Context Sensitivity in Logical Modeling with Time Delays

Heike Siebert and Alexander Bockmayr

DFG Research Center Matheon,
Freie Universität Berlin, Arnimallee 3, D-14195 Berlin, Germany
siebert@mi.fu-berlin.de, bockmayr@mi.fu-berlin.de

Abstract. For modeling and analyzing regulatory networks based on qualitative information and possibly additional temporal constraints, approaches using hybrid automata can be very helpful. The formalism focussed on in this paper starts from the logical description developed by R. Thomas to capture network structure and qualitative behavior of a system. Using the framework of timed automata, the analysis of the dynamics can be refined by adding a continuous time evolution. This allows for the incorporation of data on time delays associated with specific processes. In general, structural aspects such as character and strength of interactions as well as time delays are context sensitive in the sense that they depend on the current state of the system. We propose an enhancement of the approach described above, integrating both structural and temporal context sensitivity.

1 Introduction

Logical modeling of bioregulatory networks started more than thirty years ago with the work of Sugita, Kauffman, Glass, and Thomas [11, 5, 4, 13]. R. Thomas [13] introduced a logical formalism in the 1970s, which, over the years, has been further developed and successfully applied to different biological problems (see [14], [15] and references therein). Network components are modeled as discrete variables, the values of which correspond to the different expression levels of the component. In the simplest case, there are only two expression levels, 0 and 1, representing for instance whether or not a substance has surpassed some threshold concentration associated with a network interaction. A directed graph is used to represent interactions between the network components. Edges are labeled with signs corresponding to the character of the interaction, i. e., activating or inhibiting, and information on the thresholds associated with the interaction. In order to derive the dynamics of the system, parameters are introduced that take discrete values and determine for each network state the influence of enabled interactions on the system's behavior. To obtain a deterministic behavior, one would need sufficient data on the time delays associated with the different changes of expression level for all network components. In the classical Thomas formalism, the only assumption made is that all those time delays are different, resulting in a non-deterministic, asynchronous description of the dynamics. If more temporal data, e. g. in the form of time constraints, is available, we need a refined modelling approach allowing for the incorporation of such data. In [9] we proposed such a formalism based on the theory of timed automata. Here, each component is equipped with a clock used to evaluate conditions imposed on the time delays of this component during the evolution of the system.

In both approaches described so far the assumption is made that the information used for modeling is valid regardless of the state of the system. This assumption is often not met in reality. We have to deal with *context sensitivity* in the sense that characteristics of network interactions as well as time delays associated with changes in expression level may depend on the current state of the system. Whether some substance has an activating or inhibiting influence on the transcription of a gene, for example, may very well depend on the concentration of that substance. This is an example for *structural context sensitivity.* Moreover, the time delay for some component a to reach expression level 1 from 0 may be significantly shorter if the expression level change follows from activation by component b rather than from some influence exerted by component c, indicating *temporal context sensitivity.* Both kinds of phenomena may be of crucial importance for the system's functional purpose. Changes in the modeling formalism pertaining to both the logical modeling of the network structure and the parameter definition as well as the construction of the timed automaton model are necessary to capture the behavior of such context sensitive systems.

Specification of a model comprises three levels, allowing for a stepwise extension of the model in accordance with the available data. All the structural and discrete dynamical information about the network can be expressed in a context sensitive logical modeling framework based on the classical Thomas formalism. We develop this framework in Section 2. In a second step, discussed in Section 4, we show the translation of the context sensitive Thomas model into the framework of timed automata. At this stage we are able to incorporate information on time delays, using the properties of timed automata we introduced in Section 3. The resulting modeling approach constitutes a generalization of the one presented in [9]. As a final contribution we present a way to integrate context sensitive time delays in Section 5. We illustrate our methodology by examples, in particular the analysis of a regulatory network of bacteriophage λ, which we have implemented using the verification tool UPPAAL. We end the paper by discussing problems and perspectives of our approach

