Lecture Notes in Computer Science 2602
Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Tokyo

Corrado Priami (Ed.)

Computational Methods in Systems Biology

First International Workshop, CMSB 2003
Rovereto, Italy, February 24-26, 2003
Proceedings

Springer

Volume Editor

Corrado Priami
Dipartimento di Informatica e Telecomunicazioni
Università di Trento
Via Sommarive, 14, 38050 Povo (TN), Italy
E-mail: priami@dit.unitn.it

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie;
detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): F.1.1-2, F.4.3, E.1, F.2.2, G.2, I.6, J.3

ISSN 0302-9743
ISBN 3-540-00605-2 Springer-Verlag Berlin Heidelberg New York

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

http://www.springer.de

Printed in Germany

Typesetting: Camera-ready by author, data conversion by DA-TeX Gerd Blumenstein
Printed on acid-free paper SPIN: 10872734 06/3142 5 4 3 2 1 0

Preface

Molecular biology has until now mainly focussed on individual molecules, on their properties as isolated entities or as complexes in very simple model systems. However, biological molecules in living systems participate in very complex networks, including regulatory networks for gene expression, intracellular metabolic networks and both intra- and intercellular communication networks. Such networks are involved in the maintenance (homeostasis) as well as the differentiation of cellular systems of which we have a very incomplete understanding.

Nevertheless, the progress in molecular biology has made possible the detailed description of the components that constitute living systems, notably genes and proteins. Large-scale genome sequencing means that we can (at least in principle) delineate all macromolecular components of a given cellular system, and microarray experiments as well as large-scale proteomics will soon give us large amounts of experimental data on gene regulation, molecular interactions and cellular networks. The challenge of the 21st century will be to understand how these individual components integrate into complex systems and the function and evolution of these systems, thus scaling up from molecular biology to systems biology. By combining experimental data with advanced formal theories from computer science, "the formal language for biological systems" to specify dynamic models of interacting molecular entities would be essential for: (i) understanding the normal behaviour of cellular processes, and how changes may affect the processes and cause disease – it may be possible to correlate genetic properties and symptoms in new and more efficient ways, based on an actual understanding of how various processes interact; (ii) providing predictability and flexibility to academic, pharmaceutical, biotechnology and medical researchers studying gene or protein functions. In particular, it may save time by reducing the number of experiments needed, if inadequate hypotheses can be excluded by computer simulation.

In response to the call for papers 39 were submitted to CMSB 2003. All the submitted papers were reviewed and the programme committee (listed below) selected 11 high-quality papers for publication in this volume. The care of the reviewers and of the programme committee members in reviewing the papers was surely valuable. A further 11 papers were selected only for presentation at the workshop in order to stimulate discussions (an abstract is included).

At the workshop Ehud Shapiro and Michael Stern gave two invited talks whose topics are described in papers included in this volume. The programme committee decided to accept for publication in this volume also some position papers that highlight the research trends in this new field of computational methods in systems biology. The reason is that because this is the first edition of a workshop in this fast-growing field a large view of potential topics of research was considered extremely important.

Rovereto, December 2002 Corrado Priami

Programme Committee of CMSB 2003

Corrado Priami (Chair), University of Trento (Italy),
Charles Auffray, CNRS, Villejuif (France),
Cosima Baldari, Università di Siena (Italy),
Alexander Bockmayr, Université Henri Poincaré (France),
Luca Cardelli, Microsoft Research Cambridge (UK),
Vincent Danos, Université Paris VII (France),
Pierpaolo Degano, Università di Pisa (Italy),
François Fages, INRIA, Rocquencourt (France),
Drabløs Finn, Norwegian University of Science and Technology, Trondheim (Norway),
Monika Heiner, Brandenburg University of Technology at Cottbus (Germany),
Ina Koch, University of Applied Sciences Berlin, (Germany),
John E. Ladbury, University College London (UK),
Patrick Lincoln, SRI (USA),
Satoru Miyano, University of Tokyo (Japan),
Gordon Plotkin, University of Edinburgh (UK),
Simon Plyte, Pharmacia Corporation (Italy),
Aviv Regev, Weizmann Institute of Science (Israel),
Magali Roux-Rouquié, BSMI Pasteur Institute (France),
Vincent Schachter, Hybrigenics Paris (France),
Masaru Tomita, Keio University (Japan),
Adelinde Uhrmacher, University of Rostock (Germany),
Alfonso Valencia, CNB-CSIC, Centro Nacional de Biotecnologia (Spain),
Olaf Wolkenhauer, UMIST, Manchester (UK)

Local Organizing Committee

Corrado Priami, Linda Brodo, Michela de Concini, Debora Schuch da Rosa Machado, and the University of Trento Events and Meetings Office.

List of Referees

F. Abascal, N. Chabrier, A. Cimatti, M. Curti, M.D. Devignes, S. Gnesi, J. Guijarro, K. Hafez, E. Klipp, C. Laneve, P. Lopez Romero, F. Luccio, R. Marangoni, M. Padron, M.C. Pinotti, R. Rizzi, S. Tini.

Acknowledgement

The workshop was sponsored and partially supported by the University of Trento, Comune di Rovereto, APT, and the EU project IST-32072-DEGAS.

Table of Contents

III Presentation Abstracts

Cells as Computation*

Amitai Regev and Ehud Shapiro

Department of Computer Science and Applied Mathematics
and Department of Biological Chemistry
Weizmann Institute of Science
Rehovot 76100, Israel

Although we are successfully consolidating our knowledge of the 'sequence' and 'structure' branches of molecular cell biology in an accessible manner, the mountains of knowledge about the function, activity and interaction of molecular systems in cells remain fragmented. Sequence and structure research use computers and computerized databases to share, compare, criticize and correct scientific knowledge, to reach a consensus quickly and effectively. Why can't the study of biomolecular systems make a similar computational leap? Both sequence and structure research have adopted good abstractions: 'DNA-as-string' (a mathematical string is a finite sequence of symbols) and 'protein-as-three-dimensional-labelled-graph', respectively. Biomolecular systems research has yet to find a similarly successful one.

The hallmark of scientific understanding is the reduction of a natural phenomenon to simpler units. Equally important understanding comes from finding the appropriate abstraction with which to distill an aspect of knowledge. An abstraction – a mapping from a real-world domain to a mathematical domain – highlights some essential properties while ignoring other, complicating, ones. For example, classical genetic analysis uses the 'gene-as-hereditary-unit' abstraction, ignoring the biochemical properties of genes as DNA sequences. A good scientific abstraction has four properties: it is relevant, capturing an essential property of the phenomenon; computable, bringing to bear computational knowledge about the mathematical representation; understandable, offering a conceptual framework for thinking about the scientific domain; and extensible, allowing the capture of additional real properties in the same mathematical framework.

For example, the DNA-as-string abstraction is relevant in capturing the primary sequence of nucleotides without including higher- and lower-order biochemical properties; it allows the application of a battery of string algorithms, including probabilistic analysis using hidden Markov models, as well as enabling the practical development of databases and common repositories; it is understandable, in that a string over the alphabet A, T, C, G is a universal format for discussing and conveying genetic information; and extensible, enabling, for example, the addition of a fifth symbol denoting methylated cytosine.

We believe that computer science can provide the much-needed abstraction for biomolecular systems. Advanced computer science concepts are being used to investigate the 'molecule-as-computation' abstraction, in which a system of

* Paper reproduced with permission from NATURE – Vol. 419 – 26 September 2002 – p. 343.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 1–3, 2003.

interacting molecular entities is described and modelled by a system of interacting computational entities. Abstract computer languages, such as Petrinets, Statecharts and the Pi-calculus, were developed for the specification and study of systems of interacting computations, yet are now being used to represent biomolecular systems, including regulatory, metabolic and signalling pathways, as well as multicellular processes such as immune responses. These languages enable simulation of the behaviour of biomolecular systems, as well as development of knowledge bases supporting qualitative and quantitative reasoning on these systems' properties.

Processes, the basic interacting computational entities of these languages, have an internal state and interaction capabilities. Process behaviour is governed by reaction rules specifying the response to an input message based on its content and the state of the process. The response can include state change, a change in interaction capabilities, and/or sending messages. Complex entities are described hierarchically – for example, if a and b are abstractions of two molecular domains of a single molecule, then (a parallel b) is an abstraction of the corresponding twodomain molecule. Similarly, if a and b are abstractions of the two possible behaviours of a molecule in one of two conformational states, depending on the ligand it binds, then (a choice b) is an abstraction of the molecule, with the choice between a and b determined by its interaction with a ligand process.

Using this abstraction opens up new possibilities for understanding molecular systems. For example, computer science distinguishes between two levels to describe a system's behaviour: implementation (how the system is built, say the wires in a circuit) and specification (what the system does, say an 'AND' logic gate). Once biological behaviour is abstracted as computational behaviour, implementation can be related to a real biological system, for example the detailed molecular machinery of a circadian clock, and the corresponding specification to its biological function, such as a 'black-box' abstract oscillator. Ascribing a biological function to a biomolecular system is thus no longer an informal process but an objective measure of the semantic equivalence between low-level and high-level computational descriptions. Equivalence between related implementations in different organisms can also be a measure of the behavioural similarity of entire systems, complementary to sequence and structure similarity.

Computer and biomolecular systems both start from a small set of elementary components from which, layer by layer, more complex entities are constructed with evermore sophisticated functions. Computers are networked to perform larger and larger computations; cells form multicellular organisms. All existing computers have an essentially similar core design and basic functions, but address a wide range of tasks. Similarly, all cells have a similar core design, yet can survive in radically different environments or fulfil widely differing functions. Of course, biomolecular systems exist independently of our awareness or understanding of them, whereas computer systems exist because we understand, design and build them. Nevertheless, the abstractions, tools and methods used to specify and study computer systems should illuminate our accumulated knowledge about biomolecular systems.

Further Reading

[1] www.wisdom.weizmann.ac.il/∼aviv
[2] Milner, R. *Communicating and Mobile Systems: The Pi-calculus.* (Cambridge Univ. Press, 2000).
[3] Fontana, W. and Buss, L. W. in *Boundaries and Barriers* (eds Casti, J. and Karlqvist, A.) 56–116 (Addison-Wesley, New York).

Formal Modeling of C. elegans Development: A Scenario-Based Approach

Na'aman Kam[1], David Harel[1], Hillel Kugler[1], Rami Marelly[1], Amir Pnueli[1], E. Jane Albert Hubbard[2], and Michael J. Stern[3]

[1] Dept. of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot 76100, Israel
{kam,dharel,kugler,rami,amir}@wisdom.weizmann.ac.il
[2] Dept. of Biology, New York University, New York, NY
jane.hubbard@nyu.edu
[3] Dept. of Genetics, Yale University School of Medicine, New Haven, CT
michael.stern@yale.edu
http://www.wisdom.weizmann.ac.il/~kam/CelegansModel/CelegansModel.htm

Abstract. We present preliminary results of a new approach to the formal modeling of biological phenomena. The approach stems from the conceptual compatibility of the methods and logic of data collection and analysis in the field of developmental genetics with the languages, methods and tools of scenario-based reactive system design. In particular, we use the recently developed methodology consisting of the language of *live sequence charts* with the *play-in/play-out* process, to model the well-characterized process of cell fate acquisition during *C. elegans* vulval development.

1 Introduction

Our understanding of biology has become sufficiently complex that it is increasingly difficult to integrate all the relevant facts using abstract reasoning alone. This is exacerbated by current high throughput technologies that spew data at ever-increasing rates. While bioinformatic approaches to handling this mass of data have generated databases that ease the storage and accessibility of the data, rigorous modeling approaches are necessary to integrate these data into useable models that can exploit and analyze the available information. There are many current efforts aimed at biological modeling, and it is likely that different approaches will be appropriate for various types of biological information and for various research objectives [1, 2]. Here, we present a novel approach to modeling biological phenomena. It utilizes in a direct and powerful way the mechanisms by which raw biological data are amassed, and smoothly captures that data within tools designed by computer scientists for the design and analysis of complex reactive systems.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 4-20, 2003.

A considerable quantity of biological data is collected and reported in a form that can be called “condition-result” data. The gathering is usually carried out by initializing an experiment that is triggered by a certain set of circumstances (conditions), following which an observation is made and the results recorded. The condition is most often a perturbation, such as mutating genes or exposing cells to an altered environment. For example, genetic data often first emerge as phenotypic assessments (anatomical or behavioral outputs) that are compared between a mutant background and a defined "wild-type." Another example includes observations of the effects of anatomical manipulations (e.g. cell destruction or tissue transplantation) on the behavior of the remaining structures. These types of experiments test specific hypotheses about the nature of the system that is perturbed. Many inferences about how biological systems function have been made from such experimental results, and our consequent understanding based on these logical inferences is becoming increasingly, even profoundly, complex.

One feature of these types of experiments is that they do not necessitate an understanding of the particular molecular mechanisms underlying the events. For example, much information can be ascertained about a gene's function by observing the consequences of loss of that function before the biochemical nature or activity of the gene product is known. Moreover, even when the biochemical activity is known, the functional significance of that activity in the context of a biological system is often deduced at the level of phenotypic output. Naturally, with knowledge of molecular mechanisms, increasingly sophisticated inferences can be made and more detailed hypotheses tested, but the outputs, be they a certain cell fate acquisition, changes in gene expression patterns, etc., are often recorded and analyzed at the level of a phenotypic result. Thus, a large proportion of biological data is reported as stories, or "scenarios," that document the results of experiments conducted under specific conditions. The challenge of modeling these aspects of biology is to be able to translate such "condition-result" phenomena from the "scenario"-based natural language format into a meaningful and rigorous mathematical language. Such a translation process will allow these data to be integrated more comprehensively by the application of high-level computer-assisted analysis. In order for it to be useful, the model must be rigorous and formal, and thus amenable to verification and testing.

We have found that modeling methodologies originating in computer science and software engineering, and created for the purpose of designing complex *reactive systems*, are conceptually well suited to model this type of condition-result biological data. Reactive systems are those whose complexity stems not necessarily from complicated computation but from complicated reactivity over time. They are most often highly concurrent and time-intensive, and exhibit hybrid behavior that is predominantly discrete in nature but has continuous aspects as well. The structure of a reactive system consists of many interacting components, in which control of the behavior of the system is highly distributed amongst the components. Very often the structure itself is dynamic, with its components being repeatedly created and destroyed during the system’s life span.

The most widely used frameworks for developing models of such systems feature *visual formalisms*, which are both graphically intuitive and mathematically rigorous. These are supported by powerful tools that enable full model executability and analy-

sis, and are linkable to graphical user interfaces (GUIs) of the system. This enables realistic simulation prior to actual implementation. At present, such languages and tools --- often based on the *object-oriented* paradigm --- are being strengthened by verification modules, making it possible not only to execute and simulate the system models (test and observe) but also to verify dynamic properties thereof (prove).

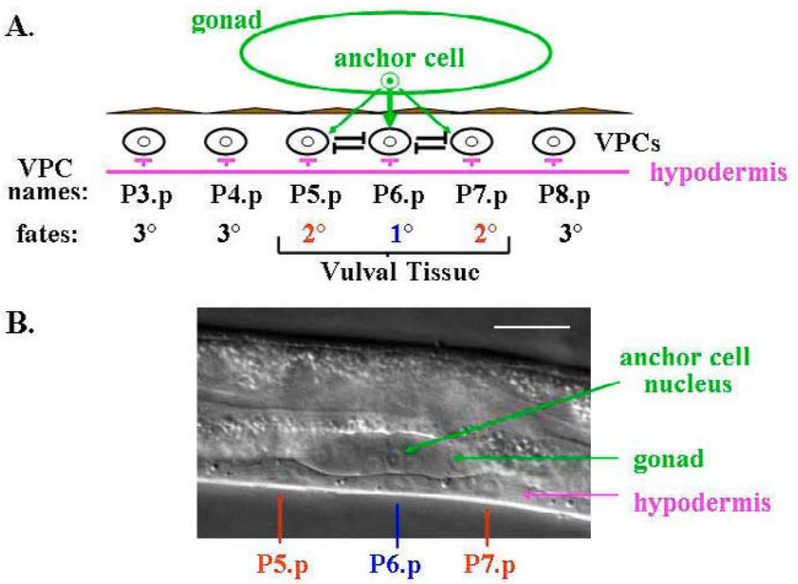


Fig. 1. Vulval cell fate determination. (A) Schematic representation of cellular interactions that determine vulval fates. Signals (arrows and T-bars) are color-coded based on their source. The anchor cell promotes vulval fates. VPC-VPC interactions inhibit adjacent cells from acquiring a 1º fate (and instead promote a 2º fate). The ventral hypodermis inhibits vulval (1º and 2º) fates. (B) Differential interference contrast (DIC) photomicrograph of the central body region of a live *C. elegans* L3-stage worm. White bar ~ 15µm

A central premise of this paper is that many kinds of biological systems exhibit characteristics that are remarkably similar to those of reactive systems. The similarities apply to many different levels of biological analysis, including those dealing with molecular, cellular, organ-based, whole organism, or even population biology phenomena. Once viewed in this light, the dramatic concurrency of events, the chain-reactions, the time-dependent patterns, and the event-driven discrete nature of their behaviors, are readily apparent. Consequently, we believe that biological systems can be productively modeled as reactive systems, using languages and tools developed for the construction of man-made systems.

In previous work on T cell activation [3] and T cell behavior in the thymus [4], the feasibility of this thesis was addressed on a small scale, and the results have been very encouraging. In particular, that work demonstrated the adequacy of object-oriented analysis to modeling biological systems, and showed the applicability of the visual formalism of statecharts for representing their behavior.

In our current work, we not only begin to tackle a more complex system, but also incorporate two levels of data into our model. One of these levels, shared also by the models from the previous work, represents the 'rules' of the behavior of the system. These rules are based on abstractions and inferences from various sets of primary data. The second level of data being incorporated into our model, is new, and it includes the "observations" that comprise the primary data itself. The set of observations utilized is a crucial component of the model, allowing both execution and verification, including consistency checks between the observations and the inferred rules. To accomplish this, we also depart in a significant way from the intra-object, state-based, statechart approach used in the previous work, and instead use a more recently-developed inter-object, scenario-based approach to reactive system specification. The language we use to formalize the data is called *live sequence charts* (LSCs) [5], and capturing the data and analyzing it is carried out using the *play-in/play-out* methodology, supported by the *Play-Engine* tool [6,7].

1.1 The Biological System

Our current effort is focused on a means to formalize and analyze primary data such that the consistency of inferences made from these data can be tested as part of the model. We have chosen the development of the nematode *Caenorhabditis elegans* as a subject since this organism is extremely well defined at the anatomical, genetic, and molecular level. Specifically, the entire cell lineage of *C. elegans* has been traced, many genetic mutants have been described, and the entire genome is sequenced [8, 9]. Moreover, virtually all of the researchers working on this organism use conceptually similar methodologies and logic in their genetic experiments, and use a uniform substrate wild-type strain (N2). These circumstances make feasible computer-assisted integration of the primary data. Finally, genome-wide efforts to define the function, expression levels, expression patterns and interactions of all genes are underway [10–13].

As a specific test-case, we have begun to model *C. elegans* vulval development (see [14] for a recent review). The vulva is a structure through which eggs are laid. This structure derives from three cells within a set of six cells with an equivalent set of multiple developmental potentials (Fig. 1). These six cells are named P3.p, P4.p, P5.p, P6.p, P7.p and P8.p (collectively termed P(3-8).p). Due to their potential to participate in the formation of the vulva, they are also known as the vulval precursor cells (VPCs). Each has the potential to acquire either a non-vulval fate (a 3º fate) or one of two vulval cell fates (a 1º or a 2º cell fate; see Fig. 1). During normal development, after a series of cell divisions in a characteristic pattern, a vulva consisting of 22 nuclei is formed. Vulval development was one of the first areas to which considerable effort was applied to achieve a molecular understanding of how cells acquire their particular fates. This system, though limited in cell number, is quite complex and ultimately integrates at least four different molecular signaling pathways in three different interacting tissue types. Cell fate acquisition in development also occurs in the context of cell cycle control and the general global controls on the growth and development of the organism. Hence, vulval development indeed represents a rather complex reactive system.

Here, we describe our progress applying the visual formalism of LSCs and the play-in/play-out methodology to the representation of biological scenarios from one particular study, the data published in a paper by Sternberg and Horvitz in 1986 [15]. LSCs appear to be highly accessible to biologists, while retaining mathematical rigor. Designed specifically to capture scenario-based behavior, the structure of LSCs fits extremely well into our framework of condition-result data. In this paper, data are reported regarding a series of experiments in which cells are destroyed using a laser microbeam (ablation). Since signals among the VPCs and between the VPCs and adjacent tissues cooperate to determine the fates of these cells, these experiments test the nature of these interactions and the relative potential of the VPCs to adopt vulval (versus non-vulval) fates and, among those that adopt a vulval fate, the specific type of vulval fate. These data are used to infer specific properties of the events that occur in the wild-type situation and to generate a "model" for how the unperturbed system works based on the behavior of the perturbed system. We present a subset of this test-case below with special emphasis on the LSC language and the play-in/play-out methodology. In addition, we present our solutions to particular challenges posed by the nature of biological data itself and its formalization from the natural language used by *C. elegans* biologists. In representing even a small and simple set of experimental results, we have addressed a number of issues that provide evidence for the flexibility and potential of this modeling approach.

1.2 The Modeling Methodology

We are adopting an inter-object, scenario-based modeling approach, using the language of *live sequence charts* (LSCs) [5] and the *play-in/play-out* methodology [6, 7], both supported by the *Play-Engine* modeling tool [6,7]. The decision to take this approach, rather than the statechart-based one, emerged from the consideration of how to best represent the C. elegans data formally, and how to best carry out the formalization process.

LSCs constitute a visual formalism for specifying sequences of events and message passing activity between objects. They can specify scenarios of behavior that cut across object boundaries and exhibit a variety of modalities, such as scenarios that can occur, ones that must occur, ones that may not occur (called *anti-scenarios*), ones that must follow others, ones that overlap with others, and more. Technically, there are two types of LSCs, universal and existential. The elements of LSCs (events, messages, guarding conditions, etc.) can be either mandatory (called *hot* in LSC terminology) or provisional (called *cold*). Universal charts are the more important ones for modeling, and comprise a *prechart* and a *main* chart, the former triggering the execution of the latter. Thus, a universal LSC states that whenever the scenario in the prechart occurs (e.g., the user has flipped a switch), the scenario in the main chart must follow it (e.g., the light goes on). Thus, the relation between the prechart and the chart body can be viewed as a dynamic condition-result: if and when the former occurs, the system is obligated to satisfy the latter.

Play-in/play-out is a recently developed process for modeling in LSCs, with which one can conveniently capture inter-object scenario-based behavior, execute it, and simulate the modeled system in full. The play-in part of the method enables people

who are unfamiliar with LSCs to specify system behavior using a high level, intuitive and user-friendly mechanism. The process asks that the user first build the graphical user interface (GUI) of the system, with no behavior built into it. The user then 'plays' the GUI by clicking the graphical control elements (in electronic systems these might be buttons, knobs, and so on), in an intuitive manner, in this way giving the engine sequences of events and actions, and teaching it how the system should respond to them. As this is being done, the Play-Engine continuously constructs the corresponding LSCs automatically.

While play-in is the analogue of writing programs, play-out is the analogue of running them. Here the user simply plays the GUI as he/she would have done when executing the real system, also by clicking buttons and rotating knobs, and so on, but limiting him/herself to end-user and external environment actions. As this is going on, the Play-Engine interacts with the GUI and uses it to reflect the system state at any given moment. The scenarios played in using any number of LSCs are all taken into account during play-out, so that the user gets the full effect of the system with all its modeled behaviors operating correctly in tandem. All specified ramifications entailed by an occurring event or action will immediately be carried out by the engine automatically, regardless of where in the LSCs it was originally specified. Also, any violations of constraints (e.g., playing out an anti-scenario) or contradictions between scenarios, will be detected if attempted. This kind of sweeping integration of the specified condition-result style behavior is most fitting for biological systems, where it can often be very difficult to connect the many pieces of behavioral data that are continuously discovered or refined.

2 Results

2.1 The GUI of Vulval Fate Determination

A critical aspect of the GUI is that it can be designed by the user to reflect his/her view of the system. In our case, the GUI (Fig. 2) is a simplified representation of the actual anatomical situation under study (Fig. 1). The six VPCs and the interacting adjacent tissues, that include the gonad and the ventral hypodermis, are represented. This GUI represents the developmental decisions that occur during a discrete window of time, and we developed it the way we did in order for it to best capture that particular stage in the organism's development It can be expanded to capture more parts and stages in the development process, and also to include inputs from other GUIs (possibly constructed by other people working on a growing distributed model) that represent other connected developmental vignettes. The beauty of this kind of GUI is that it can be made to directly reflect the way biologists represent their system's anatomy, and the particular GUI in Figure 2 is intuitive for anyone working on vulval development. However, the more important aspect of the GUI is that it allows the model to “come alive” in the context of the play-in/play-out approach.

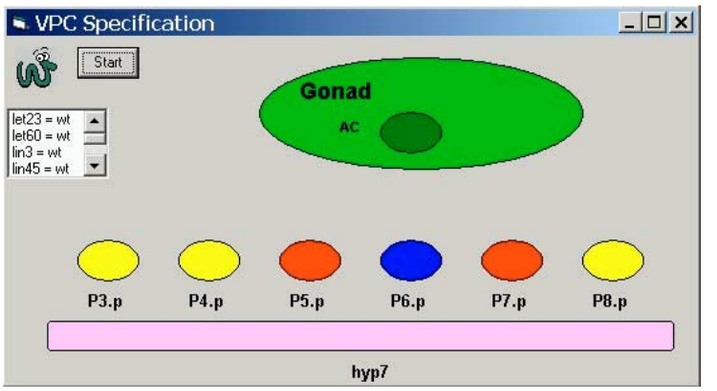


Fig. 2. The GUI

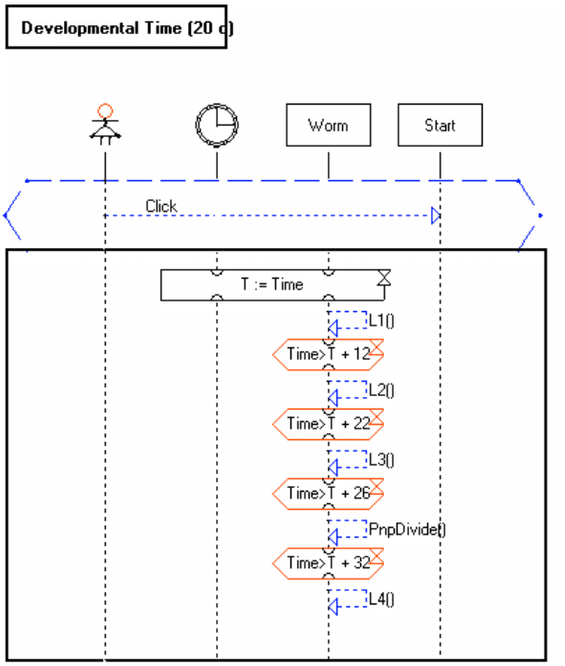


Fig. 3. An LSC depicting developmental time

2.2 Case Study: Sternberg and Horvitz, 1986

We present several examples of scenarios that were described in [15], which we translated into LSCs. These examples demonstrate how data is entered into the model and give context to the progress we have made regarding broad conceptual aspects of

the formalization of biological data. Concepts already incorporated into the model include the representation of developmental time, symbolic instances, default assumptions, and non-deterministic behavior. Examples of LSCs are included that represent the formalization of both primary observations and inferred rules.

Developmental Time. In the current model, developmental time drives the behavior of the whole system. (See [16] for the way the issue of time is treated in LSCs and the Play-Engine.) Thus, we begin by presenting the corresponding LSC (Fig. 3). We decided to concentrate on post-embryonic development (after the embryo has hatched from the egg), but earlier developmental stages can easily be incorporated as well. Post-embryonic development is divided into four larval stages (denoted L1 to L4), each of which end in a molt. The time units represent hours from hatching (the beginning of L1). This LSC is activated when the user clicks the start button on the GUI. The time at which this operation was performed is stored in a variable called T, and all other events in this LSC refer to this time point. Thus, for example, the L1 phase lasts 12 hours, the L2 phase lasts 10 hours, and the Pn.p cells divide 26 hours after hatching, which is 4 hours after entry into L3.

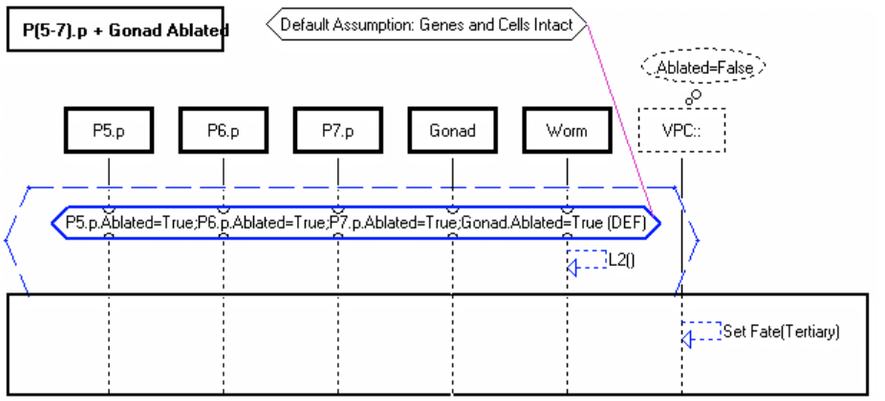


Fig. 4. A scenario involving the ablation of P(5-7).p and the gonad

Representing the Experimental Setup in the Prechart. The experiment represented by the LSC in Fig. 4 was devised to establish the ground-state fate of the P(3,4,8).p cells, that is to see what cell fate these cells would acquire if they were not influenced by any known interactions from adjacent cells. At the time of this publication, the influence of the hypodermis on vulval fate specification was not known (it is also irrelevant under these specific conditions). The cells known to influence VPC cell fate were removed by laser ablation, including the three VPCs that are normally induced to form the vulva (P5.p, P6.p and P7.p). The fates of the remaining VPCs were observed and recorded. The results of this set of experiments were reported in the following statement:

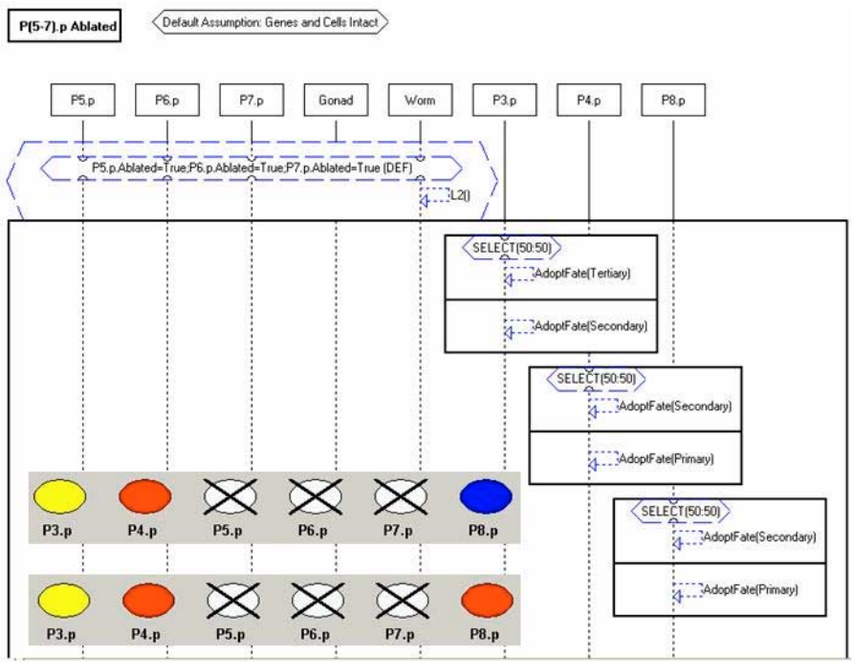


Fig. 5. Representing non-deterministic events. The results of two consecutive executions of this LSC are shown. It can be seen that P8.p adopted a different fate on each of these executions

> *"We ablated P(5-7).p and the gonad in five L1 hermaphrodites; in these animals P(3,4,8).p each adopted a tertiary fate" [15].*

In this experiment, the cell ablations took place during the first larval stage (L1). This fact is stated in the prechart by placing the condition before the L2 event: the prechart will be satisfied only if by the time the L2 event occurs (determined in the developmental time LSC) the specified cells will already be ablated.

Experimental setups, such as the above one, implicitly assume that besides the reported perturbations all other elements of the system remained intact. We represent such implicit assumptions in a set of *default assumptions*, which includes all the properties of the system for which we would like to set some default value (e.g., genes are not mutated, or "wild type"). If a given condition within the LSC includes assigning a non-default value to some property, this requirement will override the default assumption. Thus, the condition that appears in the pre-chart of Fig. 4 corresponds to a situation in which P(5-7).p and the gonad were ablated *while all other genes and cells remained intact*. During a play-in session, a condition can be bound to a set of default assumptions by clicking a checkbox within the condition window. Within the LSC, this action is reflected by the DEF (default) tag that appears at the end of the condition, as well as by a line that connects the condition with the set of default assumptions that appears at the top of the LSC (the line appears only when the mouse cursor is placed over the condition).

Symbolic Instances. The LSC in Fig. 4 illustrates the concept of symbolic instances. (See [17] for how symbolic instances are dealt with in LSCs and the Play-Engine.) In this experiment, the behavior of all remaining VPCs is similar: they all adopt tertiary fates. This is expressed in the LSC by making a general statement that applies to a symbolic instance of the VPC class. This depiction indicates that during execution, all instances of the VPC class that were not ablated will be bound to this LSC and execute a tertiary fate.

Non-deterministic Behavior. Part of the experiment designed to assess the ground state of the VPCs involved ablating the cells that normally adopt vulval fates [P(5-7).p] but leaving the gonad intact. Three animals were observed after performing these ablation conditions; the fates of the VPCs are shown below (reported in Table 3, line 3, [15]):

Pn.p cells:	P3.p	P.4p	P5.p	P6.p	P7.p	P8.p
Fates:	2º or 3º	1º or 2º	ablated	ablated	ablated	1º or 2º

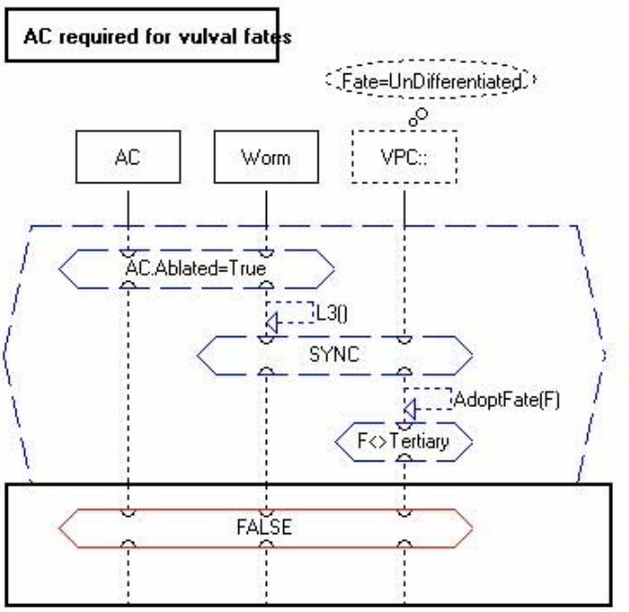


Fig. 6. An anti-scenario

As is often the case in biological experiments, the results of the conditions of this experiment were not deterministic: P3.p adopted either a 2º or a 3º fate, while P4.p and P8.p adopted either a 1º or a 2º fate. Each of these fates is recognizable by a specific pattern of cell divisions and cellular morphologies. Non-determinism is represented in the LSC that depicts this experiment (Fig. 5) using selection boxes within the

main chart. This representation enables the model to execute non-deterministic behavior of objects using a selected list of outcomes at prescribed frequencies.

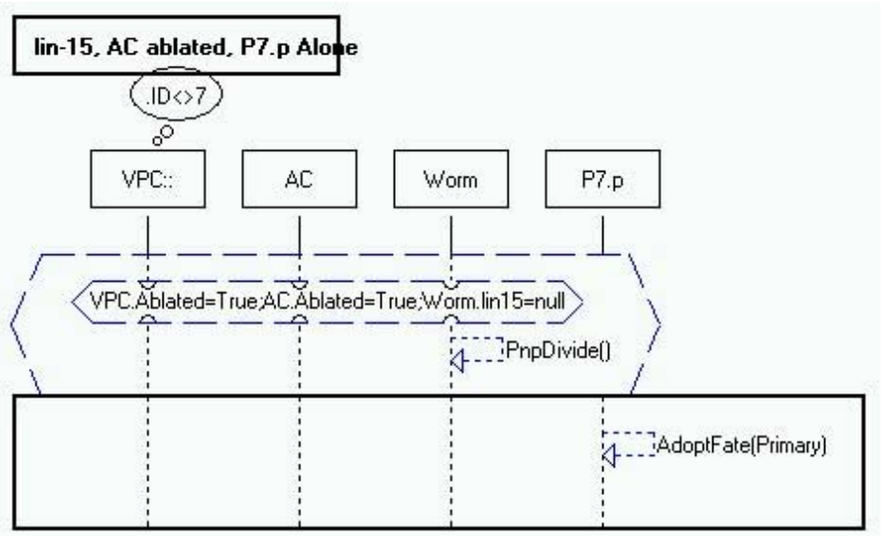


Fig. 7. A VPC can adopt a vulval fate in the absence of the anchor cell

Representing General Rules as Anti-scenarios. In addition to the input of raw data from specific experiments, general rules of system behavior can also be included as LSCs. Some rules can be represented most easily by what are known as *anti-scenarios*. For example, to represent the statement:

> *"The anchor cell is required for Pn.p cells to adopt vulval (1° or 2°) as opposed to non-vulval (3°) fates" [15],*

an LSC representing the following anti-scenario can be used:

"it cannot be the case that a VPC will adopt a vulval fate in the absence of the anchor cell" (Fig. 6).

The ability to represent behavior using "must" and "must not" statements lends greater potential and power to modeling efforts.

System Analysis: The Play-Out Mechanism. The play-out mechanism can be used to test the system in various ways:

1. *Detecting inconsistencies among LSCs.* In its anti-scenario guise, an LSC representing, for example, a scenario in which a VPC adopts a vulval fate in the absence of the anchor cell would cause the Play-Engine to announce that a 'hot' condition was violated. Certain mutations can obviate the requirement of the anchor cell for vulval cell fates. For example, if the hypodermal inhibitory mechanism (see Fig. 1) is compromised by mutation, vulval fates can

occur even in the absence of the gonad (e.g. see [19]). Executing a play-out session in which an 'experiment' of this type was performed resulted in a violation of the 'old' rule. Thus, this modeling approach can help integrate new results into the framework of existing data, pointing out inconsistencies that might have been ignored by the experimentalist.

2. *Predictability*. The play-engine can juxtapose LSCs generated either from specific information results or from general biological rules, which are inferred from many data sets. This juxtaposition has the potential to simulate novel scenarios and to highlight behaviors that were not previously observed. One such example was observed over our model while playing-out a scenario that involved a combination of two mutations ('double mutant'), one in the dig-1 gene and the other in the lin-12 gene. In dig-1 mutants the gonad is shifted anteriorly, and in lin-12(0) mutants the three VPCs P(5-7).p were observed to adopt a 1° fate. An actual experiment that detects the effect of combining these two mutations together has not been done yet. However, based on the LSCs that represent the observed phenotypes for each individual mutant, together with a couple of LSC that represent deduced rules regarding signaling mechanisms involved in vulval induction, the play-engine predicted that such an experiment would result in P(4-6).p adopting a 1° fate. Such a play-out session illustrates that the play-engine can execute scenarios that were not played-in explicitly. (For further explanations, including a demo of this execution, see [19].)
3. *Query the system for scenarios that satisfy a given behavior*. Play-out has been extended with a powerful new module, called *smart play-out* [18], which utilizes tools from program verification in order to analyze an LSC model in much richer ways than just executing it. Many of its uses come from asking the system to find a way to do something on its own, or to ask it "Is this possible?" kinds of questions. For example, we can ask the system to automatically figure out if there is some possible way of satisfying a particular scenario, and to then run the resulting discivered sequence on its own. An example for such a test is depicted in Figure 7, in which an existential LSC is used to query the system for scenarios that result in P7.p adopting a 3° fate. (For further explanations, including a demo of applying this test to the model, see [19].)

Thus, play-out, with its smart play-out enrichment, makes it possible to detect, or predict, the outcome of combinations of conditions not previously tested, and to query the system for many different kinds of desired (or undesired) outcomes.

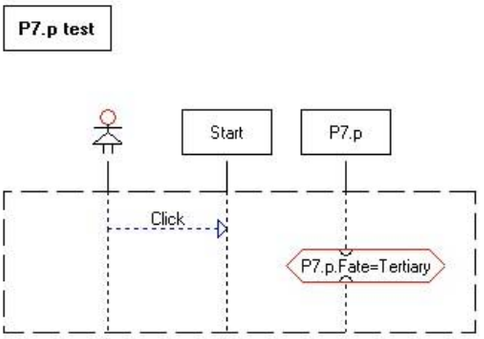


Fig. 8. An existential chart is used to test whether there is any scenario in which P7.p adopts a primary fate

3 Discussion

3.1 Strengths of Our Approach

In previous sections, we illustrated the capability of our modeling approach to facilitate the transformation of biological data into formal computer-analyzable statements. In particular, our modeling approach provides solutions for the following challenges involved in this formalization process:

Incorporating Raw Biological Data. One of the unique features of our approach is the incorporation of primary data, rather than modeling the general rules that are derived or inferred from the actual data. One challenge to modeling primary data is the integration of information obtained in different ways and using different approaches. The assessment of results can vary with the individual experimentalist and changes over time as new information about the system and technologies to evaluate it become available. The variety of data collection methods can be integrated by incorporating LSCs that link the different types of data together. We can then use the model to validate equivalence statements against the data, thus enhancing our understanding of the biological system.

Importantly, primary data, once entered as LSCs, can be used even when the general rules of behavior are modified with time. Primary data scenarios can be derived from many different laboratories and many different types of experimentation, and can be used to detect conflicts within the primary data, highlighting important aspects to be later tested more rigorously.

Rules as a Second Level. A computer-assisted approach to a formalized list of primary data is useless unless it can be tested for overall consistency and logical conflict, in ways that are better than can be done by human reasoning alone. This possibility is also built into our approach. A logical system is routinely used by experimentalists to infer abstractions about the real system from condition-result types of data. In our

model, these abstractions can be formalized as 'rules'. For example, they can be entered in the form of anti-scenarios (as shown above). Thus, it may be possible to formalize a system that is already in use in a non-formal way so that the rigorous tools of computer-based verification can be brought to bear on the data. Moreover, having both primary data and associated 'rules', allows novel scenarios to be obtained from the input data, as illustrated above.

Non-deterministic Events. One of the hallmarks of biological data is non-determinism. While some non-determinism likely derives from the paucity of our knowledge of the important details of the systems of interest, the very nature of biological systems contains some inherent non-determinism. For example, stochastic events, threshold phenomena and feedback systems, all operate at the molecular and cellular levels and it is crucial that methods and tools for their modeling support explicit choice-based, or probability-based, branching of behavior.

3.2 What Can We Get Out of It?

A Behavioral Database. At the most basic level, a model of the type we propose forms a behavioral database that can be used for retrieving dynamic data. Existing biological databases can be used to retrieve the sequence of a given gene, or even to find its homologues, but there is no current mechanism for asking questions such as 'what will happen if gene X will be mutated together with gene Y', or 'what will be the phenotypic outcome of ablating cell A over a specific genetic background'.

Detecting Inconsistencies. New inferences that are made can be checked for consistency against existing observations and rules. As illustrated in section 2.2 above, 'old' rules that are contradicted by new experiments can be detected as well.

Predictions. The play-out approach offers the additional possibility of asking the system for the predicted result of an experiment that has not yet been performed. Such predictions can then be tested in the lab. If the model predicts an outcome different from the actual outcome, there are likely factors that have not been included in the model that are relevant to the experimental system. Identifying such putative factors can then lead to new experiments that will aim at filling in these gaps. Alternatively, such a mismatch between prediction and experiment can result from improper rules or interpretations that currently constitute the model. Thus, the kind of model analysis we propose can assist in improving, correcting and sharpening our understanding of the biological system.

Explaining Surprising Results. The smart play-out mechanism can be used to answer questions such as: 'Is there any scenario in which a given behavior will be observed?' Such tests can be applied to explaining surprising experimental outcomes that were observed in the lab ('based on the current data, is there any scenario that can produce this surprising result?'), or to test our understanding of the system ('is there indeed no scenario, in which, although the product of gene X was eliminated, a given tissue T will still develop normally?').

3.3 Current Status and Future Directions

To date, we have formalized as LSCs only a small set of data pertaining to VPC specification. These have served to highlight some of the critical issues that need to be addressed in modeling this system in its entirety. Several of these issues have already been addressed, as described in this paper. In addition to the conceptual advances this small data set has prompted, these LSCs reveal much of the promise and feasibility of applying LSCs and play-in/play-out to the modeling of biology. One of the strengths of the entire approach is the ability to execute a model even with incomplete data sets. As we continue to fill out our test-case model, including the integration of different signal-transduction inputs into the determination of vulval fates, we are also considering the expansion of the methodology in several areas:

Expanding the Experimental Repertoire. Although our current efforts are concentrated on condition-result data from genetic and anatomical manipulations, there is no *a priori* reason why the same methods should not be applied to other types of data, representing other levels of biological inquiry. These include biochemical data (such as signal transduction pathways, protein-protein interactions, etc.) and gene expression data (microarray data, anatomical expression pattern information, etc.).

Distributed Play-In/Play-Out. It is already possible to connect separate GUIs and Play-Engines to each other. This will enable multiple labs to participate in the modeling effort. Each lab can design its own GUI to represent its sub-system of interest, and then play in the relevant scenarios. Connecting Play-Engines to each other can facilitate a distributed play-out mechanism, in which an event that occurs in an LSC that is being executed on one computer will activate and LSC that belongs to a specification running on another computer. The Play-Engine development team is also working on tolls to connect the Play-Engine to other environments as well, e.g., tools that enable statechart-based modeling, such as Rhapsody, and ones that deal with the continuous aspects of systems. These too will significantly expand the possibilities of modeling biological systems.

Another possibility is to connect to the C. elegans field's database Wormbase (or its equivalent in other systems), so that just as we can obtain the sequence and homologues, etc., for each gene, we would be able to retrieve via Wormbase the LSCs in which it participates.

The Long-Term Goal. Finally, we truly believe that this research effort could turn out being the first step in a far, far more ambitious project. Namely, to construct a full 4-dimensional model of a multi-cellular animal, which is true to all known facts about it, and which is easily extendable as new facts are discovered (see [21] for a more detailed discussion of this).

References

[1] Bower, J.M., Bolouri, H. (eds.): Computational modeling of genetic and biochemical networks. The MIT Press, Cambridge, MA (2001)

[2] Wilkins, A.S. (ed.): Modelling complex biological systems: a special issue. BioEssays 24(12) Wiley Periodicals, Inc., Hoboken, NJ (2002)

[3] Kam, N., Cohen, I.R., Harel, D., The Immune System as a Reactive System: Modeling T Cell Activation with Statecharts. To appear in Bull. Math. Bio. An extended abstract of this paper appeared in Proc. Visual Languages and Formal Methods (VLFM01), part of IEEE Symposia on Human-Centric Computing Languages and Environments (HCC01), pp. 15-22 (2001)

[4] Efroni, S., Harel, D. and Cohen I.R., Reactive Animation, submitted (2002)

[5] Damm, W. and Harel, D., LSCs: Breathing Life into Message Sequence Charts, Formal Methods in System Design 19:1 (2001). (Preliminary version in Proc. 3rd IFIP Int. Conf. on Formal Methods for Open Object-Based Distributed Systems (FMOODS'99), (P. Ciancarini, A. Fantechi and R. Gorrieri, eds.), Kluwer Academic Publishers, 1999, pp. 293-312.)

[6] Harel, D. and Marelly, R Come, Let's Play: A Scenario-Based Approach to Programming, Springer- Verlag, to appear (2003)

[7] Harel, D. and Marelly, R., Specifying and Executing Behavioral Requirements: The Play In/Play-Out Approach, Software and System Modeling (SoSyM), to appear (2003)

[8] Riddle, D.L., Blumenthal, T. , Meyer, B.J., Priess, J.R. (eds.): C. elegans II Cold Spring Harbor Laboratory Press Plainview, NY (1997)

[9] The C. elegans Sequencing Consortium: Genome sequence of the nematode C. elegans: a platform for investigating biology. The C. elegans Sequencing Consortium Science 282 (1998) 2012-2018

[10] Fraser, A.G., Kamath, R.S., ZipperlenI, P., Martinez-Campos, M., Sohrmann, M., Ahringer, J.: Functional genomic analysis fo C. elegans chromosome I by systematic RNA interference Nature 408 (2000) 325-330

[11] Piano, F., Schetter, A. J., Mangone, M., Stein, L., Kemphues, K. J.: RNAi analysis of genes expressed in the ovary of Caenorhabditis elegans Curr Biol 10(24) 2000 1619-1622

[12] Gonczy, P., Echeverri, C., Oegema, K., Coulson, A., Jones, S. J., Copley, R. R., Duperon, J., Oegema, J., Brehm, M., Cassin, E., Hannak, E., Kirkham, M., Pichler, S., Flohrs, K., Goessen, A., Leidel, S., Alleaume, A. M., Martin, C., Ozlu, N., Bork, P., Hyman, A. A.: Functional genomic analysis of cell division in C. elegans using RNAi of genes on chromosome III Nature 408 (2000) 331-336

[13] Maeda, I., Kohara, Y., Yamamoto, M., Sugimoto, A.: Large-scale analysis of gene function in Caenorhabditis elegans by high-throughput RNAi Curr Biol 11(3) (2000) 171-176

[14] Wang, M., Sternberg, P. W.: Pattern formation during C. elegans vulval induction Curr Top Dev Biol 51 (2001) 189-220

[15] Sternberg, P. W., Horvitz, H. R.: Pattern formation during vulval development in C. elegans Cell 44 (1986) 761-772

[16] Harel, D. and Marelly, R., Playing with Time: On the Specification and Execution of Time-Enriched LSCs, Proc. 10th IEEE/ACM Int. Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2002), Fort Worth, Texas (2002)

[17] Marelly, R., Harel, D. and Kugler, H., Multiple Instances and Symbolic Variables in Executable Sequence Charts, Proc. 17th Ann. AM Conf. on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 2002) 83-100

[18] Harel, D., Kugler, H., Marelly, R. and Pnueli, A., Smart Play-Out of Behavioral Requirements, Proc. 4th Int. Conf. on Formal Methods in Computer-Aided Design (FMCAD 2002) 378-398

[19] http://www.wisdom.weizmann.ac.il/mathusers/kam/CelegansModel/Demos.htm

[20] Sternberg, P. W.: Lateral inhibition during vulval induction in Caenorhabditis elegans Nature 335 (1988) 551-554

[21] Harel, D., A Grand Challenge for Computing: Full Reactive Modeling of a Multi-Cellular Animal, position paper, UK Workshop on Grand Challenges in Computing Research, Oct. 2002; available at http://www.wisdom.weizmann.ac.il /~dharel/papers/GrandChallenge.doc

Causal π-Calculus for Biochemical Modelling*

Michele Curti[1], Pierpaolo Degano[1], and Cosima Tatiana Baldari[2]

[1] Dipartimento di Informatica, Università di Pisa
Via Filippo Buonarroti 2, I-56127 Pisa, Italy
{curtim,degano}@di.unipi.it
[2] Dipartimento di Biologia Evolutiva, Università di Siena
Via Aldo Moro, 2 I-53100 Siena, Italy
baldari@unisi.it

Abstract. We present a reduction semantics for the π-calculus from which causality and concurrency can be mechanically derived. We prove that our semantics agrees with the causal definitions presented in the literature. We then apply the causal reduction semantics in the domain of biochemical systems to study the interactions of components.

Keywords: Concurrency, Causality, Reduction Semantics, Bio-informatics, Formal Description of Biochemical Processes

1 Introduction

Causality in concurrency has received a lot of attention. Its supporters claim that it permits more accurate and often more concise representations of system behaviour. Degano & Priami [3] proposed a structural operational semantics for the π-calculus that is interleaving in style, but has enough information to (mechanically) derive causal description.

Recently, Regev, Silvermann & Shapiro [22] proposed the π-calculus as a model of biochemical processes, seen as network of proteins. The authors claim that the main advantage over other proposals (see among others [8, 5, 6, 11, 12]) is that the π-calculus permits to better integrate dynamics, molecular and biochemical details. This calculus has solid theoretical basis, widely investigated. Additionally, the π-calculus has a linguistical structure, unlike other approaches leading to similar descriptions (*e.g.* Petri Nets [9]). Further work lead to a more precise model [20], that describes also quantitative aspects of reactions, such as time and probability. Its authors use the stochastic π-calculus [19], an extension of the original calculus with probabilistic distributions that govern the race conditions. Slight extensions are only needed to the standard reduction semantics to take care of the quantitative information (plus a further rule to model homodimerization).

* The first two authors have been partially supported by UE project IST-32072-DEGAS within the FET initiative on Global Computing and by the MIUR Project MEFISTO.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 21–34, 2003.

Although it offers a qualitative view of processes, causality seems to play a relevant role in understanding complex biochemical reactions. No reduction semantics was available expressing also causality. In this paper we offer such a causal semantics for the π-calculus, and we apply it to a small biological example. Our semantics can be considered unconventional, because our reductions do carry labels, from which the causality and the concurrency relations are mechanically derived. Indeed, we mimic the so-called enhanced operational semantics [3]. The information stored in reduction labels make our proposal flexible: also other aspects of processes can be derived from labels, in particular quantitative aspects [2]. Separation of concerns suggest us to discuss here only causality; for a survey on enhanced operational semantics, see [4].

Our example models a well-characterized biochemical process: the activation of the transcription factor $NF\text{-}AT$, which plays a crucial role in the process of T-cell activations. This process is particularly important in immunology and oncology. Our simulation reflects faithfully experiments *in vitro*, especially the two pathways activated by the T-cell antigene receptor (TCR) that lead to the activation of the transcription factors AP-1 and $NF\text{-}AT$.

This makes us confident that our proposal can be used both as a descriptive tool and as a prescriptive one. Also, the causal relation we extract from the computations of processes of the π-calculus has a meaningful graphical representation. These pictures seem to be superior to those commonly used by biologists to describe biochemical processes (*e.g.* KEGG [13] or EcoCyc [11]), because they explicitly describe pathway *evolutions*, originated from a formal model. Consequently, the attention is focused on the flow of reactions that occur in the process, rather then on reactants only. Furthermore, while other computational tools have been developed for the description of biological pathways, our approach appears particularly suitable to the analysis of signaling pathways madiated by protein-protein interactions based on the conversion of their individual molecular components from an *off* to an *on* state. Also, a formal description of the pathway should permit software simulations that offer cheap pre-views of tests before actually carrying them out.

As a matter of fact, the qualitative and the quantitative aspects are orthogonal to each other. So, the stochastic and the causal semantics can be combined together to yield even more detailed and accurate models of biochemical processes, taking care of the probability that reaction have to occur. As said above, our reduction semantics is flexible and supports both aspects (and more): the stochastic [18] and the causal descriptions of biological evolution can be specified within a single model.

The paper is organized as follows. In the next section we shall present our causal reduction semantics of the π-calculus, along with a comparison with similar work. In this extended abstract we only state our results and omit their proofs, that often are by structural induction. In Section 3 we shall briefly survey how biochemical processes may be represented as processes of the π-calculus; then we present our example and, driven by the model, we derive a graphical representation of the causal relations in the resulting pathways.

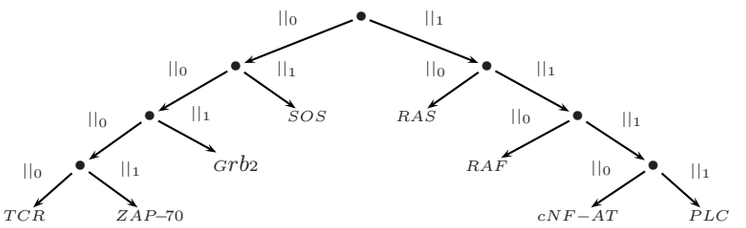


Fig. 1. The tree of (sequential) processes of NF-AT activation (see Sec. 3)

2 A Causal Reduction Semantics for the π-Calculus

We assume the reader familiar with the π-calculus [15] and with its reduction semantics. In order to show the correctness of our causal reduction semantics we shall compare it with the causal structural operational semantics [3] that we omit here because of lack of space.

Our starting point is the notion of *address* of a sequential component of a system R (roughly, a process with a prefix or a summation as top-level operator). Consider the abstract syntax tree of R, built using binary parallel composition as the main operator. So parallel processes are the nodes of the tree and sequential processes are its leaves. Now, label the arc leading to the right (resp. left) son of a parallel composition with $||_0$ (resp. $||_1$). The (label of the) path from the root to a sequential process is its address. Figure 1 shows the tree of processes of the network of proteins used in our example. The process ZAP–70 has address $||_0||_0||_0||_1$. Once chosen a particular tree of processes, the addresses uniquely identify the sequential processes of the system (more in [2]).

SYNTAX. Differently than in [3], we change the syntax of processes so that each action also carries the address of the sequential component it belongs to. In what follows, we shall denote by $A, B, \ldots$ the processes of the standard π-calculus, while the "extended" processes that we use are defined below, in a way that facilitates the comparison with [3]. For the sake of presentation we give a simplified syntax of our extended processes; a simple static check ensure well-formedness of the tags prefixing the action.

Definition 1 (Extended processes). *An* address *is* $\vartheta \in \{||_0, ||_1\}^*$*, denoting with* λ *the empty one. Let* $\mathcal{N}$ *be a countable, infinite set of* names *ranged over by* $a, b, \ldots, x, y$*, and let* τ *be a distinguished element such that* $\mathcal{N} \cap \{\tau\} = \emptyset$*.* Extended processes*, or simply* processes *when unambiguous, are denoted by* $P, Q, R, \ldots$*, and are built from names according to the syntax*

$$P ::= \mathbf{0} \mid \vartheta\mu.P \mid P + P \mid P \,|\, P \mid (\nu x)P \mid [x = y]P \mid X(y)$$

where μ *is either* $x(y)$ *for* input*, or* $\overline{x}\langle y\rangle$ *for* output *or* τ *for* silent *moves and* X *is an agent identifier (with a single parameter* y*, for simplicity).*
We assume that processes are guarded*, so summands are on the form* $\mu.P$ *or* $\mathbf{0}$ *(that will be omitted when in trailing position).*

Below, we define a function $\mathcal{T}$ that maps standard processes into extended ones, by inductively prefixing actions with the address of their sequential processes; note that $\mathcal{T}$ unwinds the syntactic structure of processes until reaching a $\mathbf{0}$ or a constant. It is straightforward proving that $\mathcal{T}$ is a bijection between the two different presentation of processes, its inverse being the function that discards addresses (note that no congruence is assumed, yet). Given a process on its standard form, this transformation operates in linear time with the number of prefixes. Also, $\mathcal{T}$ specifies a pre-processing step, that however has to be re-applied to a constant when invoked (see the definition of structural congruence below).

Definition 2. *Let A, B be standard processes, and let $\rhd$ be the operator introduced in the next definition. The following is a bijection:*

- $\mathcal{T}(\mathbf{0}) = \mathbf{0}$
- $\mathcal{T}(\mu.A) = \mu.\mathcal{T}(A)$
- $\mathcal{T}(A + B) = \mathcal{T}(A) + \mathcal{T}(B)$
- $\mathcal{T}(A|B) = \|_0 \rhd \mathcal{T}(A) \mid \|_1 \rhd \mathcal{T}(B)$
- $\mathcal{T}((\nu a)A) = (\nu a)\mathcal{T}(A)$
- $\mathcal{T}([x = y]A) = [x = y]\mathcal{T}(A)$
- $\mathcal{T}(X(x)) = X(x)$

Definition 3 (Distributing addresses).

- $\vartheta \rhd \mathbf{0} = \mathbf{0}$
- $\vartheta \rhd (\vartheta'\mu.P) = \vartheta\vartheta'\mu.(\vartheta \rhd P)$
- $\vartheta \rhd P + Q = \vartheta \rhd P + \vartheta \rhd Q$
- $\vartheta \rhd P \mid Q = \vartheta \rhd P \mid \vartheta \rhd Q$
- $\vartheta \rhd (\nu a)P = (\nu a)\ \vartheta \rhd P$
- $\vartheta \rhd [x = y]P = [x = y]\vartheta \rhd P$
- $\vartheta \rhd X(x) = \vartheta X(x)$

For our subsequent treatment, it is convenient introducing two auxiliary operators. The first one selects the subprocess of a process, seen as a tree of processes, that is reachable through a given address. Below, let $i \in \{0, 1\}$, and P_i, Q be extended processes.

Definition 4 (Selector). *The selector operator is defined as follows:*

- $P@\lambda = P$
- $(\nu a)P@\vartheta = P@\vartheta$
- $[x = y]P@\vartheta = P@\vartheta$
- $(||_0 P_0 \mid ||_1 P_1)@||_i\vartheta = P_i@\vartheta$

In what follows, given a process P, we shall occasionally substitute a process Q for the process $P@\vartheta$, within P. The next operator take care of this. Note that Q will replace a whole summation, when ϑ is empty (as $P + R$ is considered as sequential; see also the selector operator).

Definition 5. *The localized substitution of Q at ϑ within P is defined as:*

- $P[\lambda \mapsto Q] = Q$
- $\big(||_0 P_0 \mid ||_1 P_1\big)[||_0\vartheta \mapsto Q] = (||_0 P_0[\vartheta \mapsto Q]) \mid ||_1 P_1$
- $\big(||_0 P_0 \mid ||_1 P_1\big)[||_1\vartheta \mapsto Q] = ||_0 P_0 \mid (||_1 P_0[\vartheta \mapsto Q])$
- $((\nu a)P)[\vartheta \mapsto Q] = P[\vartheta \mapsto Q]$
- $([x = y]P)[\vartheta \mapsto Q] = P[\vartheta \mapsto Q]$

CONGRUENCE. Below we assume the *structural congruence* $\equiv$ on processes, that is the standard one of the π-calculus, apart from a slightly different treatment of constant definition. Ours is the least congruence satisfying the following clauses (as usual $fn(P)$ is the set of the free names of P):

1. $P \equiv Q$ if P and Q are α-equivalent;
2. $(\mathcal{P}/_{\equiv}, +, \mathbf{0})$ and $(\mathcal{P}/_{\equiv}, |, \mathbf{0})$ are commutative monoids;
3. $[x = x]P \equiv P$;
4. $(\nu\, x)(\nu\, y)P \equiv (\nu\, y)(\nu\, x)P$ (simply written as $(\nu\, \{x, y\})P$),
   $(\nu\, x)P \equiv P$ if $x \not\in fn(P)$, $\quad (\nu\, x)(P \mid Q) \equiv (\nu\, x)P \mid Q$ if $x \not\in fn(Q)$;
5. $\vartheta X(x) \equiv \vartheta \rhd P$ if $X(x) \stackrel{def}{=} A$ and $\mathcal{T}(A) = P$.

It is immediate recovering from the above the standard structural congruence [16]: it suffices discarding from extended processes their addresses (and obviously $\mathcal{T}(A) = P$ from the last item).

STRUCTURE PRESERVING MANIPULATIONS. We now state a couple of properties that relate the syntactic structure of standard and of extended processes. These facts facilitate the proof of the equivalence between the causal SOS semantics and the causal reduction semantics we are going to introduce below. (Recall that A, B are processes of the standard π-calculus.)

Our first property (*a1-2* below) shows that the function $\mathcal{T}$, while generating extended processes from standard ones, preserves structural congruence, provided that associativity and commutativity of the parallel operator is not assumed (nor that $\mathbf{0}$ is its neutral element) — a basic choice in the causal semantics of [3]. Our second property (*b1-2* below) shows that also the selector operator preserves structural congruence in the same way.

Property. Let $\equiv'$ be the least congruence on standard processes satisfying the clauses defining $\equiv$, except that $(\mathcal{P}/_{\equiv}, |, \mathbf{0})$ is not a commutative monoid. Then

*a*1. $A \equiv' B$ implies $\mathcal{T}(A) \equiv \mathcal{T}(B)$ *and*
*a*2. $\mathcal{T}(A) \equiv \mathcal{T}(B)$ implies $A \equiv B$

*b*1. $A@\vartheta \equiv' \mu.B$ implies $\mathcal{T}(A)@\vartheta \equiv \vartheta\mu.\mathcal{T}(B)$ *and*
*b*2. $\mathcal{T}(A)@\vartheta \equiv \vartheta\mu.\mathcal{T}(B)$ implies $A@\vartheta \equiv \mu.B$

From now onwards, we feel free to re-arrange processes in such a way that restrictions are all put in outermost position, by enlarging their scope and possibly α-converting names (recall Turner-Milner theorem [14]).

REDUCTION SEMANTICS. Table 1 shows our reduction semantics. Unconventionally, it carries labels on the arrows to extract causality, just as it happens with the SOS causal semantics of [3]. The labels record the address where the action took place. In case of a communication, also the addresses and the actions $\vartheta_0 x(w)$ and $\vartheta_1 \overline{x}\langle y\rangle$ of the partners P_{in} and P_{out} are recorded in a pair, each prefixed by $||_0$ and $||_1$ respectively, because they necessarily lay in two opposite sides of a parallel operator. In turn, $P_{in} \mid P_{out}$ is plugged in some context, to which the address ϑ corresponds. So, ϑ prefixes the pair input/output.

