Artificial Intelligence in Biological Modelling

François Fages

AI in Biological Modelling

François Fages

Abstract Systems Biology aims at elucidating the high-level functions of the cell from their biochemical basis at the molecular level. A lot of work has been done for collecting genomic and post-genomic data, making them available in databases and ontologies, building dynamical models of cell metabolism, signalling, division cycle, apoptosis, and publishing them in model repositories. In this chapter we review different applications of AI to biological systems modelling. We focus on cell processes at the unicellular level which constitutes most of the work achieved in the last two decades in the domain of Systems Biology. We show how rule-based languages and logical methods have played an important role in the study of molecular interaction networks and of their emergent properties responsible for cell behaviours. In particular, we present some results obtained with SAT and Constraint Logic Programming solvers for the static analysis of large interaction networks, with Model-Checking and Evolutionary Algorithms for the analysis and synthesis of dynamical models, and with Machine Learning techniques for the current challenges of infering mechanistic models from temporal data and automating the design of biological experiments.

François Fages
Inria Saclay – Ile de France, 1 rue Honoré d'Estienne d'Orves, Campus de l'École Polytechnique
91120 Palaiseau, France, e-mail: Francois.Fages@inria.fr

Contents

AI in Biological Modelling 1
François Fages
1 Introduction 4
2 Modelling Biochemical Interaction Networks 6
2.1 Reaction Systems 6
2.2 Influence Systems 12
2.3 Logic Programming 15
3 Automated Reasoning on Model Structures 16
3.1 Petri Net Invariants 16
3.2 Graph Matching 18
4 Modelling Dynamical Behaviours 20
4.1 Propositional Temporal Logics 20
4.2 Quantitative First-Order Temporal Logics 22
5 Automated Reasoning on Model Dynamics 25
5.1 Symbolic Model-Checking of Biochemical Circuits 25
5.2 Parameter Sensitivity and Robustness Computation 26
5.3 Parameter Search with Temporal Logic Constraints 27
6 Learning Mechanistic Models from Temporal Data 28
6.1 Probably Approximatively Correct Learning 29
6.2 Answer Set Programming 31
6.3 Budgeted Learning 32
7 Perspectives 32
References 32

1 Introduction

"All life is problem solving", Karl Popper.

In the early history of Computer Science, the biological metaphor played an important role in the design of the first models of computation based on neural networks and finite state machines. The Boolean model of the behaviour of nervous systems given by McCulloch and Pitts in 1943 turned out to be the model of a finite state machine [80]. This model of events in nerve nets was reworked mathematically in the mid 50's by Kleene who created the theory of finite automata [75], and later on, by Von Neumann in the mid 60's with the theory of self-replicating automata [86].

In return for Biology, that logical formalism was applied in the early 70's by Glass and Kaufman [57] and Thomas [106, 107, 109, 108] to the analysis of Gene Networks and the prediction of cell qualitative behaviours. In particular, the existence of positive circuits in the influence graph of a gene network conjectured by Thomas and later proved in [94, 99, 104], to be a necessary condition for the existence of multiple steady states which interestingly explains cell differentiation for genetically identical cells [110, 85, 100]. Similarly, the existence of negative circuits is a necessary condition for genetic oscillations and homeostasis [102], Some sufficient conditions for multi-stability have also been given in Feinberg's Chemical Reaction Network Theory [48] and implemented in the early nineties in the "Kineticist's Workbench" at MIT AI lab [36].

Nowadays, with the progress made on SAT solving, Model-Checking and Constraint Logic Programming, the logical modelling of biological regulatory networks is particularly relevant to reasoning on cell processes, and not only on gene networks, but also on RNA and protein networks for the study of cell division cycle control [47, 111], cell signalling [60], and more generally for the study of interaction systems at different scales from unicellular to multicellular, tissues and ecosystems.

This research belongs to a multidisciplinary domain, called Systems Biology [68] which emerged at the end of the 90's with the end of the Human Genome Project, to launch a similar effort on post-genomic data (RNA and protein interactions) and the molecular interaction mechanisms that implement signalling modules and decision processes responsible for cell behaviours. A lot of work has been done for collecting genomic and post-genomic data, making them available in databases and ontologies [6, 72], building dynamical models of cell metabolism [64], signalling, division cycle, apoptosis, and publishing them in model repositories [88].

The biological data about cell processes are however more and more quantitative, and not only about the mean of cell populations, but also more precisely about single cells tracked over time under the microscope. The advances made in the last two decades in molecular biology with highthroughput technologies, have thus made crucial the need for automated reasoning tools to help

- analyzing both qualitative and quantitative data about the concentration of molecular compounds over time,

- aggregating knowledge on particular cell processes,
- building phenomenological and mechanistic models, either qualitative or quantitative,
- learning dynamical models from temporal data,
- designing biological experiments
- and automating those experiments.

It thus makes a lot of sense now to go beyond qualitative insights, toward quantitative predictions, by developing quantitative models in, either deterministic formalisms (e.g. Ordinary Differential Equations, ODE) or non-deterministic (e.g. Continuous-Time Markov Chains, CTMC), and calibrating models accurately according to experimental data. On this route, Quantitative Biology pushes the development of AI techniques for reasoning both qualitatively and quantitatively about analog and hybrid analog/digital systems, taking also into account continuous time, continuous concentration values and continuous control mechanisms,

In this chapter, we review some applications of AI techniques to biological systems modelling. We mainly focus on cell processes at the unicellular level which constitutes most of the work achieved in the last two decades in the domain of computational systems biology. We also focus on a *logical paradigm for systems biology* which makes the following identifications:

$$\begin{gathered}\text{biological model = transition system } K\\ \text{dynamical behavior specification = temporal logic formula } \phi\\ \text{model validation = model-checking } K,\ s\ \models ?\ \phi\\ \text{model reduction = submodel-checking } K'? \subset K,\ K',\ s\ \models\ \phi\\ \text{model prediction = valid formula enumeration } K,\ s\ \models\ \phi ?\\ \text{static experiment design = symbolic model-checking } K,\ s?\ \models\ \phi\\ \text{model inference = constraint solving } K?,\ s\ \models\ \phi\\ \text{dynamic experiment design = constraint solving } K?,\ s?\ \models\ \phi\end{gathered}$$

This approach allows us to link biological systems to formal transition systems (either discrete or continuous), and biological modelling to program verification and synthesis from behavioural specifications. This chapter is organized in that perspective. The next section reviews some formal languages for modelling biochemical interaction networks, namely reaction systems and influence systems, and their representation by logic programs. The following section presents the successful use of SAT and Constraint Logic Programming tools, for solving NP-hard static analysis problems on biological models, such as the detection of Petri Net invariants, and the detection of model reduction relationships within large model repositories, often with better performance than with dedicated tools. Section 4 reviews some temporal logic languages used for modelling the (imprecise) behaviour of biological systems, both qualitatively and quantitatively. Section 5 presents some model-checking methods and evolutionary algorithms for constraint reasoning on dynamical models and the crucial problem of parameter search in high dimension. Finally Section 6 is dedicated to Machine Learning methods for automating model building and biological experiment design, that probably constitutes the main challenge now in Computational Systems Biology, and an important promise of AI.

2 Modelling Biochemical Interaction Networks

The Systems Biology Markup Language (SBML) [67, 66] provides a common exchange format for modelling biochemical interaction systems using essentially reactions or influences, events, and various annotations for linking the objects to external databases and ontologies. SBML has made possible the exchange of models between modellers, and the building of model repositories such as BioModels [88][1] or KEGG [72]. BioModels currently contains 612 manually curated models, 873 non curated, and 150000 models imported from other pathway resources, including 2641 models of whole genome metabolisms. This flat list of models can be accessed through the Gene Ontology[2] which defines a set of concepts used to describe gene function, and relationships between these concepts. It classifies functions along three aspects:

- molecular function, i.e. molecular activities of gene products,
- cellular component where gene products are active,
- biological process pathways and larger processes made up of the activities of multiple gene products.

SBML is nowadays supported by a majority of modelling tools such as Copasi [120] or Biocham[3] [19] used in the examples below, and graphical editors such as Cell Designer [50]. In this section we present the basic formalisms of reaction and influence systems with some details, in order to explain in the following sections various automated reasoning tools that have been used to reason about them and build predictive models of biological processes.

2.1 Reaction Systems

2.1.1 Syntax

In this chapter, unless explicitly noted, we will denote by capital letters (e.g. S) sets or multisets, by bold letters (e.g., $\mathbf{x}$) vectors and by small roman or Greek letters elements of those sets or vectors (e.g. real numbers, functions). For a multiset M, $\mathrm{Set}(M)$ will denote the set obtained from the support of M, and brackets like $M(i)$ will denote the multiplicity in the multiset (usually the stoichiometry). $\geq$ will denote the pointwise order for vectors, multisets and sets (i.e. inclusion).

We give here general definitions for directed reactions with inhibitors [39]. A *reaction* over molecular species $S = \{x_1, \ldots, x_s\}$ is a quadruple (R, M, P, f), also noted below in Biocham syntax (`f for R / M => P`), where R is a multiset of *reactants*, M a set of *inhibitors*, P a multiset of *products*, all composed of elements of

[1] `http://biomodels.net`

[2] `http://geneontology.org`

[3] `http://lifeware.inria.fr/biocham`

S, and $f : \mathbb{R}^s \to \mathbb{R}$ is a mathematical function over molecular species concentrations, called the *rate function*. A *reaction system* $\mathscr{R}$ is a finite multiset of reactions.

It is worth noting that a molecular species in a reaction can be both a reactant and a product (i.e. a catalyst), or both a reactant and an inhibitor (e.g. Botts–Morales enzymes). Such molecular species are not distinguished in SBML and are both called reaction *modifiers*. Unlike SBML, we consider directed reactions only (reversible reactions being represented here by two reactions) and enforce the following compatibility conditions between the rate function and the structure of a reaction: a reaction (R, M, P, f) over molecular species $\{x_1, \ldots, x_s\}$ is *well formed* if the following conditions hold:

1. $f(x_1, \ldots, x_s)$ is a partially differentiable function, non-negative on $\mathbb{R}^s_+$;
2. $x_i \in R$ if and only if $\partial f / \partial x_i(\mathbf{x}) > 0$ for some value $\mathbf{x} \in \mathbb{R}^s_+$;
3. $x_i \in M$ if and only if $\partial f / \partial x_i(\mathbf{x}) < 0$ for some value $\mathbf{x} \in \mathbb{R}^s_+$.

A reaction system is well formed if all its reactions are well formed. This is the case for instance of reaction systems with mass action law kinetics which take as rate functions the product of the concentration of the reactants with some constant rate parameter.

Example 1. The classical prey-predator model of Lotka–Volterra can be represented by the following well-formed reaction system with mass action law kinetics, between a proliferating prey A and a predator B:

```
k1*A for A => 2*A.
k2*A*B for A+B => 2*B.
k3*B for B => _.
```

$k1$ is the birth rate constant of the prey, $k2$ the rate constant for the consumption of the prey by the predator, and $k3$ the predator death rate constant. Note that in this example, the reactions have no inhibitors. If the prey A were competing with another species C for its nutrients for instance, this could be represented with birth reactions with inhibitors as follows:

```
k2*A/(k4+C) for A / C => 2*A.
k5*C/(k6+A) for C / A => 2*C.
```

2.1.2 Hierarchy of Semantics

The dynamics of a reaction system $\mathscr{R}$ can be defined either qualitatively or quantitively in several formalisms. However, those multiple interpretations can be formally related by abstraction relationships in the framework of abstract interpretation [30] to form a hierarchy of semantics corresponding to different abstraction levels [43].

The *differential semantics* associates a time varying concentration to each molecular species, and an Ordinary Differential Equation (ODE) system to the reactions, by summing for each molecular variable the rate functions multiplied by the stochiometric coefficients of the reactions that modify it, i.e. for $1 \leq j \leq s$

$$\frac{dx_j}{dt} = \sum_{(R_i,M_i,P_i,f_i)\in\mathscr{R}} (P_i(j) - R_i(j)) \times f_i$$

It is worth noting that in this interpretation, the inhibitors are supposed to decrease the reaction rate, but do not prevent the reaction to proceed.

In Example 1, we get the classical Lotka–Volterra equations

$$dB/dt = k1 * A * B - k3 * B$$

$$dA/dt = k2 * A - k1 * A * B$$

and the well-known oscillations between the concentrations of preys and predators, as shown in Figure 1 left.

The *stochastic semantics* associates to each molecule its discrete quantity, and to reactions a transition relation $\longrightarrow_S$ between discrete states, i.e. vectors $\mathbf{x}$ of $\mathbb{N}^s$. A transition is enabled in state $\mathbf{x}$ by a reaction $(R_i,M_i,P_i,f_i) \in \mathscr{R}$ if there are enough reactants, and the propensity is defined by evaluating the rate function f_i in $\mathbf{x}$:

$$\forall (R_i,M_i,P_i,f_i) \in \mathscr{R}, \mathbf{x} \longrightarrow_S \mathbf{x}' \text{ with propensity } f_i \text{ if } \mathbf{x} \geq R_i, \mathbf{x}' = \mathbf{x} - R_i + P_i$$

The transition probabilities between discrete states are obtained by normalization of the propensities of all the enabled reactions, and the time of the next reaction is given by the propensities with an exponential distribution [56]. It is worth noting that in this interpretation like in the differential semantics, the inhibitors decrease the reaction propensity but do not prevent the reaction to proceed.