2 Context Sensitive Thomas Formalism

In this section we give a formal definition of a regulatory network based on the modeling approach of R. Thomas (see for example [14] and [15]). In contrast to Thomas' original approach our formalism allows for a rigorous description of a context sensitive system. The product of the *cI* gene in bacteriophage λ, for example, activates the *cI* gene by positive autoregulation. However, as the product concentration increases, the influence on its gene becomes inhibiting, thus preventing overexpression. In the classical Thomas formalism the interaction of

cI with itself is represented by a signed edge in the network graph, the sign characterizing the interaction as activating or inhibiting. In the described situation neither choice of sign reflects reality. It is still possible to choose the parameter values governing the system's dynamics to accommodate both activating and inhibiting effects, but that renders the edge sign meaningless. Since the edge signs play an important role in formulating general mathematical relations between motifs in the network graph and possible dynamical behavior, it seems important to keep them in the model of the network structure. Thus, we represent interactions capable of displaying activating as well as inhibiting effects by two edges, one negative, the other positive. In other words, while Thomas uses a directed graph to capture the structure of a network, we use a directed multigraph allowing for parallel edges. Consequently, we also need to alter Thomas' method of determining the system's dynamics from the graph. Multigraphs have already been used in [3] in a similar way. Also, multigraphs are used in [7] and [8], however, those graphs are derived from given discrete functions describing the system's behavior.

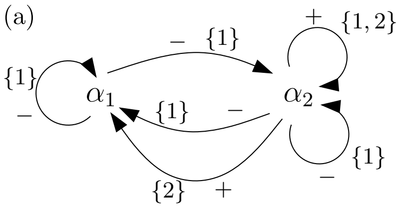
Definition 1. *An* interaction (multi-)graph *(or bioregulatory (multi-)graph) $\mathcal{I}$ is a labeled directed multigraph with*

- *vertex set $V := \{\alpha_1, \ldots, \alpha_n\}, n \in \mathbb{N}$, and*
- *edge (multi-)set $E \subseteq V \times V$. Let $\mathcal{T}(\alpha_j)$ be the set of all edges whose tail is α_j, and $\mathcal{H}(\alpha_i)$ the set of edges whose head is α_i. Each edge e from α_j to α_i is labeled with a sign $\varepsilon_e \in \{+, -\}$ and a set $M_e \subseteq \{1, \ldots, d_j\}$, $d_j \in \mathbb{N}$. For each sign $\varepsilon \in \{+, -\}$ there is at most one edge $\alpha_j \to \alpha_i$ labeled with ε.*

Let p_j be the maximal value of the union of all sets M_e with $e \in \mathcal{T}(\alpha_j)$. We call $\{0, \ldots, p_j\}$ the range *of α_j. For each $i \in \{1, \ldots, n\}$ we denote by $Pred(\alpha_i)$ the set of vertices α_j such that $\alpha_j \to \alpha_i$ is an edge in E.*

The vertices of this graph represent the components of the regulatory network, e. g. genes, the range of a vertex the different expression levels of the corresponding component affecting the behavior of the network. For example, if we only consider a component to be active or inactive, its range is $\{0, 1\}$. Thus, the vertices can be interpreted as variables that take values in the corresponding range. An edge e from α_j to α_i signifies that α_j influences α_i in a positive or negative way, depending on ε_e and provided that the current expression level of α_j is in M_e. If there is more than one edge leading from α_j to α_i, then the character of the interaction depends on the context. Figure 1 (a) shows a simple interaction graph comprising two vertices. The two different edges leading from α_2 to α_1 signify that α_2 has an inhibiting influence on α_1 if its expression level is 1. However, on the higher expression level 2 the influence becomes activating. Furthermore, there are two edges, e_3 and e_4, from α_2 to itself. The corresponding sets M_{e_3} and M_{e_4} intersect. It is not clear from the interaction graph alone, whether α_2 with expression level 1 inhibits or activates itself. The outcome depends on the interplay between the influence from α_2 on itself and the influence of α_1 on α_2. Thus, in order to determine the dynamical behavior of the system we need further information.

