Inferring parameters of genetic regulatory networks with symbolic formal methods

Daniel Mateus[1], Jean-Paul Comet[2], Jean-Pierre Gallois[1] & Pascale Le Gall[2]

[1] CEA, LIST, Saclay, F-91191 Gif sur Yvette Cedex, France
[2] IBISC, FRE CNRS 2873, Université d'Évry Val d'Essonne & Epigenomics Project, Genopole®, 523 terrasses de l'Agora, 91000 Évry, France

Abstract

Understanding the functioning of genetic regulatory networks supposes a modeling of biological processes in order to simulate behaviors and to reason on the model. Unfortunately, the modeling task is confronted to incomplete knowledge about the system. To deal with this problem we propose a methodology that uses the qualitative approach developed by R. Thomas. A symbolic transition system can represent the set of all possible models in a concise and symbolic way. We introduce a new method based on model-checking techniques and symbolic execution to extract constraints on parameters leading to dynamics coherent with known behaviors. Our method allows us to efficiently respond to two kinds of questions: is there any model coherent with a certain hypothetic behavior? Are there behaviors common to all selected models? The first question is illustrated with the example of the mucus production in *Pseudomonas aeruginosa* while the second one is illustrated with the example of immunity control in bacteriophage lambda.

Keywords: Gene networks; qualitative dynamical models; symbolic execution; temporal properties; model-checking.

1 Introduction

Genetic regulatory networks are constituted of various interacting components, mainly genes and proteins, usually forming a complex network of interleaved feedback loops. As it is impossible to use intuitive reasoning to really understand these networks and predict their possible behaviors, modeling and simulation become necessary [1]. The lack of reliable quantitative data available about a given system is a typical difficulty of the modeling approach. To overpass this problem, qualitative models have been developed, whose goal consists in abstracting details of the system although preserving qualitative observations.

Boolean models of genetic regulatory networks [2] are one of such formalisms. In these models, the constituents of the network are represented by variables that can only take two values, 0 or 1, meaning that the associated component is absent or present (or that the associated gene is inactive or active). R. Thomas proposed an *asynchronous* boolean modeling [3]: his approach takes into account the fact that the delays of synthesis or degradation are different from one protein to another, whereas it is not the case in previous boolean models [4]. The relation between boolean models and piecewise-linear differential equations have been first discussed in [5]. R. Thomas' approach has been generalized to a multilevel discrete modeling [4, 6, 7]; in this generalized formalism the concentrations of the constituents of the network are represented by integer variables which can take a finite number of values. Such a discrete model can be seen as a precise qualitative abstraction of a system of piecewise-linear differential equations, as demonstrated by E.H. Snoussi [8]. This formalism is described in Sec. 2, where the convenience of introducing more than two levels of expression for the variables is explained.

This generalized discrete approach has been used to model various gene networks (for example in [9, 10, 11, 12, 13]). H. de Jong et al. [14] have recently proposed a refinement of R. Thomas' discrete modeling that takes into account singular states (corresponding to frontiers between qualitative states).

Nevertheless, even in such a discrete and finite formalism there are usually more than one model compatible with the knowledge on the system. Knowledge generally consists, on the one hand, in inhibitions or activations between genes and other constituents of the network, and on the other hand, in behaviors, observed in experiments. Inhibitions or activations allow one to constrain the possible values of the parameters of the model, on which the evolution depends. It is more difficult to select the parameters corresponding to observed behaviors. For example, the properties relating homeostasis (stable cyclic behavior) or multi-stationarity to the steadiness of characteristic states of feedback circuits [15, 16] can be used to decrease the number of parameter values to be considered, as in the GINsim tool [17].

To go further, two main ideas have been proposed. The first one consists in using constraint logic programming, to manipulate partially known models [18]. As this approach does not allow one to describe all observed behaviors, the difficulty of selecting parameters according to observations remains. The other one consists in formally specifying temporal properties and in verifying if the constructed model satisfies the specification. For example Shaub et al. [19] proposed a method for determining all infinitely visited states for which the observed behaviors have to be verified. More generally the specification can be expressed in a formal temporal language (like computational tree logic – CTL) and verification of behavior specification is then studied for each possible complete model (*i.e.* where each parameter has a precise value) independently. Implementing this idea, the tool SMBioNet [20] selects the models with respect to a given specified behavior after having exhaustively generated all possible models. In the tool GNA [21], CTL is also used to specify behaviors but only one complete model can be simulated.

Description of the proposed method

In this chapter we propose a method combining the advantages of both approaches described above. The set of possible models can be represented by a unique formal model, a symbolic transition system (STS) [22]. Symbolic execution techniques allow the simulation of the STS, generating all possible behaviors. We specify behaviors using linear temporal logic (LTL) [23], and we select parameters with respect to LTL formulas by building constraints: parameters satisfying these constraints define the set of all models verifying the specified behavior.

Thus we propose a methodology to analyze partially known systems. On the one hand, an interaction graph of the system leads to a STS, representing the set of discrete models compatible with the interactions; on the other hand, the known behaviors of the system are translated into LTL formulas. Constraints associated with these formulas restrict the possible values of the parameters; then these constraints are added to the initial STS, which represents the set of discrete models with the specified behavior. We will see in the sequel two different types of questions that can be asked after this construction:

- is there any model coherent with a certain hypothetic behavior? The hypothetic behavior is translated into LTL formulas, and the method finds the possible parameters coherent with this hypothesis or shows that this behavior is impossible over the set of selected models. This case is illustrated on the example of mucus production in *Pseudomonas aeruginosa*.

- Are there behaviors common to all selected models? We will see that the symbolic representation of possible parameters allows to exhibit common behaviors of the selected models, without having to enumerate the models. The example of immunity control in bacteriophage lambda illustrates this point.

After having described the R. Thomas' discrete modeling, we introduce, in Sec. 2, constraints deduced from gene interactions, and show their use in the system associated to mucus production in *Pseudomonas aeruginosa*. This system will be used as a running example to illustrate our method. Section 3 is divided in three parts. We firstly explain the translation of a set of models into a STS model. We secondly introduce symbolic execution techniques. We thirdly explain how behaviors can be specified with LTL formulas, and the way we extend usual model-checking techniques to characterize parameters coherent with the LTL formulas. Then we show how this framework can be fruitfully applied to discover the unknowns (parameters or behaviors) of the genetic regulatory network. Section 4 illustrates the whole methodology on the example of immunity control in bacteriophage lambda.

This chapter is a synthesis of recent works [24, 25, 26, 27] and is based on results presented in [28] that have been enriched and completed. From a practical point of view this proposed methodology has been implemented in the Agatha tool, which is also used for validation purposes of industrial specifications [29, 30].

2 Discrete modeling of genetic regulatory networks

In this section we first present the notion of discrete descriptions, also called complete or basic models in the sequel. They correspond to the generalized discrete models introduced by R. Thomas [4]. These models are based on the interaction graph of the system: interaction graphs are directed graphs whose nodes abstract genes and associated proteins (called variables in the sequel) and whose edges are labeled by signs and thresholds of interactions. The threshold of the interaction $a \xrightarrow{\theta,+} b$ (resp. $a \xrightarrow{\theta,-} b$) defines when the interaction takes place: variable a activates (resp. inhibits) variable b if its concentration level is above θ. The effect of a on b does not depend on the concentration of a as soon as the concentration of a is above θ.