Table 1. Non-interleaving reduction semantic

$$\textbf{Com}: \ (R+\vartheta||_0\vartheta_0 x(w).P)|(S+\vartheta||_1\vartheta_1\overline{x}\langle y\rangle.Q) \xrightarrow{\vartheta\langle||_0\vartheta_0 x(w),||_1\vartheta_1\overline{x}\langle y\rangle\rangle}_R P\{y/w\}|Q$$

$$\textbf{Par}: \frac{P \xrightarrow{\theta}_R P'}{P|Q \xrightarrow{\theta}_R P'|Q} \qquad \textbf{Res}: \frac{P \xrightarrow{\theta}_R P'}{(\nu a)\ P \xrightarrow{\theta}_R (\nu a)\ P'}$$

$$\textbf{Tau}: \vartheta\tau.P \xrightarrow{\vartheta\tau}_R P \qquad \textbf{Struct}: \frac{Q \equiv P \quad P \xrightarrow{\theta}_R P' \quad P' \equiv Q'}{Q \xrightarrow{\theta}_R Q'}$$

Definition 6. *A* label *is defined by the following syntax*

$$\theta ::= \vartheta\mu \mid \vartheta\langle||_0\vartheta_0 x(w), ||_1\vartheta_1\overline{x}\langle y\rangle\rangle$$

Note that the axiom **Com** originates the label of the reduction by singling out the common prefix ϑ of the input and the output actions. We also have an axiom for a silent move, because this permits to keep short the representation of biological behaviour. The other rules are quite standard.

We now show that our reduction semantics is consistent with those presented in the literature. In particular, we shall consider the SOS causal semantics [3], that is based on the so-called proved approach. We refer the reader to the original paper for a complete presentation of this semantics, and here we only give its flavour, by reporting below the rules for asynchrony and for communication (there are also two further symmetric rules); note that this semantics involves standard processes.

$$\frac{A \xrightarrow{\theta}_S A'}{A|B \xrightarrow{||_0\theta}_S A'|B} \qquad \frac{A \xrightarrow{\vartheta_0 x(w)}_S A',\ B \xrightarrow{\vartheta_1\overline{x}\langle y\rangle}_S B'}{A|B \xrightarrow{\langle||_0\vartheta_0 x(w),||_1\vartheta_1\overline{x}\langle y\rangle\rangle}_S A'|B'}$$

In showing that the reduction and the SOS semantics coincide, the following two lemmata help, one for each semantics, having exactly the same meaning and shape. They show that it is possible to single out the sub-processes active in a transition labelled by Θ, by only inspecting Θ itself; similarly for their residuals.

Lemma 1. *$A \xrightarrow{\vartheta\langle||_0\vartheta_0 x(w),||_1\vartheta_1\overline{x}\langle y\rangle\rangle}_S B$ if and only if*
$A@\vartheta||_0\vartheta_0 \equiv x(w).A_0 + C_0, \quad A@\vartheta||_1\vartheta_1 \equiv \overline{x}\langle y\rangle.A_1 + C_1,$
$B \equiv A\big[\vartheta||_0\vartheta_0 \mapsto A_0\{y/w\}\big]\big[\vartheta||_1\vartheta_1 \mapsto A_1\big].$

The following is the analogous lemma for the reduction semantics.

Lemma 2. *$P \xrightarrow{\vartheta\langle||_0\vartheta_0 x(w),||_1\vartheta_1\overline{x}\langle y\rangle\rangle}_R Q$ if and only if*
$P@\vartheta||_0\vartheta_0 \equiv x(w).P_0 + C_0,\ P@\vartheta||_1\vartheta_1 \equiv \overline{x}\langle y\rangle.P_1 + C_1,$
$Q \equiv P[\vartheta||_0\vartheta_0 \mapsto P_0\{y/w\}][\vartheta||_1\vartheta_1 \mapsto P_1].$

Now, the equivalence between the SOS and the reduction semantics follows from the above by a simple inductive argument.

Theorem 1. *$A \xrightarrow{\vartheta\langle\theta_0,\theta_1\rangle}_S B$ if and only if $\mathcal{T}(A) \xrightarrow{\vartheta\langle\theta_0,\theta_1\rangle}_R \mathcal{T}(B)$.*

Note that in the statement, we restrict our attention to extended processes coming from standard ones; in other words, we only consider extended processes well-formed with respect to the parallel structure (*e.g.* $||_1 x(w) \mid ||_1 \overline{x}\langle y\rangle$ will be rejected).

CAUSALITY. Theorem 1 enables us to carry over the computations defined through the reduction semantics all the non-interleaving aspects defined for the SOS semantics in [3]. Here, we consider only one aspect of those considered there, namely *causality*. It will be used to accurately describe qualitative aspects of biological evolutions. A different interpretation of transition labels permits to represent also quantitative aspects (see *e.g.* the stochastic π-calculus in [19] where occurrence rates are assigned to reactions, or other aspects captured by the relabelling functions of [18, 4]). The two interpretations are orthogonal and can be combined together to enhance the expressivity of our descriptions.

The idea underlying causality between two transitions t and t' is that t occurs before t' and that the two are nested in a prefix chain. Such a nesting is reflected in their labels: the (address in the) label of t is a prefix of (the address of) the other. Following this intuition, we rephrase below the definition of causality given in [3] and called there enabling. We make use of the following auxiliary function that "flattens" labels:

$$- \ f(\vartheta\tau) = \{\vartheta\tau\}; \qquad\qquad - \ f(\vartheta\langle\theta_0,\theta_1\rangle) = \{\vartheta\theta_0, \vartheta\theta_1\}$$

Definition 7 (Causal Relation). *Let $P_0 \xrightarrow{\theta_0}_R P_1 \xrightarrow{\theta_1}_R \ldots \xrightarrow{\theta_n}_R P_{n+1}$ be a computation, and stipulate that $\theta_i \sqsubset \theta_j$ iff $i < j$ and $\exists \theta = \vartheta\mu \in f(\theta_i)$ and $\theta' \in f(\theta_j) = \vartheta\vartheta'\mu'$ (*i.e. *ϑ is a prefix of $\vartheta\vartheta'$). Then:*

$\theta_i \sqsubseteq \theta_j$ iff $\theta_i \sqsubset^* \theta_j$ (in words θ_i causes *θ_j)*

where $\sqsubseteq$ is the reflexive, transitive closure of $\sqsubset$.

Example 1. Consider the transitions t_0, t_1 and t_3, that occur in this temporal ordering and that have the following labels:

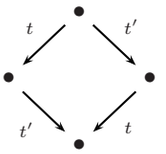
- $\langle ||_0||_0||_0||_0\overline{bind\_z}\langle tcr\rangle, ||_0||_0||_0||_1 bind\_z(tcr)\rangle$
  - $\langle ||_0||_0||_0||_1||_0\overline{bind\_grb2}\langle lat\rangle, ||_0||_0||_1 bind\_grb2(lat)\rangle$
    - $\langle ||_0||_0||_0||_1||_1\overline{bind\_PLC}\langle lat1\rangle, ||_1||_1||_1||_1 bind\_PLC(lat1)\rangle$.

Now, $t_0 \sqsubseteq t_1$ because $||_0||_0||_0||_1 bind\_z(tcr) \sqsubset ||_0||_0||_0||_1||_0\overline{bind\_grb2}\langle lat\rangle$ and $t_0 \sqsubseteq t_3$ because $||_0||_0||_0||_1 bind\_z(tcr) \sqsubset ||_0||_0||_0||_1||_1\overline{bind\_PLC}\langle lat1\rangle$. Instead, neither $t_1 \sqsubseteq t_3$ nor $t_3 \sqsubseteq t_1$.

In the example above, there are two transitions that are not causally related, and we call them *concurrent*, as usual. In symbols,

$$t \ / \ \overleftrightarrow{\sqsubseteq} \text{ if and only if } t \smile t'.$$

Due to the Theorem 6.4 of [3], the notion of concurrency can be lifted to processes. Intuitively, this theorem says that whenever two transitions t and t' are concurrent, they form a diamond in the transition system with the following shape:



Therefore, if t and t' are detected to be concurrent in a computation, they will be such in *all* the computations in which they both occur.

A graphical representation of causality can be immediately derived from a computation and from the causal relations introduced above. Indeed, we construct a directed (acyclic) graph whose nodes are the transitions themselves, and there is an arc from a transition t_1 to t_2 if and only if $t \sqsubseteq t'$. As an example consider Figure 2 that represent the causality relation arising from our modeling of the NF-AT activation (see next section). As usual, we only draw the Hasse diagram of the causal relation: an arc from t_1 to t_3 is omitted if there exists t_2 such that there are arcs from t_1 to t_2 and from t_2 to t_3. So causality is explicitly represented, while absence of (a chain of) arrows between t_1 to t_2 means that the two are concurrent: they may occur in any order and even at the same time. We hope that this graphical representation may help biologists to describe biochemical processes.

3 An Example

In this section we apply our reduction semantic to a well characterized biochemical process: the activation of the transcription factor $NF\text{-}AT$, which plays a crucial role in the process of T-cell activations. $NF\text{-}AT$ is composed of two subunits, a cytosolic subunit, which belongs to the family of $NF\text{-}AT$ proteins, and a nuclear subunit, identified as $AP\text{-}1$. Activation of $NF\text{-}AT$ requires translocation of the cytosolic subunit to the nucleus, where it can assemble with $AP\text{-}1$. The latter in turn requires to be phosphorylated to become transcription-competent. Regulation of the subcellular localization of $NF\text{-}AT$ is achieved by phosphorylation on specific serine residues. When phosphorylated, the nuclear localization signal of $NF\text{-}AT$ is masked and $NF\text{-}AT$ is therefore segregated to the cytosol. T-cell antigen receptor (TCR) engagement triggers an increase in intracellular calcium ions, which induces the activation of the phosphatase calcineurin. Dephosphorylation of $NF\text{-}AT$ by calcineurin results in exposure of the nuclear localization sequence and translocation of $NF\text{-}AT$ to the nucleus. $AP\text{-}1$ on the other hand is heterodimer of Jun and Fos and is activated by phosphorylation of Jun and Fos on serine residues. Activation of the MAP kinase pathway, which is also triggered by the TCR, is required for this process to occur [21]. Hence TCR engagement results in the activation of both pathways, the Ras/MAP kinase and the calcium/calcineurin pathway, required for $NF\text{-}AT$ activation [10].

Essentially, molecules are a set of *domains* denoted by *motifs*. Two molecules can interact if they have two *complementary* motifs. Such an interaction produces a result called the *residual* of the reaction. Any molecule has some private "information", the *backbone*, that determines its identity. The interaction between two molecules can be seen then as sharing a backbone. The same happens with protein complexes or cellular compartments [20].

Following [22], we view molecules and domains as concurrent processes in the π-calculus. Motifs are represented by global channels, and a reaction is a communication. Residuals of a reaction are processes prefixed by an input/output action. Backbones are private names of processes declared by a restriction. So, interaction between molecules results also in a scope enlargement. For simplicity, we will left many residuals empty. Indeed, in our example residuals are molecules activated (or deactivated) by phosphorylations (or dephosphorylations) and we ignore these activities here.

Our starting point is the system Sys composed by the (active and inactive) proteins needed to activate $NF\text{-}AT$. To show how causality helps describing this biochemical production, it suffices to have a single instance for each reagent. More copies only affect the readability of the system and the representation of computations (see Figure 2). Each protein is represented by a process, that runs in parallel with each other:

$$Sys = (((TCR \mid ZAP\text{–}70) \mid Grb2) \mid SOS) \mid (RAS \mid (RAF \mid (cNF-AT \mid PLC)))$$

The part in common with the Ras/MAP kinase and the calcium/calcineurin pathway involves two reactants, TCR and $ZAP\text{-}70$, that perform a phosphorylation of the Z-chains of TCR and an interaction between the two. The residual of recruitment of $ZAP\text{-}70$ by TCR enables the phosphorylation of LAT (*i.e.* his activation), that concludes the initialization. As said above, phosphorylation and dephosphorylation are ignored, as well as the residuals they activate. So we group the three steps above in a single activity, represented by the communication of the backbone tcr of TCR to $ZAP\text{-}70$.
Our specification of the proteins TCR and $ZAP\text{-}70$ becomes then:

$$-\ TCR = (\nu\ tcr)\ \overline{bind\_z}\langle tcr\rangle \qquad -\ ZAP\text{–}70 = bind\_z(tcr).LAT$$

Activated LAT has two specific phosphorylated tyrosine residuals responsible of interaction with $GRB2$ ($bind\_grb2$) and PLC ($bind\_PLC$). Also, it has a backbone that will be shared with the proteins which it will interact with, namely $Grb2$ and PLC. The two domains may interact in parallel along the two pathways. To model correctly these interactions, it is convenient to represent the backbone with two different names, each with its own scope. We call them $lat1$ and $lat2$.

Now we start modelling the first pathway, which leads to the activation of the nuclear subunit $AP\text{-}1$. The domain $(\nu\ lat1)\overline{bind\_grb2}\langle lat1\rangle$ of LAT has no residual and its motif $bind\_grb2$ is complementary to that of the protein $Grb2$. So the two proteins interact and activate the residual of $Grb2$. In turn, this interacts with SOS and activates RAS. A further interaction sequentially activates

RAF, MEK and ERK. The pathway ends with the phosphorylation of Jun and Fos by ERK that stimulates the ability of the AP-1 subunit to interact with the DNA in the cell nucleus. We specify the above in the standard π-calculus as follows, where the members of the map kinase cascade are represented by (indexed) internal actions τ.

$$\begin{aligned}
&\mathit{LAT} = (\nu\ \mathit{lat1})\overline{\mathit{bind_grb2}}\langle \mathit{lat1}\rangle \ \mid\ (\nu\ \mathit{lat2})\overline{\mathit{bind_PLC}}\langle \mathit{lat2}\rangle\\
&\mathit{Grb2} = \mathit{bind_grb2}(\mathit{lat1}).\overline{\mathit{bind_SOS}}\langle \mathit{lat1}\rangle\\
&\mathit{SOS} = \mathit{bind_SOS}(\mathit{lat1}).\overline{\mathit{bind_RAS}}\langle \mathit{lat1}\rangle\\
&\mathit{RAS} = \mathit{bind_RAS}(\mathit{lat1}).\overline{\mathit{bind_RAF}}\langle \mathit{lat1}\rangle\\
&\mathit{RAF} = \mathit{bind_RAF}(\mathit{lat1}).\tau_{MEK}.\tau_{ERK}.\mathit{AP\text{–}1}
\end{aligned}$$

Next, we model the second pathway that activates the cytosolic subunit, represented by the process $cNF-AT$. The domain $(\nu\ lat2)\overline{bind\_PLC}\langle lat2\rangle$ of LAT interacts with phospholipase C. This interaction results in an increase of the free intracellular calcium and activation of calcineurin CN. This molecule is a phosphatase that can dephosphorylate cNF-AT, enabling it to translocate to the nucleus. We group the interaction of CN with cNF-AT and the dephosphorylation of the latter as a single communication between the two molecules.

$$\begin{aligned}
&\mathit{PLC} = \mathit{bind_PLC}(\mathit{lat2}).\mathit{CN}\\
&\mathit{AP\text{–}1} = \mathit{bind_ap1}(\mathit{cn})\\
&\mathit{CN} = (\nu\ \mathit{cn})\overline{\mathit{bind_sub}}\langle \mathit{cn}\rangle\\
&\mathit{cNF{-}AT} = \mathit{bind_sub}(\mathit{cn}).\mathit{nNF{-}AT}\\
&\mathit{nNF{-}AT} = \overline{\mathit{bind_ap1}}\langle \mathit{cn}\rangle
\end{aligned}$$

As mentioned above, the tuning of NF-AT depends on both pathways. The first one, RAS/MAP kinases, activates the nuclear subunit AP-1. The second pathway, calcium/calcineurin, enables AP-1 to translocate within the nucleus of a cell, where it interacts with the DNA.

To show these causality relations, we first extend the specification above according to Def. 2. (Admittedly, this representation is heavy, but it is intended to be internal, only, and is mechanically generated.) Only the whole system and the constant LAT are affected and prefixed by suitable strings made of tags $||_0$ and $||_1$:

$$\begin{aligned}
\mathit{Sys} = (((&||_0||_0||_0||_0\mathit{TCR} \mid ||_0||_0||_0||_0||_1\mathit{ZAP\text{–}70}) \mid ||_0||_0||_1\mathit{Grb2}) \mid ||_0||_1\mathit{SOS}) \mid\\
&(||_1||_0\mathit{RAS} \mid (||_1||_1||_0\mathit{RAF} \mid (||_1||_1||_1||_0\mathit{cNF{-}AT} \mid ||_1||_1||_1||_1\mathit{PLC})))
\end{aligned}$$

$$\mathit{LAT} = (\nu\ \mathit{lat1})||_0\overline{\mathit{bind\_grb2}}\langle \mathit{lat1}\rangle \mid (\nu\ \mathit{lat1})||_1\overline{\mathit{bind\_PLC}}\langle \mathit{lat1}\rangle$$

A computation of Sys is in Figure 2. The labels of the transitions are:

$t_0 - \langle ||_0||_0||_0||_0\overline{bind\_z}\langle tcr\rangle, ||_0||_0||_0||_1 bind\_z(tcr)\rangle$

$t_1 - \langle ||_0||_0||_0||_1||_0\overline{bind\_grb2}\langle lat\rangle, ||_0||_0||_1 bind\_grb2(lat)\rangle$

$t_2 - \langle ||_0||_0||_1\overline{bind\_SOS}\langle lat\rangle, ||_0||_1 bind\_SOS(lat)\rangle$

$t_3 - \langle ||_0||_0||_0||_1||_1\overline{bind\_PLC}\langle lat1\rangle, ||_1||_1||_1||_1 bind\_PLC(lat1\rangle$

$t_4 - \langle ||_0||_1\overline{bind\_RAS}\langle lat\rangle, ||_1||_0 bind\_RAS(lat)\rangle$

$t_5 - \langle ||_1||_0\overline{bind\_RAF}\langle lat\rangle, ||_1||_1||_0 bind\_RAF(lat)\rangle$

$t_6 - \langle ||_1||_1||_1||_1\overline{bind\_sub}\langle cn\rangle, ||_1||_1||_1||_0 bind\_sub(cn)\rangle$

$t_7 - ||_1||_1||_0\tau_{MEK}$

$t_8 - ||_1||_1||_0\tau_{ERK}$

$t_9 - \langle ||_1||_1||_1||_0\overline{bind\_ap1}\langle cn\rangle, ||_1||_1||_0 bind\_ap1(cn)\rangle$

The example 1 shows that the initial interaction t_0 causes both t_1 and t_3, that instead are unrelated. It is also straightforward deducing that also $t_1 \sqsubseteq t_2 \sqsubseteq t_4 \sqsubseteq t_5 \sqsubseteq t_7 \sqsubseteq t_8$ but none of them depend on t_3, t_6, or *viceversa*. Finally, $t_5, t_6 \sqsubseteq t_9$, so reflecting that $NF\text{-}AT$ needs both pathways.

4 Conclusions

We gave the π-calculus a reduction semantics. It is unconventional because reductions have labels. A mechanical interpretation of them provides us with the causal relations between the transitions in a computation.

Besides its interest *in se*, a causal reduction semantics receives motivation from its ability to faithfully describe biochemical processes. In fact, the dynamic analysis of this kind of specifications can predict "real" biological evolutions at a certain degree of accuracy [22, 20]. We demonstrated that causality enhances the precision of such simulations, through a simple example, that however suffices to make our point. The description of the network of proteins we considered shows the potential interactions between them, stage by stage. The causal behaviour of the π-process agrees with the experiments *in vitro*. In our example, there is a single computation; of course our abstract semantics produces all the possible computations, covering all the evolution of the system.

Another way to improve accuracy of modelling biological systems is by endowing the π-calculus with stochastic information [20], an extension of the original calculus with probabilistic distributions that govern the race conditions. This quantitative aspect and causality are orthogonal, so they can be combined together. Remarkably, the labels of our enhanced reduction semantics can also be interpreted to give raise to a stochastic model [18]. Thus, our semantics provides the basis for a single framework permitting accurate qualitative and quantitative descriptions and analyses of biochemical process. The designer writes a single specification and, without changing it, s/he can derive these different descriptions

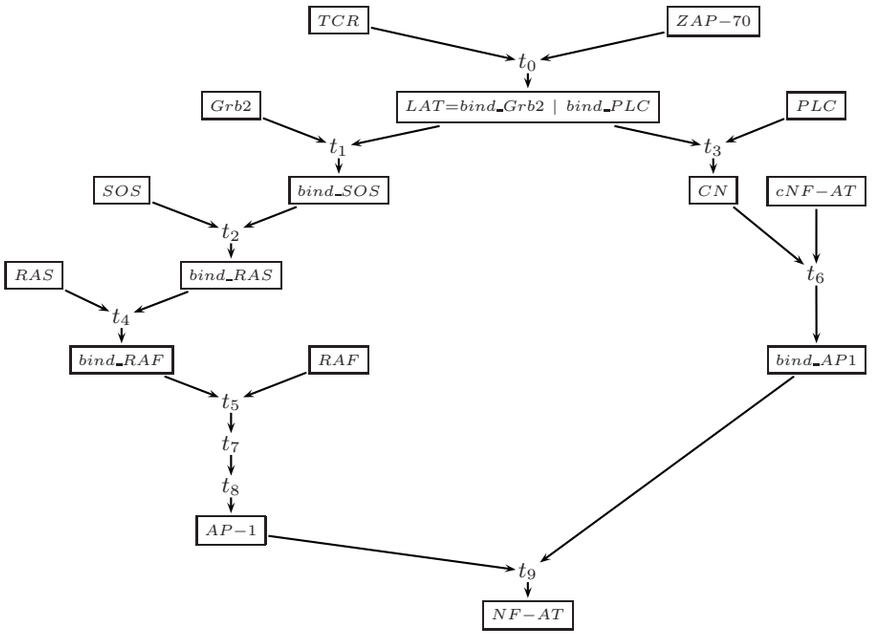


Fig. 2. A computation of Sys. For readability, the processes, enclosed in boxes, have no address. Causality (both on transitions and processes) is represented by the (Hasse diagram resulting from the) arrows; their absence makes it explicit concurrent activities

of its evolution. We claim that our modelization makes biochemical descriptions simpler (cf. Figure 2) than those relying on other models, e.g. [8, 5, 6, 11, 12], because here we establish a close link between each interaction of proteins and each step in the system evolution.

A (semi-)automatic tool is under development, that supports the definition of a biochemical system as a π-calculus process, its execution and the display of the resulting pathway(s) in a graphical form. Among the features of our tool, there should be the possibility of changing the grain of the graphical representation by grouping together families of reactants. In this way, long or complex pathways can be represented at different levels of detail, so enhancing the readability of the representation, even in presence of many interacting components. Also, different views of the same pathway should be displayed, e.g. with or without quantitative information. Finally, it could be a mess representing in a single screen all the many different computations that may arise from the same biochemical process being modelled: our tool will offer its users a facility for interactively choosing among them.

References

[1] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96:217–248, 1992.

[2] C. Bodei, Degano P., and Priami C. Names of mobile agents handled locally. *Theoretical Computer Science*, 253, 2:155–184, 2000. 22, 23

[3] P. Degano and C. Priami. Non-interleaving semantics for mobile processes. *Theoretical Computer Science*, 216, 1-2:237–270, 1999. 21, 22, 23, 25, 26, 27, 28

[4] P. Degano and C. Priami. Enhanced operational semantics: A tool for describing and analysing concurrent systems. *ACM Computing Surveys*, 33, 2:135–176, 2001. 22, 27

[5] S. Eker, M. Knapp, K. Laderoute, P. Lincoln, J. Meseguer, and K. Sonmez. Pathway logic: Simbolic analysis of biological signaling. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 400–412, 2002. 21, 32

[6] D. Endy and R. Brent. Modelling cellular behaviour. *Nature*, 409:391–395, 2001. 21, 32

[7] G. Fournet and G. Gonthier. The reflexive chemical abstract machine and its join calculus. In *Proc. 23rd ACM POPL*, pages 372–385, 1996.

[8] P. J. E. Goss and J. Peccoud. Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. In *Proceedings of the National Academy of Sciences USA*, number 12, pages 6750–6754, 1998. 21, 32

[9] M. Heiner, I. Koch, and K. Voss. Analysis and simulation of steady states in metabolic pathways with petri nets. In K. Jensen, editor, *CPN'01-Third Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pages 15–34, 2001. 21

[10] L. P. Kane, J. Lin, and A. Weiss. Signal transduction by the tcr for antigen. *Curr. Opin. Immunol.*, 12:242–249, 2000. 28

[11] Peter D. Karp. Pathway databases: A case study in computational symbolic theories. *Science*, 293:2040–2044, 2001. 21, 22, 32

[12] Kurt W. Kohn. Molecular interaction map of the mammalian cell cycle control and dna repair systems. *Molecular Biology of the Cell*, 10:2703–2734, 1999. 21, 32

[13] Kanehisa M. and Goto S. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, 28(1):27–30, Jan 1 2000. 22

[14] R. Milner. Functions as processes. *Math. Struct. in CS*, 2(2):119–141, 1992. 25

[15] R. Milner. *Communicating and Mobile Systems: the π-calculus.* Cambridge Univ. Press, 1999. 23

[16] R. Milner, J. Parrow, and D. Walker. Modal logics for mobile processes. *Theoretical Computer Science*, 114:149–171, 1993. 25

[17] M. Nagasaki, S. Onami, S. Miyano, and Kitano H. Bio-calculus: Its concept and molecular interaction. *Genome Informatics*, 10:133–143, 1999.

[18] C. Nottegar, C. Priami, and P. Degano. Performance evaluation of mobile processes via abstract machines. *IEEE Trans. on Software Engineering*, 27, 10:867–889, 2001. 22, 27, 31

[19] C. Priami. Stochastic π-calculus. *The Computer Journal*, 38, 6:578–589, 1995. 21, 27

[20] C. Priami, A. Regev, W. Silverman, and E. Shapiro. Application of a stochastic passing-name calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80:25–31, 2001. 21, 29, 31

[21] A. Rao, C. Luo, and P. G. Hogan. Transcription factors of the nfat family: regulation and function. *Annu. Rev. Immunol.*, 15:707–747, 1997. 28

[22] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. In *Pacific Symposium of Biocomputing (PSB2001)*, pages 459–470, 2001. 21, 29, 31

Graphs for Core Molecular Biology

Vincent Danos[1] and Cosimo Laneve[2]

[1] Équipe PPS, CNRS and University of Paris 7
[2] Department of Computer Science, University of Bologna

Abstract. A graphic language—the *graphic κ calculus*—modeling protein interactions at the domain level is introduced. Complexation, activation and mixed forms of interaction are expressed as graph rewriting rules. A compilation in a low-level graph rewriting calculus, called *mobile κ*, is given and shown to be correct up to some suitable notion of observational equivalence. This intermediate language is of independent interest and can be easily implemented in π-like calculi.

1 Introduction

While the traditional means of modeling in biology has been mostly continuous-time differential systems, Regev, Shapiro [12, 13, 14] and Nagasaki [11], with their respective collaborators, have convincingly argued that low-level name-based communication languages [10, 7] may prove excellent tools for compositional and modular modeling of biological systems at the protein interaction level. As fascinating as this new and potentially huge field of application may be, it is still unclear which specific formalism one would like to use.

We proposed one in [4], namely the κ-calculus, and a simplified form was shown in [3] to be expressive enough to translate Kohn's molecular maps (a compilation of molecular data pertaining to the mammalian cell cycle control [8]). In the present paper, we explore an alternative and independent possibility, namely the *graphic κ calculus*. This new formalism embeds our former tentative calculus while going one step further in the details of molecular interaction.

Biological objects, bundles of proteins also known as complexes, which were taken in κ to be simple *multisets*, will now be connected *graphs*. Rules, or reactions, will see and manipulate the internal wiring and states of these objects. As a result, decomplexation (when complexes are dissolved) and more general graphical rewrite rules which κ could not express, are now easily representable. This is not to say one formalism is better than the other, they're just not working at the same level of abstraction.

Going further in details is not necessarily a good option and we stop before introducing in the model a true geometric aspect. There is no sign that we'll have in our hands, any time soon, a strong predictive theory of how interactive properties of proteins and compounds are deduced from their folding in space (a folding which itself is hardly computable from the sequence), so metric models seem a bit off-topic for the moment. We take interactive capabilities as a black box, and start the study of combinatorial protein networks from that level of

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 34–46, 2003.

abstraction. Additionally, metrics would require a quite different apparatus as the one we have in concurrency.

With the formalism in place, we construct a uniform translation to an intermediate language with only binary interactions and limited resources for name creation. We firmly believe this intermediate formalism, called *mobile* κ, to be translatable into π-calculus, join-calculus [10, 7] and their variants. It also provides an excellent abstraction from the details of doing a direct translation of graphic κ into one of these calculi.

This embedding maps formal proteins to agents and handles graph-rewritings, or reactions, as transactions between the agents involved. Specific initiator nodes are defined for each reaction, which try to set a transitory tree, represented by sharing of private names, spanning all participants in the reaction. Along this recruitment phase, it is also checked that agents are in the correct state that would fire the reaction. When and if this first phase succeeds, updates are performed. There can't be any failure past the first phase since recruited reactants are "booked" until the wave of updates has passed through them.

That such arbitrary transactions are reducible to peer-to-peer interactions is a fact which should be, but for some reason is not, a classical result in the theory of π-calculus.

Our translation can be taken literally as a distributed and asynchronous implementation of a reaction network. One can in principle run a virtual cell on the Internet! Not very useful perhaps, but quite in line with how things proceed at the molecular level. Thinking of concurrent processes as chemical reactions has been a long-standing working intuition in concurrency, as in for instance the Chemical Abstract Machine [1]. In a way, our embedding is bringing to the fore a precise counterpart of the converse intuition, namely that protein interactions constitute a rudimentary process algebra. And therefore, we can reasonably expect this analysis to give means to import and adapt automatic tools already developed by process algebraists [2, 9, 15].

How relevant this could prove to be for the understanding of biological systems remains to be seen.

Related Work. A language with quite similar concerns, but with an eye on applying rewriting theory rather than concurrency theory, called "Pathway Logic", has been recently proposed by Lincoln and collaborators in [6]. It is a rewriting system formalism, where reactants are proteins and reactions are modelled by rewriting rules. As in κ, internal wirings of proteins are not modeled so that it is actually closer to κ than to graphic κ. The implementation of Pathway Logic in the Maude system gives potential access to useful analytic tools.

Acknowledgements. The first author gladly acknowledges discussions with Marc Chiaverini, Magali Roux-Rouquié and Vincent Schächter, on formal approaches to molecular biology.

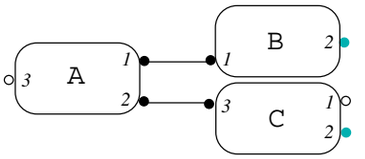
2 Graphic κ

We first present our formal calculus of protein interactions, called the *graphic κ calculus*, as a graph rewriting system. We'll avoid a formalistic presentation and rely mostly on drawings.

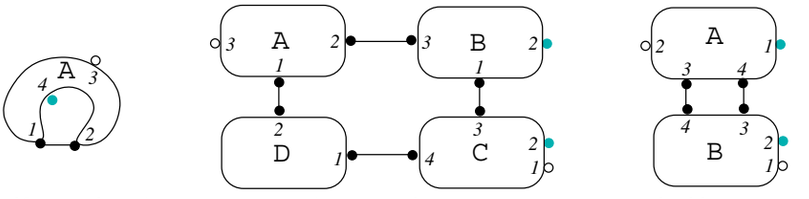
Sites, Proteins, and Complexes. We assume a countable set of *protein names* $\mathcal{A}$. Each protein name is associated with a finite set of *sites*, taken from a countable set $\mathcal{N}$, which we call its *interface*. Interfaces are given by a *signature* map $\mathfrak{s}(.)$ from $\mathcal{A}$ to finite sets over $\mathcal{N}$.

Formal proteins, or simply proteins, are drawn as boxes, with sites being written on the boundary of the box. We discriminate sites in the pictures by pairwise different natural numbers, 1, 2, 3, ... or letters a, b, c,..., written within the box. Sites may be either connected to other sites, in which case we say they are *bound*, or may be free in which case they are in either of two states: *hidden* or *visible*. We picture proteins as follows, where it is understood that all sites represented are in $\mathfrak{s}(A)$:

Proteins may be assembled into *protein complexes*, or simply *complexes*. Complexes are drawn by connecting two-by-two bound sites of proteins, thus building connected graphs such as:



which represents a compound made of A, B, and C, where A is connected with B and C, and there is no direct connection between B and C. Other examples of complexes are:



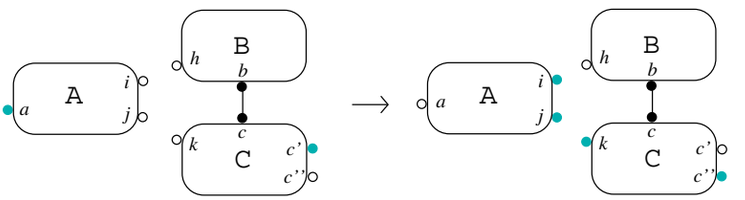
Solutions and Reactions. Solutions are multisets of proteins and complexes. A protein will have in general multiple occurrences in a given solution, so that to be precise one should refer to a particular *occurrence* of protein, but most of the time we don't do this. As the name suggests, solutions are taken up to associativity and commutativity, their complexes sort of "freely float" inside. Given a solution $\mathcal{S}$, we'll say a protein is *single* in $\mathcal{S}$, if it is not connected to any other protein; we'll use pairs (A, i) of a protein $A \in \mathcal{N}$, and a site $i \in \mathfrak{s}(A)$ to refer to a precise site in a given solution.

Reactions are pairs of solutions $(\mathcal{L}, \mathcal{R})$, written $\mathcal{L} \longrightarrow \mathcal{R}$ and satisfying:

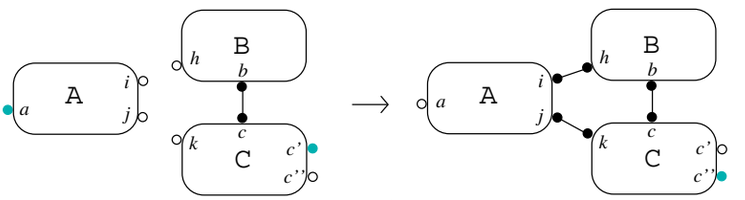
1. *virtual connectedness:* there exists a set of pairs of free sites $(A_1, a_1; B_1, b_1)$, $\ldots$, $(A_n, a_n; B_n, b_n)$ in $\mathcal{L}$, such that the corresponding edge set connects $\mathcal{L}$.
2. *components preservation:* $\mathcal{R}$ has the same proteins as $\mathcal{L}$.

We'll often refer to $\mathcal{L}$ and $\mathcal{R}$ as the *left hand side* and the *right hand side*, or sometimes also as the *reactants* and the *products* of the reaction.

Activations and Complexations. Two kinds of reactions deserve special attention: *activations*, when connections are left untouched and only states change, and *complexations* when the right hand side is connected. Both kinds intersect when the left hand side is connected. Here is an example of each kind:



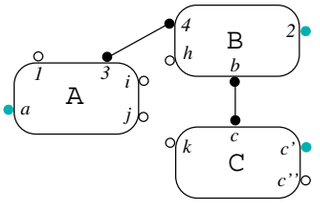
activation



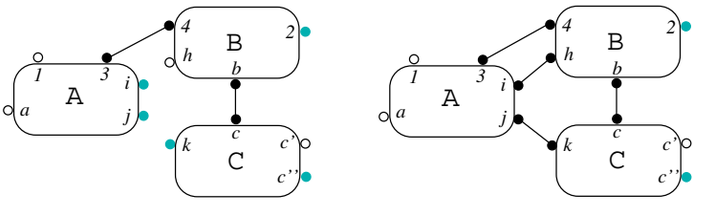
complexation

One observes that both left hand sides above are connectable through the pairs of visible sites $(A, i; B, h)$ and $(A, j; C, k)$, so the virtual connectedness condition is met. As is evidently the other condition. The free sites i, j, h and k are bound by the complexation rule, thus making "real" the virtual connection in the left hand side. Intuitively, virtual connectedness is a causality constraint saying that all reactions need to go through a temporary super complex for anything to happen. If not, then the reaction is not elementary in some sense and should be decomposed further. It might be interesting to study the restricted system with explicit temporary complexes but we don't do this here.

Firing a reaction. A reaction fires when a solution contains proteins *matching* the pattern in the left hand side of a rule. For instance, the following protein:



matches the patterns in the left hand sides of the two rules above. Observe that the convention in our reactions is to mention only *active* sites, *i.e.* sites which are tested and perhaps modified by the rule. Accordingly, a reaction affects only sites which are mentioned in the left hand side, all the other sites being kept intact. Our two example reactions produce the following proteins when applied to our example solution:



Given a set of reactions $\mathfrak{R}$, the firing conditions explained above generate an obvious transition relation, written $\longrightarrow$, over solutions.

About κ-calculus. In the κ-calculus, wirings were not visible because we took the more abstract point of view where complexation is an associative and commutative operator. We were making sure that such wirings were existing. Graphic κ extends κ, by allowing reactions to look up the wiring of complexes. One can embed easily the latter in the former.

3 Mobile κ

We show now that a graphic calculus based on binary interactions and with limited name-creation resources is strong enough to implement graphic κ. This puts an upper bound on how powerful graphic κ is as a process algebra.

The translation parallels the two direct encodings in π-calculus given in [4] for κ. Yet, in the present case more work has to be done, since dealing with generic graph rewriting rules needs a transactional mechanism to be set up. That additional complexity led us to introduce an intermediate graphical formalism, called *mobile* κ, which is abstract enough so that one can follow the encoding, and still low-level enough to be close to an implementation.

Indeed, if one seriously thinks of graphic κ as a programming language for biologists, then it should be implemented on some machine for running virtual experiments, and our target language provides a model of computation that we think is easily amenable to a distributed implementation. Nomadic π-calculus [16], for instance, could prove a hospitable platform for running an implementation of mobile κ.

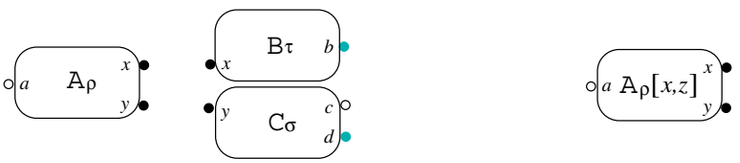
The development which follows is pretty technical and a biologist will wonder if it is worth the effort. We think it is. It certainly gives a sense of how our formal biological objects can be implemented. But more significantly it also raises the question, which we feel is largely independent of the actual formal language under consideration, of how biologically plausible a reduction of complex events to concurrent sequences of binary interactions is. Molecular combinatorics could actually rely only on binary interactions with failure mechanisms and complex events be only a working abstraction.

Before proceeding to the definition of our intermediate language, we must say that many variants are reasonable and that we choose here what seems more convenient for the specific purpose of embedding graphic κ.

Agents and Solutions. First we need two disjoint countable sets of names $\mathcal{N}_s$ and $\mathcal{N}_p$ for *private* and *public channels* (generically called channels and sometimes simply names). Symbols such as x, y, z, ... and a, b, c, ... will be used to represent respectively private and public channels. We still address agent names with the letters A, B, C, ...

Agents have *ports* associated to them, and the signature function of A, written $\mathfrak{s}_A$ maps A's ports to public channels in $\mathcal{N}_p$. These channels will be the means by which A interacts with its environment. They can be hidden, visible or even replaced with a private channel, but never by another public one.

In this name-based calculus, complexes can be built abstractly by replacing public channels with private ones, and by sharing these private channels between agents. The following picture on the left shows an example of an abstract complex where A is connected to B through x, and to C through y:



Agents also carry internal states, written ρ, σ, τ, ..., and private channel lists (see the picture above on the right). To summarize, and be a bit more formal, an agent is a tuple (A, θ, τ, l), written $A_\tau(\theta, l)$, with:

- A an agent name;
- θ a map from A's ports to the disjoint union $\mathcal{N}_p + \mathcal{N}_p + \mathcal{N}_s$, compatible with $\mathfrak{s}_A$ and describing how the ports are currently occupied;
- τ a state living in an unspecified finite state space;
- l a list over $\mathcal{N}_s$ of private channels carried by A.

Part of the information carried by θ tells whether the current public channels are visible or hidden. That part could be incorporated in the agent state τ, but we prefer it this way, to help in understanding the translation. We still refer to θ as the interface of A; by θ being compatible with $\mathfrak{s}_A$, or signature compatible, we mean that θ agrees with $\mathfrak{s}_A$ on those ports which it maps to public channels. Lists of private channels will be used to form dynamical transaction groups.

Solutions are simply multisets of agents. We'll only consider here solutions satisfying a well-formedness condition, namely that private channels occurring in the solution label at most two ports. One can recover in this case a graph structure on agents: each private channel labeling two ports represents an edge. The solution shown above is well-formed. (Agents can also share private channels in the lists they carry but this is a different matter).

Interactions. Mobile κ calculus interaction rules are written $\mathcal{L} \longrightarrow \mathcal{R}$ where both solutions $\mathcal{L}$ and $\mathcal{R}$ consist in the *same two agents*, with possibly different interfaces, internal states, and private channel lists.

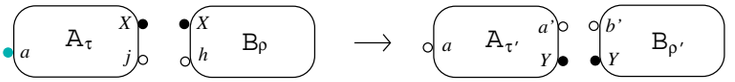
Rules have to match private channels which by definition can only be tested for equality. To handle this, we assume a countable set $\mathcal{V}$ of *meta-variables*, written X, Y, ... to match private channels, and define a further notion of *extended interfaces*, which are signature compatible one-one maps from agent ports to $\mathcal{N}_p + \mathcal{N}_p + \mathcal{V}$ and will be written Θ, Γ, ...

So a rule is a pair: $A_\tau(\Theta, l), B_\rho(\Gamma, k) \longrightarrow A_{\tau'}(\Theta', l'), B_{\rho'}(\Gamma', k')$ where:

- Θ, Θ' (resp. Γ, Γ') are extended interfaces for A (resp. B);
- Θ and Γ have either a common metavariable, or each a public name;
- l, l', k, k' are finite repetition-less lists of metavariables.

What the second condition really is saying is that when one applies the rule, A and B have to be either "connected" by a common private channel, or have public channels. We could have added a fourth condition restricting the transformations on the left hand side lists l and k. The reader will see that in the encoding below very limited transformations are actually used, namely propagation of lists from the left hand side and/or dynamic creation of new names.

An example of interaction rule:



As said, reactions can test equality on private channels. This is one crucial difference with graphic κ, where we asked that proteins had a given interface, and there were suitable edges connecting them. Here, we are enforcing that two different agents have common *labels* of ports. For instance, in the example above, the left hand side fires only if the two names in X position are private and equal. Similarly, left hand sides can test equality on the private channel lists. In addition, a rule may replace a public channel with a private one. That private channel can be taken from the agent private channels list, or be created anew by the reaction rule. Conversely ports occupied by private channels can be released *i.e.* substituted by the public channel returned by $\mathfrak{s}_A$ in a given state.

Firing an interaction rule. An interaction occurs when the solution contains two agents matching the pattern in the left hand side of the rule. Keeping with the notations introduced above, this means that there exists an injective renaming α from metavariables to private channels, and suitable partial maps θ and γ such that the solution contains the agents:

$$A_\tau(\Theta\alpha + \theta, l\alpha), B_\rho(\Gamma\alpha + \gamma, k\alpha).$$

The notation $i + i'$ is used for the disjoint union (or the coproduct) of i and i', so whenever we write $i + i'$ it is always understood that i' and i have disjoint domains. A such pair can then be rewritten as follows:

$$A_{\tau'}(\Theta'\alpha + \theta, l'\alpha), B_{\rho'}(\Gamma'\alpha + \gamma, k'\alpha),$$

where α is asked to map new meta-variables in the right hand side to actual fresh private names in the solution.

4 From Graphic κ to Mobile κ

To embed graphic κ it is enough to show how to encode a reaction $\rho = \mathcal{L}_\rho \to \mathcal{R}_\rho$. The idea is that some initial agents will decide they want to do that rule and try to gradually construct, if it is possible at all, an instance of $\mathcal{L}_\rho$. The protocol consists of two phases. The first one, called the "recruitment" phase, has two sets of rules: one set to send a signal around to recruit the necessary left hand side, and the other to send a signal back up to report the success to the initiators. All agents are checked and booked in the process, and when the initiators get the signal back, they are sure everybody is aboard and in the correct state. Then comes the second phase, or the "update" phase, led by a second series of propagations where the actual changes from $\mathcal{L}_\rho$ to $\mathcal{R}_\rho$ take places.

Flow graph and Initiators. We first need to define statically a structure on top of $\mathcal{L}_\rho$ that will allow the necessary routing of information in the target language. To this aim, we fix a connecting edge set, say X_ρ, which the virtual connectedness condition says always exists. Over the obtained connected graph $X_\rho + \mathcal{L}_\rho$, we can construct (not in a unique way but we choose one) an orientation such that the resulting directed graph $\mathcal{F}_\rho$ is acyclic. This is done, for instance, by constructing a depth-first tree spanning $X_\rho + \mathcal{L}_\rho$ (see [5]). Then, we choose a pair of proteins, say I and J, called the *initiators* of the reaction, which have to be neighbours in $\mathcal{F}_\rho$.

State space. Now that we have a directed acyclic graph (in brief, dag) to flow the information around, and initiators to get the reaction started, we need to describe the state space associated to ρ. For all elementary components $A(\theta)$ occurring in ρ, we define a state of the corresponding agent as $A^p_\rho(i, o, \theta, l)$ with:

- p an integer representing the current phase of the simulation;
- ρ an integer representing the reaction ρ;
- $i, o : \mathcal{F}_\rho \to \{0, 1, 2\}$ a pair of disjoint partial maps on A's neighbours in $\mathcal{F}_\rho$, with respective domains, the predecessors and successors of A in $\mathcal{F}_\rho$;
- θ a state of A in graphic κ recast as a mapping from sites in $\mathfrak{s}(A)$ to names in $\mathcal{N}_p + \mathcal{N}_p + \mathcal{N}_s$;
- l a list of private names.

We observe that since $\mathcal{F}_\rho$ is acyclic, the domains of i and o are indeed disjoint. These two maps i, o are used to represent the three possible states of contacts in the propagation algorithm below: 0 will stand for "not contacted yet", 1 for "contacted" and 2 for "contacted and answered back". The "in" map i is in charge of keeping track of the contacts received from above in the dag and the "out" map o of the contacts sent below. In the absence of failures these values will always increase until every value is set at 2 which means success.

We need a few notations to handle these maps. We write $\mathbf{0}$, $\mathbf{1}$ and $\mathbf{2}$ for the associated constant partial maps. Attention: all these maps are equal when their domain is taken empty, and this is used below for the two limit cases: initiators (when i is empty) and sinks (when o is empty). We also write $\mathbf{0}_A$, $\mathbf{1}_A$ and $\mathbf{2}_A$, when these constant maps have their domain consisting only in A.

First phase: Recruitment. To ease reading, we simply write X_j for the pairs (j, X) associating a port and a meta-variable, and in the same way we write a_j for the pair (j, a) associating a port and a public channel.

First I and J decide they're going to try to do ρ. There are two cases, depending on whether these are already bound or not (in which case the connection is created).

(start)
$$I(\theta + X_j), J(\gamma + X_{j'}) \longrightarrow I^1_\rho(\mathbf{0}, \mathbf{0}, \theta + X_j, R), J^1_\rho(\mathbf{0}, \mathbf{0}, \gamma + X_{j'}, R)$$
$$I(\theta + a_j), J(\gamma + b_{j'}) \longrightarrow I^1_\rho(\mathbf{0}, \mathbf{0}, \theta + X_j, R), J^1_\rho(\mathbf{0}, \mathbf{0}, \gamma + X_{j'}, R)$$

Then, for all edges (A, B) in $\mathcal{F}_\rho \setminus \{(I, J)\}$, we have the *top-down recruitment rules*, again in two flavours to distinguish edges already existing in $\mathcal{L}_\rho$ and edges to be created on the fly in X_ρ:

(first contact)
$$A^1_\rho(i, o + \mathbf{0}_B, X_j, R), B(\theta_B + X_{j'}) \longrightarrow A^1_\rho(i, o + \mathbf{1}_B, X_j, R), B^1_\rho(\mathbf{1}_A, \mathbf{0}, \theta_B + X_{j'}, R)$$
$$A^1_\rho(i, o + \mathbf{0}_B, a_j, R), B(\theta_B + b_{j'}) \longrightarrow A^1_\rho(i, o + \mathbf{1}_B, X_j, R), B^1_\rho(\mathbf{1}_A, \mathbf{0}, \theta_B + X_{j'}, R)$$

(further contacts)
$$A^1_\rho(i, o + \mathbf{0}_B, X_j, R), B^1_\rho(i' + \mathbf{0}_A, o', X_{j'}, R) \longrightarrow A^1_\rho(i, o + \mathbf{1}_B, X_j, R), B^1_\rho(i' + \mathbf{1}_A, o', X_{j'}, R)$$

$$A^1_\rho(i, o + \mathbf{0}_B, a_j, R), B^1_\rho(i' + \mathbf{0}_A, o', b_{j'}, R) \longrightarrow A^1_\rho(i, o + \mathbf{1}_B, X_j, R), B^1_\rho(i' + \mathbf{1}_A, o', X_{j'}, R)$$

Agents are gradually booked when receiving the "first contact" request. Since cross-talk with other concurrent applications of the *same* rule ρ are possible, the new name, R, created by I and J and which uniquely identifies the tentative reaction has to be passed around. The integer ρ would not be enough alone.

Take note that in the "further contacts" rule, we avoid contacting B twice by making sure we have a $\mathbf{0}_B$ on the left. Everyone will be contacted exactly once by each ancestor (in other words, of the two possible contact rules for a given $(A, B) \in \mathcal{F}_\rho$ exactly one will be used).

Remarkably, we also test exactly once, the first time they are recruited, whether the interface of recruited agents conforms with the prescription of ρ. For instance, in "first contact" we verify that A has a port to be connected with B, while we check much more about B's interface (the constraint θ_B in the left hand side). The reader may observe that this is not the case for "further contact" rules. Then comes the set of *backward response rules*, again for all edges $(A, B) \in \mathcal{F}_\rho \setminus \{(I, J)\}$:

$$\begin{array}{l}\text{(back to the top)}\\ B^1_\rho(i + \mathbf{1}_A, \mathbf{2}, X_{j'}, R), A^1_\rho(i', o' + \mathbf{1}_B, X_j, R) \longrightarrow\\ B^1_\rho(i + \mathbf{2}_A, \mathbf{2}, X_{j'}, R), A^1_\rho(i', o' + \mathbf{2}_B, X_j, R)\end{array}$$

The reader will protest at this point that nothing is ever creating a $\mathbf{2}$ with the rules given before. But $\mathbf{2} = \mathbf{0}$ when the domain is empty, so sinks do get $\mathbf{2}$ as soon as they're contacted, and they reply back with the 'back' rule.

Second phase: Update. And we go for the next phase:

$$\begin{array}{l}\text{(to the next phase)}\\ I^1_\rho(\mathbf{0}, \mathbf{2}, \theta + X_j, R), J^1_\rho(\mathbf{0}, \mathbf{2}, \gamma + X_{j'}, R) \longrightarrow I^2_\rho(\mathbf{0}, \mathbf{0}, \theta', R.\widetilde{Z}), J^2_\rho(\mathbf{0}, \mathbf{0}, \gamma', R.\widetilde{Z})\end{array}$$

When the "next phase" rule fires, everybody is in state $(\mathbf{2}, \mathbf{2})$ and success is guaranteed. As many different names are created as there are connections to be created in $\mathcal{R}_\rho$. These names will be propagated to the recruited agents to set up the right connections. Remark that we have to keep the reaction identifier R created in the recruitment phase since cross talk is still possible. Of course, name creation may be avoided in "next phase" if $\mathcal{F}_\rho$ is the complex in $\mathcal{R}_\rho$, because the recruitment phase has already set the right connections. In general, this is not the case.

It remains to sweep a last time through the graph with the *update rules* below to modify all booked agents as described in the right hand side $\mathcal{R}_\rho$. It is understood that the partial functions θ, θ' and γ, γ' are set acordingly. No back wave will be needed and therefore the i map is no longer used. The list $\widetilde{Z}$ will be carried along in the private channel list, to be used to map the appropriate newly bound ports to the correct private name chosen by the initiators.

$$\begin{array}{l}\text{(if } o \neq \mathbf{1})\\ A^2_\rho(\mathbf{0}, o + \mathbf{0}_B, \theta + X_j, R.\widetilde{Z}), B^1_\rho(\mathbf{2}, \mathbf{2}, \gamma + X_{j'}, R) \longrightarrow\\ A^2_\rho(\mathbf{0}, o + \mathbf{1}_B, \theta', R.\widetilde{Z}), B^2_\rho(\mathbf{0}, \mathbf{0}, \gamma', R.\widetilde{Z})\end{array}$$

(if $o = \mathbf{1}$, B is not a sink)
$A^2_\rho(\mathbf{0}, o + \mathbf{0}_B, \theta + X_j, R.\widetilde{Z}), B^1_\rho(\mathbf{2}, \mathbf{2}, \gamma + X_{j'}, R) \longrightarrow A(\theta'), B^2_\rho(\mathbf{0}, \mathbf{0}, \gamma', R.\widetilde{Z})$

(if $o = \mathbf{1}$, B is a sink)
$A^2_\rho(\mathbf{0}, o + \mathbf{0}_B, \theta + X_j, R.\widetilde{Z}), B^1_\rho(\mathbf{2}, \mathbf{2}, \gamma + X_{j'}, R) \longrightarrow A(\theta'), B(\gamma')$

Observe that A is freed asynchronously from the transaction, when he's done with all his successors. He then may participate to other interactions. Also observe that, in the update phase, cross connection checks are not needed.

Failures. The protocol has been presented without dealing with failures at all. Failures may occur due to competitions, or to the lack of proper reactants. There is an easy, theoretical way to deal with failures: duplicating the rules of phase1, by making them reversible (as we remarked, failures cannot occur in phase 2). A careful analysis of failures, with suitable rules propagating them could do better in terms of efficiency, but in this paper, we keep with the simplest policy. This simple solution has also the merits of introducing only a simulation cost (number of rules applied) which is linear in the size of the reaction original left hand side $\mathcal{L}_\rho$, which seems reasonable.

The formal correspondence. Systems of agents obtained by encoding rules as described above correctly may simulate any original system described in graphic κ. To state this in a satisfying way, we first define an observation relation on solutions, which relies on basic observations of free sites of proteins/agents.

Definition 1. *The* barb $\downarrow$ *is the least relation satisfying the rules below.*

1. *$A \downarrow a$ if the protein A has a free site a (respectively, if the agent A has a public channel a which is free);*
2. *if $\mathcal{C}$ is a complex, $\mathcal{C} \downarrow a$ if there is a component A of $\mathcal{C}$, such that $A \downarrow a$;*
3. *if $\mathcal{S}$ is a solution, $\mathcal{S} \downarrow a$ if there is a complex $\mathcal{C}$ of $\mathcal{S}$, such that $\mathcal{C} \downarrow a$.*

Next we write $[\![\cdot]\!]$ for the evident mapping from graphic κ solutions to mobile κ solutions that respects hidden and visible attributes and translates edges by means of private channels.

Proposition 1. *Let S be a solution in graphic κ. Then $\mathsf{S} \downarrow a$ iff $[\![\mathsf{S}]\!] \downarrow a$.*

The following theorem connects reductions in graphic κ to reductions in mobile κ. This theorem, in conjunction with Proposition 1, gives the correctness of our simulation with respect to the above observations.

Theorem 1. *Let S be a solution in graphic κ. If $\mathsf{S} \longrightarrow \mathsf{S}'$ then $[\![\mathsf{S}]\!] \longrightarrow^* [\![\mathsf{S}']\!]$.*

The converse correspondence does not hold because, in graphic κ, out of two competing reactions one will always succeed. In the finer-grained encoded system in mobile κ, graphic κ competitions may result into deadlocks. A correct failure recovery mechanism should support the converse direction of Theorem 1, but we will be content with this simple soundness result for the moment.

5 Conclusion

The purpose of this paper was to provide a formalism that could be a suitable modeling language allowing direct descriptions of molecular events, and offering a low-level distributed implementation. In so doing, we were able to relate our formalism to an intermediate language which is comparable to a process algebra.

Therefore, an important pending question is clearly whether techniques of process algebra can be adapted to answer biologically relevant questions about formalized protein networks.

Another interesting question is whether one can progress in the understanding of higher-level behaviours of such systems. Specifically, it might be rewarding to look after a rigourous definition of modules based on congruence bisimulation.

References

[1] Gérard Berry and Gérard Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96:217–248, 1992. 35

[2] A. Bouali, S. Gnesi, and S. Larosa. JACK: Just another concurrency kit. *Bulletin of the European Association for Theoretical Computer Science*, 54:207–224, October 1994. Technical Contributions. 35

[3] Marc Chiaverini and Vincent Danos. A core modeling language for the working molecular biologist. Communication to the International Workshop on Computational Methods in Systems Biology, 2003. 34

[4] Vincent Danos and Cosimo Laneve. Core formal molecular biology. To appear in Proceedings of ESOP 2003 - European Symposium on Programming, Lecture Notes in Computer Science, Springer Verlag, 2003. 34, 38

[5] Reinhard Diestel. *Graph Theory*. Springer, New-York, 2000. 41

[6] Steven Eker, Merrill Knapp, Keith Laderoute, Patrick Lincoln, José Meseguer, and Kemal Sonmez. Pathway logic: Symbolic analysis of biological signaling. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 400–412, January 2002. To appear. 35

[7] Cédric Fournet and Georges Gonthier. The reflexive chemical abstract machine and the join-calculus. In *23rd ACM Symposium on Principles of Programming Languages (POPL'96)*, 1996. 34, 35

[8] Kurt W. Kohn. Molecular interaction map of the mammalian cell cycle control and DNA repair systems. *Molecular Biology of the Cell*, n. 10:2703–2734, 1999. 34

[9] H. Lin. An interactive proof tool for process algebras. In Alain Finkel and Matthias Jantzen, editors, *Proceedings of Symposion on Theoretical Aspects of Computer Science (STACS '92)*, volume 577 of *LNCS*, pages 617–618, Berlin, Germany, February 1992. Springer. 35

[10] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes I and II. *Information and Computation*, 100:1–41, 42–78, 1992. 34, 35

[11] Masao Nagasaki, Shuichi Onami, Satoru Miyano, and Hiroaki Kitano. Bio-calculus: Its concept and molecular interaction. *Genome Informatics*, 10:133–143, 1999. 34

[12] Corrado Priami, Aviv Regev, Ehud Shapiro, and William Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 2001. in press. 34

[13] Aviv Regev and Ehud Shapiro. Cells as computation. *Nature*, 419, September 2002. 34

[14] Aviv Regev, William Silverman, and Ehud Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. In R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein, editors, *Pacific Symposium on Biocomputing*, volume 6, pages 459–470, Singapore, 2001. World Scientific Press. 34

[15] Björn Victor and Faron Moller. The Mobility Workbench — a tool for the π-calculus. In David Dill, editor, *CAV'94: Computer Aided Verification*, volume 818 of *Lecture Notes in Computer Science*, pages 428–440. Springer-Verlag, 1994. 35

[16] Pawel T. Wojciechowski and Peter Sewell. Nomadic Pict: Language and infrastructure design for mobile agents. *IEEE Concurrency*, 8(2):42–52, April/June 2000. 39

Contribution of Computational Tree Logic to Biological Regulatory Networks: Example from Pseudomonas Aeruginosa

Sabine Peres[1] and Jean-Paul Comet[2]

[1] Physiologie Mitochondriale, INSERM EMI 9929,
146, rue Léo Saignat 33076 Bordeaux cedex, France
sabine_peres@yahoo.fr
[2] LaMI, Tour Evry 2,
523 Place des terrasses de l'agora 91000 Evry Cedex France
comet@lami.univ-evry.fr

Abstract. The Computational Tree Logic allows us to express some properties of genetic regulatory networks. These systems are studied using the feedback circuits evolved by René Thomas which constitute the semantic of our formal approach. We illustrate this formal language with the system of mucus production in *pseudomonas aeruginosa*, which is a mucoid bacteria that plays an important role in the cystic fibrosis. With the Thomas' theory, we could wonder if the mucoid state could be a steady state alternative to the non-mucoid state. We would like to know whether it is possible to have a recurrent mucoid state. Model-checking allows us to prove that the formula which expresses this property is satisfied by certain models. Moreover, using this formal language we can propose scenarii for confronting the model to experimentation.

1 Introduction

Biological experiments are motivated by hypotheses that biologists suggest. These hypotheses are more and more complex and it would be useful to express them in a form which can be handled by computer. All models which are planed to be proposed possess some properties corresponding to hypotheses previously mentioned [B02]. But a model which validates the hypotheses is not always the true model, it may fail with biological experiments. We have to call into question and to improve it. Thus, a model which agrees to biologists' experiments, ties up with the true model.

We present a formal language allowing to express in a modal logic [R00] some properties of a genetic regulatory network. Firstly, we present a formal description of the regulatory networks which are the semantic of this formal language. Secondly, we explain how model-checking is used to eliminate a majority of models. The model-checking is an automatic method to check if a model satisfies a logical formula which correspond for example to a temporal behaviour. To validate the retained models, a plan of experiment is proposed. Finally, we

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 47–56, 2003.

illustrate this formal language using the system of mucus production in *Pseudomonas aeruginosa*, which is a mucoid bacteria that plays an important role in the cystic fibrosis [G01].

2 Formal Description

Biological regulatory networks describe the interactions between genes and adjust production rate of system elements. Simulating them by a computer in order to predict their behaviour requires a formal description. The logical analysis developped by Thomas [TTK95-1, TT95-2] allows to study this type of models. It constitutes the basis of the purposed frame to study the regulatory networks, represented by an oriented graph (graph of interactions) of which the vertices represent the variables of the system (genes, proteins ...) and the edges represent the interactions between the variables. When the interactions form an oriented circuit, this constitutes a feedback loop. These loops regulate the production of the system variables and the behaviour of a specific variable depends on the possible paths.

Generally the interactions are described by differential equations and their solutions are sigmoid functions in shape. These functions determine values called thresholds, which correspond to the minimal concentration rate necessary for interaction between two variables. One approximates the sigmoids with step functions. In such a simplification, the number and location of steady states are preserved [ST93]. The expression level of each variable is discretised according to threshold values. But, there is no reason for all the thresholds of one variable to be equal because a variable does not act on the others with the same concentration. The interactions are ordered using the associated thresholds and the n^{th} threshold of a variable x is named s_x^n.

For example, let us suppose that a variable x acts on two variables x and y. The expression level of the different target variables in function of the expression level of x are sigmoids (figure 1). This figure shows three behaviours of x. For each of them, a specific discretised level of x is associated.

- x=0 : x acts on no variable
- x=1 : x acts only on y
- x=2 : x acts on y and x.

Generally, each edge $m \rightarrow n$ of the interaction graph is labelled with one of these thresholds and with the sign + if m has a positive influence on n and with the sign − if m represses n (graph of thresholds). For the same example, if x has a positive influence on y and on itself, the graph of interactions is presented in figure 2.

2.1 Graphs of States

To describe the evolution of the system with r variables, one has to make explicit the level reached by each variable depending on the presence (resp. absence) of

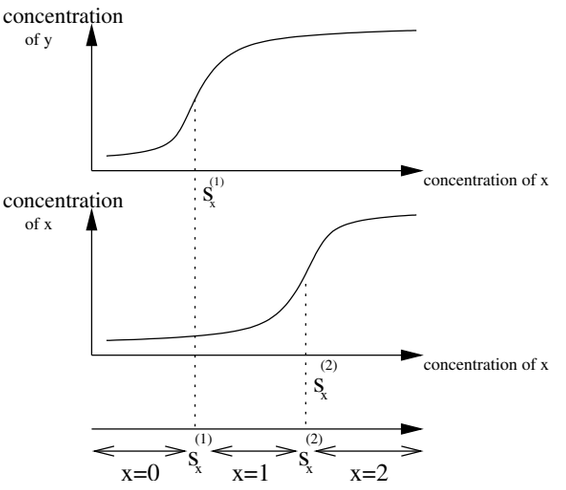


Fig. 1. Discretisation of the concentration of x

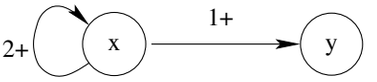


Fig. 2. Graph of interactions

variables which activate it (resp. which repress it). These values are represented by a function $\mathcal{K}$. This function takes two arguments as input (a target variable and the set of its activators) and returns the value towards which the target variable tends in presence of activators. This function can be determined using the same approach than Thomas [TT95-2, ST93], who used the different feedback loops and their functionality to find the parameters of the system. A positive feedback loop is functional if it ensures multi-stationarity and a negative one is functional if it ensures homeostasis. A loop is functional if the state located at the thresholds of the loop (*characteristic state*) is steady. One looks for the values of function $\mathcal{K}$ that make loops functional, obtains many contraints on the function $\mathcal{K}$ and can deduce its values (several solutions are possible).

So graphs of states are constructed :

- vertices are the r-uplets corresponding to each possible combination of values of the r variables
- the oriented edges ($n_1 \longrightarrow m_1$, $n_1 \longrightarrow m_2$, ...) give the possible futures of a state n_1.

The interactions are considered to be asynchronous : a variable does not act simultaneously on the others. Thus, the future states m_1, m_2,... of n_1 changes

the value of one and only one variable in one step. The obtained level traces are the possible paths of these graphs which shows the possible behaviours of the system when time passes.

2.2 Model-Checking

Biologists could frequently propose hypotheses when studying a real phenomenon. These hypotheses often express a temporal property of the entire system. For example, we could wonder in figure 2: if at a given time x has a positive influence on y ($x = 1$) then at the next time x has always a influence on y ($x = 1$). When a set of models is proposed, it is necessary to keep those which validate the hypotheses. We want to check if a model fulfills the temporal properties associated to the hypothesis.

In this aim a formal language is introduced, based on the temporal logic [E90] which is adapted to this kind of models. It allows to express temporal properties of the system. The temporal logic chosen here is CTL[1][CE81, EH82]. It is a tree structure which can define several futures. It can be applied to our model because all the possible outcoming edges from a state of the graph define a different future. The precedent hypothesis on figure 2 can be translated in CTL:

$$(x = 1) \Rightarrow AX(x = 1)$$

where

- $\Rightarrow$ is the logical connective of implication
- AX is a temporal connective which means the next time in all possible futures.