(a)



$e_1 := (\alpha_1, \alpha_1, -, \{1\}),$
$e_2 := (\alpha_1, \alpha_2, -, \{1\}),$
$e_3 := (\alpha_2, \alpha_2, +, \{1, 2\}),$
$e_4 := (\alpha_2, \alpha_2, -, \{1\}),$
$e_5 := (\alpha_2, \alpha_1, -, \{1\}),$
$e_6 = (\alpha_2, \alpha_1, +, \{2\})$

(b)

α_1	α_2	α_1'	α_2'	α_1'	α_2'
0	0	$K_{1,\{e_1,e_5\}}$	$K_{2,\{e_2,e_4\}}$	1	1
1	0	$K_{1,\{e_5\}}$	$K_{2,\{e_4\}}$	0	0
0	1	$K_{1,\{e_1\}}$	$K_{2,\{e_2,e_3\}}$	0	2
1	1	$K_{1,\emptyset}$	$K_{2,\{e_3\}}$	0	0
0	2	$K_{1,\{e_1,e_5,e_6\}}$	$K_{2,\{e_2,e_3,e_4\}}$	1	2
1	2	$K_{1,\{e_5,e_6\}}$	$K_{2,\{e_3,e_4\}}$	1	1

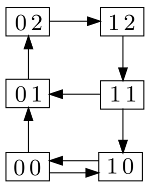


Fig. 1. In (a), interaction graph of a regulatory system comprising two components and six interactions. In (b), state table for general parameters with specific values, and the resulting state transition graph.

Definition 2. *Let $\mathcal{I}$ be an interaction graph. A* state *of the system described by $\mathcal{I}$ is a tuple $s \in S^n := \{0, \dots, p_1\} \times \dots \times \{0, \dots, p_n\}$. The set of* resource edges *$R_i(s)$ of α_i in state s is the set*

$$\{e \in \mathcal{H}(\alpha_i) \mid (\varepsilon_e = + \wedge s_j \in M_e) \vee (\varepsilon_e = - \wedge s_j \notin M_e)\}.$$

Finally, given a set

$$K(\mathcal{I}) := \{K_{i,R_i(s)} \mid i \in \{1, \dots, n\},\ s \in S^n\}$$

of (logical) parameters *$K_{i,R_i(s)}$, which take values in the range of α_i, we call the pair $(\mathcal{I}, K(\mathcal{I}))$ a* bioregulatory network*.*

In a given state s only the edges e with $s_{tail(e)} \in M_e$ represent active influences. The set of edge resources $R_i(s)$ contains all active positive edges and all inactive negative edges reaching α_i. That is, we interpret the absence of an inhibiting influence as activating influence. The value of the parameter $K_{i,R_i(s)}$ then indicates how the expression level of α_i will evolve. It will increase (resp. decrease) if the parameter value is greater (resp. smaller) than s_i. The expression level stays the same if both values are equal.

Typically, the set of resources $R_i(s)$, and with it the logical parameters, is defined as the set of predecessors of α_i (instead of edges reaching α_i) having an activating influence on α_i (see [2]). Since we allow for context sensitivity, knowledge of the current expression level of a predecessor of α_i is not enough to determine the character of the corresponding interaction. In the table given in Figure 1(b) we see in the second column the logical parameters determining the state s' the system will evolve to from the state s given in the first column. The third column provides a specification of the parameter values. The resulting network behavior cannot be described in the classical Thomas formalism, since it is highly context sensitive. For example, we see that if α_2 has expression level 1, it activates itself in the absence of α_1, since $K_{2,\{e_2,e_3\}} = 2$. If α_1 is present, resulting in an effective inhibiting edge e_2, then α_2 inhibits itself, as indicated by the parameter choice $K_{2,\{e_3\}} = 0$. For an interaction graph without parallel edges, the notion of edge resources and vertex resources are equivalent.

The signs on the edges together with the sets M_e determine whether a component is an activator or an inhibitor of some other component in a given state. An activating influence, i. e., an effective activator or a non-effective inhibitor, cannot induce a decrease in expression level of the target component. This is reflected in the following parameter constraint:

$$\omega \subseteq \omega' \subseteq \mathcal{H}(\alpha_i) \Rightarrow K_{i,\omega} \leq K_{i,\omega'} \tag{1}$$

for all $i \in \{1, \ldots, n\}$. In the following we will always assume that this constraint is satisfied.

