10. Results from ARC-X4

#Nets	8	16	25	50	75	100
AHA	70,49	72,89	72,76	73,77	73,27	73,77
AHC	78,68	79,71	79,66	80,09	79,87	80,09

5.2 Reservoir

For simplicity, the classes for these experiments are relabeled as follows: class 3 = class 1; class 4 = class 2; class 6 = class 3; class 7 = class 4; class 8 = class 5.

The reference network created for the reservoir group of experiments, trained only with the reference training set TR1b, gave the following results:

Table 11. Results from TR1b

AHA	82,73
AHC	75,42

Again, before applying ensemble techniques, the reference training set was equalized, forming a second training set (TR2b):

Table 12. TR2b training set (TR1b Equalized)

Label	1(3)	2(4)	3(6)	4(7)	5(8)
#Observations (TRAIN)	90	105	90	90	74
Replication Factor	1	3	1	10	2
Total	449				

After appropriated training, the summarized results are:

Table 13. Results from TR2b

AHA	77,27
AHC	73,58

Unlike the previous case study, the AHA percentage has lowered but no improvement has been made over the AHC. The equalization step did not play its desired role, and this may be due to the reduced number of observations in the present case.

The DPR application over TR2b, using $\gamma=5$ (chosen from [12]), formed 5 new training sets. At this time, 5 neural networks were trained to be combined. The ensemble results achieved are the following:

Table 14. Results from the DPR Ensemble (AVERAGING Combination)

AHA	83,18
AHC	81,87

Table 15. Results from the DPR Ensemble (FUZZY INTEGRALS Combination)

AHA	82,27
AHC	84,09

Again, the competences were taken from the training set performance.

Following the same sequence from section 5.1, the next table shows the results for the ARC-X4 method applied to TR2b (using averaging as the combining method):

Table 16. Results from ARC-X4

#Nets	5	10	25	50	75	100
AHA	73,64	73,18	79,55	81,82	83,18	82,73
AHC	71,80	71,58	74,76	79,42	80,09	79,87

5.3 Discussion

In the first case study, the best AHC result was achieved by the DPR/Fuzzy Integrals pair (84,39%), which raised about 31% the reference AHC. The best global performance occurred for the reference network itself (85,50%).

For the second case study, the best AHC result was again achieved by the DPR/Fuzzy Integrals pair (84,09%), which raised about 8,5% the reference AHC. The best global performance occurred for the DPR/Averaging pair, along with the ARC-X4(75) /Averaging pair (83,18%), which actually did not differ much from the reference (82,73%).

As the test sets for both cases were *non-stratified*, i.e., there were some classes that had much more observations than others, AHC is the fairest percentage to rate each method. In this way, it can be said that for the first case study, the best result achieved was 84,39%, while for the second case the best result was 84,09%. Both results were obtained by a DPR / Fuzzy Integrals ensembles.

6 Conclusions

Concerning the lithology recognition problem, the results endorse that committees of network classifiers improve recognition performance when compared to schemes with a single network. This is specially true when the training sets are naturally *non-stratified*, i.e., some classes have much more observations than others, which is generally the case for geological facies datasets.

The experiments carried out were divided in two case studies □ *Whole well*, concerning all the available observations for a particular brazilian well, and *Reservoir*, concerning only the reservoir portion observations. The first case study dealt with a training sample of 896 observations, while the second dealt with a training sample of 261 observations.

In both cases, the best performance was achieved by ensembles using a DPR/Fuzzy Integrals association. This may be due to the fact that both training sets were *non-stratified*, which led to a perfect environment for *driven pattern replications*. The ARC-X4 method did not show good responses for the problem at hand, perhaps because of the same non-stratified environment.

Like in [12], the trade-off between global and local classification was once detected, as the methods with the best final AHC results had never the best AHA responses.

Although the experimental results obtained in this work may not provide a decisive assessment of the analysed methods, they can surely provide some guidance for future lithology recognition models.

References

- [1] Doveton, J. H., *Log Analysis of Subsurface Geology: Concepts and Computer Methods*, 1986, John Wiley & Sons, 1986.
- [2] Saggaf, M. M., I Marhoon, M., and Toksöz, M. N., "Seismic facies mapping by competitive neural networks" SEG/San Antonio 2001, San Antonio, 2001, CD-ROM.
- [3] Ford, D. A., Kelly, M. C., "Using Neural Networks to Predict Lithology from Well Logs" SEG/San Antonio 2001, San Antonio, 2001, CD-ROM.
- [4] Taner, M. T., Walls, J. D., Smith, M., Taylor, G., Carr, M. B., Dumas, D., "Reservoir Characterization by Calibration of Self-Organized Map Clusters" SEG/San Antonio 2001, San Antonio, 2001, CD-ROM.
- [5] J. Kittler, M. Hatef, R. P. W. Duin and J. Matas, "On combining classifiers" IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1998), pp. 226-239.
- [6] L. Breiman, "Combining predictors" in *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net System* Perspectives in Neural Computing, ed. A. J. C. Sharkey, Springer Verlag, 1999, pp. 31-51.
- [7] K. Hansen, and P. Salamon, "Neural network ensembles" IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (1990), pp. 993-1001.
- [8] Y. Liu and X. Yao, "Evolutionary ensembles with negative correlation learning" IEEE Transactions on Evolutionary Computation, 4 (2000), pp. 380-387.
- [9] D. Opitz and R. Maclin, "Popular ensemble methods: an empirical study" Journal of Artificial Intelligence Research 11 (1999), pp. 169-198.
- [10] dos Santos, R.O.V., Vellasco, M. M. B. R., Feitosa, R. Q., Simões, M., and Tanscheit, R., "An application of combined neural networks to remotely sensed images" Proceedings of the 9th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Pilsen, Czech Republic, 2001, pp. 87-92.
- [11] N. Ueda, "Optimal linear combination of neural networks for improving classification performance" IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000), pp. 207-215.
- [12] dos Santos, R.O.V., *Combining MLP Neural Networks in Classification Problems*, MSc dissertation, Electrical Engineering Department, PUC-Rio, 2001, 105 pages (in Portuguese).
- [13] L. Breiman, "Bias, variance and arcing classifiers", Technical Report 460, University of California, Berkeley, CA.
- [14] G. A. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, 1976.
- [15] G. Klir and T. Folger, *Fuzzy Sets, Uncertainty and Information*, Prentice-Hall, 1988.
- [16] M. Sugeno, "Fuzzy measures and fuzzy integrals: a survey" in *Fuzzy Automata and Decision Processes*, North Holland, Amsterdam, 1977, pp. 89-102.

- [17] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection", Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995, pp. 1137-1145.
- [18] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall, New Jersey, 1999.
- [19] S. K. Kachigan, Multivariate Statistical Analysis: a Conceptual Introduction, Radius Press, New York, 1991.

2-Opt Population Training for Minimization of Open Stack Problem

Alexandre César Muniz de Oliveira¹ and Luiz Antonio Nogueira Lorena²

¹ DEINF/UFMA, Av. dos Portugueses, 65.085-580, São Luís MA, Brasil
acmo@deinf.ufma.br

² LAC/INPE, Av. dos Astronautas, 12.201-970, São José dos Campos SP, Brasil
lorena@lac.inpe.br

Abstract. This paper describes an application of a Constructive Genetic Algorithm (CGA) to the Minimization Open Stack Problem (MOSP). The MOSP happens in a production system scenario, and consists of determining a sequence of cut patterns that minimizes the maximum number of opened stacks during the cutting process. The CGA has a number of new features compared to a traditional genetic algorithm, as a population of dynamic size composed of schemata and structures that is trained with respect to some problem specific heuristic. The application of CGA to MOSP uses a 2-Opt like heuristic to define the fitness functions and the mutation operator. Computational tests are presented using available instances taken from the literature.

1 Introduction

Minimization of Open Stacks Problem (MOSP) appears in a variety of industrial sequencing settings, where distinct patterns need to be cut and each one may contain a combination of piece types. For example, consider an industry of woodcut where pieces of different sizes are cut of big foils. Pieces of equal sizes are heaped in a single stack that stays open until the last piece of the same size is cut.

A MOSP consists of determining a sequence of cut patterns that minimizes the maximum number of opened stacks during the cutting process. Typically, this problem is due the limitations of physical space, so that the accumulation of stacks can cause the temporary need of removal of one or other stack, delaying the whole process.

This paper describes the application of a Constructive Genetic Algorithm (CGA) to MOSP. The CGA was recently proposed by Lorena and Furtado [1] and applied to Timetabling and Gate Matrix Layout Problems [2], [3], and differs from messy-GAs [4]-[6], basically, for evaluating schemata directly. It also has a number of new features compared to a traditional genetic algorithm. These include a population of dynamic size composed of schemata and structures, and the possibility of using heuristics in structure representation and in the fitness function definitions.

The CGA evolves a population, initially formed only by schemata, to a population of well-adapted structures (schemata instantiation) and schemata. Well-adapted structures are solutions, which cannot be improved using a specific problem heuristic. In this work, it is used a 2-Opt like heuristic to train the population of structures and schemata.

The CGA application can be divided in two phases, the constructive and the optimal: a) the constructive phase is used to build a population of quality solutions, composed of well-adapted schemata and structures, through operators as selection, recombination and specific heuristics; and b) the optimal phase is conducted simultaneously and transforms the optimization objectives of the original problem on an interval minimization problem that evaluates schemata and structures in a common way. In this paper, CGA is applied to MOSP and further conjectures are approached, as the performance of 2-Opt heuristic that is used to define the fitness functions and the mutation operator.

This paper is organized as follows. Section 2 presents theoretical aspects of MOSP. Section 3 presents the aspects of modeling for schema and structure representations and the consideration of the MOSP as a bi-objective optimization problem. Section 4 describes the some CGA operators, namely, selection, recombination and mutation. Section 4 shows computational results using instances taken from the literature.

2 Theoretical Issues of MOSP

The data for a MOSP are given by an $I \times J$ binary matrix P , representing patterns (rows) and pieces (columns), where $P_{ij}=1$, if pattern i contains piece j , and $P_{ij}=0$ otherwise. Each pattern is processed by your time, piece by piece, opening stacks (when a new piece type is cut) and closing stacks (when all items of a same that piece type were cut). The sequence of patterns being processed determines the number of stacks that stays open at same time.

Another binary matrix, here called of open stack matrix Q , can be used to calculate the maximum of open stacks for a certain pattern permutation. It is derived from the input matrix P , by following rules:

- $Q_{ij} = 1$ if there exists x and $y \mid \pi(x) \leq i \leq \pi(y)$ and $P_{xj} = P_{yj} = 1$;
- $Q_{ij} = 0$, otherwise; where $\pi(b)$ is the position of pattern b in the permutation.

Considering matrix Q , the maximum of open stacks (MOS) can be easily computed as:

$$MOS = \max_{i \in \{1, \dots, I\}} \sum_{j=1}^J Q_{ij} \quad (1)$$

The matrix Q clarifies the stacks that are open (consecutive-ones in the columns) along the cutting of patterns. The Table 1 shows an example of matrix P , your corresponding matrix Q , and MOS calculated for same example. The Q shows the consecutive-ones property [7] for columns being applied to P . In each column, one can see when a stack is open (first "1"), and when it is closed (last "1"). Between first and last "1" 's, the stack stays opened ("1" 's sequence).

The sum of "1" 's by rows, computes the number of open stacks when each pattern is processed. For the example of Table 1, when pattern 1 is cut there are 2 open stacks, then pattern 2 is cut opening 5 stacks, and so on. One can note that, at most, 5 stacks (MOS=5) are needed to process the permutation of patterns $\rho_0=\{1, 2, 3, 4, 5\}$.

Table 1. Example of matrices P and Q

pieces	1	2	3	4	5	6	7	8		1	2	3	4	5	6	7	8	Σ	
$P_{ij} =$	pattern 1	0	0	1	0	1	0	0	0	$Q_{ij} =$	0	0	1	0	1	0	0	0	2
	pattern 2	1	1	0	0	1	1	0	0		1	1	1	0	1	1	0	0	5
	pattern 3	1	0	1	0	0	0	0	0		1	0	1	0	1	0	0	0	3
	pattern 4	0	0	0	1	1	0	0	1		0	0	1	1	1	0	0	1	4
	pattern 5	0	0	1	0	0	0	1	0		0	0	1	0	0	0	1	0	2
$MOS = \max \{2, 5, 3, 4, 2\} =$																			5

In MOSP, the objective is to find out the optimal permutation of patterns that minimizes the MOS value. The Table 2 shows Q of the optimal permutation, $\rho_1=\{5, 3, 1, 2, 4\}$, for the example of Table 1.

Table 2. Optimal solution

pieces	1	2	3	4	5	6	7	8	Σ
pattern 5	0	0	1	0	0	0	1	0	2
pattern 3	1	0	1	0	0	0	0	0	2
pattern 1	1	0	1	0	1	0	0	0	3
pattern 2	1	1	0	0	1	1	0	0	4
pattern 4	0	0	0	1	1	0	0	1	3
$MOS = \max \{2, 2, 3, 4, 3\} =$									4

Other permutations with MOS=4 can exist, for example $\rho_2=\{2, 3, 1, 5, 4\}$, but ρ_1 holds an advantage to the others: the time that the stacks stay open (TOS). The TOS can be calculated by the sum of all "1" 's in Q. It comes from the distance, in the permutation, between the pattern that opens and the pattern that closes each stack. This would be a second objective in MOSP: to close the stacks as soon as possible, allowing that the customer's requests be available.

A more detailed introduction to MOSP can be found in Becceneri [8] and practical applications in [9]. With respect to complexity of MOSP, some works approaching the NP-hardness of MOSP have been published in the last decade. Andreatta et al. (1989) formulated the cutting sequencing problem as a minimum cut width problem on a hypergraph and showed that it is NP-Complete [10]. Recently, Linhares (2002) presented several aspects of MOSP and other related problems, like the GMLP (Gate Matrix Layout Problem), including the NP-hardness of them [11].

The GMLP is a known NP-hard problem and arises on VLSI design [12], [13]. Its goal is to arrange a set of circuit nodes (gates) in an optimal sequence, such that the layout area is minimized, i.e., it minimizes the number of tracks necessary to cover the gates interconnection. The relationship between MOSP and GMLP resides in the consecutive-ones property: a) a stack is open at moment that the first piece of a type is cut and stays open until the cut of the last piece of this same type, occupying a physical space during this time; at same way, b) a metal link is begun from the

leftmost gate requiring connection in a net and passes by all gates in circuit until the rightmost gate requiring connection, occupying a physical space inside of a track. Concerning input matrix P of MOSP, this property occurs in the columns, differently of GMLP that occurs in rows. Fig.1 shows an example of input matrix in GMLP.

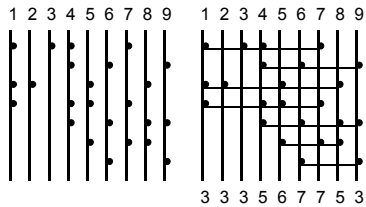


Fig. 1. Example of an input matrix in GMLP. a) Original gate matrix; b) Gate matrix derived by consecutive-ones property applied to rows and, in bottom, the number of track overlaps

3 CGA Modeling

Very simple structure and schema representations are implemented to the MOSP. A direct alphabet of symbols (natural numbers) represents the pattern permutation and each pattern is associated to a row of binary numbers, representing the piece type presence in each pattern. The symbol # is used to express indetermination (# - do not care) on schemata. Fig.2 shows the representation for the MOSP instance of Table 1, and examples of structures and a schema. The symbols ‘?’ mean there is no information in this row, once the pattern number is an indetermination ‘#’.

1	0 0 1 0 1 0 0 0	2	1 1 0 0 1 1 0 0	#	?????????
2	1 1 0 0 1 1 0 0	5	0 0 1 0 0 0 1 0	5	0 0 1 0 0 0 1 0
3	1 0 1 0 0 0 0 0	3	1 0 1 0 0 0 0 0	#	?????????
4	0 0 0 1 1 0 0 1	1	0 0 1 0 1 0 0 0	#	?????????
5	0 0 1 0 0 0 1 0	4	0 0 0 1 1 0 0 1	4	0 0 0 1 1 0 0 1
$s_i =$	(1 2 3 4 5)	$s_j =$	(2 5 3 1 4)	$s_k =$	(# 5 # # 4)

Fig. 2. Examples of structures (S_i and S_j) and schema (S_k)

To attain the objective of evaluating schemata and structures in a common way, two fitness functions are defined on the space X of all schemata and structures that can be obtained this representation. The MOSP is modeled as the following Bi-objective Optimization Problem (BOP):

$$\begin{aligned} &Min \quad \{g(s_k) - f(s_k)\} \\ &Max \quad g(s_k) \\ &Subject\ to\ g(s_k) \geq f(s_k)\ \forall\ s_k \in X \end{aligned}$$

(2)

Function g is the fitness function that reflects the total cost of a given permutation of patterns. To increase the fitness differentiation among the individuals of the population, it is used in the formulation that considers the MOS minimization as

primary objective and TOS minimization as a secondary one. Therefore, it is defined as $g(s_k) = I \cdot J \cdot MOS(s_k) + TOS(s_k)$, or

$$g(s_k) = I \cdot J \cdot \max_{i \in \{1, \dots, I\}} \sum_{j=1}^J Q_{ij} + \sum_{i=1}^I \sum_{j=1}^J Q_{ij} \quad (3)$$

where the I-J product is a weight to reinforce the part of the objective considering the maximum number of open stacks and to make it proportional to the second part of the objective concerning the time of open stacks. If s_k is schema, the non-defined columns (# label) are bypassed. It seems as these columns do not exist and the Q matrix used to compute $g(s_k)$ contains only columns with information. In the example of Fig 2, the MOS is $\max\{?, 2, ?, ?, 3\} = 3$ and the TOS is $\sum\{0+2+0+0+3\} = 5$.

The other fitness function f is defined to drive the evolutionary process to a population trained by a heuristic. The chosen heuristic is the 2-Opt neighborhood. Thus, function f is defined by:

$$f(s_k) = g(s_v), s_v \in \{s_1, s_2, \dots, s_V\} \subseteq \Phi^{2-Opt}, g(s_v) \leq g(s_k) \quad (4)$$

where Φ^{2-Opt} is a 2-Opt neighborhood of structure or schema s_k .

By definition, f and g are applied to structures and schemata, just differing in the amount of information and consequently in the values associated to them. More information means larger values. In this way, the g maximization objective in *BOP* drives the constructive phase of the CGA aiming that schemata will be filled up to structures.

4 Evolution Process

The *BOP* defined above is not directly considered as the set X is not completely available. Alternatively is considered an evolution process to attain the objectives (*interval minimization* and *g maximization*) of the *BOP*. At the beginning of the process, two *expected values* are given to these objectives:

- g maximization: a non-negative real number $g_{max} > \max_{s \in X} \{g(s)\}$ that is an upper bound on the objective value;
- interval minimization: an interval length $d \cdot g_{max}$, obtained from g_{max} considering a real number $0 < d \leq 1$.

The evolution process is then conducted considering an adaptive rejection threshold, which contemplates both objectives in *BOP*. Given a parameter $\alpha \geq 0$, the expression

$$g(s_k) - f(s_k) \geq d \cdot g_{max} - \alpha \cdot d \cdot [g_{max} - g(s_k)] \quad (5)$$

presents a condition for rejection from the current population of a schema or structure s_k . The right hand side of (5) is the threshold, composed of the expected value to the interval minimization $d \cdot g_{max}$, and the measure $g_{max} - g(s_k)$, that shows the difference of $g(s_k)$ and g_{max} evaluations.

Expression (5) can be examined varying the value of α . For $\alpha=0$, both schemata and structures are evaluated by the difference $g-f$ (first objective of *BOP*). When α

increases, schemata are most penalized than structures by the difference $g_{max} - g$ (second objective of *BOP*).

Parameter α is related to time in the evolution process. Considering that the good schemata need to be preserved for recombination, the *evolution parameter* α starts from 0, and then increases slowly, in small time intervals, from generation to generation. The population at the evolution time α , denoted by P_α , is dynamic in size accordingly the value of the adaptive parameter α , and can be emptied during the process. The parameter α is now isolated in expression (6), thus yielding the following expression and corresponding rank to s_k :

$$\alpha \geq \frac{d \cdot g_{max} - [g(s_k) - f(s_k)]}{d[g_{max} - g(s_k)]} = \delta(s_k). \quad (6)$$

At the time they are created, structures and/or schemata receive their corresponding *rank value* $\delta(s_k)$. These ranks are compared with the current evolution parameter α . The higher the value of $\delta(s_k)$, and better is the structure or schema to the *BOP*, and they also have more surviving and recombination time.

For the MOSP, the overall bound g_{max} is obtained at the beginning of the CGA application, by generating a random structure and making g_{max} receive the g evaluation for that structure. In order to ensure that g_{max} is always an upper bound, after recombination, each new structure generated s_{new} is rejected if $g_{max} \leq g(s_{new})$.

4.1 Selection and Recombination

The structures and schemata in population P_α are maintained in ascending order, according to the key:

$$\Delta(s_k) = (1 + \frac{g(s_k) - f(s_k)}{g(s_k)}) \cdot \frac{1}{\eta} \quad (7)$$

where η is the number of genes containing information (not #). Thus, well-adapted individuals (small $g(s_k) - f(s_k)$) with more genetic information (higher η) appear in first order places on the population.

Two structures and/or schemata are selected for recombination. The first is called the *base* (s_{base}) and is randomly selected out of the first positions in P_α and in general it is a good structure or a good schema. The second structure or schema is called the *guide* (s_{guide}) and is randomly selected out of the total population. The objective of the s_{guide} selection is the conduction of a guided modification on s_{base} .

In the recombination operation, the current labels in corresponding positions are compared. Let s_{new} be the new structure or schema (offspring) after recombination. Structure or schema s_{new} is obtained by applying only one of the following operations:

{ Recombination }	
For i from 1 to individual length	
1) if $s_{BASE}(i) = \#$ and $s_{GUIDE}(i) = \#$ set $s_{NEW}(i) = \#$	2) if $s_{BASE}(i) = \#$ and $s_{GUIDE}(i) < \#$ if $s_{GUIDE}(i)$ is not in s_{NEW} set $s_{NEW}(i) = s_{GUIDE}(i)$ else set $s_{NEW}(i) = \#$
3) if $s_{BASE}(i) < \#$ and $s_{GUIDE}(i) = \#$ if $s_{BASE}(i)$ is not in s_{NEW} set $s_{NEW}(i) = s_{BASE}(i)$ else set $s_{NEW}(i) = \#$	4) if $s_{BASE}(i) < \#$ and $s_{GUIDE}(i) < \#$ if $s_{BASE}(i)$ is not in s_{NEW} set $s_{NEW}(i) = s_{BASE}(i)$ else if $s_{GUIDE}(i)$ is not in s_{NEW} set $s_{NEW}(i) = s_{GUIDE}(i)$ else set $s_{NEW}(i) = \#$

Observe that s_{base} is a privileged individual to compose s_{new} , but it is not totally predominant. There is a small probability of the s_{guide} gene information to be used instead of s_{base} one. More detailed information about CGA features to permutation problems can be found in [3].

4.2 The 2-Opt Heuristic

The 2-Opt like heuristic is used to train the population by the fitness function f . The well-adapted individuals have better ranking and are maintained in the population for more generations. Another application to 2-Opt is to run a local search mutation that is always applied to structures (not to schemata).

To avoid the increasing of computational efforts, only a constant number of neighbors around the structure is inspected, looking for the best. The neighbors are generated by all the 2-move changes in a constant length part of the structure. An initial position is chosen at random and an iterative process starts from it, inspecting all possible 2-move changes in the structure until a maximum length previously established. Each 2-move generates a neighbor structure that will be evaluated and the best one will be hold on.

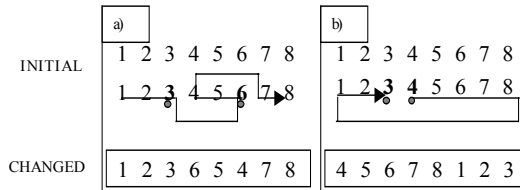


Fig. 3. Examples of one-move in 2-Opt neighborhood: a) non-consecutive reference points change; b) consecutive reference

The example of 2-move change is showed in Fig.3. The marks in positions of the structures mean reference points to be changed. Non-consecutive references cause the first change type, as showed in Fig.3a. Consecutive points cause the second change type in Fig.3b. For example, inspecting 4 neighbors, from first position in Fig.3, generates 6 pairs of reference points: $\{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$, i.e., $0.5 * nw * (nw - 1)$ pairs, where nw is the neighborhood width and, together with other parameter settings, will be described in next section.

5 Computational Tests

The CGA for MOSP was coded in *ANSI C* and it was run on Intel Pentium II (266Mhz) hardware. For the computational tests, some CGA parameters were adjusted. The d parameter was set to 0.15 (usually between 0.10 and 0.20 values, for other applications [1], [2]). This configures the interval $d \cdot g_{max}$, establishing the survival time of each individual, once the expected δ values are proportional to this interval. The ε was set to 0.001 and also contributes to the higher survival time of each individual in P_{α} . These parameters avoid the premature termination with an empty population.

Each schema of the initial population received 50% of # genes (indetermination percentage), and 20% of population (the first individuals ranked by expression 7) were considered base individuals for base-guide selection, determining a small degree of diversification in selection process.

Local search mutation rate was fixed in 100%, which means a constant improvement of individuals. The number of individuals initially generated was proportional to problem length (at least the number of patterns). Other important parameter to be tuned is the neighborhood width (nw) to each local search mutation. After some simulations the better results arise for $nw = 20$. The ideal situation would be to use greater values for nw , but this would turn the mutation very slow.

The CGA was initially applied to 300 instances taken from the paper of Fraggioli and Bentivoglio [14]. These instances are grouped by number of patterns (10,15,20, 25,30,40). Each one of these pattern groups has five piece type subgroups (10,20,30,40,50) and each piece type subgroup has ten instances with different solutions.

In Fraggioli and Bentivoglio's work are presented six solution methods, and the three best are: a) an implicit enumeration method (OPT) that enhances the implicit search procedure of Yuen and Richardson [15], and is used to verify the optimality of the found solutions; b) a tabu search method (TS) based on an optimized move selection process; and c) a generalized local search method (GLS) that works by employing multiple applications of a simplified tabu search that only accepts improving moves. In this work, besides the three previously mentioned methods (OPT, TS, GLS), another two solution methods are included for comparison with CGA: d) the 2-Opt local search heuristic (2-Opt); and e) the collective method (COL) proposed recently by Linhares [11].

The 2-Opt method employs the same heuristic used to train the population in CGA. Initially, a static population of 20 structures is randomly generated, and 2-Opt is applied for each one of them until no more improvement be found. The best solution is held. The 2-Opt parameter nw (neighborhood width) is set to maximum size, i.e., the number of patterns of the problem. This exhaustive local search demands a significant computational effort and the running time for large problems (above 100 patterns) is prohibitive.

The COL method explores distance measures among permutations to drive the search of an algorithm similar to the simulated annealing, where the moves in the search space are based on exchange in pattern positions.

The Table 3 shows the solution averages obtained by OPT, COL, TS, GLS, CGA and 2-Opt for each instance group. Only the MOS minimization is compared because the TOS is not considered on the other works. The columns I and J refer to numbers of patterns and piece types of each instance group, respectively. The entries emphasized in gray are better than the reported OPT optimum values. Observe that although claimed to be optimal in [14] some entries in OPT column (instances 15x30, 15x40, 15x50, and 40x40) have higher values than, at least, one of these methods: COL, CGA and 2-Opt. This may appear to be a contradiction but these are the new best bounds.

Considering these new best-known solutions, the CGA found the best overall average of solutions for the instance groups, i.e., 100% of success. The COL appears with the second best performance, achieving the best average in 87% of instance groups (26 of 30), followed by 2-Opt (73% or 22 of 30), TS (40% or 12 of 30) and GLS (33% or 10 of 30) of success rate, respectively.

Table 3. Solution averages obtained by OPT, COL, TS, GLS, CGA and 2-Opt

I	J	OPT	COL	TS	GLS	CGA	2-Opt	I	J	OPT	COL	TS	GLS	CGA	2-Opt
10	10	5.5	5.5	5.5	5.5	5.5	5.5	25	10	8.0	8.0	8.0	8.0	8.0	8.0
-	20	6.2	6.2	6.2	6.2	6.2	6.2	-	20	9.8	9.8	9.8	9.9	9.8	9.8
-	30	6.1	6.1	6.1	6.2	6.1	6.1	-	30	10.5	10.6	10.7	10.6	10.5	10.5
-	40	7.7	7.7	7.7	7.7	7.7	7.7	-	40	10.4	10.4	10.7	10.6	10.4	10.5
-	50	8.2	8.2	8.2	8.2	8.2	8.2	-	50	10.0	10.0	10.1	10.2	10.0	10.0
15	10	6.6	6.6	6.6	6.6	6.6	6.6	30	10	7.8	7.8	7.8	7.8	7.8	7.8
-	20	7.2	7.2	7.2	7.5	7.2	7.2	-	20	11.1	11.2	11.2	11.2	11.1	11.1
-	30	7.4	7.3	7.4	7.6	7.3	7.6	-	30	12.2	12.2	12.6	12.2	12.2	12.2
-	40	7.3	7.2	7.3	7.4	7.2	7.3	-	40	12.1	12.1	12.6	12.4	12.1	12.2
-	50	7.6	7.4	7.6	7.6	7.4	7.4	-	50	11.2	11.2	12.0	11.8	11.2	11.2
20	10	7.5	7.5	7.7	7.5	7.5	7.5	40	10	8.4	8.4	8.4	8.4	8.4	8.4
-	20	8.5	8.5	8.7	8.6	8.5	8.5	-	20	13.0	13.0	13.1	13.1	13.0	13.0
-	30	8.8	9.0	9.2	8.9	8.8	8.9	-	30	14.5	14.5	14.7	14.6	14.5	14.5
-	40	8.6	8.6	8.6	8.7	8.6	8.6	-	40	15.0	15.0	15.3	15.3	14.9	15.0
-	50	7.9	7.9	8.0	8.2	7.9	8.0	-	50	14.6	14.6	15.3	14.9	14.6	14.9

The comparison between CGA and 2-Opt procedure is meaningful, once CGA employs 2-Opt heuristic for fitness definition and local search mutation. The difference between them is the genetic constructive process that exists behind CGA. Selection, recombination and ranking contribute to the construction of well-adapted structures from an initial population of schemata. All these features seem become CGA more robust than other non-population approaches, like COL and 2-Opt.

One can also suppose that 2-Opt could achieve all best solution averages after several trials. However, Table 3 has showed that the 2-Opt is not to be able to find all best solutions. Besides, 2-Opt turns to be prohibitive for large-scale instances (above 100 patterns). This can be best verified by the following experiment. The 2-Opt was applied to an instance of another problem type, the GMLP (Gate Matrix Layout Problem), already mentioned in this paper (see section 2). There is a well-known GMLP instance (namely w4) with 141 gates and 202 nets. This is equivalent to a MOSP instance of 141 patterns of 202 piece types. The 2-Opt procedure was run 10 times for w4 instance and did not achieve the best-known solution (27 tracks). The solution 29 was found after 198 minutes. The CGA reach the 27 tracks in 30% of trials and 87 minutes (average time)[3].

6 Conclusion

This work describes an application of the Constructive Genetic Algorithm (CGA) to Minimization of Open Stack Problems (MOSP). The CGA adapted to work with MOSP uses a 2-Opt heuristic as local search mutation and on definition of the two fitness functions (f and g). The algorithm constructs a population of well-adapted structures trained by the 2-Opt heuristic.

Regarding the computational tests, the CGA reached all the best-known results for instances taken from the literature and presented the best results in comparison to other methods. It also appears to be more robust than the standalone application of procedure 2-Opt.

Acknowledgements

The first author acknowledges the Programa Institucional de Capacitação Docente e Técnica -PICDT/CAPEs for financial support. The second author acknowledges Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (proc. 300837/89-5) and Fundação para o Amparo a Pesquisa no Estado de S. Paulo - FAPESP (proc. 99/06954-7) for partial financial support.