This remark leads to the discretization of the concentration space of the system variables: if a has k outgoing edges labeled by different thresholds $\theta_1 < ... < \theta_k$, then the concentration space of a is discretized into $k + 1$ levels denoted by integers from 0 to k. Then the level i abstracts the concentrations which are above θ_i (if $i > 0$) and below θ_{i+1} (if $i < k$). Thus the real values of thresholds θ_i do not matter for the discrete dynamics. They are then modeled by integers which reflect their relative ranks.

This possibility of having different thresholds makes generalized discrete models more expressive than simpler boolean models: if a variable a has an effect on two other variables b and c, the threshold of the two interactions are generally not equal; so the possible levels of a are 0, where no interaction is effective, 1 where only one interaction is effective, and 2 where the two interactions are effective. Boolean models can not distinguish different thresholds, as the level of a would be 0 (no effective interaction) or 1 (all interactions are effective).

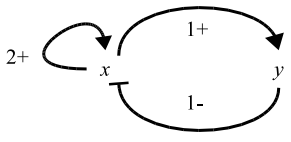

Figure 1: Example of interaction graph. Each arrow indicates an interaction from a regulator to a regulated variable; the sign indicates a positive or negative effect, and the integer is the rank of the threshold of the interaction. The blunt arrow indicates the negative interaction.

Example 1 *In Fig. 1, x and y represent two proteins produced by two genes. Variable x has two outgoing edges, with two different thresholds; the possible values for x are 0, 1 and 2; the*

threshold of the interaction on y is less than the threshold of the interaction on x itself, so the integer associated with the threshold of the interaction of x on y is 1, whereas the integer associated to the threshold of the interaction of x on itself is 2. The possible values for y are only 0 or 1. The genetic regulatory network corresponding to this interaction graph is described in Sec. 2.3.

In Sec. 2.1 we present the possible discrete dynamics governed by parameters associated to discrete states. We then show how biological knowledge, in particular the gene interaction graph, can be used to construct a set of acceptable discrete descriptions.

2.1 Dynamics of a discrete description

In a discrete model, the genetic regulatory network is described by n variables, each representing the concentration of a constituent of the actual network. Each variable x_i can take an integer value between 0 and a maximum value max_i (this maximum value is deduced from the interaction graph of the system as explained above). A *state* $E = (E_1, \ldots, E_n)$ is a vector of values of the variables. With each state E, and each variable x_i, is associated a parameter $K(x_i, E)$, which has an integer value between 0 and max_i (the same maximum value than x_i). This parameter is the value toward which the associated variable tends in the associated state. It means that in the state E:

- If $K(x_i, E) > E_i$, then $(E_1, \ldots, E_i + 1, \ldots, E_n)$ is a successor of E;
- If $K(x_i, E) < E_i$, then $(E_1, \ldots, E_i - 1, \ldots, E_n)$ is a successor of E;
- If $K(x_i, E) = E_i$ for all i, then E is called a steady state, and has only itself as successor.

The associated *transition graph* is constituted of the states, and the transitions between each state and its successors. This complete model, for which each parameter has been instantiated, is called in the sequel a *discrete description*.

Let us remark that a successor of a state E differs from E in at most one coordinate: only one value from E_1 to E_n is modified (by adding or subtracting 1), if E is not a steady state. This property is called *asynchronous* updating of the variables. The reason is that when the concentration of two (or more) constituents of the network increase or decrease, there is no reason that these concentrations reach their threshold at the same time. So one of the concentration reach the threshold first; then the state of the system becomes different, with different interactions leading to different behaviors (*i.e.* the associated parameter can be different). Without knowledge on these delays, there can be more than one successor to a given state. See [4] for details about this point, and also the notion of desynchronization formally defined in [20].

Example 2 *We consider the system corresponding to Fig. 1. A state $(E_1, E_2) \in \{0, 1, 2\} \times \{0, 1\}$ is defined by the values of variables x and y. If $K(x, (0, 0)) = 1$ and $K(y, (0, 0)) = 1$ then the state $(0, 0)$ has two successors, $(1, 0)$ and $(0, 1)$. It means that if in the system the concentrations of the two proteins are at the lowest level, the concentrations increase to reach a state corresponding to $(1, 0)$ or $(0, 1)$.*

Until now the sign of the interactions between the constituents of the network is not taken into account. As shown in Sec. 2.2, equalities and inequalities between parameters can be deduced when positive or negative interactions between genes are known.

2.2 Constraints on parameters deduced from interactions

We have seen that each edge of the interaction graph is associated with a threshold. If a protein a activates a gene producing a protein b, the rate of synthesis of b is a sigmoid function of the concentration of a: it means that when the concentration of a is under a threshold θ, the rate of synthesis of b is not affected; but if the concentration of a is greater than the threshold θ, the rate of synthesis gets rapidly a maximal value. In piecewise-linear differential descriptions and associated qualitative models, these sigmoid functions are approximated by step functions [4, 5]. So, if in a discrete description a variable a has more than two discrete levels, and has an effect on b at the level 1, a has no effect on b when the level of a is 0, and a has the same effect on b when the level of a is 1, 2 or more.

More generally, the following equalities can be deduced from the interaction graph: we suppose that a variable x_i has one interaction on a variable x_j, and that the associated threshold has an integer level (or rank) t. Let $E = (E_1, \ldots, E_n)$ and $E' = (E'_1, \ldots, E'_n)$ be two states such that $E_i < t$, $E'_i < t$ and for every $k \neq i$, $E_k = E'_k$. E' differs from E at most in its i^{th} coordinate. Then $K(x_j, E) = K(x_j, E')$. Similarly, if $E_i \geq t$ and $E'_i \geq t$ then $K(x_j, E) = K(x_j, E')$.

These equalities allow the introduction of a new notation of the parameters: let Y be the subset of the variables $\{x_1, \ldots, x_n\}$ whose elements can have an action on x_j, and X a subset of Y; then if E is a state where the value of each variable in X is greater than or equal to the threshold of its interaction on x_j, and values of variables in $Y \backslash X$ are less than their thresholds, then the value of $K(x_j, E)$ is denoted by $K(x_j, X)$.

Example 3 *In discrete descriptions associated to Fig. 1,* $K(x, \emptyset) = K(x, (0, 0)) = K(x, (1, 0))$ *(the value of* x*, 0 or 1, is under the threshold of the interaction on itself, which is 2, and the value of* y*, 0, is under the threshold of the interaction on* x*, which is 1). Similarly* $K(x, \{y\}) = K(x, (0, 1)) = K(x, (1, 1))$ *(here the value of* y*, 1, is equal to the threshold).*

Moreover the sign of the interactions imply constraints on the parameters. We suppose again that a variable x_i has one interaction on a variable x_j, and X denotes a set of variables such that $x_i \notin X$. Then we have:

- $K(x_j, X) \leq K(x_j, X \cup \{x_i\})$ if x_i has a positive interaction on x_j;
- $K(x_j, X) \geq K(x_j, X \cup \{x_i\})$ if x_i has a negative interaction on x_j.