To conclude this section, we describe the dynamics of the system by means of a state transition graph.

Definition 3. *The* state transition graph $\mathcal{S}_N$ *corresponding to the bioregulatory network* $N = (\mathcal{I}, K(\mathcal{I}))$ *is a directed graph with vertex set* S^n. *There is an edge* $s \rightarrow s'$ *if there is* $i \in \{1, \ldots, n\}$ *such that* $s'_i = s_i + sgn(K_{i,R_i(s)} - s_i) \neq s_i$ *and* $s_j = s'_j$ *for all* $j \in \{1, \ldots, n\} \setminus \{i\}$.

To describe the dynamics of the system we use the so-called asynchronous update, i. e., a state differs from a successor state in one component only. If s is a state such that an evolution in more than one component is indicated, then there will be more than one successor of s. Note that s is a steady state if s has no outgoing edge. In Figure 1(b) we see the state transition graph corresponding to the state table also given in the figure.

3 Timed Automata

In this section we formally introduce timed automata. We mainly use the definitions and notations given in [1]. To introduce the concept of time in our system, we consider a set $C := \{c_1, \ldots, c_n\}$ of real variables that behave according to the differential equations $\dot{c}_i = 1$. These variables are called *clocks*. They progress synchronously and can be reset to zero under certain conditions. We define the set $\Phi(C)$ of *clock constraints* φ by the grammar

$$\varphi ::= c \leq q \,|\, c \geq q \,|\, c < q \,|\, c > q \,|\, \varphi_1 \wedge \varphi_2 \,,$$

where $c \in C$ and q is a rational constant.

A *clock interpretation* is a function $u : C \rightarrow \mathbb{R}_{\geq 0}$ from the set of clocks to the non-negative reals. For $\delta \in \mathbb{R}_{\geq 0}$, we denote by $u + \delta$ the clock interpretation that maps each $c \in C$ to $u(c) + \delta$. For $Y \subseteq C$, we indicate by $u[Y := 0]$ the clock interpretation that maps $c \in Y$ to zero and agrees with u over $C \setminus Y$. A clock interpretation u satisfies a clock constraint φ if $\varphi(u) = true$. The set of all clock interpretations is denoted by $\mathbb{R}^C_{\geq 0}$.

Definition 4. *A* timed automaton *is a tuple* $(L, L^0, \Sigma, C, I, E)$*, where*

- L *is a finite set of* locations*,*
- $L^0 \subseteq L$ *is the set of* initial locations*,*
- Σ *is a finite set of* events *(or labels),*
- C *is a finite set of* clocks*,*
- $I : L \to \Phi(C)$ *is a mapping that labels each location with some clock constraint called the* invariant *of the location,*
- $E \subseteq L \times \Sigma \times \Phi(C) \times 2^C \times L$ *is a set of* switches.

A timed automaton can be represented as a directed graph with vertex set L. The vertices are labelled with the corresponding invariants and are marked as initial locations if they belong to L^0. The edges of the graph correspond to the switches and are labelled with an event, a clock constraint called *guard* specifying when the switch is enabled, and a subset of C comprising the clocks that are reset to zero with this switch. While switches are instantaneous, time may elapse in a location. To describe the dynamics of such an automaton formally, we use the notion of a transition system.

Definition 5. *Let A be a timed automaton. The* (labelled) transition system T_A *associated with A is a tuple $(Q, Q^0, \Gamma, \to)$, where Q is the set of states $(l, u) \in L \times \mathbb{R}^C_{\geq 0}$ such that u satisfies the invariant $I(l)$, Q^0 comprises the states $(l, u) \in Q$ where $l \in L^0$ and u ascribes the value zero to each clock, and $\Gamma := \Sigma \cup \mathbb{R}_{\geq 0}$. Moreover, $\to \subseteq Q \times \Gamma \times Q$ is defined as the set comprising*

- *transitions* $(l, u) \xrightarrow{\delta} (l, u + \delta)$ *for* $\delta \in \mathbb{R}_{\geq 0}$ *such that for all* $0 \leq \delta' \leq \delta$ *the clock interpretation* $u + \delta'$ *satisfies the invariant* $I(l)$*, and*
- *transitions* $(l, u) \xrightarrow{a} (l', u[R := 0])$ *for* $a \in \Sigma$ *such that there is a switch* (l, a, φ, R, l') *in* E*,* u *satisfies* φ*, and* $u[R := 0]$ *satisfies* $I(l')$.