References

- [1] Lorena, L.A.N., Furtado, J.C.: Constructive genetic algorithm for clustering problems, *Evolutionary Computation*, Vol. 9(3) (2001) 309-327
- [2] Ribeiro Filho, G., Lorena, L.A.N.: A Constructive Evolutionary Approach to School Timetabling, In: *Applications of Evolutionary Computing*, Boers, E.J.W., Gottlieb, J., Lanzi, P.L., Smith, R.E., Cagnoni, S., Hart, E., Raidl, G.R., Tijink, H., (eds.): *Lecture Notes in Computer Science*, Vol. 2037, Springer-Verlag (2001) 130-139
- [3] Oliveira, A.C.M., Lorena, L.A.N.: A Constructive Genetic Algorithm for Gate Matrix Layout Problems. Accepted to *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems* (2002)
- [4] Goldberg, D.E., Korb, B., Deb, K.: Messy genetic algorithms: motivation, analysis, and first results, *Complex Systems*, Vol. 3 (1989) 493-530
- [5] Goldberg, D.E., Deb, K., Kargupta, H., Harik, G.: Rapid, accurate optimization of difficult problems using fast messy genetic algorithms, *IlliGAL Report No. 93004*, Illinois Genetic Algorithms Laboratory, Department of General Engineering, University of Illinois, Urbana (1993)
- [6] Kargupta, H.: Search, polynomial complexity, and the fast messy genetic algorithm, Ph.D. thesis, *IlliGAL Report No. 95008*, Illinois Genetic Algorithms Laboratory, Department of General Engineering, University of Illinois, Urbana (1995)
- [7] Golumbic, M.: *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York (1980)

- [8] Becceneri, J.C.: O problema de sequenciamento de padrões para minimização do número máximo de pilhas abertas em ambientes de corte industriais. Doctoral Thesis, Instituto Tecnológico de Aeronáutica, São José dos Campos, Brazil (1999)
- [9] Yanasse, H.H.: Minimization of open orders-polynomial algorithms for some special cases. *Pesquisa Operacional*, Vol. 16 (1996) 1-26
- [10] Andreatta, G., Basso, A., Caumo, A., Deserti, L.: Un problema min cutwidth generalizzato e sue applicazionni ad un FMS. *Atti delle giornate di lavoro AIRO*, (1989) 1-17
- [11] Linhares, A.: Industrial Pattern Sequencing Problems: Some Complexity Results And New Local Search Models. Doctoral Thesis, Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, Brazil (2002)
- [12] Möhring, R.: Graph problems related to gate matrix layout and PLA folding, *Computing*, Vol. 7 (1990) 17-51
- [13] Kashiwabara, T., Fujisawa, T.: NP-Completeness of the problem of finding a minimum clique number interval graph containing a given graph as a subgraph, In: *Proc. Symposium of Circuits and Systems* (1979)
- [14] Fraggioli, E., Bentivoglio, C. A.: Heuristic and exact methods for the cutting sequencing problem. *European Journal of Operational Research*, Vol. 110 (1998) 564-575
- [15] Yuen, B.J., Richardson, K.V.: Establishing the optimality of sequencing heuristics for cutting stock problems. *European Journal of Operational Research*, Vol. 84 (1995) 590-598

Grammar-Guided Genetic Programming and Automatically Defined Functions

Ernesto Rodrigues¹ and Aurora Pozo²

¹ Fundação de Estudos Sociais do Paraná, Departamento de Informática,
Rua General Carneiro, 216, Centro, 80060-150, Curitiba, Paraná, Brazil
`ernesto@fesppr.br`

² Universidade Federal do Paraná, Departamento de Informática,
Caixa Postal 19081, Centro Politécnico, 81531-990, Curitiba, Paraná, Brazil
`aurora@inf.ufpr.br`

Abstract. Genetic Programming (GP) is a powerful software induction technique that has been recently applied for solving a wide variety of problems. Attempts to extend GP have focussed on applying type restrictions to the language to control genetic operators and to ensure that only valid programs are created. In this sense, the use of context free grammar (CFG) was proposed. This work studies the use of a CFG to define the structure of the initial population and direct crossover and mutation operators. Chameleon, a Grammar-Guided Genetic Programming system (GGGP) is also presented. On a suite of experiments composed of even-parity problems, the performance of Chameleon is compared to traditional GP. Furthermore, the automatic discovery of sub-functions, one of the most important research areas in GP, is also explored. We describe how to use ADFs with GGGP and, using Chameleon, we demonstrate that GGGP has similar results to Koza's Automatically Defined Functions (ADF) approach.

1 Introduction

Genetic Programming (GP) is the automatic generation of computer programs, using a process analogous to biological evolution [1]. This technique exploits the process of natural selection based on a fitness measure to breed a population of trial solutions that improves over time.

Due to the many possibilities of application using GP, it has been utilized in the most diverse areas of knowledge, such as: biotechnology, electrical engineering, art, financial market, image processing, pattern recognition, natural language and many others [3].

Previous works have presented GP using grammars mainly to overcome the closure problem, the generation and preservation of valid programs. In particular, the grammar allows the user to bias the initial GP structures, and automatically ensure that typing and syntax are maintained by manipulating the explicit derivation tree from the grammar [4]. Further, Gruau formally proved that using syntactical constraints is possible to reduce the size of search space [9].

One of the most widely recognized extensions to GP is the use of Automatically Defined Functions (ADFs) [2]. In the ADF approach, each program in the population contains an expression, the result-producing branch, and definitions of one or more functions, which may be invoked by the result-producing branch. The result-producing branch is evaluated to produce the fitness of the program.

Recently, O'Neil described the use of grammar based function definition as an approach to ADF in Grammatical Evolution [5]. Grammatical Evolution is a context free grammar (CFG) based genetic algorithm that evolves programs using linear genomes. As far as we know, the use of ADF in a Grammar-Guided Genetic Programming has not yet been studied.

This paper focuses on the behavior of the Grammar-Guided Genetic Programming (GGGP) [6]. Chameleon, a GGGP tool, is also introduced. In a first stage, it is used to solve the Fourier Series and even-parity problems. Then, due to its interesting results, Chameleon is extended to allow ADF evolution. The text is organized as follows. Next section reviews GGGP concepts. Section 3 introduces ADFs and grammars. The Chameleon tool is presented in Section 4. Section 5 analyzes the experiments conducted. Finally, the last section concludes the paper and establishes goals for future studies and improvements to Chameleon.

2 Genetic Programming and Grammars

The purpose of this section is to present basic concepts of GGGP. It starts with an overview of the GP algorithm [1].

2.1 Genetic Programming Algorithm Overview

The GP algorithm can be summarized as follows:

- Create a random population of programs;
- Perform the following steps until the termination criterion is satisfied:
 - Evaluate the fitness of each program;
 - Select programs from the current population according to fitness;
 - Apply genetic operators, such as: reproduction, crossover and mutation. Reproduction copies the program to the next generation. Crossover combines characteristics of two individuals to create two new ones. Mutation randomly changes portions of a program;
- End of algorithm.

Each run of this loop represents a new generation of computer programs that substitutes the previous one. The evolution is halted when an optimal solution is achieved or some specified maximum number of generations has passed.

2.2 Context Free Grammars

A context free grammar (CFG) is a four-tuple $\{N, \Sigma, P, S\}$, where N is the non-terminal alphabet, Σ is the terminal alphabet, P is the set of productions and S is the start symbol. The productions have the format $x \Rightarrow y$ where $x \in N$, $y \in \{\Sigma \cup N\}^*$. When there are a number of productions that can be applied to one particular $x \in N$, the choice is delimited with the disjunctive symbol '|', as $x \Rightarrow y \mid z$.

The productions specify how should the non-terminal symbol be rewritten into one of their derivations until the expression contains terminal symbols only. For example, a CFG for generating one variable simple arithmetic expression could be:

$$\begin{aligned} S &\Rightarrow \langle \text{exp} \rangle \\ \langle \text{exp} \rangle &\Rightarrow \langle \text{var} \rangle \mid \langle \text{exp} \rangle \langle \text{op} \rangle \langle \text{exp} \rangle \\ \langle \text{op} \rangle &\Rightarrow + \mid - \mid * \mid / \\ \langle \text{var} \rangle &\Rightarrow x \end{aligned}$$

Figure 1 shows an example of the expression $x * x + x$ that can be produced by this CFG.

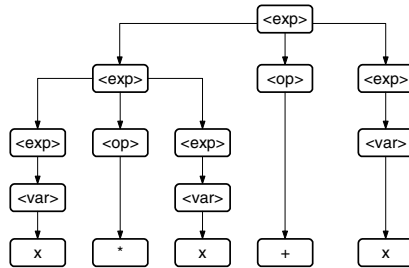


Fig. 1. A derivation tree for $x * x + x$

2.3 Initial Population

The initialization of a derivation tree is straightforward. From the start symbol S , the tree is built up by applying productions to leaf nodes until it is impossible to apply any further rules. A maximum tree depth is usually specified to avoid very deep trees. Unfortunately, this simple method hardly produces a sufficiently diversified initial population. To overcome this initialization problem within GGGP [6], we use a variant of the grow method.

The following steps are done to create initial population using the CFG $\{N, \Sigma, P, S\}$ which is restricted by a minimum and a maximum depths:

1. For each production $A \Rightarrow \alpha$ where $A \in N$ and $\alpha \in \{\Sigma \cup N\}^*$, compute the minimum number of derivations steps to create only terminals.
2. Select the start symbol S . Label this as the current non-terminal A .
3. Randomly select a production $p \in P$ of the form $A \Rightarrow \alpha$ with derivation steps to Σ^* between minimum and maximum depths.
4. For each non-terminal $B \in \alpha$, label B as the current non-terminal. Decrease minimum and maximum depths by one and repeat steps 3 and 4.

2.4 Selection Strategies

There are two main selection strategies used in GP: *fitness proportionate* and *tournament* [3]. In the fitness proportionate selection, programs are selected randomly with probability proportional to its fitness. In the tournament selection, a fixed number of programs are taken randomly from the population and the program with the best fitness within this group is chosen.

2.5 Genetic Operators

In GGGP, crossover and mutation operators must produce a legal offspring according to the CFG. In crossover, we use *point-typing* to preserve program syntax [4]. A pair of programs is selected from the current population based on the selection strategy. A crossover point with the non-terminal A is then determined on the first program. If the second program has no node with A , crossover is rejected. Otherwise, a node with A is randomly selected in the second program and the sub-trees are swapped.

In the mutation operation, only one program is selected. A mutation point is then determined, and a new sub-tree replaces the sub-tree rooted at that point. The new sub-tree is created using the same generation process as in the initial population.

The maximum tree depth parameter is used to indicate the deepest derivation tree that may be allowed.

3 Automatically Defined Functions and Grammars

This section explains how to use GGGP and ADF, using an approach to ADF as defined by Koza [2]. A candidate solution is composed of a result-producing branch (rpb) and one or more functions called ADFs. Distinct primitive sets are designated for the rpb and each ADF. This facilitates hierarchical sequence, provides benefits of parameterization [7] and imposes a syntactic constraint that some primitives cannot be used if they are not within the correct program scope. In the case of GGGP with ADF, we use a separate CFG for each branch to allow this.

To induce a program using GGGP and ADFs, we have to determine the CFG and the fitness cases. For example, the CFG for the even- k -parity problem with two ADFs is presented below. The rpb branch consists of: a complete set of

four primitive Boolean functions and the two ADFs: {AND, OR, NAND, NOR, ADF0, ADF1}; and the k Boolean arguments: $\{D_0, D_1, D_2, \dots, D_{k-1}\}$. In this way, the CFG for the rpb branch in the even-3-parity is:

$$\begin{aligned} S &\Rightarrow \langle \text{exp} \rangle \\ \langle \text{exp} \rangle &\Rightarrow \langle \text{var} \rangle \mid \langle \text{bf} \rangle (\langle \text{exp} \rangle, \langle \text{exp} \rangle) \\ \langle \text{bf} \rangle &\Rightarrow \text{AND} \mid \text{OR} \mid \text{NAND} \mid \text{NOR} \mid \text{ADF0} \mid \text{ADF1} \\ \langle \text{var} \rangle &\Rightarrow D0 \mid D1 \mid D2 \end{aligned}$$

The ADF0 branch contains only the four primitive functions and two arguments named ARG0 and ARG1. The CFG for the ADF0 branch in the even-3-parity is:

$$\begin{aligned} S &\Rightarrow \langle \text{exp} \rangle \\ \langle \text{exp} \rangle &\Rightarrow \langle \text{var} \rangle \mid \langle \text{bf} \rangle (\langle \text{exp} \rangle, \langle \text{exp} \rangle) \\ \langle \text{bf} \rangle &\Rightarrow \text{AND} \mid \text{OR} \mid \text{NAND} \mid \text{NOR} \\ \langle \text{var} \rangle &\Rightarrow \text{ARG0} \mid \text{ARG1} \end{aligned}$$

The function set for ADF1 branch consists of the union of primitive functions and the now-defined function ADF0, thereby enabling the ADF1 branch to call ADF0. The CFG for the ADF1 branch in the even-3-parity is shown next:

$$\begin{aligned} S &\Rightarrow \langle \text{exp} \rangle \\ \langle \text{exp} \rangle &\Rightarrow \langle \text{var} \rangle \mid \langle \text{bf} \rangle (\langle \text{exp} \rangle, \langle \text{exp} \rangle) \\ \langle \text{bf} \rangle &\Rightarrow \text{AND} \mid \text{OR} \mid \text{NAND} \mid \text{NOR} \mid \text{ADF0} \\ \langle \text{var} \rangle &\Rightarrow \text{ARG0} \mid \text{ARG1} \end{aligned}$$

In this example, each program has three derivation trees, one for each branch. During the fitness evaluation, the rpb is evaluated first and the others only if needed. The crossover operation maintains valid programs by ensuring that the same branch is selected at each crossover site.

4 The Chameleon Tool

The main difference between Chameleon and other systems is that Chameleon is a complete GGGP tool with ADF support. Further, there is no need to perform module compilation or changes on its kernel to solve a specific problem. Chameleon only needs a configuration file that specifies the nature of the problem at hand as well as certain conditions for the evolution process. Chameleon

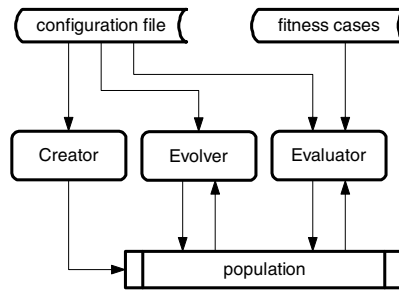


Fig. 2. Chameleon System Structure

was developed in portable C++ source code, so it can be compiled for Microsoft Windows or Linux platforms.

Chameleon has three major components: *Creator*, *Evaluator* and *Evolver*. Figure 2 illustrates the high-level structure of Chameleon and is followed by an explanation of its main components.

Creator. This component constructs the initial population. The creator implements the steps of the algorithm presented in section 2.3.

Evaluator. The Evaluator computes the fitness for each program in the population based on the *fitness cases*. The fitness cases consist of the input data for the program and the expected corresponding output. At present, Evaluator implements only the standardized fitness measure [1], that is, it computes the fitness of a program by the sum of the Hamming distance (absolute error) between the value returned by the program and the correct value. A lower result is always a better value.

This component can use one of two techniques to evaluate the programs: (a) interpretation of the derivation tree or (b) external call to a compiler and the management of the execution of each program.

An internal interpreter was developed to solve most problems presented in [1]-[2]. Nevertheless, for new problems that require new functions, it is possible to use an external compiler.

Furthermore, Evaluator implements a sophisticated control over anomalous code (overflow, infinite loop among others) by associating the worst fitness to such programs.

Evolver The Evolver performs genetic operation (crossover or mutation) on selected programs to generate new ones. A program consists of many nodes in its derivation tree. Potentially, a genetic operator can be applied on any of these nodes. However, the Evolver may use two kinds of restriction:

- First, there is a list of non-terminals that can be selected;
- Second, with a probability of 90%, we choose a non-terminal.

4.1 The Configuration File

A suitable configuration file must be supplied to adapt Chameleon to different domains. A configuration file is a simple text file. Its contents are separated in enclosed bracket sections. The main sections are:

Parameters. The usual genetic programming parameters are defined here. It includes population size, selection strategy, and derivation tree depths, among others.

Result-Producing Branch. In this part, the terminal and function sets are defined followed by the production rules.

ADF. Each ADF has its own section (ADF0, ADF1 etc). Their terminal and function sets are defined, followed by the production rules.

Crossover and Mutation. The rates and behaviors for each genetic operator are defined in this part.

Fitness Evaluation. This section defines how the program evaluation must be done.

5 Experiments

One progression of the even- k -parity problem (i.e. versions that increase in size) was chosen. This same problem was used by Koza [2] in three versions: even-3-parity, even-4-parity and even-5-parity. In this sense, using the results of Koza [2], it's possible to compare: GP (with and without ADFs) and GGGP (with and without ADFs).

A correct solution to even- k -parity problem takes a binary sequence of length k as input and returns: true (one) if the number of ones in the sequence is even; and false (zero) otherwise. Fitness cases are all combinations of the input variables.

In this section, the results are presented and analyzed. A statistical test is made to prove the effectiveness of ADFs in a GGGP.

5.1 Results and Analysis

GGGP versus GP without ADFs. Table 1 shows the maximum number of generations needed to find a satisfactory solution. It was impossible to compare the even-5-parity, because not all runs were successful, like in [2]. GGGP without ADFs is capable of solving the two problems in a similar way of traditional GP without ADFs, but it seems that its convergence is slower.

Table 1. Maximum number of generations to obtain a satisfactory solution without ADFs for: 34 runs (even-3-parity), 18 runs (even-4-parity), population size of 16000, crossover rate of 90% and no mutation [2]

Problem	GP	GGGP
3-even-parity	5	6
4-even-parity	23	28

Table 2. Maximum number of generations to obtain a satisfactory solution with ADFs for: 33 runs (even-3-parity), 18 runs (even-4-parity), 19 runs (even-5-parity), population size of 4000, crossover rate of 90% and no mutation [2]

Problem	GP	GGGP
3-even-parity	3	5
4-even-parity	10	14
5-even-parity	28	35

Table 3. Results using a Student's t-test at 95% of confidence [8]

Problem	Significance level	Conclusion
3-even-parity	0.0357043	Reject H_0
4-even-parity	$4.92430 \cdot 10^{-6}$	Reject H_0
5-even-parity	$3.04624 \cdot 10^{-7}$	Reject H_0

GGGP versus GP with ADFs. Table 2 shows the maximum number of generations needed to find a satisfactory solution. Column 2 (GP) lists the values from [2]. GGGP with ADFs is capable of solving the three problems in a similar way of traditional GP with ADFs but, once again, it seems that its convergence is slower.

The Effectiveness of ADFs in GGGP. The Student's t-test at 95% of confidence [8] was used to compare the performance of GGGP with and without ADF. The three problems were used (even-3-parity, even-4-parity and even-5-parity). The hypotheses considered are:

- *Null Hypothesis* (H_0): GGGP with ADFs does not perform better than GGGP without ADFs
- *Research Hypothesis* (H_1): The inverse of the null hypothesis; in other words, the GGGP with ADFs does perform better than GGGP without ADFs.

Table 3 shows the results using a Student's t-test at 95% of confidence [8]. The table indicates that the performance gain in all three problems is statistically

Table 4. Results for the Fourier series with a population size of 2000, crossover rate of 90% and no mutation. The maximum depth was set to 17

Generation	Best fitness
0	62.4293
10	17.4578
20	15.7404
30	15.7122
40	15.6459
50	15.3773

significant. This means that GGGP with ADFs performs better than GGGP without ADFs.

More details are found at <http://www.fesppr.br/~ernesto/sbia02>

6 Fourier Series

Many problems require some form of constrained structure in terms of the form of the resulting programs. In the traditional GP approach, for each constrained problem, there are modifications that should be done in the algorithm [1]. In the case of GGGP, the algorithm does not need any modification, since the grammar is responsible for this specialization.

To illustrate the power of this paradigm, here is presented the syntactic constraints for the Fourier series. To resolve this problem, Koza [1] performed significant alterations in his GP algorithm. This procedure becomes unnecessary in the grammar approach as can be seen in the grammar for this problem shown below.

$$\begin{aligned}
 S &\Rightarrow \text{< code >} \\
 \text{< code >} &\Rightarrow \text{< trig >} + \text{< trig >} \\
 \text{< trig >} &\Rightarrow \text{< trig >} + \text{< trig >} \mid \text{< fun >} (\text{< term >}, \text{< term >}) \\
 \text{< fun >} &\Rightarrow \text{XCOS} \mid \text{XSIN} \\
 \text{< term >} &\Rightarrow \text{ERC} \mid (\text{< term >} \text{< op >} \text{< term >}) \\
 \text{< op >} &\Rightarrow + \mid - \mid * \mid /
 \end{aligned}$$

Chameleon was executed with this grammar and the results are presented in Table 4. As we can see, the best fitness falls quickly in the first ten generations and it remains about 15 for the rest of the generations, due to the limit of the tree depth.

More details are found at <http://www.fesppr.br/~ernesto/sbia02>

7 Conclusions

Genetic Programming induces programs by searching for a highly fit program in the space of all possible programs. With the use of grammars, the search space can be declared explicitly, avoiding the generation of invalid programs in the problem's domain. This is done without any modification in the algorithm.

In this work, two extensions for GP were integrated: Automatically Defined Functions (ADFs) and Grammar-Guided Genetic Programming (GGGP). Chameleon, a tool with these features, was also presented. The results for different experiments showed that the GGGP with ADFs produces better solutions faster than GGGP without ADFs.

It was noticed that GGGP results are quite similar to those of traditional GP. The small difference can be due to different tree structures or the initial population creation mechanism. Further work is needed to elucidate these differences.

Due to its interesting results, Chameleon's GGGP features will allow the exploration of others domains and new challenges.

References

- [1] Koza, J.R. : Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA (1992) 324, 325, 329, 332
- [2] Koza, J.R. : Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge, MA (1994) 325, 327, 329, 330, 331
- [3] Banzhaf, W.; Nordin, P.; Keller, R. E.; Franconi, F. D.: Genetic Programming - an introduction. Morgan Kaufmann, San Francisco, CA (1998) 324, 327
- [4] Whigham, P. A.: Grammatically based Genetic Programming. In: Proceedings of ML'95 Workshop on Genetic Programming - From Theory to Real-World Applications. Lake Tahoe, CA (1995) 33-41 324, 327
- [5] O'Neil, M.; Ryan, C.: Grammar based Function Definition in Grammatical Evolution. In: Genetic Programming 2000: Proceedings of the 5th Annual Conference, MIT Press (2000) 485-490 325
- [6] Ratle, A.; Sebag, M.: Genetic Programming and Domain Knowledge: beyond the limitations of grammar-guided machine discovery. In: Proceedings of the Sixth Conference on Parallel Problem Solving from Nature, LNCS, Springer, Berlin (2000) 211-220 325, 326
- [7] O'Reilly, U.: An Analysis of Genetic Programming. PhD thesis. Ottawa-Carleton Institute for Computer Science (1995) 327
- [8] Cohen. P. R.: Empirical Methods for Artificial Intelligence. MIT Press, Cambridge, MA (1995) 331
- [9] Gruau, F.: On Using Syntactic Constraints with Genetic Programming. In: Advances in Genetic Programming, MIT Press (1996) 377-394 324

An Evolutionary Behavior Tool for Reactive Multi-agent Systems

Andre Zanki Cordenonsi¹ and Luis Otavio Alvares²

¹Centro Universitário Franciscano □ UNIFRA, Curso de Sistemas de Informação
Rua dos Andradas 1614 □ CEP 97030 020, Santa Maria □ RS □ Brasil
andrezc@unifra.br

²Instituto de Informática □ UFRGS
Av. Bento Gonçalves 9500 □ CEP 91501 970, Porto Alegre □ RS □ Brasil
alvares@inf.ufrgs.br

Abstract. Multi-agent Systems (MAS) are a sub-area of Distributed Artificial Intelligence which focus on the study of autonomous agents and their actions in an environment. This paper presents a simulation environment for Reactive Multi-agent Systems called Simula++, where an evolutionary algorithm can modify the set of behavior rules of each agent. Our major goal is to define and develop a model to dynamically change the agents' behavior in order to adapt the agents to their environment. In the Simula++ environment, an user can define a Reactive MAS where the predefined rules set of each agent can be modified to create new rules during simulation. That would happen through the precepts of Artificial Life and Evolutionary Algorithms.

1 Introduction

One of the main objectives of Distributed Artificial Intelligence (DAI) is the construction of intelligent systems formed by autonomous entities (agents). Multi-agent Systems (MAS) is a sub-area of DAI and concentrates on the development of autonomous agents in a multi-agent environment. Usually, each agent has a set of behavior capacities which define its competence, a set of objectives, and the necessary autonomy to use its capacities in order to reach its objectives [1]. The current state of the environment and the agent's desire to reach its objectives will define the action each agent will execute. The main idea in a MAS is that an intelligent global behavior can be reached starting from the agents' individual behavior. In this paper, we work with reactive agents, without any kind of deliberation [2].

The development of MAS would be facilitated if the agents could adapt and develop their actions according to the dynamic changes in the environment. This paper presents the specification and implementation of an evolutionary behavior environment for Reactive Multi-agent Systems, called Simula++, where the agents' behavior can be modified during simulation. The tool, whose main objective is MAS

technology teaching , was based on the Simula [3] system and is composed of three main elements: Simulation Environment, Reactive Agents and the Evolutionary Algorithm.

2 Artificial Life and Evolutionary Algorithms

Artificial Life main objectives are the understanding of life through the abstraction of its fundamental dynamic principles, and the creation of such dynamics in other physical medias, such as computers (which will be accessible to new types of experimental manipulations and tests), as defined by Langton [4]. Therefore, we can characterize Artificial Life as the study of systems built by human beings (artificial) that exhibit characteristic behaviors of natural alive systems (biological). The main concept in Artificial Life is the emergent behavior.

Steels [5] defines the system behavior as emergent if it is defined using descriptive categories that are not necessary to describe the fundamental components of the system. The nature shows many examples in which simple actions and the local interactions among its components create highly organized global behaviors (insects colonies, assembly of cells, the ocular retina). The biological evolution can be defined as the progressive change in the genetic material of a population in many generations.

John Holland [6] develops a generic algorithm for all the adaptive systems (natural or artificial), and demonstrates that an evolutionary processes can be applied in artificial systems. To Holland, any adaptive systems could be formulated using genetic terms, through the Genetic Algorithm (G.A.).

The Simula++ system uses a G.A. to evolve each agent's behavior. This algorithm is based on the concepts of Artificial Life, where the natural selection is only based on the agents's survival. The agents in Simula++ do not make action plans[7].

3 The Simula++ Tool

The main objective of Simula [3] Tool is the Reactive Multi-agent Systems (RMAS) technology teaching through a system that allows a fast simulation, using an interactive graphic interface, unlike other simulation environments for RMAS such as Swarm [8] and Sieme [9], whose objectives are the construction and simulation of complex models. The user does not need to have any knowledge concerning the low level operations of the simulation, such as: the movement of the agents, the perception of the environment and the synchronism of the simulation. After this prototype, extensions that allow the system to work with agents whose behavior can evolve were specified and implemented, using the paradigms of Artificial Life and Evolutionary Algorithms. The main objective of Simula++ is to provide a didactic tool for modeling and simulation of RMAS, where behavior defined for each agents's classes can evolve during the simulation.

3.1 Behavior Definition for Each Class of Agents

Initially, the user defines the several classes of agents, where each agent begins the simulation with a set of rules that defines the agent's behavior. The user defines also the number of agents of each class. The set of rules of each class, which is known as Initial State, is the same for all agents of the same class. The set of rules can suffer dynamic modifications, at execution time, evolving from the Initial State, aiming at a better adaptation to a particular defined environment. Therefore, each agent, individually, evolves its own set of rules, in an independent way. The Simula++ environment uses a declarative representation for the set of rules. Each rule has three components:

- precondition: condition to execute the behavior,
- activated-action: behavior which will be executed,
- priority: order for behavior execution, considering the whole group of rules.

The precondition and activated-action definition use a set of predefined primitives that can be combined through the logical operators AND, OR and NOT. These primitives use colloquial expressions of natural language, to make the construction of the set of rules that define the agent's behavior easier.

3.2 Evolution and Adaptability

The main characteristic of Simula++ is centered on the agents' adaptability. This adaptability results from the use of the evolutionary algorithm in each agent individually. Through crossover and mutation operators on the Initial State of rules, the system builds and dynamically tests new rules and behavior patterns.

4 The Evolutionary Model for the Set of Rules

This section presents the model used for behavior (set of rules) evolution for each agent of the system. The structure of an agent in Simula++ is divided in two components:

- Independent Elements: variables that do not suffer modifications by the evolutionary algorithm during the simulation process. These variables are used to control the evolutionary process. The main independent variables are:
 - IE (Initial Energy): amount of available initial energy for the agent,
 - EA (Energy Amount): amount of energy of the agent. This quantity decreases by one unit each simulation step. The agent can increase its amount of energy, according to its set of rules,
 - MLT (Maximum Life Time): maximum time of simulation for each agent,
 - LT (Life Time): number of simulation steps already executed by the agent,
 - SMT (Sexual Maturity Time): the minimum simulation steps necessary for the agent to have the capacity to generate offspring,
 - NGT (New Generation Time): the number of simulation steps necessary for the agent to generate a new offspring (after the period defined by the SMT).

- **Chromosome:** the set of rules. Each agent has a chromosome that codes the rules which define the behavior of the agent. These rules can be modified through crossover and mutation operators.

The classes of agents (section 3.1) and their structures (Independent Elements and Chromosome) are easily configured by the user through a graphic interface, as it is shown in fig. 1(a) and 1(c). After definition of the classes of agents and their respective set of rules, the simulation can be executed graphically, as shows fig. 1(d). The simulation control uses two main structures:

- **Agents List (AL):** it stores all agents which are acting in the system,
- **Fertile Agents List (FAL):** it stores a list of pointers for all agents which have already surpassed the SMT.

All classes of agents have a predefined maximum period of life (MLT) and a sexual maturity time (SMT). These periods are the same for all agents of the same class, which guarantees the equal survival conditions for all agents of the same class. The period of sexual maturity should be considered as a test for the set of rules of each agent. If an agent stays alive after this period, it is considered adapted for this specific environment and capable to generate an offspring. When an agent reaches the SMT, it is included in the Fertile Agents List that contains all the capable agents to generate new offspring. After that, the agent should choose, randomly or through the fitness value, a second agent of the Fertile Agents List. This second agent is used, with the first one, to create a new offspring, using the genetic operators, as we will see further on. After being included in the Fertile Agents List, each agent will generate a new offspring in intermittent cycles predefined by the user (NGT).

Figure 2(a) presents the execution algorithm for all agents defined in the system Simula++, considering n_{Agents} the number of agents of the environment modeled. The number of offsprings of each agent is variable. This fact can be explained by the agents' life time. If the EA of an agent arrives to zero, even if MLT has not been reached, it will be eliminated from the simulation. Therefore, the survival and the number of offsprings of each agent depend on its adaptability to the environment in which he acts. New offspring agents inherit the rules of their parents' behaviors, through the application of the three genetic operators: external crossover, internal crossover and mutation.

5 Genetic Operators

The Simula++ tool defines three genetic operators. Crossover operators are used during the offspring generation, using the chromosomes (representing the set of rules) of the parent agents. The user must define just one type of crossover, before the simulation starts. Let us consider the following definitions:

- **Parent One:** agent which will generate a new offspring,
- **Parent Two:** agent chosen in the List of Fertile Agents. There are two methods to choose this agent: randomly or through the elitism concept [11] (the probability of choice is proportional to the fitness value),
- **Offspring:** the agent which is being generated.

5.1 External Crossover

The *External Crossover* combines the set of rules of Parent One and Parent Two, generating an offspring which have characteristics (rules of behavior) inherited from the two agents. This operator works with the whole set of rules: if a rule is selected, the three components (precondition, activated-action and priority) are selected together. There are two types of External Crossover: One-point and Two-point.

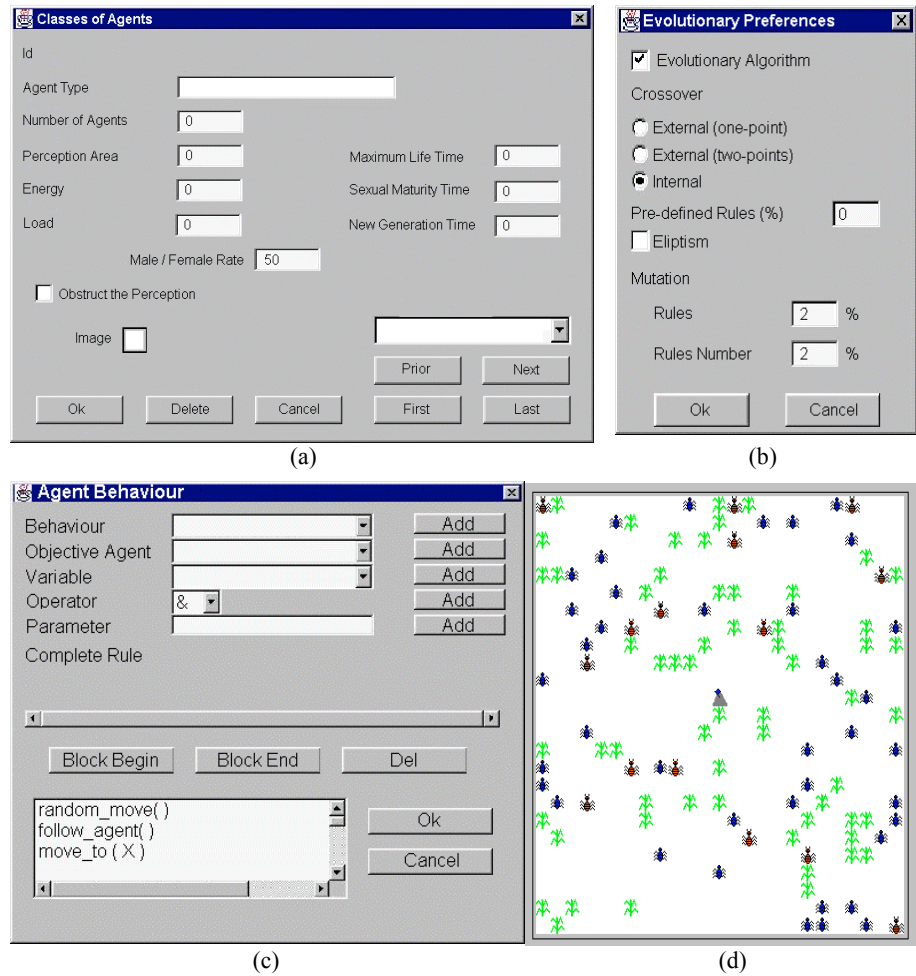


Fig. 1. (a) Definition of the classes of agents. (b) Definition of the evolutionary algorithm. (c) Definition of the set of rules for each class of agent. (d) The simulation environment

The One-point, as seen in fig. 2(b), defines a random break point in the set of rules, selecting the first rules from Parent One and the others from Parent Two. All rules selected generate a new chromosome which represents the new set of rules that will be set for the new agent offspring. The Two-point, as seen in fig. 2(c), defines, randomly, two break points in the set of rules. This operator selects the first rules

from Parent One, the intermediary rules from Parent Two and the last rules again from Parent One. These rules will generate the new chromosome, just like the One-Point External Crossover. In both, the priority of the rules selected for the new chromosome does not change and the number of rules in the new offspring is the average of the number of rules of Parent One and Parent Two. If two equal rules are selected, one of them is discharged, independent of the priority value.