Let us point out that the inequalities are not strict: for example we can say that $K(x_j, X) \leq K(x_j, X \cup \{x_i\})$ rather than $K(x_j, X) < K(x_j, X \cup \{x_i\})$. The reason is that even if there is a positive or negative interaction, it is not sure that the interaction is sufficient to make the regulated variable reach a greater or lower threshold.

Example 4 *In discrete descriptions associated to Fig. 1,* $K(x, \{y\}) \leq K(x, \emptyset) \leq K(x, \{x\})$ *and* $K(x, \{y\}) \leq K(x, \{x, y\}) \leq K(x, \{x\})$ *(because* y *has a negative interaction on* x*, and* x *has a positive interaction on itself), and similarly* $K(y, \emptyset) \leq K(y, \{x\})$ *(*x *has a positive interaction on* y*).*

Sometimes more precise knowledge about the interactions is available. For example the presence of two different products x *and* y can be necessary to activate a gene z, or x can activate z but the simultaneous presence of x and y produces an inhibition. These two facts are respectively translated into constraints: $K(z, \{x\}) = K(z, \{y\}) = K(z, \emptyset)$ and $K(z, \{x, y\}) \geq K(z, \emptyset)$ in the first case, or $K(z, \{x, y\}) \leq K(z, \emptyset) \leq K(z, \{x\})$ in the second case.

2.3 *Mucus production in* Pseudomonas aeruginosa

Pseudomonas aeruginosa are bacteria that secrete mucus (alginate) in lungs affected by cystic fibrosis, but not in common environment. As this mucus increases respiratory deficiency, this phenomenon is a major cause of mortality. Details of the regulatory network associated with the mucus production are described by Govan and Deretic [31]. The simplified regulatory network, as proposed by Guespin and Kaufman [32], contains the protein AlgU (product of algU gene) and an inhibitor complex anti-sigma (product of muc genes). AlgU has a positive effect on anti-sigma and on itself, while anti-sigma has a negative effect on AlgU. A sufficient concentration of AlgU leads to the production of mucus (by activating different alg genes). If we consider that the threshold of the interaction of AlgU on anti-sigma is under the threshold of auto-activation of AlgU, then Fig. 1 is the interaction graph corresponding to the discrete descriptions where x and y represent respectively AlgU and anti-sigma. We consider that the production of mucus occurs precisely when the value of x is 2.

Constraints on parameters are described in the examples of Sec. 2.2. Moreover we assume that $K(x, \{y\}) = 0$ and $K(y, \emptyset) = 0$. This additional constraints mean that x tend toward its basal level (*i.e.* 0) without auto-activation and under inhibition of y and, similarly, that y tend toward its basal level when x does not activate it. The set of all these constraints will be denoted by $\mathcal{C}$ in the sequel.

It has been observed that mucoid *P. aeruginosa* can continue to produce mucus isolated from infected lungs. It is commonly thought that the mucoid state of *P. aeruginosa* is due to a mutation which cancels the inhibition of algU gene. An alternative hypothesis has been made: this mucoid state can occur in reason of an epigenetic modification, *i.e.* without mutation [32]. The models compatible with this hypothesis have been constructed in [33, 20]. We use the same example to explain our methodology in Sec. 3.

2.4 Manipulating sets of discrete descriptions

The only knowledge of the interaction graph is not sufficient to precisely determine which is the behavior of the biological system: numerous discrete descriptions can fit the constraints deduced from the interaction graph. In the example of Fig. 1, there are 6 states, so 6 parameters associated with x (with 3 possible values) and 6 with y (with 2 possible values). It results in $3^6 \times 2^6 = 46656$ different discrete descriptions.[1] With the equalities described in example 3, there remain $3^4 \times 2^2 = 324$ discrete descriptions, since parameters $K(x, \emptyset)$, $K(x, \{x\})$, $K(x, \{y\})$ and $K(x, \{x, y\})$ can take three different values (0, 1 or 2), and parameters $K(y, \emptyset)$ and $K(y, \{x\})$ can take two different values (0 or 1). The assumption that $K(x, \{y\}) = 0$ and $K(y, \emptyset) = 0$ reduce the set of possible discrete descriptions to $3^3 \times 2 = 54$ elements. Finally 28 of these discrete descriptions verify the inequalities deduced from the signs of interactions in example 4.

In order to precise the behavior of the biological system, complementary biological knowledge, different from previously used interaction graphs, have to be taken into consideration. To reduce the set of acceptable discrete descriptions we will express biological knowledge by temporal logic formulas involving equalities and inequalities on gene expression levels. Then model checking techniques combined with symbolic execution of the symbolic model denoting sets of acceptable discrete description will give will give us the set of acceptable parameters.

[1] Let us recall that a discrete description is completely defined by the values of parameters. However there are only $2^{10} \times 3^2 = 9216$ different dynamics, *i.e.* different transition graphs, for these discrete descriptions. Indeed, two different values of parameters can lead to the same dynamics because the parameters give only the directions of evolution.

3 Symbolic formal methods

3.1 Symbolic transition systems

A symbolic transition system (STS) [22] is a transition system whose transitions are labeled by conditions on STS variables and assignments of STS variables. Each initialization of STS variables yields a basic model where each variable has a precise initial value, and all transitions are defined according to the STS transitions. Thus a STS is parameterized by an initialization function.

Let M be a STS, $V = \{v_1, v_2, \ldots, v_k\}$ the set of STS variables; then an initialization function of M is a map from V to the set of possible values of the variables. If σ is an initialization function, M_σ denotes a basic model whose first state is $(\sigma(v_1), \ldots, \sigma(v_k))$. So we can associate to M the set of all basic models obtained by applying an initialization function: $\{M_\sigma \mid \sigma \text{ initialization function}\}$ denotes this set.

A STS can represent a set of discrete descriptions associated to an interaction graph. In this case, STS variables are divided into two subsets:

- the set of variables $\{x_i \mid 0 \leq i \leq n\}$;
- the set of parameters $\{K(x_i, E) \mid 0 \leq i \leq n, E \in \{0, \ldots, max_1\} \times \cdots \times \{0, \ldots, max_n\}\}$ of the associated discrete descriptions (max_i is the maximal value of variable x_i).

The transitions are labeled according to the rules defined in Sec. 2.1. Nevertheless we need to take into account additional knowledge corresponding to constraints deduced from interactions. These constraints can naturally be expressed as first order formulas over the set of parameters. So we call symbolic model any couple $(M, \mathcal{C})$, where M is the STS with parameters $\{K(x_i, E)\}$ and variables $\{x_i\}$ as STS variables and $\mathcal{C}$ a set of constraints over parameters $\{K(x_i, E)\}$. It defines a set of basic models $\{M_\sigma \mid \sigma \text{ initialization function} \wedge \forall C \in \mathcal{C}, \sigma \models C\}$, where $\sigma \models C$ means that the parameters instantiated by σ satisfy the constraint C. Each basic model M_σ is then a discrete description associated to the values of parameters defined by σ (but with one distinguished initial state).

For the same instantiation of the parameters, every instantiation of the variables $\{x_i\}$ corresponds to the same discrete description; so a discrete description is completely defined by an initialization function σ' assigning a value only to parameters. Initialization of variables x_i allows one to specify initial states of the system if necessary.