The elements of $\to$ *are called* transitions.

The first kind of transition is a state change due to elapse of time, while the second one is due to a location-switch and is called *discrete*. Again we can visualize the object T_A as a directed graph with vertex set Q and edges corresponding to the transitions given by $\to$. Note, that by definition the set of states may be infinite and that the transition system is in general nondeterministic, i. e., a state may have more than one successor. Moreover, it is possible that a state is the source for edges labelled with a real value as well as for edges labelled with events. However, although every discrete transition corresponds to a switch in A, there may be switches in A that do not lead to a transition in T_A. That is due to the additional conditions placed on the clock interpretations.

Finally, we obtain a modified transition system by considering only the location vectors as states, dropping all transitions labelled with real values, but keeping every discrete transition of T_A. We call this the *discrete (or symbolic) transition system* of A.

4 Augmenting the Context Sensitive Thomas Formalism with Time Delays

We now present a generalization of the modeling approach using timed automata introduced in [9] suitable for regulatory networks displaying structural context sensitivity. Basically, we model each component of the system individually as a timed automaton, and then present a procedure to combine those elements to a timed automaton capturing the dynamical behavior of the system. This is done much in the same way a product automaton is derived from n timed automata (see [1]) and has been explained in detail in [9] and [10]. We illustrate the procedure and in particular the differences occurring due to the incorporation of context sensitivity using the example in Figure 1. Rigorous definitions will of course also be given for the alterations necessary in this more general approach.

4.1 Modeling the Components

We start modeling the components α_1 and α_2 of the system given in Figure 1 as timed automata A_1 and A_2.

Clocks. We use a single clock c_i for each component in order to measure the time needed for changes in expression level of that component. Even when introducing context sensitive time delays later on, we do not need more than one clock.

Locations. For each A_i we need a set of locations L_i. We introduce locations α_i^k for k in the range of α_i representing a situation where α_i maintains expression level k. They are called *regular* locations. Since we want to measure time delays corresponding to the increase or decrease of expression level, we furthermore define locations α_i^{k+} (resp. α_i^{k-}) for $k \in \{0, \ldots, p_i - 1\}$ (resp. $k \in \{1, \ldots, p_i\}$) indicating that the expression level is still k but is in the process of increasing (resp. decreasing). We call them *intermediate* locations. Lastly, we define $L_i^0 := \{\alpha_i^k\,;\, k \in \{0, \ldots, p_i\}\}$. In Figure 2 the four locations of A_1 and the seven locations of A_2 are drawn as ellipses.

Invariants. The invariants of regular and intermediate locations are fundamentally different. Whether or not the component remains in a regular location does not depend on how much time has passed since it entered that location. In that sense regular locations are stable. This is reflected by assigning the invariant $c_i \geq 0$, which is true for every clock value, to every location α_i^k. In contrast, the intermediate locations are of transient character. The component will leave an intermediate location when the time needed for the corresponding expression level change has passed. We assign the location $\alpha_i^{k\varepsilon}$ the invariant $c_i \leq T_i^{k\varepsilon}$, $\varepsilon \in \{+, -\}$, where $T_i^{k\varepsilon} \in \mathbb{Q}_{\geq 0}$ denotes the maximal time delay of the corresponding expression level change. In Figure 2 the invariant of each location is given in the line beneath the location name.