In Example 1, the stochastic interpretation can exhibit some noisy oscillations similar to the differential interpretation, but also, and almost surely, the extinction of the predator as shown in Figure 1 right.

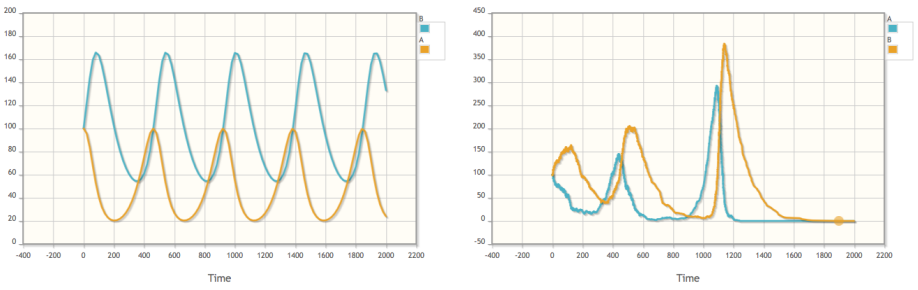


Fig. 1 ODE and stochastic simulation of Lotka Volterra prey-predator model.

The *discrete* or *Petri Net semantics* defines a similar transition relation $\longrightarrow_D$ over discrete states, but ignoring the rate functions. It is thus a trivial abstraction of the stochastic semantics by a forgetful functor, we have

$$\forall (R_i, M_i, P_i, f_i), \mathbf{x} \longrightarrow_D \mathbf{x}' \text{ if } \mathbf{x} \geq R_i, \mathbf{x}' = \mathbf{x} - R_i + P_i$$

The *Boolean semantics* is similar to the discrete semantics but on Boolean vectors x of $\mathbb{B}^s$, obtained by the "zero, non-zero" abstraction of integers $(> 0 : \mathbb{N} \to \mathbb{B}$. With this abstraction, when the number of a molecule is decremented, it can still remain present, or become absent. It is thus necessary to take into account all the possible complete consumption or not of the reactants in order to obtain a correct Boolean abstraction of the discrete and stochastic semantics [43]. The *Boolean transition system* $\longrightarrow_B$ is thus defined by considering all subsets of the set of reactants $\mathrm{Set}(R_i)$:

$$\forall (R_i, M_i, P_i, f_i), \forall C \in \mathscr{P}(\mathrm{Set}(R_i)), \mathbf{x} \longrightarrow_B \mathbf{x}' \text{ if } \mathbf{x} \supseteq \mathrm{Set}(R_i), \mathbf{x}' = \mathbf{x} \setminus C \cup \mathrm{Set}(P_i)$$

Intgerestingly, with these definitions, the last three semantics are related by successive Galois connections [43]. The set of Boolean traces is thus a correct abstraction of the stochastic traces for any rate functions, in the sense that the Boolean abstraction of the stochastic traces is contained in the set of traces of the Boolean semantics. This means that *if a behaviour is not possible in the Boolean semantics, it is not possible in the stochastic semantics* whatever the reaction rate functions are, and justifies the use of Boolean reasoning tools for many questions.

On the other hand, the differential semantics does not constitute an abstraction of the stochastic semantics, but provides, under strong assumptions, an approximation of the mean stochastic behavior, for instance when the number of each molecule tends to the infinity [56].

Example 2. In the Lotka-Volterra example, one can show that the extinction of the predator is almost sure in the stochastic semantics, whereas the differential semantics exhibits sustained oscillations (the condition on large numbers of molecules is clearly not satisfied). The Boolean semantics exhibits a set of possible Boolean behaviors which over-approximates the set of stochastic traces. Under this Boolean interpretation, one can observe either the stable existence of the prey (in case of extinction of the predator), the unstable existence of the predator (which can always disappear), or the disappearance of both of them, but not the extinction of the prey without the preceding extinction of the predator, nor any Boolean oscillation in absence here of synthesis reaction (e.g. migration). These properties can be directly expressed by Temporal Logic formulae described in Section 4.1, and automatically generated by the model-checking techniques described in Section 5.1 as follows:

```
biocham: present({A,B}).
biocham: generate_ctl_not.
reachable(stable(A))
reachable(steady(B))
reachable(stable(not A))
reachable(stable(not B))
checkpoint(B,not(A))
```

In presence of synthesis reactions such as protein synthesis, the discrepancies between the differential and stochastic interpretations may be less extreme. For these

reasons the differential semantics is widely used for quantitative biological modelling. The following example shows a typical case of biochemical reaction system for signalling, where the differential semantics approximates the mean stochastic behavior.

Example 3. The MAPK (Mitogen Activated Protein Kinase) biochemical reaction system is an extremely frequent signalling module that exists in several copies in eukaryote cells for different signalling tasks. This network is composed of three stages for a total of 30 reactions, where at each stage a protein gets phosphorylated once or twice, and under this phosphorylated form, catalyzes the phosphorylations of the next stage. The input `E1` of this signalling cascade, directly linked to the transmembrane receptor, phophorylates the kinase `KKK` of the first stage which then phosphorylates the kinase `KK` which itself phosphorylates the protein `K` to produce the output of the cascade `PP_K` which can migrate to the nucleus and modify gene transcription. Figure 2 shows the three levels structure of the reaction system.

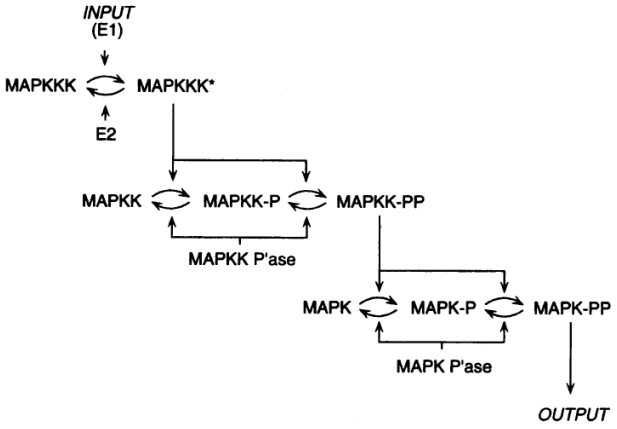


Fig. 2 MAPK signalling reaction network structure, with three levels of simple (at the first stage) and double (at the second and third stages) phosphorylations, with reverse dephosphorylation reactions catalyzed by phosphatases [65].

Figure 3 shows the ODE simulation and the dose-response diagram (i.e. `PP_K`, `PP_KK` and `P_K` at steady state versus `E1` varying in the range $[1e-6, 1e-4]$). This shows that MAPK acts as an analog-digital converter in the cell, with the stiffest response at the third level output [65].

It is worth noticing that the reaction inhibitors have not been used for the definition of the hierarchy of semantics in this section. The reason is that in the differential semantics an inhibitor decreases the rate of a reaction without preventing it completely from proceeding. One can also define a *Boolean semantics with negation* where the inhibitors of a reaction are seen as a conjunction of negative conditions

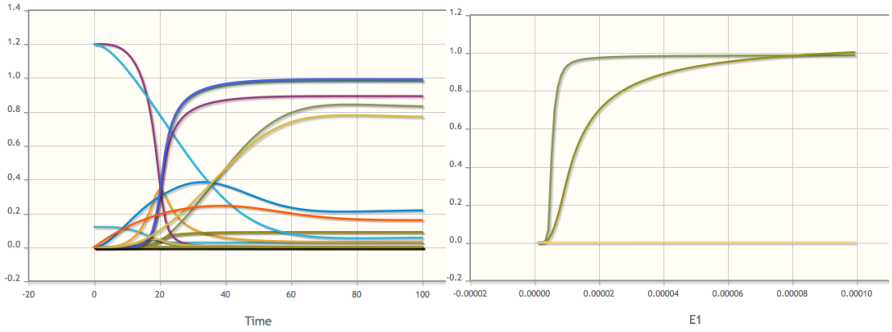


Fig. 3 ODE simulation of the MAPK signalling model, and dose-response diagram showing stiffer all-or-nothing response at lower levels of the cascade, revealing the analog-digital converter function of the MAPK circuit.

that must be satisfied for the reaction to proceed, by:

$$\forall (R_i, M_i, P_i, f_i) \forall C \in \mathscr{P}(\mathrm{Set}(R_i)) \mathbf{x} \longrightarrow_{BN} \mathbf{x}'$$
$$\text{if } \mathbf{x} \supseteq \mathrm{Set}(R_i), \mathbf{x} \cap M_i = \emptyset, \mathbf{x}' = \mathbf{x} \setminus C \cup \mathrm{Set}(P_i)$$

This interpretation is used in many systems, including Boolean Petri Nets and Rewriting Logic [37] yet with no connection to the other semantics.

2.1.3 Hybrid Discrete-Continuous Models

In the perspective of applying engineering methods to the analysis and control of biological systems, the issue of building complex models by composition of elementary models is a central one. Reaction systems can be formally composed by the multiset union of the reactions and interpreted in one common semantics, but there is also a need to compose models with different semantics. For instance, it makes a lot of sense to combine a differential model of protein activation for high numbers of molecules, with a Boolean or stochastic model of gene expression, since genes are in single or double copies in a cell.

The hierarchy of semantics of reaction systems provides a clear picture for studying the combination of several reaction models with different semantics and designing hybrid discrete/continuous digital/analog models of cell processes. A *hybrid model* is a model obtained by composition of models with heterogeneous semantics (continuous, stochastic, Boolean, etc.), and *hybrid simulation* is the topic of simulating such hybrid models. In [23], it is shown that the combination of events with kinetic reactions, as already present in SBML, provides enough expressive power for combining the discrete and continuous semantics of reaction systems. Such hybrid reaction systems can also be visualized as hybrid automata [63] in which there is a state with a particular ODE for each combination of the trigger values, and there is a transition from one state to another state when at least one trigger changes value from false to true in the source state.

Hybrid modelling is used in Systems Biology for reducing the complexity of modelling tasks [2, 15], e.g. in signalling [55] cell cycle control [101], gene regulation [79, 1], and most notably, for achieving whole cell simulation [73].

2.2 Influence Systems

Influence systems are a somewhat simpler formalism which is also widely used by modellers to merely describe the positive and negative influences between molecular species, without fixing their implementation by biochemical reactions. In particular, Thomas's regulatory networks form a particular class of Boolean influence systems, implemented in modelling tools such as GINsim [84], GNA [11] or Griffin [98]. It is also worth mentioning that influence systems with spatial information are developed in [22] as a formalism particularly suitable for describing natural algorithms in life sciences and social dynamics.

2.2.1 Syntax

In Thomas's framework, a regulatory network is defined by an influence graph given with a Boolean update function for each node. In order to define the other interpretations of an influence system, we shall distinguish here in the syntax the conjunctive conditions from the disjunctive conditions, with the convention that an influence on a target with several sources denotes a conjunctive condition, while different influences on a same target express a disjunction of conditions. Given a set $S = \{x_1, \ldots, x_s\}$ of molecular species, an *influence system I* is a set of quintuples (P, N, t, σ, f) called *influences*, where $P \subset S$ is called the *positive sources* of the influence, $N \subset S$ the *negative sources*, $t \in S$ is the *target*, *sign* $\sigma \in \{+, -\}$ is the sign of the influence, and f is a real-valued mathematical function of $\mathbb{R}^s$, called the *force* of the influence. The influences of sign $+$ are called *positive influences* and those of sign $-$, *negative influences*. They are noted in Biocham syntax (`f for R/M -> P`) and (`f for R/M -< P`) respectively.

Example 4. The prey-predator model of Lotka–Volterra of Example 1 can also be presented by the following system of four influences

```
k1*A*B for A,B -< A.
k1*A*B for A,B -> B.
k2*A for A -> A.
k3*B for B -< B.
```

The variant where a species `C` competes with `A` for nutrients gives an example of negative sources in the positive influences for proliferation:

```
k2*A/(k4+C) for A/C -> A.
k5*C/(k6+A) for C/A -> C.
```

The distinction between the positive and negative sources of an influence (either positive or negative) is similar to the distinction between the reactants and the

inhibitors of a reaction. An influence (P,N,t,σ,f) is *well formed* if the following conditions hold:

1. $f(x_1,\ldots,x_s)$ is a partially differentiable function, non-negative on $\mathbb{R}^s_+$;
2. $x_i \in P$ if and only if $\sigma = +$ (resp. $-$) and $\partial f/\partial x_i(\mathbf{x}) > 0$ (resp. < 0) for some value $\mathbf{x} \in \mathbb{R}^s_+$;
3. $x_i \in N$ if and only if $\sigma = +$ (resp. $-$) and $\partial f/\partial x_i(\mathbf{x}) < 0$ (resp. > 0) for some value $\mathbf{x} \in \mathbb{R}^s_+$;
4. $t \in P$ if $\sigma = -$.

2.2.2 Semantics

Given a set of species $S = \{x_1,\ldots,x_s\}$ and an influence system $\mathscr{I}$ over S, the *differential* semantics associates the following ODE system:

$$\frac{dx_k}{dt} = \sum_{(P_i,N_i,x_k,+,f_i)\in\mathscr{I}} f_i - \sum_{(P_j,N_j,x_k,-,f_j)\in\mathscr{I}} f_j$$

Intuitively, it adds up all the forces of the positive influences on x_k and subtracts all the forces of the negative influences on x_k in the derivative of x_k over time. For instance, in Example 4, one can check that we get the same ODEs as in Example 1.