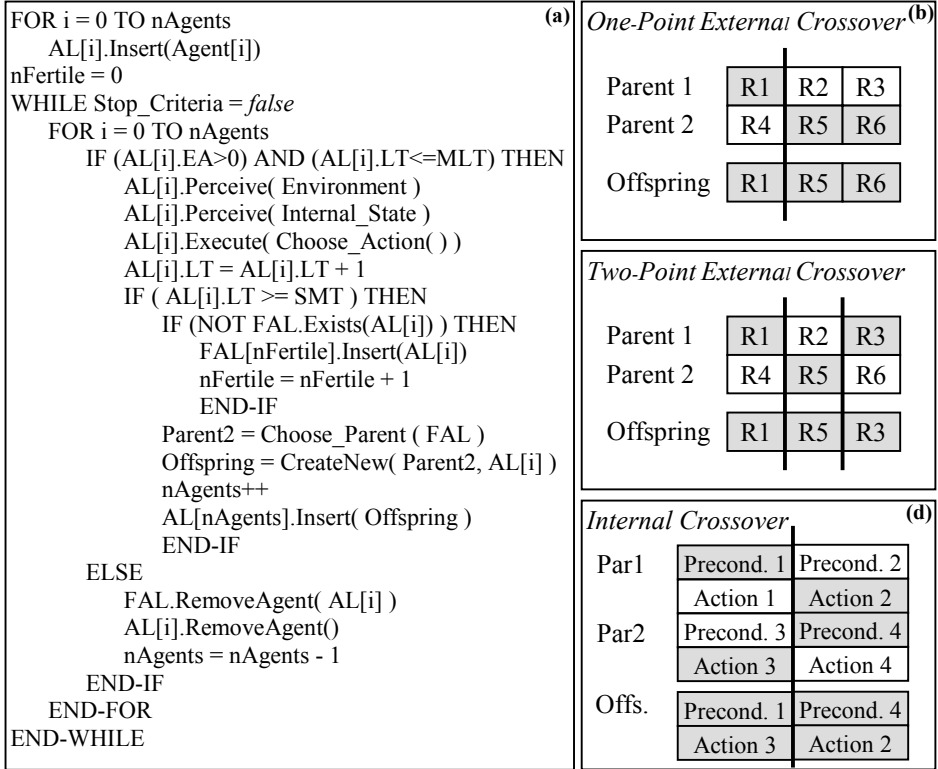


Fig. 2. (a) Execution Algorithm. (b) Model for the one-point external crossover. (c) Model for the two-point external crossover. (d) Model for the internal crossover.

5.2 Internal Crossover

The *Internal Crossover* operator is used for the same purpose of the *External Crossover* operator: generation of a new offspring, combining the chromosomes of Parent One and Parent Two. However, the Internal Crossover changes parts of rules (parts of precondition and activated-action) from Parent One and Parent Two, using an one point crossover. This type of crossover defines a randomly break point in the set of rules, separating them in two groups. The first group is formed by the preconditions from Parent One and the activated-actions from Parent Two. The second group is formed by the preconditions from Parent Two and the activated-actions from Parent One. The union among these two groups will form the new chromosome of the new offspring, as seen in fig. 2(d).

The Internal Crossover expands the search space, generating a great number of rules which were not tested previously, while the External Crossover will generate new combinations of rules already created by the system. The Internal Crossover produces new different behaviors, but it can produce a great number of agents that do not present good rules of behavior, increasing the simulation time.

5.3 Mutation

The mutation operator is used to randomly modify the offspring chromosome, altering the set of rules after the crossover operator generated it. This alteration is accomplished through a random change in the rule elements by expressions of same syntactic value. For example, the algorithm can change an *and* operator by an *or* operator, as well as a *precondition function* by another *precondition function*. This operator acts after the generation of the new chromosome in two very different ways: alteration in the number of rules of the chromosome and modification of a specific rule. In the first one, the operator increases or decreases the number of rules using the algorithm above:

- In the simulation initialization, a mutation rate (M) is defined, usually between 0.01 and 0.02, and it is used during all the simulation,
- For each new generated agent, a random number (R) is generated between 0 and 1. If $R \leq M$, the chromosome of the offspring will be altered:
 - A new raffle is executed to define if there will be an insert of new rules in the chromosome (copying rules from one of the two Parent agents) or a removal (random elimination of rule from the offspring)
 - After the choice for the action that will be executed, the number of rules that should be inserted or removed is raffled. This operator can not insert or remove more than $\square$ of the total number of offspring rules.

In the second way, the mutation operator is used under the new group of rules of the offspring, which already suffered the modifications discussed previously. In this case, the mutation operator can produce small modifications in the expressions behavior rules use, with the purpose of altering them and, consequently, expanding the search space. The algorithm for this kind of modification can be described this way:

- In the simulation initialization, a mutation rate (M2) is defined;
- For each syntactic element of the rules, a random number (R2) is generated between 0 and 1. If $R \leq M$, the expression of the offspring will be altered.

The mutation operator, used to modify the chromosome, acts as a generator of a new set of rules, because the small modifications can alter the behavior of an agent. The modification alters two components with the same characteristics: an action must be replaced by another action and a precondition must be changed by another precondition. The new components are randomly chosen from the predefined primitives of the system, defined in section 3.1.

5.4 Fitness Function

The Simula++ simulations do not work with an explicit fitness. The adaptation of an agent's ability is only measured by its own survival in the environment. If an agent can survive for a long time, it increases its fitness degree. However, some specific environments must use an explicit fitness function, to measure the solution quality founded by the agent behavior. Simula++ tool defines three variables (success, load and energy) for each agent, which can be used for the explicit fitness function definition.

6 The Food Foraging Problem

The Food Foraging Problem was developed as a model for the emergent functionality of a RMAS in [5]. Given a certain unknown environment, there are minerals (or food) deposits which a collection of robots transport to a central base. There is no information about the environment characteristics. The experiment introduced in this section presents the characteristics described in Drogoul [10]. In his model, the environment is bidimensional, defined by a grid (100x100), where each agent can occupy a space of this grid. There are three types of agents: (1) Base: placed in the central area of the environment, the base is used as warehouse of the minerals collected by the robots. The base is fixed. (2) Mines: there are three mines spread in the environment, at a fixed distance of 40 cells of the Base. Each Mine has 100 units of mineral. (3) Robots: the active agents of the simulation. They can move all over the environment in search for minerals. When they find it, they should come back to the base to discharge its load. A robot can carry only one mineral unit at a time.

The Robots have two perception sensors. The first is used to find the mines and his range is defined as a circumference of ray two. The second sensor is used to find the Base, after the agent has been loaded with the mineral. The Base emits a sign that an agent can realize in a range of 40 cells. The number of Robots involved in the system can vary, but it is fixed during a simulation. There are two kinds of behavior predefined for all the Robots:

- If a robot not loaded finds a Mine, he loads an unit of mineral,
- If a robot is loaded and finds the Base, he discharges the unit of mineral.

All the Robots start the simulation with a random set of rules. The MLT of all Robots is 200 simulation steps. After this period, the agent is removed from the environment and a new offspring is created to keep the number of agents fixed. There is a fitness function used to choose the Parent One and Parent Two agents, which is measured through the following rules: (1) if an agent loads a mineral, its fitness is increased by one point, (2) if an agent discharges a mineral in the Base, its fitness is increased by two points.

The number of Robots varied from 1 to 100 units. For each situation three simulations were executed. This group of simulations was denominated Evolutionary (E) Group, because the evolutionary algorithm was always present and the set of rules were always randomly initialized for each new simulation. We used these simulations to accomplish a comparative analysis with the results obtained by Drogoul [10].

Drogoul presents four kinds of agents to solve this problem: *petit poucet 1*, *petit poucet 2*, *petit poucet 3* and *Dockers*. The *petit poucet* agents use small variations from the following strategy: (1) random movement to find a Mine, (2) if an agent finds a Mine, it loads a mineral and returns to the Base, leaving a mark track, (3) if an agent finds a mark track, it follows them and removes the marks. The *dockers* use a different approach: (1) random movement to find a Mine, (2) if an agent finds a Mine, it loads a mineral and returns to the Base, leaving a mark track, (3) if an agent finds a mark track, it follows them and removes the marks, (4) if an agent finds another agent with a mineral, it picks his mineral and returns to the base. This strategy generates trails of workers from the Mine to the Base.

From the analysis of data obtained the following rules were listed, denominated Static (S) Group: (1) if an agent does not find a Mine and does not perceive a Robot, random movement, (2) if an agent does not find a Mine and perceive a Robot, it follows the Robot, (3) if an agent does not find a Mine and reaches a Robot, flees the Robot, (4) if an agent finds a Mine, it loads a mineral and returns to the base, leaving a mark track, (5) if an agent finds a mark track, it follows it and removes the marks. It is important to point out that the analysis of the twenty best agents compiled these rules. Analyzing this group of agents, the rules had been chosen based on the degree of occurrence. The similar or innocuous rules have been removed. We made another 300 simulations with the S Group (without evolution) and a comparative graph was plotted, as seen in fig. 3(a).

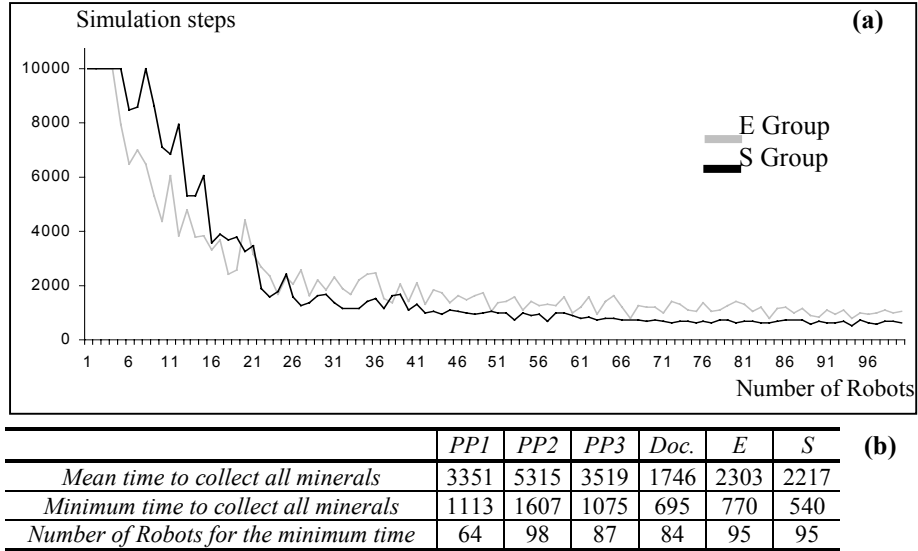


Fig. 3. (a) S Group x E Group. (b) A comparative table between the *Petit-poucet* and *Dockers* robots from Drogoul [10] and the S Group and E Group presented in this paper

There are likeness between the rules from S Group and the *Petip-Poucet* from [10]. The largest difference occurs when the robots are looking for the Mines. In this situation, the robots, in agreement with the rule 2, tend to form small search groups. These groups move randomly in the environment. These groups decrease the search

space, because the agents are less dispersed. This situation was solved by the evolutionary algorithm through the rule 3 that does not allow a great agglutination. An interesting advantage in this strategy is the largest possibility that all the robots indeed act in the load of the minerals, after some Robots have discovered a Mine. A table comparing the results of Drogoul and of E and S Groups is presented in fig 3(b). The S Group is better than the *petit poucet* approaches and it can collect the minerals more quickly than the *dockers*, in the best case. However, the agents are less dispersed, so it can waste time to find the minerals. An interesting approach would use the S rules to find the mine and the *dockers* rules to get the minerals to the base.

7 Conclusions and Further Works

The main advantages in the use of Simula++ tool are the visual analysis of the RMAS behavior that presents adaptability and evolution, as well as allowing the discovery of new rules for existent problems and systems, through the precepts of the natural selection. The graphical interface allows to the user to have a rapid prototype of his model. In addition, the user can study many approaches and models for the same problem. A very important characteristic is the easy paradigm change. The user can define that its application does not need to use the evolutionary algorithm, coming back to the simulation with a static set of rules. These rules can be the same as those the evolutionary algorithm discovered. Analyzing the results obtained, we can notice some important characteristics of E.A. and MAS methodologies: (1) the use of a fitness function, explicit or implicit, and its definition has a directly influence on the adaptation of the agents; (2) the use of two fitness measures, a global (problem) and a local (agent), can improve the adaptation of the agents to the proposed problems. As a further work, we intend to use a representation schema of these fitness functions that the user can change directly.

References

- [1] Alvares, L.O., Sichman, J. Introdu o aos Sistemas Multiagentes. In: Jornada de Atualiza o em Inform tica, 16.; Congresso da SBC, 17. 1997. Bras lia. p.1-38.
- [2] Wergner, B. B. Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams. In: Artificial Intelligence. Vol. 110. Elsevier (1999) 293-320.
- [3] Frozza, R. Simula   Ambiente para Desenvolvimento de Sistemas Multiagentes Reativos. Porto Alegre: CPGCC da UFRGS, 1997, Master Thesis.
- [4] Langton, C.G. Self-reproduction in Cellular Automata. In: Physica D, v. 10, p.135-144. 1984.
- [5] Steels, L. Cooperation Between Distributed Agents Through Self-Organization. In: Decentralized A.I. Demazeau, Y. e Muller, J-P. (Eds.), Amsterdam, 1990.

- [6] Holland, J.H. *Adaptation in Natural and Artificial Systems*, Massachusetts: MIT Press, 1992.
- [7] Gordon, D.F. Asimovian Adaptive Agents. In: *Journal of Artificial Intelligence Research*. Vol 13. (2000) 95-153.
- [8] Hiebeler, D. *The Swarm Simulation System and Individual-based Modeling*. In: *Decision Support 2001: Advanced Technology for Natural Resource Management*, Toronto, 1994.
- [9] Magnin, L. SIEME: an Interaction Based Simulation Model. In: *XII European Simulation Multiconference (ESM 98)*, Manchester, United Kingdom, 1998.
- [10] Drogoul, A. *De la Simulation Multi-Agents à la Résolution Collective de Problèmes*. Paris, France: Université Paris VI, 1993. PhD. Thesis.
- [11] Koza, J. R. *Genetic Programming: on the programming of computers by means of natural selection*. Massachusetts Institute of Technology. 1998.

Controlling the Population Size in Genetic Programming

Eduardo Spinoso and Aurora Pozo

Computer Science Department, Federal University of Paraná (UFPR)
P.O. Box 19081, Zip Code 81531-990, Curitiba, PR, Brazil
ejspin@yahoo.com
aurora@inf.ufpr.br

Abstract. Evolutionary Computation (EC) introduces a new paradigm for solving problems in Artificial Intelligence, representing solution candidates as individuals and evolving them based on Darwin's Theory of Natural Selection. Genetic Algorithms (GA) and Genetic Programming (GP), two important EC techniques, have been successfully applied both in theoretical scenarios and practical situations. This work discusses an issue of great relevance and impact on this type of algorithm: the automatic adjustment of the parameters that control the search process. Based on a recent research, a method that controls the population size in a GA is adapted and implemented in GP. A series of classic experiments has been performed before and after the modifications, showing that this method can improve the algorithms' robustness and reliability. The data allow a discussion about the method and the importance of the adaptation of parameters in EC algorithms.

1 Introduction

Evolutionary Computation (EC) uses concepts of natural selection of beings to create powerful search techniques, such as Genetic Algorithms (GA) and Genetic Programming (GP).

EC algorithms are controlled by certain parameters. Some of those have a very important impact on how the search is performed, its limitations and, consequently, on its performance as a whole. Adequate choices can lead to faster and better results. However, it is difficult to predict which would be the optimum set of values for a given problem without actually running the algorithm. This situation has motivated many researchers to develop methods for adapting some critical parameters automatically.

This paper focuses on a very important parameter: the population size. A method that controls its value initially proposed for a GA is adapted and implemented in a GP system. Further information can be found on the original research that motivated this paper [19].

The following section reviews basic information on GP. Section 3 summarizes the types of methods used to adapt the parameters. Section 4 explains the method that controls the population size in detail and how it has been implemented in GP. Sec-

tion 5 discusses some of the experiments performed and analyses the results. Finally, Section 6 concludes the paper and gives insights to future works.

2 Genetic Programming

Darwin's Natural Selection Theory [4] shows that individuals that better adapt to the environment have a greater chance of surviving and passing their genetic characteristics to their offspring.

Genetic Programming (GP) is the use of the Natural Selection theory in computers, to automatically generate programs. It was presented by John Koza [12], based on the idea of Genetic Algorithms introduced by John Holland [9] [17]. Instead of a population of beings, in GP there is a population of computer programs. And the main goal is to naturally select the program that better solves a given problem. The GP algorithm can be summarized as follows [12] [3]:

The first step of the evolution process is to randomly generate an initial population. Then, the algorithm enters a loop that is executed, ideally, until a desired solution is found. This loop consists of two major tasks:

- Evaluate each program, by the use of a special heuristic function "fitness" that shows how close each one is to the ideal solution.
- Create a new population by selecting individuals based on their fitness and applying the basic genetic operators: reproduction, crossover and mutation.

The behavior of the algorithm is determined by a set of parameters that, among other things limit and control how the search is performed. Some of them are: genetic operator's rates (crossover rate, mutation rate), population size, selection rate (tournament size), maximum depth of the individual etc. The following section presents techniques that have been used to perform control over these parameters.

3 Previous Works

In EC algorithms, parameters can determine, among other things, the success probability of the search. Adequate choices will make good use of the resources and provide better results. Still, a great amount of GA and GP applications do not implement any sort of dynamic control over the parameters, which motivates studies in that field. Many techniques have been proposed and a recent classification, shown in Fig. 1, indicates the major pathways that can be explored [6].

Parameter adjustment is the traditional choice of values done in the initialization process. After that, parameters remain fixed for the entire run. Parameter control represents any method that performs dynamic adaptation during the run.

A deterministic control is guided by a fixed heuristic rule, as opposed to the adaptive control, where the adaptation is based on statistical information about previous generations [5] [10] [8] [15]. The self-adaptive technique goes one step beyond, encoding parameters inside the individuals and evolving them as well [2] [1] [18].

Finally, a meta-algorithm can also be used to control parameters, experimenting different sets of values and evolving them in several slave GA algorithms until the optimal set is discovered [16] [7] [11].

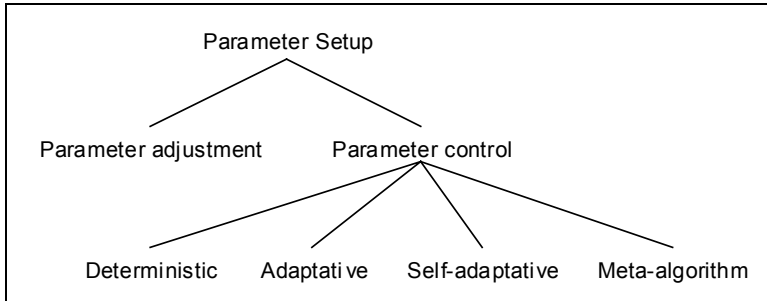


Fig. 1. Parameter setup techniques

This paper focuses on a recent adaptative control method called "Parameterless GA" [15] that dynamically controls one of the most important parameters: the population size. The population size determines the number of individuals in the population and, consequently, the number of exploration points in the search space. It has a direct impact on the diversity of the population, which is a critical element that allows the formation of solutions.

Besides that, Lobo [15] analyses the Schema Theorem to perform the configuration of two other parameters: the crossover rate and the selection rate. After a theoretical analysis of the Schema Theorem [15], it is possible to; disregarding the effect of the mutation operator, express the growth rate of a schema as:

$$\frac{m(H, t+1)}{m(H, t)} = s \cdot (1 - p_c) \quad (1)$$

From (1) we can see that the growth rate for schema H (left side of the equation) depends on both the selection rate (s) and the crossover rate (p_c). Extremely high or low values for growth rate will not allow the algorithm to compose a proper solution. Considering a growth rate of 2, Lobo fixes the crossover rate in 0,50 and the selection rate in 4. According to his studies, these values provide good results, allowing useful schemata to survive in the population and participate in the formation of valid solutions.

The following step in Lobo's method is the dynamic adaptation of the population size (GA). The next section shows how this technique has been adapted and implemented to control the population size in GP.

4 Controlling the Population Size in GP

This work uses the same "Parameterless" approach of Lobo [15], which has been originally created and tested in a Genetic Algorithm to control the population size in

Genetic Programming. No modifications were made to the original method. This section presents the strategy and how it has been implemented in GP.

The method proposes a competition among multiple populations of various sizes. Starting with an initial population of size x , other populations (of sizes $2x$, $4x$, $8x$ and so on) will be automatically created by the algorithm when needed.

Generations are executed according to a counter that assures that a new population is created whenever the previous one has been executed 3 times. In other words, it will execute populations in the following order: 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 3, 1, 1, 1, 2,... Smaller populations are executed more frequently because the main goal is to find the solution without the need to create larger populations.

The algorithm constantly compares the quality of populations by their *fitness* value (the average of the *fitness* of all individuals). If a larger population has a better *fitness* than a smaller one, the larger *overtakes* the smaller, and the smaller is removed from the process. The race continues until one of the populations finds a solution, but it can also be stopped by an optional termination criterion that prevents an infinite loop.

4.1 Implementation

To implement the method in GP, a classical GP tool called *lil-gp* [20] has been used. An external control module (Fig. 2) is responsible for providing commands to *lil-gp* and receiving statistical information.

This approach allows a direct comparison between results with and without the adaptation of the population size, since the internal GP algorithm is treated as a *black box*. The internal architecture of the control module includes a counter for each population and a *fitness* table. A special procedure checks the occurrence of *overtake* and eliminates useless populations from the disk. At each point, there is only one active population, which assures that populations are completely independent among themselves.

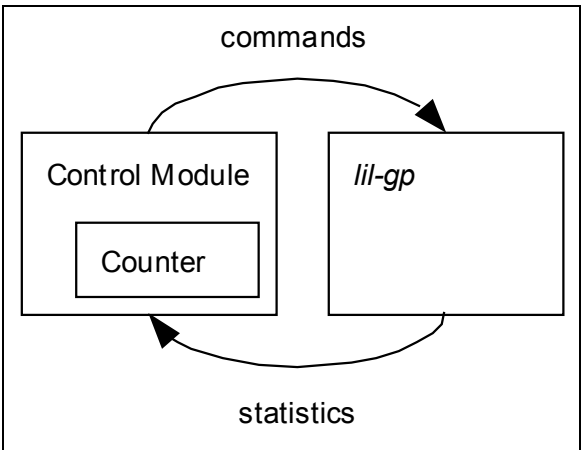


Fig. 2. Control Module interaction with *lil-gp*

5 Experiments

To test the influence of the control module and compare the results with standard GP, two classic problems were implemented in *lil-gp*. This section presents these experiments and discusses the most significant results.

The Artificial Ant Problem: The artificial ant problem [12] is a classic GP application frequently used for benchmarking [14] due to its difficulty level. The problem consists of a grid (e.g. Santa Fe Trail) where some cells are marked as "food" cells. The goal is to generate a computer program to guide the ant in the grid passing by as many food cells as possible. To do that, a set of terminals and functions is provided, allowing the program to move the ant forward, turn right or left, and detect if there is food in the cell directly in front of it.

The Symbolic Regression Problem: This is another classic problem where the goal is to find the equation of the curve that better adjusts to a given set of (x,y) points. A certain number of arithmetic functions is provided and terminals can either be a variable (x) or an ERC (*ephemeral random constant*), chosen at the beginning of each run.

5.1 Methodology

A total of 12 experiments were performed, in an attempt to isolate the influence of each modification made. They are:

- Population size - In the tests with and without adaptation of the population size, *lil-gp* used the same configuration file. The only difference is the absence or presence of the control module. In the tests without adaptation, the population sizes used were: 500, 1000, 2000, 4000, 8000, 16000 and 32000.
- Problem type - For each situation three types of tests were performed: artificial ant problem (Santa Fe Trail), symbolic regression of a 3rd order polynomial and symbolic regression of a 4th order polynomial.
- Algorithm's parameters - Two sets of parameters were used. The first is the standard parameters set proposed by Koza in his second book [13], in which the crossover rate is 90% and the tournament size is 7. The second set is the one proposed by Lobo, and modifies these two values to 50% and 4 respectively, according to the theoretical analysis based on the Schema Theorem [15].

In each experiment, 50 runs were executed, which assures that the results represent the average behavior and not extreme situations.

To allow some kind of comparison between the two main techniques (standard GP and GP with the adaptation of the population size) the same type of graph has been plotted (Fig. 4 and Fig. 5). It shows the success probability versus the number of individuals processed. Additionally, another type of graph (Fig. 6) is used to provide a different perspective, showing the best fitness of each population.

This paper presents the results for one type of problem: the artificial ant with the Santa Fe Trail configuration with Koza's II parameters [13]. Similar results were obtained with the other two implementations [19].

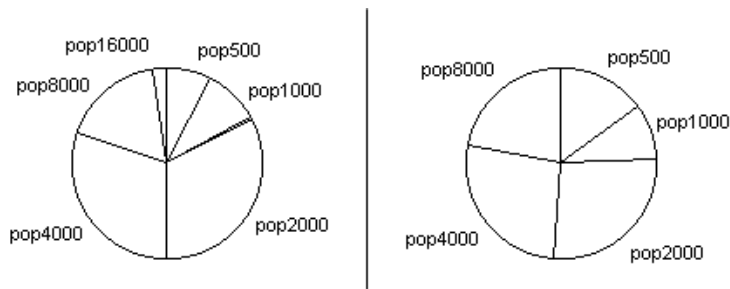


Fig. 3. Participation of each population in the discovery of solutions with Koza's II parameters (left) and Lobo's parameters (right)

5.2 Analysis of the Results

Both sets of parameters (Koza II and Lobo) provided similar levels of success probability. When the dynamic control of population size is used, the only difference is that with Lobo's parameters the participation of smaller populations in the discovery of solutions was increased, thus reducing the need to use larger populations (Fig. 3). A deeper study on the impact of the crossover rate and tournament size is needed to provide a better understanding of the change observed in these results.

In the first set of experiments without adaptation (shown in Fig. 4), consistent results were found, showing that larger populations have a greater chance of solving the problem. Considering that larger populations have a higher number of individuals and that each individual represents a certain configuration of a possible solution, larger populations have a greater chance of discovering a solution by recombining individuals. In other words, the higher diversity of larger populations allows them to better explore the search space.

On the other hand, it was also observed that extremely small populations tend to be useless and almost never able to find a solution. Considering that it is impossible to know *a priori* what the optimal population size will be for any given problem, a self-adaptation mechanism such as this can avoid the use of an excessively small population that will not manage to solve the problem. This fact confirms the importance of performing some kind of automatic adaptation in the population size.

Furthermore, it was also noticed (Fig. 6) that larger populations are slower in achieving higher levels of quality (fitness of the best individual). This is an indication that it is not always true that the higher the population size, the better it is. Sometimes a medium sized population can achieve good results with less computational cost than a larger one. For instance, in Fig. 6, the population of 16000 individuals achieves almost as good results as the population of 32000, with only half the processing. Once again, the "parameterless method" provides a more intelligent way of identifying the most suitable population size.

The second set of experiments with adaptation is shown in Fig. 5. A single curve represents the execution of the algorithm, starting with a population of 500 individuals and adapting (increasing) the size as needed. Although the number of individuals executed has increased, the algorithm is now smarter and able to detect that a small

population is useless, replacing it with a larger one. This can be seen in the constantly growing curve of Fig. 5.

It is important to point out that the increasing in the robustness of the algorithm provided by this method pays a price: a higher number of executions is required for achieving good levels of success probability. This can be explained by the fact that every time the algorithm discards an entire population of size x that has already been executed for g generations, a total of $x.g$ executions are lost. It is not possible to avoid this, since the algorithm cannot determine in advance the optimal population size for each new problem.

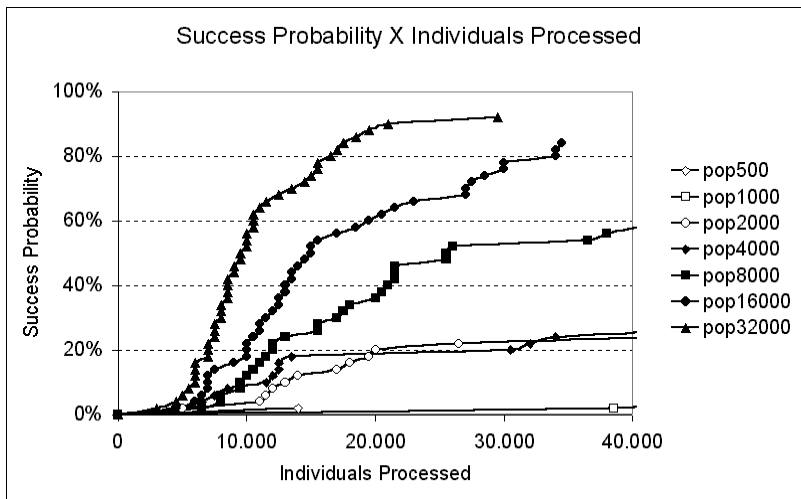


Fig. 4. Success probability without population control

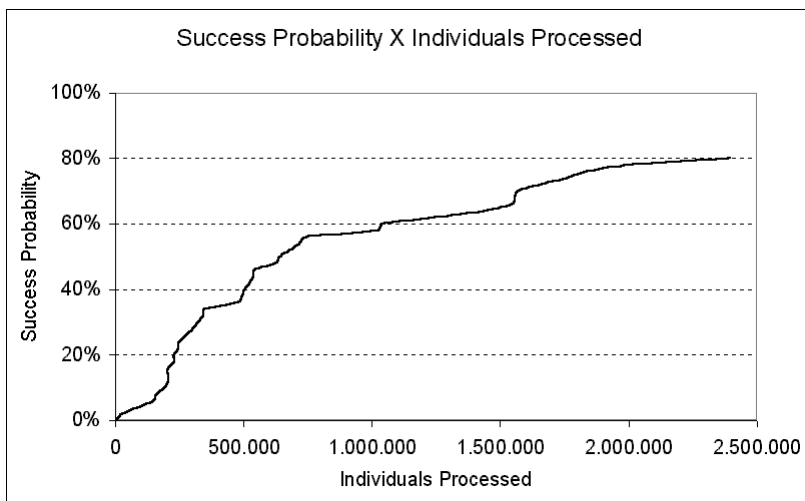


Fig. 5. Success probability with population control

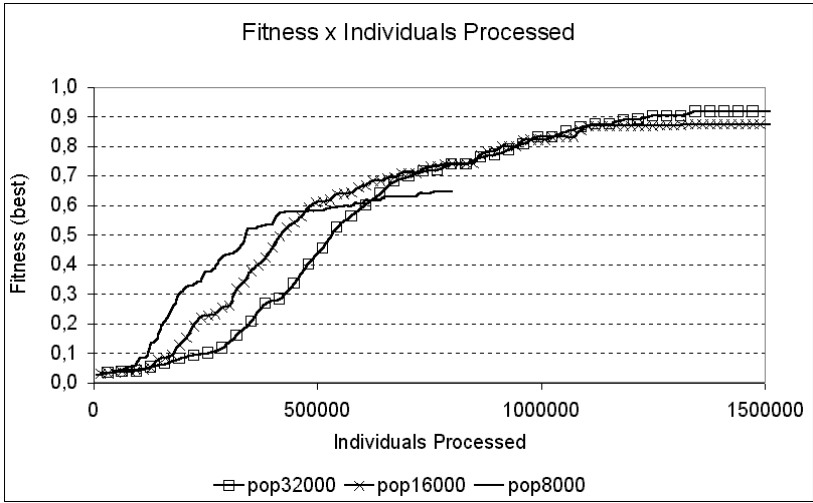


Fig. 6. Best fitness for each population (without adaptation)

6 Conclusions and Future Works

This work reinforces that some kind of dynamic adaptation of parameters is a very important issue in Evolutionary Computation algorithms. Specifically, it shows that the population size is a crucial parameter with a very high impact on the success probability.