The model-checking [R00] proves that a given system of finite states, satisfies or does not satisfy a temporal formula. It allows to find out properties of the system, for example the fact that the system tends to a given state. The algorithm takes a model, here a specific graph of states, and a formula ϕ as input and returns all the states which satisfy ϕ. The algorithm describes in [R00] can be summarized as follow:

1. Translate the fomula ϕ with the six chosen connectives (see Appendix).
2. For all sub-formula ψ of ϕ, determine the states which are labelled with ψ.
3. Return all the states which are labelled with ϕ.

There are several implementations of the model-checking algorithm. At the moment, the regulatory subnetworks only contain few variables [MTA99, ST2001], so a simple ad hoc checker is sufficient.

A model satisfies a formula if all the states which belong to the model satisfy the formula. So, if the algorithm returns all the states of the model, the model satisfies the hypothesis. The next section shows how we have used this method to select models of the bacteria *pseudomonas aeruginosa* which satisfy a temporal hypothesis emitted by biologists.

[1] For Computation Tree Logic, see the connective definition in Appendix.

3 Results

Pseudomonas aeruginosa is a bacteria which secretes mucus in lung affected by cystic fibrosis. The mucus production increases the respiratory deficiency of the patients. But in a healthy lung, there is no production of mucus. Moreover, if one isolates a population of cells from a sick lung and if one puts it in a healthy environment, the mucus production can persist or resume progressively its non-mucoid phenotype after numerous generations. Biologists have shown that when the production persists, mucoid strain is generated by mutation. But, this model does not explain why it may happen that the mucus production stops.

The main regulator for the mucus production, AlgU, supervises an operon which is made up of 4 genes among which one codes for a protein that is a repressor of AlgU[2]. Moreover AlgU favors its own synthesis, which constitutes a positive feedback loop. This simplified model takes into account only interactions which are involved in feedback circuits (figure 3).

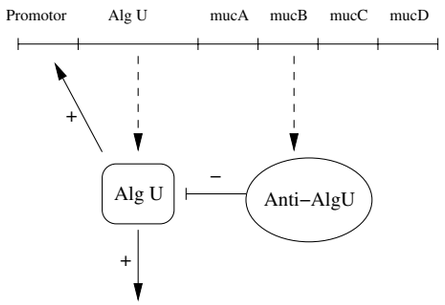


Fig. 3. Interactions of genes involved in the mucus production in P. aeruginosa

3.1 Hypothesis and Modelisation of the Bacteria

Thomas' theory [TTK95-1, TT95-2] allows us to assert that a positive feedback loop is a necessary condition for multi-stationarity. Moreover, epigenetic modifications could be a consequence of multi-stationarity [TK01]. Epigenetic modifications are phenotypic changes (transmitted from a cell to its progeny) without genetic or environmental modifications [G01].

This is a reason why biologists researchers [G01] wonder whether the mucoid state is a steady state alternative to the non-mucoid state, which is activated by an external signal and self-maintained by a high concentration rate of AlgU. In other words, they wonder if the mucus production is an epigenetic phenomenon.

[2] See for details [G01].

The model is represented by an oriented graph with two vertices x and y (Figure 4). x represents gene AlgU, y represents the inhibitor genes of gene AlgU. The edges depict:

- $x \rightarrow x$: self-maintenance of variable AlgU,
- $y \rightarrow x$: inhibitor effect on gene AlgU,
- $x \rightarrow y$: influence of AlgU on its own inhibitors.

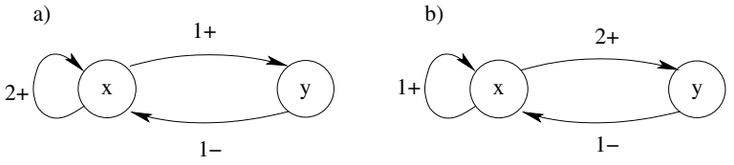


Fig. 4. Two possible graphs of thresholds

y acts on one variable, so the edge $y \longrightarrow x$ is labelled with 1. For biological reasons, one knows that the mucus production occurs when x is over its second threshold value (x=2). But we do not know if x acts on y at a lower threshold than it acts on x or at a upper one. So there are two possible graphs of thresholds (Figures 4a & 4b).

The state table of figure 4a is deduced. (X,Y) is the state towards which (x,y) tends. $\mathcal{K}_x\{y\}$ is the future value of x when y does not repress x in other word y is considered as an activator.

State table of figure 4a:

x y	X	Y
0 0	$\mathcal{K}_x\{y\}$	$\mathcal{K}_y\{\}$
0 1	$\mathcal{K}_x\{\}$	$\mathcal{K}_y\{\}$
1 0	$\mathcal{K}_x\{y\}$	$\mathcal{K}_y\{x\}$
1 1	$\mathcal{K}_x\{\}$	$\mathcal{K}_y\{x\}$
2 0	$\mathcal{K}_x\{x,y\}$	$\mathcal{K}_y\{x\}$
2 1	$\mathcal{K}_x\{x\}$	$\mathcal{K}_y\{x\}$

State table of figure 4b:

x y	X	Y
0 0	$\mathcal{K}_x\{y\}$	$\mathcal{K}_y\{\}$
0 1	$\mathcal{K}_x\{\}$	$\mathcal{K}_y\{\}$
1 0	$\mathcal{K}_x\{x,y\}$	$\mathcal{K}_y\{\}$
1 1	$\mathcal{K}_x\{x\}$	$\mathcal{K}_y\{\}$
2 0	$\mathcal{K}_x\{x,y\}$	$\mathcal{K}_y\{x\}$
2 1	$\mathcal{K}_x\{x\}$	$\mathcal{K}_y\{x\}$

3.2 Models Fulfilling the Formula

The values of $\mathcal{K}$ are calculated with the Thomas' theory. Several solutions are possible. The figure 5 is a possible state graph of figure 4a [3]. For $x = 1$ and $y = 0$, $X = \mathcal{K}_x\{y\} = 0$ and $Y = \mathcal{K}_y\{x\} = 1$. Because the state 10 tends to 01, we write in the state 10 the values 01. As the interactions are asynchronous, only one variable can change at a given moment. So the possible future states of 10 are 00 and 11.

[3] One could construct the state graph of the figure 4b using the same method.

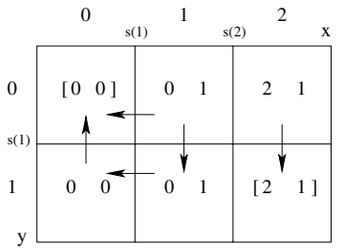


Fig. 5. Possible state graph of figure 4a

27 possible functions $\mathcal{K}$ are found, with the hypothesis that there is a functional positive feedbak loop in the system which is a necessary condition for multi-stationarity. So 27 state graphs can be constructed.

Two different cycles cohabit in the system: one positive feedback loop which imposes two steady states and a negative feedback loop which makes the system switch to one of the steady states according to the values of $\mathcal{K}$.

The aim is to show that the mucoid state can be reached without mutation, in other words that the loop $x \rightarrow x$ can be functional in the presence of y and so that it is possible, with the previous model, to obtain a recurrent state where $x = 2$. The following formula expresses the previous property :

$$(x = 2) \Longrightarrow AX\ AF(x = 2)$$

where $AX\ AF(\phi)$ means that for all possible futures (excluding the present) ϕ will be satisfied at a given time. The formula means that if the bacteria products mucus at a given time, then in a future time it will again product mucus. We want to show that for certain values of $\mathcal{K}$ which can be computed, $x = 2$ is a recurrent property.

We implemented a program which takes a function $\mathcal{K}$, a graph of interactions and a temporal formula (written in CTL) and returns true if the model satisfies the formula and false if not. The formula tested on the 27 models highlights 14 models which satisfy the hypothesis.

For example, the model which is represented in the figure 5 fulfills the formula which expresses the mucus production as an epigenetic modification.

The fact that these 14 models don't reject the epigenetic hypothesis can open new therapeutics in prospects. We have two classes of models : 14 models fulfill the hypothesis and 13 don't fulfill it. From this language, we can propose scenarii for confronting the models to experimentation. After experiment, we hope to be able to reject a class of models.

3.3 Experimentation

The experiment plan is deduced from the formula written in temporal logic. The following scenario allows one to test the property of the recurrent mucoid state :

pulsing x to the value 2 by using an external signal, waiting a period of time in order to pass the transitory phase due to the pulse, and measuring whether x is above its second threshold. If, for a period of time, the state of the bacteria is mucoid, then the 14 models are considered as right and the others are rejected. But, if the experiment fails we cannot reject any class of model. If the experiment fails, it means that the mucus production has not been observed but it does not mean that the mucus production will never occur.

3.4 Limitations

The formal language does not take into account the external elements of the graph of figure 3 which is a subgraph of the graph with all the variables of the organism. Having neglected the edges outgoing from the subgraph has no serious consequences on our study since one is only interested on the subsystem supervising the mucus production. On the other hand, neglecting all the incoming edges could lead to an excessive simplification. If there are some edges regulating x and y whose influence does not change in the future, the only consequence of having extracted a subgraph is to shift the different thresholds associated to variables x and y. The system will have some other values for these thresholds, possibly some other values for $\mathcal{K}$, the steady states will not necessarily be the same, but the formula to be proved will remain identical. Only one case is a real problem : if there are some external regulators of x and y that have an influence which depends on time, the language will not allow one to translate the real behaviour of the system. The current study makes the hypothesis that these influences are negligible.

4 Conclusion

The presented method highlights the exixtence of models which are consistent with biologists' hypothesis. Thomas' theory is used to construct all the models and the temporal logic CTL is used to express a temporal property we want the system to have. Model-checking, a classical method in computer science, eliminates the models which do not fulfill the wished behaviours.

This method is applied to the production of mucus in *pseudomonas aeruginosa*. Thomas' theory on feedback loops is used to model some gene interactions of this bacteria. This theory says that a positive feedback loop is a necessary condition to have multi-stationarity. So to explain the phenomenon as an epigenetic shift, we only consider models that contain a functional positive feedback loop.

The hypothesis expressing the mucus production as an epigenetic modification would be in CTL : $(x = 2) \Longrightarrow AX\ AF(x = 2)$. 27 models are found by Thomas' theory, and 14 are kept by model-checking. But these models have to be validated by biological experiments. Thus, according to the CTL formula we propose a plan of experiment for confronting the model.

1. pulsing x to the value 2 by using an external signal

2. waiting a period of time in order to pass the transitory phase due to the pulse
3. measuring whether x is above its second threshold.

We do not have the result of the experiment yet, but if the experiment says the 14 models as right, we could have new points of view on the mucus production of this bacteria and on the therapeutic treatments. For example, instead of killing the bacteria, it would become possible to prevent a phenotypic shift to pathogeny [G01].

Acknowledgments

The research has led at the LaMI (LAboratoire des Méthodes Informatique) in Evry, France. We would like to thank Gilles Bernot for stimulating discussions and Janine Guespin for generous support.

Appendix: Note on Computation Tree Logic

Definition 1. *CTL formula is defined inductively with:*

- *a finite state of propositional variables:* $\{p_i\}$.
- *logical connectives:* $\bot, \top, \neg, \wedge, \vee, \Rightarrow$.
- *temporal connectives:* AX, EX, AG, EG, AU, EU, AF, EF.
- *rules of formation:*
  - p_i, $\bot$ *and* $\top$ *are formulas.*
  - *if* ϕ *and* ψ *are formulas, then* $(\neg\phi), (\phi \wedge \psi), (\phi \vee \psi), (\phi \Rightarrow \psi), AX\phi, EX\phi, A[\phi U \psi], E[\phi U \psi], AG\phi, EG\phi, AF\phi, EF\phi$ *are formulas.*

The logical connectives are the classical ones: *false*, *true*, *not*, *and*, *or*, *implication*. All the temporal connectives are pairs of symbols. The first element of the pair is A or E. The second one is X, F, G or U.
Meaning of the Connectives:

- A : along All paths
- E : along at least one path (there Exist)
- X : neXt state
- F : some Future state
- G : all future states (Globally)
- U : Until

All the CTL connectives are equivalent to a combination of a set of six of them, for example $\{\bot, \neg, \wedge, AF, EU, EX\}$. So, all the formulas can be written with this set of connectives using the following equivalences.

Equivalences between CTL Formulas [R00]:

1. $\neg AF\phi \equiv EG\neg\phi$
2. $\neg EF\phi \equiv AG\neg\phi$
3. $\neg AX\phi \equiv EX\neg\phi$
4. $AF\phi \equiv A[\top U\phi]$
5. $EF\phi \equiv E[\top U\phi]$
6. $A[pUq] \equiv \neg(E[\neg qU(\neg p \wedge \neg q)] \vee EG\neg q)$

References

[B02] G. Bernot, J. Guespin-Michel, A. Zemirline, J.-P. Comet, F. Delaplace, P. Ballet, and P. Amar. Modelling, observability and experiment: a case study, positive feedback loop in a genetic regulatory network. *Research report, LaMI, Univ. Evry, France*, 2002. 47

[CE81] E. M. Clarke and E. A. Emerson. Design and syntheses of synchronization skeletons using branching time temporal logic. In *Proc. Logics of Programs Workshop, Yorktown Heights, New York*, volume 131 of *LNCS*, pages 52–71. Springer, May 1981. 50

[E90] E. A. Emerson. *Handbook of theoretical computer science, Volume B: formal models and semantics*, chapter Temporal and modal logic, pages 995–1072. MIT Press, 1990. 50

[EH82] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, San Francisco, California*, pages 169–180, 5-7 May 1982. 50

[G01] J. Guespin-Michel and M. Kaufman. Positive feedback circuits and adaptive regulation in bacteria. *Acta Biotheoretica*, 49, 2001. 48, 51, 55

[R00] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge University Press, 2000. 47, 50, 55

[ST93] E. H. Snoussi and R. Thomas. Logical identification of all steady states: the concept of feedback loop characteristic states. *Bulletin of Mathematical Biology*, 1993. 48, 49

[TK01] R. Thomas and M. Kaufman. Multistationarity, the basis of cell differentiation and memory. i structural conditions of multistationarity and other nontrivial behavior. *Chaos*, 2001. 51

[TT95-2] R. Thomas and D. Thieffry. Dynamical behaviours of regulatory networks – II. immunity control in bactériophage lambda. *Bulletin of Mathematical Biology*, 57(2):277–297, 1995. 48, 49, 51

[TTK95-1] R. Thomas, D. Thieffry, and M. Kaufman. Dynamical behaviours of regulatory networks – I. biological role of feedback loops and pratical use of the concept of feedback loop. *Bulletin of Mathematical Biology*, 57(2):247–276, 1995. 48, 51

[MTA99] L. Mendoza, D. Thieffry, and E. R. Alvarez-Buylla. Genetic control of flower morphogenesis in arabidopsis thaliana: a logical analysis. *Bioinformatics*, 15(7/8):593–606, 1999. 50

[ST2001] L. Sanchez and D. Thieffry. A logical analysis of the drosophila gap-gene system. *J. Theor. Biol.*, 211:115–141, 2001. 50

Modeling Cellular Behavior with Hybrid Automata: Bisimulation and Collapsing

Marco Antoniotti[1], Bhubaneswar Mishra[1,2], Carla Piazza[3], Alberto Policriti[4], and Marta Simeoni[3]

[1] Courant Institute of Mathematical Science, NYU
New York, USA
marcoxa@cs.nyu.edu
[2] Watson School of Biological Sciences, Cold Spring Harbor
New York, USA
mishra@cs.nyu.edu
[3] Dept. of Computer Science, University Ca' Foscary of Venezia
Venezia, Italy
{piazza,simeoni}@dsi.unive.it
[4] Dept. of Mathematics and Computer Science, University of Udine
Udine, Italy
policrit@dimi.uniud.it

Abstract. Many biological systems can be modeled using systems of ordinary differential algebraic equations (e.g., S-systems), thus allowing the study of their solutions and behavior automatically with suitable software tools (e.g., PLAS, Octave/Matlab^{tm}). Usually, numerical solutions (*traces* or *trajectories*) for appropriate initial conditions are analyzed in order to infer significant properties of the biological systems under study. When several variables are involved and the traces span over a long interval of time, the analysis phase necessitates automation in a scalable and efficient manner. Earlier, we have advocated and experimented with the use of automata and temporal logics for this purpose (XS-systems and Simpathica) and here we continue our investigation more deeply.
We propose the use of *hybrid automata* and we discuss the use of the notions of *bisimulation* and *collapsing* for a "qualitative" analysis of the temporal evolution of biological systems. As compared with our previous proposal, hybrid automata allow maintenance of more information about the differential equations (S-system) than standard automata. The use of the notion of bisimulation in the definition of the *projection operation* (restrictions to a subset of "interesting" variables) makes possible to work with reduced automata satisfying the same formulae as the initial ones. Finally, the notion of collapsing is introduced to move toward still simpler and equivalent automata taming the complexity of the automata whose number of states depends on the level of approximation allowed.

1 Introduction

The emerging fields of *system biology* [21], and its sister field of bioinformatics, focuses on creating a finely detailed and "mechanistic" picture of biology

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 57–74, 2003.

at the cellular level by combining the part-lists (genes, regulatory sequences, other objects from an annotated genome, and known metabolic pathways), with observations of both transcriptional states of a cell (using micro-arrays) and translational states of the cell (using proteomics tools).

Recently, the need has arisen for more and more sophisticated and mathematically well founded computational tools capable to analyze the models that are and will be at the core of *system biology*. Such computational models should be implemented in software packages faithfully while exploiting the potential trade-offs among usability, accuracy, and scalability dealing with large amounts of data. The work described in this paper is part of a much larger project still in progress, and thus only provides a partial and evolving picture of a new paradigm for computational biology.

Consider the following scenario. A biologist is trying to test a set of hypotheses against a corpus of data produced in very different ways by several *in vitro*, *in vivo*, and *in silico* experiments. The system the biologist is considering may be a piece of a *pathway* for a given organism. The biologist can access the following pieces of information:

- raw data stored somewhere about the temporal evolution of the biological system; this data may have been previously collected by *observing* an in vivo or an in vitro system, or by *simulating* the system in silico;
- some mathematical model of the biological system[1].

The biologist will want to formulate *queries* about the evolution encoded in the data sets. For example, he/she may ask: *will the system reach a "steady state"?*, or *will an increase in the level of a certain protein activate the transcription of another?* Clearly the set of numerical *traces* of very complex systems rapidly becomes unwieldy to wade through for increasingly larger numbers of variables.

Eventually, many of these models will be available in large public databases (e.g. [6, 18, 19, 20, 28, 23]) and it is not inconceivable to foresee a biologist to test some hypotheses *in silico* before setting up expensive wet-lab experiments. The biologist will mix and match several models and raw data coming from the public databases and will produce large datasets to be analyzed.

To address this problem, we have proposed a set of theoretical and practical tools, XS-systems and Simpathica, that allow the biologist to formulate such queries in a simple way [3, 4, 5]. The computational tool Simpathica derives its expressiveness, flexibility, and power by integrating in a novel manner many commonly available tools from numerical analysis, symbolic computation, temporal logic, model-checking, and visualization. In particular, an *automaton-based* semantics of the temporal evolution of complex biochemical reactions starting from their representations as sets of differential equations is introduced. Then *propositional temporal logic* is used to qualitatively reason about the systems.

[1] We note that simulating a system *in silico* actually requires a mathematical model. However, we want to consider the case when such mathematical model is unavailable to both the biologist and the software system.

Notice that here the adverb qualitatively does not mean that we completely abstract away from the quantities of the substances involved in the system, but that we concentrate on particular properties of the system, e.g. *will a certain protein reach level* 0.5*?*

In this paper we continue our research on the computational models at the core of our approach. We bring in several techniques from the fields of Verification, Logic and Control Theory, while maintaining a trade off between the need to manipulate large sets of incomplete data[2] and the requirements arising from the needs to provide a mathematically well founded system. In particular, we propose the use of *hybrid automata* together with the notions of *bisimulation* and *collapsing*. Hybrid automata are equipped with states embodying time-flow, initial and final conditions, and therefore allow maintenance of more information about the differential equations (S-system). The use of the notion of bisimulation in the definition of the *projection operation* (restrictions to a subset of "interesting" variables) provides a way to introduce *reduced* automata satisfying the same formulae as the initial ones. Notice that the idea and potential behind this use of the notion of bisimulation can be exploited as fruitfully as here also in the context of standard automata. Finally, the notion of collapsing is introduced to improve along the direction of a qualitative study of the automata extracted from the analysis of traces as well as to tame its complexity of the automata (whose size depends on the level of approximation allowed).

The cellular and biochemical processes analyzed using XS-systems and Simpathica [4, 3] provide a large set of application examples for the framework we present here. In order to motivate the choices of our modeling framework, we give some details about one of such examples, the *repressilator system* described by Elowitz and Leibler in [13].

We conclude pointing out that the analysis presented in this paper is not limited to XS-Systems, but could be extended to more general hybrid system models.

2 Related Works

A survey on the different approaches for modeling and simulating genetic regulatory systems can be found in [11]: the author takes into consideration different mathematical methods (including ordinary and partial differential equations, qualitative differential equations and others) and evaluates their relative strengths and weaknesses.

The problem of constructing an automaton from a given mathematical model of a general dynamical system has been previously considered in the literature. In particular, it has been investigated by Brockett in [7]: our approach in [4] is certainly more focussed, since it deals with specific mathematical models (i.e. S-systems). Here we take the distance from pure discrete models, and propose

[2] The range of parameters tested in each *in vitro* and *in vivo* experiment is just too large, and coping with this combinatorial effects is a subject of research (c.f.r. [26]).

the use of hybrid automata instead of standard automata, with the aim of taking advantage of their continuous component for allowing quantitative besides qualitative reasoning.

The use of hybrid automata for the modeling and simulation of biomolecular networks has been proposed also by Alur et. al. in [1]. In that paper the discrete component of an hybrid automaton is used to switch between two different behaviors (models) of the considered biological system, (for example) depending on the concentration of the involved molecules. In our case, the continuous component is used to regulate the permanence on a given state depending on the values of the involved variables (reactants), and the discrete component is used for enabling the transition to another state.

Moreover, in [1], as well as in other formalisms which model biochemical systems (e.g., [25, 10]), the notion of concurrency is explicitly used since the involved reactants are represented as processes running in parallel. In our case this kind of concurrency becomes implicit since in all the states of the automaton representing an S-system the values of all the reactants and their evolutions are represented.

3 Setting the Context

3.1 S-systems

We begin presenting the basic definitions and properties of S-systems. The definition of S-systems we use in this paper is basically the one presented in [27] augmented with a set of *algebraic constraints*. The constraints characterize the conditions under which a given set of equations is derived from a set of maps (see also [4]).

Definition 1 (S-system). *An S-system is a quadruple* $S = (DV, IV, DE, C)$ *where:*

- $DV = \{X_1, \ldots, X_n\}$ *is a finite non empty set of* dependent variables *ranging over the domains* $D_1, \ldots, D_n$*, respectively;*
- $IV = \{X_{n+1}, \ldots, X_{n+m}\}$ *is a finite set of* independent variables *ranging over the domains* $D_{n+1}, \ldots, D_{n+m}$*, respectively;*
- DE *is a set of* differential equations*, one for each dependent variable, of the form*

$$\dot{X}_i = \alpha_i \prod_{j=1}^{n+m} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{n+m} X_j^{h_{ij}}$$

  with $\alpha_i, \beta_i \geq 0$ *called* rate constants*;*
- C *is a set of* algebraic constraints *of the form*

$$C_j(X_1, \ldots, X_{n+m}) = \sum(\gamma_j \prod_{k=1}^{n+m} X_k^{f_{jk}}) = 0$$

  with γ_j *called* rate constraints*.*

In what follows we use $\boldsymbol{X}$ to denote the vector $\langle X_1, \ldots, X_n, X_{n+1}, \ldots, X_{n+m}\rangle$ of variables and $\boldsymbol{d}$ ($\boldsymbol{a}$, $\boldsymbol{b}$,...) to denote the vector $\langle d_1, \ldots, d_n, d_{n+1}, \ldots, d_{n+m}\rangle \in D_1 \times \ldots \times D_n \times D_{n+1} \times \ldots \times D_{n+m}$ of values. Similarly given a set of variables $U = \{X_{U_1}, \ldots, X_{U_u}\} \subseteq DV \cup IV$ we use $\boldsymbol{X} \restriction U$ to denote the vector of variables of U, while $\boldsymbol{d} \restriction U$ denotes the vector of values $\langle d_{U_1}, \ldots, d_{U_u}\rangle \in D_{U_1} \times \ldots \times D_{U_u}$.

The dynamic behavior of an S-system can be simulated by computing the approximate values of its variables at different time instants (*traces*). To determine a trace of an S-system it is necessary to fix an initial time (t_0), the values of the variables at the initial time ($\boldsymbol{X}(t_0)$), a final time (t_f), and a step (s).

Definition 2 (Trace). *Let $S = (DV, IV, DE, C)$ be an S-system. Let $\boldsymbol{f}(t) = \langle f_1(t), \ldots, f_{n+m}(t)\rangle$ be a (approximated) solution for the S-system S in the time interval $[t_0, t_f]$ starting with initial values $\boldsymbol{X}(t_0)$ in t_0. Let $s > 0$ be a time step such that $t_f = t_0 + j * s$. The sequence of vectors of values*

$$tr(S, t_0, \boldsymbol{X}(t_0), s, t_f) = \langle \boldsymbol{f}(t_0), \boldsymbol{f}(t_0 + s), \ldots, \boldsymbol{f}(t_0 + (j-1) * s), \boldsymbol{f}(t_0 + j * s)\rangle$$

is a trace *of S. When we are not interested in the parameters defining the trace we use the notation tr.*

Notice that $\boldsymbol{f}(t_0) = \boldsymbol{X}(t_0)$. A trace is nothing but a sequence of values of $D_1 \times \ldots \times D_{n+m}$ representing a solution of the system in the time instants $t_0, t_0 + s, \ldots, t_0 + j * s$. By varying the initial values of the variables, we obtain different system traces, for the same parameters t_0, s and t_f. Simulations of the behavior of an S-system can be automatically obtained by using the tool PLAS (see [27]). In fact, PLAS takes in input an S-system and approximates the values of the system variables, once the parameters in Definition 2 have been specified. The output is exactly a trace describing the behavior of the given system.

Example 1. The following feedback system is taken from [27], Chapter 6, and can be found in PLAS (see `\Book_Examples\Feedback.plc`). It represents a system in which the reactant X_1 is inhibited by X_2, while X_3 is an independent input variable and X_4 an independent inhibitor for the degradation of X_2. Hence, we have $DV = \{X_1, X_2\}$, $IV = \{X_3, X_4\}$, and

$$\dot{X_1} = 0.5X_2^{-2}X_3^{0.5} - 2X_1 \qquad \dot{X_2} = 2X_1 - X_2^{0.5}X_4^{-1}$$

Let $t_0 = 0$ be the initial time, $\boldsymbol{X}(t_0) = \langle 1, 1, 4, 2\rangle$ be the initial values of the reactants, $s = 1$ be the time step, and $t_f = 18$ be the final time. By simulating the system in PLAS with these values and setting the Taylor method with tolerance $1E-16$ we obtain the following trace

$\langle\ \langle 1, 1, 4, 2\rangle, \langle 0.33, 1.59, 4, 2\rangle, \langle 0.22, 1.48, 4, 2\rangle, \ldots$
$\ldots, \langle 0.28, 1.31, 4, 2\rangle, \langle 0.28, 1.31, 4, 2\rangle, \langle 0.28, 1.31, 4, 2\rangle\ \rangle$

where we have not reported all the decimals and states for space reasons. In this trace, for instance, we can observe that the quantity of X_1 is 0.28 in the last tree steps.

The solutions of an S-system have some nice properties. First of all they admit all the derivatives everywhere except when they intersect one of the hyperplane $X_i = 0$, for $i = 1, \ldots, n+m$. There could be problems when $X_i = 0$ for $i \in \{1, \ldots, m+n\}$ in the case one of the exponent is, for instance, of the form 0.5. As noticed in [1], this corresponds to the fact that at reasonably high molecular concentrations, one can adopt continuum models which lend themselves to deterministic models, while at lower concentrations, the discrete molecular interactions become important and deterministic models are more difficult to obtain. However, the existence of all the derivatives implies that if at a given instant t_1 all the X_i, for $i = 1, \ldots, n+m$, are different from 0, then there exists a unique solution in an interval $[t_1, t_1 + \epsilon]$ and this solution can be extended if it still holds that all the variables are different from 0. Moreover, if two solutions $\boldsymbol{f}(t)$ and $\boldsymbol{g}(t)$, obtained with different initial values, pass both in a point $\boldsymbol{d}$, possibly at different times, i.e., there exist two instants t_1 and t_2 such that $\boldsymbol{f}(t_1) = \boldsymbol{g}(t_2) = \boldsymbol{d}$, then from those instants on they always coincide, i.e., for all $p \geq 0$, $\boldsymbol{f}(t_1 + p) = \boldsymbol{g}(t_2 + p)$. This is a consequence of the fact that the *variable* time does not explicitly occurs in the differential equations. What we have just stated in mathematical terms can be restated from the biological point of view saying that if the biological system modeled by the S-system reaches a state $\boldsymbol{d}$, its evolution does not depend on the states in which the system was before reaching $\boldsymbol{d}$ (i.e., the system is *without memory*). In particular, on a set of traces this last property has the following consequence.

Proposition 1. *Let $\langle \boldsymbol{a}_0, \ldots, \boldsymbol{a}_j \rangle$ and $\langle \boldsymbol{b}_0, \ldots, \boldsymbol{b}_i \rangle$ be two traces of an S-system S obtained by using the same time step s. If there exist h and k such that $\boldsymbol{a}_h = \boldsymbol{b}_k$, then for all $r \geq 0$ it holds $\boldsymbol{a}_{h+r} = \boldsymbol{b}_{k+r}$.*

Obviously in the above proposition we are assuming that we are using the same approximation method to obtain both traces. Moreover, it can be the case that the two traces are equal. This property of sets of traces of an S-system implies what is known in the area of Model Checking as *fusion closure* (see [14]). We anticipate here that all the results we present in the rest of this paper are consequences of Proposition 1, i.e., they hold every time we deal with a set of traces satisfying it. We formalize this as follows.

Definition 3 (Convergence). *A set of traces Tr is* convergent *if for all the traces $\langle \boldsymbol{a}_0, \ldots, \boldsymbol{a}_j \rangle$ and $\langle \boldsymbol{b}_0, \ldots, \boldsymbol{b}_i \rangle$ belonging to Tr, if there exist h and k such that $\boldsymbol{a}_h = \boldsymbol{b}_k$, then for all $r \geq 0$ it holds $\boldsymbol{a}_{h+r} = \boldsymbol{b}_{k+r}$.*

Corollary 1. *If Tr is a set of traces of an S-system S obtained by using the same time step s, then Tr is convergent.*

Example 2. Let us consider again the simple feedback system described in Example 1. If we simulate it using $\boldsymbol{X}(t_0) = \langle 0.33, 1.59, 4, 2 \rangle$, i.e., $\boldsymbol{X}(t_1)$ of the trace in Example 1, we obtain

$\langle\ \langle 0.33, 1.59, 4, 2 \rangle, \langle 0.22, 1.48, 4, 2 \rangle, \ldots, \langle 0.28, 1.31, 4, 2 \rangle, \langle 0.28, 1.31, 4, 2 \rangle\ \rangle$

which is exactly the trace we had before without the first state.

3.2 XS-systems

The basic idea of XS-systems (introduced in [4]) is to associate an S-system S with a finite automaton, obtained by suitably encoding a set of traces on S. Essentially, each trace on S can be encoded into a simple automaton, where states correspond to the trace elements (i.e., the values of the system variables observed at each step), and transitions reflect the sequence structure of the trace itself (i.e., there exists a transition from a state v_i to a state v_j if they are consecutive in the trace). When more than one trace is involved in the process, coinciding elements of different traces correspond to the same state in the automaton.

Consider an S-system and a set of traces on it: the automaton derived from the system traces is defined as follows.

Definition 4 (S-system Automaton). *Let S be an S-system and Tr be a set of traces on S. An* S-system automaton *is $\mathcal{A}(S, Tr) = (V, \Delta, I, F)$, where*

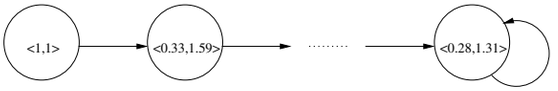
- $V = \{\boldsymbol{v} = \langle v_1, \ldots, v_{n+m}\rangle \mid \exists tr \in Tr : \boldsymbol{v} \text{ is in } tr\} \subseteq D_1 \times \ldots \times D_{n+m}$ *is the set of states;*
- $\Delta = \{(\boldsymbol{v}, \boldsymbol{w}) \mid \exists tr \in Tr : \boldsymbol{v}, \boldsymbol{w} \text{ are consecutive in } tr\}$ *is the transition relation;*
- $I = \{\boldsymbol{v} \mid \exists tr \in Tr : \boldsymbol{v} \text{ is initial in } tr\} \subseteq V$ *is the set of initial states;*
- $F = \{\boldsymbol{v} \mid \exists tr \in Tr : \boldsymbol{v} \text{ is final in } tr\} \subseteq V$ *is the set of final states.*

Automata can be equipped with labels on nodes and/or edges (see [17]). Labels on the nodes maintain information about the properties of the nodes, while labels on the edges are used to impose conditions on the action represented by the edge (see [8]). In the case of S-system automata edges are unlabelled, while the label we assign to each node is actually the name (identifier) of the node itself, i.e. the concentrations of the reactants for that state. In this way S-system automata maintain qualitative information about the system only in the instants corresponding to the steps.

We say that an automaton is *deterministic* if each node has at most one outgoing edge for each edge-label, i.e., in our case, at most one outgoing edge. From Proposition 1 we get the following result.

Proposition 2. *Let S be an S-system and Tr be a convergent set of traces on S. The automaton $\mathcal{A}(S, Tr)$ is deterministic.*

Example 3. The trace shown in Example 1 gives us the following automaton, where we omit the values of the independent variables.



The initial state is the one on the left, while final state is the one on the right. By using both the trace of Example 1 and the trace of Example 2 we obtain

the same automaton, but with two initial states. The automaton represents the fact that all the simulations of this feedback system with initial values of the reactants equal to the values in one of the states of the automaton reach a steady state in which $X_1 = 0.28$ and $X_2 = 1.31$.

In [4], a language called ASySA (*Automata S-systems Simulation Analysis* language) has been presented to inspect and formulate queries on the simulation results of XS-systems. The aim of this language is to provide the biologists with a tool to formulate various queries against a repository of simulation traces. ASySA is essentially a *Temporal Logic* language (see [14]) (an English version of CTL) with a specialized set of predicate variables whose aim is to ease the formulation of queries on numerical quantities. The fusion closure of sets of traces (see Proposition 1 and Corollary 1) is necessary in order to reflect the behavior of the set of traces with temporal logic semantics (see [14]). This means that a formula is true on the S-system automaton if and only if it is true in the set of traces. Intuitively, the behavior of the traces is not approximated in the automaton because two traces which reach the same state always coincide in the future.

Example 4. The automaton in Example 3 satisfies the formula

$$\textsc{Eventually}(\textsc{Always}(X_2 > 1))$$

which means that the system admits a trace such that, from a certain point on, X_2 is always greater than 1. Similarly, it does not satisfies the formula

$$\textsc{Always}(\textsc{Eventually}(X_1 > X_2))$$

since it reaches a steady state in which X_1 is less than X_2.

Unfortunately, in the practical cases the automata built from sets of traces have an enormous number of states. In [4] two techniques have been proposed to reduce the number of states of an S-system automaton, namely *projection* and *collapsing*.

Definition 5 (Projection). *Let S be an S-system and U be a subset of the set of variables of S. Given a trace $tr = \langle \boldsymbol{a}_0, \ldots, \boldsymbol{a}_j \rangle$ of S the* projection over U *of tr is the sequence $tr \restriction U = \langle \boldsymbol{a}_0 \restriction U, \ldots, \boldsymbol{a}_j \restriction U \rangle$. Given a set of traces Tr the* projection over U *of Tr is the set of projected traces $Tr \restriction U = \{tr \restriction U \mid tr \in Tr\}$. The U-*projected S-system automaton *from Tr and S is $\mathcal{A}(S, Tr \restriction U)$.*

The automaton $\mathcal{A}(S, Tr \restriction U)$ has usually less states than $\mathcal{A}(S, Tr)$. However, the set of traces $Tr \restriction U$ does not always satisfies neither convergence nor fusion closure and the automaton $\mathcal{A}(S, Tr \restriction U)$ can be non-deterministic. This can introduce an approximation, i.e., the formulae satisfied by the automaton $\mathcal{A}(S, Tr \restriction U)$ are not the same satisfied by the set of traces $Tr \restriction U$.

Example 5. As a simple yet very interesting example, consider the repressilator system constructed by Elowitz and Leibler [13]. First the authors constructed a mathematical model of a network of three interacting transcriptional regulators and produced a trace of the interaction using a traditional mathematical package (Matlab^{tm}). Subsequently, they constructed a plasmid with the three regulators and collected data from in vivo experiments in order to match them with the predicted values. In particular, this contains three proteins, namely lacI (X_1), tetR (X_2), and cI (X_3). The protein lacI represses the protein tetR, tetR represses cI, whereas cI represses lacI, thus completing a feedback system. The dynamics of the network depend on the transcription rates, translation rates, and decay rates. Depending on the values of these rates the system might converge to a stable limit circle or become unstable. The following S-system represents[3] the repressilator system: rate values have been set in such a way that the system converges to a stable limit circle.

$$\begin{aligned}\dot{X}_1 &= X_4 X_3^{-1} - X_1^{0.5}\\ \dot{X}_2 &= X_5 X_1^{-1} - X_2^{0.578151}\\ \dot{X}_3 &= X_6 X_2^{-1} - X_3^{0.5}\end{aligned}$$

If we simulate it in PLAS, with $t_0 = 0$, $\boldsymbol{X}(t_0) = \langle 0.01, 0.2, 0.01, 0.2, 0.2, 0.2\rangle$, $s = 0.05$, and $t_f = 30$, we obtain a trace whose automaton reaches the loop shown on the left of Figure 1: we omit the independent variables and we use dotted lines to represent the fact that there are other intermediate states.

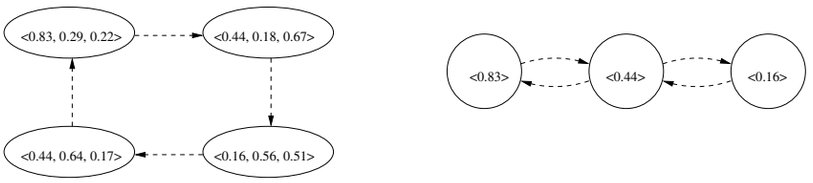


Fig. 1. Repressilator: automaton and projected automaton

The automaton does not satisfies Eventually(Always($X_1 \geq 0.3$)). In fact in the limit circle reached by the repressilator, the values of X_1 range in the interval $[0.16, 0.83]$. Hence, the formula is false also in the projected trace. However, the formula is satisfied by the projected automaton, partially depicted on the right of Figure 1. In fact, the projected automaton represents a system in

[3] To be precise the system described in [13] is not an S-system. However, it can be reasonably approximated through an S-system, as proved by the general theory presented in [27]. Notice that our automaton-model can be built using directly traces of the system in [13].

which it is possible that after a certain instant the variable X_1 assumes values in the interval $[0.44, 0.83]$.

The collapsing operation is defined in such a way that a state is removed from a trace when it behaves similarly to the previous one, i.e., when the derivatives computed in it can be approximated by the derivatives computed in the previous state (see [4] for the formal definition). Also this operation can introduce approximation as shown in the following example.

Example 6. Let S be an S-system with dependent variables X_1 and X_2. Let us assume that S admits a trace of the form $\langle\ \langle 1,5\rangle, \langle 2,4\rangle, \langle 3,3\rangle, \langle 4,2\rangle, \langle 5,1\rangle\ \rangle$.

We also assume that the derivative $\dot{X}_1$ is 1 in all the states except the last one, and, similarly, $\dot{X}_2$ is -1 in all the states except the last one. By applying the definitions presented in [4] we can collapse some of the states obtaining the reduced trace $\langle\ \langle 1,5\rangle, \langle 5,1\rangle\ \rangle$. The formula $\textsc{Eventually}(|X_1 - X_2| \leq 3)$ is true in the trace of S, but is false in the collapsed one.

Consider again the repressilator system of Example 5, whose automaton is partially represented on the left of Figure 1. If all the intermediate states on the dotted lines are collapsed, then we obtain an automaton with 4 states which does not satisfy the formula $\textsc{Eventually}(|X_1 - X_2| \leq 0.1)$, while it is easy to check that the same formula is satisfied by the repressilator system.

In order to avoid these approximations and to obtain a more powerful and flexible framework in the next sections we propose the use of hybrid automata together with a reformulation of projection and collapsing.

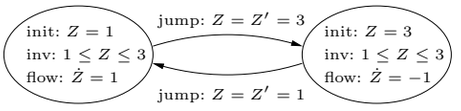
4 Hybrid Automata to Model S-systems

The notion of hybrid automata has been first introduced in [2] as a model and specification language for hybrid systems, i.e., systems consisting of a discrete program within a continuously changing environment.

Definition 6 (Hybrid automata). *An* hybrid automaton $H = (Z, V, \Delta, I, F, \mathit{init}, \mathit{inv}, \mathit{flow}, \mathit{jump})$ *consists of the following components:*

- $Z = \{Z_1, \ldots, Z_k\}$ *a finite set of variables;* $\dot{Z} = \{\dot{Z}_1, \ldots, \dot{Z}_k\}$ *denotes the first derivatives during continuous change;* $Z' = \{Z'_1, \ldots, Z'_k\}$ *denotes the values at the end of discrete change;*
- (V, Δ, I, F) *is an automaton; the nodes of* V *are called* control modes*, the edges of* Δ *are called* control switches*;*
- *each* $v \in V$ *is labeled by* $\mathit{init}(v)$*,* $\mathit{inv}(v)$*, and* $\mathit{flow}(v)$*; the labels* $\mathit{init}(v)$ *and* $\mathit{inv}(v)$ *are constraints whose free variables are in* Z*; the label* $\mathit{flow}(v)$ *is a constraints whose free variables are in* $Z \cup \dot{Z}$*;*
- *each* $e \in E$ *is labeled by* $\mathit{jump}(e)$*, which is a constraint whose free variable are in* $Z \cup Z'$*.*

Example 7. Consider the following simple hybrid automaton.



The initial state is the left one, with $Z = 1$. In this state Z grows with constant rate 1. After 3 instants we have $Z = 3$ and we jump on the right state. In this second state Z decreases and when Z becomes 1 we jump again in the state on the left.

The usefulness of hybrid automata has been widely proved in the area of verification (see, e.g., [22]). In order to exploit the expressive power of hybrid automata their properties have been deeply studied (see [15]) and model checkers have been developed to automatic test temporal logic properties on them. Among the model checkers which deal with hybrid automata we mention HyTech (see [16]) developed at Berkeley University, Charon (see [1]) developed at the University of Pennsylvania.

In the S-system automata introduced in the previous section the only quantitative information maintained is the values of the variables in the instants corresponding to the steps. The values at instants between two steps were lost. This becomes particularly dangerous when we intend to apply a reduction operation like the collapsing one. Our idea here is to use the continuous component of hybrid automata to maintain also some approximate information about the values of the variables between two steps.

Let us introduce some notations which simplify the definition of an hybrid automaton modeling a convergent set Tr of traces of an S-system. Given the vectors $\boldsymbol{X} = \langle X_1, \ldots, X_{n+m} \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_{n+m} \rangle$ we use the notation $\boldsymbol{X} = \boldsymbol{v}$ to denote the conjunction $X_1 = v_1 \wedge \ldots \wedge X_{n+m} = v_{n+m}$. The notation $\boldsymbol{v} \leq \boldsymbol{X} \leq \boldsymbol{w}$ has a similar meaning, while $\dot{\boldsymbol{X}} = (\boldsymbol{w} - \boldsymbol{v})/s$ stands for $\dot{X}_1 = (w_1 - v_1)/s \wedge \ldots \wedge \dot{X}_{n+m} = (w_{n+m} - v_{n+m})/s$.

Definition 7 (S-system Hybrid Automaton). *Let S be an S-system and Tr be a convergent set of traces on S. Consider the S-system automaton $\mathcal{A}(S, Tr)$. The* S-system hybrid automaton *built on $\mathcal{A}(S, Tr)$ is $\mathcal{H}(S, Tr) = (X, V, \Delta, I, F, init, inv, flow, jump)$, where:*

- $X = \{X_1, \ldots, X_{n+m}\} = DV \cup IV$;
- (V, Δ, I, F) *is the automaton $\mathcal{A}(S, Tr)$;*
- *for each $\boldsymbol{v} \in V$ let $init(\boldsymbol{v}) = \boldsymbol{X} = \boldsymbol{v}$;*
- *for each $\boldsymbol{v} \in V$ such that $(\boldsymbol{v}, \boldsymbol{w}) \in \Delta$ let*[4] *$inv(\boldsymbol{v}) = \boldsymbol{v} \leq \boldsymbol{X} \leq \boldsymbol{w}$;*
- *for each $\boldsymbol{v} \in V$ such that $(\boldsymbol{v}, \boldsymbol{w}) \in \Delta$ let $flow(\boldsymbol{v}) = \dot{\boldsymbol{X}} = (\boldsymbol{w} - \boldsymbol{v})/s$;*
- *for each $(\boldsymbol{v}, \boldsymbol{w}) \in \Delta$ let $jump((\boldsymbol{v}, \boldsymbol{w})) = \boldsymbol{X} = \boldsymbol{X}' = \boldsymbol{w}$.*

[4] We invert the interval when $w_i < v_i$.

Notice from the above definition that being in a state $\boldsymbol{v}$ does not necessarily mean that the values of the variables are exactly $\boldsymbol{v}$: they can in fact assume values between $\boldsymbol{v}$ and $\boldsymbol{w}$. In particular, they grow linearly in this interval and when they reach $\boldsymbol{w}$ the system jump on a new state.

The automaton $\mathcal{H}(S, Tr)$ is a rectangular singular automaton and the temporal logic CTL is decidable for this class of automata (see [15]). The model checker HyTech can be used to check whether a temporal formula is satisfied by $\mathcal{H}(S, Tr)$. Moreover, $\mathcal{H}(S, Tr)$ is deterministic, since we require Tr to be convergent and hence $\mathcal{A}(S, Tr)$ is deterministic. Notice also that all the information needed to build $\mathcal{H}(S, Tr)$ is already encoded in $\mathcal{A}(S, Tr)$, i.e., it is possible to work on $\mathcal{H}(S, Tr)$ by only maintaining in memory $\mathcal{A}(S, Tr)$.

Example 8. From the traces of the feedback system of Examples 1 and 2 we obtain the hybrid automaton shown in Figure 8. In the first state (the one on

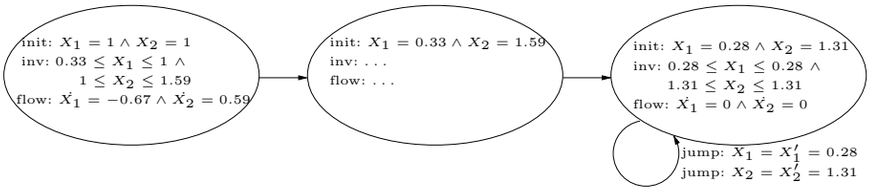


Fig. 2. Feedback hybrid automaton

the left) variable X_1 starts with value 1 and decreases until it reaches value 0.33, while variable X_2 starts with value 1 and grows until it reaches value 0.59. Then, we jump on the second state. When we reach the last state the values of the variables become stable and the system loops forever.

The additional quantitative information stored in each state of an S-system hybrid automaton allow to deeply investigate the behavior of the system in the instants within a step. This becomes really relevant when we apply a collapsing technique to reduce the number of states as we will see in Section 6.

5 Bisimulation and Projection

As pointed out in Example 5 the projection operation can cause incorrect prediction on the reduced automaton. In order to avoid this problem, we define in this section a projection operator based on the notion of bisimulation. Since bisimulation is an equivalence relation preserving temporal logic formulae (see, e.g., [9]), the obtained projected automata will satisfy the same formulae of the original one.

Let us fix some notations. Given a condition $\mathit{init}(\boldsymbol{v})$ ($\mathit{inv}(\boldsymbol{v})$, $\mathit{flow}(\boldsymbol{v})$, resp.) and $U \subseteq DV \cup IV$ we use $\mathit{init}(\boldsymbol{v}) \restriction U$ ($\mathit{inv}(v) \restriction U$, $\mathit{flow}(v) \restriction U$, resp.) to denote that we consider only the conditions relative to the variables in U.

Definition 8 (U-bisimulation). *Let $\mathcal{H}(S, Tr)$ be an S-system hybrid automaton. Let $U \subseteq \{X_1, \ldots, X_{n+m}\}$ be a subset of variables. A relation $R \subseteq V \times V$ is a* U-bisimulation *if*

- *if $\boldsymbol{v} R \boldsymbol{w}$, then $init(\boldsymbol{v}) \restriction U = init(\boldsymbol{w}) \restriction U \wedge inv(\boldsymbol{v}) \restriction U = inv(\boldsymbol{w}) \restriction U \wedge flow(\boldsymbol{v}) \restriction U = flow(\boldsymbol{w}) \restriction U$;*
- *if $\boldsymbol{v} R \boldsymbol{w}$ and $(\boldsymbol{v}, \boldsymbol{v}') \in \Delta$, then $(\boldsymbol{w}, \boldsymbol{w}') \in \Delta$ and $\boldsymbol{v}' R \boldsymbol{w}'$;*
- *if $\boldsymbol{v} R \boldsymbol{w}$ and $(\boldsymbol{w}, \boldsymbol{w}') \in \Delta$, then $(\boldsymbol{v}, \boldsymbol{v}') \in \Delta$ and $\boldsymbol{v}' R \boldsymbol{w}'$.*

The idea is that two states $\boldsymbol{v}$ and $\boldsymbol{w}$ are U-bisimilar if not only the variables in U have the same values in $\boldsymbol{v}$ and $\boldsymbol{w}$, but from $\boldsymbol{v}$ and $\boldsymbol{w}$ the system evolves in the same way with respect to the variables in U. In fact, for instance,it is possible that there are two states in which the variables in U have the same values, but the first state evolves into a state in which the variables are incremented while the second one evolves into a state in which the variable are decremented. Intuitively we do not want to consider equivalent these two states.

Lemma 1. *There always exists a unique maximum U-bisimulation $\approx_U$ which is an equivalence relation. Moreover, if $\boldsymbol{v} \approx_U \boldsymbol{w}$ and $(\boldsymbol{v}, \boldsymbol{v}') \in \Delta$, then $(\boldsymbol{w}, \boldsymbol{w}') \in \Delta$ and $jump((\boldsymbol{v}, \boldsymbol{v}')) \restriction U = jump((\boldsymbol{w}, \boldsymbol{w}')) \restriction U$.*

Proof. The first part follows immediately from the fact that a U-bisimulation on $\mathcal{H}(S, Tr)$ is nothing but a strong bisimulation on $\mathcal{A}(S, Tr)$ whose nodes have been labeled using part of the conditions defining the hybrid automaton $\mathcal{H}(S, Tr)$.

The second fact is a consequence of the fact that *jump* is uniquely defined once we know *init* and *inv*. □

Definition 9 (Projected Hybrid automaton $\mathcal{H}(S, Tr, U)$). *Let $\mathcal{H}(S, Tr) = (X, V, \Delta, I, F, init, inv, flow, jump)$, be an S-system hybrid automaton and U be a subset of variables. The* projected hybrid automaton $\mathcal{H}(S, Tr, U) = (U, V_U, \Delta_U, I_U, F_U, init_U, inv_U, flow_U, jump_U)$ *is defined as follows:*

- *$V_U = V/\approx_U$;*
- *$\Delta_U = \{([\boldsymbol{v}], [\boldsymbol{w}]) \mid \exists \boldsymbol{v}' \in [\boldsymbol{v}], \boldsymbol{w}' \in [\boldsymbol{w}] : (v, w) \in \Delta\}$;*
- *for each $[\boldsymbol{v}] \in V_U$ let $init_U([\boldsymbol{v}]) = init(\boldsymbol{v}) \restriction U$;*
- *for each $[\boldsymbol{v}] \in V_U$ let $inv_U([\boldsymbol{v}]) = inv(\boldsymbol{v}) \restriction U$;*
- *for each $[\boldsymbol{v}] \in V_U$ let $flow_U([\boldsymbol{v}]) = flow(\boldsymbol{v}) \restriction U$;*
- *for each $([\boldsymbol{v}], [\boldsymbol{w}]) \in \Delta_U$ such that $(\boldsymbol{v}', \boldsymbol{w}') \in \Delta$ let $jump_U(([\boldsymbol{v}], [\boldsymbol{w}])) = jump((v', w')) \restriction U$.*

The above definition does not depend on the representative element of each class. This is a consequence of the definition of $\approx_U$ as far as the *init*, *inv*, and *flow* conditions are concerned, and of Lemma 1 as far as the *jump* conditions are concerned. For those who are familiar with automata and bisimulation reductions we can say that the hybrid automaton $\mathcal{H}(S, Tr, U)$ is nothing but the hybrid automaton built on the bisimulation reduced automaton $\mathcal{A}(S, Tr)/\approx_U$ with conditions defined only on the variables of U.

The automaton $\mathcal{H}(S, Tr, U)$ is still a rectangular singular automaton, hence CTL is still decidable on it. Moreover, $\mathcal{H}(S, Tr, U)$ is deterministic, since bisimulation preserves determinism. The fact that we are working on deterministic automata implies that the bisimulation relation $\approx_U$ can be computed in linear (see [12]) time using the procedure defined in [24].

As far as the correctness of the reduction is involved, we have the following result.

Proposition 3. *Let TL be a temporal logic which is a fragment of the μ-calculus. Let φ a formula of TL involving only the variables in U. $\mathcal{A}(S, Tr)$ satisfies φ if and only if $\mathcal{A}(S, Tr)/\approx_U$ satisfies φ. $\mathcal{H}(S, Tr)$ satisfies φ if and only if $\mathcal{H}(S, Tr, U)$ satisfies φ.*

Proof. The first part is a consequence of the fact that $\approx_U$ is a strong bisimulation and strong bisimulations preserve all the formulae of the μ-calculus.

The second part is a consequence of the first part and of the fact that $\mathcal{H}(S, Tr, U)$ is basically the hybrid automaton built on $\mathcal{A}(S, Tr)/\approx_U$. □

In the following example we show the difference between $\mathcal{A}(S, Tr \restriction U)$ and $\mathcal{A}(S, Tr)/\approx_U$. This difference is at the basis of the correctness of $\mathcal{H}(S, Tr, U)$.

Example 9. Consider again the repressilator system of Example 5. Part of the projected automaton we obtain by applying bisimulation is shown on the left of Figure 3. The two states in which $X_1 = 0.44$ do not coincide when we use bisimulation. In fact, the first state in which X_1 is 0.44 evolves to a state in which X_1 is 0.43 (the protein is decreasing), while the second state in which X_1 is 0.44 evolves to a state in which X_1 is 0.47 (the protein is growing). Hence, the projected automaton does not satisfies the formula EVENTUALLY(ALWAYS($X_1 \geq 0.3$)). This is correct, since the repressilator system we are simulating reaches a steady loop in which X_1 oscillates between 0.16 and 0.83. Part of the projected hybrid automaton is shown on the right of Figure 3.

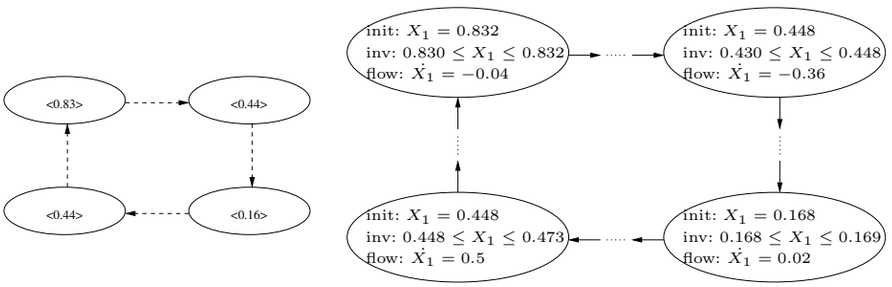


Fig. 3. Repressilator: bisimulation quotiented automata

6 Collapsing States

In this section we introduce the definition of collapsing of a trace. The definition we present is similar but not equivalent to the one given in [4]. If fact, we do not consider the difference between the derivatives calculated in the states, but the degree of growth within a step. This is inspired by hybrid automaton in which in the *flow* condition of a state we do not use the derivatives calculated at the beginning of a time step, but the coefficients of the lines connecting the values at the beginning to the ones at the end of a time step. In the following collapsing definition we use a compact notation similar to the one already introduced in Section 4.

Definition 10 (Collapsing). *Let $\boldsymbol{\delta} = \langle \delta_1, \ldots, \delta_{n+m} \rangle$ be a $n+m$-vector of values. Let $tr = \langle \boldsymbol{a}_0, \ldots, \boldsymbol{a}_j \rangle$ be a trace obtained by simulating the S-system S with time step s. A $\boldsymbol{\delta}$-collapsing of tr is a partition of the states of tr such that:*

- *the blocks are sub-traces of tr;*
- *if a block is formed by the states from $\boldsymbol{a}_i$ to $\boldsymbol{a}_{i+h}$, and $\boldsymbol{a}_j, \boldsymbol{a}_{j+1}$ belong to the block, then $|(\boldsymbol{a}_{j+1} - \boldsymbol{a}_j)/s - (\boldsymbol{a}_{i+1} - \boldsymbol{a}_i)/s| \leq \boldsymbol{\delta}$.*

The collapsing operation in [4] is based on the difference between the first derivatives computed in the elements of the trace. Here, instead, we consider as collapsing criterion the degree of growth within a step. In practice the definition requires that the lines connecting $\boldsymbol{a}_i$ to $\boldsymbol{a}_{i+1}$ are good approximations of the lines connecting $\boldsymbol{a}_j$ to $\boldsymbol{a}_{j+1}$. As a consequence we obtain that the lines connecting $\boldsymbol{a}_i$ to $\boldsymbol{a}_{i+h}$ are good approximations of all the small lines. In particular, the following result holds.

Lemma 2. *If a block of a $\boldsymbol{\delta}$-collapsing is formed by the sequence of states from $\boldsymbol{a}_i$ to $\boldsymbol{a}_{i+h}$ and $\boldsymbol{a}_j, \boldsymbol{a}_{j+1}$ belong to the block, then $|(\boldsymbol{a}_{j+1} - \boldsymbol{a}_{j+r})/s - (\boldsymbol{a}_{i+h} - \boldsymbol{a}_i)/(h * s)| \leq 2 * \boldsymbol{\delta}$.*

Proof. It is not restrictive to prove that the result holds on the first component. Let $(a_{i+r+1,1} - a_{i+r,1})/s = coef_{r+1}$ for $r = 0, \ldots, h-1$. It is easy to prove that $(a_{i+h,1} - a_{i,1})/(h*s) = (1/h)*((a_{i+h,1} - a_{i+h-1,1})/s + (a_{i+h-1,1} - a_{i+h-2,1})/s + \ldots + (a_{i+1,1} - a_{i,1})/s) = (1/h)*\sum_{r=0}^{h-1} coef_{r+1}$. By hypothesis we have $coef_1 - \delta_1 \leq coef_{r+1} \leq coef_1 + \delta_1$, hence we obtain $(1/h)*h*(coef_1 - \delta_1) \leq (a_{i+h,1} - a_{i,1})/(h*s) \leq (1/h)*h*(coef_1 + \delta_1)$, i.e. $coef_1 - \delta_1 \leq (a_{i+h,1} - a_{i,1})/(h*s) \leq coef_1 + \delta_1$. From this last we get $(coef_1 - \delta_1) - (coef_1 + \delta_1) \leq coef_{r+1} - (a_{i+h,1} - a_{i,1})/(h*s) \leq (coef_1 + \delta_1) - (coef_1 - \delta_1)$, i.e. $-2*\delta_1 \leq coef_{r+1} - (a_{i+h,1} - a_{i,1})/(h*s) \leq 2*\delta_1$, which is equivalent to our thesis. □

Given the trace tr and the vector $\boldsymbol{\delta}$ the partition in which each state constitutes a singleton class is a $\boldsymbol{\delta}$-collapsing.

Definition 11 (Maximal Collapsing). *Let C_1 and C_2 be two $\boldsymbol{\delta}$-collapsing of tr. We say that C_1 is* coarser *than C_2 if each block of C_2 is included in a block of C_1. We say that the $\boldsymbol{\delta}$-collapsing C_1 is* maximal *if there does not exist another $\boldsymbol{\delta}$-collapsing coarser than C_1.*

The uniqueness of a coarsest $\boldsymbol{\delta}$-collapsing is not guaranteed. However, we can give an algorithm to find a maximal $\boldsymbol{\delta}$-collapsing. The algorithm performs the following steps: it starts from $\boldsymbol{a}_0$, it check if $\boldsymbol{a}_1$ can be collapsed with $\boldsymbol{a}_0$, if this is the case it goes on with $\boldsymbol{a}_2$, and so on. Assume that $\boldsymbol{a}_i$ is the first state which does not collapse with $\boldsymbol{a}_0$, then the algorithm starts another block from $\boldsymbol{a}_i$ and it goes on in the same way.

The following proposition states that if we use maximal δ-collapsing, then two traces which match in one state always match in the future.

Proposition 4. *Let Tr be a convergent set of traces of an S-system S. Let $Tr/\boldsymbol{\delta}$ be the set of collapsed traces obtained by applying to each trace of Tr a maximal $\boldsymbol{\delta}$-collapsing. The set $Tr/\boldsymbol{\delta}$ is convergent.*

This property is sufficient to guarantee that taking a set of traces and collapsing them using maximal δ-collapsing, the set of collapsed traces can be used to build automata and hybrid automata as defined in the previous sections. In fact, as pointed out along the paper, the correctness of our framework holds whenever we use convergent sets of traces. This does not mean that the automaton we build from a set of collapsed traces satisfies the same formulae of the original set of traces, but that it satisfies the same formulae of the set of collapsed of traces.

Example 10. Consider again the S-system and the trace of Example 6. The collapsed trace we obtain is again $\langle\ \langle 1, 5\rangle, \langle 5, 1\rangle\ \rangle$. The hybrid automaton we build from this trace has two states. In the first state, let us call it $\boldsymbol{v}$, we have the conditions

$$\mathit{inv}(\boldsymbol{v}) = 1 \leq X_1 \leq 5 \wedge 1 \leq X_2 \leq 5 \qquad \mathit{flow}(\boldsymbol{v}) = \dot{X}_1 = 1 \wedge \dot{X}_2 = -1$$

which make $\textsc{Eventually}(|X_1 - X_2| \leq 3)$ true in the automaton, as it was in the original trace.

Similarly, we can safely collapse the states of the repressilator system and obtain an hybrid automaton with four states which correctly satisfy the formula $\textsc{Eventually}(|X_1 - X_2| \leq 0.1)$.

This last example shows that the additional information maintained in the hybrid automaton is particularly useful when we use techniques as collapsing to reduce the number of states.

7 Conclusions

In this paper we have described how hybrid automata can be used to model and analyze set of traces representing the behavior of a biological system. Automata give a qualitative view of a set of traces by abstracting from the time instants, thus allowing a compact representation in which the properties of the system can be easily investigated. The use of hybrid automata, instead of standard ones, improves along the direction of a qualitative, but *complete*, modeling of the

biological system. In fact, powerful techniques such as (bisimulation-)projections and collapsing can be "safely" applied to hybrid automata in order to reduce the number of states. In particular, while the bisimulation based projection we present could be applied also to standard automata, the "good" behavior of the collapsing operation w.r.t. the verification of temporal formulae strongly depends on the information which is stored in each state of hybrid automata. Notice that, although we have presented a construction of hybrid automata from standard S-systems, it is not difficult to modify our framework in order to deal with more complicated systems, e.g., systems whose differential equations change during the evolution of the system itself.

As future work, we intend to extend our tool set in order to (1) integrate some Time-Frequency analysis tools to identify the XS-system states by analyzing the traces, and (2) incorporate a *learning* scheme in our approach that keeps track of what automata (read: model) is being manipulated at a given stage of the analysis carried out by a biologist, and that amends it semi-automatically before stepping in the next stage. This could be particularly useful in the evaluation phase for the constant rates involved in the system.