Switches and events. In the same way we use the invariants in the intermediate locations to include maximal time delays, we use the guards of the switches in E_i to introduce the minimal time delay $t_i^{k\varepsilon} \in \mathbb{Q}_{\geq 0}$ needed for an expression level change. For all $k \in \{0, \ldots, p_i - 1\}$, we have $(\alpha_i^{k+}, a_i^{k+}, \varphi_i^{k+}, \{c_i\}, \alpha_i^{k+1}) \in E_i$, with $\varphi_i^{k+} = (c_i \geq t_i^{k+)})$, representing increase of expression level. Furthermore, for $l \in \{1, \ldots, p_i\}$, the switch $(\alpha_i^{l-}, a_i^{l-}, \varphi_i^{l-}, \{c_i\}, \alpha_i^{l-1})$ with $\varphi_i^{l-} = (c_i \geq t_i^{l-})$ belongs to E_i and represents expression level decrease. Every switch entails a reset of the component clock. The events $a_i^{k\varepsilon} \in \Sigma_i$ will be used later to identify location changes due to elapse of time, and thus correspond to the switches in E_i. We set $\Sigma_i := \{a_i^{k+}, a_i^{m-} \;;\; k \in \{0, \ldots, p_i - 1\}, m \in \{1, \ldots, p_i\}\}$.

The automaton A_1 has only two switches, A_2 only four, as can be seen in Figure 2. Thus it is clear that the dynamics of the system given in Figure 1 is not yet captured by the automata A_1 and A_2. This does not surprise since we have not incorporated the way both components interact with each other. In a next step we translate the information inherent in the interaction graph and the parameter values of the system in Figure 1 into conditions determining when a location change, independent of clock values, should occur in A_i. We call the resulting conditions *switch conditions*. Note that they can only be evaluated in the network context since they generally depend on the expression level of more than one component.

Switch conditions. The definition of the switch conditions deviates from the one given in [9] and [10], despite appearing similar due to notation similar to that in the earlier papers. However, we have to keep in mind that we use the context sensitive version of the Thomas formalism resulting in basic conceptual differences. Here, we give the new definition for the general situation of a network comprising n components, i. e., n automata $A_i = (L_i, L_i^0, \Sigma_i, C_i, I_i, E_i)$ defined as above.

To formulate the switch conditions, we need to know how to obtain from a location the expression level of the corresponding component. We use the function $\iota : \bigcup_{j \in \{1, \ldots, n\}} L_j \to \mathbb{N}_0$ that maps the locations α_j^k, α_j^{k+} and α_j^{k-} to k. Let $k \in \{1, \ldots, p_i - 1\}$ and consider a location of A_i that represents expression level k. First we determine the resource edges (see Def. 2) that influence the behavior of A_i in this location. For every edge $e \in \mathcal{H}(\alpha_i)$ with tail α_j and l_j a location of A_j let

$$\lambda_i^e(l_j) := \begin{cases} \iota(l_j) \in M_e\,, & \varepsilon_{ij} = + \\ \iota(l_j) \notin M_e\,, & \varepsilon_{ij} = - \end{cases}, \qquad \overline{\lambda}_i^e(l_j) := \begin{cases} \iota(l_j) \notin M_e\,, & \varepsilon_{ij} = + \\ \iota(l_j) \in M_e\,, & \varepsilon_{ij} = - \end{cases}.$$

Thus, if $\lambda_i^e(l_j)$ evaluates to *true*, then e is a resource edge if the system is in location l_j of A_j (and thus α_j has expression level k). If the negation is true, e is no resource edge in location l_j. We are now interested in the sets of resource edges that effect a change in the expression level, and thus the location, of A_i. This information lies in the parameter values. Let $\omega_1, \ldots, \omega_{m_i^1}, \upsilon_1, \ldots, \upsilon_{m_i^2}$ be the subsets of $\mathcal{H}(\alpha_i)$ such that the parameter inequalities $K_{i,\omega_h} > k$ for all $h \in \{1, \ldots, m_i^1\}$ as well as $K_{i,\upsilon_h} < k$ for all $h \in \{1, \ldots, m_i^2\}$ hold. In our example, if we choose location α_2^1, i. e., $k = 1$, we can derive from the table in Figure 1(b) that $\omega_1 = \{e_2, e_3\}$ and $\omega_2 = \{e_2, e_3, e_4\}$, and $\upsilon_1 = \{e_3\}$ and $\upsilon_2 = \{e_4\}$.