It is worth noticing that the negative sources in a well-formed influence decrease the force of the influence but do not disable it. Consequently, the *stochastic* semantics of an influence system with forces, can be defined similarly to reaction systems, by a transition system, noted $\longrightarrow_S$, between discrete states, i.e. vectors $\mathbf{x}$ of $\mathbb{N}^s$, with the condition that the positive sources are present in sufficient number, without any condition on the negative sources:

$$\forall (P_i,N_i,A_i,\sigma_i,f_i), \mathbf{x} \longrightarrow_S^{f_i} \mathbf{x}' \text{ with propensity } f_i \text{ if } \mathbf{x} \geq P_i, \mathbf{x}' = \mathbf{x}\ \sigma_i\ A_i$$

Transition probabilities between discrete states are obtained through normalization of the propensities of all the enabled transitions, with time of next reaction [56]. As before, the *discrete* (or *Petri Net*) semantics simply ignores the forces:

$$\forall (P_i,N_i,A_i,\sigma_i,f_i), \mathbf{x} \longrightarrow_D \mathbf{x}' \text{ if } \mathbf{x} \geq P_i, \mathbf{x}' = \mathbf{x}\ \sigma_i\ A_i$$

and the *Boolean semantics* is defined on Boolean vectors x of $\mathbb{B}^s$, by the "zero, non-zero" abstraction. It is worth noticing that in this view, and similarly to reaction systems, the Boolean semantics associates two transitions to a negative influence:

$$\forall (P_i,N_i,A_i,+,f_i), \mathbf{x} \longrightarrow_B \mathbf{x}' \text{if } \mathbf{x} \geq P_i, \mathbf{x}' = \mathbf{x} + A_i$$

$$\forall (P_i,N_i,A_i,-,f_i), \mathbf{x} \longrightarrow_B \mathbf{x}' \text{if } \mathbf{x} \geq P_i, \mathbf{x}' = \mathbf{x} - A_i \text{ or } \mathbf{x}' = \mathbf{x}$$

That Boolean semantics is positive in the sense that it ignores the negative sources of an influence and contains no negation in the influence enabling condition.

In Lotka-Volterra Examples 1 and 4, the Boolean transitions are the same in this particular case, since there is no reaction that can produce a simultaneous change of the Boolean values of both the prey and the predator. However in general, reaction systems can produce simultaneous Boolean updates which cannot be represented by an influence system.

2.2.3 Expressive Power Compared to Reaction Systems

One can show that any (well-formed) influence system with forces can be represented by a (well-formed) reaction system, with the same Boolean, discrete, stochastic and differential semantics [40], i.e. an influence system can always be simulated by a reaction system for the different semantics. The converse does not hold for the discrete semantics. For instance for the Boolean semantics, the decomplexation reaction C=>$A+B$, has a transition from the state $(A,B,C)=(0,0,1)$ to $(1,1,0)$ which is obviously not possible in any influence system since only one variable can change in one transition. What is possible is to simulate a reaction system by an influence system which over-approximates its Boolean semantics.

However, the converse holds for the differential semantics, i.e. (well-formed) influence and reaction systems have the same expressive power [40]. This means that as far as the differential semantics is concerned, the influence systems have the same expressive power as reaction systems and there is no theoretical reason to develop a reaction model. This does not mean that there is a canonical reaction system associated with an influence system. Generally, different implementations with reactions are possible without changing the differential semantics. They represent extra information that is irrelevant to the analysis or simulation of the differential equations, but can lead to different stochastic simulations for instance.

2.2.4 Functional Boolean Semantics with Negation *à la* Thomas

The formalism of René Thomas [109] is a Boolean variant of influence systems which considers negative conditions and deterministic functional updates instead of relational updates. The success of this formalism lies, on the one hand, in the beautiful theory of necessary conditions for oscillations and multistability [94, 99] which explains for instance cell differentiation by the purely qualitative existence of a positive circuit in the influence graph of the system, and, on the other hand, for its widespread use for the logical modelling of a variety of cell processes beyond gene networks, such as cell cycle [46] cell signalling [60] or morphogenesis [58, 100].

In the *Boolean semantics with negation*, the negative sources are interpreted as negations in the enabling condition, as follows:

$$\forall (P_i,N_i,A_i,\sigma_i,f_i),\mathbf{x}\longrightarrow_{BN}\mathbf{x}'\text{if }\mathbf{x}\geq P_i,\ \mathbf{x}\cap N_i=\emptyset,\ \mathbf{x}'=\mathbf{x}\ \sigma_i\ A_i$$

This interpretation makes it possible to represent any *Boolean unitary transition system*, i.e. any transition system that updates at most one variable of $\mathbf{x}$ in each transition [40]. Furthermore, the Boolean semantics of Thomas's networks is *functional*, in the sense that the next Boolean state $\mathbf{x}'$ is defined by a Boolean function $\phi(\mathbf{x})$. The synchronous semantics is thus deterministic and the non-deterministic asynchronous semantics is obtained by interleaving, i.e. by considering all the possible transitions that change the Boolean value of one of the genes at a time.

For these reasons, a truly non-deterministic influence system such as

$$\{(A,\emptyset,B,+,f),\ (A,\emptyset,B,-,g)\}$$

(for which the transition relation is not a function) cannot be represented in Thomas's setting. This excludes self-loops in the state transition graph (on non-terminal states). This is even more striking in Thomas's multilevel setting, where the above system can (in the discrete semantics) have transitions from $(1,1)$ both to $(1,0)$ and to $(1,2)$. That would necessitate the corresponding logical parameter for B to be at the same time < 1 and > 1. Conversely, any Thomas's gene regulatory network can be represented by an influence system with the Boolean semantics with negation.

2.3 Logic Programming

The transition systems defined in Section 2.2.2 can be straightforwardly represented by Logic Programs (LP), and Constraint Logic Programs (CLP) for the quantitative semantics, where the states and the transition relation are defined by atoms, and the transition enabling conditions are defined by Horn clauses. This LP representation of reaction and influence systems suggests the use of a variety of LP tools for reasoning about them, such as deductive model-checking [32, 20], inductive logic programming [82, 44], and probabilistic logic programming [3].

In [69], it is shown how Thomas's Boolean networks can be directly represented by Normal Logic Programs (NLP), and how their trajectories and attractors can be computed with methods based on the similarity between the fixed points of Boolean networks and the immediate consequence operator T_P operator of NLPs. In particular, point attractors of both synchronous and asynchronous Boolean networks are characterized as the supported models of their associated logic programs so that SAT techniques can be applied to compute them.

Furthermore, NLPs provide a first-order representation which can be used to to describe the dynamics of influence systems on an infinite domains, such as the Petri Net semantics. In return for Logic Programming, this shows that logic programs that have cyclic attractors and are inconsistent under the supported or stable model semantics [38] can have meanings under the "attractor semantics" for NLPs.

3 Automated Reasoning on Model Structures

3.1 Petri Net Invariants

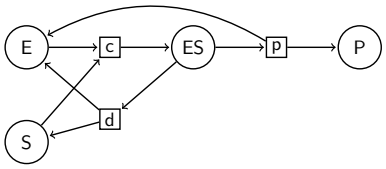
Beyond being a useful interpretation of reaction and influence systems in its own right, the Petri Net semantics provides interesting information on the differential and stochastic semantics of reaction systems. Petri nets have been introduced historically as a simple chemically-inspired formalism for describing and analyzing concurrent, asynchronous, non-deterministic, and possibly distributed, information processing systems [92]. The use of Petri nets for studying biochemical reaction systems, by mapping molecular species to places and reactions to transitions, was considered quite late in [93] for the analysis of metabolic networks. In this context, the traditional Petri net concepts of place-invariants (P-invariants), transition-invariants (T-invariants), siphons and traps have shown to have important applications especially in metabolism [71, 121, 119, 95, 77, 49, 64]. This motivated the search for efficient algorithms to scale-up to the size of biological models in model repositories, and revealed the astonishing performance of SAT and Constraint Logic Programming solvers which can outperform dedicated algorithms through a straigthforward Boolean or Finite Domain constraint modelling [103, 83].

A P-invariant is a multiset of places V (i.e. molecular species) such that the sum of the markings (i.e. numbers of molecules) remains constant for any scheduling of the transitions, i.e. $V.I = 0$ where I is the incidence matrix of the Petri net $I = \sum_i P_i - R_i$ with the notations of Section 2.1.2, i.e. I_{ij} is the number of arcs from transition i to place j, minus the number of arcs from place j to transition i. Such a P-invariant represents a structural conservation law between molecular species, and corresponds to a linear invariant in the ODE semantics of the reactions, i.e. a multiset of differential functions having their sum equal to zero which corresponds to a multiset of molecules whose sum of concentrations remains constant.

Example 5. The Michaelis-Menten enzymatic reaction system is composed of three reactions: one of complexation and one of decomplexation of the enzyme with the substrate, and one of transformation of the product with release of the enzyme. This simple system shown in Figure 4 has two minimal P-invariants which express the conservation of the enzyme in free and complexed form, and the conservation of the substrate in free, complexed and product form. These structural conservation laws can also be seen in the ODE semantics of the model by summing the corresponding differential functions.

The MAPK model of Example 3 uses Michaelis-Menten reactions for each phophorylation and dephosphorylation step. It has seven P-invariants, one for each kinase and phosphatase expressing its conservation among its different phosphorylated and complexed forms.

P-invariants can be computed either by standard Fourier-Motzkin elimination [26], or by linear algebra methods such as QR-factorization, Mixed Integer Pro-




```
E+S => ES.
ES => E+S.
ES => E+P.

biocham: search_conservations.
E+ES
ES+P+S

biocham: list_ode.
d(P)/dt=ES
d(E)/dt=2*ES-E*S
d(ES)/dt=E*S-2*ES
d(S)/dt=ES-E*S
```

Fig. 4 Michaelis-Menten system of three reactions representing the binding of an enzyme on its substrate and its transformation in a product, and computation of the two minimal P-invariants $\{E, ES\}$ and $\{ES, P, S\}$ corresponding to linear invariants of the differential semantics.

gramming, or more simply, and in fact more efficiently, by Constraint Logic Programming methods over finite domains, CLP(FD). The idea here is to solve the equation $V.I = 0$ in $V \in \mathbb{N}^s$ as a Constraint Satisfaction Problem (CSP) over finite domains by posting

- $V.R_i = V.P_i$ for each reaction i,
- $V.1 > 0$,

and by enumerating the values of V from low to high for finding P-invariants that are then checked for minimality by subsumption check [103].

Beyond its efficiency, the beauty of the CSP approach is that it generalizes straightforwardly to the computation of other invariants. T-invariants are the dual notion of P-invariants. A T-invariant is a multiset V of transitions such that $I.V = 0$, i.e. a multiset of reaction firing that leave invariant any marking. T-invariants revealed to be equivalent to the notion of extremal fluxes in metabolic networks [71, 121, 119], one of the main tools for analyzing and optimizing metabolic networks [95, 77, 49, 64]. Furhtermore in CSP, just by replacing equality constraints by inequalities, for instance $V.I \leq 0$ or $I.V \geq 0$, one can compute static subinvariants of markings or fluxes which can only grow or decrease during simulation [103]. To reduce the combinatorial complexity, recent results using SAT modulo theory (SMT) solver have shown further improvments for the enumeration of extremal flux modes [91].

Siphons and traps are other interesting Petri Net concepts. They denote meaningful pools of places that display a specific behaviour in the Petri net dynamics, and that guarantee some persistence properties, independently of the rate functions. A siphon is a set of places that, once unmarked, remains unmarked. A trap is a set of places that, once marked, can never loose all its tokens. These structural properties provide sufficient conditions for reachability (whether the system can produce a given protein or reach a given state from a given initial state) and liveness (deadlock freedom from a given initial state) properties in ordinary Petri nets. It has been shown that the problems of existence of a minimal siphon of a given cardinality, or

containing a given place, are NP-complete. In [83], a Boolean model is proposed to solve these minimal enumeration problems, either by calling a SAT solver iteratively, or by backtracking with a Constraint Logic Program (CLP) over Booleans. Interestingly, the SAT and CLP solvers both outperfom by one or two orders of magnitude the state-of-the-art algorithms from the Petri net community described in [27] for computing minimal sets of siphons and traps, that have already been shown to outperform Mixed Integer Linear Programs. On a benchmark of 345 biological models from the curated part of the BioModels repository [88], the Boolean method for enumerating the set of all minimal siphons takes a few seconds in MiniSAT. It also scales very well in the size of the net. The CLP(B) program also solves all but one instances of the benchmark, with a better performance than MiniSAT in average, but does not scale-up as well on the largest size Petri nets, such as for instance on Kohn's map with 509 species and 775 reactions. The efficiency of the MiniSAT and CLP(B) methods for enumerating in a few seconds the set of all solutions of an NP-complete problem for all, including large, instances of the BioModels benchmark is quite surprising. In [83], it is shown that the SAT phase transition threshold and complexity wall is traversed on those instances, but that the problem is tractable on graphs with bounded treewidth which seems to be the case of biochemical networks since most models in BioModels have a small treewidth less than 10. Still this does not explain why SAT and CLP solvers perform so well on this problem.