References

[1] R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. J. Pappas, H. Rubin, and J. Schug. Hybrid modeling and simulation of biomolecular networks. In *Hybrid Systems: Computation and Control*, volume 2034 of *LNCS*, pages 19–32. Springer-Verlag, 2001. 60, 62, 67

[2] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In R. L. Grossman, A. Nerode, A. P. Ravn, and H. Richel, editors, *Hybrid Systems*, LNCS, pages 209–229. Springer-Verlag, 1992. 66

[3] M. Antoniotti, F. C. Park, A. Policriti, N. Ugel, and B. Mishra. Foundations of a Query and Simulation System for the Modeling of Biochemical and Biological Processes. In *Proc. of the Pacific Symposium of Biocomputing (PSB'03)*, 2003. 58, 59

[4] M. Antoniotti, A. Policriti, N. Ugel, and B. Mishra. XS-systems: extended S-systems and algebraic differential automata for modeling cellular behaviour. In *Proc. of Int. Conference on High Performance Computing (HiPC'02)*, 2002. 58, 59, 60, 63, 64, 66, 71

[5] M. Antoniotti, A. Policriti, N. Ugel, and B. Mishra. Model Building and Model Checking for Biological Processes. *Cell Biochemistry and Biophysics*, 2003. To appear. 58

[6] U. S. Bhalla. Data Base of Quatitative Cellular Signaling (DOQCS). Web site at `http://doqcs.ncbs.res.in/`, 2001. 58

[7] R. W. Brockett. Dynamical systems and their associated automata. In *Systems and Networks: Mathematical Theory and Applications*, volume 77. Akademie-Verlag, Berlin, 1994. 59

[8] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model checking*. MIT Press, 1999. 63

[9] E.M. Clarke and E.A. Emerson. Design and synthesis of synchronization skeletons using brancing time temporal logic. In *Proc. Workshop Logic of Programs*, volume 131 of *LNCS*. Springer, 1981. 68

[10] M. Curti, P. Degano, C. Priami, and C. T. Baldari. Casual π-calculus for biochemical modelling. DIT 02, University of Trento, 2002. 60

[11] H. de Jong. Modeling and simulation of genetic regulatory systems: A literature review. DIT 4032, Inria, 2000. 59

[12] A. Dovier, C. Piazza, and A. Policriti. A fast bisimulation algorithm. In G. Berry, H. Comon, and A. Finkel, editors, *Proc. of Int. Conference on Computer Aided Verification (CAV'01)*, volume 2102 of *LNCS*, pages 79–90. Springer-Verlag, 2001. 70

[13] M. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403:335–338, 2000. 59, 65

[14] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 995–1072. MIT Press, 1990. 62, 64

[15] T. A. Henzinger. The theory of hybrid automata. In *Proc. of IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 278–292. IEEE Press, 1996. 67, 68

[16] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. HYTECH: A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1–2):110–122, 1997. 67

[17] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979. 63

[18] P. D. Karp, M. Riley, S. Paley, and A. Pellegrini-Toole. The MetaCyc Database. *Nucleic Acid Research*, 30(1):59, 2002. 58

[19] P. D. Karp, M. Riley, M. Saier, and S. Paley A. Pellegrini-Toole. The EcoCyc Database. *Nucleic Acids Research*, 30(1):56, 2002. 58

[20] KEGG database. `http://www.genome.ad.jp/kegg/`. 58

[21] H. Kitano. Systems Biology: an Overview. *Science*, 295:1662–1664, March 2002. 57

[22] O. Müller and T. Stauner. Modelling and verification using linear hybrid automata. *Mathematical and Computer Modelling of Dynamical Systems*, 6(1):71–89, 2000. 67

[23] Nat. Cent. Genome Resources. Pathdb database. `http://www.ncgr.org/pathdb/`. 58

[24] R. Paige, R. E. Tarjan, and R. Bonic. A linear time solution to the single function coarsest partition problem. *Theoretical Computer Science*, 40:67–84, 1985. 70

[25] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. In *Proc. of the Pacific Symposium of Biocomputing (PSB'01)*, pages 459–470, 2003. 60

[26] D. Shasha, A. Kouranov, L. Lejay, C. Chou, and G. Coruzzi. Combinatorial Design to study regulation by multiple input signals: A tool for parsimony in the post-genomics era. *Plant Physiology*, 127:1590–1594, December 2001. 59

[27] E. O. Voit. *Computational Analysis of Biochemical Systems. A Pratical Guide for Biochemists and Molecular Biologists*. Cambridge University Press, 2000. 60, 61, 65

[28] WIT database. `http://wit.mcs.anl.gov/WIT2/`. 58

Multiscale Modeling of Alternative Splicing Regulation

Damien Eveillard[1], Delphine Ropers[2], Hidde de Jong[3], Christiane Branlant[2], and Alexander Bockmayr[1⋆]

[1] LORIA, Université Henri Poincaré
BP 239, 54506 Vandœuvre-lès-Nancy, France
Damien.Eveillard@loria.fr
[2] Laboratoire de Maturation des ARN et Enzymologie Moléculaire
UMR 7567 CNRS-UHP, BP 239, 54506 Vandoeuvre-lès-Nancy, France
[3] INRIA Rhône-Alpes, Helix Project
655 avenue de l'Europe, Montbonnot, 38334 Saint Ismier, France

Abstract. Alternative splicing is a key process in post-transcriptional regulation, by which several kinds of mature RNA can be obtained from the same premessenger RNA. Using a constraint programming approach, we model the alternative splicing regulation at different scales (single site vs. multiple sites), thus exploiting different types of available experimental data.

1 Introduction

Alternative splicing is a biological process occurring in post-transcriptional regulation of RNA. Through the elimination of selected introns, alternative splicing allows generating several kinds of mRNA from the same premessenger RNA. The combinatorial effect of splicing contributes to biological diversity, especially in the case of the human immunodeficiency virus (HIV-1). Recent biological studies show the impact that SR proteins have on the dynamics of post-transcriptional regulation via the control of the splicing process [8]. SR proteins can be divided into two functional classes: they may either activate or inhibit splicing. Due to the complexity of alternative splicing regulation, the knowledge that can be gained from experiments is limited. Each experiment focus on one splicing site. In a first approach, we model SR regulation in this restricted context. Using differential equations, we develop a qualitative model for the A3 splicing site in HIV-1. The qualitative behavior depends on the values of the reaction kinetic parameters. Experimental results available to us validate this first approach in the equilibrium phase. Our second approach aims at validation on a higher scale. The ultimate goal is to obtain a model that can be validated qualitatively both on the scale of a single splicing site and on the scale of the whole HIV-1, in order to represent the global effect of alternative splicing in the HIV-1 cycle.

⋆ Part of this work was done within the ARC INRIA "Process Calculi and Biology of Molecular Networks", http://contraintes.inria.fr/cpbio

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 75–87, 2003.

Our models are developed in a constraint programming framework [2, 3]. Constraint programming seems well-suited for modeling biological systems because it allows one to handle partial or incomplete information on the system state. Each constraint gives one piece of information on the system. The overall knowledge is accumulated in the constraint store. The constraint engine available in constraint programming systems operates on the constraint store. It may add new information to the store or check whether some property is entailed by the information already available.

While a constraint model may be refined whenever additional biological knowledge becomes available, it allows one to make useful inferences even from partial and incomplete information. Therefore, constraint programming seems to be a natural computational approach to face the current situation in systems biology as it is described by B. Palsson [16]: "Because biological information is incomplete, it is necessary to take into account the fact that cells are subject to certain constraints that limit their possible behaviors. By imposing these constraints in a model, one can then determine what is possible and what is not, and determine how a cell is likely to behave, but never predict its behavior precisely."

The organisation of the paper is as follows: we start in Sect. 2 with a description of the biological process of alternative splicing regulation. Based on a number of biological hypotheses, we develop a continuous model of the regulation at one splicing site. This model includes competition and compensation of different proteins on two binding sites, ESE and ESS2. The single-site model is validated in a qualitative way by extracting from the model a splice efficiency function, which can be measured in experiments. In Sect. 3, we first simulate the single-site continuous model in the hybrid concurrent constraint programming language `Hybrid cc` [9, 10]. Then we derive a more global model involving two generic splicing sites, which may be generalized to multiple sites. This means that we model at two different scales, using the splice efficiency as an abstraction of the local model of one site in the more global context of different sites. The two-site model uses the constraint solving and default reasoning facilities of `Hybrid cc`. This allows us to make predictions on the global behavior even in absence of detailed local information on some of the splicing sites.

2 The Biological Problem of Alternative Splicing Regulation

2.1 Biological Process

Eukaryotic and virus gene expression is based on production of RNA containing intronic sequences. The process of splicing allows for intron elimination and junction of exonic sequences [14], see Fig. 1. Splicing is a major biological process in the HIV-1 life cycle: the viral RNA either remains unchanged to serve as genomic RNA for new virions, or it is alternatively spliced to allow for the production of virion proteins [26]. The viral genome, initially RNA, is integrated

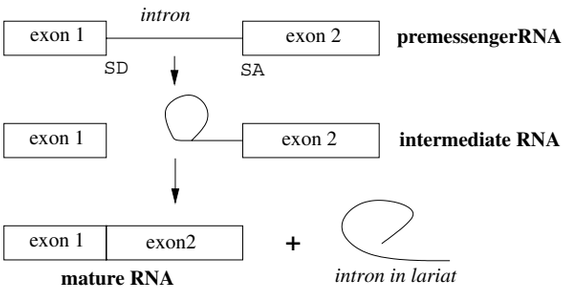


Fig. 1. Representation of the splicing process

into the host genome. In the HIV-1 case, splicing regulation is a complex phenomenon involving 4 donors sites (SD) and 8 acceptors sites (SA), which may yield 40 proteins [17].

This combinatorial complexity is achieved by regulating the selection of the acceptor site [15, 17]. Protein factors control the regulation via binding sites. We focus in our study on the acceptor site A3, where splicing can be repressed by hnRNP A/B via the ESS2 binding site [4, 6], see Fig. 2. Recent experimental studies carried out in our group [18] show that an ESE sequence can activate splicing at the A3 site when SC35 and ASF/SF2 proteins bind to it. More specifically, the ratio of hnRNP A/B and SR proteins determines the splice efficiency at the A3 site.

2.2 Biological Hypotheses

We model the regulation by SR proteins in the restricted context of the A3 splicing site (see Fig. 2) under the following hypotheses:

– We study only one splicing site. Thus, we consider regulation at the scale corresponding to our experimental results. These yield the splice efficiency as the ratio of the mature RNA over premessenger RNA.

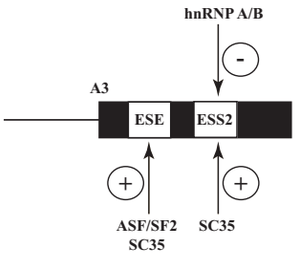


Fig. 2. Regulatory elements of the A3 splicing site

- We suppose that the splicing process involves two steps, relating three functional classes of RNA: immature, intermediate, and mature RNA, see Fig. 1. Intermediate RNA corresponds to immature RNA activated by proteins. Mature RNA corresponds to mature RNA and introns in lariat.
- The protein concentration in experiments is saturated. Therefore, we assume that it is constant, despite the binding of proteins to the RNA during regulation.
- SR proteins have two functions. They regulate the splicing process, and they initialize the splicing machinery.
- Regulation is controlled by the ESE and ESS2 binding sites, which are independent. Thus, only indirect interaction is possible between ESE and ESS2.
- The SR proteins ASF/SF2 and SC35 may activate the first splicing reaction by binding to the site ESE. We assume that these two proteins compensate each other.
- The hnRNP proteins may inhibit the first splicing reaction by binding to the site ESS2. On the other hand, if the SC35 proteins bind to ESS2, this activates the first splicing reaction. Therefore we have a competition between hnRNP and SC35.
- Due to a lack of experimental results, we assume Michaelis-Menten kinetics for all proteins. Our goal is a qualitative model validated at equilibrium. Thus, we do not consider transient states.

Our biological hypotheses are summarized in Fig. 3

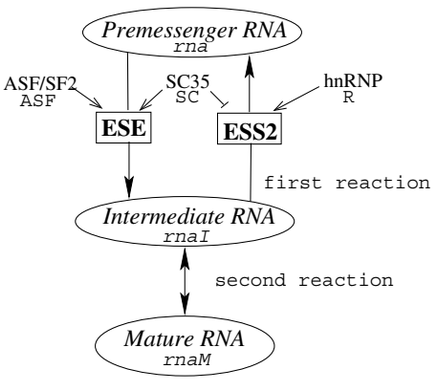


Fig. 3. Schematic representation of the splicing site regulation

2.3 Formal Model

The biological hypotheses can be translated into a mathematical formalism, which leads to a system of ordinary differential equations based on the Michaelis-Menten relation [22]. The single-site model will later be integrated into a larger

multi-site model, see Sect. 3. We describe the splicing process by three ordinary differential equations (ODEs) corresponding to the three functional classes of RNA. Two terms represent the first splicing reaction. The first term represents the cooperation between ASF/SF2 and SC35 in the regulation of ESE. Since we assume compensation, only the sum of activator proteins is important. We represent this interaction by a Michaelis-Menten function depending on the quantity of immature RNA and controlled by the sum of the proteins ASF/SF2 and SC35 [13]:

$$\frac{\varphi_{ESE}(ASF + SC)}{k_{ESE} + ASF + SC}\, rna$$

The symbols used are given in Tab. 1. The second term captures the antagonistic function of hnRNP and SC35 proteins on the site ESS2. In this case, the Michaelis-Menten function represents the inhibitive competition between two proteins : hnRNP and SC35 [27]. It depends on the quantity of intermediary RNA:

$$\frac{\varphi_R \times R}{k_R(1 + \frac{SC}{k_{SC}+R})}\, rnaI$$

The second splicing reaction is modeled by a simple first order kinetic with constant parameters κ and κ'. Different RNAs decrease proportional to the same degradation factor λ. We formalize the biological process by the system of differential equations:

$$\begin{aligned}
\frac{d(rna)}{dt} &= \frac{\varphi_R \times R}{k_R(1 + \frac{SC}{k_{SC}+R})}\, rnaI - \frac{\varphi_{ESE}(ASF + SC)}{k_{ESE} + ASF + SC}\, rna - \lambda \cdot rna \\
\frac{d(rnaI)}{dt} &= \frac{\varphi_{ESE}(ASF + SC)}{k_{ESE} + ASF + SC}\, rna - \frac{\varphi_R \cdot R}{k_R(1 + \frac{SC}{k_{SC}+R})}\, rnaI - \kappa \cdot rnaI \\
&\quad + \kappa' \cdot rnaM - \lambda \cdot rnaI \\
\frac{d(rnaM)}{dt} &= \kappa \cdot rnaI - \kappa' \cdot rnaM - \lambda \cdot rnaM
\end{aligned}$$

2.4 Validation of the Regulatory System

The formal model of regulation at a single-site can be directly simulated in the constraint programming language `Hybrid cc`, see Sect. 3.1. However, before doing this, it should first be validated with respect to existing biological knowledge.

A mathematical analysis of the ODE system in Sect. 2.3 shows that the partial derivatives in the Jacobian matrix have two characteristic properties:

- the partial derivatives on the diagonal elements are negative.
- the partial derivatives on the extra diagonal elements are positive.

Table 1. Symbols and units for the biological variables and parameters

Symbol	Variables and Parameters	unit
rna	Immature RNA	μM
$rnaI$	Intermediary RNA	μM
$rnaM$	Mature RNA	μM
ASF	Protein ASF/SF2	μM
SC	Protein SC35	μM
R	Protein hnRNP	μM
φ_{ESE}	Maximal affinity for the enhancer	s^{-1}
φ_R	Maximal affinity of hnRNP	s^{-1}
k_{ESE}	Half saturation coefficient for the enhancer	μM
k_{SC}	Half saturation coefficient for SC35	μM
k_R	Half saturation coefficient for hnRNP	μM
κ	Reaction rate	s^{-1}
κ'	Reaction rate	s^{-1}
λ	Degradation coefficient	s^{-1}

In our model, the RNA concentrations do not reach an equilibrium, but continue to decrease until total degradation of RNA. However, the above two properties imply that the *splice efficiency* defined by

$$\textit{efficiency}\ (t) = \frac{rnaM(t)}{rna(t)}$$

reaches an equilibrium [1]. From the condition $d\ \textit{efficiency}/dt = 0$, we may derive the following formula for the splice efficiency in the equilibrium phase:

$$\textit{efficiency} = \frac{\kappa \cdot \varphi_{ESE}(ASF + SC)(k_R \cdot k_{SC} + k_R \cdot SC + R \cdot k_{SC})}{\kappa'(k_{ESE} + ASF + SC) \cdot \varphi_R \cdot R \cdot k_{SC}}$$

It is reasonable to assume that this equilibrium is reached after a short transient phase, so that it can be measured by experiments. According to our formula, the splice efficiency is

- an increasing function of the activators SC and ASF.
- a decreasing function of the inhibitor R.

Experimental results show that

- $rnaM/rna$ increases with an increase of activator proteins.
- $rnaM/rna$ decreases with an increase of inhibitor proteins.

These results of our model correlate with available experimental data. In summary, the model may be considered as qualitatively validated under the hypotheses described in Sect. 2.2. We next consider simulation in the concurrent constraint language `Hybrid cc`.

3 Multiscale Modeling and Simulation with Hybrid cc

3.1 Hybrid Concurrent Constraint Programming

In constraint programming, the user specifies constraints on the behavior of the system that is being studied. Each constraint expresses some partial information on the system state. The constraint solver may check constraints for consistency or infer new constraints from the given ones. In concurrent constraint programming (`cc`), different computation processes may run concurrently. Interaction is possible via the *constraint store*. The store contains all the constraints currently known about the system. A process may *tell* the store a new constraint, or *ask* the store whether some constraint is entailed by the information currently available, in which case further action is taken [19]. One major difficulty in the original `cc` framework is that `cc` programs can detect only the presence of information, not its absence. To overcome this problem, [20] proposed to add to the `cc` paradigm a sequence of phases of execution. At each phase, a `cc` program is executed. At the end, absence of information is detected, and used in the next phase. This results in a synchronous reactive programming language, `Timed cc`. But, the question remains how to detect negative information instantaneously. `Default cc` extends `cc` by a negative ask combinator `if` c `else` A, which imposes the constraints of A unless the rest of the system imposes the constraint c. Logically, this can be seen as a default. Introducing phases as in `Timed cc` leads to `Timed Default cc` [21]. Only one additional construct is needed: `hence` A, which starts a copy of A in each phase after the current one.

Table 2. Combinators of `Hybrid cc`

Agents	Propositions
c	c holds now
`if` c `then` A	if c holds now, then A holds now
`if` c `else` A	if c will not hold now, then A holds now
`new` X `in` A	there is an instance $A[T/X]$ that holds now
A, B	both A and B hold now
`hence` A	A holds at every instant after now
`always` A	same as (A, `hence` A)
`when(`c`)` A	same as `(if` c `then` A`, hence (if` c `then` A`))`
`unless(`c`)` A `else` B	same as `(if` c `then` B`, if` c `else` A`)`

`Hybrid cc` [9, 10] is an extension of `Default cc` over continuous time. First continuous constraint systems are allowed, i.e., constraints may involve differential equations that express initial value problems. Second, the `hence` operator is interpreted over continuous time. It imposes the constraints of A at every real time instant after the current one. The evolution of a system in `Hybrid cc` is piecewise continuous, with a sequence of alternating point and interval phases.

All discrete changes take place in a point phase, where a simple `Default cc` program is executed. In a continuous phase, computation proceeds only through the evolution of time. The interval phase, whose duration is determined in the previous point phase, is exited as soon as the status of a conditional changes [10]. Tab. 2 summarizes the basic combinators of `Hybrid cc`.

`Hybrid cc` is well-suited for modeling dynamic biological systems, as argued in [2, 3].

3.2 Single-Site Model: Local Modeling

The single-site model from Sect. 2.3 with experimental values can be expressed directly in `Hybrid cc`.

```
interval t, rna, rnaI, rnaM;
t=0; rna = 0.06; rnaI = 0; rnaM = 0;
always{ t' = 200;
  rna'  = (Pr*R*rnaI)/(kr*(1+(SC/ksc))+R)
      -(Pese*(ASF+SC)*rna)/(kese+ASF+SC)-delta*rna;
  rnaI' = (Pese*(ASF+SC)*rna)/(kese+ASF+SC)
      -(Pr*R*rnaI)/(kr*(1+(SC/ksc))+R)-k*rnaI+kk*rnaM-delta*rnaI;
  rnaM' = k*rnaI-kk*rnaM-delta*rnaM;
}
sample(rna, rnaI, rnaM);
```

During the simulation, we obtain the predicted equilibrium for the splice efficiency, see Fig. 4. Under our hypotheses, which include protein competition and compensation, the model correctly simulates the alternative splicing activity. This supports the hypotheses made in the model such as the role of the ESE and ESS2 binding sites.

3.3 Two Site Model: Global Modeling

A realistic model of alternative splicing has to reflect the combinatorial complexity discussed in Sect. 2.1. Assuming that regulation is modular [12], the single-site model may be seen as one module inside a larger framework. The qualitative validation given in Sect. 2.4 justifies the introduction of the single-site model into a larger scale model involving several splicing sites. To illustrate this, we consider the generic example of two splice acceptor sites (SA) associated with one donor site (SD), see Fig. 5.

The behavior at one splicing site can be captured by a single function, the splice efficiency, which depends on the protein concentrations. This function is used in a larger-scale global model that describes the choice between two acceptor sites A3 and A4. In the HIV-1 case, the A3 site is the default splicing site. Only if the splice efficiency `eff1` gets smaller than `eff2`, regulation switches to the other state.

Recent work [5] shows the linearity of the splicing kinetics. Thus, on the larger scale, we may consider splicing as a linear process described by two ordinary

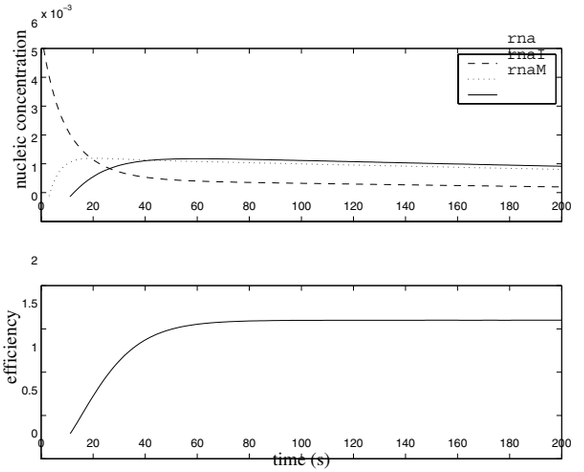


Fig. 4. Variation of the pool of RNA and the splice efficiency in the splicing reaction

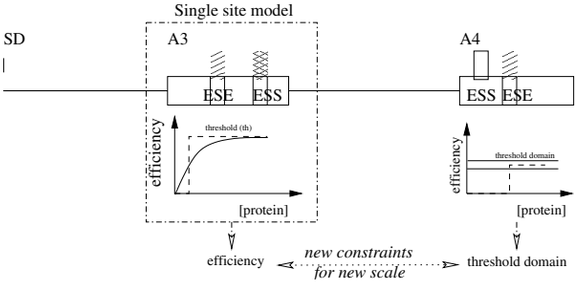


Fig. 5. Single-site model inside a general splicing regulation model

differential equations involving some kinetic constant. The first system represents the default behavior characterized by the constant `Ka4`. Following local changes on a single-site, the model may exhibit a different behavior, characterized by the kinetic constant `Ka3`. Thus, in this example, only the kinetic constants change, while the overall structure of the ODE system remains the same.

This default behavior can be naturally expressed in `Hybrid cc` using the combinator `unless(`c`)` A.

```
always {
    if (eff1 <= eff2)  {rna' = -Ka3*rna;
                        rnam1' = 0;
                        rnam2' = Ka3*rna;};
    unless ((eff1 <= eff2))  {rna' = - Ka4*rna;
                              rnam1' = Ka4*rna;
                              rnam2' = 0;};
}
```

Note that `unless`(c) A is not equivalent to `if` $\neg c$ `then` A. The second alternative will be chosen if the solver cannot verify that `(eff1 <= eff2)`. This may have *two* reasons:

- `(eff1 <= eff2)` is false, i.e. `(eff1 > eff2)`, or
- `(eff1 <= eff2)` is unknown (default behavior).

Simulation in `Hybrid cc` yields the behavior illustrated in Fig.6. We observe the switch from the first system of ODEs to the second when `eff2` passes the upper threshold for `eff1`.

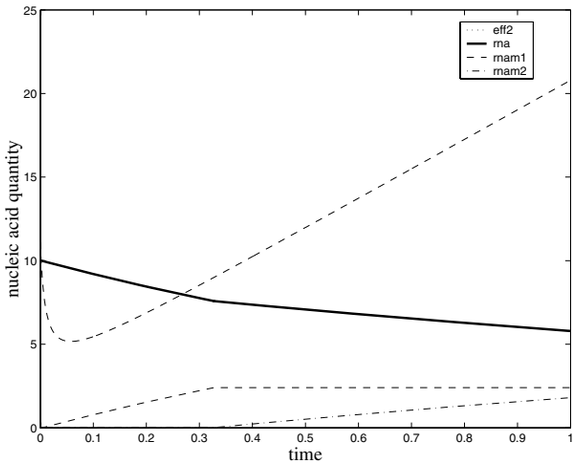


Fig. 6. Variation of RNA with a variation of splice efficiency

4 Conclusion and Further Research

Our approach combines mathematical and computational methods. Mathematical analysis allows us to validate the single-site model in a qualitative way. This is possible using the experimental data obtained in our group. The validation

shows the consistency of our biological hypotheses. Based on this, we can extract the splice efficiency as a suitable abstraction of the local behavior at one site inside a more global model involving different sites. For the experimental biologist, the single-site model may serve as a computational tool to evaluate his knowledge on a fine-grained biological process.

On the computational side, the constraint solving and default reasoning capabilities of `Hybrid cc` allow us to exploit as much as possible the incomplete knowledge that is typical for ongoing biological research. Indeed, default behavior may compensate the lack of experimental data. Thus, using constraint programming, we can delimit with our model the possible splicing behavior. Similar to mathematical analysis, constraint reasoning therefore may provide a powerful qualitative validation.

The combination of mathematical analysis and computational methods is also the key to the multiscale modeling developed in this paper. It leads to the qualitative validation represented by the extraction of the splice efficiency function. The splice efficiency characterizes the modularity of the regulation. Thus, the smaller-scale behavior is represented in the larger-scale model, based on the single-site splice efficiency. The extraction of a suitable criterion on the smaller scale is crucial to understanding an experimental process from a systems biology perspective. Furthermore, constraints can be used to handle the problem of missing data in a multiscale model. Different scales usually correspond to biological experiments yielding different types of results. Despite the variety of possible experiments, these must be integrated into a global model in order to better understand the biological process.

Multiscale modeling requires a close interaction between biological and computational approaches, as illustrated by Fig.7.

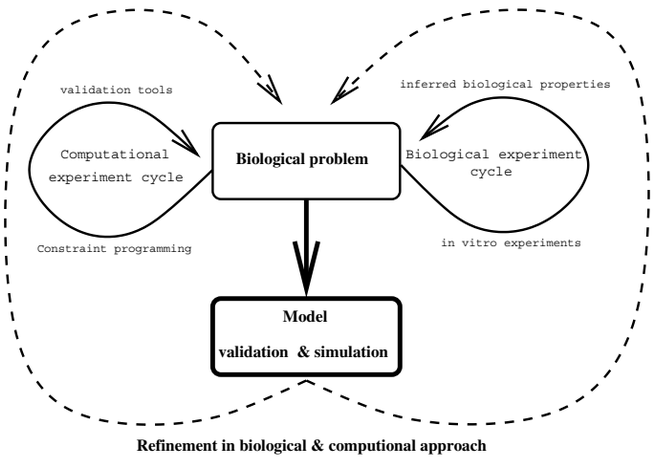


Fig. 7. Representation of ideal interactions for modeling biology

In the context of alternative splicing regulation, we are currently working on new experimental data for the quantitative validation of our models. On the computational side, we have integrated our model into a general HIV-1 model [11]. Preliminary results show that the modification of a splice constant may induce different behaviors in the HIV-1 life cycle model. Using the extended model, we may validate several biological hypotheses on the global effect of alternative splicing in the full HIV-1 life cycle.

References

[1] Bernard, O., and Gouzé J.-L.: Transient behavior of biological models as a tool of qualitative validation - Application to the Droop model and to a N-P-Z model. J. Biol. Syst, 4(3) (1996) 303–314 80

[2] Bockmayr, A., Courtois, A.: Modeling biological systems in hybrid concurrent constraint programming (Abstract). In 2nd Int. Conf. Systems Biology, ICSB'01, Pasadena, CA (2001) 76, 82

[3] Bockmayr, A., Courtois, A.: Using hybrid concurrent constraint programming to model dynamic biological systems. In 18th International Conference on Logic Programming, ICLP'02, Copenhagen Springer, LNCS 2401 (2002) 76, 82

[4] Caputi, M. A., Mayeda, M. A., Krainer, A. R., and Zahler, A. M.: hnRNP A/B proteins are required for inhibition of HIV-1 pre-mRNA splicing. EMBO 18(14) (1999) 4060–7 77

[5] Dautry, F., Weill, D.: Kinetic analysis of mRNA metabolism. In Interdisciplinary School on imaging, modelling and manipulating transcriptional regulatory networks, Ambleteuse (2002) 20/92 82

[6] Del Gatto-Konczak, F., Olive, M., Gesnel, M. C., and Breathnach, R.: hnRNP A1 recruited to an exon in vivo can function as an exon splicing silencer. Mol. Cell. Biol. 19 (1) (1999) 251–60 77

[7] Graveley, B. R., Hertel, K. J., Maniatis, T.: A systematic analysis of the factors that determine the strength of pre-mRNA splicing enhancers. EMBO 17(22) (1998) 6747–6756

[8] Graveley, B. R.: Sorting out the complexity of SR protein functions. RNA 6 (2000) 1197–1211 75

[9] Gupta, V., Jagadeesan, R., Saraswat V.: Computing with continuous change. Science of computer programming 30(1-2) (1998) 3–49 76, 81

[10] Gupta, V., Jagadeesan, R., Saraswat V., Bobrow, D. G.: Programming in hybrid constraint languages. In *Hybrid Systems II*, 226–251. Springer, LNCS 999 (1995) 76, 81, 82

[11] Hammond, B. J.: Quantitative Study of the Control of HIV-1 Gene Expression. J. Theor. Biol. 163 (1993) 199–221 86

[12] Hartwell, L. H., Hopfield, J. J., Leibler, S., Murray, A. W.: From molecular to modular cell biology. Nature 402 (1999) C47–C52 82

[13] Heinrich, R., Schuster, S.: *The regulation of cellular systems.* Int'l Thomson Publishing, New York (1996) 79

[14] Moore, M. J., Query, C. C., Sharp, P. A.: Splicing of precursors to mRNA by the spliceosome. In *The RNA World*, Cold Spring Harbor Laboratory Press (1993) 303–357 76

[15] O'Reilly, M., McNally, M. T., Beemon, K. L.: Two strong 5' splice sites and competing, suboptimal 3' splice sites involved in alternative splicing of human immunodeficiency virus type 1 RNA. Virology 213(2) (1995) 373–85 77
[16] Palsson, B.: The challenges of in silico biology. Nature Biotechnology 18 (2000) 1147 – 1150 76
[17] Purcell, D. F., Martin, M. A.: Alternative splicing of human immunodeficiency virus type 1 mRNA modulates viral protein expression, replication, and infectivity. J. Virol. 67(11)(1993) 6365–6378 77
[18] Ropers, D., Ayadi, L., Jacquenet, S., Méreau, A., Thomas, D., Mougin, A., Bilodeau, P., Stoltzfus, C. M., Gattoni, R., Stévenin, J., Branlant, C.: A complex regulation of the central A3 to A5 acceptor sites in HIV-1 RNA. In Eukaryotic mRNA Processing (2001) 77
[19] Saraswat, V. A.: *Concurrent constraint programming.* ACM Doctoral Dissertation Awards. MIT Press (1993) 81
[20] Saraswat, V. A., Jagadeesan, R., Gupta, V.: Foundations of timed concurrent constraint programming. In 9th Symp. Logic in Computer Science, LICS'94, Paris IEEE (1994) 71 – 80 81
[21] Saraswat, V. A., Jagadeesan, R., Gupta, V.: Timed default concurrent constraint programming. Journal of Symbolic Computation 22(5/6) (1996) 475–520 81
[22] Segel, L. A.: *Modelling dynamic phenomena in molecular and cellular biology.* Cambridge University Press (1984) 78
[23] Si, Z. H., Amendt, B. A., Stoltzfus, C. M.: Splicing efficiency of human immunodeficiency virus type 1 tat RNA is determined by both a suboptimal 3' splice site and a 10 nucleotide exon splicing silencer element located within tat exon 2. N. A. R. 25(4) (1997) 861–867
[24] Si, Z. H., Rauch, D., Stoltzfus, C. M.: The exon splicing silencer in human immunodeficiency virus type 1 Tat exon 3 is bipartite and acts early in spliceosome assembly. Mol. Cell. Biol 18(9) (1998) 5404–13
[25] Staffa, A., Cochrane, A.: The tat/rev intron of human immunodeficiency virus type 1 is inefficiently spliced because of suboptimal signals in the 3' splice site. J. Virol. 68(5) (1994) 3071–9
[26] Tang, H., Kuhen, K. L., Wong-Staal, F.: Lentivirus replication and regulation. Annu. Rev. Genet 33 (1999) 133–170 76
[27] Voit, E.: *Computational Analysis of Biochemical Systems.* Cambridge University Press (2000) 79

A Method for Estimating Metabolic Fluxes from Incomplete Isotopomer Information

Juho Rousu[1], Ari Rantanen[1], Hannu Maaheimo[2], Esa Pitkänen[1], Katja Saarela[1], and Esko Ukkonen[1]

[1] Department of Computer Science, University of Helsinki, Finland
juho.rousu@cs.helsinki.fi
[2] VTT Biotechnology, Finland
hannu.maaheimo@vtt.fi

Abstract. Metabolic flux estimation—the problem of finding out the rates of reactions in metabolic pathways—is an important problem area in the study of metabolism. The most accurate technique for this task today is the use of isotopic tracer experiments, where a mixture of differently isotope-labeled substrates is fed to a cell culture and the propagation of the labels is observed from the products and intermediate metabolites, where possible. We present a generic methodology for solving the fluxes of a metabolic network in a steady state. The method differs from most previous approaches by not making prior assumptions about the topology of the metabolic network. Also, only very mild assumptions are made about the available measurement data, for example, both positional enrichment and mass isotopomer data can be used.

1 Introduction

Metabolic flux estimation, the problem of determining intracellular reaction velocities, is an important problem in metabolic engineering [18], where the goal is to optimize the production of some metabolite in microbial cells. Knowledge of the steady-state reaction velocities, the fluxes, along different pathways is a prerequisite for pathway optimization [13]. Steady state flux estimation has potential to be an important tool in a wider context of systems biology as well, for example, in characterizing the physiology of the organism [6] and in metabolic reconstruction, where one aims at reverse engineering the metabolic network of an unfamiliar organism [10, 3, 1, 16].

Currently the most appealing framework for flux estimation is the use of isotopic tracer experiments, where one feeds the cell culture with a predefined mixture of natural and ^{13}C-labeled nutrients. The fate of the ^{13}C atoms can be observed by measuring the *isotopomer distributions* of metabolic products and intermediates [17] with NMR [19, 8] or mass spectrometry [2, 21, 5, 11].

The mathematical tools for relating the isotopomer distributions of metabolites to the fluxes have developed over the last decade. On one hand, the linear algebraic framework for flux analysis, relying on elementary balances of chemical compounds, has been extended to isotopomers [9, 14, 18, 7] to analyze key

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 88–103, 2003.

pathways in metabolism. This approach attempts to solve the fluxes explicitly. However, a rigorous theory of these extensions, enabling automatic flux estimation given an arbitrary metabolic network, has been lacking. On the other hand, an iterative approach to flux estimation has been studied by Wiechert, Möllney, Isermann, Wurzel and de Graaf [20]. This framework, to the authors' knowledge, is the only existing method that is not tied to a particular metabolic topology.

In this paper, we propose a generic methodology for flux estimation in a direct manner—as opposed to the iterative method, so that the (possibly partial) solution is given in terms of linear constraints to the flux values, coupled with estimates of the error. The proposed framework is completely free of topological assumptions; the user can input any network topology to base the analysis upon. In addition, we make only very mild *prior* assumptions of the kind and quality of isotopomer and metabolite data, although the quality of the solution is affected by the data. In particular, we allow as measurements any set of linear constraints to the isotopomer distribution. This means that both NMR and mass spectrometric data can be used in this framework.

The structure of this article is the following. In Section 2 we present the terminology to be used in the article and review the basic principles underlying isotopomer tracing experiments. In Section 3 we present our approach to flux estimation, relying on the concept of isotopomer vector space and its subspaces. Section 4 gives some test results in analyzing glycolysis with generated data. Section 5 describes the implementation status of the software and directions for further work, and Section 6 presents some conclusions.

2 Analyzing Isotope Tracing Experiments

Let $\mathcal{M} = \{M_1, \ldots, M_m\}$ denote the set of metabolites, $\mathcal{L} = \{1, \ldots, L\}$ a numbering of all *carbon locations*, and $\mathcal{R} = \{\rho_1, \ldots, \rho_n\}$ the set of reactions in a metabolic network. Ignoring all other atoms but carbons, we associate the metabolite M with the set of its carbon locations: $M = \{M(1), \ldots, M(|M|)\} \subset \mathcal{L}$. A *reaction* $\rho = (\boldsymbol{\alpha}, \lambda)$ consists of a vector $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{|\mathcal{M}|}]^T \in \mathbb{Z}^{|\mathcal{M}|}$ of stoichiometric coefficients—stating the number of molecules consumed ($\alpha_i < 0$) or produced ($\alpha_i > 0$) in one reaction event – and a *carbon mapping* $\lambda : R_\lambda \to P_\lambda$ between *reactant locations* $R_\lambda \subset \mathcal{L}$ and *product locations* $P_\lambda \subset \mathcal{L}$.

We associate with each location $l \in \mathcal{L}$ a binary random variable *label* $\ell_l : \mathtt{C}_l \to \{0, 1\}$, where $\mathtt{C}_l$ is a fixed pool of carbons occupying the location l. The label is defined as $\ell_l(\mathtt{c}) = 1$ if and only if a (randomly drawn) carbon $\mathtt{c}$ is ^{13}C isotope. The *labeling* of a set of carbons $\mathtt{c}_1, \ldots, \mathtt{c}_k$ (e.g., a metabolite or its fragment) occupying locations $F = \{F(1), \ldots, F(k)\}$ is the sequence $\ell_{F(1)}(\mathtt{c}_1) \cdots \ell_{F(k)}(\mathtt{c}_k) \in \{0, 1\}^k$. To denote different labelings of a set of carbons we use the shorthands $^{\mathbf{b}}F = {}^{j}F$, where $\mathbf{b} = \mathtt{b}_1 \cdots \mathtt{b}_k$ is the k bits long binary representation of number j.

A pool $\mathtt{C}_M$ of molecules $(\mathtt{c}_1, \ldots, \mathtt{c}_{|M|})$, $\mathtt{c}_i \in \mathtt{C}_{M(i)}$, is associated with each metabolite M. The probability of randomly drawing from this pool a molecule

with labeling $\mathbf{b} \in \{0,1\}^{|M|}$ is

$$\mathbb{P}\{^{\mathtt{b}}M\} = \frac{|\{(\mathtt{c}_1,\ldots,\mathtt{c}_{|M|}) \in \mathtt{C}_M | \ell_{M(i)}(\mathtt{c}_i) = \mathtt{b_i}, \forall M(i) \in M\}|}{|\mathtt{C}_M|}.$$

The *isotopomer distribution* of the metabolite is then the vector

$$\mathbb{I}_M = [\mathbb{P}\{^0M\}, \mathbb{P}\{^1M\} \ldots, \mathbb{P}\{^{2^{|M|}-1}M\}]^T \in [0,1]^{2^{|M|}},$$

satisfying $\mathbb{P}\{^{\mathbf{b}}M\} \geq 0$, for all $\mathbf{b}$, and $\sum_{\mathbf{b}} \mathbb{P}\{^{\mathbf{b}}M\} = 1$. The isotopomer distribution of a metabolite *fragment* $F = \{F(1), \ldots F(|F|)\} \subset M$, also called *cumulative isotopomer distribution* or *cumomer distribution* [20], is defined in analogous manner by

$$\mathbb{P}\{^{\mathtt{b}_1\cdots\mathtt{b}_{|\mathtt{F}|}}F\} = \frac{|\{(\mathtt{c}_1,\ldots,\mathtt{c}_{|M|}) \in \mathtt{C}_M | \ell_{F(j)}(\mathtt{c}_i) = \mathtt{b_j}, \forall M(i) = F(j)\}|}{|\mathtt{C}_M|}$$

and $\mathbb{I}_F = [\mathbb{P}\{^0F\}, \mathbb{P}\{^1F\} \ldots, \mathbb{P}\{^{2^{|F|}-1}F\}]^T \in [0,1]^{2^{|F|}}$.

2.1 Flux Estimation and ^{13}C Isotopomer Tracing Experiments

Consider an intermediate metabolite M_i, produced by reactions $\rho_1, \ldots, \rho_r$ and consumed by reactions $\rho_{r+1}, \ldots, \rho_s$. Assuming that the system is in metabolic steady-state, the reaction rates v_j of the reactions ρ_j – the *fluxes*—satisfy

$$\sum_{j=1}^{r} v_j \alpha_{ji} = -\sum_{i=r+1}^{s} v_j \alpha_{ji}, \tag{1}$$

stating that the flows producing and consuming the metabolite are of equal size. Unfortunately, equations of these kind are not sufficient to uniquely determine the fluxes v_j if there are alternative routes in between two metabolites in the metabolic network. In reality such alternative routes of course occur.

In isotopomer tracing experiments one feeds differentially labeled substrates to the cell culture in some predefined fractions, for example, 90% normal glucose and 10% ^{13}C-glucose. The rationale behind isotope tracing is that the labels usually combine differently depending on the metabolic route that was used. Thus, by observing the isotopomer distributions of the intermediates and products one tries to deduce the flux ratios among the alternative pathways.

Assuming that the metabolic system is in an isotopomeric steady-state—meaning that the isotopomer distributions do not change over time—the above balance holds individually for each labeling. This can be written as an equation system [18, 7]:

$$\sum_{j=1}^{r} v_j \alpha_{ji} \mathbb{I}_{ji} = -\sum_{j=r+1}^{s} v_j \alpha_{ji} \mathbb{I}_{M_i} \tag{2}$$

where $\mathbb{I}_{ji}$ denotes the isotopomer distribution of the flow between ρ_j and M_i, and $\mathbb{I}_{M_i}$ denotes the isotopomer distribution of the metabolite (and the flows

consuming it). This system has rank $1 \leq r_i \leq 2^{|M_i|}$, so the fluxes are more likely to be fully determined than with (1).

There are two difficulties in applying (2) in practise: First, due to the low concentration, intermediate metabolites are difficult to measure, although mass spectrometry seems to be a promising tool in this respect [17]. Second, NMR and mass spectrometers (even tandem MS) rarely uncover the isotopomer distributions in their entirety.

Methods involving incomplete measurements in such balance equations have been considered by [18, 14], but only tied to a particular topology (such as the TCA cycle), apparently due to the difficulty in deriving general expressions for these balances and the constituent isotopomer distributions. A generic approach to flux estimation—free from topological assumptions and capable of handling incomplete data—has been described by Wiechert et al. [20]. They solve the problem iteratively, starting from an initial guess of the fluxes, computing predicted isotopomer distributions that should result with those fluxes, comparing the predicted isotopomer distributions to the measured ones, altering the fluxes based on the observed differences, predicting isotopomer distributions, and so on. In that approach, computing isotopomer distributions given the fluxes is non-trivial.

Our method, described in the next section, shares the genericity of the above approach by not being tied to any particular topology and also admitting incomplete measurements. The method differs from the method by Wiechert et al. by solving the fluxes non-iteratively. The benefit is that we get a closed form solutions for the fluxes or at least constraints limiting the set of solutions.

3 Direct Flux Estimation Using Incomplete Isotopomer Information

In practical situations, we may not know the relevant isotopomer distributions in their entirety, for example, if the measurement data consists of tandem mass spectrometric or positional enrichment data. In that case, (2) cannot be applied, since it relies on completely measured isotopomer distributions.

To generalize the approach, we associate with each labeling $\mathbf{b}$ of a metabolite M a vector $\boldsymbol{e}_{\mathbf{b}} \in \{0,1\}^{2^{|M|}}$ that contains 0's as all other components except the $\mathbf{b}$'th location. The set of vectors $\boldsymbol{e}_0, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_{2^{|M|}-1}$ form the standard basis of the *isotopomer space* $\mathcal{I}_M = \mathbb{R}^{2^{|M|}}$ of metabolite M. This representation unifies many concepts previously used in analyzing isotopic tracer experiments:

- An isotopomer distribution is a point $\boldsymbol{x} = [x_1, \ldots, x_{|M|}]^T \in \mathcal{I}_M$ satisfying $x_j \geq 0$, $\sum_j x_j = 1$.
- The trivial isotopomer measurement, stating that the isotopomer frequencies sum up to unity is represented by the constraint $\boldsymbol{i}_M^T \boldsymbol{x} = \sum_j x_j = 1$, where $\boldsymbol{i}_M$ is a length-$|M|$ vector of ones. Since this constraint always holds, it can be added to any set of constraints to the isotopomer distribution.

- A cumomer distribution related to a fragment $F \subset M$ lies in a subspace $\mathcal{U}_F \subset \mathcal{I}_M$ spanned by the vectors $\boldsymbol{i}_M, \boldsymbol{u}_{F,0}, \ldots, \boldsymbol{u}_{F,2^{|F|}-1}$, where

$$\boldsymbol{u}_{F,\mathbf{a}} = \sum_{\{\mathtt{a_j}=\mathtt{b_i},\ \text{if } F(j)=M(i)\}} \boldsymbol{e}_{\mathbf{b}}.$$

- A *mass isotopomer distribution*—the relative frequency distribution of equal mass labelings— lies in the subspace $\mathcal{W}_M \subset \mathcal{I}_M$ spanned by the vectors $\boldsymbol{i}_M, \boldsymbol{w}_0, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{|M|}$, where $\boldsymbol{w}_i = \sum_{weight(\mathbf{b})=i} \boldsymbol{e}_{\mathbf{b}}$, and $weight(\mathtt{b_1} \cdots \mathtt{b_{|M|}}) = \sum_j \mathtt{b_j}$. Tandem mass spectrometers produce data that, in addition to the above, also contains mass isotopomer distributions of some of the fragments, so we get extra basis vectors $\boldsymbol{w}_{i,F} = \sum_{weight(\mathbf{b})=i} \boldsymbol{u}_{F,\mathbf{b}}$. where $\mathbf{b} = \mathtt{b_1} \cdots \mathtt{b_{|F|}}$ is a labeling of the fragment F.
- *Positional labeling enrichment* data lie in the subspace $\mathcal{P}_M \subset \mathcal{I}_M$, spanned by the vectors $\boldsymbol{i}_M, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_{|M|}$, where each vector $\boldsymbol{p}_j = \sum_{\mathtt{b_j}=1} \boldsymbol{e}_{\mathbf{b}}$ denotes the sum of labeling frequencies where location j contains a ^{13}C isotope.

Using the vector space representation, instead of restricting ourselves to complete isotopomer distributions $\mathbb{I}_M$, we can handle linear *isotopomer constraints* of the form $S^T\mathbb{I}_M = \boldsymbol{d}$ (i.e., different linear combinations of isotopomers as given by S), where the matrix $S = [\boldsymbol{s}_1 \cdots \boldsymbol{s}_r]$ contains basis vectors spanning some subspace $\mathcal{S} \subset \mathcal{I}_M$. To see how that helps in flux estimation, pre-multiply (2) by S^T:

$$\sum_{j=1}^{k} v_j \alpha_{ji} S^T \mathbb{I}_{ji} = -\sum_{j=k+1}^{l} v_j \alpha_{ji} S^T \mathbb{I}_{M_i}. \tag{3}$$

Thus, (2) implies a balance in any subspace $\mathcal{S}$.

One can view (3) as a generalization of both (1) and (2). To get the former, set $S = [\boldsymbol{i}_M]$, and to get the latter set $S = I_{|M|}$, the $|M| \times |M|$ identity matrix. Thus (3) constrains the fluxes at least as strongly as (1) and at most as strongly as (2).

The challenge in applying (3) lies in finding the *isotopomer constraints* $\boldsymbol{d}_{ji} = S^T\mathbb{I}_{ji}$ for each flow assuming that some constraints are known for some of the metabolite pools. We aim at obtaining for each metabolic junction a balance where the associated S matrix has as high rank as possible—ideally rank of $2^{|M_i|}$ which would make (3) coincide with (2). In practise, the rank depends on which metabolites and which kind of isotopomer constraint data is available.

Our method can be decomposed into the following three steps

1. Computing for each metabolite M_i that is produced by two or more reactions an isotopomer constraint $S_i^T\mathbb{I}_{M_i} = \boldsymbol{d}_i$ and for each path producing the metabolite a constraint $S_{ji}^T\mathbb{I}_{ji} = \boldsymbol{d}_{ji}$ where the rank of S_{ji} is as high as possible. The matrices S_i and the S_{ji}'s are not necessarily the same—not even of the same rank—as required by (3).
2. Computing for the same M_i's the intersection $\mathcal{S} = \mathcal{S}_i \cap_{j=1}^{k} \mathcal{S}_{ji}$ of the column spaces of S_i, and of $S_{ji}, 1 \leq j \leq k$, and projecting each constraint to the intersection space to finally obtain (3).

3. Collecting the balance equations of different metabolites to a common matrix and solving the resulting linear system of equations.

The first step is described in Sections 3.1 and 3.2, and the last two steps in Section 3.3.

3.1 Propagation of Isotopomer Information over Reactions

Consider an arbitrary reaction ρ, converting reactant metabolites $M_1, \ldots, M_k$ to product metabolites $M_{k+1}, \ldots, M_l$. Given arbitrary linear constraints $S_1^T \mathbb{I}_{M_1} = \boldsymbol{d}_1$, ..., $S_k^T \mathbb{I}_{M_k} = \boldsymbol{d}_k$ to the reactant isotopomer distributions—these can be measurements or intermediate computation results—we would like to compute constraints $S_{k+1}^T \mathbb{I}_{M_{k+1}} = \boldsymbol{d}_{k+1}, \ldots, S_l^T \mathbb{I}_{M_l} = \boldsymbol{d}_l$ to the isotopomer distributions of the products.

For this end, take an arbitrary reactant-product pair M, M'. In this short paper we assume that the carbon mappings are bijections[1], so that the inverse λ^{-1} is defined. Denote by $\lambda(M)$ and $\lambda^{-1}(M')$ the *image* of M and the *pre-image* of M' in the mapping λ, respectively. From the carbon mapping λ we can compute the destiny and the origin of each carbon. The carbons of M' that originate from M are in the *product fragment* $F' = \lambda(M) \cap M'$ and the carbons of M that are destined to M' are contained in the *reactant fragment* $F = \lambda^{-1}(M') \cap M$.

Consider now a reaction event converting a reactant molecule to a product molecule. It is easy to see that, if the carbon mapping λ is a bijection, a reactant fragment F with the labeling $\mathbf{b}$ of is converted to a product fragment F' with labeling $\mathbf{a}$ that is a permutation of $\mathbf{b}$. Therefore, for each reactant-product fragment pair one can assign a permutation $\pi_{\lambda,F} : \{0,1\}^{|F|} \to \{0,1\}^{|F'|}$, defined by $\pi_{\lambda,F}(\mathtt{b}_1 \cdots \mathtt{b}_{|F|}) = \mathtt{a}_1 \cdots \mathtt{a}_{|F|}$, where $\lambda(F(i)) = F'(j)$ implies $\mathtt{b}_i = \mathtt{a}_j$. Furthermore, one can construct a permutation matrix $\Pi_{\lambda,F}$ defined by $(\Pi_{\lambda,F})_{i,j} = 1$ if $\pi_{\lambda,F}(i) = j$ and 0, otherwise.

We will further assume that the reactions are *simple* in the sense that carbons within a single fragment remain physically attached (i.e. no carbon-carbon bonds are broken) throughout the reaction. Note that any complex reaction can be broken into a set of simple reactions, so this is not a severe restriction. Given this assumption, the labeling frequencies within the fragments remain the same in the reaction, thus $\mathbb{P}\{{}^{\mathbf{b}}F)\} = \mathbb{P}\{{}^{\pi_{\lambda,F}(\mathbf{b})}F'\}$ for any labeling $\mathbf{b}$ of the reactant fragment, making the corresponding *cumomer distributions* coincide (modulo permutation):

$$U_F^T \mathbb{I}_M = \mathbb{I}_F = \Pi_{\lambda,F}^{-1} \mathbb{I}_{F'} = \Pi_{\lambda,F}^{-1} U_{F'}^T \mathbb{I}_{M'} \tag{4}$$

However, above we only assumed that a constraint $S^T \mathbb{I}_M = \boldsymbol{d}$ is known for $\mathbb{I}_M$, where the column space of S is some subspace $\mathcal{S} \subset \mathcal{I}_M$. Thus, (4) cannot be used directly. Instead, let us denote by $S_\perp$ and $U_{F,\perp}$ orthonormal matrices with column spaces $\mathcal{S}$ and $\mathcal{U}_F$, respectively.

[1] A carbon mapping λ_j involving symmetrical metabolites or a reaction with stoichiometric coefficients $|\alpha_{ji}| > 1$ is not a bijection. However, these cases can be handled with only slight modifications [12].

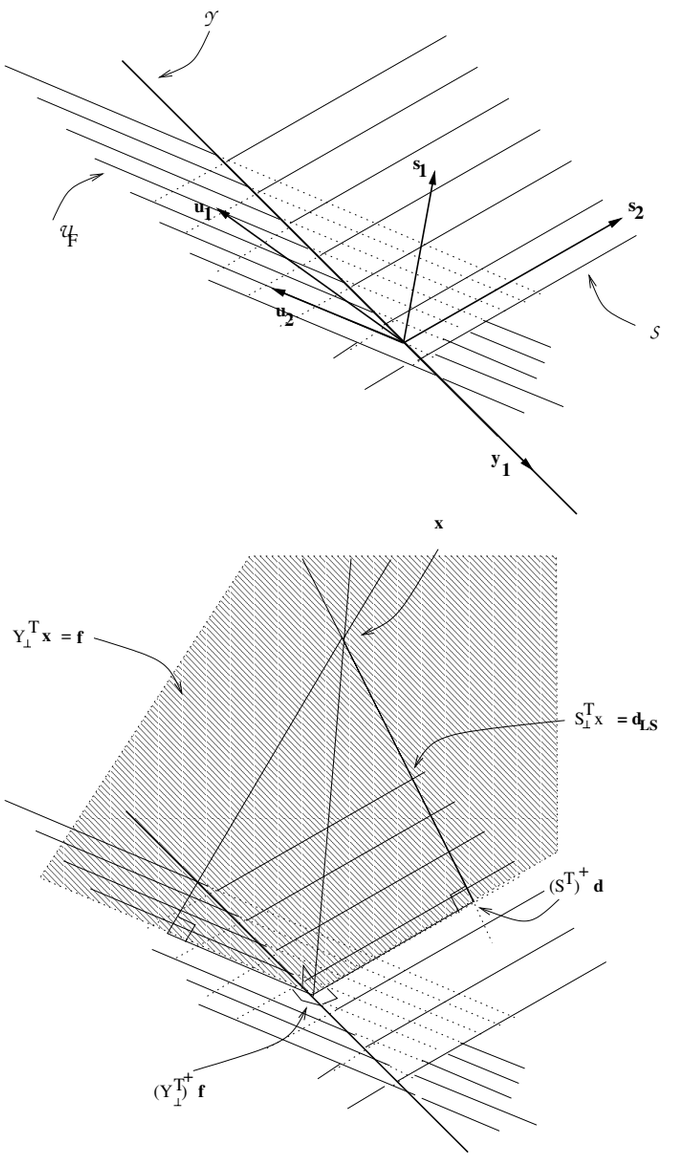


Fig. 1. At top, the column spaces $\mathcal{S}$ and $\mathcal{U}_F$ of matrices $S = [\boldsymbol{s}_1, \boldsymbol{s}_2]$ and $U_F = [\boldsymbol{u}_1, \boldsymbol{u}_2]$ together with their intersection $\mathcal{Y}$, the column space of $Y = [\boldsymbol{y}_1]$, are depicted. At bottom, the orthogonal projection of isotopomer distribution $\boldsymbol{x}$ into $\mathcal{Y}$ is depicted. The same projection can be computed directly or via either of the subspaces $\mathcal{S}$ and $\mathcal{U}_F$. The set of solutions consistent with $Y_\perp^T\boldsymbol{x} = \boldsymbol{f}$ is shown in grey

We compute the vector space intersection $\mathcal{Y} = \mathcal{S} \cap \mathcal{U}_F$ (see Figure 1), and let columns of $Y_\perp$ compose an orthonormal basis for it.

The orthogonal projection [15] of an arbitrary vector $\boldsymbol{x} \in \mathcal{I}_M$ to $\mathcal{Y}$ is given by

$$Y_\perp Y_\perp^T \boldsymbol{x} = Y_\perp Y_\perp^T S_\perp S_\perp^T \boldsymbol{x} = Y_\perp Y_\perp^T U_{F,\perp} U_{F,\perp}^T \boldsymbol{x},$$

stating that the same projection can be computed either directly, or via the space $\mathcal{S}$, or via the space $\mathcal{U}_F$. Noticing that the product $A^T A$ is an identity matrix for any matrix A with orthonormal columns, the above can be simplified by premultiplication with $Y_\perp^T$ to

$$Y_\perp^T \boldsymbol{x} = Y_\perp^T S_\perp S_\perp^T \boldsymbol{x} = Y_\perp^T U_{F,\perp} U_{F,\perp}^T \boldsymbol{x}. \tag{5}$$

To apply the projection, we need to handle the case where $S^T \mathbb{I}_M = \boldsymbol{d}$ is an inconsistent system of equations, having no isotopomer distribution vectors $\mathbb{I}_M$ that satisfy the equations. Therefore, we translate the system into one that is consistent with the least-squares solution of the original system. If the original system already is consistent, it remains as it is. This is done as follows. Denoting by A^+ the *Moore-Penrose pseudo-inverse* [15] of matrix A, the vector $\boldsymbol{x}_{LS} = (S^T)^+ \boldsymbol{d}$ is a least squares solution to $S^T \mathbb{I}_M = \boldsymbol{d}$. Moreover, by the properties of A^+, $\boldsymbol{x}_{LS}$ is in fact the orthogonal projection into $\mathcal{S}$ of any vector $\boldsymbol{x}$ consistent with $S_\perp^T \boldsymbol{x} = \boldsymbol{d}_{LS}$, where $\boldsymbol{d}_{LS} = S_\perp^T \boldsymbol{x}_{LS}$.

Above the columns of matrix $S_\perp$ are assumed to form an orthonormal basis to $\mathcal{S}$; this basis can be obtained from S by using standard matrix algebra. Computing $U_{F,\perp}$ from U_F is even more straightforward: since for any fragment F, the columns of U_F are already orthogonal, we only need to normalize the columns by multiplication with a diagonal matrix containing the inverses of the column norms: $U_{F,\perp} = D_F^{-1} \cdot U_F$, where $D_F = diag(\| \boldsymbol{u}_{F,\mathbf{0}} \|, \ldots, \| \boldsymbol{u}_{F,\mathbf{2^F-1}} \|)$.

Applying the above to (4) we get

$$\begin{aligned} \boldsymbol{d}_{F'} = Y_\perp^T S_\perp \boldsymbol{d}_{LS} = Y_\perp^T U_{F,\perp} U_{F,\perp}^T \mathbb{I}_M = \\ Y_\perp^T U_{F,\perp} D_F^{-1} U_F^T \mathbb{I}_M = Y_\perp^T U_{F,\perp} D_F^{-1} \Pi_{\lambda,F}^{-1} U_{F'}^T \mathbb{I}_{M'} \\ = S'^T \mathbb{I}_{F'}. \end{aligned} \tag{6}$$

We immediately see that the equation is of the required form, an isotopomer constraint to the fragment F', denoting $S' = (Y_\perp^T U_{F,\perp} D_F^{-1} \Pi_{\lambda,F}^{-1})^T$.

By the above method, we can compute a constraint to the isotopomer distribution of each product fragment of metabolite M'. To determine a constraint to the isotopomer distribution of $\mathbb{I}_{M'}$ as a whole, we still need to combine the fragment constraints. Let, therefore, $F', F'' \subset M'$ be two arbitrary product fragments with constraints $S'^T \mathbb{I}_{F'}$ and $S''^T \mathbb{I}_{F''}$, respectively. Let us compute a constraint for their union $F = F' \cup F''$.

Assuming that the reactions draw their reactants independently from their respective pools, the isotopomer distributions of the fragments are independent as well. Thus, the probability of observing a labeling $\mathbf{b}$ for F is given by the product

$$\mathbb{P}\{^{\mathbf{b}}F\} = \mathbb{P}\{^{\mathbf{b}'}F'\} \cdot \mathbb{P}\{^{\mathbf{b}''}F''\}$$

where the labelings $\mathbf{b}'$ and $\mathbf{b}''$ match $\mathbf{b}$:

$$\mathtt{b_i} = \begin{cases} \mathtt{b'_k}, & \text{if } F(i) = F'(k) \text{ for some } k, \text{ and,} \\ \mathtt{b''_k}, & \text{if } F(i) = F''(k) \text{ for some } k. \end{cases} \tag{7}$$

Consider now the constraints ${\boldsymbol{s}'}^T \mathbb{I}_{F'} = d'$ and ${\boldsymbol{s}''}^T \mathbb{I}_{F''} = d''$ corresponding to an arbitrary pair of rows in matrices S' and S''. The product

$$d' \cdot d'' = ({\boldsymbol{s}'}^T \mathbb{I}_{F'})({\boldsymbol{s}''}^T \mathbb{I}_{F''}) = \left(\sum_{\mathbf{b}'} {s'}_{\mathbf{b}'} \mathbb{P}\{{}^{\mathbf{b}'}F'\} \right) \cdot \left(\sum_{\mathbf{b}''} {s''}_{\mathbf{b}''} \mathbb{P}\{{}^{\mathbf{b}''}F''\} \right),$$

by above reasoning, simplifies into

$$d = d' \cdot d'' = \sum_{\mathbf{b}', \mathbf{b}''} {s'}_{\mathbf{b}'} \cdot {s''}_{\mathbf{b}''} \mathbb{P}\{{}^{\mathbf{b}'}F'\} \mathbb{P}\{{}^{\mathbf{b}''}F''\} = \sum_{\mathbf{b}} s_{\mathbf{b}} \mathbb{P}\{{}^{\mathbf{b}}F\} = \boldsymbol{s}^T \mathbb{I}_F,$$

choosing here $\mathbf{b}$ according to (7) and denoting $s_{\mathbf{b}} = {s'}_{\mathbf{b}'} \cdot {s''}_{\mathbf{b}''}$.

Similar constraints can be computed for each pair of rows and collected to a matrix $S^T \mathbb{I}_F = \boldsymbol{d}_F$. The whole process can be applied iteratively to a union of product fragments $\cup_{t<k} F_t$ and the next unprocessed fragment F_k, $2 \leq k \leq l$, finally resulting in a constraint ${S'}^T \mathbb{I}_{M'} = \boldsymbol{d}_{M'}$ to the product metabolite $M' = \cup_{k=1}^{l} F_k$.

The inverse problem of the above, that of computing an isotopomer constraint to the reactant isotopomer distributions from the product constraints, is solved, for the first part, analogously. The relationship between the reactant and product fragment labeling distributions remains essentially the same: given a constraint ${S'}^T \mathbb{I}_{F'} = \boldsymbol{d'}$ to the product fragment F', from (4), by computing the intersection $\mathcal{Y}' = \mathcal{S}' \cap \mathcal{U}_{F'}$ and projecting the least-squares solution $\boldsymbol{d'}_{LS}$ to the intersection space, we obtain a isotopomer constraint to the reactant fragment F

$$\boldsymbol{d}_F = {Y'_\perp}^T S'_\perp \boldsymbol{d'}_{LS} = {Y'_\perp}^T U_{F',\perp} D_{F'}^{-1} \Pi_{\lambda,F} U_F^T \mathbb{I}_M = S^T \mathbb{I}_F,$$

where we denote $S = ({Y'_\perp}^T U_{F',\perp} D_{F'}^{-1} \Pi_{\lambda,F})^T$.

However, the fragment labeling probabilities cannot be multiplied to obtain a distribution of the reactant: in the general case, the isotopomer distribution of the reactant cannot be guaranteed to be a *product distribution*, as is the case in forward propagation. Hence, the isotopomer constraints $S_k^T \mathbb{I}_{F_k} = \boldsymbol{d}_{F_k}$ computed to the reactant fragments $F_1, \ldots, F_k$ can only be grouped into a common matrix,

$$S_i^T \mathbb{I}_{M_i} = \begin{bmatrix} S_1^T U_{F_1}^T \\ \vdots \\ S_l^T U_{F_l}^T \end{bmatrix} \cdot \mathbb{I}_{M_i} = \begin{bmatrix} \boldsymbol{d}_{F_1} \\ \vdots \\ \boldsymbol{d}_{F_l} \end{bmatrix}$$

representing the fact that each fragment constraint need to be satisfied separately.

3.2 The Propagation Algorithm

Using the above methods we can compute constraints to the isotopomer distributions of the products of a reaction from arbitrary linear constraints to the reactant distributions and vice versa. The process can be applied iteratively along an *unbranched* pathway—where no metabolite is produced by more than one reaction.

The propagation methods described above are used for networks with junctions in the following manner. Let us denote by $\mathcal{M}_B \subset \mathcal{M}$ the metabolites produced by two or more reactions.

1. Starting from every external product and every intermediate metabolite just above a metabolite $M \in \mathcal{M}_B$, propagate isotopomer constraints backward, until the next metabolite $M' \in \mathcal{M}_B$ is encountered. For metabolites M'' that are consumed by two or more reactions, the propagation will produce a separate set of constraints to $I_{M''}$, collected into a single matrix equation.
2. Next, starting from the metabolites $M \in \mathcal{M}_B$ and from the external reactants, we propagate isotopomer information forward until the next metabolite $M' \in \mathcal{M}_B$ is encountered. By this process, we obtain isotopomer constraints for each path entering a junction.

The propagation works in a synchronous manner so that, in backward propagation, only when constraints of all reactions consuming a metabolite M have been computed, the constraint to $\mathbb{I}_M$ is computed, and only when constraints to all products of a reaction are computed, the constraints are propagated backward over the reaction. Similarly in forward propagation, only when the constraints to all reactants are known, the forward propagation proceeds over the reaction.

The propagation frequently produces systems of equations $S^T\mathbb{I}_M = \boldsymbol{d}$ that have linearly dependent rows, and hence the systems are potentially inconsistent. These cases are handled in the same manner as described in Section 3.1: the equation system is transformed into a system $S_\perp^T\mathbb{I}_M = \boldsymbol{d}_{LS}$ that is consistent with the least-squares solution of the original system, and the coefficient matrix $S_\perp$ is chosen to have orthonormal columns.