Now we formulate conditions that check whether the system is in a state that provides the sets of resource edges necessary for an expression level change. Let

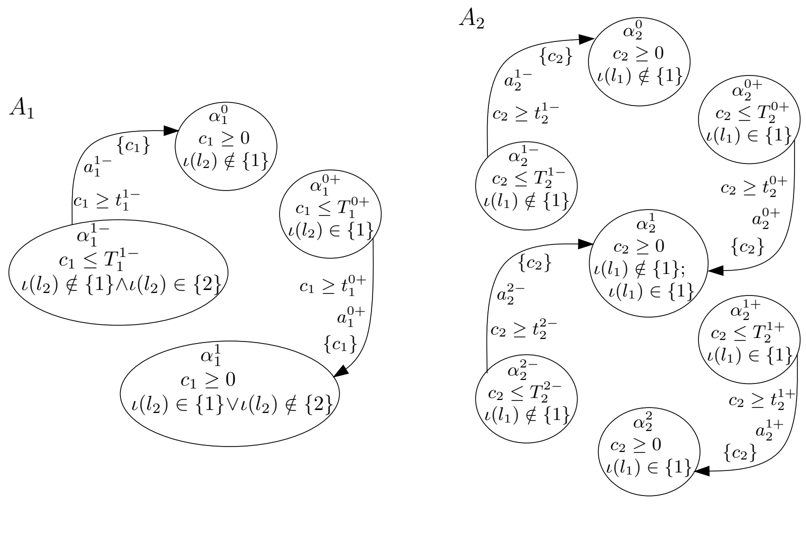


Fig. 2. Components A_1 and A_2 representing α_1 and α_2 in Figure 1.

$l \in L_1 \times \cdots \times L_n$, where l_i is the chosen location in A_i. Denote for each edge e by $t(e)$ the index of the tail of e, i. e., the tail of e is $\alpha_{t(e)}$. Then we define

$$\lambda_i^{\omega_h}(l) := \bigwedge_{e \in \omega_h} \lambda_i^e(l_{t(e)}) \qquad \text{and} \qquad \lambda_i^{\upsilon_h}(l) := \bigwedge_{e \in \mathcal{H}(\alpha_i) \setminus \upsilon_h} \overline{\lambda}_i^e(l_{t(e)}).$$

If $\lambda_i^{\omega_h}(l)$ is true for some ω_h, then an increase of expression level of gene α_i is indicated. If $\lambda_i^{\upsilon_h}(l)$ is satisfied for some υ_h, then the component will start the process of expression level decrease. In our example in location α_2^1 we obtain for ω_1 and υ_1 as determined above the conditions $\lambda_2^{\omega_1}(l) = (\iota(l_1) \notin \{1\}) \wedge (\iota(l_2) \in \{1, 2\})$ and $\lambda_2^{\upsilon_1}(l) = (\iota(l_1) \in \{1\}) \wedge (\iota(l_2) \in \{1\})$.

In order to induce a corresponding change in expression level, it is sufficient if the condition $\lambda_i^{\omega_h}(l)$ resp. $\lambda_i^{\upsilon_h}(l)$ holds for some ω_h resp. υ_h. Due to this observation we set

$$\Lambda_i^{k+}(l) := \bigvee_{h \in \{1,\ldots,m_i^1\}} \lambda_i^{\omega_h} \qquad \text{and} \qquad \Lambda_i^{k-}(l) := \bigvee_{h \in \{1,\ldots,m_i^2\}} \lambda_i^{\upsilon_h}\,.$$

We define Λ_i^{0+} and $\Lambda_i^{p_i-}$ accordingly.

Now, we assign all locations α_i^k, $k \in \{1, \ldots, p_i - 1\}$ the conditions Λ_i^{k+} and Λ_i^{k-}. The location α_i^0 resp. $\alpha_i^{p_i}$ is labelled with Λ_i^{0+} resp. $\Lambda_i^{p_i-}$ only, since the location represents the lowest resp. highest expression level possible. Furthermore, we want to check in an intermediate location whether the condition that led to the process of changing the expression level is still valid. If that is not the case, the system should not remain in that location. Thus, we associate with location α_i^{k+} the condition $\neg\Lambda_i^{k+}$ for all $k \in \{0, \ldots, p_i - 1\}$, and allot to location α_i^{k-} the condition $\neg\Lambda_i^{k-}$ for all $k \in \{1, \ldots, p_i\}$.