3.2 Graph Matching

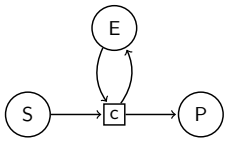
Models in Systems Biology are built with two somewhat contradictory perspectives:

- Models for aggregating knowledge on particular cell processes, in this perspective the more detailed the better;
- Models for answering particular questions on cell processes, in this perspective the more abstract the better, for getting rid of useless details that are not necessary to the questions at hand.

One way to reconcile these two perspectives is to relate models by model reduction relationships, that is currently not the case in model repositories. Model reduction is a central topic in dynamical systems theory, for reducing the complexity of detailed models, finding important parameters, and developing multi-scale models for instance. While perturbation theory is a standard mathematical tool to analyze the different time scales of a dynamical system, and decompose the system accordingly, Systems Biology needs novel methods for comparing and reducing models on a very large scale.

Graph matching techniques can be used to detect model reduction relationships between models within large repositories like BioModels. However the standard notion of subgraph isomorphism (SISO) for finding graph motifs is not adequate. For instance, the very basic reduction of Michaelis Menten which consists in re-

ducing the system of three reactions of Example 5 to one single catalytic reaction `E+S => E+P`, produces the graph



which is not isomorphic to a subgraph of the graph of Example 5. In this example, the reduced graph can be obtained from the source graph by a sequence of *delete and merge operations on species and reaction* vertices. These transformations can typically be justified in chemistry by considering for instance: (i) reaction deletions for slow reverse reactions, (ii) reaction mergings for reaction chains with a limiting reaction, (iii) molecular species deletions for species in excess and (iv) molecular mergings for quasi-steady state approximations.

This operational view of graph reduction by graph transformation operations is equivalent to the existence of a subgraph (corresponding to delete operations) epimorphism (i.e. surjective homomorphism, corresponding to merge operations) from a source graph to a reduced graph [52]. Formally, let G and G' denote graphs, with $G = (V,A)$ and $G' = (V',A')$, an *epimorphism* from G to G' is a surjective function $f : V \to V'$ such that

- for all $u,v \in V$, if $(u,v) \in A$, then $(f(u),f(v)) \in A'$ (graph homomorphism), and,
- for all $(u',v') \in A'$, there exists $(u,v) \in A$ such that $f(u) = u'$ and $f(v) = v'$ (surjectivity on arcs).

The subgraph of G induced by a subset of vertices $U \subseteq V$ of G, is $G_{\downarrow U} = (U, A \cap (U \times U))$. A *subgraph epimorphism* (SEPI) from G to G' is an epimorphism f from an induced subgraph G_0 of G to G'.

In Example 5, the two graphs of the Michaelis-Menten reduction, are related by a SEPI where the induced subgraph of the first graph is obtained by deleting the vertices ES and d, and where both reaction vertices c and p are mapped to the vertex c of the second graph.

Subgraph epimorphisms differ from subgraph isomorphisms by allowing merge operations in addition to delete operations. On undirected graphs, SEPIs differ from graph minors in several points: non adjacent vertices may be merged, merging adjacent vertices creates loops, and arcs cannot be deleted without deleting or merging vertices. Determining whether there exists a SEPI from a graph G to a graph G' is NP-complete [51]. Nevertheless a simple CLP(FD) program or SAT solver can solve this problem on all pairs of reaction graphs in the repository BioModels with just a few timeouts for some pairs of models.

Graph morphisms can be modelled by introducing one variable per node of the source graph, with the set of nodes of the target graph as integer domain. A variable assignment then represents a mapping from the source nodes to the target nodes. The morphism condition itself is written with a tabular constraint of CLP(FD) which forces a tuple of variables to take its value in a list of tuples of integers. The surjectivity property can be enforced by creating variables for the target arcs with the

set of source arcs as domain, and using the global constraint `all_different` of CLP(FD). Then, the enumeration on the target arc variables enforces surjectivity and the enumeration of node variables enforce the computation of a complete morphism [51].

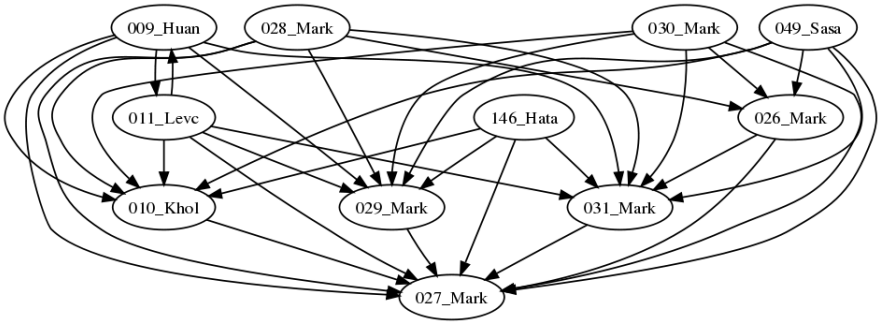


Fig. 5 Hierarchy of MAPK models in BioModels automatically constructed by SEPI matching [52]. Schoeberl's model 14 and Levchenko's model 19 are not represented here, they do not map each other but map to the other models.

Figure 5 shows the hierarchy of MAPK signalling models in BioModels that has been automatically reconstructed by graph matching, i.e. by computing SEPIs between all pairs of models. The arrows between models denote model reductions and double arrows denote reaction graph isomorphisms, e.g. between models 9 and 11 which differ just by molecule names and rate functions. These models have the same structure shown in Example 3. They reduce to model 10 which is also three level but without the reverse dephosphorylation reactions. It reduces also to models 29 and 27 which are one level models with ad without the dephosphorylation reactions.

4 Modelling Dynamical Behaviours

4.1 Propositional Temporal Logics

In the early days of computational Systems Biology, propositional temporal logic was soon proposed by computer scientists to formalize the Boolean properties of the behaviour of biochemical reaction systems [37, 20] and gene influence systems [16, 13]. In this approach, it is possible to evaluate qualitatively, at a high level of abstraction, what may or must happen in interaction networks of large size (e.g. of one thousand reactions and species), and also to compute the initial conditions that exhibit a particular behaviours. This can be achieved by using the powerful symbolic model-checking tools designed over the last decades for circuit and program verification [25, 24] using SAT solvers.

The *Computation Tree Logic* CTL* [25] is an extension of classical logic which allows reasoning on an infinite tree of Boolean state transitions from an initial state. It uses modal operators about branches (non-deterministic choices) and time (state transitions) to qualify where and when a proposition is true. Two path quantifiers A and E are thus introduced to handle non-determinism: $A\phi$ meaning that ϕ is true on all paths, and $E\phi$ that it is true on at least one path. Several time operators are introduced, $X\phi$ means that ϕ is true at the next state, $G\phi$ (globally) that ϕ is true in all future states, $F\phi$ (finally) that ϕ is true in some future state, $\phi\mathbf{U}\psi$ (until) that ϕ is always true before ψ becomes true, and $\phi\mathbf{R}\psi$ (release) that ψ is either globally true or always true up to the first occurrence of ψ included. Table 1 defines the truth value of a formula in a Kripke structure where the states are defined by Boolean variables. In this logic, $F\phi$ is equivalent to $\mathit{true}\mathbf{U}\phi$, $G\phi$ to $\phi\mathbf{R}\mathit{false}$, and we have the following duality properties: $\neg\mathbf{X}\phi = \mathbf{X}\neg\phi$, $\neg\mathbf{E}\phi = \mathbf{A}\neg\phi$, $\neg\mathbf{F}\phi = \mathbf{G}\neg\phi$, $\neg(\phi\mathbf{U}\psi) = \neg\phi\mathbf{R}\neg\psi$.

$s \models \alpha$	if α is a propositional formula true in the state s,
$s \models E\phi$	if there exists a path π starting from s s.t. $\pi \models \phi$,
$s \models A\phi$	if for all paths π starting from s, $\pi \models \phi$,
$\pi \models \neg\phi$	if $\pi \not\models \phi$,
$\pi \models \phi \wedge \psi$	if $\pi \models \phi$ and $\pi \models \psi$,
$\pi \models \phi \vee \psi$	if $\pi \models \phi$ or $\pi \models \psi$,
$\pi \models \phi \Rightarrow \psi$	if $\pi \models \neg\phi$ or $\pi \models \psi$,
$\pi \models \phi$	if $s \models \phi$ where s is the first state of π,
$\pi \models \mathbf{X}\phi$	if $\pi^1 \models \phi$,
$\pi \models \mathbf{F}\phi$	if $\exists k \geq 0$ s.t. $\pi^k \models \phi$,
$\pi \models \mathbf{G}\phi$	if $\forall k \geq 0$, $\pi^k \models \phi$,
$\pi \models \phi\mathbf{U}\psi$	if $\exists k \geq 0$ s.t. $\pi^k \models \psi$ and $\pi^j \models \phi\ \forall j\ 0 \leq j < k$.
$\pi \models \phi\mathbf{R}\psi$	if $\forall k \geq 0\ \pi^k \models \psi$ or $\exists j < k\ \pi^j \models \phi$

Table 1 Inductive definition of the truth value of a CTL* formula in a given state s or path π, for a Kripke structure K.

The LTL fragment of CTL* contains no path quantifier. An LTL formula is true if it is true on all paths. The CTL fragment of CTL* enforces that each temporal operator is preceded by a path operator, and each path operator is immediately followed by a temporal operator. In the context of computational Systems Biology, the following abbreviations for CTL formulae are particularly useful to analyze Boolean attractors [20, 111]:

- `reachable(P)` stands for $\mathbf{EF}(P)$;
- `steady(P)` stands for $\mathbf{EG}(P)$;
- `stable(P)` stands for $\mathbf{AG}(P)$;
- `checkpoint(Q,P)` stands for $\neg E(\neg Q\mathbf{U}P)$;
- `oscil(P)` stands for $\mathbf{AG}((\mathbf{EF}\ P) \wedge (\mathbf{EF}\ \neg P))$.

It is worth noting that that notion of checkpoint here is correlational but not necessarily causal. The last abbreviation is actually a necessary but not sufficient condition for oscillations. The correct formula for oscillations is indeed the

CTL* formula $\mathbf{EG}(\mathbf{F}P \wedge \mathbf{F}\neg P)$ which cannot be expressed in CTL. The formula `reachable(stable(P))` which is not expressible in LTL, expresses that the state denoted by formula P is a reachable stable state. In Example 2, these formulae are used as patterns to enumerate the interesting properties of the Boolean semantics of the prey-predator system.

4.2 Quantitative First-Order Temporal Logics

Generalizing temporal logic techniques to quantitative models can be done in two ways: either by discretizing the different regimes of the dynamics in piece-wise linear or affine models [12, 10, 70], or by taking a first-order version of temporal logic with constraints on concentrations, as query language for the numerical traces [5, 41, 35]. The first approach brings us back to symbolic propositional methods to analyze quantitative models [11]. In this section, we present the second approach.

The idea is to lift it to a first-order setting with numerical (linear) constraints over the reals, in order to express threshold and timing constraints and more complex constraints on the concentrations of the molecular compounds. For instance, the reachability of a threshold concentration for a molecule A can be expressed with the formula $\mathbf{F}(A > v)$ for some value or free variable v. Such formulae can then be interpreted on a finite numerical trace (extended with a loop on the last state) obtained either from a biological experiment, or from the numerical simulation of an ODE model, giving the concentrations of the molecules at discrete time points, e.g. Figure 6.

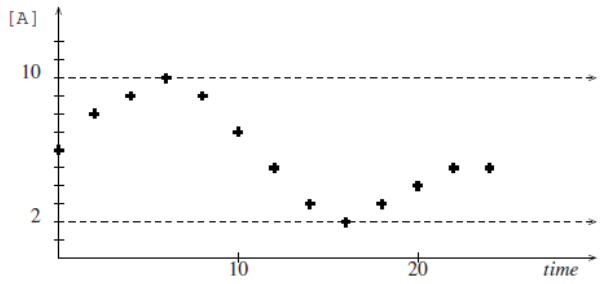


Fig. 6 Numerical trace depicting the time evolution of a protein concentration.

$$\phi ::= c \mid \phi \Rightarrow \psi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathbf{X}\phi \mid \mathbf{F}\phi \mid \mathbf{G}\phi \mid \phi \mathbf{U} \phi \mid \phi \mathbf{R} \phi$$

Table 2 Grammar of FO-LTL($\mathbb{R}_{\text{lin}}$) formulae where c denotes linear constraints over molecular concentrations, free variables and the time variable.

This is possible in the First-Order Linear Time Logic with linear constraints over the reals (FO-LTL($\mathbb{R}_{\text{lin}}$)) and in different variants like Signal Temporal Logic [35]. Table 2 summarizes the grammar of FO-LTL($\mathbb{R}_{\text{lin}}$) formulae. Timing constraints can be expressed with the time variable and free variables to relate the time of differents events. For instance, the formula $\mathbf{G}(Time \leq t_1 \Rightarrow [A] < 1 \wedge Time \geq t_2 \Rightarrow [A] > 10) \wedge (t_2 - t_1 < 60)$ expresses that the concentration of molecule A is always less than 1 up to some time t_1, always greater than 10 after time t_2, and the switching time between t_1 and t_2 is less than 60 units of time. A local maximum for molecule concentration A can be defined with the formula $\mathbf{F}(A \leq x \wedge \mathbf{X}(A = x \wedge \mathbf{X}A \leq x))$. This formula can be used to define oscillation properties, with period constraints defined as time separation constraints between the local maxima of the molecule, as well as phase constraints between different molecules [45].