The use of both backward and forward propagation may seem superfluous, but notice that separate paths consuming the metabolite may give independent information regarding the metabolite's isotopomer distribution. Propagating this extra information forward along the other paths may help in solving the fluxes in downstream junctions.

3.3 Forming and Solving the System of Balance Equations

After the propagation of the isotopomer information, we have for each metabolic junction a set of constraints $S_{ji}^T\mathbb{I}_M = \boldsymbol{d}_{ji}$, one for each path producing the metabolite, and a constraint $S_i^T\mathbb{I}_{M_i} = \boldsymbol{d}_i$ to the isotopomer distribution of the metabolite itself. However, the constraints do not yet conform to (3); the matrices S_i and $S_{ji}, 1 \leq j \leq k$ are not necessarily the same. To transform the constraints to the required form, we compute the intersection $\mathcal{Y} = \mathcal{S}_i \cap_{j=1}^k \mathcal{S}_{ji}$,

of the column spaces of the matrices and compute an orthonormal basis $Y_\perp = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{r_i}]$ for it. Then the orthogonal projections (assuming here orthonormal coefficient matrices) $Y_\perp^T S_{ji} S_{ji}^T \mathbb{I}_M = Y_\perp^T S_{ji} \boldsymbol{d}_{ji} = \boldsymbol{z}_{ji}$ and $Y_\perp^T S_i S_i^T \mathbb{I}_M = Y_\perp^T S_i \boldsymbol{d}_i = \boldsymbol{z_i}$ lie in $\mathcal{Y}$ and (3) applies:

$$\sum_{j=1}^{k} v_j \alpha_{ji} \boldsymbol{z}_{ji} = -\sum_{k+1}^{l} v_i \alpha_{ji} \boldsymbol{z}_i$$

The constraints for the metabolites are combined into a common equation system

$$A\ \boldsymbol{v} = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_m \end{bmatrix} \tag{8}$$

where, for metabolites with more than one producer, there is a block

$$A_i \boldsymbol{v} = \begin{bmatrix} \alpha_{1i}(\boldsymbol{z}_{1i})_1 & \cdots & \alpha_{ni}(\boldsymbol{z}_{ni})_1 \\ \vdots & \ddots & \vdots \\ \alpha_{1i}(\boldsymbol{z}_{1i})_{r_i} & \cdots & \alpha_{ni}(\boldsymbol{z}_{ni})_{r_i} \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} b_{i1} \\ \vdots \\ b_{ir_i} \end{bmatrix} = \boldsymbol{b_i}.$$

The right-hand side is defined by

$$\boldsymbol{b}_i = \begin{cases} \boldsymbol{0} & \text{if } M_i \text{ is an intermediate metabolite, and} \\ q_i \boldsymbol{z}_i & \text{otherwise,} \end{cases}$$

where the coefficients α_{ji} correspond to 1 and q_i is the net consumption/production rate of the metabolite. For other metabolites, the block is given by

$$A_i \boldsymbol{v} = [\alpha_{1i}, \ldots \alpha_{ni}] \cdot \boldsymbol{v} = [q_i] = \boldsymbol{b}_i.$$

The system (8) can be solved with standard linear algebra [15]. If the system is less than full rank, one, however, can only output an equation system that describes the set of all feasible solutions.

4 Experiments

We tested the described flux estimation method in the analysis of fluxes in glycolysis (Fig. 2). The network contains 26 metabolites and 16 reactions, 12 of which are bidirectional, resulting in 28 fluxes to be estimated.
The test setting was the following:

1. The 13CFlux software [20] was used to simulate complete isotopomer distributions of the metabolites, given pre-assigned fluxes, the 'correct' fluxes.
2. Different subsets of metabolites were randomly chosen as those 'having measurements'. Complete isotopomer distributions of those were given to our software as input.

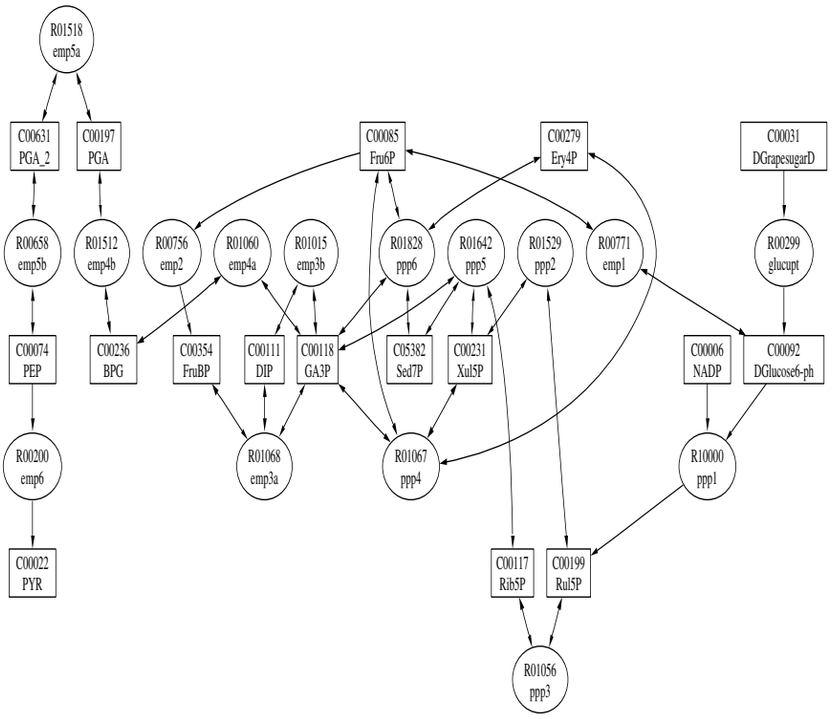


Fig. 2. Glycolysis—glucose uptake, Embden-Meyerhof-Parnas (emp1–6) and pentose-phosphate pathways (ppp1–6)—as analyzed in the experiments. The directionality of the reactions is denoted by arrowheads

3. The rank of the flux matrix (8) was used as the goodness criteria: the higher the rank the better constrained the fluxes are; full rank indicating that a point estimate has been given to each flux independently.

Figure 3 depicts the results of this experiment. The mean rank of the flux matrix is given by the solid line, the standard deviations with dashed lines. The rank of the flux matrix is 16 even when not measuring any metabolites; the topology of the network together with the steady-state assumption already constrains the fluxes in a significant manner. The average rank increases gradually as more and more metabolites are measured, reaching the rank of 24 at its best. Thus, there are fluxes that cannot be determined independently from the other even when measuring complete isotopomer distributions of all metabolites. This is no surprise, since whenever there are alternative routes that manipulate carbon chain the same way, isotopomer distributions remain the same regardless

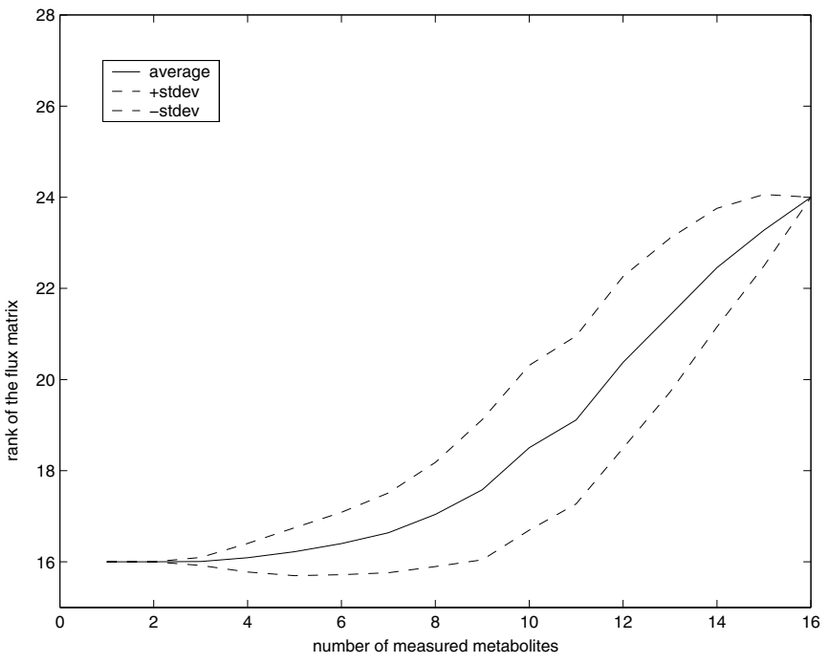


Fig. 3. The average rank (solid line) and standard deviation (dashed line) of the equation system constraining the fluxes as a function of the number of measured metabolites

of the fluxes of these routes. Also, the choice of labeling that is used for the substrates has an effect.

The standard deviation of the rank is the highest when a little over a half of the metabolites have measurements; this seems to indicate that—regarding flux estimation—there are'good' and 'bad' metabolite subsets to measure. Pinpointing such subsets still requires further analysis.

In our preliminary studies we also found out that positional enrichment data—stating only the amount of label in each position independently—seems to give as good rank to the flux matrix as complete isotopomer data. Whether this is an artifact of the network analyzed or a more general truth, we do not know at this point.

As a sanity check, we tested whether the correct fluxes were identified in the tests. We studied the estimation error — the distance of the correct flux vector from the solution space of the system (8). Ideally this distance should be zero, so that the correct flux vector is contained in the solution space. In practise, however, this rarely happens due to the round-off errors in numerical computations

and in the input data. Nevertheless, we found out that the estimation error was small in the test cases examined.

5 Implementation Status and Further Work

At the time of writing, the described estimation algorithms have been implemented, and integrated with a stand-alone version of the KEGG/LIGAND database [4] containing 5260 enzymatic reactions from 72 organisms. As LIGAND—or any other public enzyme database that we are aware of—does not contain carbon mappings, the mappings required for the experiments described in the previous section were encoded by hand. We are currently in process of acquiring the atom mappings for a few microbial organisms, including S.Cerevisiae and E.Coli, computed automatically using the graph-matching methods by Arita [1].

The framework is by no means perfect in its presented form. Several areas of improvement can be distinguished:

- Compartments. The current system works in the 'bag-of-enzymes' model, which ignores any membranes separating metabolic pools. Taking the compartments into account is certainly possible. However, we expect flux estimation to be more involved in that case.
- Bidirectional reactions. In the current version, bidirectional reactions conceptually induce metabolic junctions in the network, hence requiring the measurement of all reactants and products of the reaction in order to compute the backward fluxes. This can be done more cleverly so that the measurement requirements are not as high.
- 'White areas' of the network, pathways where no isotopomer information is available for propagation towards a junction, present a problem for analysis. This is mostly due to the linear modeling framework which dictates that balance equations need to be constructed junction-per-junction basis. Using higher-order balance equations could enable for us the analysis of larger subnetworks than just one-junction systems.

6 Conclusions

The presented framework for flux estimation fulfills several important criteria for a tool that is intended for biochemical engineers and biologists. First, the system is capable of tackling any network topology that is given to it for the basis of analysis. Second, the system uses the isotope tracing data that is available, with no prior assumption of completeness. Both NMR and mass spectrometric data can be used as the basis of analysis. Third, in situations where there is no unique point-solution to the problem, or the system can not find one, the system outputs the feasible (in its view) solution set as a whole, instead of one solution from that set. Moreover, the system outputs the error, 'lack-of-fit' between the

experimental data and the fluxes, which can be used as a basis for, for example, revising the topological assumptions.

References

[1] Arita, M.: Metabolic reconstruction using shortest paths. *Simulation Practise and Theory* 8 (2000), 109–125. 88, 101

[2] Christensen, B., Nielsen, J.: Isotopomer Analysis Using GC-MS. *Metabolic Engineering* 1 (1999), (E8)–(E16). 88

[3] Gaasterland, T., Selkov, E.: Reconstruction of metabolic networks using incomplete information. Proc. 3rd International Conference on Intelligent Systems for Molecular Biology, ISMB-95, 1995, 127–135. 88

[4] Goto, S., Nishioka, T., Kanehisa, M.: LIGAND: chemical database for enzyme reactions. *Bioinformatics* 14 (1998), 591–599. 101

[5] Jeffrey, M., Roach, S., Storey, C., Sherry, D., Malloy, C.: ^{13}C Isotopomer Analysis of Glutamate by Tandem Mass Spectrometry. *Analytical Biochemistry* 300 (2002), 192–205. 88

[6] Kelleher, J.: Flux estimation Using Isotopic Tracers: Common Ground for Metabolic Physiology and Metabolic Engineering. *Metabolic Engineering* 3 (2001), 100–110. 88

[7] Klapa, M., Park, S., Sinskey, A., Stephanopoulos, G.: Metabolite and Isotopomer Balancing in the Analysis of Metabolic Cycles: I. Theory. *Biotechnology and Bioengineering* 62 (1999), 375–391. 88, 90

[8] Maaheimo, H., Fiaux, J., Cakar, P., Bailey, J., Sauer, U., Szyperski, T.: Central carbon metabolism of Saccharomyces cerevisiae explored by biosynthetic fractional 13C labeling of common amino acids. *European Journal of Biochemistry* 268 (2001), 2464–2479. 88

[9] Marx, A., de Graaf, A., Wiechert, W., Eggeling, L., Sahm, H.: Determination of the fluxes in the central metabolism of Corynebacterium glutamicum by nuclear magnetic resonance spectroscopy combined with metabolite balancing. Biotechnology and Bioengineering 49 (1996), 111–129. 88

[10] Mavrovouniotis, M., Stephanopoulos, G., Stephanopoulos, G.: Synthesis of Biochemical Production Routes. *Computers and Chemical Engineering* 16, 6 (1992), 605–619. 88

[11] Rantanen, A., Rousu, J., Kokkonen, J., Tarkiainen, V., Ketola, R.: Computing Positional Isotopomer Distributions from Tandem Mass Spectrometric Data. *Metabolic Engineering*, to appear. 88

[12] Rousu, J., Rantanen, A., Ukkonen, E.: Flux estimation using incomplete isotopomer information. Report C-2002-55, Department of Computer Science, University of Helsinki, 2002. 93

[13] Sanford, K., Soucaille, P., Whited, G., Chotani, G.: Genomics to fluxomics to physiomics—pathway engineering. *Current Opinion in Microbiology* 5 (2002), 318–322. 88

[14] Schmidt, K., Carlsen, M., Nielsen, J., Viladsen, J.: Modeling Isotopomer Distributions in Biochemical Networks Using Isotopomer Mapping Matrices. *Biotechnology and Bioengineering* 55 (1997), 831–840. 88, 91

[15] Schott, J. R. *Matrix Analysis for Statistics*. Wiley, 1997. 95, 98

[16] Širava, M., Schäfer, T., Eiglsperger, M., Kaufmann, M., Kohlbacher, O., Bornberg-Bauer, E., Lenhof, H.-P.: Biominer— Modeling, analysing, and visualizing buichemical pathways and networks. Proc. Europan Conference on Computational Biology 2002. *Bioinformatics* 18 suppl. 2 (2002), S219–S230. 88

[17] Soga, T., Ueno, Y., Naraoka, H., Ohashi, Y., Tomita, M., Nishioka, T.: Simultaneous Determination of Anionic Intermediates for Bacillus subtilis Metabolic Pathways by Capillary Electrophoresis Electrospray Ionization Mass Spectrometry. *Analytical Chemistry* 74, 10 (2002), 2233–2239. 88, 91

[18] Stephanopoulos, G., Aristidou, A., Nielsen, J.: *Metabolic engineering: Principles and Methodologies*. Academic Press, 1998. 88, 90, 91

[19] Szyperski, T., Glaser, R., Hochuli, M., Fiaux, J., Sauer, U., Bailey, J., Wütrich, K.: Bioreaction Network Topology and Metabolic Flux Ratio Analysis by Biosynthetic Fractional ^{13}C Labeling and Two-Dimensional NMR Spectrometry. *Metabolic Engineering* 1 (1999), 189–197. 88

[20] Wiechert, W., Möllney, M., Isermann, N., Wurzel, M., de Graaf, A.: Bidirectional Reaction Steps in Metabolic Networks: III. Explicit Solution and Analysis of Isotopomer Systems. *Biotechnology and Bioengineering* 66 (1999), 69–85. 89, 90, 91, 98

[21] Wittmann, C., Heinzle, E.: Mass Spectrometry for Metabolic Flux Analysis. *Biotechnology and Bioenginering* 62 (1999), 739–750. 88

Dynamic Bayesian Network and Nonparametric Regression for Nonlinear Modeling of Gene Networks from Time Series Gene Expression Data

SunYong Kim, Seiya Imoto, and Satoru Miyano

Human Genome Center, Institute of Medical Science, University of Tokyo
4-6-1 Shirokanedai, Minato-ku, Tokyo, 108-8639, Japan
{sunk,imoto,miyano}@ims.u-tokyo.ac.jp

Abstract. We propose a dynamic Bayesian network and nonparametric regression model for constructing a gene network from time series microarray gene expression data. The proposed method can overcome a shortcoming of the Bayesian network model in the sense of the construction of cyclic regulations. The proposed method can analyze the microarray data as continuous data and can capture even nonlinear relations among genes. It can be expected that this model will give a deeper insight into the complicated biological systems. We also derive a new criterion for evaluating an estimated network from Bayes approach. We demonstrate the effectiveness of our method by analyzing *Saccharomyces cerevisiae* gene expression data.

1 Introduction

The development of microarray technology provides us a huge amount of gene expression data and a new perspective of the analysis of whole genome mechanism. The estimation of a gene network from cDNA microarray gene expression data becomes one of the important topics in the field of bioinformatics and can be viewed as the first step of systems biology.

Using the Bayesian network model (Friedman *et al.* [13]; Imoto *et al.* [14,15]; Pe'er *et al.* [19]) for estimating a gene network from cDNA microarray gene expression data has received considerable attention and many successful investigations have been reported. However, a shortcoming of the Bayesian network model is that this model cannot construct cyclic networks, while a real gene regulation mechanism has cyclic regulations. Recently, the dynamic Bayesian network model (Bilmes *et al.* [3]; Friedman *et al.* [12]; Murphy and Mian [18]; Someren *et al.* [21]) has been proposed for constructing a gene network with cyclic regulations. The dynamic Bayesian network is based on time series data, and usually the data has to be discretized into the several classes. Therefore, the resulting network of the dynamic Bayesian network model depends strongly on the thresholds that are chosen for the discretization. Unfortunately, the discretization leads to information loss. Rangel *et al.* [20] used the state space

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 104–113, 2003.

model for constructing a gene network. Their method is based on linear models. However, there is no guarantee that the relationship between genes is linear. Imoto *et al.* [14,15] proposed a network estimation method based on a Bayesian network and nonparametric regression to avoid discretization and for capturing nonlinear relations among genes. However, the Bayesian network and nonparametric regression model [14,15] still has a remaining problem to be solved in the construction of cyclic regulations.

In this paper, we extend the Bayesian network and nonparametric regression model to the dynamic Bayesian network model, which can construct cyclic regulations when we have a time series gene expression data. We can include the time delay information into the proposed model naturally. The model can extract even nonlinear relations among genes automatically. For constructing a gene network with cyclic regulations based on time series gene expression data, an ordinal differential equation model (Chen *et al.* [5]; De Hoon *et al.* [8]) is an alternative method. However, this model is based on a linear system and probably unsuitable for capturing complex phenomena. We derive a new criterion for choosing an optimal network from the Bayesian statistical point of view [2]. The proposed criterion can optimize the network structure such that it gives the best representation of the gene interactions described by the data with noise. The efficiency of the proposed method is shown by analyzing *Saccharomyces cerevisiae* gene expression data.

2 Dynamic Bayesian Network and Nonparametric Regression

Suppose that we have an $n \times p$ microarray gene expression data matrix $\boldsymbol{X}$, where n and p are the numbers of microarrays and genes, respectively. Usually, the number of genes p is much larger than the number of microarrays, n. In the estimation of a gene network based on the Bayesian network, a gene is considered to be a random variable. When we model a gene network by using statistical models described by the density or probability function, the statistical model should include p random variables. However, we have only n samples and n is usually much smaller than p. In such case, the inference of the model is quite difficult or impossible, because the model has many parameters and the number of samples is not enough for estimating the parameters. The Bayesian network model has been advocated in such situations.

In the context of the dynamic Bayesian network, we consider time series data, with the ith row vector $\boldsymbol{x}_i$ of $\boldsymbol{X}$ corresponds to the states of p genes at time i. As for the time dependency, we consider the first order Markov relation described in Figure 1. Under this condition, the joint probability can be decomposed as

$$P(X_{11}, ..., X_{np}) = P(\boldsymbol{X}_1)P(\boldsymbol{X}_2|\boldsymbol{X}_1) \times \cdots \times P(\boldsymbol{X}_n|\boldsymbol{X}_{n-1}), \tag{1}$$

where $\boldsymbol{X}_i = (X_{i1}, ..., X_{ip})^T$ is a random variable vector of p genes at time i. The conditional probability $P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1})$ can also be decomposed into the product

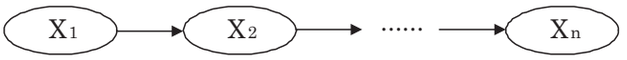


Fig. 1. Time dynamics. $\boldsymbol{X}_i$ is the states of the genes at time i for $i = 1, ..., n$

of conditional probabilities of the form

$$P(\ \boldsymbol{X}_i|\ \boldsymbol{X}_{i-1}) = P(X_{i1}|\ \boldsymbol{P}_{i-1,1}) \times \cdots \times P(X_{ip}|\ \boldsymbol{P}_{i-1,p}), \tag{2}$$

where $\boldsymbol{P}_{i-1,j}$ is the state vector of the parent genes of jth gene at time $i-1$. The equations (1) and (2) hold when we use the density function instead of the probability measure. Hence, the dynamic Bayesian network can then be represented by using densities as follows:

$$\begin{aligned} f(x_{11}, ..., x_{np}) &= f_1(\ \boldsymbol{x}_1) f_2(\ \boldsymbol{x}_2|\ \boldsymbol{x}_1) \times \cdots \times f_n(\ \boldsymbol{x}_n|\ \boldsymbol{x}_{n-1}) \\ &= f_1(\ \boldsymbol{x}_1) \prod_{i=2}^{n} g_1(x_{i1}|\ \boldsymbol{p}_{i-1,1}) \times \cdots \times g_p(x_{ip}|\ \boldsymbol{p}_{i-1,p}) \\ &= f_1(\ \boldsymbol{x}_1) \prod_{j=1}^{p} \left\{ \prod_{i=2}^{n} g_j(x_{ij}|\ \boldsymbol{p}_{i-1,j}) \right\}, \end{aligned}$$

where $\boldsymbol{p}_{i-1,j} = (p^{(j)}_{i-1,1}, ..., p^{(j)}_{i-1,q_j})^T$ is a q_j-dimensional observation vector of parent genes. The decomposition is given by equation (2)

$$f_i(\ \boldsymbol{x}_i|\ \boldsymbol{x}_{i-1}) = g_1(x_{i1}|\ \boldsymbol{p}_{i-1,1}) \times \cdots \times g_p(x_{ip}|\ \boldsymbol{p}_{i-1,p}),$$

For modeling the relationship between x_{ij} and $\boldsymbol{p}_{i-1,j}$, we use the nonparametric additive regression model as follows:

$$x_{ij} = m_{j1}(p^{(j)}_{i-1,1}) + \cdots + m_{jq_j}(p^{(j)}_{i-1,q_j}) + \varepsilon_{ij},$$

where ε_{ij} depends independently and normally on mean 0 and variance σ_j^2. Here, $m_{jk}(\cdot)$ is a smooth function from R to R and can be expressed by using a linear combination of basis functions

$$m_{jk}(p^{(j)}_{i-1,k}) = \sum_{m=1}^{M_{jk}} \gamma^{(j)}_{mk} b^{(j)}_{mk}(p^{(j)}_{i-1,k}), \quad k = 1, \ldots, q_j,$$

where $\gamma^{(j)}_{1k}, ..., \gamma^{(j)}_{M_{jk}k}$ are unknown coefficient parameters and $\{b^{(j)}_{1k}(\cdot), ..., b^{(j)}_{M_{jk}k}(\cdot)\}$ is the prescribed set of basis functions. Then we define a dynamic Bayesian network and nonparametric regression model of the form

$$\begin{aligned} &f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G) \\ &\quad = f_1(\ \boldsymbol{x}_1) \prod_{j=1}^{p} \left[\prod_{i=2}^{n} \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{ -\frac{(x_{ij} - \mu(\ \boldsymbol{p}_{i-1,j}))^2}{2\sigma_j^2} \right\} \right], \end{aligned}$$

where $\boldsymbol{\theta}_G$ is the parameter vector included in the model and $\mu(\ \boldsymbol{p}_{i-1,j}) = m_{j1}(p^{(j)}_{i-1,1}) + \cdots + m_{jq_j}(p^{(j)}_{i-1,q_j})$. When jth gene has no parent genes, $\mu(\ \boldsymbol{p}_{i-1,j})$ reduces to the constant μ_j.

We assume $f_1(\ \boldsymbol{x}_1) = g_1(x_{11}) \times \cdots \times g_1(x_{1p})$ and the joint density $f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G)$ can then be rewritten as

$$f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G) = \prod_{j=1}^{p} \left[g_1(x_{1j}) \prod_{i=2}^{n} \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{ -\frac{(x_{ij} - \mu(\ \boldsymbol{p}_{i-1,j}))^2}{2\sigma_j^2} \right\} \right]$$
$$= \prod_{j=1}^{p} \prod_{i=1}^{n} g_j(x_{ij} |\ \boldsymbol{p}_{i-1,j}; \boldsymbol{\theta}_j), \tag{3}$$

where $\boldsymbol{p}_{0j} = \emptyset$. Thus, $g_j(x_{ij} |\ \boldsymbol{p}_{i-1,j}; \boldsymbol{\theta}_j)$ represents the local structure of jth gene and its parent genes.

3 Derivation of a Criterion for Selecting Network

The dynamic Bayesian network and nonparametric regression model introduced in the previous section can be constructed when we fix the network structure. However, the gene network is generally unknown and we should estimate an optimal network based on the data. This problem can be viewed as a statistical model selection problem (see e.g., Akaike [1]; Burnham and Anderson [4]; Konishi [16]; Konishi and Kitagawa [17]). We solve this problem from the Bayesian statistical approach and derive a criterion for evaluating the goodness of the dynamic Bayesian network and nonparametric regression model.

Let $\pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda})$ be a prior distribution on the parameter $\boldsymbol{\theta}_G$ in the dynamic Bayesian network and nonparametric regression model and let $\log \pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda}) = O(n)$. The marginal likelihood can be represented as

$$\int f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G) \pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda}) d\boldsymbol{\theta}_G.$$

Thus, when the data is given, the posterior probability of the network G is

$$\pi_{post}(G|\ \boldsymbol{X}) = \frac{\pi_{prior}(G) \int f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G) \pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda}) d\boldsymbol{\theta}_G}{\sum_G \left\{ \pi_{prior}(G) \int f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G) \pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda}) d\boldsymbol{\theta}_G \right\}}, \tag{4}$$

where $\pi_{prior}(G)$ is the prior probability of the network G. The denominator of (4) does not relate to model evaluation. Therefore, the evaluation of the network depends on the magnitude of numerator. Hence, we can choose an optimal network as the maximizer of

$$\pi_{prior}(G) \int f(x_{11}, ..., x_{np}; \boldsymbol{\theta}_G) \pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda}) d\boldsymbol{\theta}_G.$$

It is clear that the essential point for constructing a network selection criterion is how to compute the high dimensional integral. Imoto *et al.* [14,15] used the Laplace approximation for integrals (see also Tinerey and Kadane [23]; Davison [6]). This technique can be applied to the dynamic Bayesian network and nonparametric regression model directly. Hence, we have a criterion, named $\mathrm{BNRC}_{dynamic}$, of the form

$$\begin{aligned}
&\mathrm{BNRC}_{dynamic}(G) \\
&\quad = -2\log\left\{\pi_{prior}(G)\int f(x_{11},...,x_{np};\boldsymbol{\theta}_G)\pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda})d\boldsymbol{\theta}_G\right\} \\
&\quad \approx -2\log\pi_{prior}(G) - r\log(2\pi/n) + \log|J_\lambda(\hat{\boldsymbol{\theta}}_G)| - 2nl_\lambda(\hat{\boldsymbol{\theta}}_G|\ \boldsymbol{X}), \qquad (5)
\end{aligned}$$

where r is the dimension of $\boldsymbol{\theta}_G$,

$$\begin{aligned}
l_\lambda(\boldsymbol{\theta}_G|\ \boldsymbol{X}) &= \log f(x_{11},...,x_{np};\boldsymbol{\theta}_G)/n + \log\pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda})/n, \\
J_\lambda(\boldsymbol{\theta}_G) &= -\partial^2\{l_\lambda(\boldsymbol{\theta}_G|\ \boldsymbol{X})\}/\partial\boldsymbol{\theta}_G\partial\boldsymbol{\theta}_G^T
\end{aligned}$$

and $\hat{\boldsymbol{\theta}}_G$ is the mode of $l_\lambda(\boldsymbol{\theta}_G|\ \boldsymbol{X})$. The optimal graph is chosen such that the criterion $\mathrm{BNRC}_{dynamic}$ (5) is minimal.

4 Estimation of a Gene Network

In this section, we show the concrete strategy for estimating a gene network from cDNA microarray time series gene expression data.

4.1 Nonparametric Regression

We use the basis function approach for constructing the smooth function $m_{jk}(\cdot)$ described in Section 2. In this paper we use B-splines [7] as the basis functions. De Boor's algorithm (see, de Boor [7], Chapter 10, p.130 (3)) is a useful method for computing B-splines of any degree. We use 20 B-splines of degree 3 with equidistant knots (see also, Dierckx [10]; Eiler and Marx [11] for the details of B-spline).

4.2 Prior Distribution on the Parameter

For the prior distribution on the parameter $\boldsymbol{\theta}_G$, suppose that the parameter vectors $\boldsymbol{\theta}_j$ are independent of one another. The prior distribution can then be decomposed as $\pi(\boldsymbol{\theta}_G|\boldsymbol{\lambda}) = \prod_{j=1}^{p}\pi_j(\boldsymbol{\theta}_j|\boldsymbol{\lambda}_j)$. Suppose that the prior distribution $\pi_j(\boldsymbol{\theta}_j|\boldsymbol{\lambda}_j)$ is factorized as $\pi_j(\boldsymbol{\theta}_j|\boldsymbol{\lambda}_j) = \prod_{k=1}^{q_j}\pi_{jk}(\boldsymbol{\gamma}_{jk}|\lambda_{jk})$, where λ_{jk} are hyper parameters. We use a singular M_{jk} variate normal distribution as the prior distribution on $\boldsymbol{\gamma}_{jk}$,

$$\pi_{jk}(\boldsymbol{\gamma}_{jk}|\lambda_{jk}) = \left(\frac{2\pi}{n\lambda_{jk}}\right)^{-(M_{jk}-2)/2}|K_{jk}|_+^{1/2}\exp\left(-\frac{n\lambda_{jk}}{2}\boldsymbol{\gamma}_{jk}^T K_{jk}\boldsymbol{\gamma}_{jk}\right),$$

where K_{jk} is an $M_{jk} \times M_{jk}$ symmetric positive semidefinite matrix satisfying $\boldsymbol{\gamma}_{jk}^T K_{jk} \boldsymbol{\gamma}_{jk} = \sum_{\alpha=3}^{M_{jk}} (\gamma_{\alpha k}^{(j)} - 2\gamma_{\alpha-1,k}^{(j)} + \gamma_{\alpha-2,k}^{(j)})^2$. This setting of the prior distribution on $\boldsymbol{\theta}_G$ is the same as Imoto *et al.* [14,15].

4.3 Proposed Criterion

By using the prior distributions in Section 4.2, the $\mathrm{BNRC}_{dynamic}$ can be decomposed as follows:

$$\mathrm{BNRC}_{dynamic} = \sum_{j=1}^{p} \mathrm{BNRC}_{dynamic}^{(j)}, \tag{6}$$

where $\mathrm{BNRC}_{dynamic}^{(j)}$ is a local criterion score of jth gene and is defined by

$$\begin{aligned}
&\mathrm{BNRC}_{dynamic}^{(j)} \\
&\quad = -2 \log \left\{ \int \pi_{prior}(L_j) \prod_{i=1}^{n} g_j(x_{ij} |\ \boldsymbol{p}_{i-1,j}; \boldsymbol{\theta}_j) \pi_j(\boldsymbol{\theta}_j | \boldsymbol{\lambda}_j) d\boldsymbol{\theta}_j \right\} \\
&\quad \approx -2 \log \pi_{prior}(L_j) - r_j \log(2\pi/n) + \log |J_{\lambda_j}^{(j)}(\hat{\boldsymbol{\theta}}_j)| - 2n l_{\lambda_j}^{(j)}(\hat{\boldsymbol{\theta}}_j |\ \boldsymbol{X}),
\end{aligned}$$

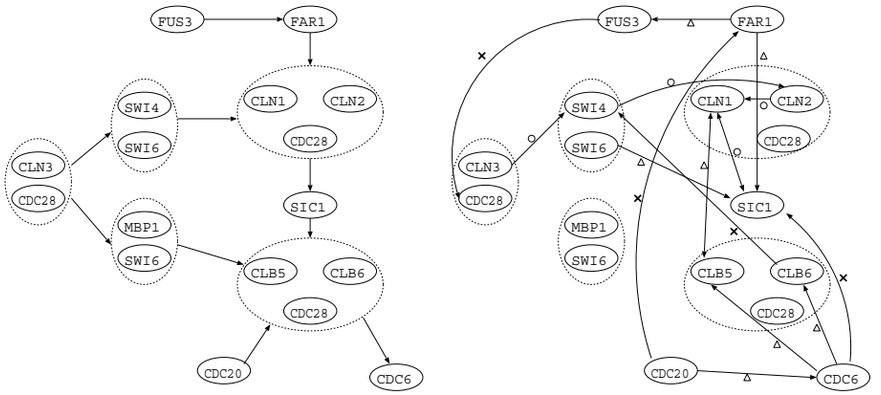
where r_j is the dimension of $\boldsymbol{\theta}_j$,

$$\begin{aligned}
l_{\lambda_j}^{(j)}(\hat{\boldsymbol{\theta}}_j |\ \boldsymbol{X}) &= \sum_{i=1}^{n} \log g_j(x_{ij} |\ \boldsymbol{p}_{i-1,j}; \boldsymbol{\theta}_j)/n + \log \pi(\boldsymbol{\theta}_j | \boldsymbol{\lambda}_j)/n, \\
J_{\lambda_j}^{(j)}(\hat{\boldsymbol{\theta}}_j) &= -\partial^2 \{ l_{\lambda_j}^{(j)}(\hat{\boldsymbol{\theta}}_j |\ \boldsymbol{X}) \} / \partial \boldsymbol{\theta}_j \partial \boldsymbol{\theta}_j^T
\end{aligned}$$

and $\hat{\boldsymbol{\theta}}_j$ is the mode of $l_{\lambda_j}^{(j)}(\boldsymbol{\theta}_j |\ \boldsymbol{X})$. Here $\pi_{prior}(L_j)$ are prior probabilities satisfying $\sum_{j=1}^{p} \log \pi_{prior}(L_j) = \log \pi_{prior}(G)$. We set the prior probability of local structure $\pi_{prior}(L_j)$ as $\pi_{prior}(L_j) = \exp\{-(\text{The number of parent genes ofthe } j \text{ th gene})\}$.

4.4 Algorithm for Learning Network

By using the dynamic Bayesian network and nonparametric regression model together with the proposed criterion, $\mathrm{BNRC}_{dynamic}$, we can formulate the network learning process as follows: it is clear from (3) and (6) that the optimization of network structure is equivalent to choosing the parent genes that regulate the target genes. However, it is a time-consuming task to consider all possible gene combinations as the parent genes. Therefore, we reduce the learning space by selecting candidate parent genes. After this step, a greedy hill-climbing algorithm is employed for finding better networks. Our algorithm can be expressed as follows:



(a)Target pathway (b)Result of the proposed method

Fig. 2. Cell cycle pathway compiled in KEGG

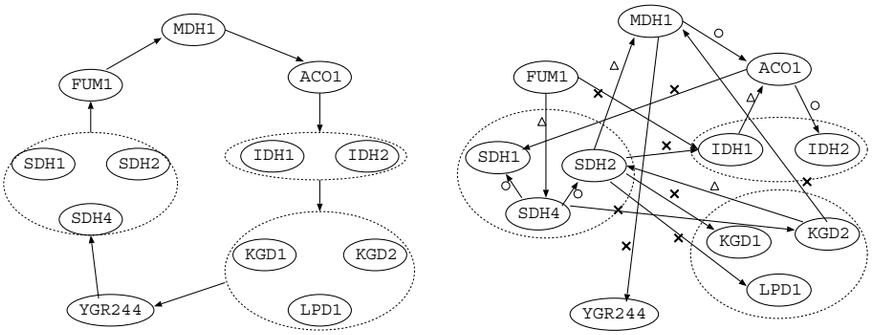
Step 1: *Preprocessing stage*
We make the $p \times p$ matrix whose (i, j)th element is the $\text{BNRC}^{(j)}_{dynamic}$ score of the graph "$\text{gene}_i \rightarrow \text{gene}_j$" and we define the candidate set of parent genes of gene_j that gives small $\text{BNRC}^{(j)}_{dynamic}$ scores.

Step 2: *Learning stage*
For a greedy hill-climbing algorithm, we start form the empty network and repeat the following steps:

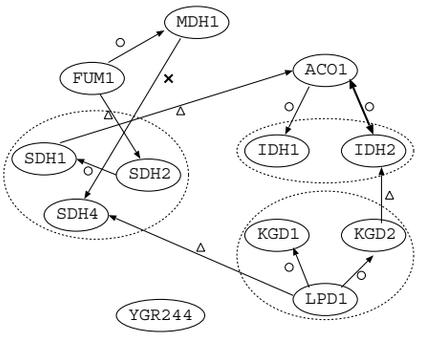
Step 2-1: For gene_j, implement one from two procedures that *add* a parent gene or *delete* a parent gene, which gives smaller $\text{BNRC}^{(j)}_{dynamic}$ score.
Step 2-2: Repeat Step2-1 until we find the best set of parent genes of jth gene.
Step 2-3: Repeat Step2-1 and 2-2 for all genes.
Step 2-4: We choose the optimal network that gives the smallest $\text{BNRC}_{dyanmic}$ score.

5 Computational Experiment

We demonstrate our proposed method through the analysis of the *Saccharomyces cerevisiae* cell cycle gene expression data collected by Spellman *et al.* [22]. This data contains two short time series (two time points; cln3, clb2) and four medium time series (18, 24, 17 and 14 time points; alpha, cdc15, cdc28 and elu). In the estimation of a gene network, we use four medium time series. For combining four time series, we ignore the first observation of the target gene and last one of



(a)Target pathway (b)Result of the Bayesian network



(c)Result of the proposed method

Fig. 3. Metabolic pathway reported by DeRisi *et al.* [9]

the parent genes for each time series when we fit the nonparametric regression model. We set the number of the candidate parent genes to 10, since the resulting network did not change while using a larger set of candidate parents.

At first, we focus on the cell cycle pathway stored in the KEGG database [24]. The target network is around CDC28 (YBR160w; cyclin-dependent protein kinase). This network contains 45 genes. The partial pathway registered in KEGG is shown in Figure 2 (a) and the estimated network is in Figure 2 (b). The edges in the dotted circles can be considered as the correct edges. We can model some correct relations by using the proposed method. We denote the correct estimation by a circle next to the edge. A triangle represents the edge incorrectly directed or the edge which bypassed one gene, while a cross represents an estimated edge that is not present in the target graph.

Our second example is the metabolic pathway reported by DeRisi *et al.* [9]. This network contains 57 genes and the target pathway is partially shown in

Figure 3 (a). We apply the Bayesian network and nonparametric regression model [14,15] to these data. The resulting network is shown in Figure 3 (b). The network of Figure 3 (c) is obtained by the dynamic Bayesian network and nonparametric regression model. It is difficult to estimate the metabolic pathway from cDNA microarray data. However, our model can detect some correct relations. Comparing with the Bayesian network and nonparametric regression, the number of false positives of the proposed method in Figure 3 (c) is much smaller than those in Figure 3 (b).

We observed that the Bayesian network and nonparametric regression can work well in many cases. However, when there is a cyclic gene regulation, the Bayesian network and nonparametric regression model tends to estimate many false positives in the cyclic regulation. In such case, the proposed method can reduce the number of false positives and estimate gene regulations effectively.

6 Discussion

In this paper, we proposed a new statistical gene network estimation method based on the dynamic Bayesian network and nonparametric regression model. Our proposed method has several advantages compared with other network estimation method such as the Bayesian and Boolean networks. First, our model can take time information into account naturally. Second, our model can analyze the microarray data as the continuous data without additional data pretreatments such as discretization. Last, even nonlinear relations can be detected and modeled by our proposed method.

Simulating a genetic system is one of the central topics in systems biology. Since the simulation is based on biological knowledge, our network estimation method can support the biological simulation by constructing the unknown regulations. In this paper, we only demonstrate the model based on the first-order Markov relation between time points described in Figure 1. However, the relationship between time points is arbitrary and we can choose the time dependency structure based on our proposed criterion. We would like to discuss this topic in our future work.

References

1. Akaike, H.: Information theory and an extension of the maximum likelihood principle. In: Petrov, B. N., Csaki, F. (eds.): 2nd International Symposium on Information Theory. Akademiai Kiado, Budapest (1973) 267–281 107
2. Berger, J. O.: Statistical Decision theory and Bayesian analysis. Springer-Verlag New York (1985) 105
3. Bilmes, J. A.: Dynamic Bayesian multinets. Proc. 16th Conf. on Uncertainty in Artificial Intelligence (2000) 38–45 104
4. Burnham, K. P., Anderson, D. R.: Model selection and inference, a practical information-theoretical approach. Springer-Verlag New York (1998) 107
5. Chen, T., He, H. L., Church, G. M.: Modeling gene expression with differential equations. Proc. Pacific Symposium on Biocomputing **4** (1999) 29–40 105

6. Davison A. C.: Approximate predictive likelihood. Biometrika **73** (1986) 323–332 108
7. De Boor, C.: A practical guide to splines. Springer-Verlag Berlin (1978) 108
8. De Hoon, M. J. L., Imoto, S., Kobayashi, K., Ogasawara, N., Miyano, S.: Inferring gene regulatory networks from time-ordered gene expression data of *Bacillussubtilis* using differential equations. Proc. Pacific Symposium on Biocomputing **8** (2003) in press 105
9. DeRisi, J., Lyer, V. R., Brown, P. O.: Exploring the metabolic and gene control of gene expression on a genomic scale. Science **278** (1997) 680–686 111
10. Dierckx, P.: Curve and surface fitting with splines. Oxford (1993) 108
11. Eiler, P. H. C., Marx, B.: Flexible smoothing with B-splines and penalties (with discussion). Statistical Science **11** (1996) 89–121 108
12. Friedman, N., Murphy, K., Russell, S.: Learning the structure of dynamic probabilistic networks. Proc. Conf. on Uncertainty in Artificial Intelligence (1998) 139–147 104
13. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian network to analyze expression data. J. Comp. Biol. **7** (2000) 601–620 104
14. Imoto, S., Goto, T., Miyano, S.: Estimation of genetic networks and functional structures between genes by using Bayesian network and nonparametric regression. Proc. Pacific Symposium on Biocomputing **7** (2002) 175–186 104, 105, 108, 109, 112
15. Imoto, S., Kim, S., Goto, T., Aburatani, S., Tashiro, K., Kuhara, S., Miyano, S.: Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. Proc. IEEE Computer Society Bioinformatics Conference (2002) 219–227 104, 105, 108, 109, 112
16. Konishi, S.: Statistical model evaluation and information criteria. In: Ghosh, S. (ed.): Multivaliate Analysis, Design of Experiments and Survey Sampling. Marcel Dekker, New York (1999) 369–399 107
17. Konishi, S., Kitagawa, G.: Generalized information criteria in model selection. Biometrika **83** (1996) 875–890 107
18. Murphy, K., Mian, S.: Modelling Gene Expression Data using Dynamic Bayesian Networks. *Technical report, Computer Science Division, University of California, Berkeley, CA* (1999) 104
19. Pe'er, D., Regev, A., Elidan, G., Friedman, N.: Inferring subnetworks from perturbed expression profiles. Bioinformatics **17** (ISBM2001) 215–224 104
20. Rangel, C., Wild, D. L., Falciani, F.:Modelling biological responses using gene expression profiling and linear dynamical systems. Proc. 2nd International Conference on Systems Biology (2001) 104
21. Someren, E. V., Wessels, L., Reinders, M.: Linear modeling of genetic networks from experimental data. Bioinformatics **18** (ISMB2002) 355–366 104
22. Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. Molecular Biology of the Cell **9** (1998) 3273–3297 110
23. Tinerey, L., Kadane, J. B.: Accurate approximations for posterior moments and marginal densities. J. Amer. Statist. Assoc. **81** (1986) 82–86 108
24. http://www.genome.ad.jp/kegg/ 111

Discrete Event Simulation for a Better Understanding of Metabolite Channeling – A System Theoretic Approach

Daniela Degenring[1], Mathias Röhl[2], and Adelinde M. Uhrmacher[2]

[1] Institute of Biotechnology II, Center of Research Juelich
52425 Juelich, Germany
d.degenring@fz-juelich.de
[2] Department of Computer Science, University of Rostock
D-18051 Rostock, Germany
lin@informatik.uni-rostock.de

Abstract. Typically differential equations are employed to simulate metabolic processes. To develop a valid continuous model based on differential equations requires accurate parameter estimations; an accuracy which is often difficult to achieve, due to the lack of data. In addition processes in metabolic pathways, e.g. metabolite channeling, seem to be of a rather qualitative and discrete nature. With respect to the available data and to the perception of the underlying system a discrete rather than a continuous approach to modeling and simulation seems more adequate. However, a discrete approach does not necessarily imply a more abstract view on the system. If we move from macro to micro and multi-level modeling, aspects of subsystems and their interactions, which have been only implicitly represented, are now explicit part of the model. Based on the simulation environment JAMES we started exploring phenomena of metabolite channeling on different levels of abstractions. JAMES is a discrete event simulation system and supports a modular hierarchical composition of models, the change of modeling structure during simulation, and a distributed, parallel execution of models.

1 Introduction

The most common approach in natural and engineering sciences is to use differential equations for modeling and simulation. Researchers interested in the dynamics of cellular systems feel comfortable with the well known formalism. Difficulties of determining suitable numerical algorithms can be solved transparent to the user. Simulation tools like Gepasi [15], PathwayPrism^{TM} [1] support a comfortable modeling of cellular systems and a numerical execution of models that describe regulation in great detail. However, the development of detailed and accurate continuous models depends on the availability of suitable measurements that allow an estimation of parameter values. Often measurements are difficult to achieve in cellular systems and so the modeler resorts to tuning the model by adapting the parameter values until the simulation reproduces the

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 114–126, 2003.

expected behavior which endangers the validity of the model. Another problem of modeling with differential equations is that systems proceed not necessarily continuously and deterministically. The latter can be addressed by stochastic differential equations [14], whereas the former calls for qualitative and discrete modeling formalisms.

The variety of formalisms that are employed for modeling and simulating cellular systems, e.g. qualitative differential equations, boolean networks [6], Petri Nets [3] and π-calculus [21], gives evidence not only of the wish to test formalisms in a new application area but also of the importance of diversity: one formalism will hardly suffice to describe and analyze all phenomena of cellular systems in a natural manner. Typical criteria to help us structuring the space of different modeling formalisms, see e.g. in [28], are whether parameters and variables are qualitatively or/and quantitatively scaled, whether the time scale is of a continuous or discrete nature and whether stochastic aspects are supported. Modeling with differential equations presupposes a continuous state space and a continuous time scale. In contrast discrete event approaches model the dynamics of a system based on a mixed quantitatively and qualitatively scaled state space and a continuous time scale. Events are triggered by situations or by the flow of time often integrating stochastic aspects. Purely qualitative approaches, like e.g. QSIM, describe the system in terms of ordinal scaled values and constraints, same as the time space which evolves as a tree of time lines with alternating time points and time intervals. Another interesting aspect in distinguishing between modeling approaches is whether phenomena are described on macro level, e.g. by change rates of concentrations, or by updating a cell's beliefs and intentions [11], or on multi-level, e.g. [13] focusing on subsystems and their interactions [2].

However, before we will discuss the modeling formalism we wish to apply, the system of interest and the objective of the simulation study shall be discussed.

2 Metabolite Channeling

Traditionally the cellular metabolism is described based on a number of enzymes in a well-stirred aqueous phase. The phase also entails substrates with which the enzymes interact and the products of these interactions. The high cellular protein concentrations and intensive protein activities which are caused by macromolecular crowding [22] favor enzyme-enzyme interactions and enzyme associations. These interactions lead to a direct transfer of intermediates from the active side of one enzyme to the active side of another: the phenomenon of channeling or tunneling, which might or might not release intermediates into the bulk solution [9].

2.1 The Importance of Metabolite Channeling

The channeling of substrates may be dynamic or static depending on the various types of enzyme association. Static channeling occurs when enzymes catalyzing a reaction sequence form a stable complex during the entire conversion process.

The lifetime of the enzyme complex is longer than the time required for one reaction sequence. Dynamic channeling denotes dynamic enzyme associations involving metabolic intermediates, that are formed and broken down at each cycle in the conversion of a substrate molecule into the product of the reaction sequence [12].

Two indications for the importance of channeling phenomena in cellular organism are emphasized in the literature. Some of the enzymes which operate on sequentially related substrates have been observed lying closely juxtaposed in the cell. This supports the assumption that this arrangement has evolved due to some biological advantages. In a number of cases the cytoplasma does not show the expected concentration of an intermediate which has been produced by the enzymatic transformation. This observation might be explained by the hypothesis that the intermediates seem to be trapped thermodynamically or are kept by barriers from a free diffusion [33].

The assumed advantages of channeling over free diffusion are the reduction of the transit time required for moving reaction products from the active side of one enzyme to the other, the protection of chemically labile species from decomposition within the aqueous environment, the circumvention of unfavorable equilibria and the segregation of reaction intermediates from competing enzyme transformations [17]. Examples have been mentioned for several biochemical processes including the DNA replication, RNA and protein biosynthesis as well as metabolic pathways like the tricarboxylic acid cycle, the glycolysis and the amino acid metabolism [25, 32, 26, 31]. Experimentally the tunneling of intermediates has been mainly investigated by measurements of the transient time [27, 10] and the istope dilution of reaction intermediates [19, 4]. Theoretical papers are concerned with the effects of channeling on the kinetics of metabolic pathways and metabolite concentrations [5].

2.2 Simulating Metabolite Channeling

Most modeling and simulation approaches use differential equations to study these phenomena. Based on elementary reaction steps, only part of which are measurable differential equations are defined and composed to produce the overall transformation rate of one enzyme. To develop such a fine-grained model of the channeling process requires tuning parameters and assumptions that have not been validated [16]. The differential equations are enriched with stochastic aspects to capture the molecular kinetics and to predict the behavior of multiple enzymes and their interaction [34]. Another problem is that available qualitative information about these processes is not integrated or it is transformed into quantitative information. The effect of this negligence or of the transformation on the overall produced dynamics cannot easily be assessed. Moreover, the question is whether the objective of the simulation study requires a continuous simulation. In our study we are interested in exploring, when and under which physiological conditions the channeling is induced, the kind and amount of intermediates produced during this process and the implications of perturbations by effectors in the bulk solution. What happens inbetween these events is of minor

interest. Thus, a discrete event approach seems most suitable. It allows to evaluate quantitative and qualitative knowledge about the phenomena of interest. Its continuous time scale supports the definition of events at arbitrary time points, and thus the adaptation to individual time scales of involved processes. Varying structure observed in metabolite channeling, e.g. in the dynamic channeling, where the interaction patterns vary over time present a challenge for simulation systems.

3 The Simulation Environment James

James (A Java based Agent Modeling Environment for Simulation) [29] has been developed for simulating multi-agent systems, systems that are interpreted as communities of interacting autonomous entities, with the ability to adapt their behavior, interaction, and composition pattern. James is based on the formalism dynDevs [30] which adds reflection to Devs to capture the notion of self aware and self manipulating entities.

3.1 The Basis – Devs

Devs [36] belongs to the formal and general approaches to discrete event simulation whose development has been firmly rooted in systems theory [18]. The model design supports a hierarchical, compositional construction of models. It distinguishes between atomic and coupled models. Atomic models are equipped with input and output ports (X, Y) by which they communicate with their environment. Their behavior is defined by transition functions, an output function (λ), and a time advance function (ta) which determines how long a state persists "per se". An internal transition function (δ_{int}) dictates state transitions due to internal events, the time of which is determined by ta. At the time an internal event is due, the output function (λ) produces an output which is sent via the output ports. The external transition function (δ_{ext}) is triggered by the arrival of external events via the input ports. A coupled model is a model consisting of different components and specifying the couplings of its components. Same as atomic models, coupled models are equipped with input and output ports. A coupled model is described by a set of component models, which may be atomic or coupled, and by the couplings that exists among the components and between the components and its own input and output ports. Coupled models do not add functionality to atomic models since each coupled model can be expressed as an atomic model, i.e. Devs models are closed under coupling. Thereby a hierarchy of modular models can be build. An abstract simulator equips the Devs formalism with a clear execution semantics. Several extensions of the formalism exist, e.g. to support continuous and hybrid simulations [20]. As Devs models can be interpreted as time triggered automata a relation to StateCharts exists, for a detailed discussion see e.g. [24]. The development of Devs has been less motivated by the specification of software than by systems theory in the 70ties. Devs dedication to modeling complex dynamic systems is also reflected

A DYNDEVS is a structure

$dynDevs =_{df} \quad \langle X, Y, m_{init}, \mathcal{M}(m_{init})\rangle$ with	
X, Y	sets of inputs, outputs
$m_{init} \in \mathcal{M}(m_{init})$	the initial model
$\mathcal{M}(m_{init})$	is the least set having the structure
$\{\langle S, s_{init}, \delta_{ext}, \delta_{int}, \rho_\alpha, \lambda, ta\rangle \mid$	
S	set of states
$s_{init} \in S$	the initial state
$\delta_{ext} : Q \times X \to S$	the external transition function is triggered by the arrival of events which have been sent by other models (its influencers) with $Q = \{(s, e) : s \in S, 0 \leq e \leq ta(s)\}$
$\delta_{int} : S \to S$	the internal transition function is triggered by the flow of time
$\rho_\alpha : S \to \mathcal{M}(m_{init})$	the model transition function determines the next "incarnation" of the model in terms of state space and behavior pattern
$\lambda : S \to Y$	the output function fills the output port and triggers external transitions within influenced components
$ta : S \to \mathcal{R}_0^+ \cup \{\infty\}\}$	the time advance function determines how long a state persists

satisfying the property

$$\forall n \in \mathcal{M}(m_{init}).(\exists m \in \mathcal{M}(m_{init}).\ n = \rho_\alpha(s^m)\ with\ s^m\ \in S^m) \vee n = m_{init}$$

Fig. 1. The formalism DYNDEVS. (For a detailed discussion, including the definition of coupled models and the abstract simulator see [30])

in its widespread use in simulating ecological, sociological, manufacturing, and military systems.

3.2 A Reflective Extension – dynDevs

To support models that are able to assess and access their composition, interaction, and behavior structure from an agent's perspective, the DYNDEVS formalism has been developed (Figure 1).

An atomic model in DYNDEVS is defined as a set of models that inherit state set, transition, output and time advance functions from DEVS atomic models. The reflectivity is introduced by the model transition (ρ_α) which maps the state space of a model into a set of models to which the model belongs. Thereby, sequences of models are produced over time and a model can change its own state and its behavior pattern during simulation.

As the coupled model holds the information about composition and interaction between components, a change of composition or interaction, even though induced by an atomic model, takes effect at the level of the coupled model. Therefore, coupled dynDevs models are introduced. Coupled models are formed by sets of models as well. A so called network transition maps the current state of the coupled model in terms of the states of its components into a possibly new network with new components, new couplings, new domains of these components, and a new network transition function.

It could be shown that dynDevs and Devs are bisimular and dynDevs is closed under coupling. Often only the implementation defines the semantics of the formalism, e.g. with respect to events happening at the same time [8]. In dynDevs the semantics is defined by the abstract simulator which is responsible for executing the model and for dissolving conflicts between structural and non structural changes in an unambiguous way [30]. Each model is interpreted and executed by a tree of processors which reflects the hierarchical compositional structure of the model. Each of the processors is associated with a component of the model and is responsible for invoking the component's methods and controlling the synchronization by exchanging messages with the other processors of the processor hierarchy.

Further extensions have been aimed at supporting the interaction of simulation with real-time processes by introducing peripheral ports into the model definition. The original formalism has also been enriched to integrate continuous behavior within the discrete models. Another extension is the introduction of coupling functions at the level of coupled models. This later extension, i.e. introducing coupling functions, can be used to describe comfortably the competition for resources and thus comes in handy in simulating metabolite channeling.

3.3 The Implementation – James

The current implementation directly translates the dynDevs formalism into Java. Models can be defined and refined based on component libraries. During simulation models are able to create and add models in the coupled model they belong to, they can delete and remove themselves, and they can access their interaction structure. To initiate structural changes outside their boundary, models have to turn to communication and negotiation. Thus, a movement from one coupled model to another implies that another atomic model complies with the request to add the moving model into the new interaction context. Different distributed, parallel, conservative and optimistic execution strategies have been implemented. James has been used in different application areas, e.g. to analyze the behavior of planning agents in dynamic environments [23], to model and simulate mobile agents, and to simulate the effects of urban disasters and the intervention of local authorities in medieval towns with several thousands of partly utility-based, partly belief, desires and intention (BDI) based actors [7].

4 Modeling Channeling Phenomena in James

The first model describes the static channeling of the tryptophan synthase whereas the second model is dedicated to the dynamic channeling of glycolysis processes.

4.1 Tryptophan Synthase

The tryptophan synthase catalyzes the final two reactions of the biosynthesis of L tryptophan. In bacteria it exists as a nearly linear $(\alpha\beta)_2$ complex. Each α subsystem catalyses the cleavage of indole 3-glycerol phosphate (IGP) to produce indole and D-glyceraldehyde-3-phosphate, and each β subsystem produces tryptophan from L-serine and the channeled indole.

To model the tryptophan synthase we utilize the concept of experimental frames. Basically an experimental frame specifies a limited set of circumstances under which a system is to be obersved or subjected to experimentation. The experimental frame of our model tryptophan synthase comprises a generator and an acceptor. Whereas the generator is responsible for generating L-serine, IGP, and G3P molecules with a certain interarrival time, the acceptor collects the products of the processes and calculates performance indicators. The acceptor might also affect the generation of molecules, e.g. defining dynamically the end of the experiment. Both models are connected to the coupled model representing the tryptophan synthase (Figure 2).

The coupled model comprised two pairs of interconnected α, β models, whose input ports are connected to the input ports of the overall model via the coupling function. Coupling functions are used to describe the competition among consumable resources, by selecting randomly one recipient and forwarding "null-events" to the other components (Figure 2).

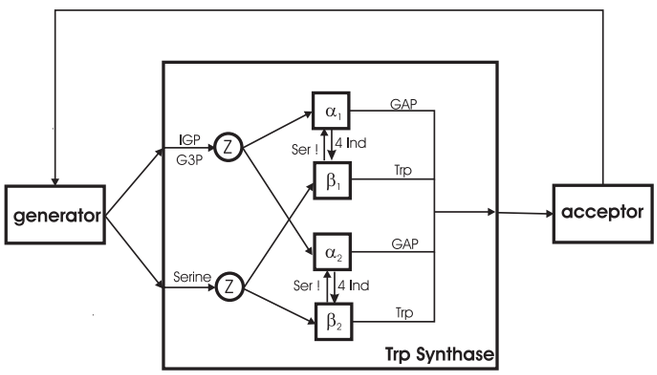


Fig. 2. The experimental frame of the tryptophan synthase in vitro experiment

The α and the β subsystem are connected by a largely hydrophobic tunnel. The α subsystem transforms IGP to indole and D-glyceraldehyde 3-phosphate (GAP). The first is forwarded to the β subsystem whereas the latter is released into the bulk solution. Its functionality is hampered by glycerol 3-phospate (G3P) (see Figure 3).

The behavior of the β subsystem presents itself slightly more complex. It produces tryptophan from L-serine and the channeled indole. The binding of L-serine increases the cleavage of IGP in the α subsystem by a factor of 20-30 [9]. The allosteric communication between the subunits appears to be reciprocal,

```
class SingleAlphaTrp extends AtomicModel {
public State deltaExt (State state, double elapsedTime) {
// the external transition is responsible for the reaction of
// alphaTrp to external events
   Object input = readInportMessage();
   if (phase == idle && input instanceof IGP) { phase = active; }
   else
     if (phase == idle && input instanceof G3P &&
        input.concentration > criticalG3P)
        { phase = inhibited; c_G3P = input.concentration; }
     else
        if (phase == inhibited && input instanceof IGP &&
           input.concentration > c_G3P) {
             phase = active;  c_G3P = 0; }
        else if (phase == active && input instanceof Serine)
             { speedUpFactor = 1/30; }
}

public State deltaInt (State state) {
// the internal transition is triggered by the flow of time
  if (phase == active) { phase = idle; speedUpFactor=1; }
}

void lambda(State state) {
    if (phase == active) {
        outPortPut("outToBeta", new Indole());
        outPortPut("outToBulkSolution", new GAP()); }
}

public double timeAdvance (State state) {
  if (phase == active) {
      return speedUpFactor * exponentialRandom(tIGP2Indole); }
  else return infinity;
} }
```

Fig. 3. The alpha tryptophan synthase as a single atomic model in JAMES

```
public real timeAdvance(State state) {
  if (phase == active) { return exponentialRandom(1/1000); }
  else
    if (phase == releasing) { return exponentialRandom(1/8); }
  else
    return infinity;
}
```

Fig. 4. The transition times for producing and releasing tryptophan modeled in the time advance function of the SingleBetaTrp

since the α unit increases the affinity of the β unit for amino acids. Thus, both α and β subsystems are interacting with each other. The β subunit needs some time to build tryptophan, however this time span is dominated by the time which is needed to release the finished product. Whereas the production rate of tryptophan is about $1/1000\ s^{-1}$, it is released after $1/8\ s^{-1}$ on average. Both time delays can easily be modeled by the time advance function (Figure 4).

The tunnel provides 4 places for the storage of indole. This storage capability is integrated into the β system. A more fine grained model of the functioning of the tunnel with its doors and inhibitors could easily be added between the β and α model, if data allowed the modeling and the objective of the study asked for such a refined dynamic model of the tunnel.

4.2 Glycolysis

In contrast to the static channeling of the tryptophan synthase, the channeling of the glycolysis is of a dynamic nature. Enzyme associations are formed and broken down at each cycle in the conversion of a substrate molecule into the product of the reaction sequence. These changing enzyme associations are translated in JAMES into changing couplings between models during simulation. The density and diversity of interactions between subsystems increases compared to the tryptophan synthase. The subsystems compete for substrates and intermediates.

We adapt the experimental frame of the first example. Now only fructose-1,6-bisphosphate is generated by the experimental frame, whereas the majority of substrates that are flowing in the bulk solution are intermediates of the glycolysis process, e.g. glyceraldehyde-3-phosphate, 3-phosphoglycerate.

Whereas it is possible to simulate several thousands of enzymes as separate models in JAMES the question is whether this kind of micro simulation is really required if we are interested in certain macro phenomena, e.g. the pyruvate that is produced in the cell. As we have the possibility to model a system at macro, micro and multi-level in JAMES, we have to choose a suitable organizational level, i.e. to identify the entities of interest, to simulate the dynamic metabolite channeling of the glycolysis.

Figure 5 shows a subsystem of the glycolysis where groups of enzymes belonging to the same type are described as separate models. As all enzymes that

belong to the same type share the same behavior pattern, each group can easily be modeled as one atomic model, whose state comprises the macro variables of the group, e.g. number of enzymes and average activity of the enzymes over time, and micro information about the state of the different individuals including phase and location.

An intensive exchange of intermediates between the different groups of enzymes takes place. Aldolase (Aldo) forms a complex with glyceraldehyde-3-phosphate dehydrogenase (GAPDH), which catalyses the subsequent reaction in glycolysis. Glycerol-3-phosphate dehydrogenase (G3PDH) can also bind to aldolase. Both dehydrogenases transfer NADH directly to lactate dehydrogenase (LDH), i.e. a coupling between the formation of NADH and its oxidation by lactate dehydrogenase is enabled (Figure 5 (left)).

An external perturbation of the glycolysis subsystem, i.e. the signal of fructose-1,6-bisphosphate or an highly accumulated NADH lead to the movement of LDH from neighbored subsystems into the glycolysis subsystem. There it starts its interaction with G3PDH and GAPDH. This leads to the production of lactate and NAD. An increase of the NADH concentration above a certain threshold moves the LDH enzyme out of the glycolysis subsystem. Its associations with G3PDH and GAPDH are dissolved. As a consequence LDH consumes no longer its share of NADH molecules which now are offered to subsequent subsystems of the glycolysis. This movement into and out of the glycolysis subsystem can easily be modeled in JAMES; it is similar to the movement of agents that move from one interaction context into another interaction context, thereby preserving their internal state.