The results obtained show that a control over the population size in Genetic Programming increases the robustness of the algorithm, making it less susceptible to failure. Using this approach, it is possible to start a run of any GP problem with any population size and still have a high level of confidence that a solution will be found.

A further study about the crossover rate and the tournament size will allow a conclusion about how they affect the performance of small populations.

Besides this, many other future research efforts are needed to provide a better understanding of parameters' impact and what can be done to adapt them automatically. Some of those future works are:

- Divide the process in two phases: training and execution. In the first one, the optimal parameters would be determined and, in the second, they would be used and no longer adapted.
- Instead of multiple populations that can be discarded, experiment the use of a single population with variable size.
- Alter the base of the counter, modifying the number of times that a population i has to be executed for a population $i+1$ to be created.
- Combine the control of the population size [15] with the method that adapts the genetic operator rates [10].

References

- [1] BICK, T. Self-adaptation in genetic algorithms. In *Proceedings of the First European Conference on Artificial Life* (pp. 263-271). Cambridge, USA: MIT Press, 1992.
- [2] BAGLEY, J. D. The behavior of adaptative systems which employ genetic and correlation algorithms. PhD Thesis. University of Michigan. 1967.
- [3] BANZHAF, W.; NORDIN, P. et al. Genetic programming, an introduction: on the automatic evolution of computer programs and its applications. Morgan Kaufmann, 1998.
- [4] DARWIN, C. On the origin of species by means of natural selection or the preservation of favored races in the struggle for life. London, UK: Murray, 1859.
- [5] DAVIS, L. Adapting operator probabilities in genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 61-69). San Mateo, USA: Morgan Kaufmann, 1989.
- [6] EIBEN, A.; HINTERDING, R.; MICHALEWICZ, Z. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation* (vol. 3, pp. 124-141). IEEE, 1999.
- [7] GREFENSTETTE, J. J. Optimization of control parameters for genetic algorithms. SAGE, A. P. (Ed.). *IEEE Transactions on Systems, Man and Cybernetics*. SMC-16(1). pp. 122-128. New York, USA: IEEE, 1986.
- [8] HARIK, G.; LOBO, F. A parameter-less genetic algorithm. Technical report 99009, Illinois Genetic Algorithm Laboratory. Illinois, USA: 1999.
- [9] HOLLAND, J. H. *Adaptation in natural and artificial systems*. MIT Press, 1992.
- [10] JULSTROM, B. A. Adaptative operator probabilities in a genetic algorithm that applies three operators. In *Proceedings of the 1997 ACM Symposium on Applied Computing*. pp. 233-238. New York, USA: ACM Press, 1997.
- [11] KOCH, T.; SCHEER, V. et al. A parallel, hybrid meta optimization for finding better parameters of an evolution strategy in real world optimization problems. *Proceedings of the Genetic and Evolutionary Computation Conference Workshop Program* (pp. 17-19). Morgan Kaufmann, 2000.
- [12] KOZA, J. R. *Genetic programming: on the programming of computers by means of natural selection*. MIT Press, 1992.
- [13] KOZA, J. R. *Genetic programming II: automatic discovery of reusable programs*. MIT Press, 1994.
- [14] LANGDON, W.B.; POLI, R. Why ants are hard. Technical report CSRP-98-4. THE University of Birmingham. School of Computer Science. 1998.
- [15] LOBO, F. G. The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic algorithm operation. PhD Thesis. Universidade Nova de Lisboa. Faculdade de Ciências e Tecnologia. Lisboa: 2000.
- [16] MERCER, R. E.; SAMPSON, J. R. Adaptative search using a reproductive meta-plan. *Kybernetes*. 7. pp. 215-228. 1978.
- [17] MITCHELL, M. *An introduction to genetic algorithms*. MIT Press, 1996.

- [18] SMITH, R. E.; SMUDA, E. Adaptively resizing populations: algorithm, analysis, and first results. *Complex Systems* (vol. 9 pp. 47-72). 1996.
- [19] SPINOSA, E. Adaptação dinâmica de parâmetros em Computação Evolucionária: o controle do tamanho da população em um sistema de Programação Genética. MSc Dissertation. Federal University of Paraná Computer Science Department. Curitiba, Brazil, 2002.
- [20] ZONGKER, D.; PUNCH, B. Lil-gp 1.01 user's manual. Michigan State University. <http://garage.cps.msu.edu/software/lil-gp/lilgp-index.html>.

The Correspondence Problem under an Uncertainty Reasoning Approach

José Demisio Simões da Silva^{1,2} and Paulo Ouverá Simoni^{2,3}

¹Instituto Nacional de Pesquisas Espaciais - INPE
Laboratório Associado de Computação e Matemática Aplicada - LAC
Av. dos Astronautas, 1758, São José dos Campos, 12227010, SP, Brazil

²Universidade Braz Cubas - UBC
Av. Francisco Rodrigues Filho, 1233, Mogi das Cruzes, SP, Brazil

³Universidade de Guarulhos - UnG
Praça Tereza Cristina, 1, Guarulhos, SP, Brazil

Abstract. In this paper, the Dempster-Shafer Theory for uncertainty reasoning is presented as a computation tool in designing a model to approach the correspondence problem in Computer Vision. In previous works (Silva and Simoni, 2001a; Silva and Simoni, 2001b) the proposed methodology showed its effectiveness in establishing the correspondence of a pair of images with similar brightness and contrast. In this paper, the efficiency of the uncertainty reasoning methodology is evaluated by applying the method to pairs of real world images with different brightness and contrast. Contextual and structural features of a point are treated as corresponding evidences. The Dempster's rule of combination is used to combine the existing evidences leading to an evidential interval for each candidate point. A search process maximizes the Belief on the combined evidences. The conducted experiments showed the robustness of the approach in establishing the correspondence in situations for which there is illumination and/or focus changes from one real world image to the other.

1 Introduction

The uses of spatially separated cameras make it possible for one to infer 3D information from images of a scene, in computer vision. Such 3D-information recovery requires the establishment of the correspondence among the images that assures the images represent the same phenomena in the scene. This problem is referred to as the correspondence problem in Computer Vision and it is considered to an ill-posed problem due to inherent features of the images.

In Anadan (1985), Barnard and Thompson (1980), and Jones (1997), different methods are commented that approach the correspondence problem, but a general method has not been achieved yet, due to the nature of the problem. Adaptive approaches include neural networks, regularization, learning strategies, optimization

techniques (Jones, 1997), and genetic algorithms (Saito and Mori, 1995; Silva and Simoni, 2000a; Silva and Simoni, 2000b; Silva and Simoni, 2001a).

The correspondence problem has been revisited recently, encouraged by the development of new stereo heads, with a large number of degrees of freedom, and the availability of more powerful personal computers. Such facts makes it possible to concentrate on the temporal integration of image information under the active vision (Jones, 1997).

The existing methods and algorithms to solve the correspondence problem are area-based, token-based or a hybridization of both. Area-based methods require the definition of a window size that may capture relevant information to be used as corresponding evidence. Token-based methods, in general, require a preprocessing phase for edge detection and token definition. Hybrid methods use a combination of both approaches. The features are compared by measuring the similarity by using Cross-Correlation, Sum of Squared Differences, Euclidean Distance, Hamming Distance, etc. New methodologies have been proposed that introduce new aspects and possibilities as in Kanade and Okutomi (1994), to satisfy the constraints for the correspondence problem, that is, compatibility, uniqueness, and map continuity (Marr, 1982). However, the methods generally fail in establishing the correspondence due to illumination changes, existing occlusion from one image to the other, or/and the size and shape of the windows used.

In order to achieve the goal of corresponding images, it is necessary first to extract and establish reference features. Area-based methods require the definition of a window for sub-image extraction, with as much relevant information as possible. The works of Kanade and Okutomi (1994), and Saito and Mori (1995) use adaptive windows to achieve the optimal window size that provides the closest disparity map in relation to its statistical model. In token-based methods, token extraction requires edge detection and token choice and description that may be time consuming. Hybrid methods combining both area and token features present the advantages and disadvantages of both area and token-based methods.

After establishing reference features, a search for a number of similar features in the other images is conducted that may result in a large number of candidates, from which a unique correspondence has to be chosen to satisfy the problem constraints (Marr, 1982). An area-based Genetic Algorithm (GA) search has been proposed by Saito and Mori (1995), to implement a search for an adequate window as described in Kanade and Okutomi (1994). This GA approach searches for a disparity map that optimizes both the compatibility of the corresponding points and the map continuity.

In previous works (Silva and Simoni, 2001a; Silva and Simoni, 2001b) we proposed a point wise hybrid approach to the correspondence problem, in which the Island Model Parallel Genetic Algorithm was used as a search method. The population of individuals represented possible correspondences, whose fitness was measured by combining the corresponding evidences under the Uncertainty Reasoning theory. The model was used to simultaneously establish multiple correspondences. Area features were restricted to a window around the reference point that represented the point context (*contextual* features). The extracted tokens formed the *structural* features of a point in its neighborhood. They were related to existing edges in the window to which the point belonged.

Such *contextual* and *structural* features were the corresponding evidences whose similarities were measured by applying difference, correlation, and distance metrics, to the same features of the reference point. In addition to the features of the points, the simultaneous correspondence of multiple (N) points also considered the *structural coherence constraint*, related to the geometry of the polygonal regions emerged from the interconnection of the N points. Thus, the correspondence process developed in two hierarchical levels (Silva and Simoni, 2000b): a low-level point correspondence that *satisfies the uniqueness and point compatibility constraints*, and a high-level correspondence among the polygonal regions *that satisfies the structural coherence constraint related to the map continuity constraint*. The process maybe influenced by the number of candidate points for each reference point, the complexity of the geometric features of the polygonal regions, and the existence of occlusion in the images. The parallel GA addressed the problem related to the large number of candidates.

By using an uncertainty reasoning based approach we attempted at addressing the problems related to the existence of illumination changes and occlusion in the images. The motivation lies in the fact that, intuitively, one may reason that even in the presence of such adverse conditions, there is a possibility to establish partial correspondences between the images, in the sense that some parts of the images may still be corresponded despite the fact there is not a way to estimate occlusion in the images. It is also intuitive, and stated by the constraints of the correspondence problem (Marr, 1982), that for a reference, only one individual in the solution space can be found. The similarity among the corresponding evidences of the candidate points is consensually combined by the Dempster-Shafer rule of combination. The experiments used a pair of real world images with similar illumination and focus conditions.

In this paper, the uncertainty reasoning model is further explored by applying it to pairs of real world images that present illumination and focus differences from one camera to the other, aiming at checking the efficiency of the method with pairs of images with a darker or blurred image. In these regards, the model will be applied to establish the correspondence between a pair of points.

This paper is organized as follows. Section 2 reviews the methodology for extracting features used in previous works (Silva and Simoni, 2000a; Silva and Simoni, 2000b; Silva and Simoni, 2001a; Silva and Simoni, 2001b). Section 3 introduces the Dempster-Shafer theory and its application to the correspondence problem. Section 4 describes the experiments conducted and presents some results and a comparison with the results in Silva and Simoni (2001b). Finally, section 5 presents the conclusions.

2 Corresponding Evidences

In Silva and Simoni (2001b) we proposed a point-wise approach to the correspondence problem in which contextual and structural features of a point were considered as evidences. Contextual features provide local information of the context of a point within a certain neighborhood. They are related to the micro area (within a

pre-defined window size) and the macro area (within a window whose dimensions are n times the dimensions of the micro area).

The structural features are binary edge elements (tokens) detected within a window (with the same dimensions of the micro area), which limits the number of features that can be detected. In the experiments conducted in this paper, the following binary features were considered: the vertical and horizontal lines; the principal and secondary diagonals; and the bottom right, top right, top left, and bottom left corners. All of these structures were related to image locations with salient contrast information.

By considering an image window, like the one in Figure 1 where a, b, c, d, e, f, g, h, and i represent pixel gray level intensities, the structural features are computed by calculating the differences among the pixels. For instance, the *Vertical Line* (VL) binary feature, is calculated by equation (1), where $F(.)$ is the *sign* function in Figure 1. Equations 2 through 8 are used to compute the seven other binary structures, that is, the *Horizontal Line* (HL), the *Principal Diagonal* (PD) and *Secondary Diagonal* (SD), the *Bottom Right Corner* (BRC), and the *Top Right Corner* (TRC), the *Top Left Corner* (TLC), and the *Bottom Left Corner* (BLC). In this paper, such computations are performed by *Perceptron* neural networks with weight insertion that represent prior knowledge for each different structure. Figure 1, for instance, illustrates a *Perceptron* used in the computation of the *Vertical Line* structure in a 3×3 window. Larger windows require redefinition of the *Perceptrons* for each structure.

VL	[0	$F(b-a)$	0	0	$F(e-d)$	0	0	$F(h-g)$	0]	(1)
HL	[0	0	0	$F(d-a)$	$F(e-b)$	$F(f-c)$	0	0	0]	(2)
PD	[$F(d-b)$	0	0	0	$F(g-c)$	0	0	0	$F(h-f)$]	(3)
SC	[0	0	$F(f-b)$	0	$F(i-a)$	0	$F(h-d)$	0	0]	(4)
BRC	[0	0	0	0	$F(e-i)$	$F(f-i)$	0	$F(h-i)$	0]	(5)
TRC	[0	$F(b-c)$	0	$F(e-c)$	$F(f-c)$	0	0	0	0]	(6)
TLC	[0	$F(b-a)$	0	$F(d-a)$	$F(e-a)$	0	0	0	0]	(7)
BLC	[0	0	0	$F(d-g)$	$F(e-g)$	0	$F(h-g)$	0	0]	(8)

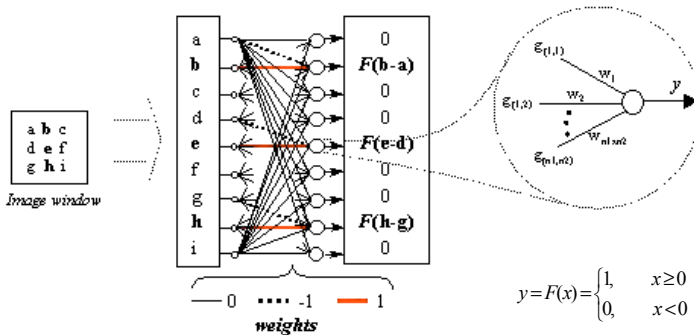


Fig. 1. The vertical line structure in a 3×3 window. Weights are inserted as shown

The *pattern of differences* and the *predominant structure* are two additional binary structures computed within a connected neighborhood of a point. Such computations,

Following feature extraction and similarity measurements, each candidate point j is assigned a vector with similarity measurements for 13 different matching criteria, together with the point line and pixel coordinates (l_j, p_j) .

$$Q_i = [l_j \ p_j \ C_{\text{macro}} \ N_1 \ N_2 \ N_3 \ N_4 \ N_5 \ N_6 \ N_7 \ N_8 \ N_9 \ N_{10} \ G \ C_{\text{micro}}] \quad (11)$$

3 Dempster-Shafer Theory in the Correspondence Problem

In Dempster-Shafer (DS) Theory, the *Universe of Discourse* (or *Frame of Discernment*) U is considered to consist of mutually exclusive alternatives that may correspond to an attribute value domain (Giarratano and Riley, 1994). For instance, in satellite image classification the set U may consist of all possible classes of interest. Each subset $S \subseteq U$ is assigned a basic probability $m(S)$, a belief $Bel(S)$, and a plausible belief (or plausibility) $Pls(S)$ so that:

$$m(S), Bel(S), Pls(S) \in [0, 1] \text{ and } Pls(S) \geq Bel(S) \quad (12)$$

The basic probability m represents the strength of an evidence. For example, for a group of pixels that belong to a certain class, m may represent the effect of the pixels as representative of a class. $Bel(S)$ summarizes all the reasons to believe S . $Pls(S)$ expresses how much one should believe in S if all currently unknown facts were to support S . Thus the true belief in S will be somewhere in the *belief interval* $[Bel(S), Pls(S)]$. The basic Probability Assignment m is defined as the function

$$m : 2^U \rightarrow [0, 1] \quad (13)$$

where $m(\emptyset) = 0$ and the sum of m over all subsets of U is 1 ($\sum_{S \subseteq U} m(S) = 1$). For a given basic probability assignment m , the belief (Bel) of a subset A of U is the sum of $m(B)$ for all subsets B of A , and the plausibility (Pls) of a subset A of U is

$$Pls(A) = 1 - Bel(A^c) \quad (14)$$

where A^c is the complement of A in U .

Different beliefs are combined by the rule of combination, that states two basic probability assignments m_1 and m_2 are combined into a third basic probability assignment by the normalized orthogonal sum $m_1 \oplus m_2$ defined as:

$$m_1 \oplus m_2(A) = \frac{\sum_{X \cap Y = A} m_1(X)m_2(Y)}{1 - k}, \quad k = \sum_{X \cap Y = \emptyset} m_1(X)m_2(Y) \quad (15)$$

Equation (15) is the original rule of combination for basic probabilities, however it is computationally expensive. A faster alternative to the combination of evidences is given in Haddawy (1987). Equations (16) and (17) directly combine beliefs and plausibilities that are directly assigned to the existing corresponding evidences. The step by step application of the methodology to pieces of a pair of images is illustrated in Silva and Simoni (2001b).

$$\text{Bel}(S) = 1 - \frac{(1 - \text{Bel}_1(S))(1 - \text{Bel}_2(S))}{1 - [\text{Bel}_1(S)(1 - \text{Pls}_2(S)) + (1 - \text{Pls}_1(S))\text{Bel}_2(S)]} \quad (16)$$

$$\text{Pls}(S) = \frac{\text{Pls}_1(S)\text{Pls}_2(S)}{1 - [\text{Bel}_1(S)(1 - \text{Pls}_2(S)) + (1 - \text{Pls}_1(S))\text{Bel}_2(S)]} \quad (17)$$

In the methodology, the application of Dempster-Shafer calculus to the correspondence problem, is preceded by evidence (similarity measurements) extraction. One point is chosen as a reference in an image, whose *context* and *structural* features are computed. In the order image, several points (*candidates* to correspondence) are picked that lie in similar *context* (that is, the correlation of its micro and macro areas to the reference $\bar{S}$ are $C_{\text{micro}} > 0.5$ and $C_{\text{macro}} > 0.5$, respectively) under the epipolar line constraint. The *candidate* points form the *Universe of Discourse* that is dynamically established for each new reference point. In any case, the set of possible solutions is a singleton, that is, one element set.

An *uncertain factor* is assigned to the problem due to the existence of possible occlusion and differences in illumination. Belief is directly assigned to the available information, that is, no probability is assigned to evidences that contradict the hypothesis, contrary to the probability theory. The Dempster-Shafer calculus combines the beliefs and plausibilities resulting in a belief and a plausibility in the combined evidence that represents a consensus on the correspondence. The present model aims at maximizing the belief in the combined evidences.

4 Implementation and Results

Previous work (Silva and Simoni, 2001b) showed the adequacy of the uncertainty reasoning approach to the correspondence problem on real world images (Figure 3.a). This paper shows results of the investigation of the robustness of the methodology when applied to image pairs that present differences in illumination or focus from one image to the other, that result in blurred or dark images.

The conducted experiments used the same images of Silva and Simoni (2001b). The effects of dark and blurred images were simulated by applying contrast enhancement and low passing filtering techniques. Changes in illumination were simulated by reducing the brightness of the right image to 75%, 50%, 40% and 25% of the original brightness. In order to blur the images, the mean and Gaussian low pass filters below were used.

Mean	Gaussian
$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$	$\begin{bmatrix} 1/15 & 2/15 & 1/15 \\ 2/15 & 3/15 & 2/15 \\ 1/15 & 2/15 & 1/15 \end{bmatrix}$

The results in Figures 3.b), 4.a) and 4.b) were obtained by applying the method to the filtered images. Figure 5 shows the results when the brightness of the right image was reduced to 50% of its original brightness. Other experiments were conducted for

75%, 40% and 25% of the original contrast. The results were not satisfactory for the last two situations. The belief and the plausibility on the combined evidence, for all the candidate points, are depicted in the graphics of Figures 3 through 5. Table 1 shows the line and pixel coordinates of the points found in each situation, as well as, the error to the coordinates found in Silva and Simoni (2001b), taken as the solution to the problem (line 1 of Table 1).

It is to be noted that fewer candidate points resulted in each of the experiments conducted, when compared to the original experiment in Silva and Simoni (2001b) (see the horizontal axis in the graphics of Figures 3 through 5).

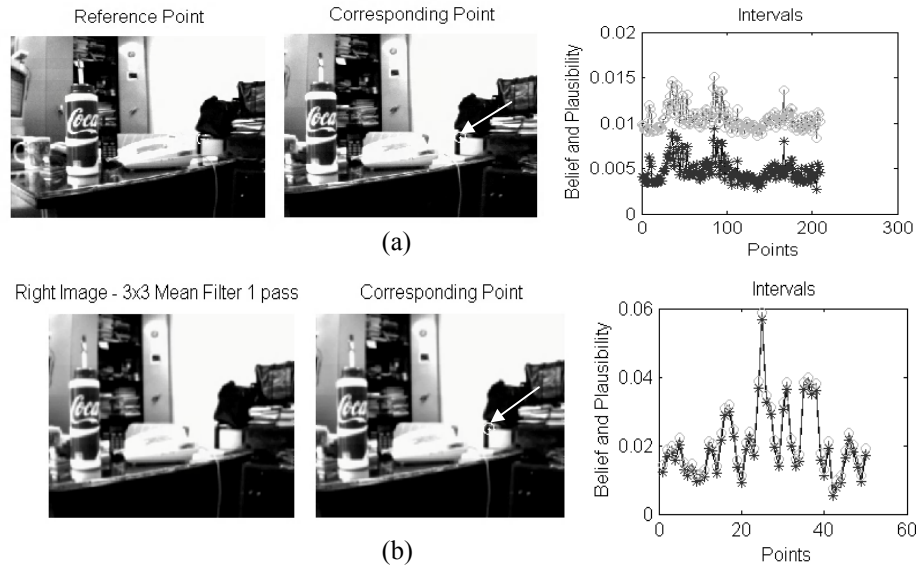


Fig. 3. (a) Correspondence found between images with similar brightness [From Silva and Simoni (2001b)]. (b) Correspondence found - Right image filtered (1 pass) with the 3x3 low pass mean filter

5 Conclusion

This paper presents results of an ongoing research that investigates the use of an uncertainty reasoning and genetic algorithms to approach the correspondence problem in Computer Vision. The present work investigates the use of the methodology developed in previous work (Silva and Simoni, 2001b) to pairs of images with differences in illumination and focus, under the same contextual and structural evidences, to establish the correspondence among points. Perceptron neural networks perform feature extraction in the neighborhood of points that belong to edges in the image.

Table 1 summarizes the results from which it is possible to infer there is a certain degree of robustness of the uncertainty reasoning approach to the correspondence problem, when applied to dark or blurred indoor images. The success achieved by applying the methodology to these degraded images, shows the importance of structural features in the process.

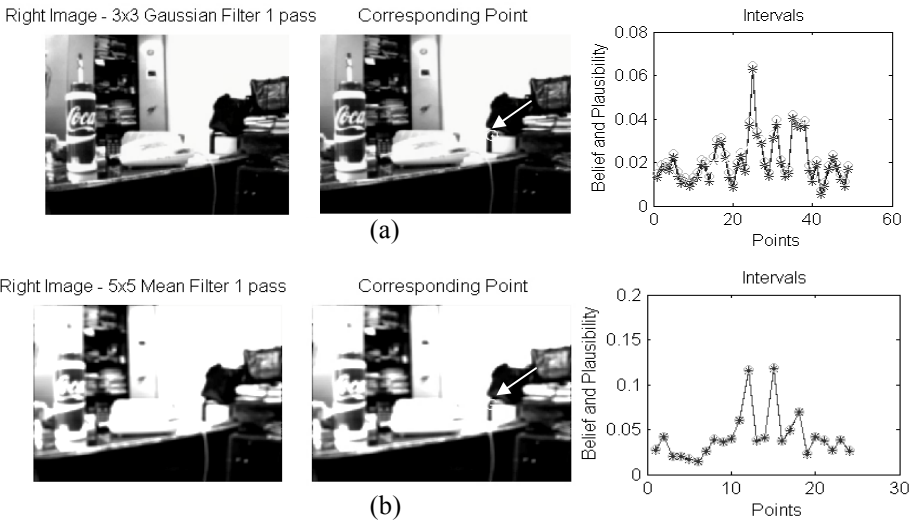


Fig. 4. (a) Correspondence found - Right image filtered (1 pass) with the 3x3 low pass Gaussian filter. (b) Correspondence found - Right image filtered (1 pass) with the 5x5 low pass mean filter

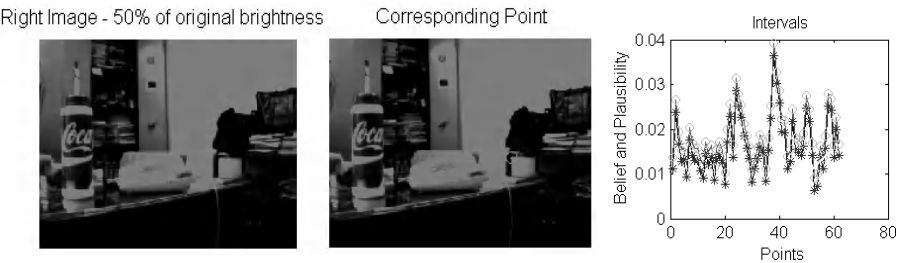


Fig. 5. Correspondence found - Right image with 50% of its original brightness

Table 1. Line and Pixel coordinates of the corresponding point found in the conducted experiments

Experiment	line	pixel	error line	error pixel
Results in (Silva and Simoni, 2001b) - Figure 3	137	224		
3x3 Mean filter (1 pass) - Figure 4	137	224	0	0
3x3 Mean filter (2 passes)	137	224	0	0
3x3 Gaussian filter (1 pass)- Figure 5	137	224	0	0
3x3 Gaussian filter (2 passes)	137	223	0	1
5x5 Mean filter (1 pass) - Figure 6	137	224	0	0
75% of original brightness	138	224	1	0
50% of original brightness - Figure 7	138	224	1	0

It is to be noted, however, that in this paper the images used were artificially degraded from their original brightness/contrast relation. In addition, only one point was corresponded, while in Silva and Simoni (2001b) simultaneous points were considered. Future work will be conducted to verify the robustness of the uncertainty reasoning methodology to correspond simultaneous points in real world images with differences in illumination and focus.

Acknowledgement

This research has been supported by CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico (Process Number 68.0050/01-9), a Brazilian funding agency for the Science and Technology Development.

References

- [1] Anadan, P.A., Review of Motion and Stereopsis Research. COINS Technical Report 85-52, University of Massachusetts at Amherst, December 1985.
- [2] Barnard, S. T., Thompson, W. B., Disparity Analysis of Images, IEEE PAMI-2(4), July 1980, pp.333-334.
- [3] Giarratano, J., Riley, G., Expert Systems: Principles and Programming. Boston, PWS Publishing Company, 1994. 644 p.
- [4] Haddawy, P. A Variable Precision Logic Inference System Employing the Dempster-Shafer Uncertainty Calculus. Urbana - Illinois, MS Thesis, UILU-ENG-86-1777, 1987.
- [5] Jones, G. A., Constraint, Optimization, and Hierarchy: Reviewing Stereoscopic Correspondence of Complex Features. Computer Vision and Image Understanding 65(1), 1997, pp. 57-78.
- [6] Kanade, T., Okutomi, M., A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment. IEEE PAMI, 16(9), September 1994, pp. 920-932.
- [7] Marr, D., Vision. Freeman. San Francisco, CA. 1982.
- [8] Saito, H., Mori, M., Application of Genetic Algorithms to Stereo Matching of Images, Pattern Recognition Letters 16, 1995, pp. 815-821.
- [9] Silva, J.D.S., Simoni, P.O., Bharadwaj, K.K., A Genetic Algorithm for the Stereo Correspondence Problem in Computer Vision. Proceedings of the IASTED International Conference on Computer Graphics and Imaging. Calgary: IASTED/ACTA Press, 2000a, v.1, p. 20-25.
- [10] Silva, J.D.S., Simoni, P.O., Bharadwaj, K.K., A Hierarchical Approach to Multiple-Point Correspondences in Stereo Vision Using a Genetic Algorithm Search. 6TH Intl. Conf. on Soft Computing, IIZUKA, Fukuoka. 2000, v.1, p. 125-130.
- [11] Silva, J.D.S., Simoni, P.O., The Island Model Parallel GA and Uncertainty Reasoning in the Correspondence Problem. In: IJCNN, Washington, DC. Proc. of IJCNN2001, July 2001, v.1, p. 2247-2252.

- [12] Silva, J.D.S., Simoni, P.O., Uncertainty Reasoning in the Correspondence Problem. In: IASTED -International Conference on Visualization, Imaging, and Image Processing, 2001, Marbella. Proceedings of the IASTED VIIP 2001. Anaheim, CA: ACTA Press, September 2001, v.1, p. 647-652.

Random Generation of Bayesian Networks

Jaime S. Ide and Fabio G. Cozman

Escola Politécnica, University of São Paulo
Av. Prof. Mello Moraes, 2231 - São Paulo, SP - Brazil
`jaime.ide@poli.usp.br`
`fgcozman@usp.br`

Abstract. This paper presents new methods for generation of random Bayesian networks. Such methods can be used to test inference and learning algorithms for Bayesian networks, and to obtain insights on average properties of such networks. Any method that generates Bayesian networks must first generate directed acyclic graphs (the “structure” of the network) and then, for the generated graph, conditional probability distributions. No algorithm in the literature currently offers guarantees concerning the distribution of generated Bayesian networks. Using tools from the theory of Markov chains, we propose algorithms that can generate uniformly distributed samples of directed acyclic graphs. We introduce methods for the uniform generation of multi-connected and singly-connected networks for a given number of nodes; constraints on node degree and number of arcs can be easily imposed. After a directed acyclic graph is uniformly generated, the conditional distributions are produced by sampling Dirichlet distributions.

1 Introduction

In this paper we describe a solution to a problem that is very simple to state, but very hard to solve. Our problem is to randomly generate Bayesian networks *with an uniform distribution*. Why is this useful? Two points should suffice to indicate the need for randomly generated networks with a uniform distribution:

1. Many algorithms for inference and learning using Bayesian networks must be tested, and uniformly generated Bayesian networks offer a natural way to produce “unbiased” experiments.
2. Properties of Bayesian networks (such as the average number of connected components, average number of independent variables) are usually very hard to derive analytically, and uniformly generated Bayesian networks can be used for exploring such questions empirically.

Because Bayesian networks occupy a prominent position as a model for uncertainty in artificial intelligence [3], it would seem that algorithms for uniform generation of Bayesian networks would be easily available. Alas, this is not the case. One reason for this is that Bayesian networks are composed of directed acyclic graphs, and it is very hard to represent the space of such graphs. Consequently, it is not easy to guarantee that a given method actually produces

a uniform distribution in that space. Another reason is that usually Bayesian networks are sparsely connected; to be able to investigate properties that are relevant to practical problems, we must generate directed acyclic graphs subject to constraints on number of arcs, degree of nodes, number of parents for nodes — generating such graphs while guaranteeing uniform distributions is quite a challenge. In this paper we present algorithms for uniformly generating random directed acyclic graphs through Markov chains.

In Section 2 we review the theory of Bayesian networks and the basic concepts used in this paper. We review the problem of Bayesian network generation and existing approaches in Section 3. In Section 4 we develop algorithms for generation of random directed acyclic graphs. We demonstrate that our methods can uniformly generate multi-connected and singly-connected Bayesian networks for a given number of nodes and limits on node degree and number of arcs (other constraints can be imposed by modifying the basic algorithms). In Section 5 we describe an implementation and tests with our methods.

2 Bayesian Networks and Graphs

This section summarizes the theory of Bayesian networks and introduces terminology used throughout the paper. All random variables are assumed to have a finite number of possible values. Denote by $p(X)$ the probability density of X , and by $p(X|Y)$ the probability density of X conditional on values of Y .

A Bayesian network represents a joint probability density over a set of variables $\mathbf{X}$ [3]. The joint density is specified through a directed acyclic graph.

A directed graph is composed of a set of nodes and a set of arcs. An arc (u, v) goes from a node u (the *parent*) to a node v (the *child*). A *path* is a sequence of nodes such that each pair of consecutive nodes is adjacent. A path is a *cycle* if it contains more than two nodes and the first and last nodes are the same. A cycle is *directed* if we can reach the same nodes while following arcs that are in the same direction. A directed graph is *acyclic* (it is a *DAG*) if it contains no directed cycles. A graph is *connected* if there exists a path between every pair of nodes. A graph is *singly-connected* if there exists exactly one path between every pair of nodes; otherwise, the graph is *multiply-connected* (or multi-connected for short). A singly-connected graph is also called a *polytree*. An *extreme sub-graph* of a polytree is a sub-graph that is connected to the remainder of the polytree by a single path.