The *validity domain* $\mathscr{D}_{(s_0,\ldots,s_n),\phi}$ of the free variables of an FO-LTL($\mathbb{R}_{\text{lin}}$) formula ϕ on a finite trace $(s_0,\ldots,s_n)$, can be computed by finite unions and intersections of polyhedra, by a simple extension of the model-checking algorithm to a constraint solving algorithm [41, 42], as follows:

- $\mathscr{D}_{(s_0,\ldots,s_n),\phi} = \mathscr{D}_{s_0,\phi}$,
- $\mathscr{D}_{s_i,c(\boldsymbol{x})} = \{\boldsymbol{v} \in \mathbb{R}^k \mid s_i \models c[\boldsymbol{v}/\boldsymbol{x}]\}$ for a constraint $c(\boldsymbol{x})$,
- $\mathscr{D}_{s_i,\phi\wedge\psi} = \mathscr{D}_{s_i,\phi} \cap \mathscr{D}_{s_i,\psi}$,
- $\mathscr{D}_{s_i,\phi\vee\psi} = \mathscr{D}_{s_i,\phi} \cup \mathscr{D}_{s_i,\psi}$,
- $\mathscr{D}_{s_i,\mathbf{X}\phi} = \mathscr{D}_{s_{i+1},\phi}$,
- $\mathscr{D}_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathscr{D}_{s_j,\phi}$,
- $\mathscr{D}_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathscr{D}_{s_j,\phi}$,
- $\mathscr{D}_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n} (\mathscr{D}_{s_j,\psi} \cap \bigcap_{k=i}^{j-1} \mathscr{D}_{s_k,\phi})$.

For instance, on the numerical trace of Figure 6, the validity domain, depicted in Figure 7, of the formula $\mathbf{F}(A \geq y_1 \wedge \mathbf{F}(A \leq y_2))$, where y_1 and y_2 are free variables, is $y_1 \leq 10 \wedge y_2 \geq 2$. This can be used for analyzing experimental traces, and extracting logical formulae from data time series.

However, for some important applications such as parameter search, sensitivity and robustness measures, presented in Section 5.2 the classical true/false valuation of a logical formula is not well suited. State-of-the-art continuous optimization algorithms such as evolutionary algorithms require a fitness function to measure progress towards satisfiability, i.e. they require to valuate TL formulae with a continuous satisfaction degree in the interval $[0,1]$.

A method based on variable abstraction is described in [96, 97] for computing the continuous satisfaction degree of an FO-LTL($\mathbb{R}_{\text{lin}}$) formula over a numerical trace. A closed formula, for instance

$$\phi_2 = \mathbf{F}(A \geq 7 \wedge \mathbf{F}(A \leq 0)),$$

is first abstracted in a formula with free variables by replacing constants with free variables, i.e.

$$\phi = \mathbf{F}(A \geq y_1 \wedge \mathbf{F}(A \leq y_2))$$

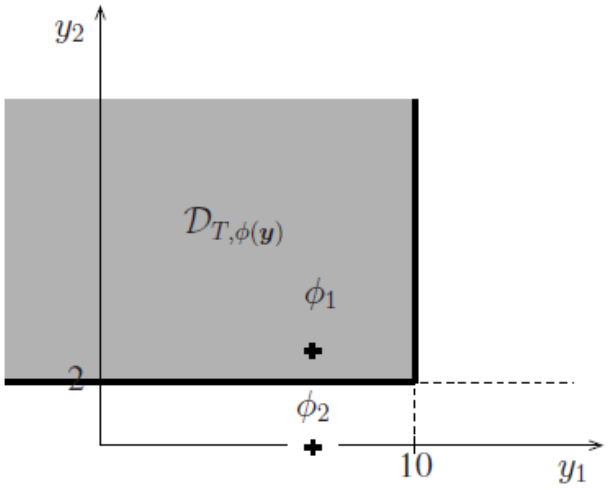


Fig. 7 Validity domain of the formula $\mathbf{F}(A \geq y_1 \wedge \mathbf{F}(A \leq y_2))$ on the trace of Figure 6. The two points correspond to the formulae $\phi_1 = \mathbf{F}(A \geq 7 \wedge \mathbf{F}(A \leq 3))$ (true) and $\phi_2 = \mathbf{F}(A \geq 7 \wedge \mathbf{F}(A \leq 0))$ (false) respectively.

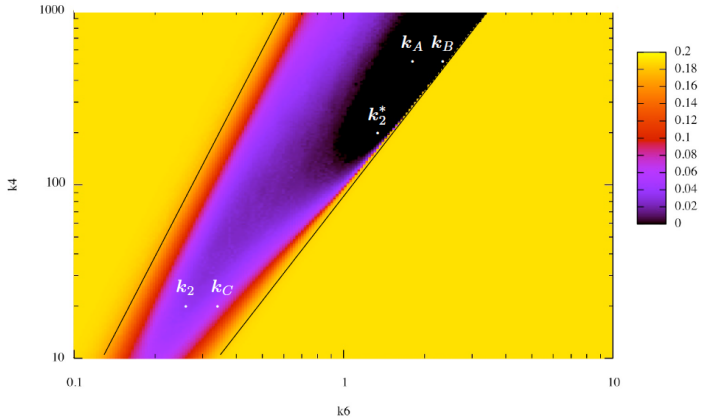


Fig. 8 Landscape of the continuous satisfaction degree of an oscillation property with amplitude constraint, on a color scale from yellow to black, as a function of two parameters in a quantitative model of the yeast cell cycle from [114]. The parameter sets $\boldsymbol{k}_A$, $\boldsymbol{k}_B$ and $\boldsymbol{k}_2^*$ satisfy the specification [97]. The parameter sets $\boldsymbol{k}_c$ and $\boldsymbol{k}_2$ violate the amplitude constraint. The non-yellow zone where there are oscillations is equivalently delimited by the bifurcation diagram considered in [114].

with the objective values 7 for y_1 and 0 for y_2. Then, the validity domain $\mathscr{D}_{T,\phi}$ of the formula ϕ on a trace T makes it possible to define the *violation degree* $vd(T,\phi,o)$ of the formula on T with objective o, simply as the distance between the validity domain and the objective point o, e.g. 2 in Figure 7. A *continuous satisfaction degree* in the interval $[0,1]$ can then be defined by normalization as the inverse of the violation degree d plus one, i.e. 1/3 in Figure 7:

$$sd(T,\phi,o) = \frac{1}{1+vd(T,\phi,o)}$$

In a model of the yeast cell cycle by Tyson [114], a FO-LTL($Rlin$) formula of oscillation with amplitude constraint produces the landscape of continuous satisfaction degree depicted in Figure 8 obtained by varying two parameters of the model. Such a landscape is compatible with bifurcation diagrams but is not limited in dimension and can be used for robustness measures and parameter search as shown in Sections 5.2 and 5.3.

5 Automated Reasoning on Model Dynamics

5.1 Symbolic Model-Checking of Biochemical Circuits

Regulatory, signalling and metabolic networks are very complex mechanisms which are far from being understood on a global scale. Data on the rate functions of the individual reactions are also rare and unreliable, making the building of quantitative models particularly challenging in many cases. In those situations, qualitative analyses can however be conducted in the Boolean semantics of the reactions, using the powerful model-checking tools developed for circuit and program verification [25] in the last decades.

Figure 9 reproduces Kohn's map of the mammalian cell cycle [76] using some graphical conventions introduced by K. Kohn to represent the different types of interactions (complexation, binding, phosphorylations, modifications, synthesis, etc.). This map has been transcribed in a reaction model of 732 reaction rules over 165 proteins and genes, and 532 variables taking into account the different forms of the molecular species [21]. The astronomical number of Boolean states in this system, 2^{532}, prevents the explicit representation of the state graph, however, a set of states in this space can be represented *symbolically* by a Boolean formula over 532 variables, and the transition relation by a Boolean formula over twice that number of variables. For instance the formula *false* represents the empty set, *true* the universe of all states, x the set of 2^{531} states where x is present, etc. The results reported in [21] showed the performance of the state-of-the-art symbolic model checker NuSMV [24] using the representation of Boolean formulae by ordered binary decision diagrams (OBDD), on this non standard transition system from biology. The compilation of the whole 732 reactions into Boolean formulae took 29 seconds, and simple reachability and oscillations properties could be checked in a few seconds. Furthermore in this example, the negative answer to the query concerning the oscillation of cyclin B revealed the omission of the synthesis of cyclin B in the map.

A symbolic model-checker can also compute the set of initial states, represented by a *boolean constraint*, for which a formula is true. This may suggest biological experiments to verify a CTL property predicted by the model, in particular conditions on the real biological object [16]. For instance, the checkpoints proved in a

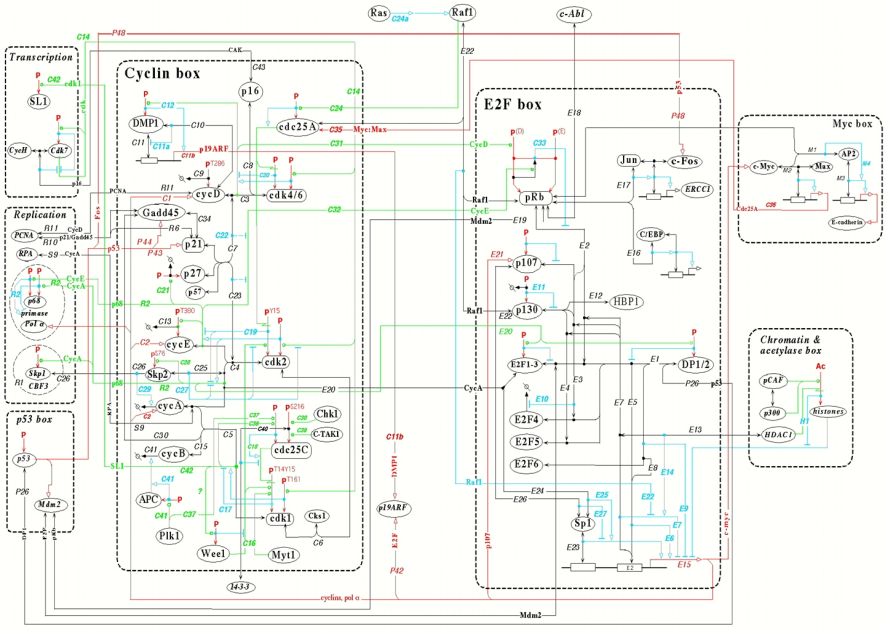


Fig. 9 Kohn's map of the mammalian cell cycle control [76].

model of the cell cycle, or of a signalling network, provide possible drug targets to block the cell cycle or a signalling cascade.

5.2 Parameter Sensitivity and Robustness Computation

In [74], Kitano gives a general definition of the *robustness* of a property ϕ in a system with respect to a set P of perturbations given with their probability distribution, as the mean functionality of the system with respect to ϕ under the perturbations., In the FO-LTL($\mathbb{R}_{\text{lin}}$) Temporal Logic framework, this definition instanciates straightforwardly to a computable notion of robustness of a property of a system, simply by taking the continuous satisfaction degree as functionality measure [96], i.e.

$$\mathscr{R}_{S,\phi,P} = \int_{p \in P} prob(p)\ sd(T_p, \phi)\ dp.$$

In a model, this mathematical definition of robustness can be evaluated by (i) sampling the perturbations according to their distribution, (ii) measuring the satisfaction degree of the property for each simulation of the perturbed model, and (iii) returning the average satisfaction degree.

This methodology has been used in [14] to design a robust switch satisfying some timing constraints implemented *in vivo* by synthetic biology means with an artificial cascade of gene inhibitions. Moreover, continuous parameter *sensitivity indices* computed in this approach determined the most important parameters for improving the robustness of the design with respect to the timing constraints, that unexpectedly appeared to be the degradation rate parameters.

On the quantitative model of the yeast cell cycle [114] and the oscillation with amplitude constraint depicted in Figure 8, the estimated degree of robustness for parameters $\boldsymbol{k}_A$, $\boldsymbol{k}_B$ and $\boldsymbol{k}_C$ are respectively 0.991, 0.917 and 0.932. This is consistent with the location of points $\boldsymbol{k}_A$, $\boldsymbol{k}_B$ and $\boldsymbol{k}_C$. Perturbations around point $\boldsymbol{k}_A$ have high probabilities of staying in the region satisfying the specification whereas perturbations around point $\boldsymbol{k}_B$ have high probabilities of moving the system to the region with no oscillation. $\boldsymbol{k}_C$ is more robust than $\boldsymbol{k}_B$ even though, as opposed to $\boldsymbol{k}_B$, its violation degree is non null. This is explained by the abrupt transition between oscillating and non oscillating regions near $\boldsymbol{k}_B$ compared to the smoother transition near $\boldsymbol{k}_C$.

5.3 Parameter Search with Temporal Logic Constraints

Probably the most central difficulty in quantitative systems biology, is that the kinetic parameter values of biochemical reactions are usually unknown, but are mandatory for building quantitative models. They must thus be estimated from the observation behaviour of the system under various conditions: gene knock-outs, differences of milieu, drugs, etc.