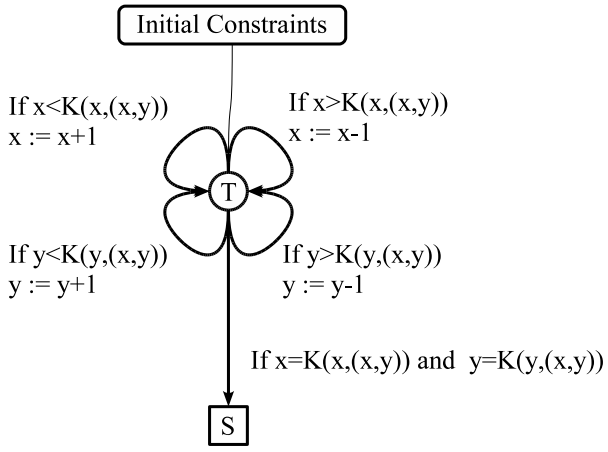

Figure 2: STS associated with Fig. 1. Arrows represent the transitions, labeled by a condition and an assignment.

Example 5 *Figure 2 represents the symbolic model associated with the interaction graph of Fig. 1, corresponding to the network of mucus production in* P. aeruginosa*. Initial constraints on parameters, denoted by $\mathcal{C}$, are specified in Secs. 2.2 and 2.3. The* control point *denoted by T in Fig. 2, indicates that the system is in a transient state (*i.e. *non-steady state), whereas the control point denoted by S indicates that the system has reached a steady state. We see that there are four different transitions from T to T: two of them correspond to a change of x and two of them correspond to a change of y. The transition from T to S occurs when all parameters of the current state are equal to the current values of the variables x and y.*

3.2 Symbolic execution

Symbolic execution has been introduced for analysis purposes of computer programs [34]. The method has been extended to STSs, and is used in the Agatha tool for behavioral analysis [35] and conformance testing [36]. As the known constraints and rules of evolution of a discrete description can easily been specified in a STS, we have adapted symbolic execution techniques to generate all behaviors compatible with the constraints on the parameters. The method constructs a tree whose vertices are states labeled by constraints, with the following rules:

- The root of the tree is a state, associated with the initial constraints $\mathcal{C}$.
- Let us suppose that E is an already constructed state of the tree, labeled by the constraints $\mathcal{C}_E$, and that there is a STS transition from E to E' labeled by the condition D. The state E' provided with the constraint $\mathcal{C}_{E'} = \mathcal{C}_E \cup \{D\}$ is built if and only if the conjunction of the constraints of $\mathcal{C}_E \cup \{D\}$ is satisfiable. A new transition is built from $(E, \mathcal{C}_E)$ to $(E', \mathcal{C}_{E'})$.
- The process is repeated until the new state has already been encountered in the tree path from the root to the current state.

Let us point out that every state in the tree is associated with constraints whose conjunction is called *path condition*; this path condition is the condition on parameters under which the path exists.

Example 6 *Figure 3 shows the symbolic execution of the symbolic model associated with mucus production system in* P. aeruginosa*, with $(x, y) = (0, 1)$ as initial state, and $\mathcal{C}$ as initial constraints, as described in Sec. 2.3. The states in circles correspond to the control point T in the STS of Fig. 2, whereas states in squares correspond to the control point S,* i.e. *to steady states.*

Each state of the figure is associated with constraints; for example:

- $(0, 0)$ *is the only successor of* $(0, 1)$ *because initial constraints contain the equalities $K(x, \{y\}) = 0$ and $K(y, \emptyset) = 0$,* i.e. *$K(x, (0, 1)) = 0$ and $K(y, (0, 1)) = 0$. So the associated constraint associated with the state $(0, 0)$ in a circle is simply $\mathcal{C}$.*
- $(1, 0)$ *is a successor of* $(0, 0)$ *if $K(x, \emptyset) > 0$. So the set of constraints associated with $(1, 0)$ is $\mathcal{C} \cup \{K(x, \emptyset) > 0\}$.*
- $(0, 0)$ *is a steady state if $(K(x, \emptyset) = 0 \wedge K(y, \emptyset) = 0)$. So the set of constraints associated with the state $(0, 0)$ in a square is $\mathcal{C} \cup \{(K(x, \emptyset) = 0 \wedge K(y, \emptyset) = 0)\}$ which is equivalent to $\mathcal{C} \cup \{(K(x, \emptyset) = 0)\}$ as $K(y, \emptyset) = 0$ is contained in $\mathcal{C}$.*
- $(0, 1)$ *is not a successor of* $(0, 0)$ *because in this case $K(y, \emptyset) > 0$, which is not compatible with the initial constraint $K(y, \emptyset) = 0$.*

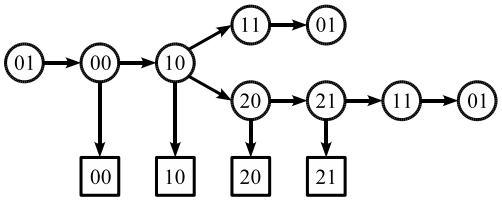

Figure 3: Symbolic execution of the STS of Fig. 2 from the initial state $(0, 1)$. Squares indicate steady states. For simplicity reason, the constraints labeling vertices are not represented in the figure.

Let us point out the reason why the construction of a path of the symbolic execution stops when the new state E has already been encountered in the tree path from the root. Actually, when this case occurs, the path condition of this new state is sufficient to lead to an infinite path repeating the states from E to E. For example in Fig. 3, under the constraints $\mathcal{C}_{01}$ associated to the last state of the path $01 \rightarrow 00 \rightarrow 10 \rightarrow 11 \rightarrow 01$, this path can be repeated infinitely because the constraints that are needed to make the path again are already contained in $\mathcal{C}_{01}$.

Very often, the construction of a path can be terminated before the occurrence of the previous condition (*i.e.* before than the new state has already been encountered in the tree path from the root). Actually, when the couple of the new state and its associated constraints have already been constructed in another path, we can be sure that the possible successors of this couple are precisely the same than the successors of the already constructed state. This case occurs when the same set of parameters leads to the same state by different pathways, which is usual in reason of the asynchronous updating of the variables. In this case the size of the symbolic execution tree can be reduced. The following example illustrates this point.

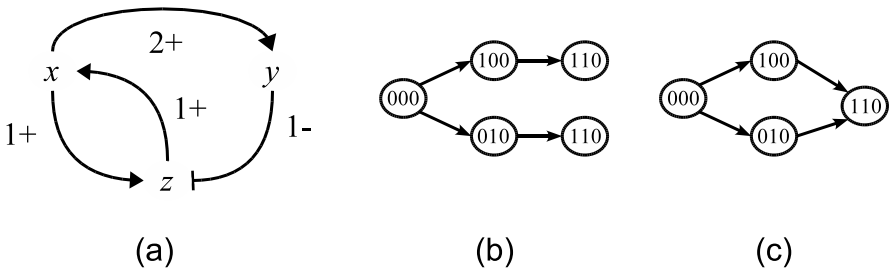

Figure 4: Illustration of the reduction of the symbolic execution. (a) is the interaction graph, (b) a part of the symbolic execution, (c) the same reduced symbolic execution.