In a Bayesian network, each node of its graph represents a random variable X_i in $\mathbf{X}$. The *parents* of X_i are denoted by $\text{pa}(X_i)$. The semantics of the Bayesian network model is determined by the *Markov condition*: Every variable is independent of its nondescendants nonparents given its parents. This condition leads to a unique joint probability density [6]:

$$p(\mathbf{X}) = \prod_i p(X_i | \text{pa}(X_i)). \quad (1)$$

Every random variable X_i is associated with a conditional probability density $p(X_i | \text{pa}(X_i))$. Figure 1 depicts examples of DAGs as Bayesian networks.

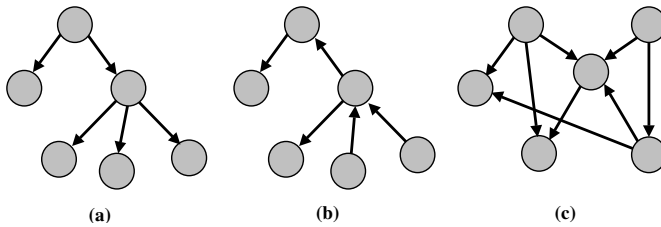


Fig. 1. Bayesian networks: (a) Tree, (b) Polytree, (c) Multi-connected Bayes net

3 Generating Bayesian Networks

To generate random Bayesian networks, the obvious method is to generate a random DAG, and then to generate the conditional probability distributions for that graph.

Given a DAG, it is relatively easy to generate uniformly distributed random conditional distributions. Suppose then that we are generating the distribution $p(X|\text{pa}(X))$ for a fixed value of $\text{pa}(X)$, where X has k values. A general method is to define a Dirichlet distribution over the k values of X with priors $(\alpha_1, \alpha_2, \dots, \alpha_k)$; we then have to sample from k Gamma distributions and normalize these k samples [7].¹ If we want to generate a uniform distribution, we simply set all α 's to 1. (It should be noted that, for the specific problem of uniformly generating distributions, Caprile has proposed a more efficient method than the one based on Gamma distributions [1].)

The real difficulty is to generate random DAGs that are uniformly distributed. Many authors have used random graphs to test Bayesian network algorithms, generating these graphs in some ad hoc manner. A typical example of such methods is given by the work of Xiang and Miller [9]. By creating some heuristic graph generator, it is usually impossible to guarantee any distribution on the generated networks; consequently, any conclusion reached by using the generated graphs may be biased in some unknown direction. On the other hand, it can be argued that any generator that produces a uniform distribution on the space of *all* DAGs is not very useful. The problem is that practical Bayesian networks usually have a reasonably small degree; if a generator produces graphs that are too dense, these graphs are not representative examples of Bayesian networks. So, we must generate graphs uniformly over the space of graphs that are *connected*, *acyclic*, and *not very dense*. We assume that the number of arcs in a graph is a good indicator of how dense the graph is, so we assume that our problem is to uniformly generate connected DAGs with restrictions either on node degrees or on number of arcs. Other constraints can be imposed using straightforward modifications of our algorithms.

A type of Bayesian network that is of great practical interest is represented by polytree structures [6]. Polytrees seem to be sufficiently general to represent

¹ Thanks to Nir Friedman for pointing this method to us.

many real-world problems while being amenable to polynomial algorithms for computation of probabilities. So, we can establish another problem: to uniformly generate polytrees with n nodes. To the best of our knowledge, there exists no algorithm for random generation of polytrees so far.

4 Markov Chains for Generating Connected DAGs

Our approach to generate random graphs is to use Markov chains. We are directly inspired by the work of Melançon et al on random graph generation [4]. The main difference between Melançon et al's work and ours is that they let their graphs be disconnected, a detail that makes considerable difference in the correctness proofs.

A few necessary concepts are briefly reviewed here. Consider a Markov chain over finite domains [2], and $P = (p_{ij})_{i,j=1}^N$ to be a $N \times N$ matrix representing transition probabilities, where, $p_{ij} = Pr(X_{t+1} = j | X_t = i)$, for all t . The s -step transition probabilities is given by $P^s = p_{ij}^{(s)} = Pr(X_{t+s} = j | X_t = i)$, independent of t . We denote the initial distribution of the Markov chain by the vector $\pi^{(0)}$. A Markov chain is *irreducible* if for all i, j , there is s that satisfies $p_{ij}^{(s)} > 0$. A Markov chain is irreducible if and only if all pair of states intercommunicate. A Markov chain is *aperiodic* if the greatest common denominator of all s such that $p_{ii}^{(s)} > 0$ is $d = 1$. Aperiodicity is ensured when $p_{ii} > 0$. A Markov chain is *ergodic* if there exists a vector π (the stationary distribution) satisfying $\lim_{s \rightarrow \infty} p_{ij}^{(s)} = \pi_j$, for all i and j . Any finite chain that is aperiodic and irreducible is ergodic. A non-negative transition matrix is called *doubly stochastic* if the rows and columns sum one (that is, if $\sum_{j=1}^N P_{ij} = 1$ and $\sum_{i=1}^N P_{ij} = 1$). A Markov chain with a doubly stochastic transition matrix has a stationary distribution that is uniform.

We can generate random graphs by simulating Markov chains. To have a Markov chain, it is enough that we can “move” from a graph to another graph in some probabilistic way that depends only on the current graph. Such a Markov chain will be irreducible if it can reach any graph from any graph. Also, the chain will be aperiodic if there exists a self-loop probability, i.e. there is a chance that the next generated graph is the same as the current one. If the moves are governed by a doubly stochastic transition matrix, the unique stationary distribution for the process is uniform over the space of possible moves.

4.1 Generating Multi-connected DAGs

Consider a set of n nodes (from 0 to $n - 1$) and the Markov chain described by Algorithm 1. We start with a connected graph. The loop between lines 3 and 7 construct the next state from the current state (this procedure defines a transition matrix). Our transitions are limited to 2 operations: adding and removing arcs provided the graph is still acyclic and connected. If we did not need to keep the graph connected, the following theorems would be immediate as pointed

Algorithm 1: Generating Multi-connected DAG's

Input: number of nodes (n), number of iterations (N).**Output:** Return a connected DAG with n nodes.

01. Initialize a simple ordered tree with n nodes, where all nodes have just one parent, except the first one that does not have any parent;
 02. Repeat the next loop N times:
 03. Generate uniformly a pair of distinct nodes i and j ;
 04. If the arc (i, j) exists in the actual graph, delete the arc, provided that the underlying graph remains connected;
 05. else
 06. Add the arc, provided that the underlying graph remains acyclic;
 07. Otherwise keep the same state;
 08. Return the current graph after N iterations.
-

Fig. 2. Algorithm for Generating multi-connected DAGs

out by Melançon et al. We have decided to present detailed proofs, resorting to constructive arguments where possible; the proofs can work as guiding tools if the reader wishes to modify the constraints imposed on graphs (for example, to limit the number of parents of a node).

Theorem 1 *The transition matrix defined by the Algorithm 1 is doubly stochastic.*

Proof. Note that we have constructed our chain to have a symmetric transition matrix; paths between two states have the same probability in both directions. There is a self-loop probability (line 7) that one minus the probability of other moves. Therefore, rows and columns of the transition matrix add to one. QED

Theorem 2 *The Markov chain generated by algorithms 1 is irreducible.*

Proof. A Markov chain is irreducible if any two states of this chain intercommunicate, that is, there is a probability from any state reach another state. Suppose that we have a multi-connected DAG with n nodes; if we prove that from this graph we can reach a simple ordered tree (Figure 3), the opposite transformation is also true, because of the symmetry of our transition matrix — and therefore we could reach any state from any other. We start by finding a loop cutset and removing enough arcs to obtain a polytree from the multi-connected DAG [6]. For each pair of extreme sub-graphs of the polytree, we have three possible cases described at Figure 4. In all three cases, we can add an arc between the last node of an extreme sub-graph and the first node, and remove the arc as depicted in the figure. Doing this we get a unique extreme sub-graph. If we have more than 2 extreme sub-graphs connected to a node, we repeat this process by pairs; we can do this recursively until get a simple polytree.

Now that we have a simple polytree, we want to get a simple tree, i.e. all arcs directed in one direction. Starting at the right extreme sub-graph of this simple

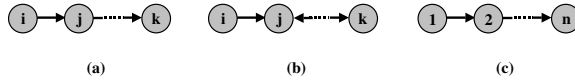


Fig. 3. (a) Simple tree, (b) Simple polytree, (c) Simple ordered tree

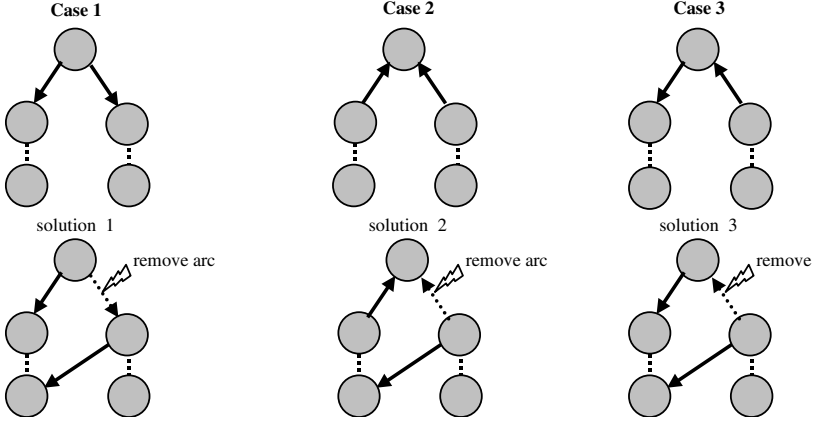


Fig. 4. Three possible cases for transforming a polytree into a simple polytree

polytree, we have to invert all arcs that are directed to the left. We run over all arcs, starting at the right side; if an arc is directed to the right, it does not need to be inverted; otherwise, we have 2 cases (Figure 5). Suppose that we have three nodes i, j, k . Add an arc between nodes i and k with appropriate direction, invert (remove and add) the arc (j, i) and at the end remove arc between i and k . Repeat this process until all arcs are processed. Notice that in the last arc we only have one possibility. At the end we get a simple tree.

The last step is to get to a simple ordered tree from the simple tree. The idea of the ordering process is illustrated in Figure 6. Start at node 0 and go on until the last node $(n - 1)$. Suppose that j is the processed node; add a possible arc $(p, 0)$ and remove the arc (i, j) (step 2); then add an arc (i, k) and remove

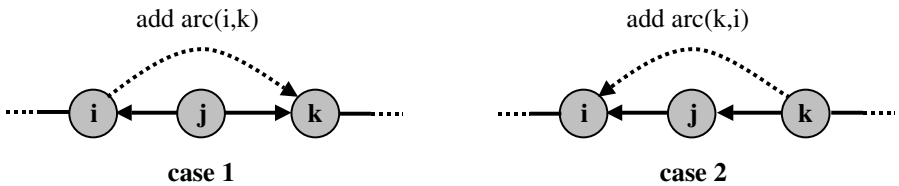


Fig. 5. Two possible cases for transforming a simple polytree into a simple tree (arcs are inverted to the right)

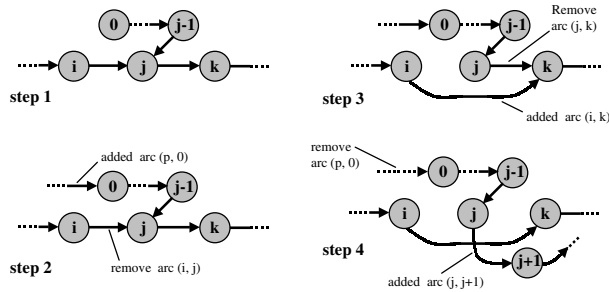


Fig. 6. Basic moves to obtain a simple ordered tree

arc (j, k) (step 3); the last step is to add an arc $(j, j+1)$ directed to the next in order and to remove arc $(p, 0)$. So, from any multi-connected DAG, it is possible to reach a simple ordered tree. The opposite proof is analogous. Consequently, we have that from any multi-connected DAG is possible to reach any other, i.e. this Markov chain is irreducible. QED

Theorem 3 *The Markov chain generated by Algorithm 1 is aperiodic.*

Proof. In any state there is an arc that will make the graph cyclic, so it is always possible to stay at the same state (there is a self-loop probability greater than zero). QED

Theorem 4 *The Markov chain generated by Algorithm is ergodic and its unique stationary converges to a uniform distribution.*

Proof. Follows from the previous theorems. QED

It is important to note that additional requirements, such as limitations on the number of arcs or on maximum degree, can be easily added to line 6. The transition matrix probabilities will change, but the proofs all carry through without problems.

4.2 Generating Polytrees

The process of generating polytrees is similar to Algorithm 1; we focus on the differences between algorithms and do not give a detailed description of properties and proofs. Consider again n nodes, and the transition matrix defined by Algorithm 2.

In line 1 we start with a simple ordered tree (this is a valid polytree). The loop between line 3 to 7 is the construction of the transition matrix. Line 4 ensures a self-loop probability greater than zero, to produce an aperiodic Markov chain. Line 6 is important to obtain symmetry for the transition matrix. Figure 8 illustrates a transition process between two neighbor states. Suppose that at

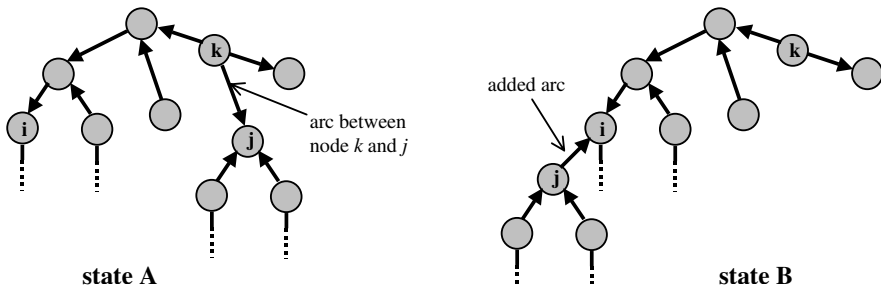
Algorithm 2: Generating Polytree**Input:** number of nodes (n), number of iterations (N).**Output:** Return a polytree with n nodes.

01. Initialize a simple ordered tree with n nodes as in Algorithm 1.
02. Repeat the next loop N times:
 03. Generate uniformly a pair of distinct nodes i and j ;
 04. If the arc (i, j) exists in the actual graph, keep the same state;
 05. else
 06. Invert the arc with probability $1/2$ to (j, i) , and then
 07. Find the predecessor node k in the path between i and j , remove the arc between k and j , and add an arc (i, j) or arc (j, i) depending on the result of line 06.
08. Return the current graph after N iterations.

Fig. 7. Algorithm for generating polytrees

state A we obtain the arc (i, j) with probability $p = 1/(n(n-1))$. As described in line 7, through a “remove and add” operation we get to a state B with probability $p_{AB} = 1/(2n(n-1))$. Note that the opposite transition state B to state A has the same probability $p_{BA} = p_{AB} = 1/(2n(n-1))$. This is possible because of the 50 percent probability factor (line 6). Therefore, Algorithm 2 produces a doubly-stochastic matrix just as Algorithm 1.

In Algorithm 1, “add” and “remove” operations are distinct, while at Algorithm 2, these operations are combined (line 7), because we cannot remove arcs from a polytree and keep it connected, and we cannot add arcs to a polytree and keep it as a polytree. The proof for irreducibility for Algorithm 1 follows the proof of Theorem 2. The operation in line 7 is simply a composition of operations, and it is easy to see that any polytree intercommunicates with any other. In addition, the “invert” operation makes it easy to invert arcs direction. We therefore have an aperiodic irreducible Markov chain whose state space contains

**Fig. 8.** Example of transition: the polytree is cut in two parts; a new polytree is constructed merging these parts randomly, through a single “add and remove” operation

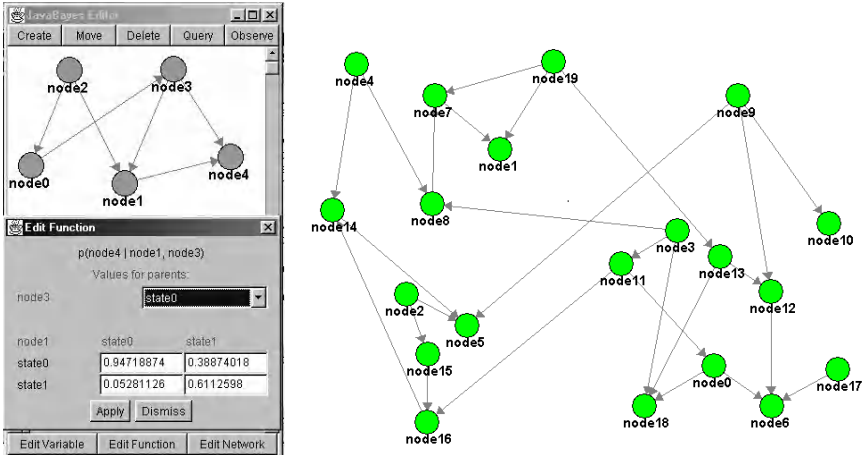


Fig. 9. Random Bayesian networks (a) with 5 nodes, showing random distribution; (b) with 20 nodes. Networks viewed in the JavaBayes system

all possible polytrees with n nodes, and that converges to a uniform stationary distribution.

5 Experimental Results

The algorithms for generating Bayes net structures and probability functions have been implemented in Java. The resulting program is called BNGenerator and is freely available under the GNU license. The program saves generated networks in the XML format read by the freely distributed JavaBayes system. In figure 9, we have the graphical representation of networks generated with different parameters. In figure 10, we have a simple histogram of samples generated with 4 nodes, illustrating that the networks have a uniform distribution.

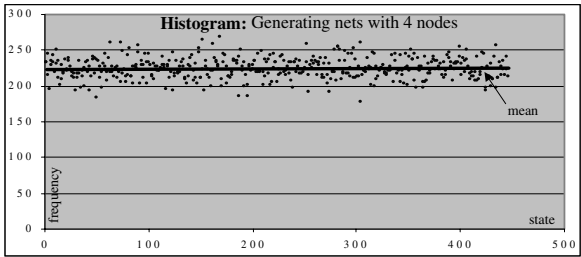


Fig. 10. Histogram of 100.000 generated nets with 4 nodes

6 Conclusion

We can summarize this paper as follows: we have introduced algorithms for generation of uniformly distributed random Bayesian networks, both as multi-connected networks and polytrees. Our algorithms are flexible enough to allow specification of maximum numbers of arcs and maximum degrees, and to incorporate any of the usual characteristics of Bayesian networks. We suggest that the methods presented here provide the best available scheme at the moment for producing valid tests and experiments with Bayesian networks. A disadvantage of our methods, compared to existing ad hoc schemes, is that many networks have to be generated before a sample can be taken (that is, it is necessary to wait for the Markov chains to converge, so the value of N in the algorithms must be high). In our implementation we have observed that the algorithms are fast, so we can easily wait for thousands of iterations before obtaining a sample.

Acknowledgements

We thank Nir Friedman for suggesting the Dirichlet distribution method, and Robert Castelo for pointing us to Melançon et al's work. We thank Guy Melançon for confirming that the idea of Algorithm 1 was sound and for making his DagAlea software available. We also thank Jaap Suermondt and Alessandra Potrich for providing important ideas, and Y. Xiang, P. Smets, D. Dash, M. Horsh, E. Santos, and B. D'Ambrosio for suggesting valuable procedures.

References

- [1] Caprile, B.: Uniformly Generating Distribution Functions for Discrete Random Variables (2000). 368
- [2] Gamerman, D.: Markov Chain Monte Carlo. Stochastic simulation for Bayesian inference. Texts in Statistical Science Series. Chapman and Hall, London (1997). 369
- [3] Jensen, F. V.: An Introduction to Bayesian Networks. Springer-Verlag, New York (1996). 366, 367
- [4] Melançon, G., Bousque-Melou, M.: Random Generation of Dags for Graph Drawing. Dutch Research Center for Mathematical and Computer Science (CWI). Technical Report INS-R0005 February (2000). 369
- [5] Chartrand, G., Oellermann, O. R.: Applied and Algorithmic Graph Theory. International Series in Pure and Applied Mathematics. McGraw-Hill, New York (1993).
- [6] Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufman (1988). 367, 368, 370
- [7] Ripley, B. D.: Stochastic Simulation. Wiley Series in Probability and mathematical Statistics. John Wiley and Sons, Inc., New York (1987). 368
- [8] Sinclair, A.: Algorithms for Random Generation and Counting: A Markov Chain Approach. Progress in Theoretical Computer Science. Birkhäuser, Boston (1993).

- [9] Xiang, Y., Miller, T.: A Well-Behaved Algorithm for Simulating Dependence Structures of bayesian Networks. *International Journal of Applied Mathematics*, Vol. 1, No. 8 (1999), 923–932. [368](#)

Evidence Propagation in Credal Networks: An Exact Algorithm Based on Separately Specified Sets of Probability

José Carlos F. da Rocha^{1,2} and Fabio G. Cozman¹

¹ Escola Politécnica, Universidade de São Paulo
Av. Prof. Mello Moraes, 2231, 05508-900, São Paulo, SP, Brazil
fgcozman@usp.br

<http://www.poli.usp.br/p/fabio.cozman.html>

² Deinfo, Universidade Estadual de Ponta Grossa
84010.790 Ponta Grossa, PR, Brazil
jrocha@uepg.br
<http://www.deinfo.uepg.br/jrocha>

Abstract. Probabilistic models and graph-based independence languages have often been combined in artificial intelligence research. The Bayesian network formalism is probably the best example of this type of association. In this article we focus on graphical structures that associate graphs with sets of probability measures — the result is referred to as a *credal network*. We describe credal networks and review an algorithm for evidential reasoning that we have recently developed. The algorithm substantially simplifies the computation of upper and lower probabilities by exploiting an independence assumption (*strong independence*) and a representation based on separately specified sets of probability measures. The algorithm is particularly efficient when applied to poly-tree structures. We then discuss a strategy for approximate reasoning in multi-connected networks, based on conditioning.

1 Introduction

Graphical languages associated with probabilistic models have been commonly used in artificial intelligence to represent independence assumptions and to reason about uncertainty. In this context, Bayesian networks are the most popular formalism [17]. A Bayesian network is a directed acyclic graph where each node represents a random variable and each arc denotes a direct probabilistic dependency. Each node X contains a table, called a conditional probability table (CPT), that stores the conditional distribution $p(X|pa(X))$, where $pa(X)$ denotes the parents of X in the graph. A Bayesian network encodes a single joint distribution over all variables in the network [6]. Inference techniques such as the junction tree algorithm [15] and variable elimination [10] produce posterior marginal probabilities for any variable in the network.

What if an agent wishes to handle imprecision and uncertainty about the numerical parameters of CPTs? This can be useful when: i) the members of a group

of experts involved in a knowledge engineering process intensely disagree about some CPT values; ii) the agent wants to represent and reason about beliefs that are incomplete or imprecise (for example, “the probability of rain is between 0.2 and 0.3”); iii) a number of experiments was made to estimate CPTs, but those experiments were not enough to obtain point estimates for every CPT entry; iv) the agent wants to perform sensitivity analysis to study how inferences are affected by variations in parameters. In some cases it may be possible or profitable to introduce hierarchical priors or second-order probabilities; here instead we focus on situations where sets of probabilities are used to model imprecision and uncertainty on probability values.

Credal networks associate graphical models with sets of probabilities [2, 8, 19]. The topology of a credal network can be used to represent several concepts of conditional independence; here we employ the concept of *strong independence*. Currently, algorithms for inference in credal networks with strong independence and general topology are very limited. Existing algorithms essentially search exhaustively over all joint probability distributions that can be generated from the sets of probability measures.

This article explores an strategy to implement credal network inference more efficiently than exhaustive search. The idea is first to encode credal networks using a representation that is directly connected to the way credal networks are usually specified in practice, and then to use this representation to reduce computations. We focus on sets of probability measures that are *separately specified*, as defined later. We have developed an algorithm, called *separable variable elimination* (SVE), that uses such sets to obtain computational gains. We have presented a thorough derivation of SVE elsewhere [18]. The objective of this paper is to present a detailed description of some inner operations of SVE and to discuss tests with random networks. We also investigate approximations that can be used in multi-connected networks.

The text is organized as follows. Section 2 introduces credal sets and the strong independence concept. Section 3 presents the credal network formalism. Section 4 describes the SVE algorithm and the propagation of evidence; experiments are described as well. Section 5 shows how SVE can be used to approximate reasoning in credal networks using conditioning operations. Finally, Section 6 indicates future efforts.

2 Credal Sets

A credal set, denoted by $\mathbf{Q}(X)$, is a closed convex set of probability measures associated with a random variable X . Usually, a credal set is represented by its extremes distributions $p_1(X), p_2(X), \dots, p_n(X)$. This article deals only with closed convex sets of probability measures. Note that $p_i(X)$ are points in polytopes and indicate probability distributions. Given a credal set $\mathbf{Q}(X)$, a common problem is to compute bounds for probability of events. These bounds are the *lower probability*, $\underline{P}(X = x) = \min_{p(X) \in \mathbf{Q}(X)} P(X = x)$ and the *upper probability* $\overline{P}(X = x) = \max_{p(X) \in \mathbf{Q}(X)} P(X = x)$.

A conditional credal set $\mathbf{Q}(X|Y = y)$ contains conditional probability densities $p(X|Y = y)$. Conditioning is defined as elementwise application of Bayes rule over the joint credal set $\mathbf{Q}(X, Y)$. Note that given sets $\mathbf{Q}(X|Y = y)$ and $\mathbf{Q}(Y)$, there may be more than one set $\mathbf{Q}(X, Y)$ that produces $\mathbf{Q}(X|Y = y)$ and $\mathbf{Q}(Y)$. Any such set $\mathbf{Q}(X, Y)$ is called an *extension* of $\mathbf{Q}(X|Y = y)$ and $\mathbf{Q}(Y)$.

For the purposes of this paper, there are two ways to represent conditional credal sets for a variable X given variable Y . First, we may store a collection of CPTs (that is, a list of functions as $p_1(X|Y), \dots, p_m(X|Y)$). Such a set is denoted by $L(X|Y)$ and is called an *extensive* conditional credal set. An alternative representation is to store a collection of credal sets, one for each value of Y (that is, a list of sets $\mathbf{Q}(X|Y = y)$). When a conditional credal set is represented as a list of credal sets, we denote it by $\mathbf{Q}(X|Y)$ and call it *separately specified*. As we show later, separately specified credal sets lead to significant computational savings. A discussion of terminology and theoretical properties of extensive and separately specified credal sets is provided in [18].

In this paper we adopt the concept of *strong independence* [7]. Two variables X and Y are strongly independent when every vertex $p(X, Y)$ of $\mathbf{Q}(X, Y)$ satisfies stochastic independence of X and Y .¹ We adopt the following direct generalization of strong independence: Variables X and Y are strongly independent given Z when the sets $\mathbf{Q}(X, Y|Z = z)$ have extreme points satisfying stochastic independence of X and Y for every value of Z .

3 Credal Networks

A credal network is a graphical structure that associates directed acyclic graphs with credal sets. Figure 1 depicts an example of credal network with separately specified sets.

Given the graphical structure and the credal sets in a credal network, we must decide how to combine the “local” credal sets into “joint” credal sets. In this paper we assume that separately specified credal sets are combined by “concatenating” the various distributions. That is, suppose that a separately specified credal set $\mathbf{Q}(X|Y)$ is given. We can produce an extensive credal set by taking every function $p(X|Y)$ that can be generated satisfying $p(X|Y = y) \in \mathbf{Q}(X|Y = y)$. An example, consider variable F in Figure 1: There are four possible values for (E, G) ; the credal set produced by concatenating $\mathbf{Q}(F|E, G)$ contains 2^4 vertices. (Technical definitions of concatenation operators are given in [18].)

In this article, we adopt the convention that, in a credal network, every variable is strongly independent of its nondescendants nonparents given its parents — that is, we adopt a Markov condition on the network [17] and we use strong independence in this Markov condition. Given this assumption, the vertices of the joint credal set must satisfy stochastic independence [8]. This joint credal

¹ That is, given $\mathbf{Q}(X, Y)$, each vertex of this set satisfies $p(X|Y) = p(X)$ and $p(Y|X) = p(Y)$.

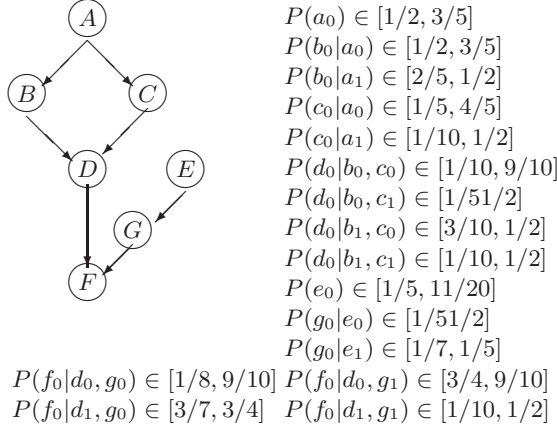


Fig. 1. Example credal network: all variables are binary, except C , which is ternary

set is called the *strong extension* of the credal network, and is the most common graphical model associated with sets of probabilities [2]:

Definition 1 *The strong extension of a credal network is the convex hull of probability densities that satisfy the Markov condition on the network:*

$$\mathbf{Q}(\mathbf{X}) = \text{CONVEXHULL} \left(\begin{array}{c} \prod_i p(X_i|pa(X_i)) \\ p(X_i|pa(X_i)) \in L(X_i|pa(X_i)) \end{array} : \right). \quad (1)$$

We assume that $L(X_i|pa(X_i))$ is always given by concatenating $\mathbf{Q}(X_i|pa(X_i))$; we say the strong extension is *separately specified*. Given a credal network where every combination of variables has positive lower probability, any graphical d-separation relation in the credal network corresponds to a valid conditional independence relation in the strong extension network [8].

Several authors have proposed algorithms for inferences with strong extensions. Here *inference* means the computation of lower or upper probabilities (given evidence or not). There have been exact [8, 12] and approximate [3, 4, 14, 19] algorithms; a review of such algorithms is provided in [18].

The main difficulty with strong extensions is the potentially enormous number of vertices that must be explored to find probability bounds. Inference is a NP-complete problem with credal networks, even for networks that are based on polytrees [18]. Several techniques can be used to organize this search for bounds, as discussed in [18]. But not all vertices of the joint credal set $\mathbf{Q}(\mathbf{X})$ may affect the posterior set of interest $\mathbf{Q}(X_q|\mathbf{E})$ (where X_q is the variable of interest and $\mathbf{E}$ is the evidence in the network). Two factors can lead to removal of vertices: (1) some credal sets may not affect bounds due to d-separation, and (2) some vertices of $\mathbf{Q}(\mathbf{X})$ may lead to redundant points of $\mathbf{Q}(X_q|\mathbf{E})$. We assume that, before running any algorithm, d-separation relations are used to discard

variables that do not affect computations; consequently, we focus here on the problem of eliminating redundant points of $Q(X_q|\mathbf{E})$.

Suppose then that we have a separately specified strong extension; can we use the fact that sets are separately specified to simplify the detection/removal of redundant points? The answer is yes, and leads to the SVE algorithm, described in the next section. Correctness proofs are contained in [18]. It seems that SVE is the first exact algorithm that explores the separability of sets, even though the important work of Cano and Moral [5] has explicitly used separability in approximate algorithms and suggested the use of separability in exact algorithms.

4 Separable Variable Elimination

To understand the use of separability in the SVE algorithm, consider first the variable elimination algorithm [10, 9, 20]. This algorithm produces inferences for standard Bayesian networks. Variable elimination essentially builds a tree of *buckets* and passes messages from the leaves to the root of the tree. Each bucket is responsible for “eliminating” a variable X_j . Elimination occurs by selecting all functions that contain X_j and summing out X_j from the product of these functions. The result of this operation is a new function that replaces all functions in the bucket and is sent to some other bucket. This message is called *separator*. When message passing finishes, the root bucket contains the distribution of the queried variables. If we perform variable elimination with strong extensions, we must send messages that contain lists of vertices; that is, the separators become *sets of functions*.

A possible way to simplify inferences with credal sets would be to discard non-extreme vertices of separators; this idea has been suggested several times [14, 19]. Unfortunately, separators are commonly defined over high-dimensional spaces; even though convex hull and redundancy elimination algorithms are efficient in low dimensions [11], in high dimensions the complexity of convex hull algorithms is very high and the number of points actually discarded tends to be very low.

A reasonable strategy then is to always deal with separately specified credal sets, without ever turning them into the description $L(X_i|pa(X_i))$. This is the idea of the *separable variable elimination* algorithm, described next. Denote an arbitrary bucket by B . Any message sent by B is a set of functions $B(\mathbf{X}|\mathbf{Z})$, where $\mathbf{Z}$ indicates the conditioning variables.