This problem amounts to solve the inverse problem of finding the parameter values of an ODE model for reproducing experimental curves, or, more appropriately, the relevant properties of the experimental curves. The formalization of those properties in quantitative temporal logic is particularly useful in biology where experimental data may be imprecise in nature, with important cell-to-cell variability, and irregular oscillation periods and phases. The continuous satisfaction degree of FO-LTL($\mathbb{R}_{\text{lin}}$) formulae provide the necessary objective or fitness function to apply black box optimization algorithms with the all bunch of meta-heuristics [105] such as Particle Swarm Optimization (PSO), Genetic Algorithms (GA), Neural Networks and portfolio algorithms for parameter estimation [8].

Of particular relevance in this context, is the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) of N. Hansen [61] which enjoy all desirable invariance properties with respect to scaling and symmetries. CMA-ES can be used with the satisfaction degree of an FO-LTL($\mathbb{R}_{\text{lin}}$) specification as fitness function, for searching kinetic parameter values, initial concentrations or control parameters [97]. On the quantitative model of the cell cycle of [114], Figure 8 depicts the landscape of the satisfaction degree of an oscillation property with amplitude constraint, as a function of two parameters of the model. This landscape is iteratively sampled by

CMA-ES meta-heuristics to find a path towards satisfaction, and optimize the model parameter values, for instance going from $\boldsymbol{k}_2$ to $\boldsymbol{k}_2^*$ in a few steps.

This strategy for optimizing parameters with respect to an FO-LTL($\mathbb{R}_{\text{lin}}$) specification makes it possible to solve a wide variety of problems in computational systems biology, for fitting models to experimental data in high dimension, up to 100 parameters. This methodology has been used in [62] to elucidate the complex quantitative dynamics of GPCR cell signalling networks, by revisiting the structure of the known reactions following the failure of CMA-ES to fit the FO-LTL($\mathbb{R}_{\text{lin}}$) properties of some mutants, making new biological hypotheses based on sensitivity analyses, and verifying them by new biological experiments.

In [112], it served to build a quantitative model of the cell cycle and the circadian clock explaining unexpected observations in embryonic fibroblasts, and make the prediction of an up-regulation of clock-gene Reverb-α during mitosis in those cells.

The same strategy for parameter optimization can also be used to compute control parameters in order to achieve a desired behaviour at the single cell or cell population levels. This has been shown for long-term model-based real-time control of gene expression in yeast cells using a microfluidic device in [115], and in the context of cancer chronotherapies, at the whole body scale, to couple models of the cell cycle, circadian clock, DNA repair system and drug metabolism, to optimize anti-cancer drug administration laws in [7, 31].

6 Learning Mechanistic Models from Temporal Data

Biological modelling is still an art which is currently limited in its applications by the number of available modellers. Automating the process of model building is thus a very desirable goal to attack new applications, develop patient-tailored therapeutics, and also design experiments that can now be largely automated at both the single cell and cell population levels, with a gain in both the quantification and the reliability of the observations.

Machine learning is revolutionarizing the statistical methods in biological data analytics, data classification and clustering, and for making predictions from static measurements. However, learning dynamical models from temporal data is more challenging, since it addresses hard issues for modeling time and causality [90]. There has been early work on the use of machine learning techniques, such as inductive logic programming [82] or heuristic breath-first tree search [116] combined with active learning in the vision of the "robot scientist", to infer gene functions [17], metabolic pathway descriptions [3, 4] or gene influence systems [16], or to revise a reaction model with respect to CTL properties [18]. A recent survey on probabilistic programming [59] highlighted the difficulties associated with modelling time, and concluded that existing frameworks are not sufficient in their treatment of dynamical systems. Since a few years, progress in those fields can be measured on public benchmarks of the "Dream Challenge" competition [81]. In this fastly mov-

ing field, we focus here on a general purpose framework for learning the structure of a mechanistic model.

6.1 Probably Approximatively Correct Learning

In his seminal paper on a theory of the learnable [117], Valiant questioned what can be learned from a computational viewpoint, and introduced the concept of probably approximate correct (PAC) learning, together with a general-purpose polynomial-time learning protocol. Beyond the learning algorithms that one can derive with this methodology, Valiant's theory of the learnable has profound implications on the nature of biological and cognitive processes, of collective and individual behaviors, and on the study of their evolution [118]. In this section, we simply recall the general theory of PAC learning, and illustrate it with the learning of Boolean gene networks from gene expression data.

The learning protocol for Boolean functions considers a finite set of Boolean variables $x_1,\ldots,x_s$. A vector is an assignment of the s variables to $\{0,1,*\}$ where the symbol $*$ denotes the undetermined. A vector is total if it contains no undetermined value. A Boolean function $F:\{0,1\}^s \to \{0,1\}$ assigns a Boolean value to each total vector. A Boolean concept $C:\{0,1,*\}^s \to \{0,1\}$ assigns similarly a Boolean value to non total vectors, with the following independence constraint: for any vector v and any total extension w of v (i.e. where the undetermined values in v are replaced by 0 or 1) we have $C(v)=C(w)$.

The PAC learning protocol considers a hidden Boolean function F, a class $\mathscr{M}$ of models to learn, $f(x_1,...,x_s)\in\{0,1,*\}$, a set of positive examples, i.e. a set of vectors v for which $F(v)=1$, and an arbitrary probability distribution D over this set for representing the relative frequency of the positive examples. The restriction to positive examples is for the sake of simplicity. The PAC learning protocol then allows for

- calls for positive examples, i.e. vectors v such that $F(v)=1$ given with probability $D(v)$,
- calls for oracle on some input v to know the value of $F(v)$

Example 6. This Boolean framework perfectly fits the Boolean semantics of Thomas's gene regulatory networks described in Section 2.2.4. Indeed in that formalism, each gene $x_1,\ldots,x_s$ is given with a Boolean function $F_{x_i}:\{0,1\}^s\to\{0,1\}$ which defines the activation update function of that gene according to the expression vector of the other genes in the different possible states. These Boolean functions are best represented by Boolean concepts in PAC terminology in order to make explicit the independent genes. Then, the problem of building such a Boolean model *à la* Thomas of gene activation is to give for each gene a Boolean transition function that is compatible with the observed temporal data of gene activation. It is worth noticing that the PAC learning protocol makes it possible to learn such Boolean models of gene regulation not only from a given finite set of positive gene activation observations,

but also from new biological experiments designed by the PAC learning algorithm itself through the queries to the oracle.

A class $\mathscr{M}$ of models is learnable in a given learning protocol, if there exists an algorithm $\mathscr{A}$ such that:

- $\mathscr{A}$ runs in polynomial time in s and h, the size of the models to learn,
- For all models f in $\mathscr{M}$, all vector distributions D on which f outputs 1, $\mathscr{A}$ deduces with probability $\geq 1-h-1$ a model g in $\mathscr{M}$ such that
  - $g(v) = 1$ implies $f(v) = 1$
  - $\sum_{v\ s.t.\ f(v)=1\ g(v)\neg=1} D(v) < h-1$

Interestingly, Valiant showed the learnability of some important classes of functions in this framework, in particular for Boolean formulae in conjunctive normal forms with at most k literals (k-CNF) and for monotone (i.e. negation free) Boolean formulae in disjunctive normal form (DNF). The computational complexity of the PAC learning algorithms for these classes of functions is expressed in terms of the function $L(h,S)$ defined as the smallest integer i such that in i independent Bernoulli trials, each with probability at least $h-1$ of success, the probability of having fewer than S successes is less than $h-1$. Interestingly, this function is quasi-linear in h and S, i.e. for all integers $S \geq 1$ and reals $h > 1$, $L(h,S) \leq 2h(S + log_e h)$.

First, for any k, the class of k-CNF formulae is learnable with an algorithm that uses $L(h,(2s)^{k+1})$ examples and no oracle [117]. The algorithm used in the proof proceeds as follows

1. Initialize g to the conjunction of all possible $(2s)^{k+1}$ disjunctions of at most k literals,
2. Call $L(h,(2t)k+1)$ positive examples v,
3. Delete all the disjunctions in g that do not contain a literal true in v.

Example 7. k-CNF formulae can be used to represent Thomas's gene regulatory network functions with some reasonable restrictions on their connectivity. In this case, the algorithm is repeated s times for learning each gene activation function. The initialization of the learned function g to the most constrained conjunction of all possible disjunctions leads to the learning of a minimal generalization of the positive examples in this representation.

Second, the class of monotone DNF formulae is also learnable with an algorithm that uses $L(h,d)$ examples and ds calls to the oracle, where d is the largest number of prime implicants in an equivalent prime DNF formula [117]. The algorithm is the following:

1. Initialize g with constant zero,
2. Do $L(h,d)$ calls to positive examples v,
3. If g is not implied by v, add the conjunction of determined literals that are essential to f which is determined by ds calls to the oracle.

Example 8. The (positive) Boolean semantics of biochemical influence systems described in Section 2.2.2 can be directly represented by the disjunction of the (positive) enabling conditions of each, either positive or negative, influence on a given target, i.e. by a monotone DNF formula for each activation or inhibition of each target. In the Lotka-Volterra influence system of Example 4, the algorithm above is thus expected to learn the structure of the influence system (without the stochiometry of course), from the observation that the prey can disappear only in presence of the predator while the predator can always disappear in presence or absence of the prey.

Example 9. Learning reaction models from observed transitions is much more tricky, since some reactions may change the Boolean value of several reactants or products in one single transition. Therefore, it is not only the activation and inhibition functions of each species which are to be learnt, but the update functions of pairs and triples of species if we restrict to elementary reactions with at most two reactants or products. In this case, the update functions can be represented by monotonic DNF formulae, since the (positive) Boolean semantics of a reaction system does not test the absence. Furthermore, one cannot expect to learn the structure of such a reaction network from the observation of the state transitions from one single initial state. The learning algorithms assumes that the positive examples of the state transition relation be distributed among the whole vector space. For instance, in the MAPK example 3, in addition to the initial state of the wild type organism where all the kinases and phosphatases are present, it is necessary to consider some mutated organisms, in which some kinases or phosphatases are absent, in order to gain information on the precise conditions of activation and deactivation of the different forms of the kinases. This strategy is essentially similar to what the biologists do to elucidate the structure of biological processes in a qualitative manner.

6.2 Answer Set Programming

Logic Programming, and especially *Answer Set Programming* (ASP), provide particularly efficient tools such as CLASP [53] to develop learning algorithms for Boolean models. They were applied in [54] to detect inconsistencies in large biological networks, and have been subsequentially applied to the inference of gene networks from gene expression data.

Interestingly, ASP has also been combined with CTL model-checking in [89] to learn mammalian signalling networks from time series data, and identify erroneous time-points in the data, a possibility not considered in the previous presentation of PAC learning.

6.3 Budgeted Learning

Budgeted learning extends active learning with a notion of cost for the calls to the oracle. The original motivation for the budgeted learning protocol came from medical applications in which the outcome of a treatment, drug trial, or control group is known, and the results of running medical tests are each available for a price [34]. In this context, multi-armed bandit methods [33] provide the best strategies. In [78], a bandit-based active learning algorithm is proposed for experiment design in dynamical system identification. These approaches are directly relevant to biological experiment design and modelling. They should gain importance in the forthcoming years with the increasing automation of biological experiments.

7 Perspectives

"What I cannot create, I do not understand", Richard Feynman.

Computer Science is born with the perspective of Artificial Intelligence, i.e. creating machines that reproduce human intelligence [113]. The application of Computer Science concepts and tools to the analysis of Biological Systems, beyond solving Bioinformatics combinatorial problems with AI techniques, provides a new perspective for Computation Science: Biology, i.e. understanding the living, how cells sense their environment and compute their decision, and beyond discribing natural biochemical interaction networks [9], understand their functions, evolution history and evolution capabilities [118].

Though one lesson of Computer Science was that analog computation does not scale up while digital computation does, the biological perspective provides a new impetus to analog computation and mixed analog/digital parallel computation. The concept of biochemical computation can now be experimented, either in Synthetic Biology, through the modification and reprogramming of living cells [87, 29], or in Synthetic Biochemistry, through the creation and programming of non-living microfluidic vesicles [28]. The social behaviors of cells and tissue homeostasis add one more dimension to the problem of designing useful computational devices at the microscale. These research fields provide numerous challenges to AI, both conceptual and algorithmic.