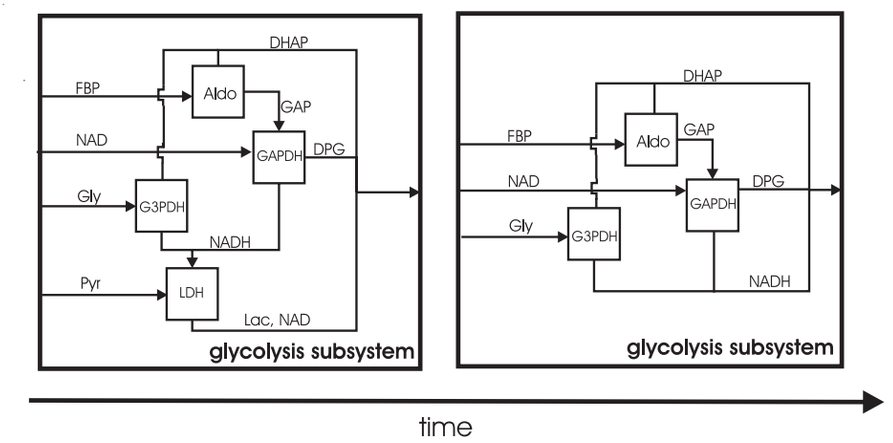


Fig. 5. LDH moves out of the glycolosis subsystem depending on changed physiological conditions

5 Conclusion – The Problem of Choosing a Suitable Level of Abstraction

At the time we started our work we were looking for a different representation of cellular systems than differential equations that would allow us to describe qualitative aspects of metabolite channeling more naturally. Intuitively, we assumed to arrive at a less detailed and thus more abstract model of metabolite channeling. However, a certain formalism does not imply a certain level of abstraction. At first glance a discrete event approach seems more abstract than a continuous approach. As it can be interpreted as an approximation of crucial events that occur during continuous simulation, e.g. approaching a threshold and newly associating with an enzyme. Also, if we turn from quantitative values to qualitative or fuzzy values this leads to a more abstract representation of concentrations and their changes, e.g. [11]. However, continuous modeling is often accompanied by a macro perception of the system. Differential equations are used to define the rate of change of state variables, like concentrations. If we substitute a macro level approach for a multi-level approach individuals, subsystems and their interactions become the focus of interest. A discrete event multi-level approach where each enzyme and its interactions are modeled seems far less abstract than the continuous model from this perspective. So a formalism alone does not tell too much about the level of abstraction nor of the induced complexity of the model. To keep the complexity of the multi-level simulation at bay, we decided not to describe each enzyme as a full fledged model but to group enzymes according to their behavior pattern which provided a suitable level of abstraction.

The work described is work in progress. The first simulation runs have been executed; however, further simulations, a refinement and extension of models and additional data are required to validate the derived model and to compare it with the results of other simulations. So it seems too early to evaluate our approach with respect to promises it holds for the area of cellular systems in general. However, our experiences have shown that the modular hierarchical approach of JAMES makes it easy to extend, refine, and even redefine models on different organizational levels and the support of variable structure models allows to capture certain phenomena of cellular cells in a natural manner, both aspects will help in future work, e.g. to integrate a model component which describes the creation of enzymes by the DNA [35].

References

[1] Pathwayprism^{TM}. http://www.physiome.com/. 114

[2] M. Bunge. *Ontology II: A World of Systems*, volume 4 of *Treatise of Basic Philosophy*. Reidel, Dordrecht, 1979. 115

[3] M. Chen, R. Hofestaedt, and A. Freier. A workable approach for modeling and simulation of biochemical processes with a hybrid petri nets system. In *1st International MTBio Workshop on Function and Regulation of Cellular Systems: Experiments and Models,*, Dresden, 2001. 115

[4] R.I. Christopherson and M.E. Jones. The overall synthesis of l-5,6-dihydroorotate by multienzymatic protein pyr1-3 from hamster-cells - kinetic studies, substrate channeling, and the effects of inhibitors. *Journal of Biological Chemistry*, (255):11381–11395, 1980. 116

[5] A. Cornish-Boden. Kinetic consequences of channelling. In L. Agius and H.S.A. Sheratt, editors, *Channelling in Intermediary Metabolism*, London, 1997. Portland Press Ltd. 116

[6] H. de Jong. Modeling and simulation of genetic regulatory systems: A literature review. Technical Report 4032, INRIA, 2000. 115

[7] U.C. Ewert, M. Röhl, and A.M. Uhrmacher. What good are deliberative interventions in large scale desasters? exploring the consequences of crisis managment in pre-modern towns with agent-oriented simulation. In A. Fürnkranz-Prskawetz and F. Billari, editors, *Agent Based Computational Demography*. Physica Verlag, 2002. 119

[8] D. Harel and A. Naamad. The STATEMATE semantics of statecharts. *ACM Transactions on Software Engineering and Methodology*, 5(4):293–333, 1996. 119

[9] X. Huang, H.M. Holden, and F. Raushel. Channeling of substrates and intermediates in enzyme-catalyzed reactions. *Annual Review in Biochemistry*, (70):149–180, 2001. 115, 121

[10] Ovadi J., P. Tompa, B. Vertessy, F. Orosz, T. Keleti, and G.R. Welch. Transient-time analysis of substrate-channelling in interacting enzyme-systems. *Biochemical Journal*, (57):187–190, 1989. 116

[11] C.M. Jonker, J.L. Snoep, J. Treur, H.V. Westerhoff, and W.C.A. Wijngaards. Putting intentions into cell biochemistry: An artificial intelligence perspective. *Journal of Theoretical Biology*, 214:105–134, 2002. 115, 124

[12] B.N Kholendenko, M. Cascante, and H.V. Westerhoff. Control and regulation of channelled versus ideal pathways. In L. Agius and H.S.A. Sheratt, editors, *Channelling in Intermediary Metabolism*, London, 1997. Portland Press Ltd. 116

[13] J.U. Kreft, G. Booth, and J.W.T. Wimpenny. BacSim a simulator for individual based modelling of bacterial colony growth. *Microbiology*, 144:3275–3287, 1998. 115

[14] H.H. McAdams and A. Arkin. Stochastic mechanisms in gene expression. In *Proceedings of the National Academy of Sciences of the USA*, pages 814–819, 1997. 115

[15] P. Mendes. GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems. *Computer Applications in the Biosciences*, 9(5):563–571, 1993. 114

[16] P. Mendes, D.B. Kell, and H. Westerhoff. Channelling can decrease pool size. *European Journal of Biochemistry*, (204):257–266, 1992. 116

[17] J. Ovadi and F. Orosz. A new concept for control of glycolysis. In L. Agius and H.S.A. Sheratt, editors, *Channelling in Intermediary Metabolism*, London, 1997. Portland Press Ltd. 116

[18] E.H. Page. *Simulation Modeling Methodology: Principles and Etiology of Decision Support*. PhD thesis, Virigina Polytechnic Institute and State University, 1994. 117

[19] B. Penverne, M. Belkaid, and Herve G. In-situ behavior of pyrimidine pathway enzymes in *Saccheromyces-Cerevisiae*. 4. the channelling of carbamylphosphate to aspartate-transcarbamylase and its partition in the pyrimidine and arginine pathways. *Archives of Biochemistry and Biophysics*, (309):85–93, 1994. 116

[20] H. Praehofer. *System Theoretic Foundations for Combined Discrete-Continuous System Simulation*. PhD thesis, Johannes Kepler University, Linz, Austria, 1992. 117

[21] C. Priami, A. Regev, E. Shapiro, and W. Silvermann. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80:25–31, 2001. 115

[22] J.M. Rohwer, P. W. Postma, B.N. B.N. Kholodenko, and H.V. Westerhoff. Implications of macromolecular crowding for signal transduction and metabolite channeling. *Proceedings of the National Academy of Sciences of the United States of America*, (95):10457–10552, 1998. 115

[23] B. Schattenberg and A.M. Uhrmacher. Planning Agents in James. *Proceedings of the IEEE*, 89(2):158–173, 2001. 119

[24] S. Schulz, T.C. Ewing, and J.W. Rozenblit. Discrete event system specification (devs) and statemate statecharts equivalence for embedded systems modeling. In *Proceedings of the 2000 IEEE Conference and Workshop on Engineering of Computer-Based Systems*, Edinburgh, Scotland, March 2000. 117

[25] P.A. Srere. Complexes of sequential metabolic enzymes. *Annual Review of Biochemistry*, (56):89–124, 1987. 116

[26] Keleti T. and J. Ovadi. Control of metabolism by dynamic macromolecular interactions. *Current Topics in Cellular Regululation*, (29):1–33, 1988. 116

[27] P. Tompa, J. Batke, and J. Ovadi. How to determine the efficiency of intermediate transfer in an interacting enzyme-system. *FEBS Letters*, (214):244–248, 1987. 116

[28] A. M. Uhrmacher. Reasoning about Changing Structure: A Modeling Concept for Ecological Systems. *International Journal on Applied Artificial Intelligence*, 9(2):157–180, 1995. 115

[29] A. M. Uhrmacher, P. Tyschler, and D. Tyschler. Modeling Mobile Agents. *Future Generation Computer System*, 17:107–118, 2000. 117

[30] A.M. Uhrmacher. Dynamic Structures in Modeling and Simulation - a Reflective Approach. *ACM Transactions on Modeling and Simulation*, 11(2):206–232, 2001. 117, 118, 119

[31] B. Vertessy and J. Ovadi. A simple approach to detect active-site-directed enzyme-enzyme interactions. the aldolase/glycerol-phosphate-dehydrogenase enzyme system. *European Journal of Biochemistry*, (164):655–659, 1987. 116

[32] S.J Wakil, J.K. Stoops, and V.C. Joshi. Fatty acid synthesis and its regulation. *Annual Review of Biochemistry*, (52):537–579, 1983. 116

[33] I.C. West. Molecular and physicochemical aspects. In L. Agius and H.S.A. Sheratt, editors, *Channelling in Intermediary Metabolism*, London, 1997. Portland Press Ltd. 116

[34] W. Westerhoff and G.R. Welch. Enzyme organization and direction of metabolic flow: physicochemical considerations. *Current Topics in Cellular Regulation*, (33):361–390, 1992. 116

[35] Z.-L. Xiu, Z.-Y. Chang, and A.-P. Zeng. Nonlinear dynamics of regulation of bacterial *trp* operon: Model analysis of integrated effects of repression, feedback inhibition, and attenuation. *Biotechnology Progress*, 18:686–693, 2002. 124

[36] B.P. Zeigler. *Multifacetted Modelling and Discrete Event Simulation*. Academic Press, London, 1984. 117

Mathematical Modeling of the Influence of RKIP on the ERK Signaling Pathway

Kwang-Hyun Cho[1*], Sung-Young Shin[1], Hyun-Woo Kim[1], Olaf Wolkenhauer[2*], Brian McFerran[3,4] and Walter Kolch[3,5]

[1] School of Electrical Engineering, University of Ulsan
Ulsan, 680-749, Korea
{ckh,syshin,hwkim}@mail.ulsan.ac.kr
[2] Dept. of Biomolecular Sciences and Dept. of Electrical Engineering and Electronics
UMIST, Manchester, M60 1QD, U.K.
olaf.wolkenhauer@umist.ac.kr
[3] Beatson Institute for Cancer Research, Cancer Research UK Beatson Laboratories
Switchback Road, Bearsdn, Glasgow, G61 1BD, U.K.
wkolch@beatson.gla.ac.uk
[4] Organon Laboratories
Newhouse, Motherwell, U.K.
B.McFerran@organon.co.uk
[5] Institute of Biomedical and Life Sciences, University of Glasgow
University Avenue, Glasgow, G12 8QQ, U.K.

Abstract. This paper investigates the influence of the Raf Kinase Inhibitor Protein (RKIP) on the Extracellular signal Regulated Kinase (ERK) signaling pathway through mathematical modeling and simulation. Using nonlinear ordinary differential equations to represent biochemical reactions in the pathway, we suggest a technique for parameter estimation, utilizing time series data of proteins involved in the signaling pathway. The mathematical model allows the simulation the sensitivity of the ERK pathway to variations of initial RKIP and ERK-PP (phosphorylated ERK) concentrations along with time. Throughout the simulation study, we can qualitatively validate the proposed mathematical model compared with experimental results.

1 Introduction

The Ras/Raf-1/MEK/ERK module is a ubiquitously expressed signaling pathway that conveys mitogenic and differentiation signals from the cell membrane to the nucleus. This kinase cascade appears to be spatially organized in a signaling complex nucleated by Ras proteins. The small G protein Ras is activated by many growth factor receptors

* Corresponding authors.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 127-141, 2003.

and binds to the Raf-1 kinase with high affinity when activated. This induces the recruitment of Raf-1 from the cytosol to the cell membrane. Activated Raf-1 then phosphorylates and activate MAPK/ERK Kinase (MEK), a kinase that in turn phosphorylates and activates Extracellular signal Regulated Kinase (ERK), the prototypic Mitogen-Activated Protein Kinase (MAPK). Activated ERKs can translocate to the nucleus and regulate gene expression by the phosphorylation of transcription factors [1]. This kinase cascade controls the proliferation and differentiation of different cell types. The specific biological effects are crucially dependent on the amplitude and kinetics of ERK activity [7]. The adjustment of these parameters involves the regulation of protein interactions within this pathway. Here the Raf-1 kinase inhibitor protein (RKIP) plays a central role. RKIP binds to Raf-1 and MEK thereby disrupting the interaction between Raf-1 and MEK. As a consequence RKIP inhibits the phosphorylation and activation of MEK by Raf-1. RKIP overexpression interferes with the activation of MEK and ERK, induction of AP-1-dependent reporter genes and transformation elicited by an oncogenically activated Raf-1 kinase. Downregulation of endogenous RKIP by expression of antisense RNA or antibody microinjection induces the activation of MEK-, ERK-, AP-1-dependent transcription. Thus, RKIP represents a new class of protein-kinase-inhibitor protein that regulates the activity of the Raf-1/MEK/ERK module [2].

In this paper we are using an integrated approach of mathematical modeling in combination with experimental data to investigate the impact of RKIP on the dynamics of the ERK pathway. For this purpose, we establish a mathematical model of the signaling pathway consisting of a set of nonlinear ordinary differential equations (ODEs) to represent enzyme kinetic reactions. To complete and to simulate this mathematical model, we need to identify (or estimate) all the parameter occurring in these equations. Parameter estimation for nonlinear differential equations is non-trival and provides a major challenge in modeling signaling pathways.

One could argue that parameter estimation is currently the limiting step in biomathematical modeling [3], [4], [6]. In order to tackle this problem, we propose a simple method for parameter estimation utilizing time course measured data. This method first discretizes the nonlinear ODEs into algebraic difference equations that are linear with respect to the parameters and then solve the transformed linear algebraic difference equations to obtain the parameter values at each frozen time point. We can then obtain the required parameter values using regression techniques. Based on the estimated parameter values, we perform a sensitivity analysis of the ERK pathway in order to discuss the influence of RKIP through simulation. The simulation study reveals the sensitivity of all the protein concentration involved in the pathway with respect to the variation of initial RKIP and phosphorylated ERK (ERK-PP) concentrations in a quantitative manner.

The paper is organized as follows. Section 2 briefly introduces the new parameter estimation method for mathematical modeling. Section 3 describes the ERK signaling pathway and its suppression by RKIP. Section 4 shows the parameter estimation of the ERK signaling pathway suppressed by RKIP and the simulation results. Finally, conclusions and further studies are found in Section 5. The detailed mathematical model of the ERK signaling pathway suppressed by RKIP, time course data, and estimated parameter values are summarized in Appendix.

2 Parameter Estimation Based on Time Course Data

In this section we discuss the topic of parameter estimation and exemplify it via the ERK signaling pathway suppressed by RKIP as illustrated in Fig. 1. To estimate parameter values of nonlinear ODEs, we first discretize the given continuous differential equations along with a sample time, which usually corresponds to the time of measurement. Then the continuous differential equations can be approximated by difference equations. Consider the following continuous nonlinear ODEs in (1).

$$\begin{aligned}
\frac{dx_1}{dt} &= f_1(x_1, x_2, \cdots, x_m; k_1, k_2, \cdots k_m) \\
\frac{dx_2}{dt} &= f_2(x_1, x_2, \cdots, x_m; k_1, k_2, \cdots k_m) \\
&\vdots \\
\frac{dx_m}{dt} &= f_m(x_1, x_2, \cdots, x_m; k_1, k_2, \cdots k_m)
\end{aligned} \tag{1}$$

We can approximate the differential operator by a difference operator for small sampling time interval as follows.

$$\frac{dx_i(t_n)}{dt} \cong \left.\frac{x_i(t_n) - x_i(t_{n-1})}{t_n - t_{n-1}}\right|_{i=1,2,\cdots m} \tag{2}$$

Note that this is only a first order approximation and we can alternatively employ any other kind of approximation to reduce the approximation error.

$$\begin{aligned}
\frac{x_1(t_n) - x_1(t_{n-1})}{t_n - t_{n-1}} &\cong f_1\left(x_1(t_n),\ x_2(t_n),\ \cdots x_m(t_n);\ k_1(t_n),\ k_2(t_n), \cdots k_m(t_n)\right) \\
\frac{x_2(t_n) - x_2(t_{n-1})}{t_n - t_{n-1}} &\cong f_2\left(x_1(t_n),\ x_2(t_n),\ \cdots x_m(t_n);\ k_1(t_n),\ k_2(t_n), \cdots k_m(t_n)\right) \\
&\vdots \\
\frac{x_m(t_n) - x_m(t_{n-1})}{t_n - t_{n-1}} &\cong f_m\left(x_1(t_n),\ x_2(t_n),\ \cdots x_m(t_n);\ k_1(t_n),\ k_2(t_n), \cdots k_m(t_n)\right)
\end{aligned} \tag{3}$$

In most of cases when we derive eq. (1) based on enzyme kinetics, (3) becomes a set of linear algebraic difference equations with respect to parameters $k_i(t_n)\big|_{i=1,2,\cdots,m}$ for each frozen time point t_n. Hence we can obtain the following set of estimated parameter values by solving such linear algebraic simultaneous difference equations.

$$\begin{aligned}
k_1(t_n) &\cong g_1\left(x_1(t_n), x_2(t_n), \cdots, x_m(t_{n-1}), x_1(t_{n-1}), x_2(t_{n-1}), \cdots, x_m(t_{n-1})\right) \\
k_2(t_n) &\cong g_2\left(x_1(t_n), x_2(t_n), \cdots, x_m(t_{n-1}), x_1(t_{n-1}), x_2(t_{n-1}), \cdots, x_m(t_{n-1})\right) \\
&\vdots \\
k_m(t_n) &\cong g_m\left(x_1(t_n), x_2(t_n), \cdots, x_m(t_{n-1}), x_1(t_{n-1}), x_2(t_{n-1}), \cdots, x_m(t_{n-1})\right)
\end{aligned} \tag{4}$$

The resultant estimated parameter values $k_i(t_n)\big|_{i=1,2,\cdots,m}$ might include time dependent experimental noise components which can be further eliminated by regression techniques. After all, if the system itself is time-independent then these time series of $k_i(t_n)\big|_{i=1,2,\cdots,m}$ will converge to certain steady-state values. Otherwise it can be approximated by interpolation of polynomial functions. This approach has two advantages: 1) it is applicable to not only time-invariant systems (estimation of constant parameter values) but also time-varying systems (estimation of a parameter function of time), 2) the estimation error can be reduced by decreasing the sampling time interval. On the other hand it is disadvantageous in that it requires multiple measurements for time course data of each protein involved in the signaling pathway and is therefore unlikely to be popular with the experimentalists. However, this does not impair the proposed method since we can transform commonly used WB (western blotting) data into the required concentration data under reasonable assumptions.

3 ERK Signaling Pathway Regulated by RKIP

This section illustrates the foregoing parameter estimation method using a mathematical model of the ERK signaling pathway regulated by RKIP. Consider first a graphical representation of the signaling pathway shown in Fig. 1 where m_1 denotes the concentration of the activated Raf-1 (also denoted as Raf-1*), m_2 denotes the concentration of RKIP, m_3 denotes the concentration of Raf-1*/RKIP complex, and so on. In fact, Fig. 1 illustrates only part of the ERK pathway, i.e. it considers the subset of the ERK pathway regulated by RKIP. In the remainder of this paper we confine our discussion on parameter estimation to this pathway. Figure 1 describes the following consecutive mechanisms: 1) RKIP inhibits Raf-1* to phosphorylate MEK by binding to Raf-1* and forming a complex Raf-1*/RKIP, 2) Free Raf-1* phosphorylates MEK and converts inactive MEK into active MEK-PP. MEK-PP binds to ERK and phosphorylates it to active ERK-PP. ERK-PP can interact with the Raf-1*/RKIP complex to form a Raf-1*/RKIP/ERK-PP complex, 3) Then ERK-PP phosphorylates RKIP into RKIP-P causing the release of Raf-1* from RKIP-P. ERK does not dephosphorylate itself. ERK is dephosphorylated by Protein Phosphatase 2A (PP2A) and MAPK Phosphatases, MKPs [8]. The expression of MKP-1 is transcriptionally induced by ERK. In contrast, MKP-3 is constitutively expressed, but binding of ERK enhances its phosphatase activity. MKP-3 seems to serve as an anchor protein which binds ERK in resting cells keeping it inactive. PP2A is a constitutively expressed and constitutively active phosphatases, which probably is regulated by selective targeting to its substrates, 4) Raf-1* returns to its original active state Raf-1* after being released from the Raf-1*/RKIP-P/ERK-PP complex, 5) The timecourse analysis of ERK-PP and Raf-1/RKIP complex formation rather suggest the following: The inrease in free Raf-1* causes a further increase of MEK-PP and eventually ERK-PP, 6) The RKIP-phosphotase (RP) is artificially introduced to complete this model by showing the

dephosphorylation of RKIP-P into the original active state RKIP. After dephosphorylation RKIP binds to Raf-1* and suppresses further phosphorylation and activation of MEK. MEK-PP and ERK-PP levels rapidly decline due to efficient dephosphorylation by PP2A and MKPs.

Based on enzyme kinetics, we can derive a mathematical model, a set of nonlinear ODEs of this signaling pathway with respect to state variables $m_i\big|_{i=1,2,\cdots 11}$ as shown by (5) in Appendix. For the purpose of simulation of this mathematical model, the initial value of all complex proteins and product proteins are assumed to be zero and other initial values such as signaling proteins, $m_1, m_2, m_7, m_9, m_{10}$ are defined according to measured time course data. Based on the time course data, found in Table 2 and 3 of the Appendix, we can apply the proposed parameter estimation to this model, which is explained in detail in next section. The resultant parameter time series are also summarized in Table 4 and 5 of the Appendix.

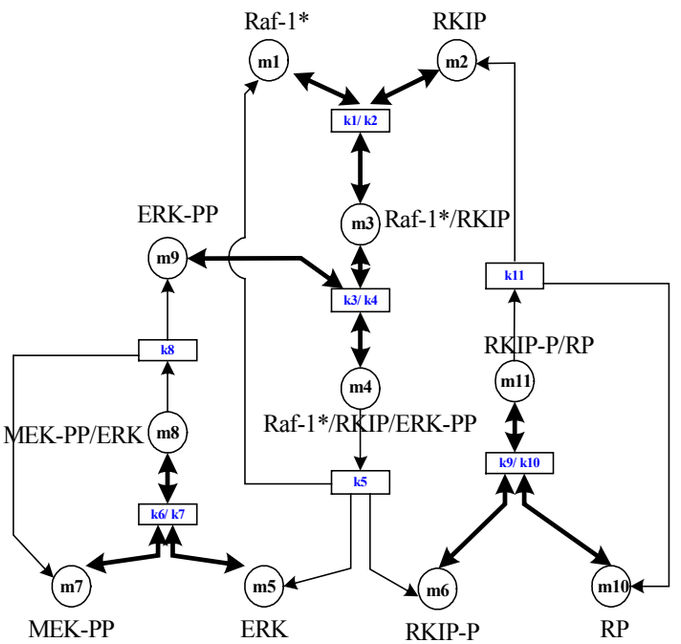


Fig. 1. Graphical representation of the ERK signaling pathway Regulated by RKIP: a circle represents a state for the concentration of a protein and a bar a kinetic parameter of reaction to be estimated. The directed arc (arrows) connecting a circle and a bar represents a direction of a signal flow. The bi-directional thick arrows represent a association and a dissociation rate at same time. The thin unidirectional arrows represent a production rate of productsEINBETTEN

4 Parameter Estimation and Simulation Studies

4.1 Parameter Estimation

The estimated parameters usually appear as a time dependent profile since the time course data include various uncertain factors such as transient responses, noise terms, etc. However, if the signal transduction system itself is inherently time-invariant then estimated parameter profile should converge to a certain constant value at steady-state. Therefore we have to find this convergence value if the system is time-invariant. Otherwise we have to derive an interpolated polynomial function of time for time-varying systems.

Figures 2 to 4 show the process of parameter estimation from the time series. Estimated parameter values can be found as the convergence point since the parameter profile shows time-invariant characteristics. In each plot, we calculate the converging parameter value and the resultant estimation as summarized in Table 1. These estimated parameter values are used for the simulation studies in the following sections.

Table 1. Summary of the estimated parameter values

Parameter	Estimated Value	Parameter	Estimated Value	Parameter	Estimated Value	Parameter	Estimated Value
k1	0.53	k2	0.0072	k3	0.625	k4	0.00245
k5	0.0315	k6	0.8	k7	0.0075	k8	0.071
k9	0.92	k10	0.00122	k11	0.87		

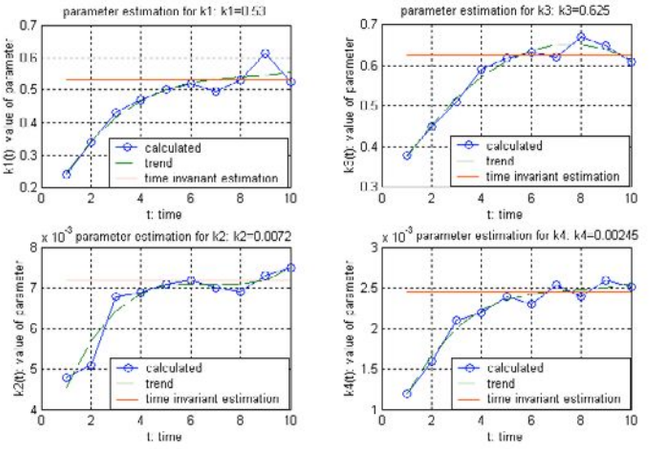


Fig. 2. Parameter estimation process from time series data: the upper left shows Raf-1*/RKIP complex association parameter k1, the upper right shows Raf-1*/RKIP/ERK-PP association parameter k3, the lower left shows Raf-1* and RKIP dissociation parameter k2, and the lower right shows ERK-PP and Raf-1*/RKIP complex dissociation parameter k4

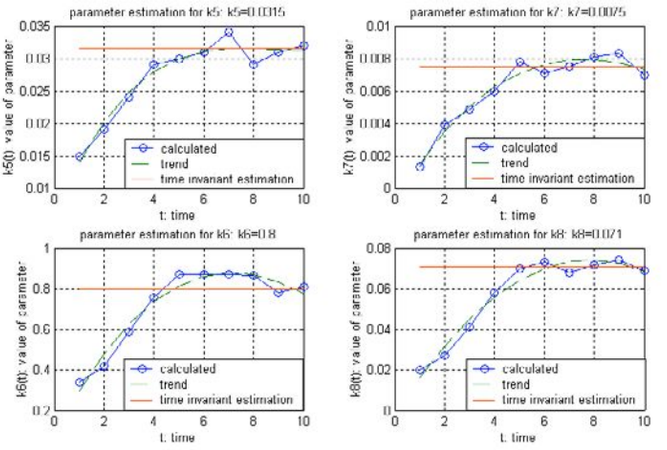


Fig. 3. Parameter estimation (cont'd): the upper left shows Raf-1*, ERK-P, and RKIP-P production parameter k5, the upper right shows MEK-PP and ERK dissociation parameter k7, the lower left shows MEK-PP/ERK association parameter k6, and the lower right shows ERK-PP production paramter k8. James O'Ferrell's data from frog oocytes that ERK phosphorylation is non-processive, ie proceeds ERK->ERK-P->ERK-PP, may not apply in mammalian cells. We looked in NIH cells and did not see it. Therefore, and for the sake of simplicity we can simply assume that the reaction catalysed by MEK is ERK->ERK-PP

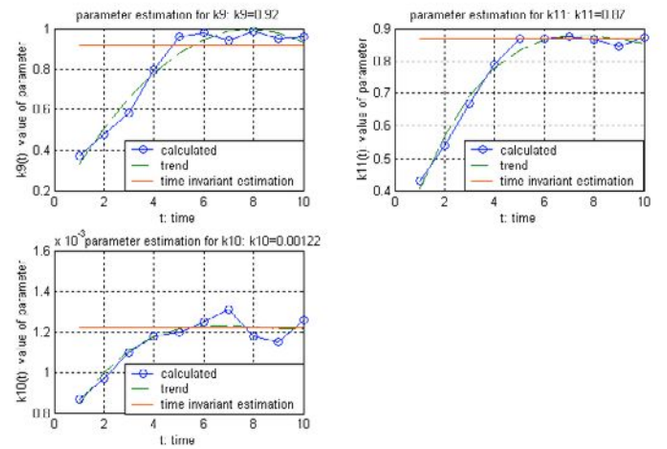


Fig. 4. Parameter estimation process upon the time course data: the upper left shows RKIP-P/RP association parameter k9, the upper right shows RKIP and RP production parameter k7, and the lower left shows RKIP-P and RP dissociation parameter k10

4.2 Two-Dimensional Simulation for Fixed Initial Conditions

Based on the mathematical model summarized in Appendix and the estimated parameter values in Table 1, we perform simulation studies to analyze the signal transduction system with respect to the sensitivity for the variation of RKIP and ERK-PP. For this purpose, we first simulate the pathway model according to the variation of the initial concentration of RKIP (RKIP sensitivity analysis). Next we perform the simulation according to the variation of the initial concentration of ERK-PP in this case (ERK-PP sensitivity analysis). Since ERK-PP causes the release of Raf-1* from the Raf-1*/RKIP/ERK-PP complex, the inhibitory effect of RKIP is dependent upon the concentration of ERK-PP and therefore this ERK-PP sensitivity analysis is very important to understand the ERK pathway suppressed by RKIP.

Figure 5 shows the simulation results for fixed initial conditions. The upper left of Fig. 5 shows the dynamics of Raf-1*, RKIP, and Raf-1*/RKIP complex. We can see Raf-1* quickly binds to RKIP, but after a while Raf-1* and RKIP reach their steady states. The upper right shows the activity of MEK-PP which phosphorylates ERK. The concentration of ERK-P/MEK-PP complex increases up to about 0.5 μM. Although it is not shown here, this complex concentration however might decrease after a while since the phosphorylated ERK-P (ERK-PP) is released from the complex. The lower left shows the activity of ERK-PP which phosphorylates RKIP in the Raf-1*/RKIP complex and then also releases Raf-1* from the complex. We can see that Raf-1*/RKIP complex also decreases to a steady state. The lower right shows the activity of the RKIP Phosphatase (RP). In this case, it changes very little; however, for different initial conditions corresponding to different microenvironments such as different *in-vivo* conditions it could change dramatically. After all, throughout these simulation results, we can confirm the crucial role of RKIP as an inhibitor of the ERK pathway by binding to Raf-1* and dissociating the Raf-1*/MEK complex .Our simulation allows us to now make quantitative predictions of the impact of RKIP on the activity of the ERK pathway in different cells lines and at different conditions of stimulation.

4.3 Sensitivity Analysis According to the Variation of Initial RKIP

In this section, we show the sensitivity analysis with respect to the variation of initial RKIP. RKIP has been recently identified as a potent Raf-1* inhibitor in the ERK pathway. RKIP binds to Raf-1* and thereby prevents Raf-1* from phosphorylating and activating MEK. Based on the variation of the concentration of RKIP from 0.5 to 5 μM, we investigate its effect to the concentration change of other proteins. We find that if the initial concentration of RKIP increases over time then the concentration of active MEK-PP and ERK-PP decreases while the concentrations of inactive MEK and ERK increase at the same time. Likewise, when the concentration of RKIP increases linearly along with time, the concentration of Raf-1* available to activate MEK decreases exponentially. Under these conditions RKIP-P increases exponentially, because more RKIP is available for phosphorylation by ERK-PP, and ERK-PP does not immediately decline when MEK activation is inhibited. Eventually, as ERK-PP is being dephosphorylated, the levels of MEK and RKIP-P reachsteady-statevalues. Likewise MEK-PP decreases to steady-state level. If all Raf-1* becomes engaged by

RKIP then the ERK pathway becomes completely suppressed since no MEK-PP can be generated.

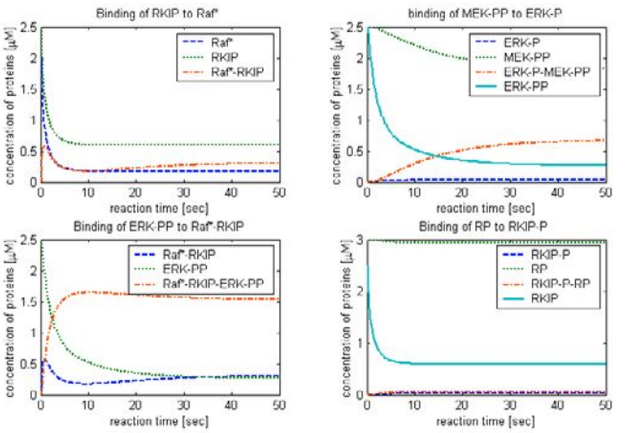


Fig. 5. Simulation results of the mathematical modeling for fixed initial condition: the upper left shows the dynamics for Raf-1*, RKIP, and their complex Raf-1*/RKIP, the upper right shows the activity of MEK-PP which phosphorylates and activates ERK, the lower left shows the activity of ERK-PP, and the lower right shows the activity of RP

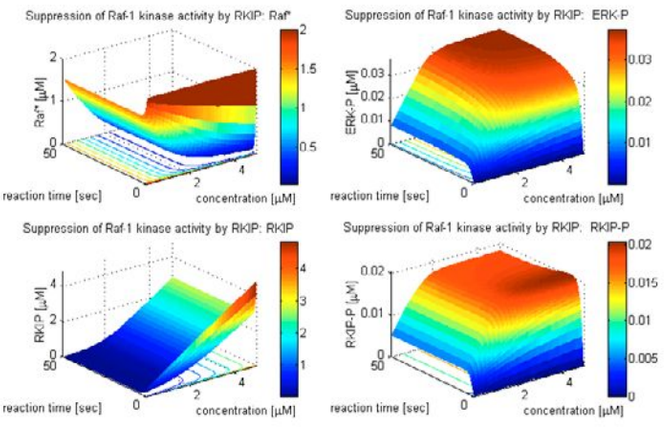


Fig. 6. The simulation results according to the variation of the concentration of RKIP: The upper left shows the change of concentration of Raf-1*, the upper right shows ERK, the lower left shows RKIP, and the lower right shows RKIP-P

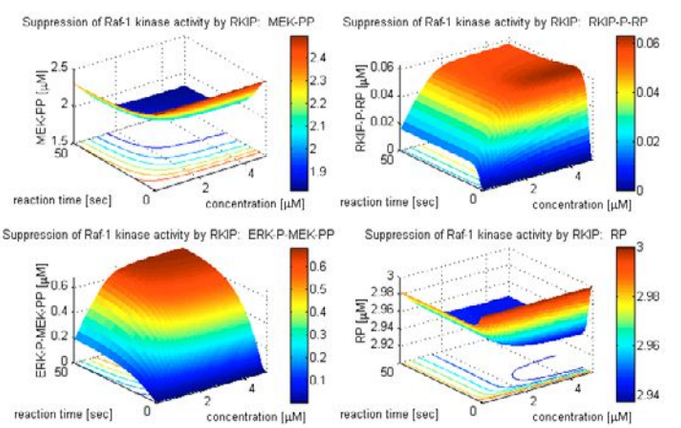


Fig. 7. The simulation results according to the variation of the concentration of RKIP (continued): The upper left shows the change of concentration of MEK-PP, the upper right shows RKIP-P-RP, the lower left shows ERK-P-MEK-PP, and the lower right shows RP

4.4 Sensitivity Analysis According to the Variation of Initial ERK-PP

In this section, we show the sensitivity analysis with respect to the variation of initial ERK-PP reflecting different basal activities of the ERK pathway in different cell types. By changing the concentration of ERK-PP from 0.5 to 5 μM, we investigate the change of concentration for other proteins.

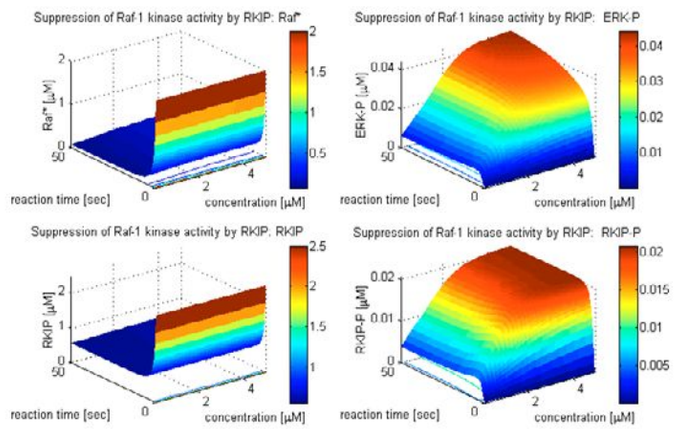


Fig. 8 The simulation results according to the variation of the concentration of ERK-PP: The upper left shows the change of concentration of Raf-1*, the upper right shows ERK-P, the lower left shows RKIP, and the lower right shows RKIP-P

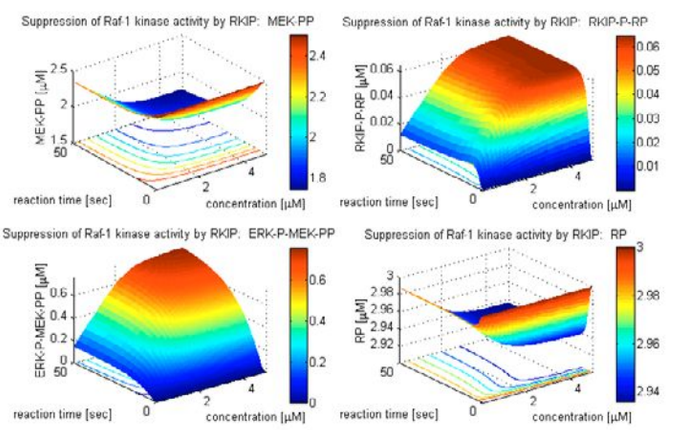


Fig. 9 The simulation results according to the variation of the concentration of ERK-PP (continued): The upper left shows the change of concentration of MEK-PP, the upper right shows RKIP-P-RP, the lower left shows ERK-P-MEK-PP, and the lower right shows RP

5 Concluding Remarks and Further Studies

In this paper, we introduced a system-theoretic approach to the analysis and quantitative modeling of the ERK pathway regulated by RKIP. To this end, a parameter estimation technique was proposed by first approximating the differential operator by a difference operator and then by utilizing time course data to resolve the transformed simultaneous linear algebraic difference equations with respect to parameters for each frozen time point. The estimated parameter time series show a profile converging to a constant value at steady-state, suggesting signal transduction system is time-invariant. For a simulation study of the ERK pathway regulated by RKIP, we made use of these parameter values in a model of nonlinear ODEs. Based on this mathematical model, we performed the sensitivity analysis of the pathway with respect to the initial RKIP and ERK-PP variation. These simulation studies provide a qualitative validation of the mathematical model compared to experimental results in view of the transient behavior and sensitivity analysis.

A remarkable result of this simulation is that RKIP inhibition of the Raf-MEK-ERK pathway is non-linear. This is completely unexpected given that RKIP acts as a stoichiometric inhibitor, which in enzyme kinetic assays behaves like a competitor for substrate, i.e. MEK [1]. The non-linearity appears to be caused by a previously unaccounted feedback phosphorylation of RKIP induced through the ERK pathway. This phosphorylation diminishes the affinity of RKIP for Raf-1 [McFerran et al., manuscript in preparation]. As Fig. 6 shows the maximal sequestration of Raf-1 by RKIP occurs at a particular RKIP expression level. However, the prevention of MEK-PP accumulation is mainly a function of time. The expression level of RKIP determines

the kinetics of MEK-PP decrease rather than the final extent (Fig. 7). At the level of ERK activation this translates into a response curve with a steep initial slope, but very different levels of final activity (Fig. 7). Importantly, this predicted behavior correlates with experimental data that revealed a non-linearity of RKIP inhibition of the Raf-MEK-ERK pathway [2]. In these experiments RKIP mediated inhibition reached a saturation level rather than showing a linear decline. Our simulation model confirms and explains this experimentally determined behavior of the pathway. It also makes a further important prediction, namely that RKIP modulates the final extent and duration of ERK activity rather than the initial activation kinetics. This property assigns RKIP a crucial role as a decision maker in situations where the level and duration of ERK activity determines the biological outcome. It will be interesting to test this prediction in a relevant biological system such as PC12 cell. The proliferation of these cells is stimulated by a short burst of ERK activity, whereas differentiation requires a sustained level of high ERK activity. Our simulation model would predict that RKIP will preferentially interfere with differentiation and would redirect the biological decision to a proliferation response.

In order to further investigate the influence of RKIP on the ERK signaling pathway, we wish to expand on the proposed mathematical modeling and simulation studies to cover more aspects of the ERK pathway. This will involve time consuming experiments to allow parameter identification from time series data. However, we deem it necessary to develop simulation models hand in hand with experimental validation. The current study shows that simulation actually can aid in the interpretation of experimental data and promote the formulation of new hypotheses that provide direction in the investigation of complex biological systems.

Acknowledgement

This work was supported by the Post-doctoral Fellowship Program of Korea Science & Engineering Foundation (KOSEF) and by a grant from the Association for International Cancer Research (AICR).

References

[1] Yeung, K., Janosch, P., Rose, D. W., Mischak, H., Sedivy, J. M., and Kolch, W.: Mechanism of Suppression of the Raf/MEK/Extracellular Signal-Regulated Kinase Pathway by the Raf Kinase Inhibitor Protein. MOL. CELL. BIOL. 20 (2000) 3079-3085

[2] Yeung, K, Seitz, T., Li, S., Janosch, P., McFerran, B., Kaiser, C., Fee, F., Katsanakis, K. D., Rose, D. W., Mischak, H., Sedivy, J. M., and Kolch, W.: Suppression of Raf-1 kinase activity and MAP kinase signaling By RKIP. Nature. 401 (1999) 173-177

[3] Bock, H. G.: Numerical treatment of inverse problems in chemical reaction kinetics. Modeling of Chemical Reaction Systems. 18 (1981) 102-125

[4] Hegger, R., Kantz, H., Schmuser, F., Diestelhorst, M., Kapsch, R. P., and Beige, H.: Dynamical properties of a ferroelectric capccitor observed through nonlinear time series analysis. Chaos. 8 (1998) 727-736
[5] Hegger, R., Kantz, H., Schmuser, F., Diestelhorst, M., Kapsch, R. P., and Beige, H.: Dynamical properties of a ferroelectric capccitor observed through nonlinear time series analysis. Chaos. 8 (1998) 727-736
[6] Voit, E. O., Computational Analysis of Biochemical Systems, Cambridge University Press: Cambridge, U.K., 2000
[7] Marshall, C. J., Specificity of receptor tyrosine kinase signaling: transient versus sustained extracellular signal-regulated kinase activation: Cell. 80 (1995) 179-185
[8] Zhou, B., Wang, Z. X., Zhao, Y., Brautigan, D. L., and Zhang, Z. Y.:The specificity of extracellular signal-regulated kinase 2 dephosphorylation by protein phosphotases. J Biol Chem. 277 (2002) 31818-31825

Appendix: The Mathematical Model of ERK Pathway Regulated by RKIP, Time Course Data, and Estimated Parameter Values

The following set of nonlinear ODEs (5) represents the developed mathematical model of the ERK pathway suppressed by RKIP shown in Fig. 1.

$$
\begin{aligned}
\frac{dm_1(t)}{dt} &= -k_1 m_1(t) m_2(t) + k_2 m_3(t) + k_5 m_4(t) \\
\frac{dm_2(t)}{dt} &= -k_1 m_1(t) m_2(t) + k_2 m_3(t) + k_{11} m_{11}(t) \\
\frac{dm_3(t)}{dt} &= k_1 m_1(t) m_2(t) - k_2 m_3(t) - k_3 m_3(t) m_9(t) + k_4 m_4(t) \\
\frac{dm_4(t)}{dt} &= k_3 m_3(t) m_9(t) - k_4 m_4(t) - k_5 m_4(t) \\
\frac{dm_5(t)}{dt} &= k_5 m_4(t) - k_6 m_5(t) m_7(t) + k_7 m_8(t) \\
\frac{dm_6(t)}{dt} &= k_5 m_4(t) - k_9 m_6(t) m_{10}(t) + k_{10} m_{11}(t) \\
\frac{dm_7(t)}{dt} &= -k_6 m_5(t) m_7(t) + k_7 m_8(t) + k_8 m_8(t) \\
\frac{dm_8(t)}{dt} &= k_6 m_5(t) m_7(t) - k_7 m_8(t) - k_8 m_8(t) \\
\frac{dm_9(t)}{dt} &= -k_3 m_3(t) m_9(t) + k_4 m_4(t) + k_8 m_8(t) \\
\frac{dm_{10}(t)}{dt} &= -k_9 m_6(t) m_{10}(t) + k_{10} m_{11}(t) + k_{11} m_{11}(t) \\
\frac{dm_{11}(t)}{dt} &= k_9 m_6(t) m_{10}(t) - k_{10} m_{11}(t) - k_{11} m_{11}(t)
\end{aligned}
\tag{5}
$$

The following Table 2 and 3 show the time course data used for parameter estimation.

Table 2. Time course data used for parameter estimation

Time \ States[μM]	m_1	m_2	m_3	m_4	m_5	m_6
t_0	120.00	48.00	1.500	4.800	12.400	4.800
t_1	67.95505535	0.3727182703	0.09086956972	58.25407508	0.08846427918	0.2285471542
t_2	66.81423242	0.05476449029	0.1696162456	59.46880595	0.044260337	0.01466380983
t_3	66.71360883	0.05019133182	0.01767995587	59.56871120	0.04176942763	0.01527704430
t_4	66.75214784	0.05490721213	0.01816772905	59.52968441	0.03971032950	0.01349091559
t_5	66.63570088	0.05405994303	0.01801622209	59.64628287	0.03590036227	0.01158526073
t_6	66.68508855	0.05314008775	0.01784579291	59.59706563	0.03603602473	0.01172030400
t_7	66.86290844	0.06026147126	0.01994358152	59.41714795	0.04015689155	0.01337477790
t_8	66.58789649	0.04974222183	0.01621135867	59.69589212	0.03435528707	0.01089921955
t_9	66.72289041	0.04488809471	0.01739192477	59.55971763	0.04010491475	0.01208455211
t_{10}	66.74940877	0.05439326527	0.01921204205	59.53137916	0.03968304675	0.01232880325

Table 3. Time course data for parameter estimation (continued)

Time \ States [μM]	m_7	m_8	m_9	m_{10}	m_{11}
t_0	87.8	3.7	240.2	160.5	2.7
t_1	70.87529418	20.62470582	182.1327548	160.1405176	3.059482367
t_2	66.32835993	25.17164007	176.4152936	160.9551958	20244804126
t_3	63.54630193	27.95369807	173.5358213	161.0518595	2.148140465
t_4	62.88014832	28.61985168	172.9107536	161.0162502	2.183749727
t_5	64.13673735	27.36326265	174.0545541	161.1299442	2.070055695
t_6	65.00392062	26.49607938	174.9708189	161.0797717	2.120228173
t_7	63.70607809	27.79392191	173.8487732	160.9107277	2.289272205
t_8	65.27236498	26.22763502	175.1421175	161.1727448	2.027255065
t_9	65.81873686	25.68126314	175.8189142	161.0340821	2.165917780
t_{10}	65.03152288	26.46847712	175.0604606	161.0173132	2.182686714

The following Table 4 and 5 summarize the estimated parameter values.

Table 4. The estimated parameter values

Time \ Parameter	k_1	k_2	k_3	k_4	k_5	k_6
t_1	0.24	0.0048	0.38	0.0012	0.015	0.34
t_2	0.34	0.0051	0.45	0.0016	0.019	0.42
t_3	0.43	0.00678	0.51	0.0021	0.024	0.59
t_4	0.47	0.00688	0.59	0.0022	0.029	0.76
t_5	0.5	0.0071	0.62	0.0024	0.03	0.87
t_6	0.52	0.0072	0.634	0.0023	0.031	0.869
t_7	0.497	0.00701	0.621	0.00254	0.034	0.871
t_8	0.531	0.00691	0.67	0.0024	0.029	0.867
t_9	0.612	0.0073	0.65	0.0026	0.031	0.78
t_{10}	0.524	0.0075	0.61	0.00251	0.032	0.81

Table 5. The estimated parameter values (continued)

Time \ Parameter	k_7	k_8	k_9	k_{10}	k_{11}
t_1	0.0013	0.02	0.37	0.00087	0.43
t_2	0.00392	0.027	0.48	0.00097	0.54
t_3	0.00487	0.0412	0.582	0.0011	0.67
t_4	0.00598	0.058	0.796	0.00118	0.789
t_5	0.0078	0.07	0.96	0.0012	0.87
t_6	0.0071	0.073	0.98	0.00125	0.869
t_7	0.0075	0.068	0.94	0.00131	0.875
t_8	0.0081	0.072	0.987	0.00118	0.867
t_9	0.0083	0.074	0.95	0.00115	0.846
t_{10}	0.007	0.069	0.961	0.00126	0.872

A Method to Identify Essential Enzymes in the Metabolism: Application to *Escherichia Coli*

Ney Lemke, Fabiana Herédia, Cláudia K. Barcellos, and José C. M. Mombach

Laboratório de Bioinformática e Biologia Computacional
Centro de Ciências Exatas e Tecnológicas,
Universidade do Vale do Rio dos Sinos,
93022-000 São Leopoldo, RS, Brazil
{Mombach,Lemke}@exatas.unisinos.br
http://www.inf.unisinos.br/~lbbc

Abstract. In this work we propose a quantitative definition of enzyme importance in a metabolic network. Using a graph analysis, we investigate the metabolism of the *Escherichia coli* to determine a relation between the damage generated on the metabolic network by the deletion of an enzyme and the experimentally determined viability of the organism in the absence of the enzyme. We predict that a large fraction (91%) of enzymes cause a small damage and have a low probability of lethality according to experimental results, while a small fraction of them (9%) cause a high damage and have a high probability of lethality. We obtain a quantitative correlation between damage and its probability of lethality. Our results may be a universal property of the metabolic network of other organisms.

1 Introduction

Cellular metabolism is characterized by a complex network of reactants connected by chemical reactions catalyzed by specialized proteins called enzymes. For practical purposes, without catalysts the reactions do not occur. The reactions are organized on modules called metabolic maps with specific catabolic or anabolic functions. The complete set of metabolic maps is called metabolic network. The understanding of the architecture and dynamics of this web of reactions is a major challenge for modern science.

The development of realistic models for metabolic networks is still beyond our capabilities, since the experimental determination of the set of kinetic parameters that characterize each of the hundreds reactions that occur in a cell is a defiant problem. On the other hand, there is an exponential growth of organisms with sequenced genomes where many encoded proteins are being determined with tolerable confidence [1]. Assuming that the annotated enzymes are expressed, we can build the metabolic network of an organism. A possible approach is to analyze the static structure of the components of the network to infer causal relationships.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 142–148, 2003.

Barabási and coworkers [2] have introduced a graph representation of the metabolic network, where the nodes and the links connecting the nodes, denote metabolites and chemical reactions, respectively. Their analysis of the networks of 43 organisms have shown that they have an inhomogeneous connectivity. A few nodes are highly connected while the majority have quite a small number of connections, obeying a power law distribution of the type: $P(k) \propto k^{-\gamma}$, where k is the node connectivity and γ is an exponent with value close to 2. γ is approximately conserved among the studied organisms suggesting a universal feature. In these networks the highly connected nodes, i.e. those that participate in a higher number of reactions (refereed to as hubs) have a central importance in the integration of the network by connecting the majority of the nodes. In order of decreasing connectivity, some of the main hubs are H_2O, ATP, ADP, phosphate, pyrophosphate, NAD etc. [2]. More recently Ravasz et al. [3], found that metabolic networks are in fact a special type of scale-free networks. The essential difference is that they are hierarchically organized in strongly interacting modules corresponding closely to the well known metabolic maps.

2 The Model

The previous works focus on the relevance of specific metabolites. However, from a practical and biological point of view, it is more important to investigate the enzymatic influence on the network. Differently of metabolites that are not encoded in a genome, enzymes are subjected to evolution and can be genetically engineered to change the metabolic output. They can also be used as targets for drugs [4], where a critical issue is the determination of the important enzymes. The main aim of the present report is to show that it is possible to determine a quantitative criterion of importance for enzymes from the analysis of a graph representation of the network.

Our quantitative criterion for enzyme importance is defined in terms of the deleterious effect of the removal of an enzyme from the network. Since the whole set of kinetic parameters are not known and we have rather incomplete information on the regulatory network we cannot predict theoretically all the consequences of the deletion of a specific enzyme. One possible result that can be determined is the number of metabolites whose production is prevented without the enzyme, what we define as damage d on the network. In order to validate the results, they have to be compared to experimental measurements of importance. The common approach is to relate importance to lethality [5]. Experiments using systematic mutagenesis have identified a substantial number of essential enzymes of *E. coli* and this information has been made publicly available. An integrative database is the *Profiling of the Escherichia coli chromosome* (PEC) database [6] maintained by the Genetic Resource Committee of Japan. The PEC database compiled information that characterizes the *E. coli* genome including a gene classification based on essentiality for cell growth. The genes are classified into three groups: essential, non-essential and unknown.

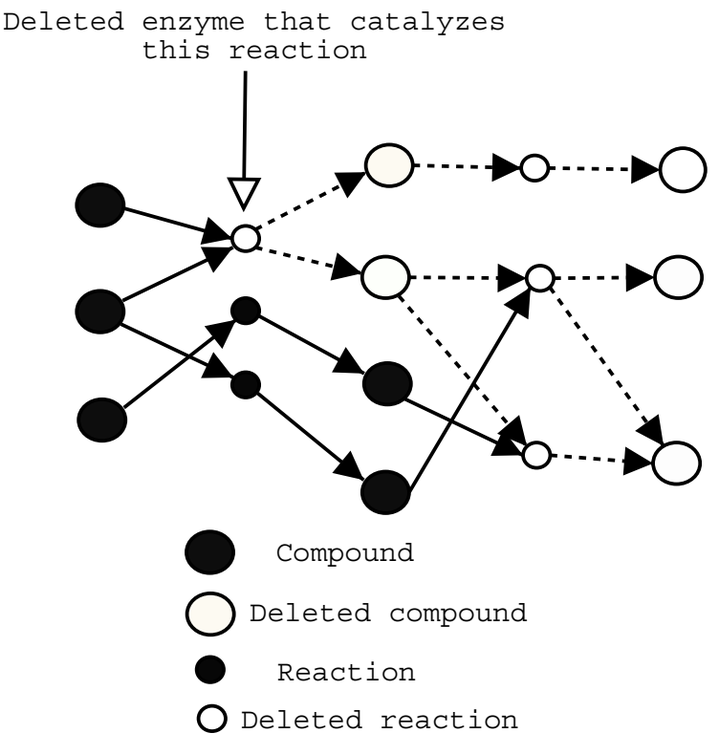


Fig. 1. Schematic representation of the damage on a metabolic network. Small and large nodes represent chemical reactions and metabolites, respectively. Open symbols represent the absence of a chemical reaction or metabolite. The figure illustrates the effect of a deleted chemical reaction at left and the subsequent metabolites and reactions affected

To calculate d, we use a special graph representation for the metabolism of *E. Coli*. The graph is directed and has two types of nodes. One type represents chemical reactions and the other metabolites (see Figure 1). This graph can be classified as a bipartite digraph [7]. A link between a reaction and a metabolite is directed towards the metabolite, if it is a product and in the opposite direction, if the metabolite is a reactant. Reversible reactions were treated as two separate reactions.

The graph was constructed based on the list of reactions catalyzed by the enzymes involved in the small molecules metabolism of *E. coli* proposed by Palsson [8]. We verified the reaction set in the KEGG [9], ERGO [10], and EcoCyc [11] databases and formatted it for our use.

The algorithm we propose is defined as follows (see Fig. 1): Initially, we choose an enzyme and remove the metabolites exclusively produced by that enzyme. In

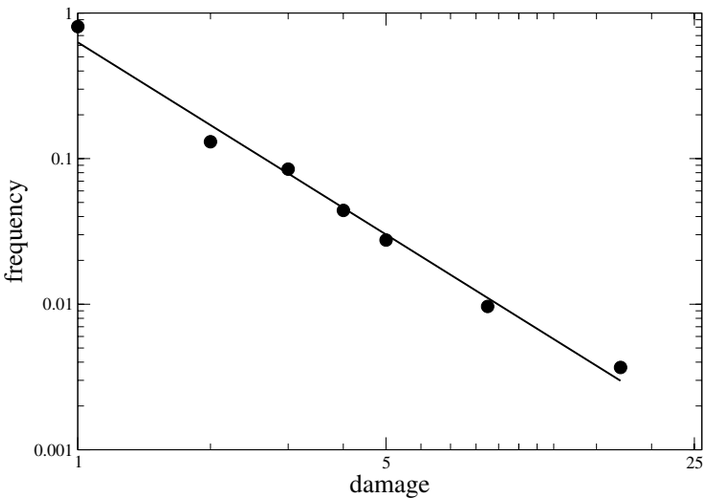


Fig. 2. Frequency of enzymes with damage d. The fitted line corresponds to a power law of the form $P(d) = \alpha d^{-\lambda}$, with $\alpha = 0.54$ and $\lambda = 2.04$. The correlation coefficient of the fit is 0.994.

Table 1. List of 10 predicted most damaging enzymes and information associated with them. First column: Enzyme name. Second column: Damage d. Third column: Experimentally determined essentiality character: E (essential), N (non-essential). Fourth column: Catalyzed metabolites.

Enzyme	d	E	**Product**
Ribose-phosphate pyrophosphokinase	22	E	Phosphoribosyl pyrophosphate
3-dehydroquinate dehydratase	21	N	Dehydroshikimate
Phosphoglucosamine mutase	20	E	Glucosamine 1-phosphate
Shikimate 5-dehydrogenase	20	N	Shikimate
UDP-N-acetylglucosamine pyrophosphorylase	19	E	UDP N-acetyl glucosamine
3-phosphoshikimate 1-carboxyvinyltransferase	18	N	3-Phosphate-shikimate
Acetyl-CoA carboxylase carboxyl transferase	18	E	Malonyl-CoA
Malonyl CoA-acyl carrier protein transacylase	17	E	Malonyl-ACP
3-oxoacyl-[acyl-carrier-protein] synthase	17	E	Acetyl-ACP
Chorismate synthase	17	N	Chorismate

some cases the damage spreads, preventing the occurrence of other reactions and the production of other metabolites. The total number of deleted metabolites is then counted to obtain the damage d.

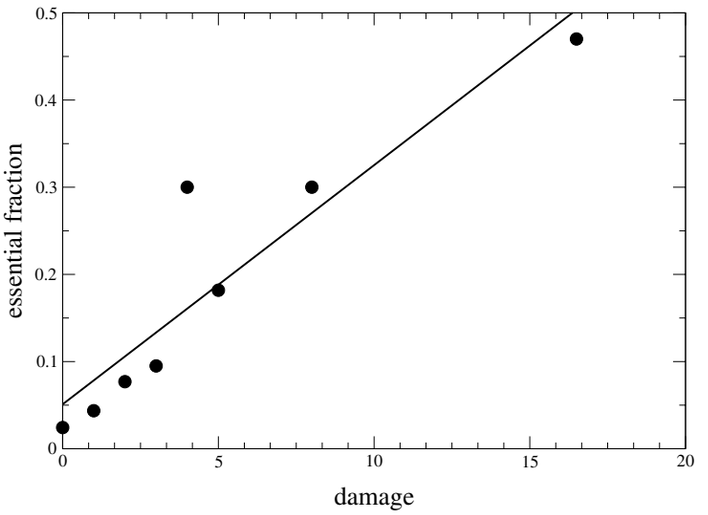


Fig. 3. Plot of the fraction of essentials in groups of enzymes sorted according to the damage d. The p-value of the fit is 0.008.

3 Results

In Fig. 2 we present a histogram of the number of enzymes with a given d. The distribution is well fitted by a power law of the form $P(d) = \alpha d^{-\lambda}$, with $\alpha = 0.54$ and $\lambda = 2.04$. This result supports the idea of metabolic network robustness [2], [5] since it shows that the vast majority of the enzymes, when deleted, cause only a small damage to the network. Table 1 presents a list of the 10 predicted most damaging enzymes. From this set of 10, 6 are essential according to data on the PEC database, suggesting that our definition of damage is useful.

To compare our results with the database of essential enzymes of *E. coli*, we sorted the enzymes according to their d values and determined the fraction of essentials in each group. The results are shown in Fig. 3. Using a F-test, we determined that the correlation between the fraction of essentials versus d is statistically significant with a P-value of 0.008.

In another statistical analysis, we separated the enzymes in two sets. One set for $d < 5$ and another for $d \geq 5$. The first group has 91% of the total number of enzymes and the second 9%. We find that the second group has 50% of all essential enzymes. According to chance, the probability of finding this or a higher frequency of essential enzymes in such a small group of the total enzymes is 10^{-7}. Both statistical analysis give strong support to the usefulness of the damage as a quantitative measure of enzyme importance.

4 Discussion

The analysis of the ten most damaging enzymes in Table 1 confirms that our model is able to identify targets for drugs since many of the listed enzymes are subject of investigation with this purpose. 6 enzymes in the list are key compounds that link different metabolic pathways. 4 enzymes are involved in the production of chorismate that is used in the biosynthesis of aromatic aminoacids, folate, and ubiquinone. The most damaging enzyme, ribose- phosphate- pyrophosphokinase, generates PRPP that is the initial compound of four different pathways and is involved in the intermediate metabolism. Two enzymes are involved in the biosynthesis of cell wall compounds which are the most attractive targets for the development of antibacterial drugs [12], [13].

We conclude from our investigations that highly damaging enzymes are involved in the production of compounds of small connectivity that connect important parts of the metabolism [2]. In opposition, highly connected compounds tend to be redundant since they are produced by many reactions.

It is surprising that some essential enzymes cause a small damage and conversely, some non-essential enzymes cause a high damage. In the case of essential enzymes with low damage, we analyzed the bibliography provided by PEC, and verified that the majority of them might be involved in other important biological functions than only the metabolism of small molecules. In the case of non-essential enzymes with high damage, they have their influence restricted to a module of the metabolism that is not necessary in the environment where the bacteria is growing. It is not clear whether the microorganism are viable in the wild.

An additional application of our method is genetic engineering. The reactions and enzymes responsible for the production of a given compound can be investigated to help the determination of a possible route to its overproduction or elimination.

Acknowledgments

We acknowledge the support of CNPq and FAPERGS. We thank the KEGG and PEC databases for providing public access to their data.

References

[1] Devos, D., Valencia, A.: Intrinsic Errors in Genome Annotation. Trends in Genetics. **17** (2001) 429–431. 142

[2] Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., Barabási, A.-L.: The Large-Scale Organization of Metabolic Networks. Nature **407** (2000) 651–654. 143, 146, 147

[3] Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N., Barabási, A.-L.: Hierarquical Organization of Modularity in Metabolic Networks. Science **297** (2002) 1551–1555. 143

[4] Karp, P. D., Krummenacker, M., Paley, S., Wagg, J.: Integrated Pathway-Genome Databases and Their Role in Drug Discovery. Trends in Biotechnology **17** (1999) 275–281. 143
[5] Jeong, H., Mason, S. P., Barabási, A.-L., Oltvai, Z. N.: Lethality and Centrality in Protein Networks. Nature **411** (2001) 41–42. 143, 146
[6] http://www.shigen.nig.ac.jp/ecoli/pec/ 143
[7] Chartrand, G.: Introductory Graph Theory. Dover Publications, New York (1977). 144
[8] Edwards, J. S., Ibarra, R. U., Palsson, B. O.: *In silico* Predictions of *Escherichia coli* Metabolic Capabilities are Consistent with Experimental Data. Nature Biotechnology **19** (2001) 125–130. (http://gcrg.ucsd.edu/downloads/index.html) 144
[9] Kanehisa, M., Goto, S.: KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Research **28** (2000) 27–30. (http://www.genome.ad.jp/kegg/kegg2.html) 144
[10] Overbeek, R. *et al.*: WIT: Integrated System for High-Throughput Genome Sequence Analysis and Metabolic Reconstruction. Nucleic Acids Research **28** (2000) 123–125. (http://ergo.integratedgenomics.com/ERGO/) 144
[11] Ouzounis, C. A., Karp, P. D.: Global Properties of the Metabolic Map of *Escherichia coli*. *Genome Research* **10** (2000) 568–576. (http://BioCyc.org/ecocyc/) 144
[12] Schroeder, E. K., de Souza, O. N., Santos, D. S., Blanchard, J. S., Basso, L. A.: Drugs that Inhibit Mycolic Acid Biosynthesis in *Mycobacterium tuberculosis*. *Current Pharmaceutical Biotechnology* **3** (2002) 197–225. 147
[13] Campbell, J. W., Cronan, J. E. Jr.: Bacterial Fatty Acid Biosynthesis: Targets for Antibacterial Drug Discovery. *Annu Rev. Microbiology* **55** (2001) 305–332. 147

Symbolic Model Checking of Biochemical Networks

Nathalie Chabrier and François Fages*

Projet Contraintes, INRIA-Rocquencourt,
BP105, 78153 Le Chesnay Cedex, France,
{Nathalie.Chabrier,Francois.Fages}@inria.fr

Abstract. Model checking is an automatic method for deciding if a circuit or a program, expressed as a concurrent transition system, satisfies a set of properties expressed in a temporal logic such as CTL. In this paper we argue that symbolic model checking is feasible in systems biology and that it shows some advantages over simulation for querying and validating formal models of biological processes. We report our experiments on using the symbolic model checker NuSMV and the constraint-based model checker DMC, for the modeling and querying of two biological processes: a qualitative model of the mammalian cell cycle control after Kohn's diagrams, and a quantitative model of gene expression regulation.