Separable variable elimination algorithm (SVE)

- Run variable elimination, but keep messages $B(\mathbf{X}|\mathbf{Z})$ sent by buckets as separately specified sets with respect to $\mathbf{Z}$.
- Before a message $B(\mathbf{X}|\mathbf{Z})$ is sent by a bucket, run redundancy elimination in each one of the sets $B(\mathbf{X}|\mathbf{Z} = \mathbf{z})$ separately, for each value of $\mathbf{Z}$.

The central difference between SVE and standard variable elimination (with set-based separators) is that SVE operates on any bucket by taking one value of

$\mathbf{Z}$ at a time. From an algorithmic point of view, this difference is apparent when performing products and sums inside buckets. These critical sections of the SVE algorithm are now described.

The SVE implementation of product and sum-out operations

- Input for bucket B :
 - The bucket variable X_B .
 - The separately specified credal sets $\mathbf{Q}(\mathbf{X}_j|\mathbf{Y}_j, \mathbf{Z}_j)$ received by B from other buckets and taken from the network CPTs. Note: Define $\mathbf{Z}_j$ so that any of its elements is a variable that is always in the conditioning side of any function where it appears. Define $\mathbf{Y}_j$ so that any of its elements are conditioning in some functions but conditioned in others.
- Output the separately specified separator $\mathbf{S} = \mathbf{Q}(\mathbf{X}, \mathbf{Y}|\mathbf{Z})$. Note: $\mathbf{X} = \cup \mathbf{X}_j \setminus X_B$, $\mathbf{Y} = \cup \mathbf{Y}_j \setminus X_B$ and $\mathbf{Z} = \cup \mathbf{Z}_j \setminus \{\mathbf{X} \cup \mathbf{Y}\}$.

For each instantiation $z \in \mathbf{Z}$:

1. collect every separately specified credal set where some variable conditioning matches with z or with X_B
2. collect every separately specified credal set where X_B is a conditioned variable
3. for each possible combination of functions collected in the previous two steps, construct a combined function v as follows:
 - (a) concatenate every function associated to v ; call the concatenated function v' ;
 - (b) sum out X_B from v' to finally obtain the function v ;
 - (c) add v to $\mathbf{S}$ given z (v is an entry of $\mathbf{S}$ given z).
4. remove redundant elements of $\mathbf{S}$ given z running a convex hull algorithm.

We have noticed that, the fact that SVE keeps credal sets in separable form makes it more efficient (in terms of space, *even without redundancy elimination*) than algorithms that use extensive credal sets. Many networks can only be solved by SVE while failing with representations that directly use extensive credal sets.

To give an example of the SVE's gains in these operations, consider the network in Figure 1 and suppose each variable is ternary. Furthermore, assume that each separately specified credal set has three vertices. The order of variable elimination is E, G, A, C, D, E, F . Now take the bucket for variable G , denoted for B_G , and consider the operation *sum-out* on this bucket. In the calculation of the B_G 's separator in SVE, the separately specified sets connected to $\mathbf{Q}(F|D, G)$ and to the separator from E 's bucket, $\mathbf{S}(E)$, are present. The separator of B_G is $\mathbf{S}(G) = \mathbf{Q}(F|D)$ and it is composed of three separately specified credal sets each one with 3^4 vertices. The total number of distributions summing all sets is 3^5 . In the standard variable elimination, this procedure processes 3^{13} functions. Additionally, this gain is propagated forward.

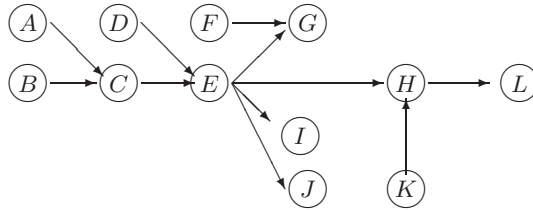


Fig. 2. An polytree credal net

In our experiments, we have seen several examples in which the redundancy elimination step leads to a significant decrease in computational effort [18]. We have noticed that in general the reduction of vertices is not a dramatic factor in inference; moreover, the cost of redundancy elimination is not negligible and sometimes outweighs its benefits.

Consider the following tests of SVE in the networks in Figures 1 (where we asked for the marginal of variable F) and 2 (where we asked for the marginal of variable L).² The first test is simply to run these networks with binary variables and with two vertices in each credal set. Both standard variable elimination (with separators containing sets of functions) and SVE obtained results in this case. Now, when we tried to deal only with ternary variables in the networks standard variable elimination failed. That is, standard variable elimination could not handle inference in these credal networks with ternary variables. But SVE dealt with both networks with ternary variables when credal sets had two vertices. When credal sets had three vertices, inferences with the network in Figure 1 failed, while inferences with the network in Figure 2 succeeded. Note that the network in Figure 2 has a strong extension with 2^{18} potential vertices; SVE succeeds basically because this network is a polytree and separators never have high dimensionality.

We remark that SVE can easily handle a body of evidence $\mathbf{E}$, just as standard variable elimination. After removing unnecessary variables by d-separation, SVE can handle messages from credal sets $Q(X_i|pa(X_i))$ where any of X_i or $pa(X_i)$ belongs to $\mathbf{E}$.

5 Approximated Reasoning

Because SVE is well suited for networks that have a polytree structure, it would seem that inferences based in loop cutsets [17] could be a profitable way to handle multi-connected networks. We could consider conditioning operations as a general way to reduce the effort spent in inference.

The main idea here is to use loop cutset algorithms to transform complex credal networks, not necessarily multi-connected, in several simpler nets. In the

² We run these networks with randomly generated credal sets (that is, the vertices of the sets are randomly generated conditional distributions). We have implemented SVE in Java, and have run the tests with Java San 1.4.0 interpreter with 72M of heap and 6M of stack, using a HP Pentium 4 with Windows 2000.

context of credal networks, the use of cutsets produces only approximations, not exact inferences [12]. This happens because loop cutset produces several values for upper and lower probabilities (one for each combination of variables in cut set), and direct combination of these values is not equivalent to global lower and upper values. The resulting values are enclosing approximations (that is, the correct bounds are inside of approximated bounds).

We denote the loop cutset by C and the conditioned upper bounds, obtained by conditioning, as :

$$\overline{P}(X|C) = (\max_{p(X|C) \in \mathbf{Q}(X|C)} P(X = x_1|C), \dots, \max_{p(X|C) \in \mathbf{Q}(X|C)} P(X = x_m|C)) \quad (2)$$

where $(x_0, \dots, x_m)$ is an exhaustive enumeration of the X 's states. To combine these bounds, we need to have probabilities for the variables in the cutset. If an inference is requested without evidence, it is easy to obtain the cutset probabilities when the cutset variables do not have parents.

Consider an example. If we take the network in Figure 1 with binary variables and two points for separately specified credal set, we obtain the exact upper bounds $\overline{P}(F) = (0.4953; 0.7767)$. Then, if we set $C = \{A\}$, the conditioned bounds are $\overline{P}(F|A) = (0.4964; 0.7777)$. By combining these bounds with $\mathbf{Q}(A)$ we obtain an approximation that matches with exact values up to the fourth decimal digit.

It may be difficult to combine conditioned bounds in more complicated topologies. The key point to apply that technique depends on following question: is it easy to obtain the loop cutset's credal set? If the answer is no, an approach is to use the conditioned bounds as an approximation. That is, instead of trying to combine them for each instantiation, we can obtain the upper values for each state of cutset variables, and then simply take the largest and smallest values respectively. The resulting values are enclosing approximations, too. In previous example (Figure 1) the approximated bounds for F are $\overline{P}^*(F) = (0.4964; 0.7777)$.

Consider a few other examples. We compute a query about the variable L in the network of the Figure 2, with ternary variables and 4 points by credal set. That query could not be computed by SVE. But, using the loop cutset in D, K , and this scheme, we obtain the next upper bounds $\overline{P}^*(L) = (0.54; 0.524; 0.459)$. Figure 3 brings two more examples, one with binary variables (left) and one with ternary variables (right). In both networks every variable is associated with credal sets that have two vertices, except that in the last network the variables E, G, C and F are associated with a single distribution. In the left network, we selected the cutset $\{B, C\}$ and asked for the marginal distribution of N . The exact results are: $\underline{P}(N) = (0.4784; 0.2577)$ and $\overline{P}(N) = (0.7422; 0.52159)$; the approximated results are: $\underline{P}^*(N) = (0.4779; 0.2569)$ and $\overline{P}^*(N) = (0.7430; 0.5220)$. In the right network we computed the lower and upper values for variable C , for each instantiation of the cutset variables $\{A, B\}$; we took just the maxima and minima to illustrate the application of Expression (2). The exact results are: $\underline{P}(C) = (0.4011; 0.4080; 0.2699)$ and $\overline{P}(C) = (0.3398; 0.3705; 0.2185)$;

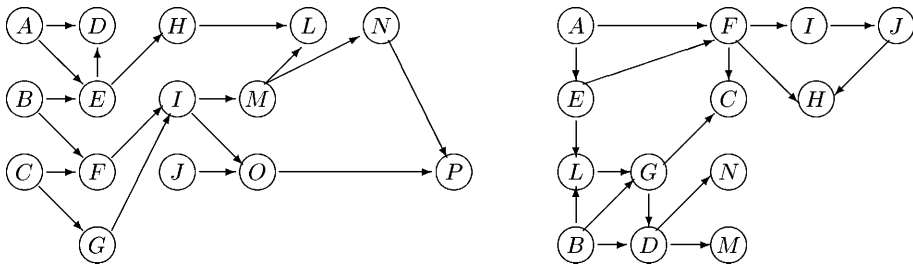


Fig. 3. Examples: credal network with binary variables (left) and with ternary variables (right)

the approximated results are: $\underline{P}^*(C) = (0.5410; 0.4551; 0.3201)$ ³ and $\overline{P}^*(C) = (0.3173; 0.3223; 0.1366)$ (showing the lower quality of this approximation).

6 Concluding Remarks

In this paper we have described SVE, an algorithm for inferences in credal networks. Our goal was describe implementation and test aspects of SVE, so as to complement the theoretical derivation of SVE presented in [18]. We have also discussed a different dimension of SVE, by exploring its use in connection with loop cutset algorithms. We have proposed a scheme for generating approximations through conditioning operations — namely, by taking maxima and minima across the values of the conditioned variables.

Experiments show that SVE is a significant advance when compared to existing exhaustive methods. The improvement obtained by SVE is derived from its reliance on separately specified credal sets. We are now investigating efficient heuristic methods to reduce the cost of convex hull operations through divide and conquer algorithms.

Acknowledgements

We thank CAPES and CNPq for partially supporting the first and second authors respectively. Thanks also to Marsha Duro from HP Labs, Edson Nery from HP Brasil and Instituto de Pesquisas Eldorado, for providing equipment that was used in tests.

References

- [1] Campos, L. M. de, Moral, S.: Independence concepts for convex sets of probabilities. Proc. of the XI Conf. on Uncertainty in Artificial Intelligence, Montreal, Canada (1995) 108–115.

³ To generate approximations for lower probabilities, substitute min for max in Expression (2).

- [2] Cano, J., Delgado, M., Moral, S.: An axiomatic framework for propagating uncertainty in directed acyclic networks. *International Journal of Approximate Reasoning*, (1993) 8:253-280. 377, 379
- [3] Cano, A., Cano, J. E., Moral, S.: Convex sets of probabilities propagation by simulated annealing. *IPMU'94*, Paris France (1994) 978-983. 379
- [4] Cano, A., Moral, S.: A genetic algorithm to approxiamte convex sets of probabilities. *IPMU'96*, Vol. 2, (1999) 859-864. 379
- [5] Cano, A., Moral, S.: Using probabilities trees to compute marginals with imprecise probabilities. *TR-DECSAI-00-02-14*, University of Granada, (2000). 380
- [6] Charniak, E.: Bayesian Networks without tears. *AI Magazine*, Vol.12. 4 (1994) 50-63. 376
- [7] Couso, I. Moral, S., Walley, P.: Examples of independence for imprecise probabilities. 1st *ISIPTA*, Ghent Belgium (1999). 378
- [8] Cozman, F. G.: Credal networks. *Artificial Intelligence* (2000) 1-35. 377, 378, 379
- [9] Cozman, F. G.: Generalizing variable elimination in Bayesian networks. *Workshop on Probabilistic Reasoning in Artificial Intelligence - IBERAMIA/SBIA*, Atibaia, Brazil (2000). 380
- [10] Dechter, R.: Bucket elimination: A unifying framework for probabilistic inference. *Proc. of the XII Conf. on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, San Francisco, CA (1996) 211-219. 376, 380
- [11] Edelsbrunner, H.: *Algorithms in combinatorial geometry*. Springer-Verlag (1987). 380
- [12] Fagioli, E., Zaffalon, M.: 2U: an exact interval propagaion algorithm for poly-trees with binary variables. *AI*, Elsevier, 106 (1998) 77-107. 379, 383
- [13] Giron, F. J., Rios, S.: Quasi-Bayesian behavior: A more realistic approach to decision making? in: Bernardo, J. M., DeGroot, J. H., Lindley, J. H. Smith, A. F. M. (eds.) *Bayesian Statistics*, University Press, Valencia, (1980) 17-38.
- [14] Ha, V. A. *et al*: Geometric foundations for interval-based probabilities. *Annals of Mathematics and Artificial Intelligence*, Vol.24, 1-4 (1998) 1-21. 379, 380
- [15] Jensen, F. V., Olensen, K. G., Andersen, S. K.: An algebra of Bayesian belief universes for Knowledge-Based Systems. *Networks*, Vol. 20. (1990) 637-659. 376
- [16] Neapolitan, R. E.: *Probabilistic reasoning in expert systems. Theory and algorithms*. John Willey and Sons, New York, (1990).
- [17] Pearl, J.: *Probabilistic reasoning in intelligent systems. Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Mateo CA (1988). 376, 378, 382
- [18] Rocha, J. C. F. da, Cozman, F. G.: Inference with Separately Specified Sets of Probabilities in Credal Networks, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, Edmonton, Canada, (2002). 377, 378, 379, 380, 382, 384
- [19] Tessem, B.: Interval Probability propagation. *Int. Journal of Approximated Reasonin* 7, (1995) 95-120. 377, 379, 380
- [20] Zhang, N. L., Poole, D. Exploiting causal independence in Bayesian networks inference. *Journal of Artificial Intelligence Research*, 5 (1996) 301-328. 380

Restoring Consistency in Systems of Fuzzy Gradual Rules Using Similarity Relations

Isabela Drummond¹, Lluís Godo², and Sandra Sandri^{1,2}

¹ LAC - INPE, 12201-970 S.J. Campos, Brazil

{isabela,sandri}@lac.inpe.br

² IIIA - CSIC, 08193 Bellaterra, Spain

{godo,sandri}@iiia.csic.es

Abstract. We present here a method that uses similarity relations to restore consistency in fuzzy gradual rules systems: we propose to transform potentially inconsistent rules by making their consequents more imprecise. Using a suitable similarity relation we obtain consistent rules with a minimum of extra imprecision. We also present an application to illustrate the approach.

Keywords: fuzzy rule-based systems, gradual rules, inconsistency, similarity.

1 Introduction

Ideally, *fuzzy rule based systems* must be capable to produce for any given input a meaningful global output, in the sense that it is neither the empty set nor the whole universe of discourse of the output variable. The inference mechanisms used in fuzzy rules to produce an output for a given input depend on the type of fuzzy rule.

Fuzzy rules are basically classified as conjunctive or implicative-based, depending on the kind of if-then operator employed to define the fuzzy relation induced by each rule¹. If we think of a fuzzy rule base $\{ \text{If } X \text{ is } A_i \text{ then } Y \text{ is } B_i \}_{i \in I}$ as modeling an imprecise description of a graph, the two models of rules respectively correspond to two possible ways of specifying an imprecise graph: either as a disjunction of fuzzy points $\cup_{i \in I} A_i \times B_i$, or as a conjunction of fuzzy implications $\cap_{i \in I} A_i \rightarrow B_i$. Therefore, conjunctive rule-based systems, widely used in real-world applications, use a t-norm² (i.e. a conjunction operator), such as min or product, in order to implement the if-then operator and a t-conorm (i.e. a disjunction operator), usually the max operator, to aggregate the output issued by the rules fired by a given input. On the other hand,

¹ Other types of rules, like those employed in Takagi-Sugeno controllers [9] or in yet other approaches, like [12], are not considered in that work.

² An operator $\top : [0, 1]^2 \rightarrow [0, 1]$ is a *t-norm* when it is commutative, associative, monotonic and has 1 as neutral element, and an operator $\perp : [0, 1]^2 \rightarrow [0, 1]$ is called a *t-conorm* (or *s-norm*) when it is commutative, associative, monotonic and has 0 as neutral element.

implicative-based systems use truly implication operators, as opposed to conjunction operators, to implement the if-then operator and a t-norm, usually the min operator, to aggregate the outputs. In [2], the authors provide a typology for conjunction and implication-based rules based on their different semantics.

A fuzzy rule system is said to be *inconsistent*³ if there exists an input such that the rules fired by that input produce somewhat conflicting outputs. Due to their different nature, a conjunctive system is never completely inconsistent from a logical point of view, while implication-based systems may easily get into inconsistency problems as soon as an input can simultaneously fire rules with non intersecting conclusions. This is one of the main reason why implicative systems are not so popular as the conjunctive ones (e.g. Mamdani systems) in real-world applications.

The issue of checking consistency or coherence in fuzzy rule sets has been addressed in a number of papers in the literature, e.g. [11, 6, 3, 4, 7] (see also [8]). In a preliminar work [5] we have presented an approach to restore consistency in the context of implication-based fuzzy rule systems, IFS for short. These rules are known as *gradual rules* [2], corresponding to statements of the form “the more X is A_i , the more Y is B_i ”, where A_i and B_i are fuzzy subsets of the corresponding variable domains U_X and U_Y respectively. The fuzzy relation defined by such a rule is given by $R_i(x, y) = A_i(x) * \rightarrow B_i(y)$, with $a * \rightarrow b = \sup \{c \in [0, 1] \mid a * c \leq b\}$, where $*$ is a continuous t-norm ($* \rightarrow$ is said to be the residuum of $*$), and the global fuzzy relation induced by a set of gradual rules $K = \{R_i\}_{i \in I}$ is the conjunctive aggregation of the relations defined by each rule $R_K(x, y) = \inf_{i \in I} R_i(x, y)$. The output corresponding to a precise input $X = x_0$ is the fuzzy relation

$$\text{output}(K, \{x_0\})(y) = \inf_{i \in I} R_i(x_0, y).$$

The rule set K is considered fully consistent if $\sup_y \text{output}(K, \{x_0\})(y) = 1$ for all x_0 , i.e. if we always get a normalized output for any precise input. This corresponds to the notion of *coherence* in [3, 4] and *1-consistency* in the context of π -reasoning in [10]. Conditions characterizing full consistency in systems of fuzzy gradual rules have been basically addressed by Dubois, Prade and Ughetto in [3, 4]. Note that one can always define the degree of consistency as the infimum of the possible output heights, that is,

$$\text{Con}(K) = \inf_x \sup_y \text{output}(K, \{x\})(y),$$

full consistency corresponding to the case $\text{Con}(K) = 1$.

In [5] we have restricted ourselves to a specific subclass of systems, namely those in which the input variables are associated to fuzzy terms forming Ruspini partitions (see Section 3). In the present work we present our formalism for restoring consistency (in the sense of full consistency!) for a larger class of systems, still restricted, but encompassing quite a large number of real world

³ This is clearly an abuse of language, it is actually meant *potentially inconsistent*.

applications. Here we present not only the theoretical framework but also an illustrative application that serves to compare the proposed approach to the ones usually employed in most applications of the fuzzy rule based framework.

The paper is organized as follows. In the next section we discuss the main ideas of our similarity-based approach to restore consistency, while in Section 3 we focus and particularize it for the case of gradual rules in a usual working framework. Finally, in Section 4 we present an application example and in Section 5 we discuss some issues related the proposed methodology.

2 Restoring Consistency

If the inconsistency of an IFS is not the result of an inherently wrong modeling problem but due to a possible over-precision in the designing of the fuzzy rules (e.g. too narrow domain partitions), then one can pose oneself the question of overcoming the consistency problem by making rule consequents more imprecise. Consider the following simple example.

Example 1. Let an IFS be composed of the following two rules

R1: “If X is A_1 then Y is B_1 ”

R2: “If X is A_2 then Y is B_2 ”

with $A_1 = \{1, 2, 3\}$, $A_2 = \{3, 4, 5\}$, $B_1 = \{2, 3\}$ and $B_2 = \{5, 6\}$. Then, clearly this IFS can lead to inconsistency since for the input $X = \{3\}$ we get the output $B_1 \cap B_2 = \emptyset$. However, if we make the rule consequents more imprecise we can eventually turn the system consistent, e.g. replacing B_1, B_2 by $B_1^* = \{1, 2, 3, 4\}$ and $B_2^* = \{4, 5, 6, 7\}$ respectively we get as output $B_1^* \cap B_2^* = \{4\}$. $\square$

Therefore, the idea we propose is to turn each rule

“If X is A_i then Y is B_i ”

in a set of inconsistent fuzzy rules into a more imprecise rule

“If X is A_i then Y is *approximately* B_i ” ,

where *approximately* B_i is a fuzzy set, bigger than B_i , interpreting the notion of being *around* B_i . Formally, this is achieved by taking *approximately* B_i as $S \circ B_i$, the image of B_i by a similarity relation S on the output domain U_Y , defined by:

$$(S \circ B_i)(v) = \sup_{v' \in U_Y} \min(S(v, v'), B_i(v')).$$

Here S is a binary fuzzy relation on U_Y interpreting a user’s notion of closeness (or conversely $1 - S$ modeling a notion of distance or metric) on the domain U_Y . Usual properties which are required to S are reflexivity, i.e. $S(v, v) = 1$ for all $v \in U_Y$, and symmetry, i.e. $S(v, v') = S(v', v)$ for all $v, v' \in U_Y$. T-norm transitivity ($S(u, v) \otimes S(v, w) \leq S(u, w)$ for all $u, v, w \in U_Y$ and some t-norm $\otimes$) does not seem to play a role here.

Given a set of conflicting fuzzy rules

$$\text{FRS} = \{R_i : \text{"If } X \text{ is } A_i \text{ then } Y \text{ is } B_i"\}_{i=1,n}$$

and a similarity relation S on U_Y , let us denote by FRS^* the set of fuzzy rules R_i^* obtained by substituting each B_i by $B_i^* = S \circ B_i$. Of course, the natural questions which arise are: (i) which similarity relations should we take to make FRS^* consistent, and (ii) whether there exists a *best* similarity relation. In the following we try to answer these questions.

The set of similarity relations on U_Y forms a lattice (not linearly ordered) with respect to the point-wise ordering (or fuzzy-set inclusion) relationship. The top of the lattice is the similarity $S_\top$ which makes all the elements in the domain maximally similar: $S_\top(v, v') = 1$ for all $v, v' \in U_Y$. The bottom of the lattice $S_\perp$ is the classical equality relation: $S_\perp(v, v') = 1$ if $v = v'$, $S_\perp(v, v') = 0$, otherwise. The higher a similarity is in the lattice (i.e. the bigger are their values), the less discriminating it is.

It is clear then that the bigger is S the more imprecise are the sets $B_i^* = S \circ B_i$, and the more imprecise are the rules "If X is A_i then Y is B_i^* ". From a knowledge representation point of view, we are interested in losing as little information as possible when passing from FRS to FRS^* , hence we shall be interested in using a similarity S as small as possible. However, from the inconsistency problem point of view, the bigger the similarity S , the more confidence we have on getting FRS^* consistent. Notice that if $S \leq S'$, then if FRS^* is coherent, then so is FRS'^* . Of course, the trivial solutions do not help at all: if we take $S = S_\top$, FRS^* will be for sure a consistent system, but completely useless since for any input we will get the totally unrestricted output U_Y ; if we take the $S = S_\perp$, then $\text{FRS}^* = \text{FRS}$, thus there is no information loss but the inconsistency problem will remain. Therefore, optimal solutions would be minimal S 's for which FRS^* remains consistent. Notice that in general there will not be a single best solution but a set of optimal ones.

On the other hand, according to fuzzy rules can be reduced to checking whether such rules are pair-wise coherent, that is, they show that a set $\text{FRS} = \{R_i\}_{i \in I}$ is coherent iff each pair $\{R_i, R_j\}_{i,j \in I}$ is coherent. So, in this sense, the problem of getting a "best" similarity S for the set FRS can also be reduced to finding for each pair $\{R_i, R_j\}$ a best similarity S^{ij} such that $\{R_i^{*ij}, R_j^{*ij}\}$ is not conflicting. Actually, one could even think of finding two best similarities, S^{ij} and S^{ji} , one per each rule respectively, such that $\{R_i^{*ij}, R_j^{*ji}\}$ becomes consistent. However we think that in doing so, if the similarities are different, it would mean a (hidden) different status of the two rules. If we do not have any extra domain knowledge to support it, it seems it is not justifiable. Therefore, the problem reduces to finding a *common* best similarity for each pair of conflicting rules. Then, it is sure that

$$\overline{S} = \max_{i,j} S^{ij}$$

(where $\max$ denotes the point-wise maximum) makes the resulting new set of rules FRS^* coherent.

Nevertheless, still giving the same status to each rule pair by pair, instead of looking for a best single similarity to apply on each rule of the rule set FRS

one could also think of looking for a possibly different similarity S^i for each rule R_i , in such a way that the resulting new set $\text{FRS}^* = \{R_i^{*i}\}_{i \in I}$ is coherent, where R_i^{*i} is obtained from R_i by substituting B_i by $B'_i = S^i \circ B_i$. Actually, the above procedure can be easily adapted, in fact once we have found a best similarity S^{ij} for each pair of rules $\{R_i, R_j\}$, then the best similarity for each rule R_i is simply

$$\overline{S}_i = \max_j S^{ij}.$$

3 A Simple Procedure to Restore Consistency for Pairs of Rules

Since the problem of inconsistency of a set of gradual fuzzy rules can be reduced to a problem of inconsistency of pairs of rules [4], first we focus on how to restore consistency of a pair of gradual rules.

We propose a conceptually very simple procedure to *minimally* transform a pair of inconsistent rules into a consistent one with respect to what we call a *covering nested family* of similarity relations on U_Y . By this we mean a parametric family $\mathcal{S} = \{S_0, S_{+\infty}\} \cup \{S_\lambda\}_{\lambda \in I \subseteq (0, +\infty)}$ of similarity relations such that:

- (i) $S_0 = S^\perp$,
- (ii) $S_{+\infty} = S^\top$, and
- (iii) $\lambda < \lambda'$, then $S_\lambda \prec S_{\lambda'}$.

Here $S \prec S'$ means $S(x, y) \leq S'(x, y)$ for all $x, y \in U_Y$ and $S(x_0, y_0) < S'(x_0, y_0)$ for some $x_0, y_0 \in U_Y$. Then, given a pair of inconsistent rules $\{R_1, R_2\}$ one has just to take the least similarity S_λ such that $\{R_1^{*\lambda}, R_2^{*\lambda}\}$ become consistent. Notice that this can be done since the family $\mathcal{S}$ is linearly ordered by fuzzy set inclusion, otherwise the infimum might not exist.

The question is then whether consistency for a pair of rules can be easily checked. To answer this question we first restrict ourselves to systems obeying very simple but usual requirements:

- The terms associated with a linguistic variable are distinct fuzzy numbers in *consecutive order*. We say that a set of terms $\{D_1, \dots, D_n\}$ is in consecutive order when for all i , if $\text{supp}(D_i) \cap \text{supp}(D_{i-1}) \neq \emptyset$ and $\text{supp}(D_i) \cap \text{supp}(D_{i+1}) \neq \emptyset$, then $\text{supp}(D_i) \cap \text{supp}(D_j) = \emptyset, \forall j \neq i-1, i+1$.
- If D and D' are consecutive terms associated with a linguistic *input variable* defined on U then, $\mu_D(\omega) + \mu_{D'}(\omega) \leq 1$ for all $\omega \in \text{supp}(D) \cap \text{supp}(D')$. When instead of ≤ 1 we have $= 1$, the terms of are said to form a Ruspini's fuzzy partition of the domain of the variable.

Here we use $\text{supp}(D)$ and $\text{core}(D)$ to denote the *support* and the *core* of a fuzzy set D . Also, from now on, the extreme points of a (closed) interval I will be denoted by $l(I)$ and $r(I)$ in such a way that $I = [l(I), r(I)]$, and the α -cut of a fuzzy set D will be denoted $[D]_\alpha$.

Let us consider a pair of fuzzy gradual rules

$$\{R_i : \text{If } X \text{ is } A_i \text{ then } Y \text{ is } B_i\}_{i=1,2}$$

belonging to a IFS's obeying the above two restrictions, in particular we consider that A_1 is before A_2 and B_1 is before B_2 , i.e. $r(\text{core}(A_1)) \leq l(\text{supp}(A_2))$ and $r(\text{supp}(A_1)) \leq l(\text{core}(A_2))$, analogously for B_1 and B_2 .

R_1 and R_2 are (fully) consistent if, for any precise input $X = x_0$, the cores of the outputs for both rules intersect, i.e. if $[B_1]_{A_1(x_0)} \cap [B_2]_{A_2(x_0)} \neq \emptyset$ for any $x_0 \in U_X$. This condition is actually independent from the particular residuated implication $*\rightarrow$ used to define the fuzzy relations induced by the rules $R_i(x, y) = A_i(x) * \rightarrow B_i(y)$, since $a * \rightarrow b = 1$ iff $a \leq b$ for any residuated operator $*\rightarrow$. Due to this fact, we can restrict our setting to the so-called Rescher-Gaines implication function, defined as $a * \rightarrow_{RG} b = 1$ if $a \leq b$ and $a * \rightarrow_{RG} b = 0$ otherwise, whose induced fuzzy relation can be seen as the intersection of those generated by all residuated implications. Besides, it is very simple and always produces non-fuzzy outputs, and as we shall show it behaves reasonably well once inconsistencies are resolved.

From a practical point of view, what is interesting is that one can find easy-to-check necessary and sufficient conditions for R_1, R_2 to be consistent. Actually, the following proposition, which generalizes results of [4], provides such necessary and sufficient conditions.

Proposition 1. *Under the above assumptions, R_1 and R_2 are consistent iff*

$$\begin{aligned} A_1(l(\text{supp}(A_2))) &\leq B_1(l(\text{supp}(B_2))), \\ A_2(r(\text{supp}(A_1))) &\leq B_2(r(\text{supp}(B_1))). \end{aligned}$$

From this, a simpler sufficient condition can be derived.

Corollary 1. *Let $\delta = \max(A_1(l(\text{supp}(A_2))), A_2(r(\text{supp}(A_1))))$. If $B_1 + B_2 \geq \delta$ in the region $\text{supp}(B_1) \cap \text{supp}(B_2)$ then R_1 and R_2 are consistent.*

The sufficient condition expressed in this corollary can also be expressed in the following way:

$$l(\text{supp}(B_2)) \leq r([B_1]_\delta), \quad l([B_2]_\delta) \leq r(\text{supp}(B_1)). \quad (\text{Con})$$

It becomes also necessary when a symmetry condition is satisfied by the A_i 's.

Corollary 2. *If A_1 and A_2 are such that $A_1 + A_2$ is constant (equal to 2δ) in $\text{supp}(A_1) \cap \text{supp}(A_2)$, then R_1 and R_2 are consistent iff $B_1 + B_2 \geq \delta$ in $\text{supp}(B_1) \cap \text{supp}(B_2)$.*

Notice that in particular, when the input partitions are Ruspini partitions we have $A_1 + A_2 = 1$ in $\text{supp}(A_1) \cap \text{supp}(A_2)$ and $\delta = 1$, and then the above conditions (Con) reduce to the well-known conditions (cf. [4, Props. 4.8 and 4.11]):

$$l(\text{supp}(B_2)) \leq r(\text{core}(B_1)), \quad l(\text{core}(B_2)) \leq r(\text{supp}(B_1)).$$

Now, coming back to our proposed procedure of restoring consistency, given a covering family of similarity relations $\mathcal{S} = \{S_\lambda\}_\lambda$, one has just to find the smallest λ such that the expanded rules

$$\{R_i^* : \text{If } X \text{ is } A_i \text{ then } Y \text{ is } B_i^* = S_\lambda \circ B_i\}_{i=1,2}$$

verify conditions (Con). This is exemplified next.