References

1. Ahmad, J., Bernot, G., Comet, J.P., Lime, D., Roux, O.: Hybrid modelling and dynamical analysis of gene regulatory networks with delays. ComplexUs **3**, 231–251 (2006)
2. Alur, R., Belta, C., Ivanicic, F., Kumar, V., Mintz, M., Pappas, G.J., Rubin, H., Schug, J.: Hybrid modeling and simulation of biomolecular networks. In: Proceedings of the 4th Inter-

national Workshop on Hybrid Systems: Computation and Control, HSCC'01, *Lecture Notes in Computer Science*, vol. 2034, pp. 19–32. Springer-Verlag, Rome, Italy (2001)
3. Angelopoulos, N., Muggleton, S.H.: Machine learning metabolic pathway descriptions using a probabilistic relational representation. Electronic Transactions in Artificial Intelligence **7**(9) (2002). Also in Proceedings of Machine Intelligence 19
4. Angelopoulos, N., Muggleton, S.H.: Slps for probabilistic pathways: Modeling and parameter estimation. Tech. Rep. TR 2002/12, Department of Computing, Imperial College, London, UK (2002)
5. Antoniotti, M., Policriti, A., Ugel, N., Mishra, B.: Model building and model checking for biochemical processes. Cell Biochemistry and Biophysics **38**, 271–286 (2003)
6. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. Nature Genetics **25**, 25–29 (2000)
7. Ballesta, A., Dulong, S., Abbara, C., Cohen, B., Okyar, A., Clairambault, J., Levi, F.: A combined experimental and mathematical approach for molecular-based optimization of irinotecan circadian delivery. PLOS Computational Biology **7**(9) (2011). DOI 10.1371/journal.pcbi.1002143
8. Banga, J.R.: Optimization in computational systems biology. BMC Syst Biol **2** (2008). DOI 10.1186/1752-0509-2-47.
9. Barabási, A.L.: Network Science. Cambridge University Press (2016)
10. Batt, G., Bergamini, D., de Jong, H., Garavel, H., Mateescu, R.: Model checking genetic regulatory networks using gna and cadp. In: Proceedings of the 11th International SPIN Workshop on Model Checking of Software SPIN'2004. Barcelona, Spain (2004)
11. Batt, G., Besson, B., Ciron, P., de Jong, H., Dumas, E., Geiselmann, J., Monte, R., Monteiro, P., Page, M., Rechenmann, F., Ropers, D.: Genetic Network Analyzer: a tool for the qualitative modeling and simulation of bacterial regulatory networks. In: Bacterial Molecular Networks, pp. 439–462. Springer (2012)
12. Batt, G., Page, M., Cantone, I., Goessler, G., Monteiro, P., de Jong, H.: Efficient parameter search for qualitative models of regulatory networks using symbolic model checking. Bioinformatics **26**(18), i603–i610 (2010)
13. Batt, G., Ropers, D., de Jong, H., Geiselmann, J., Mateescu, R., Page, M., Schneider, D.: Validation of qualitative models of genetic regulatory networks by model checking : Analysis of the nutritional stress response in *Escherichia coli*. Bioinformatics **21**(Suppl.1), i19–i28 (2005)
14. Batt, G., Yordanov, B., Weiss, R., Belta, C.: Robustness analysis and tuning of synthetic gene networks. Bioinformatics **23**(18), 2415–2422 (2007)
15. Berestovsky, N., Zhou, W., Nagrath, D., Nakhleh, L.: Modeling integrated cellular machinery using hybrid petri-boolean networks. PLoS Computational Biology **9**(11), 1003,306 (2013). DOI 10.1371/journal.pcbi.1003306
16. Bernot, G., Comet, J.P., Richard, A., Guespin, J.: A fruitful application of formal methods to biological regulatory networks: Extending Thomas' asynchronous logical approach with temporal logic. Journal of Theoretical Biology **229**(3), 339–347 (2004)
17. Bryant, C.H., Muggleton, S.H., Oliver, S.G., Kell, D.B., Reiser, P.G.K., King, R.D.: Combining inductive logic programming, active learning and robotics to discover the function of genes. Electronic Transactions in Artificial Intelligence **6**(12) (2001)
18. Calzone, L., Chabrier-Rivier, N., Fages, F., Soliman, S.: Machine learning biochemical networks from temporal logic properties. In: G. Plotkin (ed.) Transactions on Computational Systems Biology VI, *Lecture Notes in BioInformatics*, vol. 4220, pp. 68–94. Springer-Verlag (2006). DOI 10.1007/11880646_4. CMSB'05 Special Issue
19. Calzone, L., Fages, F., Soliman, S.: BIOCHAM: An environment for modeling biological systems and formalizing experimental knowledge. Bioinformatics **22**(14), 1805–1807 (2006). DOI 10.1093/bioinformatics/btl172

20. Chabrier, N., Fages, F.: Symbolic model checking of biochemical networks. In: C. Priami (ed.) CMSB'03: Proceedings of the first workshop on Computational Methods in Systems Biology, *Lecture Notes in Computer Science*, vol. 2602, pp. 149–162. Springer-Verlag, Rovereto, Italy (2003)
21. Chabrier-Rivier, N., Chiaverini, M., Danos, V., Fages, F., Schächter, V.: Modeling and querying biochemical interaction networks. Theoretical Computer Science **325**(1), 25–44 (2004)
22. Chazelle, B.: Natural algorithms and influence systems. Communications of the ACM **55**(12), 101–110 (2012). DOI 10.1145/2380656.2380679
23. Chiang, H.J., Fages, F., Jiang, J.H., Soliman, S.: Hybrid simulations of heterogeneous biochemical models in SBML. ACM Transactions on Modeling and Computer Simulation (TOMACS) **25**(2), 14:1–14:22 (2015). DOI 10.1145/2742545
24. Cimatti, A., Clarke, E., Enrico Giunchiglia, F.G., Pistore, M., Roveri, M., Sebastiani, R., Tacchella, A.: Nusmv 2: An opensource tool for symbolic model checking. In: Proceedings of the International Conference on Computer-Aided Verification, CAV'02. Copenhagen, Danmark (2002)
25. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press (1999)
26. Colom, J.M., Silva, M.: Convex geometry and semiflows in p/t nets. a comparative study of algorithms for computation of minimal p-semiflows. In: G. Rozenberg (ed.) Advances in Petri Nets 1990, *Lecture Notes in Computer Science*, vol. 483, pp. 79–112. Springer-Verlag, London, UK (1991). DOI 10.1007/3-540-53863-1_22
27. Cordone, R., Ferrarini, L., Piroddi, L.: Enumeration algorithms for minimal siphons in petri nets based on place constraints. IEEE transactions on systems, man and cybernetics. Part A, Systems and humans **35**(6), 844–854 (2005)
28. Courbet, A., Amar, P., Fages, F., Renard, E., Molina, F.: Computer aided biochemical programming of synthetic vesicles operating as logic-gated and multiplexed micro-scale diagnostic devices. Submitted
29. Courbet, A., Endy, D., Renard, E., Molina, F., Bonnet, J.: Detection of pathological biomarkers in human clinical samples via amplifying genetic switches and logic gates. Science Translational Medicine (2015)
30. Cousot, P., Cousot, R.: Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: POPL'77: Proceedings of the 6th ACM Symposium on Principles of Programming Languages, pp. 238–252. ACM Press, New York (1977). Los Angeles
31. De Maria, E., Fages, F., Rizk, A., Soliman, S.: Design, optimization, and predictions of a coupled model of the cell cycle, circadian clock, dna repair system, irinotecan metabolism and exposure control under temporal logic constraints. Theoretical Computer Science **412**(21), 2108–2127 (2011). DOI 10.1016/j.tcs.2010.10.036
32. Delzanno, G., Podelski, A.: Constraint-based deductive model checking. STTT **3**(3), 250–270 (2001)
33. Deng, K., Bourke, C., Scott, S.D., Sunderman, J., Zheng, Y.: Bandit-based algorithms for budgeted learning. In: ICDM (2007)
34. Deng, K., Zheng, Y., Bourke, C., Scott, S., Masciale, J.: New algorithms for budgeted learning. Mach Learn **90** (2013). DOI 10.1007/s10994-012-5299-2
35. Donzé, A., Maler, O.: Robust satisfaction of temporal logic over real-valued signals. In: FORMATS 2010, *Lecture Notes in Computer Science*, vol. 6246, pp. 92–106. Springer-Verlag (2010)
36. Eisenberg, M.: The kineticist's workbench: Combining symbolic and numerical methods in the simulation of chemical reaction mechanisms. Tech. Rep. 1306, MIT Technical Report (1991)
37. Eker, S., Knapp, M., Laderoute, K., Lincoln, P., Meseguer, J., Sönmez, M.K.: Pathway logic: Symbolic analysis of biological signaling. In: Proceedings of the seventh Pacific Symposium on Biocomputing, pp. 400–412 (2002)
38. Fages, F.: Consistency of Clark's completion and existence of stable models. Methods of Logic in Computer Science **1**, 51–60 (1994)

39. Fages, F., Gay, S., Soliman, S.: Inferring reaction systems from ordinary differential equations. Theoretical Computer Science **599**, 64–78 (2015). DOI 10.1016/j.tcs.2014.07.032
40. Fages, F., Martinez, T., Rosenblueth, D., Soliman, S.: Influence systems vs reaction systems. In: N.P. E. Bartocci P. Lio (ed.) CMSB'16: Proceedings of the fourteenth international conference on Computational Methods in Systems Biology, *Lecture Notes in BioInformatics*, vol. 9859, pp. 98–115. Springer-Verlag (2016). DOI 10.1007/978-3-319-45177-0_7
41. Fages, F., Rizk, A.: On temporal logic constraint solving for the analysis of numerical data time series. Theoretical Computer Science **408**(1), 55–65 (2008). DOI doi:10.1016/j.tcs.2008.07.004
42. Fages, F., Rizk, A.: From model-checking to temporal logic constraint solving. In: Proceedings of CP'2009, 15th International Conference on Principles and Practice of Constraint Programming, no. 5732 in Lecture Notes in Computer Science, pp. 319–334. Springer-Verlag (2009). DOI 10.1007/978-3-642-04244-7_26
43. Fages, F., Soliman, S.: Abstract interpretation and types for systems biology. Theoretical Computer Science **403**(1), 52–70 (2008). DOI 10.1016/j.tcs.2008.04.024
44. Fages, F., Soliman, S.: Model revision from temporal logic properties in systems biology. In: L. de Raedt, P. Frasconi, K. Kersting, S. Muggleton (eds.) Probabilistic Inductive Logic Programming, *Lecture Notes in Computer Science*, vol. 4911, pp. 287–304. Springer-Verlag (2008). DOI 10.1007/978-3-540-78652-8_11
45. Fages, F., Traynard, P.: Temporal logic modeling of dynamical behaviors: First-order patterns and solvers. In: L.F. del Cerro, K. Inoue (eds.) Logical Modeling of Biological Systems, chap. 8, pp. 291–323. John Wiley & Sons, Inc. (2014). DOI 10.1002/9781119005223.ch8
46. Fauré, A., Naldi, A., Lopez, F., Chaouiya, C., Ciliberto, A., Thieffry, D.: Modular logical modelling of the budding yeast cell cycle. Molecular Biosystems **5**, 1787–1796 (2009)
47. Fauré, A., Thieffry, D.: Logical modelling of cell cycle control in eukaryotes: a comparative study. Molecular Biosystems (2009)
48. Feinberg, M.: Mathematical aspects of mass action kinetics. In: L. Lapidus, N.R. Amundson (eds.) Chemical Reactor Theory: A Review, chap. 1, pp. 1–78. Prentice-Hall (1977)
49. de Figueiredo, L.F., Schuster, S., Kaleta, C., Fell, D.A.: Can sugars be produced from fatty acids? a test case for pathway analysis tools. Bioinformatics **25**(1), 152–158 (2009). DOI 10.1093/bioinformatics/btn621
50. Funahashi, A., Matsuoka, Y., Jouraku, A., Morohashi, M., Kikuchi, N., Kitano, H.: Celldesigner 3.5: A versatile modeling tool for biochemical networks. Proceedings of the IEEE **96**(8), 1254–1265 (2008). DOI 10.1109/JPROC.2008.925458
51. Gay, S., Fages, F., Martinez, T., Soliman, S., Solnon, C.: On the subgraph epimorphism problem. Discrete Applied Mathematics **162**, 214–228 (2014). DOI 10.1016/j.dam.2013.08.008
52. Gay, S., Soliman, S., Fages, F.: A graphical method for reducing and relating models in systems biology. Bioinformatics **26**(18), i575–i581 (2010). DOI 10.1093/bioinformatics/btq388. Special issue ECCB'10
53. Gebser, M., Kaufmann, B., Neumann, A., Schaub, T.: clasp: A conflict-driven answer set solver. In: In Proc. LPNMR'07, pp. 260–265. Springer (2007)
54. Gebser, M., Schaub, T., Thiele, S., Usadel, B., Veber, P.: Detecting inconsistencies in large biological networks with answer set programming. In: M.G. de la Banda, E. Pontelli (eds.) ICLP'08, Proceedings of the 24th International Conference on Logic Programming, *Lecture Notes in Computer Science*, vol. 5366, pp. 130–144. Springer-Verlag (2008). DOI 10.1007/978-3-540-89982-2_19
55. Ghosh, R., Tomlin, C.: Lateral inhibition through delta-notch signaling: A piecewise affine hybrid model. In: Springer-Verlag (ed.) Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control, HSCC'01, *Lecture Notes in Computer Science*, vol. 2034, pp. 232–246. Rome, Italy (2001)
56. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. Journal of Physical Chemistry **81**(25), 2340–2361 (1977)
57. Glass, L., Kauffman, S.A.: The logical analysis of continuous, non-linear biochemical control networks. Journal of theoretical Biology **39**(1), 103–129 (1973)