1 Introduction

In recent years, Biology has clearly engaged an elucidation work of high-level biological processes in terms of their biochemical basis at the molecular level. The mass production of post genomic data, such as ARN expression, protein production and protein-protein interaction, raises the need of a strong parallel effort on the formal representation of biological *processes*. Metabolism networks, extracellular and intracellular signaling pathways, and gene expression regulation networks, are very complex dynamical systems. Annotating data bases with qualitative and quantitative information about the dynamics of biological systems, will not be sufficient to integrate and efficiently use the current knowledge about these systems. The design of formal tools for *modeling* biomolecular processes and for *reasoning* about their dynamics seems to be a mandatory research path to which the field of formal verification in computer science may contribute a lot.

Several formalisms have been proposed in recent years for the modeling of biochemical networks. Regev and Shapiro [22] were the first to propose the use of a formal concurrent language, namely Milner's π-calculus, for the modeling of a biochemical processes such as the RTK/MAPK pathway. The bio-calculus of [21] introduces a more biology-oriented syntax for a similar calculus. More

* This work has been done in the framework of the INRIA Cooperative Research Action "Process Calculi and Biology of Molecular Networks", ARC CPBIO, http://contraintes.inria.fr/cpbio

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 149–162, 2003.

recently, quantitative modelings of biochemical processes have been developed with hybrid Petri nets [19, 16], hybrid concurrent constraint languages [4], and hybrid automata [1, 14].

In this paper we propose to go beyond simulation and to focus on the issue of providing automated methods for *querying and validating formal models* in systems biology. More specifically, we propose,

- first, the use of the temporal logics CTL as a query language for models of biological processes,
- second, the use of concurrent transition systems for the modeling of biological processes,
- and third, the use of symbolic model checking techniques for automatically evaluating CTL queries in both qualitative and quantitative models.

Our approach will be illustrated by two examples: a qualitative model of the mammalian cell cycle control after Kohn's diagrams [6, 17], and a quantitative model of gene expression.

1.1 Example 1: The Mammalian Cell Cycle Control

In this example, the main actors are genes, proteins with their phosphorylation sites, multimolecular complexes, and membranes. The molecules interact together to produce new proteins (synthesis), form multimolecular complexes (complexation), modify proteins (phosphorylation and dephosphorylation) degrade or transport molecules.

The cell cycle in eukaryotes is divided into four phases. Between two cell divisions, the cell is in a gap phase called G_1. The synthesis phase S starts with the replication of the nucleus. A second gap phase G_2 precedes the fourth phase: the mitose phase M during which the cell divides into two cells. The gap phase G_1 is mainly responsible for the duration of the cell cycle, it is in fact a growing phase of the cell and may contain a quiescent phase G_0 in which the cell can stay for long period of time or forever (stable state) without further division. Each phase is characterized by the activity of two major types of proteins: cyclins and cyclin-dependent kinases (Cdk). Cdk activity requires binding to a cyclin, and is controlled by specific inhibitors and by stimulatory or inhibitory phosphorylations by several kinases or phosphatases which in turn may produce positive feedback loops.

A state of the cell is defined by the values of the actors: either the presence or absence of molecules, or their number, or their concentration in each part of the cell, and by general data like the pH and the temperature. Note that a set of states can be just represented by partial information on the actual values of state variables, like for instance intervals or constraints between variables.

The biological queries one can consider about the cell cycle control are of different kinds:

About Reachability:

1 Given an initial state *init*, is there a pathway for synthesizing a protein P?
2 Which are the initial states from which another set of proteins S can be produced?

About Pathways:

3 Can the cell reach a state s while passing by another state s_2?
4 Is state s_2 a necessary checkpoint for reaching state s?
5 Can the cell reach a state s without violating certain constraints c?
6 From an initial state *init*, is it possible to synthesize a protein P without creating nor using protein Q ?

About Stable States:

7 Is a certain (partially described) state s of the cell a stable state ?
8 Can the cell reach a given stable state s from the initial state *init*?
9 Must the cell reach a given stable state s from the initial state *init*?
10 What are the stable states?

About Durations:

11 How long does it take for a molecule to become activated?
12 In a given time, how many Cyclins A can be accumulated?
13 What is the duration of a given cell cycle's phase?

About the Correctness of the Model:

14 *Can one see the inaccuracies of the model, and correct them?*

1.2 Example 2: Regulation of Gene Expression

As noted in [24, 9], the dynamics of gene regulatory networks can be modeled by a system of differential equations of the form

$$\dot{x}_i = f_i(\boldsymbol{x}) - g_i(\boldsymbol{x}) * x_i,\ x_i \geq 0,\ 1 \leq i \leq n,$$

where $\boldsymbol{x}$ is a vector of exogenous variables and cellular concentrations of gene products (proteins and mRNAs), $g_i(\boldsymbol{x})$ is the rate of degradation of protein x_i, and f_i is a highly non-linear function which expresses the effect of the other variables on the synthesis of x_i. Exogenous variables are defined by setting $\dot{x}_i = 0$. The other variables may participate to complex positive or negative feedback loops.

We are interested in answering the following types of queries:

About Activation:

15 Can protein x reach a concentration greater than a given value τ?
16 Which states may produce a concentration for x greater than some value τ?

About Invariants:

17 Is a given relationship c between concentrations always satisfied?

1.3 Plan of the Paper

The next section presents the temporal logic CTL that we propose to use as a query language for biochemical systems. This approach is illustrated by the formalization of the biological queries given above in the two case studies of this paper. Some limits of CTL are discussed w.r.t. biological queries which do not translate directly in CTL.

In Section 3, we focus on the simple formalism of concurrent transition systems for the rule-based modeling of biochemical networks. This is illustrated with a transition system over boolean variables for the cyclin box of the mammalian cell cycle control, and with a transition system over real numbers and linear constraints for a simple example of gene interaction.

Section 4 presents the basic model checking algorithm and the symbolic and constraint-based variants of model checking used in our experiments for querying our biological models in CTL. This section provides some performance figures that show the feasibility of the approach.

Section 5 provides some extra information and references to related work. The last section presents our conclusion.

2 The Temporal Logic CTL as a Query Language for Biochemical Models

2.1 Preliminaries on CTL

The Computation Tree Logic CTL is a logic for describing properties of computation trees and (non-deterministic) transition systems [8]. CTL is a temporal logic which abstracts from duration values and describes the occurrence of events in the two dimensions of the system: time and non-determinism. CTL basically extends either *propositional* or *first-order* (FO) logic [13], with two path quantifiers for non-determinism: A, meaning "for all transition paths", and E, meaning "for some transition path", and with several temporal operators: X meaning "next time", F meaning "eventually in the future", G meaning "always", U meaning "until".

A "safety" property, specifying that some situation described by a formula ϕ can never happen, is expressed by the CTL formula $AG\neg\phi$, i.e. on all paths ϕ is always false. A "liveness" property, specifying that something good ψ will eventually happen, is expressed by the formula $AF\psi$. Note that by duality we have $EF\phi = \neg AG\phi$ and $EG\phi = \neg AF\neg\phi$ for any formula ϕ.

Formally, CTL formulas are divided into state formulas and path formulas. Let AP be a set of atomic propositions, describing states. A *state formula* is either an atomic proposition, or a path formula prefixed by a path quantifier, or a logical combination of such formulas. The set of *path formulas* is the closure of the set of state formula by the temporal operators and logical connectives. Arbitrary state and path formulas form CTL* formula. CTL logic is a syntactic fragment of CTL* in which the temporal operators must be immediately prefixed by a path quantifier. For example, $A(FG\phi)$ and $E(F\phi \wedge G\psi)$ are CTL* formula

which are not expressible in CTL. In this paper we shall only be concerned with the fragment of CTL formulas.

The semantics of CTLis given by Kripke structures. A *Kripke structure* K is a triple (S, R, L) where S is a set of states, $R \subseteq S \times S$ is a (transition) relation and $L : S \to 2^{AP}$ is a function that associates to each state the set of atomic propositions true in that state. A path in K from a state s_0 is an infinite sequence of states $\pi = s_0, s_1, ...$ such that $(s_i, s_{i+1}) \in R$ for all $i \geq 0$. We denote by π^i the suffix of π starting at s_i. Now the inductive definition of the truth relation stating that a CTL formula ϕ is true in K at state s, noted $K, s \models \phi$, or true in K along path π, noted $K, \pi \models \phi$, is the following (the standard rules for logical connectives are omitted):

- $K, s \models \phi$ iff $s \models \phi$, if ϕ is a state formula,
- $K, s \models E\phi$ iff there is a path π from s such that $K, \pi \models \phi$,
- $K, s \models A\phi$ iff for every path π from s, $K, \pi \models \phi$,
- $K, \pi \models \phi$ iff $s \models \phi$ where s is the starting state of π, if ϕ is a state formula,
- $K, \pi \models X\phi$ iff $K, \pi^1 \models \phi$,
- $K, \pi \models F\phi$ iff there exists $k \geq 0$ such that $K, \pi^k \models \phi$,
- $K, \pi \models G\phi$ iff for every $k \geq 0$, $K, \pi^k \models \phi$,
- $K, \pi \models \phi U \psi$ iff there exists $k \geq 0$ such that $K, \pi^k \models \psi$ and $K, \pi^j \models \phi$ for all $0 \leq j < k$.

Following [13], assuming a Kripke strucutre K, we shall identify a CTL formula ϕ to the set of states which satisfy it, i.e. $\{s \in S | K, s \models \phi\}$. Thus, by abuse of notation, we will write $s \in \phi$ if ϕ is true in state s in K.

2.2 Example 1: The Mammalian Cell Cycle Control

The examples of biological questions listed in the previous section translate into CTL as follows.

About Reachability:

1 $init \in EF(P)$,
2 $EF(S)$, the CTL formula is indeed a representation of all states satisfying it, model checking tools provide facilities for enumerating explicitly these states.

About Pathways:

3 $EF(s_2 \wedge EFs)$,
4 $\neg E((\neg s_2)\ U\ s)$,
5 $E(c\ U\ s)$,
6 $init \in E(\neg Q\ U\ P)$,

About Stable States:

7 $s \in AG(s)$, a stable state in the strong sense is a state in which the cell stays indefinitely with no possibility of escaping; a state in which the cell can stay indefinitely but can escape from, can be modeled by $s \in EG(s)$,

8 $init \in EF(AGs)$,
9 $init \in AF(AGs)$.
10 The set of stable states of the system cannot be represented by a CTL query. In CTL, it is only possible to check whether a given (partially described) state is a stable state. One approach to computing the set of stable states (or checkpoints, etc.) of a biochemical network would be to combine model checking methods with search methods, that is an interesting open problem.

About Durations:

Time in temporal logic CTL is a purely qualitative notion, based on a single precedence relation. Reasoning about durations is thus not expressible with the temporal operators of CTL. Nevertheless, if the state description logic underlying CTL is not propositional but first-order, it is always possible in FO to model time intervals by adding to all atomic propositions extra numerical arguments representing their starting time and duration. Constraint-based model checking presented in section 4.1 provides an automatic method for evaluating such queries.

About the Correctness of the Model:

When an intended property is not verified, the pathways leading to a counterexample help the user to refine the model. Similarly, when an unintended property is satisfied, the pathway leading to a witness helps the user to refine his model by enforcing extra conditions in rules, or, if the property is not known to be biologically true or false, the witness may suggest to do biological experiments in order to validate or invalidate that property of the model. In biology, the standard loop between modeling and model-validation is in fact a three fold loop between modeling, querying the model and doing biological experiments.

2.3 Example 2: Regulation of Gene Expression

The second series of questions for the example of gene regulation can be translated in CTL as follows. It is worth noting that in this second series of examples, the setting of first-order logic is useful to express the constraints in CTL queries [10, 13].

About Activation:

15 $init \in EF(x > \tau)$,
16 $EF(x > \tau)$,

About Invariants:

17 $init \in AG(c)$.

The same remark as above about the tools for correcting the model or suggesting biological experiments, applies as well.

3 Modeling Biochemical Networks with Concurrent Transition Systems

Concurrent transition systems have been introduced in the scheme of [23] for reasoning about concurrent programs. Concurrent transition systems offer a direct way of specifying a Kripke structure by reaction rules and we shall use them for this reason for the modeling of biochemicals networks.

A *concurent transition system* is a Kripke structure presented as a triple $(\boldsymbol{x}, I, R)$ where $\boldsymbol{x}$ is a tuple of (data and control) variables, I is a formula on $\boldsymbol{x}$ expressing the initial condition as a set of values for all variables, and R is a set of condition-action rules. The rules have the following syntax:

$$\textit{condition } \phi(\boldsymbol{x}) \quad \textit{action } \boldsymbol{x}' = \rho(\boldsymbol{x})$$

where $\phi(\boldsymbol{x})$ denotes the condition under which the rule can be applied, and the primed version of the variables denotes the new values $\rho(\boldsymbol{x})$ of the variables after the rule is applied. By convention, the variables which are not modified in the right hand side of the rule keep their value unchanged.

Clearly, a concurrent transition system defines a Kripke structure, where the set of states is the set of all tuples of values for the variables, the initial state is the tuple of values satisfying the initial condition, and the transition relation is the union[1] (i.e. disjunction) of the relations between the states of all instances of the condition-action rules.

In the following, we shall associate a data variable to each molecule (protein or gene). The value of a variable will be either a numerical value expressing the concentration of the molecule, or a boolean value expressing simply the presence or absence of the molecule. The temporal evolution of the system will be modeled by the transition steps. The different transition paths will model the non-deterministic behavior of the system.

3.1 Example 1: The Mammalian Cell Cycle Control

In [17], Kohn provided an annotated diagrammatic representation of the molecular interaction map of the mammalian cell cycle control and DNA repair systems. The part concerning the cell cycle control, and with more details the Cyclin box and the E2F box, have been modeled in the ARC CPBIO by M. Chiaverini and V. Danos [6], as a set of 732 reaction rules over 165 molecules, and 532 variables taking into account the different forms of a molecule. The beauty of this model is that each rule is an instance of one of the following five rule schemas:

1. Complexation : $A \wedge B \rightarrow AB$,
   two molecules A and B bind together to form a multimolecular complex AB;

[1] Concurrent transition systems are asynchronous in the sense that one rule is executed at a time (interleaved semantics), hence the transition relation is the union of the relations associated to the rules. On the other hand, synchronous programs, that are not considered in this paper, have their transition relation defined by intersection.

2. Phosphorylation : $A \wedge B \rightarrow Ap \wedge B$,
   molecule A is modified under the action of a catalyst B, A is transformed in a phosphorylated form Ap,
3. Dephosphorylation : $Ap \wedge B \rightarrow A \ \wedge B$;
   the phosphorylated molecule Ap is dephosphorylated by catalyst B;
4. Synthesis : $A \rightarrow A \ \wedge B$,
   molecule B is synthesized by the gene with the activated promotor A;
5. Degradation : $A \wedge B \rightarrow A$,
   molecule B is degraded by molecule A.

In this model, each type of molecule is modeled by a variable. By lack of quantitative data, the variables associated to molecules are all boolean. They simply express the presence or absence of the molecule in the cell. This model of the mammalian cell cycle control is thus a purely logical model. For example, CycH, Cdk7 and CycH-Cdk7 are three variables representing respectively, Cyclin H, Cdk 7 and the dimer Cyclin H-Cdk 7. In the complexation rule schema, AB stands for a propositional variable denoting the multimolecular complex which results from the binding of the molecules denoted by A and B. An instance of this schema is the rule CycH $\wedge$ Cdk7 $\rightarrow$ CycH-Cdk7. Trimers, tetramers and more generally polymers can be formed by applying the complexation schema to dimers, etc. Note that it is possible to distinguish between the complexes (AB)C and A(BC) if there are biological reasons to make this distinction.

Similarly, one can introduce a variable named Cdk1(phosphorylated at Thr14)-cyclinB, to represent a phosphorylated form of the dimer Cdk1-Cyclin B at site Thr 14 of Cdk 1. The phosphorylation of this dimer by Mytosine 1 is modeled by the rule instance: Cdk1-CycB $\wedge$ Myt1 $\rightarrow$ Cdk1(phosphorylated at Thr14)-CycB $\wedge$ Myt1. An instance of the dephosphorylation rule is: Cdk1(phosphorylated at Thr14 and Tyr15)-CycB $\wedge$ Cdc25C(phosphorylated in N-terminal domain) $\rightarrow$ Cdk1-cyclinB $\wedge$ Cdc25C(phosphorylated in N-terminal domain).

For the sake of conciseness, we have used the following convention in the rule schemas for denoting condition-action rules. The left hand side of a rule is just its condition. The right hand side is a formula which expresses which variables are made true in the action, with the convention that the variables which do not appear in the schema remain unchanged, and the variables which appear in the left hand side and not in the right hand side of the schema may take *arbitrary values*. The rule schema of complexation is thus a short-hand for the four condition-action rules:

$condition\ A \wedge B \quad action\ AB' = true, A' = true, B' = true$
$condition\ A \wedge B \quad action\ AB' = true, A' = false, B' = true$
$condition\ A \wedge B \quad action\ AB' = true, A' = true, B' = false$
$condition\ A \wedge B \quad action\ AB' = true, A' = false, B' = false$

The condition-action rules make explicit the possible disappearance of molecules A and B by complexation.

3.2 Example 2: Regulation of Gene Expression

Following Euler's method for solving differential equations numerically, one can associate a discrete, yet infinite state, transition system to a system of differential equations.

We shall use the following pedagogical example of interaction between two genes taken from [4]:

$\dot{y} = 0.01 * x$,
$\dot{x} = 0.01 - 0.02 * x$ if $y < 0.08$,
$\dot{x} = -0.02 * x$ if $y \geq 0.08$.

Gene x activates gene y, but above a certain threshold, gene y inhibits expression of gene x.

A discretization by one time unit dt (e.g. $dt = 1$) leads to the following simple transition system:

condition $y < 0.8$ *action* $x' = x + (0.01 - 0.02 * x) * dt,\ y' = y + 0.01 * x * dt$
condition $y \geq 0.8$ *action* $x' = x - 0.02 * x * dt,\ y' = y + 0.01 * x * dt$

The transition system in this example is deterministic but it is worth noting that this is not required by the scheme. Note that the derivatives can be added to the states of the system in order to reason or express queries about them. Dynamic discretizations are possible by adding the time step dt as a state variable, similarly to multirate simulation in hybrid systems.

4 Model Checking for Systems Biology

4.1 Preliminaries on Model Checking

Model checking is an algorithm for computing, in a given Kripke structure K, the set of states which satisfy a given CTL formula ϕ, i.e. the set $\{s \in S | K, s \models \phi\}$. For the sake of simplicity, we consider only the CTL fragment of CTL, and use the fact that (by duality) any CTL formula can be expressed in terms of $\neg$, $\vee$, EX, EU and EG.

When K has a finite set of states, the model checking algorithm, in its simplest form, works with an explicit representation of K as a transition graph, and labels each state with the set of subformula of ϕ which are true in that state. First, the states are labeled with the atomic propositions of ϕ which are true in those states. The labeling of more complex formula is done iteratively, following the syntax of the subformula of ϕ. Formulas of the form $\neg\phi$ label those states which are not labeled by ϕ. Formulas of the form $\phi \vee \psi$ are added to the labels of the states labeled by ϕ or ψ. Formulas $EX\phi$ are added to the labels of the predecessor states of the states labeled by ϕ. Formulas $E(\phi U \psi)$ are added to the predecessor states of ψ while they satisfy ϕ. Formulas $EG\phi$ involve the computation of the strongly connected components of the subgraph of transitions restricted to the states satisfying ϕ. The states labeled by $EG\phi$ are the states in this subgraph for which there exists a path leading to a state in a non trivial strongly connected component. The complexity of this algorithm

is $O(|\phi| * (|S| + |R|))$ where $|\phi|$ is the size of the formula, $|S|$ is the number of states, and $|R|$ is the number transitions [8].

Symbolic model checking is a more efficient algorithm that uses a symbolic representation of finite Kripke structures with boolean formulas. In particular, the whole transition relation is encoded as a single (disjunctive) boolean formula, sets of states are encoded by boolean formulas, and ordered binary decision diagrams (OBDDs) are used as canonical forms for the boolean formulas. The symbolic model checking algorithm computes an OBDD representing the set of states satisfying a given CTL formula. The computation involves the iterative computation of the least fixed point (for EF) and the greatest fixed point (for EG) of simple predicate transformers associated to the temporal connectives [8]. In our experiments reported below, we used the state-of-the-art symbolic model checker NuSMV [7].

Constraint-based model checking applies to infinite state Kripke structures, such as Kripke structures with variables ranging over unbounded or continuous numerical domains. A constrained state is a finite representation using constraints, of a finite or infinite set of states. In the scheme of Delzanno and Podelski [10], infinite state Kripke structures are represented by constraint logic programs, and the CTL formulas, that are based on a fragment of first-order logic, are identified to the least fixed point and greatest fixed point of such programs. In our experiments reported below about the quantitative model of gene expression regulation, we used the implementation in Sicstus Prolog with constraints over finite domains and real numbers (simplex algorithm) of the model checker DMC [11].

4.2 Symbolic Model Checking of Logical Models

The Table 1 provides some performance figures about the evaluation of CTL queries in the mammalian cell cycle control model. The two first columns indicate the query and its type . The third column indicates the length of the pathway leading to a counterexample or to a witness. The fourth column indicates the

Table 1. Evaluation of CTL queries in the mammalian cell cycle control model with DMC and NuSMV

Type	Query	Pathway length	Number of DMC states	DMC time in seconds	NuSMV time in seconds
	compiling	-	-	-	47.5
2	EF(cycE)	6	279	1320	16.5
2	EF(SL1_1)	10	2107	29970	57.8
2	EF(cycA)	6	1072	23161	16.8
2	EF(PCNA)	6	245	2524	23.7
4	¬E((¬(Cdc25-active)) U Cdk1-CycB-active)	-	-	-	112

number of state expressions computed by DMC. The two last columns indicate the CPU time in seconds measured on a Pentium 4 at 660 Mhz for DMC and NuSMV. The NuSMV timings (which include the reconstruction of a pathway) show the efficiency of the OBDD representation of states compared to the simple representation of states in Prolog (without boolean constraints) used in DMC.

4.3 Constraint-Based Model Checking of Quantitative Models

Constraint-based model checking of quantitative models must be contrasted with symbolic model checking techniques which use finite domain abstraction techniques to deal with quantitative models [25]. The constraint-based model checker DMC [11] performs a backward reachability analysis, starting from a constrained state expressing the CTL property to prove, up to the computation of a state expression containing the initial state. Safety (resp. liveness) properties involve the computation of the least (resp. greatest) fixpoint of a constraint logic program.

It is worth noting that this kind of reasoning mixes symbolic computation on set of states described by *numerical constraints*, with a form of reasoning by induction. In the simple example of gene regulation, the query $EF(x \geq 0.5)$ immediately evaluates to false, as any predecessor state of a state described by the constraint $x \geq 0.5$ again satisfies that constraint and is thus subsumed.

The table below show some performance figures in the Prolog implementation of DMC. Better performance results could be obtained by using other discretizations of the problem, other translations involving Runge-Kutta method, and other implementations of the constraints.

Table 2. Examples of CTL queries in the example of gene interaction

Type	Request	Pathway length	Number of states	Computing time (sec)
16	EF(x≥0.5)	0	1	0.02
16	EF(x≥0.2)	28	29	3.65
16	EF(x≥0.45)	116	117	59
16	EF(y≥0.8)	212	213	256
16	EF(x+y≥1.3)	178	179	173
16	EF(x+y≥1.2)	194	195	206
17	AG(x>0.5)	1	2	0.03
17	AG(x<0.1)	14	127	8
17	AG(y<0.8)	211	421	7

5 Discussion and Related Work

The experiments reported in this paper should be read as a proof-of-concept rather than as providing an already usable accurate modeling of biological systems. Some errors or ambiguities were corrected with the biologists of the ARC

CPBIO, but more work with biologists is needed to validate further the formal modeling of Kohn's diagram as a concurrent transition system, and incorporate more knowledge in the model.

The pathway logic of S. Eker et al. [12] is tightly related to our approach. The modeling of biochemical networks with concurrent transition systems is of a somewhat lower level than with pathway logic. Pathway logic is indeed more expressive as it can express algebraic properties of the components, such as the commutativity and associativity of complexation. This capability can be used to infer the possible reactions of molecules from their logical structure. It is worth considering however, that the interaction capabilities of a protein are often not related to the ones of its components, as they depend on the 3D structure of the proteins which is obviously impractical to take into account in a global modeling [3].

One limitation to the modeling of biological systems with concurrent transition systems is the necessity to encode all the parameters of the system in a finite vector of data and control variables. In order to get rid of this difficulty, we are currently investigating the use of other model checkers based on linear logic or multiset rewriting, that dont't have this restriction and make it possible to reason about systems within an arbitrary context [5].

Another obvious limitation in the experiments reported here is the absence of stochastic data. There are however stochastic model checking methods [18, 20] which can be investigated to circumvent this limitation.

The performances of our model checker can be improved in many ways as we have already shown with the use of DMC and NuSMV for the logical model of the mammalian cell cycle control. Similar improvements will be necessary to show the scalability of this approach for quantitative models. In this respect, constraint-based model checking is thightly related to hybrid systems methods and to hybrid verification tools such as for example Hytech [15] or d/dt [2]. Approximation techniques coming from hybrid automata can be imported in constraint-based model checking with the framework of abstract interpretation. On the other hand, constraint-based model checking provides a method for generalizing hybrid verification tools, going from pure reachability analysis towards more general CTL query evaluation.

6 Conclusion

We have applied symbolic model checking techniques to the querying and validation of both quantitative and qualitative models of biomolecular systems. Our first experiments show some advantages over simulation. Constraint reasoning makes it possible to group large or infinite sets of states into small constrained state expressions which provide formal proofs of reachability, pathway, checkpoint and stability properties. In some cases the properties can be checked by computing a fewer number of states than by simulation. It is also possible to reason with infinite sets of initial states finitely represented by constraints. Moreover

the proof method applies to non-deterministic systems, for which simulations may be unfeasible.

We have also shown that constraint-based model checking can be applied in quantitative models described by differential equations. In our experiments we used a symbolic variant of Euler's method, but the use of the more accurate Runge-Kutta method, of non-linear constraint solving by interval propagation, and the use of abstraction techniques open many ways for improvement.

For all these reasons, we believe that, beyond simulation, verification tools such as model checking will become indispensable for querying and validating complex models in systems biology.

Acknowledgement

We gratefully acknowledge the interactions we had with our colleagues of the ARC CPBIO, especially with Magali Roux-Rouquié, Julien Renner and Grégory Sautejeau from Institut Pasteur, for interesting discussions on Systems Biology and relevant bits of Biomolecular Biology, Vincent Danos and Marc Chiaverini from CNRS PPS Lab. for their beautiful transcription of Kohn's diagrams in their core modeling language, Vincent Schächter at Genoscope Evry, for his insights on the validation of biological models, Alexander Bockmayr, Arnaud Courtois and Damien Eveillard from the ModBio group at LORIA Nancy, for fruitful discussions on quantitative models.

References

[1] R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. J. Pappas, H. Rubin, and J. Schug. Hybrid modeling and simulation of biomolecular networks. In Springer, editor, *Hybrid Systems: Computation and Control*, LNCS 2034, pages 19–32, Rome, Italy, 2001. 150

[2] E. Asarin, T. Dang, and O. Maler. d/dt: A verification tool for hybrid systems. In *Invited session "New Developments inVerification Tools for Hybrid Systems", in Proceedings of the Conference on Decision and Control*, Florida, USA, July 2001. 160

[3] R. Backofen, S. Will, and E. Bornberg-Bauer. Application of constraint programming techniques for structure prediction of lattice proteins with extended alphabets. *Bioinformatics*, 3(15):234–242, 1999. 160

[4] A. Bockmayr and A. Courtois. Using hybrid concurrent constraint programming to model dynamic biological systems. In Springer, editor, *18th International Conference on Logic Programming*, pages 85–99, Copenhagen, 2002. 150, 157

[5] M. Bozzano, G. Delzanno, and M. Martelli. Model checking linear logic specifications. Technical report, Technical report,University di Genova, March 2002. 160

[6] M. Chiaverini and V. Danos. A core modeling language for the working molecular biologist. Technical report, CNRS, PPS, Paris 7, November 2002. 150, 155

[7] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella. Nusmv 2: An opensource tool for symbolic model checking. In *In Proceeding of International Conference on Computer-Aided Verification, CAV'2002*, Copenhagen, Danmark, July 2002. 158

[8] E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999. 152, 158
[9] H. de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9(1):69–105, 2001. 151
[10] G. Delzanno and A. Podelski. Model checking in clp. In *Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems TACAS'99*, volume 1579 of *LNCS*, pages 223–239. Springer-Verlag, January 1999. 154, 158
[11] G. Delzanno and A. Podelski. *DMC user guide*, 2000. 158, 159
[12] S. Eker, M. Knapp, K. Laderoute, P. Lincoln, J. Meseguer, and K. Sonmez. Pathway logic: Symbolic analysis of biological signaling. In *the Pacific Symposium on Biocomputing*, pages 400–412, January 2002. 160
[13] E.A. Emerson. *Temporal and Modal Logic*, pages 995–1072. J. van Leeuwen Ed., North-Holland Pub. Co./MIT Press, 1990. 152, 153, 154
[14] R. Ghosh and C. Tomlin. Lateral inhibition through delta-notch signaling: A piecewise affine hybrid model. In Springer, editor, *Hybrid Systems: Computation and Control*, LNCS 2034, pages 232–246, Rome, Italy, 2001. 150
[15] T. Henzinger, J. Preusig, and H.Wong-Toi. Some lessons from the hytech experience. In *Proceedings of the 40th Annual IEEE Conference on Decision and Control, CDC'2001*, 2001. 160
[16] R. Hofestädt and S. Thelen. Quantitative modeling of biochemical networks. In *In Silico Biology*, volume 1, pages 39–53. 1998. 150
[17] K.W. Kohn. Molecular interaction map of the mammalian cell cycle control and dna repair systems. *Molecular Biology of Cell*, 10(8):703–2734, August 1999. 150, 155
[18] S. Laplante, R. Lassaigne, and F. Magniez. Probabilistic model checking: an approach based on property testing. In *Proc. of the 7th annual IEEE symposium on Logic in Computer Science LICS'02*, Copenhagen, 2002. 160
[19] H. Matsuno, A. Doi, M. Nagasaki, and S. Miyano. Hybrid petri net representation of gene regulatory network. In *Pacific Symposium on Biocomputing (5)*, pages 338–349, 2000. 150
[20] D. Monniaux. *The analysis of probabilistic programs by abstract interpretation*. PhD thesis, Ecole Normale Supérieure, Paris, France, 2001. 160
[21] M. Nagasaki, S. Onami, S. Miyano, and H. Kitano. Bio-calculus: Its concept, and an application for molecular interaction. In *Currents in Computational Molecular Biology.*, volume 30 of *Frontiers Science Series*. 2000. 149
[22] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. In *Proceedings of the Pacific Symposium of Biocomputing*, pages 6:459–470, 2001. 149
[23] U. A. Shankar. An introduction to assertionnal reasoning for concurrent systems. *ACM Computing Surveys*, 3(25):225–262, 1993. 155
[24] R. Thomas and d'Ari. *Biological feedback*. CRC press, 1990. 151
[25] A. Tiwari and P. Lincoln. Automated technique for stability analysis of delta-notch lateral inhibition mechanism. Technical report, SRI, Stanford USA, 2002. 159

Coupled Oscillator Models for a Set of Communicating Cells

Will Casey

Courant Institute Mathematical Sciences, NYU Bioinformatics lab
715 Broadway 10th floor, New York, NY 10003
wcasey@cims.nyu.edu

Abstract. We investigate how the structure of cell to cell interaction may affect the periodic behavior of a system of coupled oscillatory gene networks. Using the cellular circadian clock model of Goldbeter '95 as a cellular model, we propose a general model for coupling these cellular models via transfer of substrate from cell to cell.
We proposes a system of coupled oscillators whose dynamics are determined by multiple independent cellular models and a network of cell–cell communication links acting as perturbation forces on neighboring cell states. We will assume that no production or consumption occurs other than through the mechanisms of each individual cell; hence our model is a closed system without external stimuli. We are interested in the how the cell–cell communication network may contribute to the stability of the composite system. Under the assumption of conserved concentration levels found in the system, the nature of cell–cell interactions may be summarized by geometric properties of stochastic matrices.
The assumption of conservation of substrate allows us to describe an affine condition on the coupling model; a subclass of the affine condition is connected to stochastic matrices and finite state Markov Chains and suggests how the *transient* states of the Markov Chain act as pumps and work to entrain the system.
In the paper we demonstrate the diversity of dynamics which the general model for coupled oscillator system is capable.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 163, 2003.

Representing and Simulating Protein Functional Domains in Signal Transduction Using Maude

Steven Eker[1], Keith Laderoute[1], Patrick Lincoln[1],
M.G. Sriram[2⋆], and Carolyn Talcott[1]

[1] SRI International
333 Ravenswood Ave, Menlo Park, CA 94025
{Steven.Eker,Keith.Laderoute,Patrick.Lincoln,Carolyn.Talcott}@sri.com
www.sri.com
[2] Medical Informatics and Health Care Systems, Johnson Space Center, NASA

We describe the application of Maude, a symbolic language founded on rewriting logic, to the modeling of functional domains within signaling proteins, within the framework of the Pathway Logic project. Protein functional domains (PFDs) are a critical focus of modern signal transduction research. Recently, it has become apparent that signaling proteins must behave combinatorially to generate the high levels of complexity and versatility observed in signaling networks, and that PFDs are responsible for at least part of this capability. In our approach, the state of a system consisting of proteins and PFDs is represented as a term in an equational theory and rewrite rules.

Our executable model includes representations and algorithms for the protein structures which contribute to the formation of biological signaling complexes. The functions developed in the model facilitate the representation of relationships between and among the molecular entities, i.e., the states of the system and its dynamic evolution from state to state. An important benefit of this approach is that it enables the development of models over a spectrum of granularity. A fine level of granularity specifies interactions and changes of system state with reference to specifically named proteins, PFDs and other conditions. A more abstract level describes interactions between classes of PFDs, and expresses less constrained rules for the formation of signal complexes. This capability of Maude of encoding arbitrary levels of generality could be valuable for organizing or inferring potential protein-protein interactions within large networks of signaling proteins as they are constructed from experimental results.

To illustrate our approach we focus on the engagement of TNF-R1 by its ligand to create a death induced signaling complex (DISC). Maude models were developed at two levels of granularity. The finer level model produced signal complexes in the TNF-R1 pathway(s). Another model containing rules which described interactions at a more abstract level concerning families of PFDs. Executing the latter produced the signal complexes present in the fine model as well as other interactions. Such results from the general model should be considered as hypotheses that are either valid based on existing empirical knowledge, should be analyzed further by including additional rules and constraints, or tested by experimentation. Developing symbolic models of signaling proteins containing

⋆ Research performed while at SRI International, mgsriram@medal.org

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 164–165, 2003.

functional domains is important because of the potential to generate analyses of complex signaling networks based on structure-function relationships.

For more information:
`http://www.csl.sri.com/projects/pathwaylogic.html`

A Core Modeling Language for the Working Molecular Biologist (Abstract)

Marc Chiaverini and Vincent Danos

Equipe PPS, University of Paris VII

We propose a simple and biologically legible formalism to represent protein-protein and protein-DNA interaction networks. This seems a useful preliminary step towards computer-aided exploration and engineering of such systems, though for the moment it is still unclear what kind of biologically relevant questions one would be prompted to ask if such tools were available.

We assume an infinite set of *protein names*, written $\mathcal{N}$, and ranged over by symbols such as A, B, ... and an *arity* function $\mathfrak{a}(.) : \mathcal{N} \longrightarrow \mathbb{N}$ from protein names to integers, mapping a protein name to an integer representing its number of *sites*. A *formal protein*, or simply a protein, is a couple $(A, \mathbf{x})$, written $A\,\langle\mathbf{x}\rangle$, where $A \in \mathcal{N}$ is a name and $x \in \{0,1\}^{\mathfrak{a}(A)}$ is a vector of booleans representing the occupancy state of A's sites, or simply the *state* of A. Proteins may be assembled into *protein complexes*, or simply *complexes* ranged over by C, D, ... and we write "." for composition which we assume to be associative and commutative.

Complexes regulate the rate of synthesis of proteins by binding to promoter elements (small strings of DNA upstream of genes) affecting thereby the synthesis of the protein(s) associated to the gene. To express this, we suppose that we have a map $\Box. : \mathcal{N} \longrightarrow \mathcal{P}$ associating to each $A \in \mathcal{N}$ a promoter element. The same promoter element can be associated to many different proteins. One could do with sets of promoter elements instead of just a single one, it poses no difficulty.

We consider four kinds of reactions:

$$\underset{\text{(ACTIVATION)}}{C\langle\mathbf{x}\rangle, D\langle\mathbf{y}\rangle \longrightarrow C\langle\mathbf{x}'\rangle, D\langle\mathbf{y}'\rangle} \qquad \underset{\text{(COMPLEXATION)}}{C\langle\mathbf{x}\rangle, D\langle\mathbf{y}\rangle \longrightarrow C\langle\mathbf{x}'\rangle{\cdot}D\langle\mathbf{y}'\rangle}$$

$$\underset{\text{(SYNTHESIS)}}{F\langle\mathbf{x}\rangle{\cdot}\Box\mathrm{A} \longrightarrow F\langle\mathbf{x}\rangle{\cdot}\Box\mathrm{A}, A\langle\mathbf{y}\rangle} \qquad \underset{\text{(DEGRADATION)}}{F\langle\mathbf{x}\rangle, C\langle\mathbf{y}\rangle \longrightarrow F\langle\mathbf{x}\rangle}$$

With this simple and purely descriptive language we were able to complete the formalization of Kohn's first molecular map of the cell cycle control [1], resulting in about 600 protein-protein and protein-DNA interactions involving about 70 different proteins and DNA binding sites. A few ambiguities in Kohn's description were resolved in the process.

References

[1] Kurt W. Kohn. Molecular interaction map of the mammalian cell cycle control and DNA repair systems. *Molecular Biology of the Cell*, n. 10:2703–2734, 1999. 166

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 166, 2003.

Integrating Simulation Packages via Systems Biology Mark-Up Language

Manuel Corpas

Department of Computer Science, University of Manchester
Oxford Road, Manchester M13 9PT, United Kingdom
mcorpas@cs.man.ac.uk

Abstract. Programs that simulate large-scale biochemical networks are diverse, each of them with their strengths and weaknesses. The typical *in-silico* scenario in Metabolomics consists of the combination of different packages, so that they provide a more holistic view of the simulated model. However, this has led to problems of incompatibilities, especially when trying to adapt results from one package into another. Until recently, the way to solve this problem was by hand, in a time consuming and error-prone process. Systems Biology Markup Language (SBML) has arisen as a standard platform for information exchange in the Systems Biology community. A parser was created to allow the automatic exchange of information via SBML. This parser was tested with GEPASI (a General Pathway Simulator). It was found that only 9% of all the parameters of the specification file produced by GEPASI could be successfully translated into the variables described by the SBML Level 1 specification. Further development of SBML Level 1 is suggested in order to produce a more complete representation of biochemical information.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 167, 2003.

Recreating Biopathway Databases towards Simulation

Masao Nagasaki[1], Atsushi Doi[2], Hiroshi Matsuno[2], and Satoru Miyano[1]

[1] Human Genome Center, Institute of Medical Science, University of Tokyo
4-6-1 Shirokane-dai, Minato-ku, Tokyo, 108-8639, Japan
{masao,miyano}@ims.u-tokyo.ac.jp
[2] Graduate School of Science and Engineering, Yamaguchi University
1677-1 Yoshida, Yamaguchi 753-8512, Japan
atsushi@ib.sci.yamaguchi-u.ac.jp
matsuno@sci.yamaguchi-u.ac.jp

Genomic Object Net (GON) is a biopathway modeling and simulation platform that employs the originally developed notion, hybrid functional Petri Net with extension (HFPNe) that extends the hybrid functional Petri Net (HFPN) [2, 3], and is developed with JAVA [1]. With this platform, we have succeeded in modeling and simulating glycolytic pathway of *E. coli*, boundary formation by notch signaling in *Drosophila*, and apoptosis induced by *Fas* ligand, etc [1]. For the modeling and simulation of a biopathway, suitable information selection from public biopathway databases, such as *Kyoto Encyclopedia of Genes and Genomes* (KEGG) and BioCyc, would be useful. Although the first aim for these pathway databases is to reorganize biochemical information for usage on computers and is not for modeling and simulation of biopathways. Thus, we have developed a way to transform these pathway databases so that the converted biopathways can run on GON. The transformation of the static biopathway models in KEGG and BioCyc leads users (biologists) to modify and refine the resulting dynamic biopathway models for their own interests on GON. This will be a first step for recreating biopathway databases towards simulation.

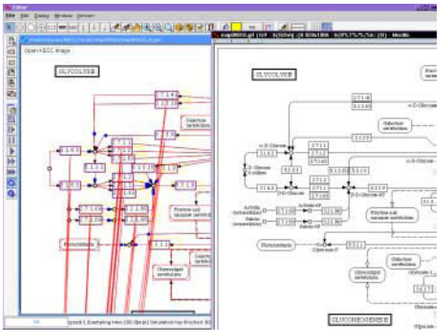


Fig. 1. Right window is a biopathway view on KEGG. Left window is a reconstructed and simulating view for the biopathway on GON

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 168–169, 2003.

References

[1] http://GenomicObject.Net/ 168
[2] Matsuno, H., Doi, A., Nagasaki, M. and Miyano, S.: Hybrid Petri net representation of gene regulatory network. *Proc. Pacific Symposium on Biocomputing 2000* (2000) 338–349 168
[3] Matsuno, H., Doi, A., Hirata, H. and Miyano, S.: XML documentation of biopathways and their simulations in Genomic Object Net, *Genome Informatics 12* (2001) 54–62 168

How to Synthesize an Optimized Genetic λ-Switching System? A System-Theoretic Approach Based on SQP

Kwang-Hyun Cho[1], Jong-Ho Cha[1], and Olaf Wolkenhauer[2]

[1] School of Electrical Engineering, University of Ulsan
Ulsan, 680-749, Korea
ckh@mail.ulsan.ac.kr
[2] Department of Biomolecular Sciences and
Department of Electrical Engineering & Electronics
Control Systems Centre UMIST, Manchester, M60 1QD, U.K.
olaf.wolkenhauer@umist.ac.uk

Abstract. The concept of gene regulation has been investigated by biologists for several decades and has led to the concept of gene regulatory circuits and biological switches. There have been several theoretical approaches to model biological switches. In this paper we present a control systems approach to synthetic genetic switches. In particular, we first analyze the dynamic model of the synthetic λ- switch to identify parameters which affect the switching performance. Based on the dynamic model, we formulate a performance measure from those parameters and then apply an optimization technique. More specifically, we employ sequential quadratic programming for this purpose and then optimize the performance of the synthetic λ- switch in view of the performance measure. The system-theoretic approach suggested here should be applicable to studies of other regulated cellular systems.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 170, 2003.

Simulation Sudy of the TNFα Mediated NF-κB Signaling Pathway

Kwang-Hyun Cho[1], Sung-Young Shin [1], Hyeon-Woo Lee [2], and Olaf Wolkenhauer[3]

[1] School of Electrical Engineering, University of Ulsan
Ulsan, 680-749, Korea
ckh@mail.ulsan.ac.kr
[2] Immunomodulation Research Centre, University of Ulsan
Ulsan, 680-749, Korea
[3] Dept. of Biomolecular Sciences and
Dept. of Electrical Engineering and Electronics
Control Systems Centre UMIST, Manchester, M60 1QD, U.K.
o.wolkenhauer@umist.ac.uk

Abstract. Tumor necrosis factor α (TNFα) is a potent pro-inflammatory cytokine that plays an important role in immunity and inflammation, in the control of cell proliferation, differentiation and apoptosis. This paper presents a simulation study of the TNFα mediated NF-κB signaling pathway based on a system-theoretic approach. Up to the present, there have been numerous approaches to analyze and model cellular dynamics. The most prominent one is utilizing nonlinear differential equations to establish a mathematical model based on reaction kinetics. This approach can provide us with mathematically well-founded and tractable interpretations regarding pathways, especially those best described by enzyme reactions. This paper introduces not only an intuitive graphical model but also a quantitative mathematical model for the TNFα mediated NF-κB signaling pathway. Throughout simulation study, we can qualitatively validate the developed mathematical model compared with experimental results along this pathway. Further investigations into the TNFα mediated NF-κB signaling pathway are in progress to include an inhibitor kinase functioning in this pathway.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 171, 2003.

Detection and Analysis of Unexpected State Components in Biological Systems

Anastasia Pagnoni[1] and Andrea Visconti[1]

Department of Computer Science – University of Milan, Italy
{Pagnoni,Visconti}@dsi.unimi.it

Abstract. This contribution presents a methodology aimed at detecting and analysing unexpected states of biological systems. We suggest using: (a) Petri nets in order to represent the causal structure of a biological system and its prescribed functional behaviour; and (b) algebraic coding theory (Hamming codes) to detect unexpected system states (mutations, unwanted situations, errors). Recent research aimed at modelling biological processes has successfully applied Petri nets to represent the causal structure and processes of biological systems in a way that makes the formal verification of the model easy. The formal language of Petri Nets is a powerful tool for representing biological systems together with their "regular" behaviour. However, in every living system sooner or later something will "go wrong" - a disease, or a mutation will occur - bringing the system into a state which our model didn't take into account, a state that we will call an "error state" for short. This paper is about introducing algebraic error-detection into Petri-net models. The basic idea is to turn reachable markings into "legal" words of a linear error-correcting code. This is achieved by adding some control places to the Petri net, so that: (a) its incidence matrix becomes the generatrix of a linear code, and (b) reachable markings can be characterised as solutions of the code's linear homogeneous system. "Illegal" markings - or, unexpected system states - may then be detected via linear algebra, without having to construct, and search, the net's (always quite complex) reachability graph. Using suitable linear codes - that is, codes for which fast error-correction algorithms are available - "mutant" components of "illegal" markings are instantaneously identified, and can be analysed with regard to other structural properties of the net, such as boundedness and liveness. This contribution considers the case of single errors (point mutations): at most one unexpected component per system state. Single errors are detected via Hamming codes, for which a very fast error detection algorithm is known. We show that the number of control places to be added to the Petri net decreases exponentially with the net's size. Coding theory offers a wide range of algorithms for the detection of terrors in transmission systems. Applying such algorithms to the detection of unexpected states of biological systems seems a very promising approach.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 172, 2003.

Model Validation of Biological Pathways Using Petri Nets - Demonstrated for Apoptosis

Monika Heiner[1], Ina Koch[2], Jürgen Will[1]

[1] Brandenburg University of Technology at Cottbus, Department of Computer Science, Postbox 10 13 44, 03013 Cottbus, Germany
{mh, will}@informatik.tu-cottbus.de
http://www.informatik.tu-cottbus.de/~wwwdssz
[2] Technical University of Applied Sciences Berlin, Dept. Bioinformatics, Seestrasse 64, 13347 Berlin-Wedding, Germany
ikoch@tfh-berlin.de
http://www.molgen.mpg.de/~koch_i

Abstract: Biological networks tend to be very dense and large - far beyond human skills. Therefore, a crucial point seems to be their concise and unambiguous representation to handle these highly integrated networks computationally in an efficient manner. Moreover, our knowledge about a particular pathway is generally widely spread over various separate data bases, using a quite large variety of different graphical schemes.

This talk demonstrates the first steps of a new integrating methodology to develop and analyse models of biological pathways in a systematic manner using well established Petri net technologies. Petri nets represent a modelling method, very well-known for its powerful combination of readability and analyzability. They provide a generic description principle, applicable on any abstraction level. At the same time, they have a sound formal semantics, allowing thorough model evaluation. Hence, Petri nets may be used for a concise formal representation, allowing an unifying view on knowledge coming from different sources.

The whole approach comprises step-wise modelling, animation, model validation as well as qualitative and quantitative analysis for behaviour prediction. In this presentation, the first phase is addressed - how to develop and validate a model.

The example used in that talk is devoted to apoptosis, the genetically programmed cell death. Apoptosis is an essential part of normal physiology for most metazoan species. Disturbances in the apoptotic process could lead to several diseases. The pathway of apoptosis includes highly complex mechanisms to control and execute programmed cell death. We have intentionally chosen that example to stress the fact that even incomplete and uncertain knowledge may be subject of our technology. Actually, we are convinced that step-wise incremental modelling accompanied by running repeated analyses are the only chance to get dependable larger models.

The results provide a mathematically unique and valid model, enabling the simulation of known properties as well as to check the model for self-consistency. The next steps are obvious: after being convinced of the model's integrity, we are ready to use the model for questions where the answers are not yet known.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 173, 2003.

An Overview of Data Models for the Analysis of Biochemical Pathways

Yves Deville[1], David Gilbert[2], Jacques van Helden[3], and Shoshana Wodak[3]

[1] Computing Science and Engineering Department, Université catholique de Louvain
Place Saint-Barbe 2, B-1348 Louvain-la-Neuve, Belgium
deville@info.ucl.ac.be
[2] Bioinformatics Research Centre, Department of Computing Science
University of Glasgow
17 Lilybank Gardens, Glasgow G12 8QQ, Scotland, UK
drg@brc.dcs.gla.ac.uk
[3] Unité de Conformation des Macromolécules Biologiques
Université Libre de Bruxelles
50 av. F.D. Roosevelt, B-1050 Bruxelles, Belgium
{jvanheld,shosh}@ucmb.ulb.ac.be

Abstract. Various forms of data models can be used for the analysis of biochemical pathways such as metabolic, regulatory, or signal transduction pathways. This paper overviews and classifies the different forms of data models found in the literature, and describes how these models have been used in the analysis of biochemical pathways.
The quantity of available information on biochemical pathways for different organisms is increasing very rapidly, and it has now become possible to perform detailed analyses of metabolic pathway structures for entire organisms. However, such analyses face difficulties due to the nature of the databases which are often heterogeneous, incomplete, or inconsistent. This makes pathway analysis a challenging problem in system biology and in bioinformatics.
In this overview, we concentrate on models of network structure, focusing on the analysis of existing information, collected from experiments and stored in databases. We overview and classify the different forms of data models found in the literature using a unified framework. We describe how these models have been used in the analysis of biochemical pathways. This enables us to underline the strengths and weaknesses of the different approaches, and at the same time highlights some relevant future research directions.

C. Priami (Ed.): CMSB 2003, LNCS 2602, p. 174, 2003.

Discrete Event Systems and Client-Server Model for Signaling Mechanisms

Gabriel Ciobanu[1] and Dorin Huzum[2]

[1] National University of Singapore, School of Computing
gabriel@comp.nus.edu.sg
[2] "A.I.Cuza" University of Iaşi, Romania
huzzy@mymail.ro

Abstract. A suitable mathematical framework for modeling and simulation will be beneficial for a better and deeper understanding of complex biological systems. Our work shows that a general framework of the mathematical theory of modeling and simulation, together with the network client-server model can lead to a novel software approach and give new insights in molecular networks. Based on the Discrete Event System formalism, in this paper we present a model of the molecular network during T cell activation and other cell responses. The approach is systemic; we do not investigate individual molecules, but describe their network behaviour. The dynamics of molecular networks is provided by a set of interactive processes and appropriate rules for various transition or communication steps. We prove that such a system based mainly on communication has the same expressive power as a Turing machine.

The Discrete Event System Specification (DEVS) formalism and its theoretical basis provides a number of important properties such as hierarchical, modular composition, universality and uniqueness that can support development of simulation models and environments. We describe the Dynamic Structure Discrete Event System Specification (DSDEVS) as a theory-based approach to dynamic structure modeling and simulation. In the context of this theory we have two model type: basic, respectively network models. A DSDEVS basic model is a DEVS model; the network model is a combination of DSDEVS basic models. A change of structure refers to adding or removing components, or component links modifications. The network structure can change when the executive performs an internal or external transition. Thus as all structure information is kept in the network executive state, any such change will be accompanied by executive transitions. Incorporating dynamic structure capabilities in DEVS environment expands the range of phenomena and systems that can be studied.

We give a short description of the basic DSDEVS model for molecular networks. Let A be a molecule type. For A we define a $DEVS$ which describes the evolution of all type A molecules (complexes), in all the possible states to appear in a molecular network. We denote by A_j the fact that A is in state j. Let $M_A = (X_A, S_A, Y_A, \delta_A^{int}, \delta_A^{ext}, \lambda_A, \tau_A)$ be the DEVS description for A. For A_j we define a set $R_{A,j}$ composed of the rules that contain the molecule A in

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 175–177, 2003.

Table 1. States and transition functions for molecule A, state j and reaction r

states $(S_{A,j,r})$	$\delta^{int}_{A,j}$	$\delta^{ext}_{A,j}$	$\tau_{A,j}$	$\lambda_{A,j}$
$s_1 = (c, 0, B_k)$	s_2		0	$(Y_{A,j}, (1, A_j, B_k))$
$s_2 = (c, 1, B_k)$		$(s_2, (X_{A,j}, (ok, B_k)) \to s_3$ $(s_2, (X_{A,j}, (no, B_k)) \to s_4$	∞	
$s_3 = (c, 2, B_k)$	s_0		tr	$(Y_{A,0}, (ok, r, A_j))$
$s_4 = (c, 3, B_k)$	s_0		0	$(Y_{A,0}, (no, r, A_j))$
$s_5 = (c, 4, B_k)$		$(s_5, (X_{A,j}, (1, B_k)) \to s_6$ $(s_5, (X_{A,j}, (1, x)) \to s_7$	∞	
$s_6 = (c, 5, B_k)$	s_8		0	$(Y_{A,j}, (ok, A_j, B_k))$
$s_7 = (c, 6, B_k)$	s_5		0	$(Y_{A,j}, (no, A_j, B_k))$
$s_8 = (c, 7, B_k)$	s_0		tr	$(Y_{A,0}, (ok, r, A_j))$

state j. Let $r = aA_j + bB_k \to cC + dD$ be such a reaction. In Table 1 we give the full description of the states and transition functions for A_j with respect to a reaction r. For A_j we define a set $X_{A,j} = \bigcup_{r \in R_{A,j}} \{(X_{A,j}, (cod, p_r))\}$ where $cod \in \{ok, no, 1\}$, and p_r is the partner of A_j with respect to the reaction r. We have $X_A = X_{A,1} \times \ldots \times X_{A,n}$. In a similar way we define $Y_{A,j} = \bigcup_{r \in R_{A,j}} \{(Y_{A,j}, (cod, A_j, p_r)), (Y_{A,0}, (cod, r, A_j)\}$, where $cod \in \{ok, no, 1\}$ and p_r is the partner of A_j with respect to reaction r. We consider $Y_A = Y_{A,1} \times \ldots \times Y_{A,n}$. Messages of type $(Y_{A,0}, (cod, r, A_j))$ are sent to the executive. We denote by $S_{A,j,r} = \{s_i | 1 \leq i \leq 8\}$ the set of states specific to the molecule A in state j when it participates to a reaction r. $S_{A,j} = \bigcup_{r \in R_{A,j}} S_{A,j,r}$ is the set of states for molecule A when in state j. Using these sets we define $S_A = S_{A_1} \times \ldots S_{A,n}$, the set of states for the DEVS description M_A corresponding to the molecule A.

Considering the defined functions $\delta^{int}_{A,j} : S_{A,j} \to S_{A,j}$ for $1 \leq j \leq n$, we define $\delta^{int}_A : S_A \to S_A$ as follows: $\forall s \in S_A$, $s = (s_1, \ldots, s_n)$, $s_j \in S_{A,j}$, $1 \leq j \leq n$, $\delta^{int}_A(s) = (\delta^{int}_{A,1}(s_1), \ldots, \delta^{int}_{A,n}(s_n))$. The functions δ^{ext}_A and λ_A are defined in a similar manner. Let s be in S_A, with $s = (s_1, \ldots, s_n)$. For each $s \in S_A$ with $s = (s_1, \ldots, s_j, \ldots s_n)$ we define $T_s = (t_{1,s}, \ldots, t_{n,s})$. If $s' = (s_1, \ldots, s'_j, \ldots s_n)$ with $\delta^{int}_{A,j}(s_j) = s'_j$ or $\delta^{ext}_{A,j}(X_{A,j}, s_j) = s'_j$, then $T_{s'} = (t1, s - \tau_A(s), \ldots, \tau_{A,j}(s_j), \ldots, t_{n,s} - \tau_A(s))$ where $\tau_A(s) = min(t_{1,s}, \ldots, t_{n,s})$. For the initial state $T_{s_0} = (\tau_{A,1}(s_{0,1}), \ldots, \tau_{A,n}(s_{0,n}))$.

In order to define a network model, we start by considering a subset D of molecules. For all $A \in D$, M_A is the above defined system for the molecule A, I_A is the set of influencers for component $A$and Z_A is the A-input function ($Z_A : \times_{B \in I_A} Y_B \to X_A$). Let $Z_{A,B}$ be the projection of Z_A onto component Y_B, and let $y \in Y_j$ where $y = (cod, exp, k_h)$, then $Z_{A,B}(y) = (X_{A,h}, (cod, exp))$ if k_h of y is A, and $Z_{A,j}(y) = \emptyset$ if k_h of y is not A. The executive receives from clients information about reaction that took place or about unsuccessful attempts to react. When quantitative changes appear in network, the executive recomputes the putative times for each reaction (according to Gibson's algorithm), and modifies the clients structures such that for the next reaction the selected client for participation is the one with the least putative time.

We consider a simplified version of DSDEVS and we prove that it has the same computational power as Turing machines. In DSDEVS model, the communication between executive and clients is not explicit. In fact, the executive offers the next rule that could be applied in such a way that it can change the structure of the involved clients. We describe this mechanism of changing the structure of the client in order to allow it the possibility of applying the rule by encoding the rule as a special type of information. We prove that this kind of system with 3 clients is computationally equivalent to a Turing machine; we use results and techniques of the formal language theory.

Enhanced Operational Semantics in Systems Biology*

Pierpaolo Degano[1] and Corrado Priami[2]

[1] Dipartimento di Informatica, Università di Pisa
Via Filippo Buonarroti, 2, I-56127 Pisa, Italy
degano@di.unipi.it
[2] Dipartimento di Informatica e Telecomunicazioni, Università di Trento
Via Sommarive, 1438050 Povo (TN), Italy
priami@dit.unitn.it

We are faced with a great challenge: the cross-fertilization between the fields of formal methods for concurrency, in the computer science domain, and systems biology in the biological realm.

From the one side, re-using the theories developed in the last years for mobile and distributed systems may spread light in the systems biology field. Being based on sound and often deep mathematics, these theories may offer solid ways to describe biological systems and to safely reason upon them. Also, formal methods may provide biologists with software tools that can make them save time and efforts. On the other hand, biological systems often have a size of one or two order of magnitude bigger than that of computer systems. More importantly, their efficiency, flexibility and reliability is incredibly superior to that of any computing machinery. So, an effort of understanding biological mechanisms in terms of computer technology will bring over the computer science field new techniques to develop and analyse complex systems that will be more robust, reliable and efficient than the present ones. As an example, we certainly lack now linguistic primitives suitable to model biological systems. A programming language rich enough will help biologists in formalizing biological systems and in predicting their behaviour. At the very same time, such new primitives will help e.g. the computer scientists to design and program systems in the so-called disappearing computer scenario — a foreseen world where everyday life object is equipped with microchips constantly interacting each other.

Process calculi are maybe the most popular framework to study *global computing*, in which a great number of computing agents that cooperate to achieve common goals, possibly exchanging information through communications or interactions. These agents are geographically dispersed, may move from one site to another, without possibly stopping their ongoing calculations. Also, the knowledge of the running environment is limited and no centralized point of control is assumed. The long term objective of these calculi and their related formal theories is then to design applications, certified to have some clearly stated properties, that may involve thousands or millions of ubiquitous, cooperating entities. Typically, process calculi are built out from a very restricted number of primitives

* Work partially supported by EU project DEGAS (IST-2001-32072) funded within the FET proactive initiative on Global Computing.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 178–181, 2003.

focused on the description of process interaction. They have extensively being investigated in the last three decades since the pioneering work by Hoare [6] and Milner [9]; a closely related sub-field is that of Petri nets, that studies concurrent sytems from the point of view of automata theory. A main result of the reasearch on process calculi are formal theories that deal with several aspects of this fascinating new computing paradigm. Among these aspects, particularly important are the *qualitative* ones, that mainly consider the behaviour of computing agents in terms of equivalences, as well as the *quantitative* aspects, that measure somehow systems behaviour, taking care of probability distributions, time constraints, and so on.

Systems biology [7] is a field that studies the system-level structure and behaviour of complex systems made up of molecular components. In turn, the nature and the behaviour of the components, when taken in isolation, is well-known from molecular biology. The overall goal of this discipline is to control the evolution of these complex systems and to design modifications to them, guaranteeing that some specific properties are satisfied.

As a matter of fact, the goals of systems biology are quite simlar to those of global computing application design. So, if we succeed in modelling biological systems as computer systems, e.g. as computing agents in some process calculus, we shall advance the state of the art in both disciplines.

The connection between the computer and biological world is very well described by the methaphor cells as computations [16]. Biological components (at various level of abstractions) are represented as processes and their interactions result to be communication between processes. Relying on calculi for mobility the effect of a communication can change the future interaction of processes as it happens on the biological side for interacting components.

The above methaphora has been reified exploiting a revised version of the stochastic π-calculus [13] implementd in the BioSPI system [17, 14]. Recently, a biological version of the ambient calculus as well has been considered to model biological systems [15].

Here, we propose to adopt enhanced operational semantics [5] as a descritpion tool to attack the complexity of biological systems within a uniform framework.

The enhanced operational semantics, EOS for short, is an operational way of specifying the behaviour of complex systems in terms of their components, reagrdless of their actual nature, computational or biological or whatelse. It enables its users to design, simulate and analyse systems keeping distinct the various aspects they may have, in particular the many facets related to qualitative and quantitative analysis. Being each aspect orthogonal to the others, it is then possible to combine some of them at a later stage, so taking advantage of a clear separation of concerns.

Recent developments in the EOS theory show that many different families of calculi can be simulated relying on a core π-calculus formalism. The main idea is that the annotations introduced on labels of transitions can be used to control the possible interaction of communicating systems [3]. A further result is that most of the transitions systems originated by calculi for mobility turns

out to be a subset of the π-calculus [10] transition system. As a consequence, we can study properties of the selected formalism in the π-calculus by suitable assignment of parameters to transitions. This unifying framework allows us to select the most suitable formalism to design biological systems and relying on the implementation of a π-calculus kernel (e.g., the BioSPI system) to study properties and performing simulations.

As mentioned above, an important aspect of enhanced operational semantics is that it allows to describe various aspect of systems without changing the actual syntax (that remains the standard syntax of many process calculi). Thus, it is possible to define *causal* or *locational* relations between actions, so expressing, e.g., the need of some bio-chemical reactions to occur before a selected one or the need of the co-location of some reactants to enable an interaction. In a similar, independent way, one can derive or assign *stochastic information* to the system activities, in terms of the probabilities or the statistical rates they have to occur. In this way, we describe system evolutions closer to reality, and we mechanically calculate the probability that a(n un)wanted chain of reactions has to show up. Exploiting a notion of *behavioural equivalence* between processes, one can then group apparently different biological systems, that however exhibit the same dynamics. This equivalence can take as a parameter the wanted (qualitative or quantitative) aspects of the systems under analysis: the user selects what has to be observed and a mechanical (and up to now inefficient) tool is able to detect similarities and dissimilarities. As another application of the equivalences mentioned above, one can study a property known in the sub-field of computer security as *non-interference* [2, 4]: a given system will not interfere with its surrounding environment, whichever it could be, or, in biological terms: does a selected component in a system interfere with others and produce unwanted effects while the whole system evolves?

We end this position paper by noting that although operational semantics is one of the most simple formal tools that can be used in the design of complex systems, it is quite far from the common practice of biologists.

An essential ingredient for the success of the challenge outlined at the beginning is the identification of a communication language between computer scientists and biologists. Such a language must have the following characteristics:

- it must hide as much technical details as possible from the user so that the biologists need not to have strong background in mathematics and formal methods to use it;
- it must be sufficiently structured (i) to avoid as much ambiguities as possible and (ii) to derive consistent formal methods to analyse and simulate systems;
- it must be possible to automatically translate specifications in this language into process calculi specification and to reflect back in the interface language the results of the formal analysis.

To satisfy all the above requirements we think that a graphical, semi-structured language could be a solution at the right level of abstraction. Indeed, biologists already have in use many graphical languages (e.g., [1, 8, 11]) and so computer

scientists do. Additionally, software engineers make use of graphical standards as UML to model systems, and there are already definitions of extractors from UML specifcations to π-calculus descriptions [12] that show the feasibility of the approach and that allow reuse of existing tools.