Example 7 *We consider the system of three variables x, y, z associated with the interaction graph of Fig. 4(a). Part of the symbolic execution of the associated symbolic model from initial state $(x, y, z) = (0, 0, 0)$ is represented in the same figure (Fig. 4(b)). The condition associated to the path $000 \rightarrow 010 \rightarrow 110$ is $C = (K(y, 000) > 0 \wedge K(x, 010) > 0)$. The path condition of $000 \rightarrow 100 \rightarrow 110$ is $C' = (K(x, 000) > 0 \wedge K(y, 010) > 0)$. But from the interaction graph, we can deduce that $K(x, 000) = K(x, 010) = K(x, \emptyset)$ and that $K(y, 000) = K(y, 010) = K(y, \emptyset)$. Therefore, C and C' can be written $K(x, \emptyset) > 0 \wedge K(y, \emptyset) > 0$. Finally, as $(110, C) = (110, C')$, successors of one couple in the symbolic execution tree are exactly successors of the other; symbolic execution tree can be represented by Fig. 4(c).*

3.3 Specification of paths and synthesis of constraints on parameters

3.3.1 Linear temporal logic

To search a specific path in the symbolic execution tree we adapt model-checking techniques for linear temporal logic (LTL) [23]. Intuitively model-checking techniques consist in exploring all states of a basic model to state whether this model satisfies or not a given temporal logic formula [37].

A LTL formula expresses properties of a path. This logic adds to the classical operators of propositional logic[2] mainly two temporal operators, called Next (N), and Until (U). If f and g are formulas, Nf means that f is true in the following state of the path, and fUg means that f is true in each state of the path, until g becomes true (and g eventually happens). We can then define the operators Finally (F) and Globally (G); Ff means that f eventually happens (and can be written $\top Uf$); Gf means that f is always true (and can be written $\neg F(\neg f)$).

As a LTL formula expresses a property of a single path, there are two ways to use it to express a property of a discrete description. On the one hand we may want to express that *all* paths of the model have the specified behavior; we say that this property is universal. On the other hand, we may want to express that there exists at least a path in the model with the specified behavior; we say that this property is existential. The distinction is important because universal or existential properties can not be treated exactly by the same method (see Sec. 3.3.2).

Examples of temporal properties

Temporal properties of interest in a model include the existence of a path from a given set of states to another one. If for example there is a path from a state where a variable x is at its basal level 0 to a state where x is at its maximal value 2, it means that there is a path verifying $x = 0 \wedge F(x = 2)$, *i.e.* a path such that in its first state $x = 0$ and that eventually reaches a state where $x = 2$. Such properties can be known from experiments or can be hypotheses of interest. We will see in Sec. 4.2.1 examples of such properties.

The negation of the previous properties are also useful: they mean that a given set of states can not be reached from another one. This kind of property is used in Sec. 3.3.3.

Another current property can be the knowledge that a set of states is stable, *i.e.* that when the system is in these states, there is no path going out. This can include steady states, or stable cyclic behaviors. For example in a system of two variables (x, y), if S is the set of stable states, all paths must verify $(x, y) \in S \Rightarrow G((x, y) \in S)$. It means that there is no path verifying $(x, y) \in S \wedge F((x, y) \notin S)$. We will use in Sec. 4.2.1 examples of such properties.

More sophisticated properties can be expressed. For example, we can express that from a given set of states there exists a path such that this set will be infinitely revisited. Such paths verify the property $(x, y) \in S \wedge GF((x, y) \in S)$ (*i.e.* there is a path whose first state is in S, and from all states of the path, S will be reached in the future). This property can also hold for *all paths* beginning in S; then all paths verify $(x, y) \in S \Rightarrow GF((x, y) \in S)$. This is the type of property used in Sec. 3.3.4.

Let us suppose that the set A of states is an attractor of the system and S is its basin of attraction; then from every state in S, the set A will eventually be reached, and the system will then stay in this set A. It means that all paths verify $(x, y) \in S \Rightarrow FG((x, y) \in A)$ (*i.e.* paths beginning in S are such that after a certain time, all their states are in A; or Finally, all states are Globally in A).

[2] As $\neg$ (not), $\wedge$ (and), $\vee$ (or), $\Rightarrow$ (implies), $\top$ (true), $\bot$ (false).

3.3.2 Extended LTL model-checking

We extend classical LTL model-checking techniques designed for basic models to STSs. Just as classical LTL model-checking only considers pertinent paths according to the formula, our method also considers pertinent paths according to the formula, but in our case each state of a path is provided with constraints on parameters. The key point is that a path is eliminated as soon as the conjunction of constraints is no more satisfiable. This leads to a minimal tree construction and gives us the solutions in term of constraints: the disjunction of the path conditions associated to all remaining paths. The resulting constraint represents all parameter valuations compatible with the behavior specified by the formula. To summarize, given a symbolic model $(M, \mathcal{C})$, extended LTL model-checking allows us to compute all initialization functions (*i.e.* parameter valuations) leading to basic models satisfying a LTL formula. In other words, the extended LTL model-checking associates to any LTL formula a characteristic constraint defining the discrete descriptions satisfying it.

Let us remark that the developed technique constructs the disjunction of constraints on possible paths. Then satisfying a LTL formula for a model means that there exists at least a path satisfying the LTL formula. As said before, such a property is qualified as existential. On the contrary we may want to select models whose all paths satisfy the formula (universal property). In such a case the negation of the universal property is unsatisfiable. We have then to specify this impossible behavior as a LTL formula. It suffices to take the negation of the associated constraint to find all models compatible with the universal property. An example is given in next subsection (Sec. 3.3.3).

3.3.3 Adding knowledge to the symbolic model

When considering behaviors, expressed as LTL formulas, supposed to be known to occur in the actual system, we can add the corresponding characteristic constraints $\mathcal{D}$ to the symbolic model $(M, \mathcal{C})$. We get the symbolic model $(M, \mathcal{C} \cup \mathcal{D})$ restricting the set of discrete descriptions.

Example 8 *From a state where AlgU is at its basal level,* P. aeruginosa *will not produce mucus in a common environment, so there is no path from a state where $x = 0$ to a state where $x = 2$. That is clearly an universal property. In order to show that it is not possible to reach $x = 2$ from $x = 0$, we consider the formula $(x = 0) \wedge F(x = 2)$. The associated constraint, generated by our method, and added to initial constraints $\mathcal{C}$ is $K(x, \emptyset) > 1$. The negation is simply $K(x, \emptyset) \leq 1$. All discrete descriptions verifying $\mathcal{C}$ and the latter constraint satisfy the universal property. In the sequel we denote $\mathcal{C}' = \mathcal{C} \cup \{K(x, \emptyset) \leq 1\}$.*

3.3.4 Extracting knowledge from the symbolic model

Let us come back to the two central questions asked in the Introduction: is there any model coherent with a certain hypothetic behavior? Are there behaviors common to all possible models?

The first question consists in specifying the hypothesis with LTL formulas, and finding the associated constraints. When the constraints are not satisfiable, there is no model compatible with the LTL formulas. When they are satisfiable, the solutions of the constraints give all parameter valuations, each one corresponding to a discrete description satisfying the LTL formulas (see example 9).