58. González, A.G., Chaouiya, C., Thieffry, D.: Qualitative dynamical modelling of the formation of the anterior-posterior compartment boundary in the drosophila wing imaginal disc. Bioinformatics **24**, 234–240 (2008)
59. Gordon, A.D., Henzinger, T.A., Nori, A.V., Rajamani, S.K.: Probabilistic programming. In: Proceedings of the on Future of Software Engineering, FOSE 2014, pp. 167–181. ACM, New York, NY, USA (2014). DOI 10.1145/2593882.2593900
60. Grieco, L., Calzone, L., Bernard-Pierrot, I., Radvanyi, F., Kahn-Perlès, B., Thieffry, D.: Integrative modelling of the influence of mapk network on cancer cell fate decision. PLOS Computational Biology **9**(10), e1003,286 (2013)
61. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9**(2), 159–195 (2001)
62. Heitzler, D., Durand, G., Gallay, N., Rizk, A., Ahn, S., Kim, J., Violin, J.D., Dupuy, L., Gauthier, C., Piketty, V., Crépieux, P., Poupon, A., Clément, F., Fages, F., Lefkowitz, R.J., Reiter, E.: Competing G protein-coupled receptor kinases balance G protein and β-arrestin signaling. Molecular Systems Biology **8**(590) (2012). DOI 10.1038/msb.2012.22
63. Henzinger, T.A.: The theory of hybrid automata. In: Proceedings of the 11th Annual Symposium on Logic in Computer Science (LICS), pp. 278–292. IEEE Computer Society Press (1996). An extended version appeared in Verification of Digital and Hybrid Systems
64. Herrgård, M.J., Swainston, N., Dobson, P., Dunn, W.B., Arga, K.Y., et al.: A consensus yeast metabolic network reconstruction obtained from a community approach to systems biology. Nature Biotechnology **26**(10), 1155–1160 (2008). DOI 10.1038/nbt1492
65. Huang, C.Y., Ferrell, J.E.: Ultrasensitivity in the mitogen-activated protein kinase cascade. PNAS **93**(19), 10,078–10,083 (1996)
66. Hucka, M., Hoops, S., Keating, S.M., Nicolas, L.N., Sahle, S., Wilkinson, D.: Systems biology markup language (SBML) level 2: Structures and facilities for model definitions. Nature Precedings (2008). DOI 10.1038/npre.2008.2715.1
67. Hucka, M., et al.: The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. Bioinformatics **19**(4), 524–531 (2003)
68. Ideker, T., Galitski, T., Hood, L.: A new approach to decoding life: Systems biology. Annual Review of Genomics and Human Genetics **2**, 343–372 (2001)
69. Inoue, K.: Logic programming for boolean networks. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two, IJCAI'11, pp. 924–930. AAAI Press (2011). DOI 10.5591/978-1-57735-516-8/IJCAI11-160
70. de Jong, H., Gouzé, J.L., Hernandez, C., Page, M., Sari, T., Geiselmann, J.: Qualitative simulation of genetic regulatory networks using piecewise-linear models. Bulletin of Mathematical Biology **66**(2), 301–340 (2004)
71. von Kamp, A., Schuster, S.: Metatool 5.0: fast and flexible elementary modes analysis. Bioinformatics **22**(15), 1930–1931 (2006)
72. Kanehisa, M., Goto, S.: KEGG: Kyoto encyclopedia of genes and genomes. Nucleic Acids Research **28**(1), 27–30 (2000)
73. Karr, J.R., Sanghvi, J.C., Macklin, D.N., Gutschow, M.V., Jacobs, J.M., Bolival, B., Assad-Garcia, N., Glass, J.I., Covert, M.W.: A whole-cell computational model predicts phenotype from genotype. Cell **150**(2), 389,401 (2012)
74. Kitano, H.: Towards a theory of biological robustness. Molecular Systems Biology **3**, 137 (2007)
75. Kleene, S.: Representation of events in nerve nets and finite automata, pp. 3–41. Princeton University Press (1956)
76. Kohn, K.W.: Molecular interaction map of the mammalian cell cycle control and DNA repair systems. Molecular Biology of the Cell **10**(8), 2703–2734 (1999)
77. Larhlimi, A., Bockmayr, A.: A new constraint-based description of the steady-state flux cone of metabolic networks. Discrete Applied Mathematics **157**(10), 2257–2266 (2009). DOI 10.1016/j.dam.2008.06.039. Networks in Computational Biology
78. Llamosi, A., Mezine, A., d'Alché Buc, F., Letort, V., Sebag, M.: Experimental design in dynamical system identification: a bandit-based active learning approach. In: T. Calders,

F. Esposito, E. Hüllermeier, R. Meo (eds.) Machine Learning and Knowledge Discovery in Databases ECML PKKDD'14, *Lecture Notes in Artificial Intelligence*, vol. 8724, pp. 306–321. Springer-Verlag (2014)
79. Matsuno, H., Doi, A., Nagasaki, M., Miyano, S.: Hybrid petri net representation of gene regulatory network. In: Proceedings of the 5th Pacific Symposium on Biocomputing, pp. 338–349. Stanford, Hawaii, USA (2000)
80. McCulloch, W., Pitts, W.: A logical calulus of ideas immanent in nervous activity. Bulletin of Mathematical Biophysics **5**, 115–133 (1943)
81. Meyer, P., Cokelaer, T., Chandran, D., Kim, K.H., Loh, P.R., Tucker, G., Lipson, M., Berger, B., Kreutz, C., Raue, A., Steiert, B., Timmer, J., Bilal, E., Sauro, H.M., Stolovitzky, G., Saez-Rodriguez, J.: Network topology and parameter estimation: from experimental design methods to gene regulatory network kinetics using a community based approach. BMC Systems Biology **8**(1), 1–18 (2014). DOI 10.1186/1752-0509-8-13
82. Muggleton, S.H.: Inverse entailment and progol. New Generation Computing **13**, 245–286 (1995)
83. Nabli, F., Martinez, T., Fages, F., Soliman, S.: On enumerating minimal siphons in petri nets using CLP and SAT solvers: Theoretical and practical complexity. Constraints **21**(2), 251–276 (2016). DOI 10.1007/s10601-015-9190-1
84. Naldi, A., Berenguier, D., Fauré, A., Lopez, F., Thieffry, D., Chaouiya, C.: Logical modelling of regulatory networks with GINsim 2.3. Biosystems **97**(2), 134–139 (2009). DOI 10.1016/j.biosystems.2009.04.008
85. Naldi, A., Carneiro, J., Chaouiya, C., Thieffry, D.: Diversity and plasticity of th cell types predicted from regulatory network modelling. PLoS Computational Biology **6**(9), e1000,912 (2010). DOI 10.1371/journal.pcbi.1000912
86. Neumann, J.V.: Theory of Self Replicating Automata. University of Illinois Press (1966)
87. Nielsen, A.A.K., Der, B.S., Shin, J., Vaidyanathan, P., Paralanov, V., Strychalski, E.A., Ross, D., Densmore, D., Voigt, C.A.: Genetic circuit design automation. Science **352**(6281) (2016). DOI 10.1126/science.aac7341
88. le Novère, N., Bornstein, B., Broicher, A., Courtot, M., Donizelli, M., Dharuri, H., Li, L., Sauro, H., Schilstra, M., Shapiro, B., Snoep, J.L., Hucka, M.: BioModels Database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. Nucleic Acid Research **1**(34), D689–D691 (2006)
89. Ostrowski, M., Paulevé, L., Schaub, T., Siegel, A., Guziolowski, C.: Boolean network identification from perturbation time series data combining dynamics abstraction and logic programming. Biosystems **149**, 139–153 (2016). DOI 10.1016/j.biosystems.2016.07.009
90. Pearl, J.: Causality: Models, Reasoning and Inference, 2nd edn. Cambridge University Press, New York, NY, USA (2009)
91. Peres, S., M., M., Simon, L.: Sat-based metabolics pathways analysis without compilation. In: P.M. et al. (Eds.): CMSB (ed.) Lecture Note in Bioinformatics, vol. 8859, pp. 20–31. Springer International Publishing (2014)
92. Peterson, J.L.: Petri Net Theory and the Modeling of Systems. Prentice Hall, New Jersey (1981)
93. Reddy, V.N., Mavrovouniotis, M.L., Liebman, M.N.: Petri net representations in metabolic pathways. In: L. Hunter, D.B. Searls, J.W. Shavlik (eds.) Proceedings of the 1st International Conference on Intelligent Systems for Molecular Biology (ISMB), pp. 328–336. AAAI Press (1993)
94. Remy, E., Ruet, P., Thieffry, D.: Graphic requirements for multistability and attractive cycles in a boolean dynamical framework. Advances in Applied Mathematics **41**(3), 335–350 (2008). DOI 10.1016/j.aam.2007.11.003
95. Rezola, A., de L. F. Figueiredo, Brock, M., Pey, J., Podhorski, A., Wittmann, C., Schuster, S., Bockmayr, A., Planes, F.J.: Exploring metabolic pathways in genome-scale networks via generating flux modes. Bioinformatics **27**(4), 534–540 (2011). DOI 10.1093/bioinformatics/btq681

96. Rizk, A., Batt, G., Fages, F., Soliman, S.: A general computational method for robustness analysis with applications to synthetic gene networks. Bioinformatics **12**(25), il69–il78 (2009). DOI 10.1093/bioinformatics/btp200
97. Rizk, A., Batt, G., Fages, F., Soliman, S.: Continuous valuations of temporal logic specifications with applications to parameter optimization and robustness measures. Theoretical Computer Science **412**(26), 2827–2839 (2011). DOI 10.1016/j.tcs.2010.05.008
98. Rosenblueth, D.A., Muñoz, S., Carrillo, M., Azpeitia, E.: Inference of Boolean networks from gene interaction graphs using a SAT solver. In: AlCoB 2014: Proceedings of the 1st International Conference on Algorithms for Computational Biology, *Lecture Notes in BioInformatics*, vol. 8542, pp. 235–246. Springer-Verlag (2014). DOI 10.1007/978-3-319-07953-0_19
99. Ruet, P.: Local cycles and dynamical properties of boolean networks. Mathematical Foundations of Computer Science **26**(4), 702–718 (2016)
100. Sánchez, L., Chaouiya, C., Thieffry, D.: Segmenting the fly embryo: logical analysis of the role of the segment polarity cross-regulatory module. International Journal of Developmental Biology **52**, 1059–1075 (2008)
101. Singhania, R., Sramkoski, R.M., Jacobberger, J.W., Tyson, J.J.: A hybrid model of mammalian cell cycle regulation. PLOS Computational Biology **7**(2), e1001,077 (2011)
102. Snoussi, E.H.: Necessary conditions for multistationarity and stable periodicity. Journal of Biological Systems **6**, 3–9 (1998). DOI 10.1142/S0218339098000042
103. Soliman, S.: Invariants and other structural properties of biochemical models as a constraint satisfaction problem. Algorithms for Molecular Biology **7**(15) (2012). DOI 10.1186/1748-7188-7-15
104. Soulé, C.: Graphic requirements for multistationarity. ComplexUs **1**, 123–133 (2003)
105. Sun, J., Garibaldi, J.M., Hodgman, C.: Parameter Estimation Using Meta-Heuristics in Systems Biology: A Comprehensive Review. IEEE/ACM Trans Comput Biol Bioinform, New Jersey (2011)
106. Thomas, R.: Boolean formalisation of genetic control circuits. Journal of Theoretical Biology **42**, 565–583 (1973)
107. Thomas, R.: On the relation between the logical structure of systems and their ability to generate multiple steady states or sustained oscillations. Springer Ser. Synergetics **9**, 180–193 (1981)
108. Thomas, R.: Regulatory networks seen as asynchronous automata : a logical description. Journal of Theoretical Biology **153**, 1–23 (1991)
109. Thomas, R., D'Ari, R.: Biological Feedback. CRC Press (1990)
110. Thomas, R., Kaufman, M.: Multistationarity, the basis of cell differentiation and memory. Chaos **11**(1), 170–195 (2001)
111. Traynard, P., Fauré, A., Fages, F., Thieffry, D.: Logical model specification aided by model-checking techniques: application to the mammalian cell cycle regulation. Bioinformatics **32**(17), i772–i780 (2016). DOI 10.1093/bioinformatics/btw457
112. Traynard, P., Feillet, C., Soliman, S., Delaunay, F., Fages, F.: Model-based investigation of the circadian clock and cell cycle coupling in mouse embryonic fibroblasts: Prediction of reverb-alpha up-regulation during mitosis. Biosystems **149**, 59–69 (2016). DOI 10.1016/j.biosystems.2016.07.003
113. Turing, A.: Computing machinery and intelligence. Mind **49**, 433–460 (1950)
114. Tyson, J.J.: Modeling the cell division cycle: cdc2 and cyclin interactions. Proceedings of the National Academy of Sciences **88**(16), 7328–7332 (1991)
115. Uhlendorf, J., Miermont, A., Delaveau, T., Charvin, G., Fages, F., Bottani, S., Batt, G., Hersen, P.: Long-term model predictive control of gene expression at the population and single-cell levels. Proceedings of the National Academy of Sciences USA **109**(35), 14,271–14,276 (2012). DOI 10.1073/pnas.1206810109
116. Valdès-Pérè, R.: Machine discovery in chemistry: new results. Artificial Intelligence **74**, 191–201 (1995)
117. Valiant, L.: A theory of the learnable. Communications of the ACM **27**(11), 1134–1142 (1984)

118. Valiant, L.: Probably Approximately Correct. Basic Books (2013)
119. Varma, A., Palsson, B.: Metabolic flux balancing: basic concepts, scientific and practical use. Nature Biotechnology **12**(10), 994–998 (1994)
120. VBI and EML Research: COPASI's manual
121. Zevedei-Oancea, I., Schuster, S.: Topological analysis of metabolic networks based on petri net theory. In Silico Biology **3**(29) (2003)