Example 2. Consider the above pair of rules R_1 and R_2 . Assume U_Y is a segment of the real line, and consider the following family of similarity relations: for each $\lambda > 0$ define

$$\begin{aligned} S_\lambda(x, y) &= \mu_\lambda(|x - y|), \\ \mu_\lambda(z) &= \max(1 - \lambda^{-1} \cdot z, 0). \end{aligned}$$

One can easily check that if B_i is a trapezoidal fuzzy number $[a, b, c, d]$, then $B_i^{*\lambda} = S_\lambda \circ B_i$ is again a trapezoidal fuzzy number defined by the 4-tuple $[a - \lambda, b, c, d + \lambda]$. Now, define

$$\lambda_0 = \max\{l(\text{supp}(B_2)) - r([B_1]_\delta), \quad l([B_2]_\delta) - r(\text{supp}(B_1))\}.$$

It is easy to check that the least similarity of the above family satisfying the consistency conditions (Con) is just S_{λ_0} . $\square$

Let us now examine the case of rules with several input variables linked by a t-norm. Assume we have two 2-input variable fuzzy rules of the form

$$R_i : \text{If } \mathbf{x}_1 = A_i^1 \text{ and } \mathbf{x}_2 = A_i^2 \text{ then } \mathbf{y} = B_i$$

$i = 1, 2$. Consider the corresponding 1-input variable rules obtained by deleting one input variable:

$$R_i^1 : \text{If } \mathbf{x}_1 = A_i^1 \text{ then } \mathbf{y} = B_i \quad R_i^2 : \text{If } \mathbf{x}_2 = A_i^2 \text{ then } \mathbf{y} = B_i$$

Then it can be proved [4] that R_1 and R_2 are consistent iff both pairs $\{R_1^1, R_2^1\}$ and $\{R_1^2, R_2^2\}$ are consistent. Thus, if conditions (Con) are satisfied for $\delta = \max(\delta_1, \delta_2)$, where δ_i ($i = 1, 2$) is defined as in Corollary 1 with respect to the sets A_1^i, A_2^i respectively, then the above 2-input variable rules R_1 and R_2 are consistent. This framework is generalized straightforwardly to any number of input variables. Therefore the restoring consistency procedure can then be easily adapted.

4 Application Example

In this section we apply our formalism to a simple fuzzy control system (see [1]). The system to be controlled, taken from the Matlab Fuzzy Toolbox, is a shower having 2 input variables, *temp* and *flow*, and two output variables, *cold* and *hot*. Input variable *temp* (respec. *flow*) is associated to the fuzzy terms $\{\text{cold}, \text{good},$

Table 1. Rule base for *cold*

flow/temp	<i>cold</i>	<i>good</i>	<i>hot</i>
<i>soft</i>	<i>os</i> (1)	<i>os</i> (4)	<i>of</i> (7)
<i>good</i>	<i>cs</i> (2)	<i>st</i> (5)	<i>os</i> (8)
<i>hard</i>	<i>cf</i> (3)	<i>cs</i> (6)	<i>cs</i> (9)

Table 2. Rule base for *hot*

flow/temp	<i>cold</i>	<i>good</i>	<i>hot</i>
<i>soft</i>	<i>of</i> (1)	<i>os</i> (4)	<i>os</i> (7)
<i>good</i>	<i>os</i> (2)	<i>st</i> (5)	<i>cs</i> (8)
<i>hard</i>	<i>cs</i> (3)	<i>cs</i> (6)	<i>cf</i> (9)

hot} (respec. {*soft*, *good*, *hard*}) and both output variables are associated to the fuzzy terms {*open-fast*(*of*), *open-slow*(*os*), *steady*(*st*), *close-slow*(*cs*), *close-fast*(*cf*)} (see Figures 1 and 2). In Tables 1 and 2 we show respectively the rule base for each output variable (rule number is within parenthesis). Notice that since the input terms do not form Ruspini partitions this example could not be used using the restricted approach presented in [5].

It is easy to observe several inconsistent areas of the rule bases. For instance, for the output variable *cold* those 2×2 areas such that a single input can address at least 3 different outputs, like the set of rules $\{R_1, R_2, R_4, R_5\}$, may be simultaneously fired by values of *flow* and *temp* between -0.4 and 0 , and between -10 and 0 respectively. Also areas containing the pairs of rules addressing two non-consecutive outputs, like $\{R_1, R_2\}$, $\{R_2, R_4\}$, $\{R_6, R_8\}$, $\{R_8, R_9\}$, $\{R_3, R_5\}$, $\{R_5, R_7\}$ for *cold*, can generate empty outputs. Indeed, in the rule-base for *cold*, an input pair (*temp* = -15 , *flow* = -0.2) would fire rules R_1 and R_2 with compatibility degrees .25 and .5 respectively. Using Rescher-Gaines implication function, we would respectively obtain the outputs $[open-slow]_{.25} = [.075, .525]$ and $[close-slow]_{.5} = [-.45, -.15]$, whose intersection clearly yields the empty subset, generating thus an inconsistency.

Let us consider the family of similarity relations defined in Example 2. One can easily check that all the inconsistencies in the rule bases for both *hot* and *cold* disappear if S_3 is applied to all the output terms. For instance, the result of applying the similarity relation S_3 to the output sets *open-slow* and *close-slow* respectively yield $open-slow^{*.3} = [-.3, .3, .3, .9]$ and $close-slow^{*.3} = [-.9, -.3, -.3, .3]$ (see Fig. 2b). Then, for an input $(-15, -0.2)$ we would respectively obtain the outputs $[open-slow^{*.3}]_{.25} = [-.15, .75]$ and $[close-slow^{*.3}]_{.5} = [-.6, 0]$, with the non-empty intersection $[-.15, 0]$.

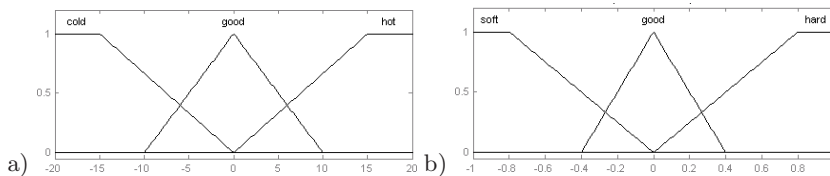


Fig. 1. Partition of the input space. a) variable *temp* b) variable *flow*

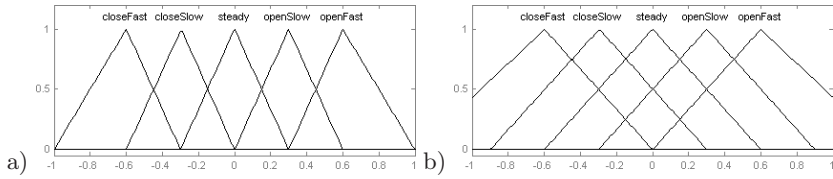


Fig. 2. a) Partition of the output space b) Transformed output terms using $S_{.3}$

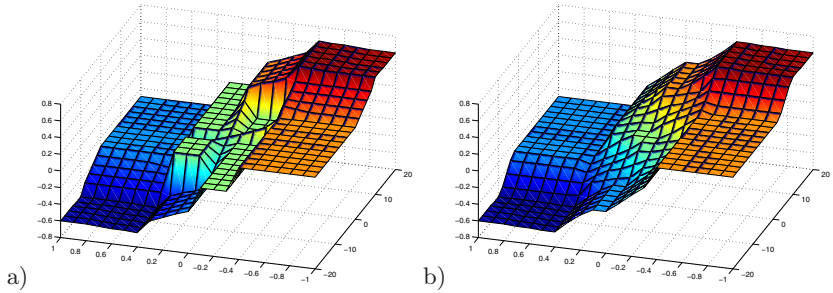


Fig. 3. Control surfaces for output variable *cold* and Rescher-Gaines with: a) S_0 , b) $S_{.3}$ global.

Figure 3 brings the surfaces generated by the output of the shower system for the Rescher-Gaines implicative method before and after the consistency restoration with $S_{.3}$. We note that in (b) we obtain a smooth surface, whereas (a) generates unreasonable 0 outputs. As a matter of fact, these points correspond to the inconsistent outputs; each of them generates the empty set which is defuzzified as the center of the domain, i.e. the value 0. Figure 4 brings a simulation of the shower system using Mamdani and Rescher Gaines with $S_{.3}$. We note that in this application our approach produced outputs that were somewhat closer to the setpoint, with the drawback of producing a slightly larger amount of oscillations (not visible in the figures). As expected, due to the inconsistencies, the use of Rescher-Gaines on the original rule base (i.e. with S_0) does not yield changes as the setpoint is modified (not illustrated here), and is thus useless.

5 Conclusion

Here we have presented an approach to restore consistency in implicative fuzzy rule based systems. We use similarity relations to restore consistency and present a simple yet effective method that can be applied to a large class of rule bases existing in the literature. We have run simulations for a toy problem in fuzzy control and obtained results very similar to Mamdani's traditional method, what allows us to conclude that the use of the approach is very promising.

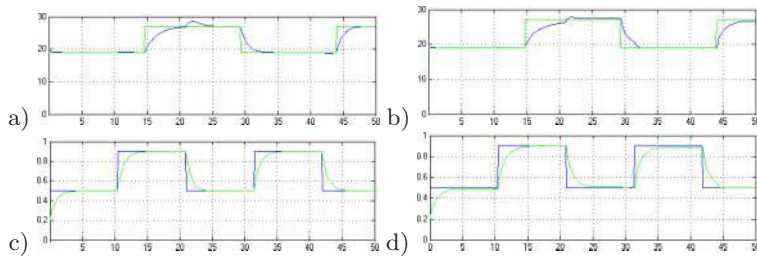


Fig. 4. Simulation flow output: a) Mamdani, b) Rescher-Gaines with S_3 global. Simulation temperature output: c) Mamdani, d) Rescher-Gaines with S_3 global.

Given an inconsistent IFS, there are several ways in which the similarity relation(s) can be used to restore consistency to the system. One basic choice to be made is to which output fuzzy terms should a similarity relation be applied to. Here we have shown a global approach, which consists to apply an adequate similarity relation on all the output terms.

The global approach has the advantage of being straightforward to apply, demanding little programming effort and little computation time. On the other hand, inputs that do not generate an inconsistent output in the original rule base may be negatively affected by the imprecision that the overall application of the similarity relation might generate. Several approaches to deal with inconsistency locally are possible.

In one of these local approaches, given a parameterized family of similarity relations, a parameter is determined for each input such that its application to all the output terms of all the rules fired by it will generate a consistent result. As in the global case, it would be desirable that, for each input leading to inconsistency, the parameter generating less imprecision should be taken. In this approach, an input that generates a consistent output in the original rule base will generate the same output no matter how inconsistent certain areas of the rule base may be. However, being dynamic, its application is more time consuming and may be not feasible in many real-time applications. We have tested this local approach on the shower system and obtained results quite similar to the global ones. However, it is reasonable to suppose that for specific applications one or several approaches ranging from a completely global to a completely local one will be more suitable than others.

Acknowledgements

Lluís Godo and Sandra Sandri respectively acknowledge support from CICYT (TIC 2000-1414, TIC 2001-1577-C03-01) and CNPq (200423/87-8).

References

- [1] Driankov, D., Hellendoorn, H., Reinfrank, M. *An Introduction to Fuzzy Control*. Springer-Verlag, 1996. 392
- [2] Dubois D., Prade H. What are fuzzy rules and how to use them. *Fuzzy Sets and Systems* 84, 169–185, 1996. 387
- [3] Dubois D., Prade H., Ughetto L. Coherence of Fuzzy Knowledge Bases. In *Proc. Fuzz-IEEE'96*, New Orleans (USA), 1858–1864, 1996. 387
- [4] Dubois D., Prade H., Ughetto L. Checking the coherence and redundancy of fuzzy knowledge bases. In *IEEE Trans. on Fuzzy Systems* 5(3), 398–417, 1997. 387, 390, 391, 392
- [5] Godo L., Sandri S. A similarity-based approach to deal with inconsistency in systems of fuzzy gradual rules. In *Proc. of IPMU'02*, Annecy (France), 1655–1662, 2002. 387, 393
- [6] Gottwald S., Petri U. An algorithmic approach towards consistency checking for systems of fuzzy control rules. In *Proc. of EUFIT'95*, Aachen (Germany) 28–31, 1995. 387
- [7] Pedrycz W., Gomide F. *An introduction to Fuzzy sets: Analysis and Design*. MIT Press, 1998. 387
- [8] Perfilieva I., Tonis A. Compatibility of systems of fuzzy relations equations. In *Int. Journal of General Systems* 29(4), 511–528, 2000. 387
- [9] Takagi T., Sugeno T. Fuzzy identification of systems and its application to modeling and control. *IEEE Trans. on Systems, Man and Cybernetics* 15, 116–132, 1985. 386
- [10] Weisbrod J., Fantana N. L. Detecting local inconsistency and incompleteness in fuzzy rule bases. In *Proc. EUFIT'96*, Aachen (Germany) 656–660, 1996. 387
- [11] Yager R. R., Larsen H. L. On discovering potential inconsistencies in validating uncertain knowledge bases by reflecting on the input. *IEEE Trans. on S. M. C.* 21, 790–801, 1991. 387
- [12] Yu W., Bien Z. Design of fuzzy logic controller with inconsistent rule base. *Journal of Intelligent and Fuzzy Systems* 2, 147–159, 1994. 386

Syntactic Analysis for Ellipsis Handling in Coordinated Clauses^{*}

Ralph Moreira Maduro and Ariadne M. B. R. Carvalho

State University of Campinas, Institute of Computing, Brazil

`rmaduro@uol.com.br`

`ariadne@ic.unicamp.br`

Abstract. This work is intended as an investigation into elliptical phenomena in natural language. We argue that some types of ellipsis can be resolved at the syntactic level since they are subject to syntactic constraints. We have dealt with four of the major types of ellipsis found in Portuguese, namely: Null VP, Gapping, Stripping and Sluicing. We have used Island Constraints in order to decide on the grammaticality of the sentence. Finally, we have developed and implemented a syntactically-based algorithm that recovers the elided constituents and reconstructs the elliptical clause, when applicable. The linguistic data in this work is drawn primarily from Portuguese, but we believe that the results can also be applied to other languages, such as English.

1 Introduction

Elliptical structures pose an important problem for Natural Language Processing systems designed to provide text understanding, text generation or dialogue handling. Ellipsis is a grammatical phenomena whereby the structure of the sentence is abbreviated, avoiding redundancy: the sentence, thus, contains a grammatical omission [8]. Although ellipsis may in general be regarded in semantic or pragmatic terms as a means of avoiding redundancy of expression, the kinds of reduction which are allowed are largely a matter of syntax. The fundamental problem posed by an elliptical construction is, therefore, to recover the elided constituent; the actual word(s) whose meaning is understood or implied must be recoverable. There seems to exist two main approaches to ellipsis resolution [6]. Whereas the first tries to associate an elliptical construction directly with a semantic representation, the latter mediates semantic interpretation through the reconstruction of the syntactic structure of the antecedent. We propose an algorithm which implements the second view of ellipsis. We have dealt with sentences involving ellipsis and coordination simultaneously, because the association between the two phenomena is so close that we cannot understand one without understanding the other. The criteria for ellipsis are [8]:

^{*} The authors acknowledge financial support from CNPq (grant 96/10030-3) and FAPESP (grant 96/10028-2).

1. The elided words are precisely recoverable;
2. The elliptical construction is grammatically defective;
3. The insertion of the missing words results in a grammatical sentence, with the same meaning as the original one;
4. The missing words are recoverable from the neighbouring text; and
5. The missing words are an exact copy of the antecedent.

Ellipsis is typically postulated in order to explain why some normally obligatory element of a grammatical sentence is missing. In a context where no ambiguity of reference arises, there is no doubt as to what words are to be supplied. Consider the following two sentences:

João tinha dado esses livros ao filho e Maria também tinha [-]. (1)

[-] = dado esses livros ao filho.

John had given these books to his son and Mary had too.

João gosta de cinema e Pedro [-] de teatro. (2)

[-] = gosta

John likes movies and Peter theater.

In (1) the verb complement “dado esses livros ao filho” is missing, which denotes a defective construction. Nevertheless, the elided words are precisely recoverable from the neighbouring text and are an exact copy of the antecedent. Therefore, the insertion of the missing words results in a grammatical sentence, with the same meaning as the original one. In (2) only the verb “gosta” is missing, but it is recoverable from the neighbouring text. Since the five criteria for ellipsis apply, the sentence is considered grammatical. These criteria undoubtedly help to decide on the grammaticality of the sentence through the reconstruction of the elliptical clause. However, when we have a sentence such as

* João conquistou a confiança de seu chefe e Maria não admite a hipótese de que Pedro também [-]. (3)

John has gained his boss's confidence and Mary doesn't admit the hypothesis that Peter too.

we cannot reconstruct the elliptical clause using the words “conquistou a confiança de seu chefe” because sentence (3) is ungrammatical at first place. This is due to the fact that some types of ellipsis are subject to syntactic constraints which determine when neighbouring text can be used to fill the gap of the elliptical clause. In general the kinds of ellipsis vary to some extent from one language to another. Specifically for Portuguese, De Matos [7] has identified five types of ellipsis and studied two of them in detail, namely Null VP and Stripping. She concluded that whereas Stripping is subject to Island Constraints [4], Null VP is not. She has shown that besides following the five criteria for ellipsis reconstruction, we must also take these constraints into account when dealing with Stripping and before reconstructing the elliptical clause. Following De Matos's approach, we have examined two other ellipsis occurrences in Portuguese, namely Gapping and Sluicing regarding syntactic constraints. We have shown that these

two types of ellipsis are also subject to Island Constraints and, therefore, during sentence reconstruction these constraints must be respected. We have developed and implemented an algorithm which takes the five criteria for ellipsis and the Island Constraints into account in order to reconstruct the elliptical clause.

The remaining part of this paper is organized as follows. The next section describes other approaches to ellipsis resolution. Section 2 presents the five major types of ellipsis found in Portuguese. Section 3 discusses how syntactic constraints can be used to decide on the grammaticality of elliptical sentences. Section 4 presents a syntactically-based algorithm to recover the elided constituents. Finally, section 5 presents our concluding remarks.

2 Other Approaches to Ellipsis

There are several other approaches to ellipsis resolution. Dalrymple et al. [3] present a generalized semantic approach which employs higher-order unification of property and relation variables to resolve ellipsis. The method presupposes a semantic representation of the antecedent clause and it is argued that the antecedent and the elliptical clause share the same property which, when applied to both clauses, allows for the correct interpretation of the sentence. The strategy is to specify the interpretation of the antecedent clause as an equation between a propositional variable and a predicate argument structure. The arguments of the predicate correspond to the fragments in the ellipsis site, and the ellipsis resolution consists in finding an appropriate value for the predicate variable which can apply to both the sequence of arguments in the interpretation of the antecedent clause, and the sequence of arguments in the ellipsis site.

However, as Lappin [6] pointed out, it is not clear how higher-order unification can be applied to sentences like

John sings, and beautifully too. (4)

where there is no corresponding element in the antecedent clause.

Lappin [6] suggested to point a free manner adverbial function variable in the lexical semantic representation of verbs like “sing”. This approach will allow for the correct semantic interpretation of sentence (4), but it still cannot be generalized to sentences such as

John sang, but not in New York. (5)

The same article also presented a syntactically-based algorithm to deal with the following types of ellipsis: VP ellipsis, Pseudo-gapping, Stripping and Gapping. The algorithm treats ellipsis resolution as the specification of a relation of correspondence between an unrealized verbal head of an elliptical clause and its arguments and adjuncts as one term of the relation, and the realized head of the antecedent clause and its arguments and adjuncts as the second term. When analysing sentence (4), for example, the algorithm will identify “sings” as the head of the antecedent clause and substitute it for the empty verb. This will produce the following sentence:

John sings and John sings beautifully too. (6)

Kehler [5] has another approach based on the following discourse relations: cause-effect and resemblance. He uses these two relations to identify which method should be used for ellipsis resolution. The resemblance relation, for example, requires diverging or converging points between the two involved clauses. He argues that in a resemblance relation the entities present in both clauses share the same property, that is, they act in a similar way in the information context. In a cause-effect relation, on the other hand, the two clauses do not have to share the same property, but there must exist an implication relation between the two clauses, that is, they must be interdependent.

Based on these discourse relations, Kehler proposed a method to identify whether syntactic or semantic analysis should be used to ellipsis resolution. If a resemblance relation holds between the two clauses, then a syntactic approach to ellipsis must be adopted; if, on the other hand, a cause-effect relation holds between the two clauses, then a semantic approach must be taken. He argues that when the identity between the two clauses is semantic, then a syntactic structure of the antecedent as well as syntactic restrictions are not necessary. When a resemblance relation holds between two clauses, the sentence is subject to syntactic restrictions. If there is not an adequate syntactic structure to recover the elliptical constituent, the sentence is considered ungrammatical.

3 Major Types of Ellipsis in Portuguese

According to De Matos [7], the major kinds of ellipsis found in Portuguese are: Null VP, Gapping, Stripping, Sluicing and Conjunction Reduction. The difference lies on the type of the structure of the missing constituent.

3.1 Conjunction Reduction

In a Conjunction Reduction ellipsis, a subject noun phrase and, eventually, a verbal constituent are elided from the sentence. In the following, the subject (“João”) and the auxiliary verb (“tem”) are elided from the second clause:

João tem comprado livros aos filhos e [-] oferecido flores à mulher. (7)

[-] = [João tem]

John has bought books for his children and offered flowers to his wife.

3.2 Gapping

In a Gapping occurrence a verb and, optionally, its complements are ellipted, but two other constituents are lexically realized, one of them being usually the subject. In the following sentence

João deu flores a sua mãe e Pedro [-] chocolates [-]. (8)

[-] = [deu]

[-] = [a sua mãe]

John gave flowers to his mother and Peter chocolates.

verb “deu” and its complement “a sua mãe” are elided from the sentence.

3.3 Sluicing

In a Sluicing occurrence an interrogative constituent remains lexically realized as the only representative of a clause. Consider the following sentence:

- Alguém veio lhe procurar, mas eu não sei quem [-]. (9)
 [-] = [veio lhe procurar].
Someone came looking for you, but I don't know who.

Here, the pronoun “quem” stands for the elided words “veio lhe procurar”.

3.4 Null VP

In a Null VP¹ occurrence the verb or an auxiliary verb, when an auxiliary verb is present in the first clause, and an adverb, are lexically realized in the elliptical clause. Consider the following sentence:

- Maria atribuiu o desastre ao motorista e Tereza também atribuiu [-]. (10)
 [-] = [o desastre ao motorista].
Mary blamed the disaster on the driver and so did Theresa.

the verb in both coordinated clauses is identical (“atribuiu”); also, an adverb (“também”) is present in the elliptical clause.

3.5 Stripping

In a Stripping occurrence all constituents, except one and an adverb, are missing. In Portuguese we can find the adverbs “não”, “sim”, “também” and “também não”, whose presence in a Stripping ellipsis is compulsory; their function is to recover the constituent which is the predicate of the elliptical clause. Consider the three sentences below:

- Maria atribuiu o desastre ao motorista e Teresa também [-]. (11)
 [-] = [atribuiu o desastre ao motorista].
Mary blamed the disaster on the driver and Theresa did too.

- Maria atribuiu o desastre ao motorista e [-] a fuga dos assaltantes também. (12)
 [-] = [Maria atribuiu]
Mary blamed the disaster on the driver and the assailants' escape too.

- Maria ouve o noticiário à hora do almoço e [-] à hora do jantar também. (13)
 [-] = [Maria ouve o noticiário]
Mary listens to the news at lunch and at dinner time too.

In (11) a verb phrase (“atribuiu o desastre ao motorista”) is missing; in (12) a subject followed by a verb (“Maria atribuiu”) is elided from the sentence; in (13) a noun phrase followed by a verb phrase (“Maria ouve o noticiário”) are elided from the second clause.

¹ Sometimes also called VP deletion.

4 Syntactic Constraints on Ellipsis

The fundamental problem of elliptical constructions is to recover the elliptical constituents. De Matos [7] has studied Null VP and Stripping in detail and observed that, although these types of ellipsis seem to be very similar on the surface, they are very different when syntactic constraints are concerned. They both require a linguistic antecedent and a lexically realized adverb in order to be grammatical. Consider the following two sentences:

Maria tinha atribuído o desastre ao motorista e Teresa também tinha [-]. (14)

[-] = [atribuído o desastre ao motorista].

Mary had blamed the disaster on the driver and Theresa had too.

Maria tinha atribuído o desastre ao motorista e Teresa também [-]. (15)

[-] = [tinha atribuído o desastre ao motorista].

Mary had blamed the disaster on the driver and Theresa too.

Sentences (14) and (15) present an elliptical predicate and, although both predicates involve a VP, the structure of the two sentences is different. Sentence (14) is an example of Null VP ellipsis because a constituent (“Teresa”), an adverb (“também”) and an auxiliary verb (“tinha”) are lexically realized. Sentence (15) is an example of Stripping since only one constituent (“Teresa”) and an adverb (“também”) are realized in the elliptical clause.

De Matos observed that only when attempting to recover elliptical constituents in a Stripping occurrence we must take Island Constraints into account. Therefore, Stripping must follow the Island Constraint, which states that, when a constituent is moved, it must cross the minimal number of barriers, preferably none [7].

Traditionally this constraint is used to restrict movement of constituents within a sentence [2]. De Matos has shown that the same principle can be applied, in a similar manner, during the search for an antecedent which can be used to reconstruct the elliptical clause in the resolution process. Consider:

* Que João vá é bom, mas [^{IP CP} [que Maria não [-]] é péssimo]. (16)

That John goes is good, but that Mary doesn't is awful.

Que João tenha ido é bom, mas [^{IP CP} [que Maria não tenha [^{VP} -]] é péssimo]. (17)

That John has gone is good, but that Mary hasn't is awful.

Sentence (16) is an example of Stripping. It is ungrammatical because, since this type of ellipsis is subject to Island Constraints, the antecedent “vá” cannot be used to fill the gap in the elliptical clause because in order to do that more than one barrier would have to be crossed. In (17), on the other hand, we have a Null VP which is not sensitive to Island Constraints. Therefore, we can use the antecedent “ido” to fill the gap in the elliptical clause and the sentence is considered grammatical.

The following sentences are examples of Stripping and Null VP ellipsis. Whereas the Stripping manifestations are ungrammatical, the Null VP ellipsis are not. In (18) and (19) we have a complex NP in a relative clause. In (18) we cannot use the constituent “falado japonês” to fill the gap in the elliptical clause because this would infringe Island Constraints. Sentence (19) is a Null VP occurrence and, therefore, the words “falado japonês” can be used to fill the gap of the elliptical clause because this type of ellipsis is not sensitive to Island Constraints.

- * João fala japonês e eu conheço $\left[\begin{smallmatrix} \text{um aluno} \\ \text{NP} \end{smallmatrix} \left[\begin{smallmatrix} \text{que também} \\ \text{CP} \end{smallmatrix} \left[\begin{smallmatrix} \text{ } \\ \text{VP} \end{smallmatrix} \right] \right] \right]$. (18)
John speaks Japanese and I know a student who too.

- João tem falado japonês ultimamente
 e eu conheço $\left[\begin{smallmatrix} \text{um aluno} \\ \text{NP} \end{smallmatrix} \left[\begin{smallmatrix} \text{que também tem} \\ \text{CP} \end{smallmatrix} \left[\begin{smallmatrix} \text{ } \\ \text{VP} \end{smallmatrix} \right] \right] \right]$. (19)
John has spoken Japanese lately and I know a student who has too.

In sentence (20) below we have a complex NP, and, again, we cannot use “está doente” to fill the gap of the elliptical clause, because the Island Constraints would be violated.

- * João está doente e Maria não admite
 $\left[\begin{smallmatrix} \text{a hipótese} \\ \text{NP} \end{smallmatrix} \left[\begin{smallmatrix} \text{de que ela também} \\ \text{CP} \end{smallmatrix} \left[\begin{smallmatrix} \text{ } \\ \text{VP} \end{smallmatrix} \right] \right] \right]$. (20)
John is ill and Mary doesn't admit the hypothesis that she too.

- João está doente e Maria não admite
 $\left[\begin{smallmatrix} \text{a hipótese} \\ \text{NP} \end{smallmatrix} \left[\begin{smallmatrix} \text{de que ela também esteja} \\ \text{CP} \end{smallmatrix} \left[\begin{smallmatrix} \text{ } \\ \text{VP} \end{smallmatrix} \right] \right] \right]$. (21)
John is ill and Mary doesn't admit the hypothesis that she is too.

In sentence (21), on the other hand, the antecedent “doente” is used to fill the gap in the elliptical clause and the sentence is considered grammatical.

Basing our work on De Matos’s approach, we have analysed two other types of ellipsis - Gapping and Sluicing - regarding Island Constraints applied to ellipsis resolution. Consider the following example of Gapping ellipsis:

- * João perguntou $\left[\begin{smallmatrix} \text{o que você comeu hoje} \\ \text{CP} \end{smallmatrix} \right]$ e Pedro $\left[\begin{smallmatrix} \text{ } \\ \text{VP} \end{smallmatrix} \right]$ ontem. (22)
John asked what you have eaten today and Peter yesterday.

This example is ungrammatical because, although it is in accordance with the definition of gapping, the elliptical clause cannot be reconstructed with the sub-constituents “perguntou o que você comeu hoje” since this would represent a violation of the Island Constraint.

Consider now the following example of Sluicing ellipsis:

- * $\left[\begin{smallmatrix} \text{Que João vá ao cinema} \\ \text{IP} \end{smallmatrix} \right]$ é bom, mas com quem $\left[\begin{smallmatrix} \text{ } \\ \text{CP} \end{smallmatrix} \right]$. (23)
That John goes to the movies is good, but with whom.

In order to fill the gap left in the elliptical clause, we would have to violate the Island Constraint. Therefore, the sentence is considered ungrammatical.

In conclusion, Gapping and Sluicing are also subject to Island Constraints. Therefore, a syntactically-based system for ellipsis resolution must take these constraints into account.

5 An Algorithm for Ellipsis Resolution

We developed an algorithm which deals only with sentences involving coordination and ellipsis simultaneously and which takes Island Constraints into account in order to reconstruct the elided material. The algorithm works as follows:

1. Decomposing the sentence into syntactic structures;
2. Identifying the type of ellipsis present in the sentence;
3. Checking if this type of ellipsis is subject to syntactic constraints;
4. Identifying the antecedent of the elided term; and
5. Reconstructing the elided constituent.

Based on this algorithm we have developed a system that decomposes the sentence into syntactic structures using a syntactic parser which deals with elliptical constructions. The grammar, thus, allows elided constituents wherever the four types of ellipsis treated in this work would. So, for example, the syntactic analyser works on the following sentence

João fala japonês e Carlos também fala [-]. (24)
John speaks Japanese and Charles speaks too.

producing the syntactic structure shown below and the derivation tree shown in Fig. 1.

$$\left[\left[\left[\text{João} \right] \left[\left[\text{fala} \right] \left[\text{japonês} \right] \right] \right] \left[e \right] \right. \\ \left. \left[\left[\text{Carlos} \right] \left[\left[\text{também} \right] \left[\text{fala} \right] \left[- \right] \right] \right] \right].$$

$\begin{matrix} S & AC & NP & & VP & V & & NP & & C \\ EC & NP & & & VP & ADV & & V & & NP \end{matrix}$

The next step is to identify the kind of ellipsis present in the sentence. In (24) the system identifies that the constituents which have been lexically realized in the elliptical clause are the noun phrase “Carlos”, the adverb “também” and

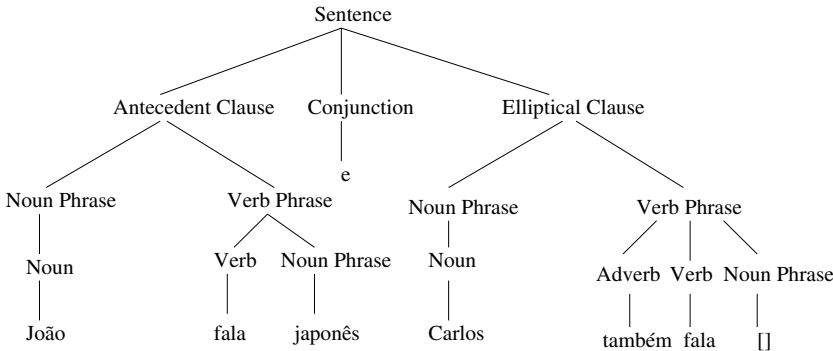


Fig. 1. An example of Null VP ellipsis

the verb “fala”. This configures a Null VP ellipsis. Since this type of ellipsis is not subject to Island Constraints the system recovers the antecedent (that is the noun phrase “japonês”) and reconstructs the sentence using it to fill the gap in the elliptical clause.

Sentence (25), on the other hand, is an example of stripping ellipsis, since the lexically realized terms are “quem” and the adverb “também”.

- * João vai ao cinema e Maria perguntou [quem ^{CP} também ^{IP} [-]]. (25)
John is going to the movies and Mary asked who too.

The system identifies this is an Island context because CP constitutes a barrier and, therefore, reconstruction does not take place and the sentence is considered ungrammatical.

Gapping is also subject to Island constraints. Consider now the following sentence, which is an example of Gapping ellipsis:

- João deu flores a sua mãe e Carlos [-] chocolates [-]. (26)
John gave flowers to his mother and Charles chocolates.