The second question consists in finding properties common to all discrete descriptions associated to a symbolic model $(M, \mathcal{C})$. The set of constraints $\mathcal{C}$ precisely represents such common properties; then every behavior implied by these constraints is a common behavior to all selected discrete descriptions (see Sec. 4.2.3 for an illustration).

Example 9 *If the hypothesis of an epigenetic change in mucoid* P. aeruginosa *is verified, bacteria which produce mucus can continue to produce mucus in a common environment. A path beginning*

with $x = 2$ which revisits infinitely a state where $x = 2$ is described by the formula $(x = 2) \wedge GF(x = 2)$. The resulting constraint, added to $\mathcal{C}'$, is

$$[K(x, \{x, y\}) = 2 \wedge K(y, \{x\}) = 1] \vee [K(x, \{x\}) = 2 \wedge K(y, \{x\}) = 0]$$

This constraint implies that the (mucoid) state $(2, 1)$ is a steady state, or that $(2, 0)$ is a steady state.

Let us point out that there is another path compatible with $\mathcal{C}$ and verifying $(x = 2) \wedge GF(x = 2)$ (given in Fig. 5). But in this path, $K(x, (1, 0)) > 1$, because there is a transition from the state $(1, 0)$ to the state $(2, 0)$; as $K(x, \emptyset) = K(x, (1, 0))$, it is not compatible with $K(x, \emptyset) \leq 1$, and therefore with $\mathcal{C}'$.

There are 8 discrete descriptions verifying the constraints; in these models the mucoid state can be related to an epigenetic modification. These constraints imply the existence of a stable mucoid state, but not that all paths from a mucoid state come back to a mucoid state. This more restrictive behavior, is achieved if $K(x, \{x, y\}) > 1$, i.e. *for 4 models from the 8.*

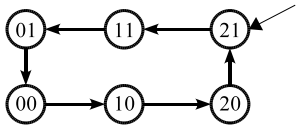

Figure 5: Example of a path of the STS of Fig. 2 verifying $(x = 2) \wedge GF(x = 2)$ (if $(2, 0)$ or $(2, 1)$ is the initial state of the path), compatible with $\mathcal{C}$ but not with $\mathcal{C}'$.

4 Application to immunity control in bacteriophage lambda

4.1 Immunity control in bacteriophage lambda

Bacteriophage lambda is a virus whose DNA can integrate into bacterial chromosome and be faithfully transmitted to the bacterial progeny. After infection, most of the bacteria display a lytic response and liberate new phages, but some display a lysogenic response, *i.e.* survive and carry lambda genome, becoming immune to infection. Figure 6 is the graph of interactions described by Thieffry and Thomas [9] which has also been studied in [38]. Four genes are involved, called cI, cro, cII and N. The states, represented by a vector (cI, cro, cII, N), are in $\{0, 1, 2\} \times \{0, 1, 2, 3\} \times \{0, 1\} \times \{0, 1\}$. Even with the constraints deduced following Sec. 2.2, the associated symbolic model represents 1 008 000 different discrete descriptions.

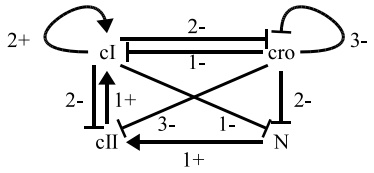

Figure 6: Graph of interactions associated with immunity control in bacteriophage lambda. Arrows are labeled by the threshold and sign of the corresponding interaction. For clarity blunt arrows indicate the negative interactions.

4.2 Lytic and lysogenic pathways of bacteriophage lambda

4.2.1 Specification of behaviors by LTL formulas

First we have to specify the set of states of interest. The lytic response leads to the states where cro is fully expressed, and other genes repressed. So $(0, 2, 0, 0)$ and $(0, 3, 0, 0)$ are called lytic states. To specify that the system is in one of these states, we use the following formula, called $lytic$:

$$lytic = (\text{cI} = 0 \wedge \text{cro} \geq 2 \wedge \text{cII} = 0 \wedge \text{N} = 0).$$

The lysogenic response leads to the state where cI is fully expressed, and the repressor produced by cI blocks the expression of the other viral genes, leading to immunity. So $(2, 0, 0, 0)$ is called lysogenic state. To specify that the system is in this state, we use the following formula, called $lysogenic$:

$$lysogenic = (\text{cI} = 2 \wedge \text{cro} = 0 \wedge \text{cII} = 0 \wedge \text{N} = 0).$$

The viral proteins are initially absent when the viral genome integrates a cell; so the initial state is $(0, 0, 0, 0)$. The system is in this initial state if it verifies the following $init$ formula:

$$init = (\text{cI} = 0 \wedge \text{cro} = 0 \wedge \text{cII} = 0 \wedge \text{N} = 0).$$

When the system reaches the set of lytic state it does not leave it; the stability of these states is an universal property. So we translate this property into the equivalent property $\mathcal{P}_1$:

- $\mathcal{P}_1$: there is no path verifying $lytic \wedge F(\neg lytic)$.

Similarly, the stability of the lysogenic state is an universal property, equivalent to the property $\mathcal{P}_2$:

- $\mathcal{P}_2$: there is no path verifying $lysogenic \wedge F(\neg lysogenic)$.

As lytic and lysogenic responses are possible from the initial state, it means that there exists at least a path from initial state to lytic states, and at least a path from initial state to lysogenic state. These properties are translated into $\mathcal{P}_3$ and $\mathcal{P}_4$:

- $\mathcal{P}_3$: there is a path verifying $init \wedge F(lytic)$;
- $\mathcal{P}_4$: there is a path verifying $init \wedge F(lysogenic)$.

4.2.2 Resulting constraints on parameters

In the sequel $\mathcal{C}_\lambda$ denotes the set of initial constraints associated with the interaction graph of Fig. 6 following the rules described in Sec. 2.2. We apply the extended model-checking method to the associated symbolic model, to find the constraints that have to be added to $\mathcal{C}_\lambda$.

To obtain the additional constraints associated with $\mathcal{P}_1$, we first generate the disjunction of the conditions leading to a path verifying $lytic \wedge F(\neg lytic)$. The negation of this disjunction is:

$$C_1 = [K(\text{cI}, \{\text{cro}\}) = 0 \wedge K(\text{cro}, \emptyset) > 1 \wedge K(\text{cII}, \emptyset) = 0 \wedge K(\text{N}, \{\text{cro}\}) = 0].$$

Similarly the negation of the constraints associated to $init \wedge F(lysogenic)$ is

$$C_2 = [K(\text{cI}, \{\text{cI}\}) = 2 \wedge K(\text{cro}, \{\text{cI}\}) = 0 \wedge K(\text{cII}, \{\text{cI}\}) = 0 \wedge K(\text{N}, \{\text{cI}\}) = 0].$$

These two constraints can be added to $\mathcal{C}_\lambda$ in the symbolic model. The discrete descriptions verifying these constraints verify $\mathcal{P}_1$ and $\mathcal{P}_2$.