References

[1] The systems biology workbench software. 2000. 180

[2] M. Abadi and A. D. Gordon. A Calculus for Cryptographic Protocols: The Spi Calculus. *Information and Computation*, 148(1):1–70, January 1999. 180

[3] L. Brodo. *A tool for comparing calculi for mobility* PhD Thesis, Dipartimento di Informatica, University of Verona. 179

[4] A. Durante, R. Focardi, and R. Gorrieri. A Compiler for Analysing Cryptographic Protocols Using Non-Interference. To appear on ACM TOSEM. 180

[5] P. Degano and C. Priami. Enhanced operational semantics: A tool for describing and analysing concurrent systems. *ACM Computing Surveys*, 33,2:135–176, 2001. 179

[6] C. A. R. Hoare *Communicating Sequential Processes*. Prentice-Hall, 1985. 179

[7] H. Kitano. *Foundations of Systems Biology* MIT Press, 2002. 179

[8] F. A. Kolpakov, E. A. Ananko, G. B. Kolesov, and N. A. Kolchanov. Genenet: a gene network database and its automated visualization. *Bioinformatics*, 14(8):529–537, 1998. 180

[9] R. Milner. *Communication and Concurrency*. Prentice-Hall, London, 1989. 179

[10] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes (I and II). *Information and Computation*, 100(1):1–77, 1992. 180

[11] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 27(1):29–34, 2000. 180

[12] K. Pokozy and C. Priami. Formal analysis of UML specifications. DEGAS TR, www.omnys.it/degas 181

[13] C. Priami. Stochastic π-calculus. *The Computer Journal*, 38(6):578–589, 1995. 179

[14] C. Priami, A. Regev, W. Silverman, and E. Shapiro. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80(1), 2001. 179

[15] A. Regev. *Computational systems biology: A calculus for biomolecular knowledge.* PhD Thesis, Department of computer Science and Applied mathematics, Weizmann Institute of Science. 179

[16] A. Regev and E. Shapiro. Cells as computation. *Nature*, Vol. 419, p. 343, 2002. 179

[17] A. Regev, W. Silverman, and E. Shapiro. Representing biomolecular processes with computer process algebra: π-calculus programs of signal transduction pathways. In *Proceedings of the Pacific Symposium of Biocomputing 2000*, 2000. 179

Issues in Computational Methods for Functional Genomics and Systems Biology

Magali Roux-Rouqui[1], Leroy Hood[2], Sandrine Imbeaud[3], and Charles Auffray[3]

[1] Biosystemics, Modeling, Engineering - Institut Pasteur - Paris - France
mroux@pasteur.fr
[2] Institute for Systems Biology - Seattle - USA
lhood@systemsbiology.org
[3] Genexpress Functional Genomics and Systemic Biology for Health
CNRS FRE 2571 7, rue Guy Moquet BP8 94801 Villejuif cedex France
{sandrine.imbeaud,auffray}@vjf.cnrs.fr

Abstract. Systems Biology starts by defining the components of a biological system and collecting the relevant previous biochemical and genetic data on a global scale, using high throughput platforms, formulating an initial model of the system, systematically perturbing the components of the system, and analysing the results. By comparing the observed responses to those predicted by the model, it is then possible to iteratively refine the model so that its prediction fit best to the experimental observations. Finally, new experimental perturbations are conceived and tested in order to distinguish between the multiple competing hypotheses.
We discuss the computational methods used for high throughput data collection in functional genomics, emphasizing the strong need for standardization and quality assurance. We then review the computational needs required for biological system modeling and semantic integration in the systemic framework, arguing that the Unified Modeling Language (UML) seems appropriate to support the iterative process of Systems Biology.

1 Introduction

An exhaustive knowledge of the structure, function and relation of the individual components of biological systems is necessary but insufficient to understand phenotypes. We need to approach biological systems as organized and organizing systems in which modularity, redundancy and feedback ensure system stability, optimisation and robustness. This should help to reveal system functioning through genotype-phenotype relationships, and allow for example *in silicio* target design or drug performance validation. Such challenges are becoming reachable thanks, on the one hand, to high throughput technologies that allow to massively characterize cell components in multiple sets of conditions (time series, internal genetic perturbations,

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 182-186, 2003.

external environmental changes, etc.) and, on the other hand, to the data integration into biological models to simulate their functioning. By data integration, we mean both *technological integration*, which is related to the standardization of experimental data required to achieve interoperability between platforms, and *semantic integration* which can be achieved through the combination of sophisticated biological hypotheses and interpretations. This approach of biological systems requires integrated software platforms for data collection, simulation, visualization, analysis and hypothesis generation. We are actively promoting the implementation of a framework for functional genomics and systems biology [1-3].

2 Computational Methods for High Throughput Data Collection in Functional Genomics

Diverse types of biological data are collected for the identification and characterization of genes, transcripts and proteins (coding regions, functional domains, regulatory sequences, polymorphisms, and structure-function relationships), and used to decipher the organisation and regulation of metabolic pathways and networks in physiological and pathological conditions in a variety of organisms during evolution and development. High throughput data collection platforms for genome, transcriptome, proteome and metabolome studies combine advanced instrumentation and automation (automated DNA and protein sequencers and synthesizers; cDNA and oligonucleotide arrays; mass spectrometry, liquid chromatography and electrophoresis; cell imaging and high speed multiparameter cell sorting), and are under constant development and refinement.

Computational methods are available for storage, distribution, analysis, annotation and integration of large amounts of the diverse types of data registered in a variety of electronic databases about DNA, RNA, protein sequences, protein interactions, transcription factors, expression profiles, gene regulatory and metabolic networks, etc. However, despite community-wide efforts for distributed annotation, and text-based analysis of the literature, the problems of information completeness, exactness and updating remain largely unaddressed. A major rate-limiting factor remains quality assessment of the data produced using these various high throughput platforms. It is not yet fully in place in functional genomics, as it is for DNA sequencing, where it took 20 years to establish accurate sequencing of large genomes.

Novel quality assurance processes are being developed for example for transcriptome studies, which include experimental design, gene and sample collections, array preparation, probe synthesis, hybridization, data transformation, and knowledge extraction. Emphasis is placed on determination of the power of the data capture scheme, of the number of samples needed to address specific biological questions, of the biological and experimental variation expected, of the number of replicas needed to ensure statistical significance of the results. Each of these steps requires specifically tailored computational models and methods.

3 Computational Needs for Biological System Modelling and Semantic Integration

As model design has a pivotal role in systems biology approaches for semantic integration of data, it has to include the conceptual construct of biological systems; such features could be figured out into metamodels that would embed classes of properties from which specific submodels could be derived. As an example, let consider generic biological systems made of large populations of molecules as well as very rare entities present in very small numbers. Such dual distributions could be conveniently specified in an upper level model, and used to guide the selection of the proper formalism for modeling specific subsystems. The subsequent modlling of a subsystem consisting of rare molecules would be more appropriately performed using stochastic rather than deterministic formalisms, as the fluctuations of rare molecules may have stronger effects on system behaviour that ones occurring within a large population, and a stochastic formalism would account for these fluctuations. Metamodeling of the general and specific properties of the living systems would help to identify what properties a peculiar formalism would be able to represent accurately. In addition, the design of such metamodels could be of great help to insure the communication between different scientific communities as the systems biology approaches need interdisciplinary teams. Metamodeling requires the use of a suitable notation for its description, its design and its implementation; we found that he Unified Modeling Language (UML) methodology, which allows graphical specification of a system in an intuitive and easy-to-use manner, is convenient to describe static-structural as well as behavior-procedural aspects of biological systems.

An essential view of the biological system dynamics is achieved through the systemic paradigm and consists in the articulation of the three prototypical changes: temporal changes (storage, memorisation, etc.), morphological changes (processing, computation, etc.) and spatial changes (transport, transmission, etc.) [4]. In this context, biological systems are *reactive* systems that transform themselves and their environment according to internal and external fluctuations; this property is known as *self-organisation* and accounts for the *robustness* of living systems that maintain their organizational structure to achieve a specific task, irrespective of the surrounding events and modifications. It has been hypothesized that robustness is achieved through *redundancy*, as redundant genes (or gene products) are able to substitute for a loss-of-function in one member of the group [5]. This is also known as *homopoiesis* and could be insured through negative regulation. Otherwise, living systems can change their organisation and differentiate to reach new states and accomplish new tasks. Accordingly, tremendous changes can be induced through minor events and point to the *non-linear* nature of living systems (think to the developmental changes induced in the egg by sperm entrance); these changes are known as *morphogenesis* and involve very often *cis*- and *trans-regulation* of gene transcription. One additional property concerns the *pleiotropy* of some genes or gene products, the property of a single entity to affect multiple phenotypic traits. Different biological features are embedded in this property, and seven types of pleiotropy have been reported [6]. Among them, *combinatorial pleiotroy* is the most frequent and concerns a gene product that interacts with different partners, and mutations affecting such entities

have broad range effects; most of the transcription factors fall into this category. Otherwise, *unifying pleiotropy* describes multifunctional proteins that carry multiple domains each involved in distinct functions.

As we mentioned above, modeling of the properties such as the *actions* performed by the systems components at the *lower level*, the fluctuations of the *internal* and/or *external environment,* and the execution of a particular *task or function* at the *higher level*, can be based on a variety of mathematical or computational formalisms, each of which will represent only some parts of the biological properties. For example to model enzymatic reactions, in steady state, stoichiometric (global) modeling should be preferred, because kinetic reactions that describe individual reaction rates, regulation and time required for ligand binding and product release are not necessary. In contrast, to account for the differential binding of substrates and allosteric effectors to enzymes, kinetic formalism is necessary (note that both models account for the structural invariance of the enzyme). Otherwise, the history of the state transformations may be assessed through formalisms such as Petri nets, multi-agent systems, and cellular automata which seem promising avenues to explore in Systems Biology.

Simulation consists in model execution, that means the actualisation of system variables or states values over time. Generic biochemical approaches are currently used; nevertheless, simulation techniques for model-checking as those used in software engineering, could be fully adapted for Systems Biology in which molecular entities are modeled as individual software objects. For example, using a real-time extension of UML, UML-RT based on ROOM methodology (Rose RealTime UML tools), we have been able to model and simulate molecular processes, expected and unwanted system behavior being modeled using state-machines and collaboration diagrams ; these state machines were further compiled into timed automata as used by the model checker. We found that one of assets of this formalism was to support high-levels of standardization [due to the Object Management Group (OMG) involvement in standardization activities] as well as to accommodate graphical models with quantitative and qualitative tool extensions (see Kennedy Carter, http://www.kc.com). For most of these reasons, we believe that UML can play a role by underlying the design of modeling languages for Systems Biology.

The systemic approach allows model design at a given organisational level. It does also make it possible the design of hybrid models as well as the coupling of different organisational levels, notably to study emergence resulting from collectively functioning molecular entities. Due to the variety of formalisms available to implement these models, their further comparison and exchange are made difficult. This strongly argues for the development of computational generic platforms that would facilitate and accommodate these couplings. One another alternative would be to conceive models according to a standardised, common formalism, integrating the modeling of processes of different natures. Modeling languages dedicated to the modeling of biochemical networks (SBML, CellML, E-Cell, etc.) have to be reconsidered to support the modeling of biological systems according to the properties mentioned above.

4 Discussion

The Systems Biology approach consists of the interplay of high throughput biological data with models of biological systems to insure the integration of these informations and to extract meaningful biological knowledge. This means that the information systems that must be designed to perform such integration have to deal with the flows and the quality of data, one the one hand, and the description, the design, the implementation and the execution of the model, on the other hand. This is very different from models in theoretical biology which consider only limited amount and variety of data. In this respect, the Systems Biology approach requires the development of large information systems as those designed for military (missile guidance systems, weapon control system, etc.) or civilian (flight control systems, cellular phone, etc.) applications. The object-oriented methodology is currently used to assess complex systems; an example is the GEANT 4 project for the simulation of the particles passing through matter. The problem domain construction allowed to delineate a sub-domains hierarchical organization that includes a large variety of physical methods implemented as classes in the solution domain construction (http://wwwinfo.cern.ch/asd/geant4/geant4.html, see OO Analysis an Design). Similar approaches could be investigated for Systems Biology that require storage of huge amount of data and their dynamic querying; the BioUML tools developed for visual modeling and simulation of biological systems (http://www.biouml.net) is accounting for these new perspectives.

References

[1] Ideker, T., Galitski, T., and Hood, L.: A new Approach to Decoding Life: Systems Biology. Annu. Rev. Genomics Hum. Genet. 2 (2001) 343-72.

[2] Ideker T., Thorsson, V., Ranish, J. A., Christmas, R., Buhler, J., Eng, J.K., Bumgarner, R., Goodlett, D.R., Aebersold, R. and Hood, L.: Integrated Genomic and Proteomic Analyses of a Systematically Perturbed Metabolic Network. Science 292 (2001) 929-34.

[3] Baliga, N.S., Pan, M., Goo, Y.-A., Yi, E.C. , Goodlett, D.R., Dimitrov, K., Shannon, P., Aebersold, R., Ng, W.-V. and Hood, L.: Coordinate Regulation of Energy Transduction Modules in Halobacterium sp. Analyzed by a Global Systems Approach. Proc. Nat. Acad. Sci. USA. 99 (2002) 14913-14918.

[4] Roux-Rouquié, M. and Lemoigne J.L. : The Systemic Paradigm and its Relevance to the Modeling of Biological Functions. C. R. Acad. Sci. Biologies 325 (2002) 419-430.

[5] Weintraub H.: The MyoD Family and Myogenesis: Redundancy, Networks and Thresholds. Cell 75 (1993) 1241-1244.

[6] Hodgkin J.: Seven types of Pleiotropy. Int J. Dev. Biol. 42 (1998) 501-505.

Integrating Biological Process Modelling with Gene Expression Data and Ontologies for Functional Genomics (Position Paper)

Liviu Badea and Doina Tilivea

AI Lab, National Institute for Research and Development in Informatics
8-10 Averescu Blvd., Bucharest, Romania
badea@ici.ro

In the current post-genomic era, various aspects of gene function are being uncovered by a large number of experiments producing huge amounts of heterogeneous data at an accelerating pace. Putting all this data together, while taking into account existing knowledge has become a pressing need for developing environments able to explore and simulate biological entities at a *system level*.

We argue for the need to create a *common bioinformatic framework for modelling biological processes* by a non-trivial *integration* of various complementary functional genomics data and knowledge with the goals of representing and simulating the relevant biological networks and pathways, discovering targets for drugs and diagnostics, as well as determining the molecular mechanisms of diseases from gene expression data. Such a synergetic use of the available data will allow partly replacing certain costly, or even impractical biological experiments by "in silico" simulations of biological processes, as well as enable new in-depth and large-scale experiments.

There are currently at least three types of extremely valuable resources, which are currently not used at their full potential:

- *gene expression data* (e.g. from microarray experiments)
- knowledge about *networks and pathways* (such as metabolic, genetic control, signalling pathways and protein interaction maps)
- *ontologies* (such as the Gene Ontology).

For example, gene expression data are extremely useful for understanding gene function at a global level, but they are typically used *without taking the relevant network and pathway knowledge into account*. (This is due not only to the incompleteness of pathway databases, but also their limited representation and interoperability.)

Also, a lot of effort has been put into manual construction and annotation of pathways – this valuable knowledge should be used in as many contexts as possible. Unfortunately however, the modelling languages used by various annotation efforts are slightly different in terms of expressiveness, making the fusion of knowledge from such different pathway databases a difficult knowledge modelling problem.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 187-193, 2003.

Modelling Language, Ontologies

The main goals of system biology are related to modelling biological entities at a system (holistic) level for various purposes: analysis, simulation, prediction, etc (listed in increasing order of complexity). Thus, while analysis does not necessarily require complete models of the systems involved, simulation and especially prediction are not feasible without complete knowledge (at least at a certain level). For example, the structure of a genetic network involved in organism development may not be enough for simulation or prediction without detailed knowledge of various reaction parameters, which might be hard to obtain for all reactions.

Thus, one of the most important requirements of this field is to allow an as detailed as possible representation of all relevant aspects of the biological processes to be modelled. Still, since most existing knowledge is either very detailed, but covering only a very few biological processes, or much less detailed, but having a wide coverage, we argue that a very expressive modelling language will not do, since it will not be able to deal with the more sketchy (less detailed) knowledge that is available today. Still, the knowledge of the structure of the system will allow certain qualitative conclusions to be drawn, even in the absence of numerical parameters. Being able to exploit various heterogeneous resources for reasoning about biological systems at various levels of detail seems to be the major challenge in the field.

The modelling language thus needs to be able to describe both qualitative (e.g. structural) and quantitative aspects of a model at various levels of detail, in order to allow the integrated use of practically all existing biological knowledge, ranging from expert-curated pathway databases (such as KEGG [8], TRANSFAC/TRANSPATH [13], CSNDB) to high-throughput but less detailed experimental data such as protein-interaction data, or even putatie computationally-derived annotations. The language should allow a *gradual transition from less detailed qualitative knowledge to very detailed quantitative knowledge about biological mechanisms* and *the use of such partial models during reasoning at all intermediate stages*.

Since the models should also be processable by computer programs (and not only by human biologists, as it is still the case today[1]), the modelling language will have to have a precisely defined formal semantics, that would allow the correct interoperation of the various software modules using them.

Of course, all of these features require much more sophisticated reasoning tools. While for a *uniform* modelling language (even a very expressive one, based for example on partial differential equations), reasoning is relatively easy using existing tools, reasoning in a heterogeneous modelling language allowing descriptions at various levels of detail is highly non-trivial and should rely on *open* architectures (open both from a technical and a conceptual point of view). *Technically*, the architecture should allow the integration of various software modules (e.g. PDE solvers, numerical packages, symbolic reasoning tools such as abductive reasoners, constraint satisfaction modules, process simulation and analysis tools, etc.). *Conceptually*, there should exist a unified model able to view the various types of knowledge in a uniform framework. A potential candidate for such an *open* modelling unified architecture could be a high-

[1] Many pathway databases (e.g. KEGG) are currently more oriented towards a human user interface, rather than a computer processable one.

level constraint reasoning environment such as a Constraint Logic Programming (CLP) system allowing the declarative implementation of constraint solvers using Constraint Handling Rules (CHR) [3].

Developing adequate representations for genes, proteins, networks, pathways, etc. is crucial for developing an integrated framework for molecular biology and genetics data. Current representations are rather fragmentary – a much tighter integration of the various representations is required. Such representations have to refer a common vocabulary of terms (in molecular biology / genetics), such as the *Gene Ontology* [4].

Since the resources in this field are distributed and currently accessible via Web-based interfaces, it is important to make their content accessible in a "semantic Web" format (e.g. using newly proposed standards such as DAML+OIL [5]). Such enhanced representations allow not just *expressive* constructs with a formally defined semantics, but also automated reasoning about them (without such inference services, the representations are useless w.r.t. automatic processing, which is essential in a field involving huge amounts of data and knowledge).

We target the following aspects:

- **Modelling Various Types of Biological Networks and Pathways** (metabolic, genetic control, signalling networks) in a unified framework that should also allow their simulation as well as automated reasoning. In our mind, it is important to devise a representation formalism for biological pathways that is not only very expressive, but also usable by sophisticated reasoning services (such as matching subnetworks by complex logical descriptions of molecular disruptions of a target disease).
- **Devising Ontologies** more sophisticated than e.g. the Gene Ontology (which is just a hierarchy of molecular biology/genetics terms), possibly with domain-specific constructs, having a limited scope. An essential part still missing in all existing ontologies is the information about networks and pathways, which are essential for the new emerging field of Systems Biology. For example, a 'molecular interaction' might be not just a vocabulary term, but also a complex object (possibly similar to a transition in a Petri net) with associated components (in this case substrates/products) as well as reasoning components (that can be invoked to reason about such interactions).

The ever-growing amount of experimental data in molecular biology and genetics requires its automated analysis, by employing sophisticated knowledge discovery tools. In [1] we used an Inductive Logic Programming (ILP) learner to induce functional discrimination rules between genes studied using microarrays and found to be differentially expressed in three recently discovered subtypes of adenocarcinoma of the lung. The discrimination rules involve functional annotations from the Proteome HumanPSD database in terms of the Gene Ontology (GO), whose hierarchical structure is essential for this task.

It is encouraging that the discriminations obtained are biologically sensible – this heavily relies on the GO and the HumanPSD annotations. But this also automatically prompts the question of whether more sophisticated knowledge representation formalisms, such as Description Logics (DL) might allow even more precise functional distinctions to be made.

A DL may allow an "on-the-fly" construction of concepts, rather than relying on a fixed hierarchy. Thus, we wouldn't need to explicitly record in the ontology all generalizations of existing concepts. For example, the current GO contains not just specific concepts like '*cyclin-dependent protein kinase inhibitor*' or '*transmembrane receptor protein tyrosine kinase activator*', but also their generalization '*kinase regulator*'. On the other hand, a DL may take advantage of the *intrinsic composite nature* of the concepts above and represent them as $\exists inhibits.CDK$ and $\exists activates.TRPTK$. Their generalization need not be explicitly represented, since it can be computed by taking the *least general generalization* ("least common subsumer" in DL terminology) $\exists regulates.kinase$ of the two concepts above. Using a DL would also simplify the update and maintainance of the ontology by not having to explicitly specify all the inheritance relationships of a new concept, as well as by providing automated consistency checking tools. Another useful extension of GO would involve integrating it with metabolic, regulatory and cell signalling pathway databases (which would allow more precise causal reasoning – for example, determining possible primary causes for complex genetic disruption profiles).

Such more sophisticated representation formalisms for pathways and genes/proteins would allow mapping gene expression data onto the pathways (thereby generating pathway *activations*), which could be used in various abductive and inductive reasoning mechanisms involving:

- disease recognition (by matching pathway activations to logical descriptions of the molecular mechanisms of diseases)
- discovery of disease mechanisms by mining pathway activations.

Knowledge Discovery and Machine Learning

A combined use of microarray gene expression data, pathways and functional annotations (e.g. in terms of the Gene Ontology) in a common framework enables not just the *interpretation* of experiments in a detailed biological context, but also the *discovery* of various types functional knowledge. Sophisticated machine learning algorithms able to deal with background knowledge (such as currently known pathways) are needed to achieve this. Already the most basic background knowledge on functional annotations, which involves *hierarchies of concepts* (as in GO, where the main type of relational information is in the form of *inheritance* relationships), is not directly treatable by propositional learners like C4.5, but could be dealt with our approach from [1]. We argue the need to go beyond attribute-value learners, i.e. towards relational learners [7], which are able to deal with structured representations and sophisticated background knowledge. Still most existing learners are either employing covering-based methods (which are inappropriate in this context), or are learning a large number of association rules, without any simple means of selecting the most relevant ones. A precise measure of interestingness of induced rules w.r.t. a given background knowledge is needed.

Pathway Reconstruction

A modelling language for system biology – even a very expressive one – is useless without the ability of aquiring knowledge about significant portions of the biological processes of interest. Knowledge aquisition is particularly difficult in biology not only due to the sheer number of entities involved, but also due to their heterogeneity. We have already advocated the need for reusing existing knowledge in whatever form it might currently exist. Sometimes, however, this is not enough and large-scale manual aquisition is not only very expensive, time-consuming, but also prone to errors.

An alternative to manual knowledge aquisition in this domain could be the *automated reconstruction of biological pathways*. There are currently several types of knowledge relevant to the reconstruction (refinement) of biological networks and pathways:

- partial knowledge about pathways (from pathway databases, textbooks, literature)
- gene expression data (e.g. from microarrays experiments)
- functional annotations (e.g. in terms of the Gene Ontology from the Proteome databases).
- There are also many approaches and algorithms for reconstructing pathways, but they have several important limitations:
- they typically deal with just the raw expression data (without taking into account partial knowledge about pathways or functional annotations that are already available)
- they typically require an impractically large number of knockout (or other genetic modification) experiments in order to be able to learn a *complete* network [9]. We would like to have a more incremental (i.e. an anytime) pathway reconstruction algorithm that is able to refine partial pathways in an incremental fashion.

Existing gene expression compendia (such as the Rosetta compendium for yeast [6]) contain data from a large number of microarray experiments (including knockout experiments) involving practically all yeast genes. We argue that the preliminary analysis attempts (involving most clustering of genes and expression profiles [eg. 6]) only scratch the surface of the knowledge hidden in such compendia. The most important and challenging task involves *extracting causal influence information* from such data.

While existing data mining and machine learning approaches (such as association rules learners) extract mainly shallow associations (correlations) present in data, we are aiming at discovering the true *causal structure* of the networks of interest. (Of course, for a limited amount of experimental data, only partial knowledge about the causal structure may be inferrable. We insist on the need to be able to express and refine such partial models.) The probabilistic nature of many biological processes (as well as the unavoidable noise present for example in microarray data) requires the use of probabilistic models, such as Bayes nets. Superficially, our problem resembles that of Bayes net *structure learning* – an already very difficult research problem. Unfortunately, existing Bayes net structure learners cannot be directly employed in this domain, due to certain specificities of biological networks:

1. Biological networks are *cyclic*, may have *latent (hidden) variables* (i.e. variables not present in the measured expression data) and may be prone to *selection bias.* (On the other hand, Bayes net structure learners deal only with *acyclic* networks[2], and only few are able to deal with latent variables and selection bias).
2. Since Bayes net *structures* are not Bayesian themselves (additional experimental data may change the most likely structure significantly), we need to *infer only the features of the network that are fully justified by the data* (rather than a full Bayes net in which certain edges and/or edge orientations are not fully justified by the data, but rather represent the best scoring structure). The most popular structure learning algorithms (the *scoring-based* ones) are thus inapplicable in our setting, while none of the existing *constraint-based* algorithms (e.g. IC [10], PC, FCI [11]) covers all the requirements from (1) above.
3. Although the constraint-based algorithms are able to infer causal structures from purely *observational* data, upgrading them to deal with a combination of observational and *experimental* data is not straight-forward and hasn't been solved yet. (The main problem seems to be related to the relatively small sub-populations of compendium samples for each perturbation, which doesn't allow reliable independence tests.) On the other hand, scoring-based methods are easily able to deal with this problem [2], but they are unapplicable due to the problems described at point (2) above.
4. Taking into account existing *background knowledge* (in the form of partial networks) is also extremely important, especially due to the very large number of variables (genes) involved in such causal inference experiments.

We are currently developing a constraint-based causal structure learner addressing all of the above problems. Since the first phase of the algorithm is very similar to clustering methods that are very popular in this domain, we may be able to compare the deeper models constructed by such a causal learner with the more shallow clusters reported in the literature. Our preliminary experiments currently show the computational feasability of our approach, as we are currently able to deal with networks of reasonably large size (e.g. 800 genes).

In conclusion, we argue for a modelling environment for system biology that is not only very expressive, but also allows a non-trivial combination of various data and knowledge sources *at various levels of detail* and supports the *automated reasoning* about the various aspects of biological function. We also advocate the potential utility of causal structure learners for obtaining at least partial drafts of biological networks from expression data.

[2] The CCD algorithm of [12] is able to deal with cyclic structures, but not with latent variables. It is also limited to linear dependencies.

References

[1] Badea, L.: Functional discrimination of gene expression patterns in terms of the Gene Ontology, Proc. of the Pacific Symposium on Biocomputing PSB-2003.
[2] Cooper, G.F. and C. Yoo, Causal discovery from a mixture of experimental and observational data, Proceedings of the Conference on Uncertainty in Artificial Intelligence (1999) 116-125.
[3] Fruewirth T. Theory and Practice of Constraint Handling Rules, JLP 37:95-138, 1998.
[4] Gene Ontology: tool for the unification of biology. Nature Genet. 25: 25-29, 2000.
[5] Ian Horrocks et al. Reviewing the design of DAML+OIL: An ontology language for the semantic web. AAAI 2002 http://www.daml.org
[6] Hughes et al.Functional discovery via a compendium of expression profiles.Cell 102(1):109-26, 2000
[7] Nienhuys-Cheng, S.H., R. de Wolf. Foundations of Inductive Logic Programming, Springer, 1997.
[8] Ogata, H., Goto, S., Fujibuchi, W., Kanehisa, M.: Computation with the KEGG patway database, BioSystems, 47, 119-128(1998), http://www.genome.ad.jp/kegg/kegg2.html
[9] Pe'er, D., Regev, A., Elidan, G., Friedman, N.: Inferring Subnetworks from Perturbed Expression Profiles, Bioinformatics, Vol.1, no.1 2001, pages1-9.
[10] Pearl, J., Verma, T.S.: "A theory of inferred causation", Proc. KR-91, 441-452.
[11] Spirtes, P., Meek, C., and Richardson, T. Causal inference in the presence of latent variables and selection bias. In Proceedings of UAI-95, pp. 499-506.
[12] Spirtes, P.: `Directed cyclic graphical representation of feedback models', Proceedings of the 11th Conference on Uncertainty in Articial Intelligence, Montreal QU: Morgan Kaufmann, pages 491-498.
[13] Wingender, E.:The TRANSFAC System on Gene Regulation, Trends in Glycoscience and Glycotechnology 12, 255-264 (2000), http://transfac.gbf.de/TRANSFAC/

Computer Simulation of Protocells

Doron Lancet

Department of Molecular Genetics, Weizmann Institute of Science
Rehovot 76100, Israel
doron.lancet@weizmann.ac.il

When attempting to analyze, simulate and comprehend a vastly complex network such as a present-day living cell, one face enormous computing burden. Our group has in last decade worked on analyzing with similar methodologies the chemical behavior of much simpler systems, namely protocells. The advantage of this computational approach to prebiotic entities is that most of the principles are already in place, but the components are much simpler. We propose that understanding of such simpler systems, as described below, could shed new light on the nature of biological complexity, and help hone new computational tools, such as advanced stochastic chemistry simulations, to assist in the understanding of present-day cellular networks.

The GARD Model. The Graded Autocatalysis Replication Domain (GARD) model (Segre et al. 1998; Segre et al. 2000; Segre and Lancet 2000; Segre et al. 2001; Segre et al. 2001) describes the behavior of a spatial domain constituting a molecular assembly. Molecules in the assembly manifest mutual catalysis, resulting in a global self-propagating behavior of the entire assembly, in a way reminiscent of autocatalysis. Under a certain set of constraints, this behavior resembles replication. Graded behavior enters in two different ways: 1) The rate enhancement values related to mutual catalysis assume graded values 2) Replication is graded both in its extent and its fidelity, because it is based on two graded variables, size and composition.

In its simplest embodiment, GARD is used to simulate the behavior of molecules that join and leave, and undergo only non-covalent reactions with other molecules. In this embodiment no biosynthesis takes place, and the only compounds present within the assembly are those supplied from the outside (complete heterotrophy). The free energy source that "fuels" such a "joining GARD" is the negative free energy gradient for lipid-like molecules transferred from an unfavorable aqueous medium to the inside of an amphiphilic micelle. Mutual catalysis events occur on these downhill reactions, in ways analogous to catalyzed amphiphile flipping within membrane or to lipid-catalyzed ligand-receptor interactions. It should be stressed, however, that the formalism explored here is not restricted to lipid micelles or vesicles, and is applicable also to other modes of molecular enclosures.

GARD's Time Dependence. At time t, a GARD assembly contains a sub-repertoire of N_E molecule types out of a total repertoire of N_G different types. Based on kinetic equations resembling replicators dynamics, the time dependent composition vector $\mathbf{n}(t)$ follows a trajectory in N_G dimensional composition. The molecules present at a given time point have positive integer components in the compositional vector, while

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 194-197, 2003.

those absent are represented by zeros. This time trajectory is far from random, and depends on the mutual catalytic interaction parameters (defined by a matrix ε_{ij}), on initial conditions and on statistical noise. The latter is related to the fact that small assemblies of discrete molecules are considered. This non-trivial behavior lasts as long as the assembly is away from equilibrium. At $t=\infty$ equilibrium sets in, where by, in a simplified case in which all molecules are thermodynamically equivalent, **n**(t) of the assembly will reflect the composition of the external milieu.

A much more interesting, non-trivial behavior, obtains when a GARD assembly is maintained in perpetuity away from equilibrium. This is achieved by adding to the simulation an external free energy source. Curiously, what is needed is *destructive* energy, i.e. the occasional disruption of the assembly. One possible route for this is assembly splitting, e.g. fission. This simulated process would mimic natural ones, which occur when micelles grow beyond a certain limit. The relevant physical forces could be turbulence, temperature changes, surface tension and the like. The argument for the high probability of fission can be reformulated as a statement that under primitive earth conditions the chances of a molecular assembly to grow and never split are small.

Bequeathing Compositional Information. Rewardingly, the splitting assumption also serves to render the GARD model with a capacity to emulate inheritance. It is obvious that if a perfect equal split occurs after a perfect homeostatic growth, the two progeny assemblies will be identical. In reality, of course, this might never happen. Even in the incredibly orchestrated modern cell division, DNA miscopying and inequality of sharing of large molecules and organelles between progeny cells might occur. This imperfection may be deleterious, in which case a daughter assembly might lose its mutually catalytic network and perish.

On the other hand, just like in the case of present-day cells, certain mutation-like compositional changes, resulting from the imperfections of growth and fission, might actually be advantageous. For example, a more effective mutually catalytic network may be established through the fortuitous entry of a compound A_x not previously included in the assembly. If this network is more efficient than the previous one, then it will now begin to acquire A_x from the environment, and the newly established composition will thus be propagated along many growth-split cycles. This amounts to a mutation being passed along to future generations. The complex dynamics displayed by the GARD model is a result of this class of phenomena.

We have developed a simple formulation for assessing the success of progeny generation. A compositional similarity measure H, computed as a normalized scalar product of two compositional vectors, assists in this computation. It is possible to accurately quantify the similarity of two progeny assemblies to each other and to the parent assembly. The red squares in the autocorrelation diagram, signify time periods during which, despite repeated growth and splitting events, the compositional vectors for different time points remain clustered in N_G-dimensional space. In other words, the assembly undergoes repeated homeostatic growth periods.

Had different runs of the program (with different catalytic parameters) always resulted in just one large red square, this would indicate success of assembly replication, but lack of any evolution-type change. In fact, only a few cases show such a “boring” behavior. In some of these the randomly sampled mutual catalysis matrix happens to have a single relatively large value on its diagonal, signifying that one of

the N_G substances is a strong autocatalyst that dominates the dynamics of the GARD assembly. Yet, mesobiotic phenomena become significant when they involve ensemble complexity. Thus, a GARD composed of single autocatalyst must be regarded as degenerate.

Most of the GARD simulation runs look much more interesting. Every so often a homeostatic assembly gives way to a completely different one, which is also homeostatic. This is akin to multiple attractors, or set points, or stationary states, observed in other dynamic systems. Because each of these states is characterized by a different molecular composition, we named them *composomes*. The transitions between composomes likely arise from the accumulation of changes that occur due to imperfect mutual catalysis and imperfect fission. In other words, one may discern here an intriguing analogy to the way by which the accumulation of genomic mutations lead to speciation. In the spirit of the continuity principle, it should be also noted that the persistence of composomes associated with faster growing sets of mutually catalytic molecules has an interesting correspondence in the optimal growth performance observed in some present-day unicellular organisms.

A Computational Origin off Life Endeavor. While the last century brought an exquisite understanding of the molecular basis of life, very little is known about the detailed chemical mechanisms that afforded the emergence of life on early earth. There is a broad agreement that the problem is in the realm of chemistry, and likely resides in the formation and mutual interactions of carbon-based molecules in aqueous medium, similar to those that take place in present-day living cells. Yet, present-day experimental approaches can only capture the synthesis and behavior of a few molecule types at a time. On the other hand, experimental simulations of prebiotic syntheses, as well as chemical analyses of carbonaceous meteorites, suggest that the early prebiotic hydrosphere might contain thousands of different compounds. We wish to explore the idea that given the limitations of test-tube approaches with regards to such a 'random chemistry' scenario, an alternative mode of analysis should be pursued. It is argued that as computational tools for the reconstruction of molecular interactions improve rapidly, it may soon become possible to perform adequate computer-based simulations of prebiotic evolution. We thus propose to launch a computational origins-of-life endeavor, involving computer simulations of realistic complex prebiotic chemical networks. Specific examples we pursue are based on a specific embodiment of GARD, the novel algorithmic approach to prebiotic evolution, which constitutes a hybrid of molecular dynamics and stochastic chemistry. As potential solution for the immense hardware requirements dictated by this approach, we have begun to implement both advanced grid computing and an idle CPU harvesting scheme, under the title ool@home.

References

Segre, D., D. Ben-Eli, D. W. Deamer and D. Lancet (2001). "The lipid world." Orig Life Evol Biosph **31**(1-2): 119-45.

Segre, D., D. Ben-Eli and D. Lancet (2000). "Compositional genomes: prebiotic information transfer in mutually catalytic noncovalent assemblies." Proc Natl Acad Sci U S A **97**(8): 4112-7.

Segre, D. and D. Lancet (2000). "Composing life." EMBO Rep **1**(3): 217-22.

Segre, D., D. Lancet, O. Kedem and Y. Pilpel (1998). "Graded autocatalysis replication domain (GARD): kinetic analysis of self-replication in mutually catalytic sets." Orig Life Evol Biosph **28**(4-6): 501-14.

Segre, D., B. Shenhav, R. Kafri and D. Lancet (2001). "The molecular roots of compositional inheritance." J Theor Biol **213**(3): 481-91.

How to Solve Semantic Puzzles of Systems Biology

Olaf Langmack

Transformal GmbH
10969 Berlin
langmack@transformal.com

Abstract. A methodological argument for a language oriented research frame-work is presented. The framework supports the transparent and explicit experi-mentation with model fragments from systems biology on one hand and from computer science and engineering on the other hand. It allows to effectively combine theoretical and experimental approaches to systems biology.

1 Rationale

Kristen Nygaard, computer scientist, coined an aphorism about the cognitive quality of programming: "Programming is to understand". In his line of reasoning one question is implicit with all research in computational biology: How to best blend subdomains from biology with concepts from computer science? – E.g., from the rationale of an example project[1]: "What if the pathways of human biology could be modeled with the same tools we use to model computer circuits?"[2]

Looking at the growing number of demonstrators in all of systems biology one can differentiate between two strategies[1] in blending computer science and biology:

- *Tool Integration;* i.e. establish engineering frameworks, establish data communication standards at different levels of abstraction to allow for data exchange or application level cooperation. (E.g. [3,4], [5], [6])
- *Concept Reuse;* adapt calculi from a variety of – mostly – theoretical computer science and other engineering domains to serve as a model for some subdomain of biology. (E.g. [7])

With respect to concept reuse, one could further differentiate tactics of such reuse with respect to the emphasis on either the computer science or biology side. Depending on whose results are taken as a starting point – or as dominating influence – one would e.g. either extend a calculus from computer science to fit the application domain; or take parts from a subdomain of biology to fit some calculus. – The implicit danger being, that either the semantic wealth of the application domain or of

[1] Each of these strategies undoubtedly has its merits. This is not to qualify any of their approaches or findings. We take this categorization only to explain our perspective on how to improve model development for systems biology.

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 198-201, 2003.

the calculus at hand are not utilized to their capacity. Or, while being restrained with dogma from either side a fresh perspective on conceptual innovation is missed altogether.

2 Deconstruct Given Semantics

Even though it may seem a play of thoughts, it explains why we look for a methodological and technical framework that inherently avoids putting to much emphasis on either side when developing models for computational biology. A framework that at the same time allows to focus on semantic modeling without restraining modeling by some already established system of abstractions.

We take this point of view not only for an abstract search for balance or purity. We believe, that an impartial view of both systems biology and computer science will yield models not to be uncovered otherwise.

For the process of model development we suggest to – metaphorically speaking – deconstruct both computer science (or other engineering) calculi as well as subdomains of systems biology. Semantic fragments resulting from this deconstruction could be partially integrated and validated through laboratory experimentation. This atomic step in model development is then to be iterated until a convincing model evolves. One could label this approach a fine-grained construction of semantics from the resources of different domains, rather than a coarse-grained reuse of given and coherent semantics from either domain.

In addition, technical frameworks already exist to experiment with fragments of model semantics. They have the capacity to generate software tools automatically from abstract specifications. Their use would allow laboratory experimentation, and so would allow to validate fragments of model semantics in the early stages of model development.

From a science history point of view, and abstracting from notational and technical tools one could well oppose: The historical trace of findings in computational biology were to be described as a series of such “hybrid” models anyway. We would agree. But we believe it will make a difference, if model development treats semantics in an explicit, fine-grained and – with respect to resources for tool development – light-weight manner.

With prototyping of partially integrated semantic models, solving semantic puzzles of systems biology can become subject of explicit construction. It will otherwise continue to appear as an act of creation.

3 Proper Modeling Framework

One of, if not *the* most established method in practical computer science is compiler construction. That is the – definition and – implementation of programming languages; or the implementation of software tools translating high level programming abstractions into machine language ready for execution on a computer.

Compiler construction is sound in terms of theory, methodology and tools. Its use for decades has stripped it from ideological egg shells like “automatic programming”, “general problem solving”, to name a few.

Chomsky’s attempt [8] to provide a sound mathematical basis for the description and (cross-)compilation of natural language had an important spin-off in the computer science domain of compiler construction. It provided the theoretical basis of *compiler generation*, i.e. the automatic generation of a compiler for a given programming language from a formal description of that language.

Today, numerous compiler generator toolsets in a variety of functionality, performance and platforms are available. (See [9].) Their advantage for the software engineer is the ready availability of established and well understood interfaces within the compilation pipeline, together with well understood methods and specialized tools for the interpretation of specifications of each segment of the compilation pipeline.

Their use for solving puzzles of systems biology will allow to separate

(a) notational and visualization issues from
(b) the description of and experimentation with model semantics and from
(c) interfacing devices, simulations, etc..

It will allow to set the focus of research on semantics without giving away the rapid *down-to-laboratory* prototyping.

4 Analogy: Programming Paradigms

An analogy from the history of computing shall serve as example.

The subject of research in programming languages is programming. The creators of programming languages contend for superior tools. Or: The domain of model development for programming languages is programming itself.

In this realm over time three paradigms have been identified. They categorize the majority of existing programming languages. Referring to them one speaks of

(i) imperative,
(ii) functional & logical (or: declarative) and
(iii) *object-oriented* programming.

For each of them there exist a number of languages and programming environments.

In the course of their identification and popularization either one at least once has been advertized with its superior ability to model reality – or at least significant parts of it, like intelligence.

But instead of interpreting Nygaards “programming is to understand” naïve, and use programming languages to only describe models of the application domain, we suggest to interpret their calculus as potential models of the application domains themselves.

Consequently, the methodological and technical framework of choice would have to efficiently support the description of a variety of calculi. While interpreting programming languages – including language extensions targeting parallelism, concurrency, real-time, distribution, persistence – as draft for domains in the realm of

systems biology, one needs a framework, that easily allows to transcend or combine whatever semantic aspects programming languages or their extensions have to offer.

The only software engineering frameworks not only offering this support by chance, but by design are compiler-generating systems. And while in science history they originated together with programming paradigms and languages; they today could simply be utilized for the purposes of systems biology.

5 Perspective

After

(a) identifying relevant domains of systems biology;
(b) identifying engineering and computer science calculi;
(c) a minimal set of abstractions to serve as interface with notational and/or visualization devices and a minimal set of abstractions to serve as interface description with actuators

has to be established. With the addition of a compiler generator system this will technically and methodically constitute a starting point of a semantic integration framework.

Its *set-up* will either yield a practical validation of given standards (like [3,4]) or it would yield newly drafted standards with their focus set at in-depth semantic modeling. Its *use* will yield models integrating domains from systems biology and computer science. Eventually it will represent a combination of theoretical and experimental approaches to computational biology.

With this kind of reuse of concepts from computer science, researchers can focus at the core of what the involved disciplines have to offer – a wealth of semantics –, rather than puzzling with petty IT detail.

References

[1] S. Eker, M. Knapp, L. Laderoute, P. Lincoln, Carolyn Talcott; "Pathway Logic: Executable Models of Biological Networks", Electronic Notes in Theoretical Computer Science 71 (2002), 18pp.
[2] Quote from: http://www.csl.sri.com/projects/bioinformatics/
[3] M. Hucka, A. Finney, H. Sauro, H. Bolouri, "Systems Biology Markup Language", 2001, http://www.cds.caltech.edu/erato/sbml/docs/
[4] SBW/SBML, http://www.sbw-sbml.org/
[5] BSML, http://www.bsml.org
[6] Interoperable Informatics Infrastructure Consortium (I3C), http://www.i3c.org/
[7] Reger, W. Silverman, E. Shapiro, "Representation and Simulation of biochemical processes using the π-calculus process algebra", in R. B. Altman et al. (Eds.), Pacific Symposium on Biocomputing, pp 459-470.
[8] Usenet newsgroup `comp.compilers`, "Catalog of Compiler Construction Products".
[9] N. Chomsky, "Aspects of the Theory of Syntax", Cambridge, MIT Press, 1965.

Evolution as Design Engineer

David L. Dill[1,2] and Patrick Lincoln[2⋆]

[1] Computer Science Department, Stanford University, Stanford, CA 94305, USA
dill@cs.stanford.edu
[2] SRI International
333 Ravenswood Avenue, Menlo Park, CA 94025, USA
{dill,lincoln}@csl.sri.com

Abstract. Exponential growth in digital information gathering, storage, and processing capabilities for biological data herald a new age of biology, when entire biological systems can potentially be understood. In order to facilitate human understanding of electronic components, the electronic design automation (EDA) industry has developed clean abstractions and design rules, enabling pervasive design reuse. We conjecture that design reuse arises as an emergent property in biological systems driven by the pressure for efficiency in evolution. Design reuse requires clean abstractions and interfaces, and these fortuitously provide an opportunity for human understanding and exploitation with computational toolsets. We draw some conclusions about future systems biology toolsets that might be inspired by EDA tools.

1 Introduction

Biological knowledge is exploding. At some point, information about individual genes, proteins, reactions, and other puzzle pieces will have to be assembled into larger pictures. But the higher-level concepts required to guide such assemblies are unknown. Indeed, it is not even known whether they exist.

As a system of many interacting components grows larger and more complex, there comes a time where the properties that emerge from interactions among the components are more important than the components themselves. At that point, understanding the system requires new models that capture this emergent behavior. Even a complete understanding of the components and the mechanisms by which they interact fails to give much insight into the behavior of the whole system. Like a jigsaw puzzle, the major barrier to insight is not understanding the pieces, but how they fit together to yield a whole picture.

In this position paper, we look at another endeavor that has required coping with complex systems, to see whether it is a reasonable model for what could be done in systems biology: the design of electronic systems. Electronic systems, especially digital systems, are arguably the most complex artifacts in existence. Their design depends on a wide variety of extremely sophisticated "electronic

⋆ Partially supported by DARPA contracts DE-AC03-765F00098 and 9N66001-00-C-801

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 202–206, 2003.

design automation" (EDA) software tools, which support almost all aspects of the design process. As of this writing, the commercial EDA industry has revenues of about $4.5 billion/year.

Is it possible that, within the forseeable future, a similar array of tools would be available for biology?

2 Models and Levels of Abstraction in Digital Design

Electronic design relies heavily on a hierarchy of models, which must be defined precisely to be processed by software tools. The models range from the very detailed, such as predicting the electronic properties of silicon when diffused with doping materials under different conditions, to very abstract, such as *system level descriptions*, which may be in conventional programming languages or in special languages.

Many different levels of abstraction are needed because of the inherent trade-off between accuracy and efficiency. Concrete models are needed when the details they represent are essential. More abstract models are needed for dealing with larger designs, or when the details are not available. In EDA, tools that work on more abstract models typically run millions of times faster than tools working on more concrete models.

Is there any reason to believe that biological mechanism implement clean higher-level abstractions which can later be exploited by software tools? At first glance, there would seem to be cause for pessimism. Engineering abstractions are contrived to be clean and understandable. Why would evolution reward understandable abstractions?

3 Design Reuse is Advantageous

Electronic design and biology share a powerful incentive for the invention of abstractions: *improved productivity through design reuse.*

Economic competition drives continuous increases in productivity in electronic design. One of the keys to designer productivity is to re-use designs, possibly with modifications. Design reuse occurs in many forms. Useful types of components are invented, along with good ways to design them. This knowledge then diffuses through publication, electronic interchange, and informal means (engineers moving from project to project). After some time, the best ones become ubiquitous. Indeed, implementations of standard components are commercially available in the form of "component libraries" and "reusable IP (intellectual property)." There are now vendors of components as large as whole microprocessors for use in large systems on a single chip.

Viewed from the perspective of design reuse, electronic design and biology no longer seem so divergent. The driving force for innovation in organisms is competition. There are great rewards for rapid improvement and adaptation to changed circumstances, so design reuse is strongly encouraged for the same reasons as

in electronic design. Biological design reuse occurs in many different ways. Organisms can copy and adapt successful components to perform other functions through various gene copying events, supported by components and processes that strongly favor such wholesale copying over completely random generation of new material. For example, bacteria are able to share genetic material containing designs of whole working components through sex pili, and cooperating unicellular eukaryotes and all higher organisms can perform sexual reproduction, yielding offspring with genetic material copied from multiple parent organisms.

Design reuse depends on the use of standard interfaces, which establish agreed-upon rules for the interaction of components. With standard interfaces, it becomes economically feasible to connect components from different sources, without expensive custom interface circuitry. In addition, standard interfaces greatly increase flexibility in system design. If an improved implementation of a component becomes available, or a different combination of features is required, a new component can be "swapped in" to a system without other modifications.

Since reuse of good design is advantageous, there are strong rewards for conformance to standard interfaces. Initially, design reuse favors standard interfaces because the standards are part of the design being reused. But there is evolutionary pressure to *maintain* conformance with a standard, since, as in electronic design, it facilitates the interoperation of components from diverse sources and promotes incremental upgrades of components without re-implementing and re-optimizing entire systems.

An obvious example of a standard in biology is the genetic code, which was just one of many ways to satisfy the basic engineering constraints of information storage, duplication, and use – and yet a single standard is universal. The genetic code is just part of a collection of standards for transcription, translation, signalling, and so on. The genetic code offers interoperability in the extreme, as retroviruses and molecular biologists have exploited. It promotes reuse of genes and the proteins they code for within the same species by modification, as well as exchange of genes between members of the same species or with other species. Other biological standards exist, some less universal, some more, such as phospholipid bilayer membranes, receptor kinase signaling, ubiquitination / proteolysis of proteins, and adesnoside triphosphate as an energy carrier in living cells.

Here is the crux of our the argument: *good standard interfaces lead to clean abstractions.* A good standard interface should maximize the flexibility in the implementation. This is achieved by minimizing dependencies of the external interface on unnecessary implementation details of the components. In other words, a good interface is *abstract.*

Let's look at an important example from electronic design: the Boolean gate. The Boolean gate has a very useful interface that can be implemented in a wide variety of technologies. There are two abstract values (0 and 1). Certain voltages are interpreted as 0, 1 (or neither) based on whether they are below or above certain thresholds. When the outputs of a gate are connected to the inputs of others, the combined circuit will act like a more complex Boolean gate if their

logic thresholds are compatible and certain other rules are followed (e.g., the outputs are not connected to too many inputs). The Boolean interface is very flexible, because it is not particularly critical what the signals do if they are on the 0 or 1 sides of their voltage threshholds, or even what they do when switching between these threshholds.

We conjecture that clean abstractions arise naturally in living systems, driven by the efficiency of design reuse. Standard interfaces that promote maximum reuse by hiding implementation details are the most advantageous to an organism. But such interfaces also embody good abstractions, fortuitously promote human understandability and enable the development of useful software tools. If this is true, the set of biological abstractions should grow well past those already found by biologists over the last few hundred years.

4 Tools

A necessary condition for the use of software tools for large-scale systems is the existence of powerful and relatively clean high-level abstractions. We speculate that organisms exhibit these abstractions because it is evolutionarily advantageous, but we wish to go beyond Herbert Simon's suggestion of "nearly decomposeable systems", and note that that these abstractions can be exploited to build software tools for comprehending and designing biological systems. We anticipate that well-designed tools tuned for this application will have as big an impact on biology as the analogous tools did in the electronics world. But what kinds of tools are these?

There is no clear taxonomy of EDA tools, but some general statements can be made. The concept of a executing a description (called *simulation*) seems to be almost universal. Almost any kind of dynamic behavior can be simulated (indeed, the ability to simulate can be considered a test of whether an abstraction of dynamic behavior is actually well understood). Furthermore, simulators are extremely useful in practice, since they can show possible behaviors of a system which may be difficult to observe in the system itself, because the systems is unavailable, or costly to exercise, or because the inputs or environmental conditions are difficult to observe, or because the behavior itself is difficult to observe. For these reasons, simulation is already extensively used in biology, from the stochastic behavior of individual molecules to the behavior of populations of organisms.

There are more sophisticated and difficult kinds of behavioral analysis. Roughly speaking, the basic idea in these techniques is to search *all possible behaviors*. Tools that do this in EDA including *logical equivalence checking*, which compares two digital circuits to see if they have implement the same function, and *model checkers,* which test whether a dynamic property holds for all possible executions of a circuit. The computational problems in such tools are very difficult. Yet, in those cases where they work well, they are extremely useful because the give universal answers which cannot be obtained by other means. There are

many cases in biology when it would be useful to know whether something can *ever* happen or whether a property is *always true*.

One possible objection to this is that many biological properties are inherently probabilistic. However, one can represent random processes as being deterministic, in some cases, and analyze them as such. Alternatively, a "property" can be regarded as a probabilistic statement, e.g., "The probability of going into lysis is .15."

Other important classes of tools in EDA are better suited to designing, not comprehending, systems. Optimization tools exist at many levels of abstraction, which transform circuits into equivalent circuits that perform better according to some metric (e.g., they are faster or smaller). Optimization techniques vary widely, depending on the abstraction and the application (e.g., using Lagrange multipliers for transistor sizing vs. performing equivalence preserving Boolean transformations at the gate level). There are also *synthesis* tools, which translate a higher-level specification into an implementation (synthesis tools usually perform optimization as well). For example, tools to translate hardware description languages (which look a lot like programs) into gate level tools have revolutionized hardware design.

At first glance, it does not seem that optimization and synthesis tools would be very useful for biology, except perhaps for synthetic biology. But such tools can still be helpful in cases of incomplete knowledge about a system. If we assume that evolution has optimized a system to perform a particular function or make the best use of certain resources, optimization and synthesis tools can be used to find plausible ways of achieving the same goals. Such an approach can find upper or lower bounds on the possible behavior, generate hypotheses about possible implementation alternatives, or, in cases where there is a unique way of satisfying constraints, extrapolate what must actually be happening in the organism. To be used in this way, it would be best if such tools could take as input complex constraints, including partial designs that are not to be changed (representing what is known *a priori* about the mechanism), and optimize or synthesize within the degrees of freedom they are allowed. In EDA, there are frequently demands for tools with these capabilities, since designers want to hand-optimize critical parts of a design or specify that existing parts of a design do not change, while allowing a tool to optimize other parts.

5 Conclusions

We have speculated that software tools will be very important in biology, and that there will be a much wider range of tools working at higher levels of abstraction than at present. There is much to be learned about the use of software tools from electronic design. We have also argued that living systems are not necessarily so different from engineered systems, because both rely on design reuse in response to competitive pressure. Effective design reuse requires standard interfaces to facilitate interoperation, which leads to natural abstractions. These abstractions will be serve as the foundations for software tools.

Thanks to Harley McAdams, Keith Laderoute, Raymonde Guindon, Carolyn Talcott, M.G. Sriram, and Adam Arkin for helpful discussions.

Inference, Modeling and Simulation of Gene Networks

Satoru Miyano

Human Genome Center, Institute of Medical Science, University of Tokyo
4-6-1 Shirokane-dai, Minato-ku, Tokyo 108-8639, Japan
miyano@ims.u-tokyo.ac.jp

Abstract. Gene networks play a central role in systems biology. This paper overviews a computational strategy focused on gene network analysis by reviewing our recent contributions.

1 Introduction

Systems biology may be soundly explored by development of computational tools and capabilities which enable us to understand complex biological systems. Strongly anticipated matters for systems biology are scientific contributions that also create practical benefits such as biomedical applications, solutions for environmental problems, etc. Thus, computational challenges in systems biology must be highly motivated with this direction and should not be regarded as yet another applications of techniques from computer science. This paper overviews our computational strategy for systems biology and indicates problems and further challenges.

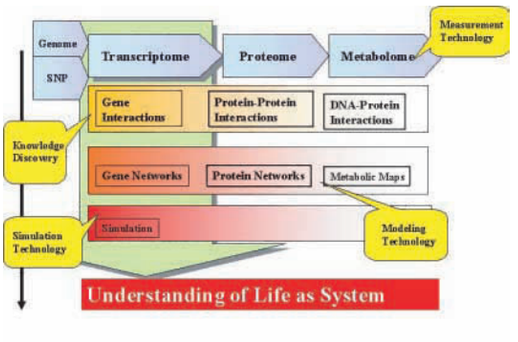


Fig. 1. Data and computational issues

Fig. 1 shows biological data production and computational topics towards understanding of biological systems, where we have taken two approaches. One is computational inference of gene networks from gene expression profile data obtained from various perturbations such as gene disruptions, shocks, etc. We have considered three gene network models for gene network inference. The other is computational modeling and simulation of biological systems. We have developed a tool Genomic Object Net [16,17,22] so that it would be a platform with which

C. Priami (Ed.): CMSB 2003, LNCS 2602, pp. 207–211, 2003.

biological scientists can comfortably model and simulate dynamic causal interactions and processes in the cell such as gene regulations, metabolic pathways, and signal transduction cascades. An application of computational methods in systems biology is also shown. Recently, we have succeeded in discovering a drug target gene by analyzing gene networks constructed from gene expression profile data based on gene disruptions and drug doses. In Section 4, we sketch how this was achieved with gene network inference methods and laboratory works.

2 Gene Network Inference from Microarray Data

With the advent of the microarray technology, a large volume of genome-wide gene expression profile data have been produced under various experimental conditions such as gene disruptions, gene overexpressions, shocks, cancer cells, etc. Along with this new data production, there have been considerable attempts to infer gene networks from such gene expression profile data. Gene network models may be roughly classified into three models; Boolean network model, model defined as a system of ordinary differential equations, and statistical network model.

In Boolean network model for gene regulatory network, numerical gene expression profile data are discretized into "ON" and "OFF" by using thresholds together with biological knowledge on genes. This model is suited for modeling qualitative relations between genes and allows mathematical and algorithmic analysis. For example, the complexity of experiments (microarray data) required for identify Boolean networks has been investigated [1,2,3,14]. A practical method is also devised in [15] for gene disruption based gene expression profile data. It is important to fill up the gap between theory and practice.

Bayesian network model was first applied to gene network modeling by Friedman *et al.* [7]. They used a discrete model where gene expression levels are categorized into $+1$, 0, and -1 and the multinomial distribution is used. Inspired by their success, we then developed a method which can analyze the continuous data and automatically detect linear and even nonlinear relationships between genes [8]. We employed nonparametric regression for capturing nonlinear relationships between genes and derive a new criterion called BNRC (Bayesian Network and Nonlinear Regression) for choosing the network in general situations. We also extended this method to Bayesian network and nonparametric heteroscedastic regression that can cope with variances in microarrays [9]. A drawback of Bayesian network model is that it does not allow any cycles while gene regulatory networks contain feedback loops. For this difficulty, we investigated dynamic Bayesian network and nonparametric regression for time-course gene expression data [13]. Unfortunately, the length of currently available microarray time-course data is not long enough. We applied these methods to the *S. cerevisiae* cell cycle data and cDNA microarray data of 120 disruptants (mostly transcription factors). The results showed us that we can infer relations between genes as networks very effectively.

Chen *et al.* [5] have considered modeling of both mRNA and protein concentrations with a system of linear differential equations. We have also devised a method [6] to infer a gene regulatory network, in terms of a linear system of differential equations, from time-course gene expression data by using Akaike's Information Criterion to determine which coefficients are nonzero.

These methods also heavily rely on characteristics of measurements, e.g. time-course data or not, disruption, overexpression, various shocks, hormone stimulus, drug response, developmental change, etc. None of them may be perfect by itself. Therefore, sophisticated combination of these methods for specific microarray measurements would be anticipated for creating a new strategy for gene network analysis. Another important challenge is to develop methods which employ sequence information on promoter regions, protein-protein interactions, protein-DNA interactions, and literature for refining gene networks. Furthermore, the gene network visualization is also a practically important issue for understanding the network because such method generates a network of several thousands genes that will fill up the computer display completely if it is layed out in a conventional way.

3 Modeling and Simulation

For understanding of biological systems, computational tools are strongly anticipated with which biological scientists (users) can comfortably model and simulate dynamic causal interactions and processes in the cell such as gene regulations, metabolic pathways, and signal transduction cascades. Genomic Object Net (GON) [22] was developed aiming at this important mission in systems biology.

There have been pioneering attempts and accumulations of knowledge for this direction, e.g. Gepasi [18], E-Cell [19], BioSPICE [20], etc. for simulation tools, and KEGG [11], BioCyc [12], etc. for biopathway databases. In appreciation of these efforts, the architecture of GON was designed so that users can get involved with modeling and simulation biologically intuitively with their profound knowledge and insights and can also be benefited from such biopathway databases. For modeling a biopathway in a mathematical way, GON employs the notion of *hybrid functional Petri net with extension* (HFPNe). HFPNe was defined for GON by enhancing some functions to hybrid Petri net [4] so that various aspects in biopathways can be intuitively modeled.

We consider that biological system modeling should be conducted by biological scientists because their minds are full of unpublished deep insights which are inevitable for right modeling. Therefore, any computational challenge for developing such modeling and simulation tools should take care of this aspect.

Another important issue is how to share such tools and databases. The idea of SBML [21] would be one solution. We have also made a step with GON towards this direction. KEGG and BioCyc compile a large number of static biopathway models. The benefits from the HFPNe architecture for biopathway modeling and the flexible features in the model editor of GON have opened

a way to recreate dynamic models from these databases. We have developed a tool which transforms biopathway models in KEGG and BioCyc to the GON XML files. Especially, all metabolic pathway models in KEGG are now ready for re-modeling and simulation with GON. This tool can also be extended to cope with another biopathway databases.

4 Appliction to Discovering Drug Target Gene

The field of drug discovery can be divided roughly into two fields of endeavor, namely target selection (biology) and drug design (chemistry). In the latter field, recently computational chemistry has been extensively used to develop rational methods of design of lead molecules using computational models as a guide. However, in the area of target selection, integrated iterative methods for discovery have been lacking. Systems biology has a chance to create a new paradigm for target selection by employing using computational modeling of gene networks. The following is our challenge for this direction.

We have recently discovered a new drug target gene by developing a method which employs gene networks inferred from cDNA microarray data [10]. An antifungal medicine is used as a drug. We have prepared two kinds of *S. cerevisiae* cDNA microarrays for constructing gene networks. One is gene expression profile data (X)obtained from 120 gene disruptions, where mostly transcription factors are disrupted, one for each microarray. The other is gene expression profile data (Y) obtained from expression experiments of several doses and time responses to the drug. The first step is to identify the genes directly affected by the drug. For this purpose, the most straightforward approach might be the fold-change analysis of the data set Y. However, in order to find genes directly affected by the drug, we took another method [15] based on Boolean network model which is more suited for inferring qualitative relations between genes. By regarding the drug as a "virtual gene", a directed acyclic graph with this virtual gene as the root was constructed from $X \cup Y$. From this graph, we can identify genes which may be directly affected by the drug. The second step is to find "druggable genes" that regulate the drug-affected genes most strongly from the upstream of the gene network. For this purpose, we employed a method based on Bayesian network model [8,9] to construct a gene network from the data set X. With this gene network, we could explore the gene network for the druggable genes related to the drug-affected genes very effectively. For this gene network exploration, we have also developed a gene network analysis tool called G.NET that provides an computational environment for various path searches among genes with annotated gene network visulization. This is the total system which made a discovery.

References

1. Akutsu, T., Maruyama, O., Kuhara, S., Miyano, S., Identification of genetic networks by strategic gene disruptions and gene overexpressions under a boolean model, *Theoretical Comuter Science*, in press. 208

2. Akutsu, T., Miyano, S., Kuhara, S., Identification of genetic networks from a small number of gene expression patterns under the Boolean network model, *Proc. Pacific Symposium on Biocomputing*, **4**, 17–28, 1999. 208
3. Akutsu, T., Miyano, S., Kuhara, S., Inferring qualitative relations in genetic networks and metabolic pathways, *Bioinformatics*, **16**, 727–734, 2000. 208
4. Alla, H., David, R., Continuous and hybrid Petri nets, *J. Circuits, Systems, and Computers*,**8**, 159–188, 1998. 209
5. Chen, T., He, H. L., Church, G. M., Modeling gene expression with differential equations, *Proc. Pac. Symposium on Biocomputing*, **4**, 29–40, 1999. 209
6. de Hoon, M. J. L., Imoto, S., Kobayashi, K., Ogasawara, N., Miyano, S., Inferring gene regulatory networks from time-ordered gene expression data of *Bacillus subtilis* using differential equations, *Pacific Symposium on Biocomputing*, **8**, in press, 2003. 209
7. Friedman, N., Linial, M., Nachman, I., and Pe'er, D., Using Bayesian networks to analyze expression data, *J. Comp. Biol.*, **7**, 601–620, 2000. 208
8. Imoto, S., Goto, T., Miyano, S., Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression, *Proc. Pacific Symposium on Biocomputing*, **7**, 175–186, 2002. 208, 210
9. Imoto, S., Kim, S., Goto, T., Aburatani, S., Tashiro, K., Kuhara, S., Miyano, S., Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network, *Proc. IEEE Computer Society Bioinformatics Conference*, 219–227, 2002. 208, 210
10. Imoto, S., Savoie, C. J., Aburatani, S., Kim, S., Tashiro, K., Kuhara, S., and Miyano, S., Use of gene networks for identifying and validating drug targets, submitted. 210
11. Kanehisa, M., Goto, S., KEGG: Kyoto encyclopedia of genes and genomes, *Nucleic Acids Res.*, **28**, 27–30, 2000. 209
12. Karp, P., Riley, M., Saier, M., Paulsen, I., Paley, S., Pellegrini-Toole, A., The Ecocyc database, *Nucleic Acids Res.*, **30**, 56–58, 2002. 209
13. Kim, S., Imoto, S., and Miyano, S., Dynamic Bayesian network and nonparametric regression, *Proc. Computational Methods in Systems Biology*, in press, 2003. 208
14. Liang, S., Fuhrman, S., Somogyi, R., REVEAL, a general reverse en-gineering algorithm for inference of genetic network architectures, *Proc. Pac. Symposium on Biocomputing*, **3**, 18–29, 1998. 208
15. Maki, Y., Tominaga, D., Okamoto, M., Watanabe, S., Eguchi, Y., Development of a system for the inference of large scale genetic networks, *Proc. Pacific Symposium on Biocomputing*, **6**, 446–458, 2001. 208, 210
16. Matsuno, H., Doi, A., Nagasaki, M., Miyano, S., Hybrid Petri net representation of gene regulatory network, *Pacific Symposium on Biocomputing*, **5**, 338–349, 2000. 207
17. Matsuno, H., Doi, A., Hirata, H., Miyano, S., XML documentation of biopathways and their simulations in Genomic Object Net, *Genome Informatics*, **12**, 54–62, 2001. 207
18. Mendes, P., GEPASI: a software for modeling the dynamics, steady states and control of biochemical and other systems, *Comput. Appl. Biosci.*, **9**, 563–571, 1993. 209
19. Tomita, M., Whole-cell simulation: a grand challenge of the 21st century, *Trends Biotechnol.*, **19**, 205–210, 2001. 209
20. http://www.biospice.org/. 209
21. http://www.cds.caltech.edu/erato/ 209
22. http://www.GenomicObject.Net/ 207, 209

Author Index