The constituents which are lexically realized are “Carlos” and “chocolates” and the correspondent syntactic structure generated by the system is shown below.

$$\begin{array}{c} \left[\left[\left[\text{João} \right] \left[\left[\text{deu} \right] \left[\text{flores} \right] \left[\text{a sua mãe} \right] \right] \right] \left[\text{e} \right] \right. \\ \text{S} \quad \text{AC NP} \quad \text{VP V} \quad \text{NP} \quad \text{PP} \quad \text{C} \\ \left. \left[\left[\text{Carlos} \right] \left[\left[\text{deu} \right] \left[\text{chocolates} \right] \left[- \right] \right] \right] \right] \right. \\ \text{EC NP} \quad \text{VP V} \quad \text{NP} \quad \text{PP} \end{array}$$

Verb “dar” and “a sua mãe” are the antecedents present in the first clause. The elliptical sentence is reconstructed because the missing terms are not inside an Island context. Therefore the sentence is reconstructed as:

$$\begin{array}{c} \left[\left[\left[\text{João} \right] \left[\left[\text{deu} \right] \left[\text{flores} \right] \left[\text{a sua mãe} \right] \right] \right] \left[\text{e} \right] \right. \\ \text{S} \quad \text{AC NP} \quad \text{VP V} \quad \text{NP} \quad \text{PP} \quad \text{C} \\ \left. \left[\left[\text{Carlos} \right] \left[\left[\text{deu} \right] \left[\text{chocolates} \right] \left[\text{a sua mãe} \right] \right] \right] \right] \right. \\ \text{EC NP} \quad \text{VP V} \quad \text{NP} \quad \text{PP} \end{array}$$

Finally, sentence (27) is an example of Sluicing ellipsis.

- João sabe que os garotos sairão, mas ele não sabe quando [-]. (27)
John knows that the boys will leave, but he doesn't know when.

Pronoun “quando” represents the antecedent clause “os garotos sairão”, as it can be seen in the syntactic structure shown below.

$$\begin{array}{c} \left[\left[\left[\text{João} \right] \left[\left[\text{sabe} \right] \left[\text{que os garotos sairão} \right] \right] \right] \left[\text{mas} \right] \right. \\ \text{S} \quad \text{AC NP} \quad \text{VP V} \quad \text{NP} \quad \text{C} \\ \left. \left[\left[\text{ele} \right] \left[\left[\text{não} \right] \left[\text{sabe} \right] \left[\text{quando} \right] \right] \right] \right] \right. \\ \text{EC NP} \quad \text{VP ADV} \quad \text{V} \quad \text{WH} \end{array}$$

Since the antecedent can be recovered without violating the Island Constraints, the sentence is reconstructed by the system as follows:

$$\begin{array}{c} \left[\left[\left[\text{João} \right] \left[\left[\text{sabe} \right] \left[\text{que os garotos sairão} \right] \right] \right] \left[\text{mas} \right] \right. \\ \text{S} \quad \text{AC NP} \quad \text{VP V} \quad \text{NP} \quad \text{C} \\ \left. \left[\left[\text{ele} \right] \left[\left[\text{não} \right] \left[\text{sabe} \right] \left[\text{quando os garotos sairão} \right] \right] \right] \right] \right. \\ \text{EC NP} \quad \text{VP ADV} \quad \text{V} \quad \text{WH} \end{array}$$

6 Remarks and Conclusion

We have proposed a syntactically-based algorithm for ellipsis resolution and we have argued that for some types of ellipsis, a syntactic structure is required in order to reconstruct the elliptical clause appropriately. We have not only used the syntactic structure of the antecedent to reconstruct the elliptical clause, but we have also taken syntactic constraints into consideration to check if the elliptical clause can actually be reconstructed. We have used Island Constraints to reconstruct the elliptical clause. Our approach is based on DeMatos' approach to deal with Stripping and Null VP. We have gone one step further dealing with Gapping and Sluicing also. The basic strategy which the algorithm encodes is to reconstruct the elided clause by (i) decomposing the sentence into syntactic structures; (ii) identifying the type of ellipsis present in the sentence; (iii) checking if this type of ellipsis is subject to syntactic constraints; (iv) identifying the antecedent; and (v) reconstructing the elided constituent. Future work includes studying other types of ellipsis, such as nominal ellipsis, as well as other syntactic restrictions on ellipsis [1]. Although the linguistic data in this work is drawn primarily from Portuguese, we believe that the results can also be applied to other languages. Future work also includes investigating how much of the work described here can be applied to other languages, such as English.

Acknowledgments

We are grateful to Ivan Santa Maria Filho who provided valuable support during the implementation phase. We are also grateful to Jorge Stolfi who provided valuable suggestions on an earlier version of the paper.

References

- [1] Chomsky, Noam. *Lectures on Government and Binding*. Dordrecht:Foris. 1981. 406
- [2] Chomsky, Noam. *Barriers*. The Massachusetts Institute of Technology. 1986. 402
- [3] Dalrymple, Mary, Stuart M. Shieber, and Fernando C. N. Pereira. Ellipsis and higher-order unification. Technical report, Computation and Language E-Print Archive. 1991. 399
- [4] Haegeman, Liliane. *Introduction to Government and Binding Theory*. Blackwell Oxford UK and Cambridge USA. 1992. 398
- [5] Kehler, Andrew. *Interpreting Cohesive Forms in the Context of Discourse Inference*. Ph.D. thesis, Harvard University. 1995. 400
- [6] Lappin, Shalom. The interpretation of ellipsis. *The Handbook of Contemporary Semantic Theory*. 1996. 397, 399
- [7] Matos, Maria Gabriela Ardisson Pereira De. *Construções de Elipse do Predicado em Português - Sv Nulo e Despojamento*. Ph.D. thesis, Universidade de Lisboa. 1992. 398, 400, 402
- [8] Quirk, Randolph, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. *A Comprehensive Grammar of the English Language*. Longman. 1985. 397

Assessment of Selection Restrictions Acquisition

Alexandre Agustini*, Pablo Gamallo**, and Gabriel P. Lopes

Departamento de Informática Universidade Nova de Lisboa, Portugal
{aagustini,gamallo,gpl}@di.fct.unl.pt

Abstract. This paper describes an automatic clustering strategy for acquiring both syntactic and semantic subcategorization restrictions from corpora. In order to test our method, preliminary experiments have been performed on a law-case Portuguese corpus. The acquired information is then used for lexicon upgrading and it is validated by a parsing diagnosis system.

1 Introduction

Development of robust syntactic parsers for natural language texts requires resolution of syntactic ambiguity. Most modern natural language processing techniques rely on a subcategorization lexicon to restrict possible parses. The goal of our work is to learn predicate-argument (subcategorization) information and to apply this information to the parsing task.

Words are combined following specific linguistic constraints. The constraints imposed by a particular word (predicate) in order to limit the words with which it can combine (arguments) are known as subcategorization restrictions. Subcategorization is expressed at both syntactic (subcategorization frames) and semantic (selection restrictions) levels of abstraction. Syntactic frames are based on constraints referring to morphosyntactic categories and syntactic position. Selection restrictions, on the other hand, require arguments to match a specific semantic class. The parser needs both syntactic constraints and selection restrictions information to prefer some parses from several possible grammatical ones.

This paper describes an unsupervised method for acquiring both syntactic frames and selection restrictions. The acquired information is validated based on the following steps. First, the learned information is introduced on the system dictionary as subcategorization hypotheses; then we use a logic based diagnosis system to parse the text using these hypotheses; finally the results are manually evaluated.

The remainder of the article is organized as follows. Section 2 introduces theoretical concepts like contextual and co-specification hypothesis. Next, in section 3 we describe an automatic method for clustering the words appearing in similar syntactic frames. Finally, section 4 deals with upgrading the lexicon with subcategorization information and validating them with a diagnosis parser.

* Research sponsored by CAPES and PUCRS – Brazil.

** Research supported by the PRAXIS XXI project, FCT/MCT, Portugal.

2 Linguistic Basics

According to Gregory Grefenstette [10, 11], knowledge-poor approaches use no presupposed semantic knowledge for automatically extracting semantic information. They are characterized as follows: no domain-specific information is available, no semantic tagging is used, and no static sources as machine readable dictionaries or handcrafted thesauri are required. Hence, they differ from knowledge-rich approaches in the amount of linguistic knowledge they need to activate the semantic acquisition process. Whereas knowledge-rich approaches require previously encoded semantic information (semantic tagged corpora and/or man-made lexical resources [20, 1]), knowledge-poor methods only need a coarse-grained notion of linguistic information: word cooccurrence. In particular, the main aim of knowledge-poor approaches is to calculate the frequency of word cooccurrences within either syntactic constructions or sequences of n-grams in order to extract semantic information such as selection restrictions [12, 4], and word ontologies [18, 10, 15]. Since these methods do not require previously defined semantic knowledge, they overcome the well-known drawbacks associated with handcrafted thesauri and supervised strategies.

Nevertheless, our method differs from standard knowledge-poor strategies on two specific issues: both the way of extracting word similarity and the way of defining syntactic contexts. Our strategy relies on two basic linguistic assumptions. First, we assume that two syntactically related words impose semantic selectional restrictions to each other (*co-specification*). Second, it is also claimed that two syntactic contexts impose the same selection restrictions if they cooccur with the same words (*contextual hypothesis*).

Co-specification is based on the following idea. Two syntactically dependent expressions are no longer interpreted as a standard pair “predicate-argument”, where the predicate has the active function imposing the semantic preferences on a passive argument, which matches such preferences. On the contrary, each word of a binary dependency is perceived simultaneously as a predicate and an argument [19, 7]. That is, each word both imposes semantic restrictions and matches semantic requirements. For instance, let’s take the phrase *infringe the law*, in classical linguistic approaches the verb *infringe* is the active head that imposes specific restrictions on its argument, it must belong to the class of legal documents, while *the law* is the passive entity that fulfils the conditions imposed by the head. On the other hand, in recent linguistic works, the noun *law* also imposes restrictions on the verb with which it is in relation of direct object. These verbs are transitive verbs like *refuse*, *apply*, *approve*, ... This is called co-specification.

In order to extract contextual word classes from the appropriate syntactic constructions, we claim that similar syntactic contexts share the same semantic restrictions on words. Instead of computing word similarity on the basis of the too coarse-grained Harris’ *distributional hypothesis* (according to this assumption, words cooccurring in similar syntactic contexts are semantically similar and, then, should be clustered into the same semantic class), we measure the similarity between syntactic contexts in order to identify common selection re-

strictions. More precisely, we assume that two syntactic contexts occurring with (almost) the same words are similar and, then, impose the same semantic restrictions on those words. That is what we call *contextual hypothesis*. Semantic extraction strategies based on the contextual hypothesis may account for the semantic variance of words in different syntactic contexts. Since these strategies are concerned with the extraction of semantic similarities between syntactic contexts, words will be clustered with regard to their specific syntactic distribution. Such clusters represent context-dependent semantic classes. Except for the co-operative system Asium introduced in [6, 5, 2], few or no research on semantic extraction have been based on such hypothesis.

3 Acquisition Method Overview

To evaluate the hypotheses presented above, a software package was developed to support the automatic acquisition of semantic restrictions. The system is constituted by three related modules: extracting, filtering and clustering.

3.1 Extracting

Raw text is tagged [17] and partially analyzed [21]. Then, an attachment heuristic is used to identify *binary dependencies*. The result is a list of cooccurrence triplets containing the syntactic relationship and the lemmas of the two related head words [9]. For example, the phrase *infringement of the law* would produce the following attachment:

$$(of; infringement^{\downarrow}, law^{\uparrow})^1$$

Binary dependencies are used to extract *Syntactic contexts*. Unlike most work on selection restrictions learning, the characterization of syntactic contexts relies on the dynamic process of co-specification. So, two syntactic contexts are generated from the previous dependency:

$$\begin{aligned} &[\lambda x^{\downarrow}(of; x^{\downarrow}, law^{\uparrow})] \\ &[\lambda x^{\uparrow}(of; infringement^{\downarrow}, x^{\uparrow})] \end{aligned}$$

3.2 Filtering

According to the contextual hypothesis introduced above, two syntactic contexts that select for the same words should have the same extensional definition and, then, the same selection restrictions. So, if two *contextual word sets*² are

¹ We represent a dependency between two words $w1$ and $w2$ as the binary predication: $(r; w1^{\downarrow}, w2^{\uparrow})$, where the binary predicate r is associated to specific prepositions, subject relations, direct object relations, etc.; the roles of the predicate, “ $\downarrow$ ” and “ $\uparrow$ ”, represent the *head* and *complement* roles, respectively.

² The set of words that occurs with the *Syntactic context*.

considered as similar [8], we infer that their associated syntactic contexts are semantically similar and share the same selection restrictions. In addition, we also infer that these contextual word sets are semantically homogeneous and represent a contextually determined class of words. Let's take the two following syntactic contexts and their associated contextual word sets:

$$\begin{aligned} [\lambda x^\uparrow(of; infringement^\downarrow, x^\uparrow)] &= \{article\ law\ norm\ precept\ statute\ \dots\} \\ [\lambda x^\uparrow(dobj; infringe^\downarrow, x^\uparrow)] &= \{article\ law\ norm\ principle\ right\ \dots\} \end{aligned}$$

Since both contexts share a significant number of words, it can be argued that they share the same selection restrictions. Furthermore, it can be inferred that their associated contextual sets represent the same context-dependent semantic class. In our corpus, context $[\lambda x^\uparrow(dobj; violar^\downarrow, x^\uparrow)]$ (*to infringe*) is not only considered as similar to context $[\lambda x^\downarrow(dobj; violação^\downarrow, x^\uparrow)]$ (*infringement of*), but also to other contexts such as:

- $[\lambda x^\downarrow(dobj; respeitar^\downarrow, x^\uparrow)]$ (*to respect*)
- $[\lambda x^\uparrow(dobj; aplicar^\downarrow, x^\uparrow)]$ (*to apply*)

As has been said in the introduction, the cooperative system Asium is also based on the contextual hypothesis [6, 5]. This system requires the interactive participation of a language specialist in order to filter and clean the word sets when they are taken as input of the clustering strategy. Such a cooperative method requires manual removal of those words that have been incorrectly tagged or analyzed from the sets. Our strategy, by contrast, automatically removes incorrect words from sets. Automatic filtering requires the following sub-tasks:

First, each word set is associated with a list of its most similar sets. Intuitively, two sets are considered as similar if they share a significant number of words. Various similarity measure coefficients were tested to create lists of similar sets. The best results were achieved using a particular weighted version of the Jaccard coefficient, where words are weighted considering both their dispersion and their relative frequency for each context. The statistical motivation of this similarity measure and its application is discussed in [8].

Then, once each contextual set has been compared to the other sets, we select the words shared by each pair of similar sets, i.e., we select the intersection between each pair of sets considered as similar. Since words that are not shared by two similar sets could be incorrect words, we remove them. Intersection allows us to clear sets of words that are not semantically homogeneous. Thus, the intersection of two similar sets represents a semantically homogeneous class, which we call *basic class*.

3.3 Conceptual Clustering

We use an agglomerative (bottom-up) clustering for successively aggregating the previously created basic classes. Unlike most research on conceptual clustering,

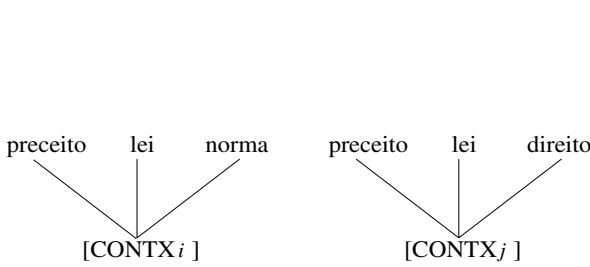


Fig. 1. Basic classes

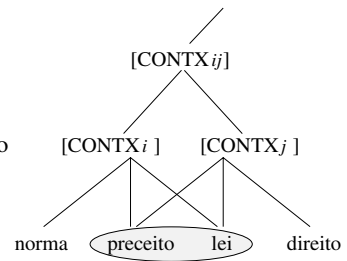


Fig. 2. Agglomerative clustering

aggregation does not rely on a statistical distance between classes, but on empirically set conditions and constraints [22]. These conditions are discussed in [8]. Figure 1 shows two basic classes associated with two pairs of similar syntactic contexts. $[CONTX_i]$ represents a pair of syntactic contexts sharing the words *preceito*, *lei*, *norma* (*precept*, *law*, *norm*, and $[CONTX_j]$ represents a pair of syntactic contexts sharing the words *preceito*, *lei*, *direito* (*precept*, *law*, *right*). Both basic classes are obtained from the filtering process described in the previous section. Figure 2 illustrates how basic classes are aggregated into more general clusters. If two classes fill the clustering conditions, they can be merged into a new class. The two basic classes of the example are clustered into the more general class constituted by *preceito*, *lei*, *norma*, *direito*. Such a generalization leads us to induce syntactic data that does not appear in the corpus. Indeed, we induce both that the word *norma* may appear in the syntactic contexts represented by $[CONTX_j]$, and that the word *direito* may be attached to the syntactic contexts represented by $[CONTX_i]$.

4 Tests and Evaluation

The system was tested over a small corpus with 1,643,579 word occurrences selected the Portuguese text corpora *P.G.R.*³. The fact of using specialized text corpora makes easier the learning task, given that we have to deal with a limited vocabulary with reduced polysemy. Furthermore, since the system is not dependent on any specific language such as Portuguese, it could be applied, in principle, to whatever natural language.

First, the corpus was tagged by the part-of-speech tagger presented in [17]. Then, it was chunked by using the partial parser presented in [21]. The chunks were attached using the right association heuristic so as to create binary dependencies. 211,976 different syntactic contexts with their associated word sets were extracted from these dependencies. Then, we filtered these contextual word sets by using the method described above so as to obtain a list of basic classes, and

³ P.G.R. (*Portuguese General Attorney Opinions*) is constituted by case-law documents.

Table 1. Some sample clusters

cl04575	aprovar definir indicar mencionar prever qualificar referir <i>approve define indicate mention foresee qualify refer</i>
cl03928	considerar constituir criar definir determinar integrar referir <i>consider constitute create define determinate integrate refer</i>
cl04141	atividade atribuição cargo função funções tarefa trabalho <i>activity attribution function/post function functions task work</i>
cl05130	administração cargo categoria exercício função lugar regime serviço <i>administration post rank practice function place regime service</i>

finally we used the clustering algorithm and obtained 6024 clusters. In table 1, we show some of the clusters generated by the algorithm.⁴

Note that some words may appear in different clusters. For instance, **cargo** (*function/post*) is associated with nouns referring to activities (e.g., **atividade**, **trabalho**, **tarefa** (*activity*, *work*, *task*)), as well as with nouns referring to the positions where those activities are produced (e.g., **cargo**, **categoria**, **lugar** (*post*, *rank*, *place*)). The sense of polysemic words is represented by natural assignment of a word to various clusters.

4.1 Dictionary Update

Our clustering strategy does not generate ontological classes like *human beings*, *institutions*, *vegetables*, ..., but context-based semantic classes associated with syntactic contexts. Consider the class of verbs *cl04575* illustrated in the first row of Table 1. These verbs belong to the same context-based semantic class because they share a significant number of syntactic contexts. Below, we show some of the syntactic contexts used to generate this class:

$$\begin{aligned}
 [\lambda x^\downarrow (dobj; x^\downarrow, carreira^\uparrow)] &= cl04575 \\
 [\lambda x^\downarrow (dobj; x^\downarrow, ordenamento^\uparrow)] &= cl04575 \\
 [\lambda x^\downarrow (iobj\_em; x^\downarrow, assembléia^\uparrow)] &= cl04575 \\
 [\lambda x^\downarrow (iobj\_em; x^\downarrow, tribunal^\uparrow)] &= cl04575
 \end{aligned}$$

That means that nouns **carreira** (*career*) and **ordenamento** (*organization/planning*) are required to be direct object (i.e., *dobj*) of the verbs belonging to class *cl04575*. Likewise, **assembléia** (*assembly*) and **tribunal** (*tribune*) are required to be “em-complement” (i.e., *iobj_in*) of these class of verbs.

Since the generated clusters are not linguistic-independent objects but semantic requirements taking part in attachment resolution, they are used to update the lexicon with subcategorization information. In Table 2 we show some examples of lexical entries.

Each entry contains both the list of syntactic contexts representing its syntactic subcategorization, and the list of word sets required by the syntactic contexts.

⁴ The left column contains the identifier codes of generated clusters.

Such word sets are viewed as the extensional mode of representation of the semantic preferences required by the syntactic contexts. Consider the information our system learnt for verb **emanar** (see table 2). It syntactically subcategorizes two kinds of “de-complements”: one semantically requires words referring to legal documents (**emana do artigo** - the literal translation *emanates from the article* should be replaced by *article prescribes*), the other selects words referring to institutions (**emana da autoridade** - *emanates from the authority; authority proposes*).

Take now the noun **fase**. It is encoded in our dictionary as a word involved in three different syntactic contexts, each of them having different semantic preferences. First, it is required to be direct object of transitive verbs (**abrange a fase** - *comprise the phase*); second, it is also required to be “em-complement” of locative verbs (**encontrar na fase** - *situates in the phase*); third, it requires “de-complements” describing temporal entities (**fase do procedimento** - *procedure phase*).

4.2 Diagnosis Parser

The work reported was carried out in order to learn syntactic and semantic subcategorization to achieve improved syntactic parsing. We started with no knowledge about subcategorization. So, no noun phrase or prepositional phrase could attach to any verb or noun. But our parsing strategy [21, 16] was developed for enabling different parsing stages. Once it is found that the parsing is

Table 2. Dictionary entries

emanar (<i>emanate</i>)	$[\lambda x^\uparrow(iobj\_de; emanar^\downarrow, x^\uparrow)] =$ {alínea artigo código decreto diploma disposição estatuto legislação lei norma regulamento} (<i>paragraph article code decret diploma disposition statute legislation law norm regulation</i>) $[\lambda x^\uparrow(iobj\_de; emanar^\downarrow, x^\uparrow)] =$ {administração autoridade comissão conselho direcção estado governo ministro tribunal órgão} (<i>administration authority commission council direction state government minister tribunal organ</i>)
fase (<i>phase</i>)	$[\lambda x^\downarrow(dobj; x^\downarrow, fase^\uparrow)] =$ {abrançar compreender, constituir contemplar designar definir determinar enunciar estabelecer implicar integrar introduzir mencionar referir } (<i>include comprise constitute consider designate define determine state establish imply integrate introduce mention refer</i>) $[\lambda x^\downarrow(iobj\_em; x^\downarrow, fase^\uparrow)] =$ {consistir encontrar integrar prever} (<i>consist situate integrate foresee</i>) $[\lambda x^\uparrow(de; fase^\downarrow, x^\uparrow)] =$ {contrato execução exercício prazo procedimento processo trabalho} (<i>agreement execution practise term procedure process work</i>)

incomplete, this means that either the knowledge the parser had was incomplete, the entry was incorrect or some error was generated by the tagger.

The underlying parsing strategy is “chart parsing” [14]. At each stage a new parser can just check for possible faults and propose corrections by filling in the agenda, while the chart will be the one obtained at previous parsing stage. Let’s take an example: **a assembleia aprovou o ordenamento do território** (*the assembly approved the land use planning*). This would be analyzed as:

$$[[a\ assembleia]_{np}\ [aprovou]_{vp}]_S\ [o\ ordenamento]_{np}\ [de_{prep}\ [o\ territorio]_{np}]_{pp}]$$

By assuming that **aprovar** (*approve*) may require a noun phrase headed by **carreira** or **ordenamento** (see section 4.1) it is assumed that the information about verb subcategorization used earlier might be incorrect. Furthermore, by assuming that **ordenamento** can subcategorize prepositional phrase headed by preposition **de** followed by a noun phrase headed by noun **território** (*territory*); together with other possible corrections. The parser uses these hypotheses in its agenda and, in this case, completely parse the input. The correct parse has been obtained. For a whole text, the number of arcs spanning over the text would dramatically decrease and the syntactic parsing would be improved.

4.3 Evaluating Performance of Attachment Resolution

Table 3 shows the evaluation of the corrections proposed by the diagnosis parser. We evaluated the accuracy and recall of the proposed corrections on three types of candidate dependencies: NP-PP, VP-NP, and VP-PP. We call *accuracy* the proportion of corrections that actually correspond to true dependencies and, then, to correct attachments. *Recall* indicates the proportion of candidate dependencies that were actually corrected.

While accuracy reaches a very promising value (about 95%), recall merely reaches 20%. This is why subcategorization information is only available for those words that have a significant frequency throughout the corpus. Indeed, only those words occurring several times in the corpus can be used to learn clusters representing their semantic requirements. Recall directly relies on the size of the corpus used to learn clusters. We believe that increasing the corpus size will also increase the values reached by recall. Moreover, we have not taken

Table 3. Evaluation of Attachment Resolution on NP-PP, VP-NP, and VP-PP candidate dependences

Candidate Dependences	Accuracy (%)	Recall (%)
NP-PP	95, 53	30, 27
VP-NP	94, 44	19, 44
VP-PP	93, 87	10, 11
Total	94, 61	19, 94

into account that words (mainly verbs) may subcategorize simultaneously more than one argument. This also explains why the recall for NP-PP dependencies is significantly higher than that for verbal dependencies (i.e., VP-NP and VP-PP). Since prepositional complements of nominal phrases are more frequent than verbal complements, our clustering method let us learn more subcategorization information for nouns than for verbs.

As we do not propose long distance attachments, our method can not be compared with other standard corpus-based approaches to attachment resolution [13, 3]. In our approach, long distance attachments will be considered later, once all corrections for immediate dependencies have been proposed.

5 Conclusion

This paper has presented a particular unsupervised strategy to automatically learn context-based semantic classes used as restrictions on syntactic combinations. The strategy is mainly based on two linguistic assumptions: co-specification hypothesis, i.e., the two related expressions in a binary dependency impose semantic restrictions to each other, and contextual hypothesis, i.e., two syntactic contexts share the same semantic restrictions if they cooccur with the same words. Such a learning process allowed us to provide the dictionary entries with both syntactic and semantic subcategorization information. This information was used to improve parsing accuracy by checking the attachment hypotheses that the parser proposed at second stage of the syntactic analysis.

References

- [1] Roberto Basili, Maria Pazienza, and Paola Velardi. Hierarchical clustering of verbs. In *Workshop on Acquisition of Lexical Knowledge from Text*, pages 56–70, Ohio State University, USA, 1993. 408
- [2] Gilles Bisson, Claire Nédellec, and Dolores Canamero. Designing clustering methods for ontology building: The mo’k workbench. In *Internal rapport*, citeseer.nj.nec.com/316335.html, 2000. 409
- [3] Eric Brill and Philip Resnik. A rule-based approach to prepositional phrase attachment disambiguation. In *COLING*, 1994. 415
- [4] Ido Dagan, Lillian Lee, and Fernando Pereira. Similarity-based methods of word cooccurrence probabilities. *Machine Learning*, 43, 1998. 408
- [5] David Faure. *Conception de méthode d’apprentissage symbolique et automatique pour l’acquisition de cadres de sous-catégorisation de verbes et de connaissances sémantiques à partir de textes : le système ASIUM*. PhD thesis, Université Paris XI Orsay, Paris, France, 2000. 409, 410
- [6] David Faure and Claire Nédellec. Asium: Learning subcategorization frames and restrictions of selection. In *ECML98, Workshop on Text Mining*, 1998. 409, 410
- [7] Pablo Gamallo. *Construction conceptuelle d’expressions complexes: traitement de la combinaison nom-adjectif*. PhD thesis, Université Blaise Pascal, Clermont-Ferrand, France, 1998. 408

- [8] Pablo Gamallo, Alexandre Agustini, and Gabriel P. Lopes. Selection restrictions acquisition from corpora. In *10th Portuguese Conference on Artificial Intelligence (EPIA'01)*, Porto, Portugal, 2001. LNAI, Springer-Verlag. 410, 411
- [9] Pablo Gamallo, Caroline Gasperin, Alexandre Agustini, and Gabriel P. Lopes. Syntactic-based methods for measuring word similarity. In V. Mautner, R. Moucek, and K. Moucek, editors, *Text, Speech, and Discourse (TSD-2001)*, pages 116–125. Berlin:Springer Verlag, 2001. 409
- [10] Gregory Grefenstette. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, USA, 1994. 408
- [11] Gregory Grefenstette. Evaluation techniques for automatic semantic extraction: Comparing syntatic and window based approaches. In Branimir Boguraev and James Pustejovsky, editors, *Corpus processing for Lexical Acquisition*, pages 205–216. The MIT Press, 1995. 408
- [12] Ralph Grishman and John Sterling. Generalizing automatically generated selectional patterns. In *Proceedings of the 15th International on Computational Linguistics (COLING-94)*, 1994. 408
- [13] Donald Hindle and Mats Rooth. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120, 1993. 415
- [14] Martin Kay. *Algorith schemata and data structures in syntactic processing*. Technical report, XEROX PARK, Palo Alto, Ca., Report CSL-80-12, 1980. 414
- [15] Dekang Lin. Automatic retrieval and clustering of similar words. In *COLING-ACL'98*, Montreal, 1998. 408
- [16] J. Gabriel Pereira Lopes, Vitor Rocio, and João Balsa da Silva. Superando a incompletude da informação lexical (overcoming lack of lexical information, in portuguese). In Mota M. A. Marrafa P., editor, *Linguística Computacional: Investigação Fundamental e Aplicações*, pages 121–149. Lisboa: Edições Colibri, 1999. 413
- [17] Nuno Marques. *Uma Metodologia para a Modelação Estatística da Subcategorização Verbal*. PhD thesis, Universidade Nova de Lisboa, Lisboa, Portugal, 2000. 409, 411
- [18] Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of english words. In *Proceedings of the 30th Annual Meeting of the Association of Computational Linguistics*, pages 183–190, Columbus, Ohio, 1993. 408
- [19] James Pustejovsky. *The Generative Lexicon*. MIT Press, Cambridge, 1995. 408
- [20] Philip Resnik. Semantic similarity in taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999. 408
- [21] V. Rocio, E. de la Clergerie, and J.G.P. Lopes. Tabulation for multi-purpose partial parsing. *Journal of Grammars*, 4(1), 2001. 409, 411, 413
- [22] Luis Talavera and Javier Béjar. Integrating declarative knowledge in hierarchical clustering tasks. In *Intelligent Data Analysis*, pages 211–222, 1999. 411

Author Index

Adamatti, Diana F.	108	Hübner, Jomi Fred	118
Agustini, Alexandre	407	Ide, Jaime S.	366
Alvares, Luis Otavio	334	Iochpe, Cirano	291
Andrade, Adja Ferreira de	140	Jaques, Patrícia Augustin	140
Antunes, Luis	85	Jung, João Luiz	140
Artola, Fredy	302		
Baião, Fernanda	216	Kaestner, Celso A.A.	205, 280
Bazzan, Ana L. C.	108	Lopes, Gabriel P.	407
Bezerra, Byron	227	Lopes, Heitor S.	259
Bianchi, Reinaldo A. C.	195	Lorena, Luiz Antonio Nogueira	313
Bittencourt, Guilherme	175		
Boissier, Olivier	118	Maduro, Ralph Moreira	397
Bordini, Rafael H.	108	Mateos, Cristian	163
Brasil, Samuel M., Jr.	52	Mattoso, Marta	216
Brito, Carlos	41	Moniz, Luís	129
		Moraes, Márcia Cristina	97
Campo, Marcelo	163		
Camponogara, Eduardo	74	Neto, Joel Larocca	205
Carvalho, Ariadne M. B. R.	397		
Carvalho, Francisco de A. T. de ...	227, 237, 248	Oliveira, Alexandre César Muniz de	313
Chopra, Samir	31	Oliveira, Eugénio	152
Coelho, Helder	63, 85, 129		
Cordenonsi, Andre Zanki	334	Pappa, Gisele L.	280
Corruble, Vincent	237, 248	Parpinelli, Rafael S.	259
Costa, Anna H. R.	195	Perrussel, Laurent	11
Costa, Augusto Loureiro da	175	Rocha Costa, Antônio Carlos da	97
Cozman, Fabio G.	366, 376	Pozo, Aurora	324, 345
Cruz, Flavia	216		
		Queiroz, Sérgio R. de M.	248
David, Nuno	63		
Drummond, Isabela	386	Ramalho, Geber L. ...	227, 237, 248
		Rocha, José Carlos F. da	376
Eklund, Sven E.	185	Rodrigues, Ernesto	324
Engel, Paulo	291		
		Sandri, Sandra	386
Finger, Marcelo	21	Santos, Rafael Valle dos	302
Fontoura, Sérgio da	302	Sarmento, Luís	152
Freitas, Alex A.	205, 259, 280	Sichman, Jaime Simão	63, 118
		Silva, Carolina	291
Gamallo, Pablo	407	Silva, José Demisio Simões da ..	355
Garcia, Berilhes Borges	52	Simoni, Paulo Ouvera	355
Godo, Lluís	386	Spinosa, Eduardo	345

Teixeira, Ivan R.	237	Veloso, Sheila R. M.	1
Tomé, José A.B.	270	Vicari, Rosa Maria	140
Urbano, Paulo	129	Wassermann, Renata	21, 31
Vellasco, Marley	302	Zaverucha, Gerson	216
Veloso, Paulo A. S.	1	Zucker, Jean-Daniel	227
		Zunino, Alejandro	163