By the same method applied on the symbolic model with the constraint $\mathcal{C}_\lambda \cup \{C_1, C_2\}$, we generate the additional constraint needed to verify $\mathcal{P}_3$. This constraint is $\top$ (the always true proposition): it means that all discrete descriptions whose parameters verify $\mathcal{C}_\lambda \cup \{C_1, C_2\}$ have a path verifying $init \wedge F(lytic)$.

Finally, the additional constraint associated with $\mathcal{P}_4$ and $init \wedge F(lysogenic)$ (obtained by disjunction of path conditions) is

$$C_4 = [K(\text{cI}, \emptyset) = 2] \vee [K(\text{cI}, \{\text{cII}\}) = 2 \wedge K(\text{cII}, \{\text{N}\}) = 1 \wedge K(\text{N}, \emptyset) = 1].$$

The discrete descriptions whose parameters verify $\mathcal{C}_\lambda \cup \{C_1, C_2, C_4\}$ are the discrete descriptions associated with immunity control that verify the properties $\mathcal{P}_1$ to $\mathcal{P}_4$.

4.2.3 Questioning the symbolic model

In this subsection we show that there are pathways to lysis or lysogeny common to all discrete descriptions whose parameters verify $\mathcal{C}_\lambda \cup \{C_1, C_2, C_4\}$. For simplicity, the states of values of $(\text{cI}, \text{cro}, \text{cII}, \text{N})$ are denoted by (0000), (0100), etc.

In all these discrete descriptions $K(\text{cro}, \emptyset) > 1$ (it is a consequence of C_1). But $K(\text{cro}, 0000) = K(\text{cro}, 0100)$ as these parameters are equal to $K(\text{cro}, \emptyset)$; so they are at least equal to 2. So it is clearly a sufficient condition to demonstrate that in all discrete descriptions there is the following path to lysis:

$$(0000) \rightarrow (0100) \rightarrow (0200) \tag{1}$$

The constraint C_4 is

$$[K(\text{cI}, \emptyset) = 2] \vee [K(\text{cI}, \{\text{cII}\}) = 2 \wedge K(\text{cII}, \{\text{N}\}) = 1 \wedge K(\text{N}, \emptyset) = 1].$$

So all discrete descriptions verify at least one of the properties C or C':

- $C = [K(\text{cI}, \emptyset) = 2]$;
- $C' = [K(\text{cI}, \{\text{cII}\}) = 2 \wedge K(\text{cII}, \{\text{N}\}) = 1 \wedge K(\text{N}, \emptyset) = 1]$.

First we look at the discrete descriptions verifying the first constraint C. As $K(\text{cI}, \emptyset) = K(\text{cro}, 0000) = K(\text{cro}, 1000)$, all discrete descriptions such that $K(\text{cI}, \emptyset) = 2$ have the following path to lysogeny:

$$(0000) \rightarrow (1000) \rightarrow (2000) \tag{2}$$

Now we consider the second constraint C'.

- As $K(\text{N}, \emptyset) = 1$, there is a transition $(0000) \rightarrow (0001)$.
- As $K(\text{cII}, \{\text{N}\}) = 1$, and $K(\text{cII}, \{\text{N}\}) = K(\text{cII}, 0001)$, there is a transition $(0001) \rightarrow (0011)$.
- As $K(\text{cI}, \{\text{cII}\}) = 2$, and $K(\text{cI}, \{\text{cII}\}) = K(\text{cI}, 0011) = K(\text{cI}, 1011)$, there is a path $(0011) \rightarrow (1011) \rightarrow (2011)$.
- The constraint C_2 implies that $K(\text{N}, \{\text{cI}\}) = 0$, then $K(\text{N}, 2011) = 0$. So there is a transition $(2011) \rightarrow (2010)$.
- The constraint C_2 implies that $K(\text{cII}, \{\text{cI}\}) = 0$, so $K(\text{cII}, 2010) = 0$. So there is a transition $(2010) \rightarrow (2000)$.

Therefore all discrete descriptions verifying C' have the following path to lysogeny:

$$(0000) \rightarrow (0001) \rightarrow (0011) \rightarrow (1011) \rightarrow (2011) \rightarrow (2010) \rightarrow (2000) \quad (3)$$

Interestingly, this last path is precisely the most likely pathway to lysogeny according to experimental knowledge, as described by Thieffry and Thomas [9].

A precise count of the number of discrete descriptions reveals that there are 2156 discrete descriptions verifying $\mathcal{C}_\lambda \cup \{C_1, C_2, C_4\}$. In all these discrete descriptions, there is a common pathway from initial state to lysis: pathway (1). There are 1176 of these discrete descriptions verifying C. They are discrete descriptions with a common pathway to lysogeny: pathway (2). Moreover there are 1470 discrete descriptions (out of the 2156) verifying C'; they have the common pathway (3) to lysogeny. 490 discrete descriptions verify C *and* C': they are the discrete descriptions with at least two different pathways to lysogeny, pathway (2) and pathway (3).

5 Conclusion

We have shown how a symbolic model representing a set of possible discrete descriptions of a genetic regulatory network permits one to deal with incomplete knowledge. Known interactions can be translated into constraints on the parameters, which can be specified in a symbolic transition system. This STS can be simulated with symbolic execution techniques. The known behaviors can be specified with LTL formulas, and then, model-checking techniques have been extended to select the constraints on parameters associated with these behaviors. Adding these contraints to the STS, a symbolic model representing all discrete descriptions coherent with the known behaviors is obtained.

Then we have explained how the symbolic model can be used to reveal new results: the possibility of hypothetic behaviors can be tested (as the epigenetic change in *P. aeruginosa*) or common behaviors between all selected descriptions can be found (as possible pathways to lysis or lysogeny in bacteriophage lambda).

By using SMBioNet to analyze the regulatory network of the cytotoxicity of *P. aeruginosa* [33], models coherent with the hypothesis of the existence of an epigenetic switch between non-inducible states and inducible ones have been constructed. The underlying interaction graph used was similar to the interaction graph associated with mucus production (in Fig. 1). This theoretical results have lead to new experimental results [39]. It is now interesting to take into account these new results into a more elaborated model, in particular by including other important proteins implicated in the network. The efficiency of the methods presented in this chapter should allow us to construct and analyze this more complex model. It is a work that we plan to do in the context of the observability working group of Epigenomics Project of Genopole®, Evry.

Acknowledgements

We gratefully acknowledge the members of the the observability working group of Genopole® for stimulating interactions. In particular Gilles Bernot and Janine Guespin have encouraged us in the development of symbolic formal methods for systems biology. We thank also Epigenomics Project for constant support.

References

[1] H. de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *J Comput Biol*, 9(1):67–103, 2002.

[2] S. A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *J Theor Biol*, 22(3):437–67, 1969.

[3] R. Thomas. Boolean formalization of genetic control circuits. *J Theor Biol*, 42(3):563–585, 1973.

[4] R. Thomas and R. D'Ari. *Biological Feedback*. CRC Press, Boca Raton, Florida, 1990.

[5] L. Glass and S.A. Kauffman. The logical analysis of continuous non-linear biochemical control networks. *J Theor Biol*, 39:103–129, 1973.

[6] R. Thomas. Regulatory networks seen as asynchronous automata : a logical description. *J Theor Biol*, 153(1):1–23, 1991.

[7] R. Thomas and M. Kaufman. Multistationarity, the basis of cell differentiation and memory. ii. logical analysis of regulatory networks in terms of feedback circuits. *Chaos*, 11(1):180–95, 2001.

[8] E.H. Snoussi. Qualitative dynamics of piecewise-linear differential equations : A discrete mapping approach. *Dyn Stability Syst*, 4:189–207, 1989.

[9] D. Thieffry and R. Thomas. Dynamical behaviour of biological regulatory networks –ii. immunity control in bacteriophage lambda. *Bull Math Biol*, 57(2):277–97, 1995.

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

[11] L. Sanchez and D. Thieffry. Segmenting the fly embryo: a logical analysis of the pair-rule cross-regulatory module. *J Theor Biol*, 224(4):517–37, 2003.

[12] L. Mendoza. A network model for the control of the differentiation process in th cells. *Biosystems*, 84(2):104–14, 2006.

[13] C. Chaouiya, D. Thieffry, and L. Sanchez. From gradients to stripes: a logical analysis of drosophila segmentation genetic network. In N. Kolchanov, R. Hofestaedt, and L. Milanesi, editors, *Bioinformatics of Genome Regulation and Structure II*, pages 379–90. Springer, 2006.

[14] H. de Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bull Math Biol*, 66(2):301–340, 2004.

[15] E.H. Snoussi and R. Thomas. Logical identification of all steady states: the concept of feedback characteristic states. *Bull Math Biol*, 55:973–991, 1993.

[16] R. Thomas, D. Thieffry, and M. Kaufman. Dynamical behaviour of biological regulatory networks –i. biological role of feedback loops and practical use of the concept of the loop-characteristic state. *Bull Math Biol*, 57(2):247–76, 1995.

[17] A.G. Gonzalez, A. Naldi, L. Sanchez, D. Thieffry, and C. Chaouiya. Ginsim: A software suite for the qualitative modelling, simulation and analysis of regulatory networks. *Biosystems*, 84(2):91–100, 2006.

[18] E. Fanchon, F. Corblin, L. Trilling, B. Hermant, and D. Gulino. Modeling the molecular network controlling adhesion between human endothelial cells: Inference and simulation using constraint logic programming. In *Proceedings of Computational Methods in Systems Biology (CMSB 2004)*, volume 3082 of *Lecture Notes in Computer Science*, pages 104–118. Springer, 2005.

[19] M.A. Schaub, T.A. Henzinger, and J. Fisher. Qualitative networks: a symbolic approach to analyze biological signaling networks. *BMC Systems Biology*, 1(4), 2007.

[20] G. Bernot, J.-P. Comet, A. Richard, and J. Guespin. Application of formal methods to biological regulatory networks: extending thomas' asynchronous logical approach with temporal logic. *J Theor Biol*, 229(3):339–347, 2004.

[21] G. Batt, D. Ropers, H. de Jong, J. Geiselmann, R. Mateescu, M. Page, and D. Schneider. 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.

[22] M. Aiguier, C. Gaston, P. Le Gall, D. Longuet, and A. Touil. A temporal logic for input output symbolic transition systems. In *Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSEC'05)*, pages 43–50. IEEE Computer Society Press, 2005.

[23] M.Y. Vardi. An automata-theoretic approach to linear temporal logic. In *Proceedings of the Banff Higher order workshop conference on Logics for concurrency : structure versus automata*, pages 238–266, Secaucus, NJ, USA, 1996. Springer-Verlag New York.

[24] D. Mateus, J.-P. Gallois, and P. Le Gall. Évaluation symbolique appliquée à l' étude de réseaux de régulation génétique. In *Journée thématique RIAMS, Réseaux d'interaction : analyse, modélisation et simulation*, pages 19–24. Rapport de Recherche LaMI 121, 2005.

[25] D. Mateus and J.-P. Gallois. Searching constraints in biological regulatory networks using symbolic analysis. In *Proceedings of the fifth international conference on Bioinformatics of Genome Regulation and Structure (BGRS'06)*, volume 3, pages 78–81, 2006.

[26] D. Mateus, J.-P. Comet, J.-P. Gallois, and P. Le Gall. Modeling genetic regulatory networks from specified behaviors. Oral presentation at Integrative Post-Genomics (IPG'06), Lyon, France, December 2006.

[27] D. Mateus, J.-P. Comet, J.-P. Gallois, and P. Le Gall. Modelling genetic regulatory networks from specified behaviours. Oral presentation at BioSysBio'07, Manchester, UK, January 2007.

[28] D. Mateus, J.-P. Comet, J.-P. Gallois, and P. Le Gall. Symbolic modeling of genetic regulatory networks. *J Bioinform Comput Biol*, (to appear), 2007.

[29] C. Bigot, A. Faivre, J.-P. Gallois, A. Lapitre, D. Lugato, J.-Y. Pierron, and N. Rapin. Automatic test generation with agatha. In *Proceedings of Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2003)*, volume 2619 of *Lecture Notes in Computer Science*, pages 591–596. Springer, 2003.

[30] D. Lugato, C. Bigot, Y. Valot, J.-P. Gallois, S. Gérard, and F. Terrier. Validation and automatic test generation on uml models: the agatha approach. *International Journal on Software Tools for Technology Transfer (STTT)*, 5(2-3):124–139, 2004.

[31] J.R. Govan and V. Deretic. Microbial pathogenesis in cystic fibrosis: mucoid pseudomonas aeruginosa and burkholderia cepacia. *Microbiol rev*, 60(3):539–74, 1996.

[32] J. Guespin-Michel and M. Kaufman. Positive feedback circuits and adaptive regulations in bacteria. *Acta Biotheor*, 49(4):207–218, 2001.

[33] J. Guespin-Michel, G. Bernot, J.-P. Comet, A. Mérieau, A. Richard, C. Hulen, and B. Polack. Epigenesis and dynamic similarity in two regulatory networks in pseudomonas aeruginosa. *Acta Biotheor*, 52(4):379–390, 2004.

[34] J.C. King. Symbolic execution and program testing. *Commun ACM*, 19(7):385–394, 1976.

[35] N. Rapin, C. Gaston, A. Lapitre, and J.-P. Gallois. Behavioral unfolding of formal specifications based on communicating extended automata. In *Proceedings of the first international workshop on Automated Technology for Verification and Analysis (ATVA'03)*, National Taipei University, Taiwan, 2003.

[36] C. Gaston, P. Le Gall, N. Rapin, and A. Touil. Symbolic execution techniques for test purpose definition. In *Proceedings of Testing of Communicating Systems (TestCom 2006)*, volume 3964 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2006.

[37] E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, Cambridge, Mass., 2000.

[38] A. Richard, J.-P. Comet, and G. Bernot. Formal methods for modeling biological regulatory networks. In A.H. Gabbar, editor, *Modern Formal Methods and Applications*, pages 83–122. Springer, 2006.

[39] D. Filopon, A. Mérieau, G. Bernot, J.-P. Comet, R. Leberre, B. Guery, B. Polack, and J. Guespin-Michel. Epigenetic acquisition of inducibility of type iii citoxicity in p. aeruginosa. *BMC Bioinformatics*, 7(272), 2006.