All the above considerations on how the switch conditions should influence the behavior of the system will be realized in the definition of the timed automaton representing the network dynamics.

Although the switch conditions look quite complicated, they often can be considerably simplified. Since condition (1) holds we can make the following observation. Whenever $\omega_{h_1} \subseteq \omega_{h_2}$ for sets ω_h, then $\lambda_i^{\omega_{h_1}}(l)$ is true if $\lambda_i^{\omega_{h_2}}(l)$ is true. Since condition (1) implies that $K_{i,\omega_{h_2}} \geq K_{i,\omega_{h_1}} > k$, we can delete condition $\lambda_i^{\omega_{h_2}}(l)$ from the expression $\Lambda_i^{k+}(l)$. Analogously, if $\upsilon_{h_1} \subseteq \upsilon_{h_2}$, we can delete the condition $\lambda_i^{\upsilon_{h_1}}(l)$ from the expression $\Lambda_i^{k-}(l)$. Furthermore, any inequality $\lambda_i^e(l_i)$ concerning the expression level $\iota(l_i)$ of the component A_i the inequality is associated with can be evaluated immediately. So, in the example we considered above, we have $\omega_1 \subseteq \omega_2$ and we can eliminate the condition $\lambda_2^{\omega_2}(l)$ from $\Lambda_2^{1+}(l) = \lambda_2^{\omega_1} \vee \lambda_2^{\omega_2}$, i. e., $\Lambda_2^{1+}(l) = \iota(l_1) \notin \{1\}$. For $\Lambda_2^{1-}(l)$ we have to consider both $\lambda_2^{\upsilon_1}(l)$ and $\lambda_2^{\upsilon_2}(l)$. However, we know that $\iota(l_2) = \iota(\alpha_2^1) = 1$, and thus we can simplify $\lambda_2^{\upsilon_1}(l) = (\iota(l_1) \in \{1\}) \wedge (\iota(l_2) \in \{1\}) = \iota(l_1) \in \{1\}$. The same reasoning shows that $\lambda_2^{\upsilon_2}(l) = (\iota(l_1) \in \{1\}) \wedge (\iota(l_2) \notin \{1,2\})$ is always false in location α_2^1. Thus we can eliminate the second condition from Λ_2^{1-} and obtain $\Lambda_2^{1-} = \iota(l_1) \in \{1\}$. In Figure 2 the switch conditions for each location are listed below the invariant of the location.

4.2 Capturing the Network Dynamics

To obtain an automaton A from which we can derive the dynamics of the network, we have to combine the automata representing the individual components of the network. Again, we omit the details, which can be found in [10] and, with slightly different notation, in [9].

Locations, invariants and clocks. The set of locations of A is the product space of the location sets of the components A_i. Each location carries the conjunction of invariants of its components. A part of the timed automaton derived from the components A_1 and A_2 of our running example can be seen in Figure 3(a). Again, locations are shown as ellipses labeled with the location name and the corresponding invariant. The automaton A is equipped with the set of all component clocks.

Switches and events. Switches from the component automata persist, representing location changes that only affect the corresponding component of the location vector of A. Those edges are labeled with the events in Σ_i and depend only on the respective time delays. In our example automaton in Figure 3 two such edges leave the location $(\alpha_1^{0+}, \alpha_2^{0+})$. The one labeled with a_1^{0+} is inherited from A_1, the other one from A_2.

Furthermore, given a location l of A, we can evaluate all the switch conditions belonging to the component locations l_i. We define switches in A starting in l leading to a location l' differing from l in all those components l_i with true switch conditions. More precisely, if $l_i = \alpha_i^k$ is a regular location with true switch condition $\Lambda_i^{k\varepsilon}$ with expression level k and $\varepsilon \in \{+,-\}$, then $l'_i = \alpha_i^{k\varepsilon}$. This corresponds to the interpretation of a true switch condition as a set of

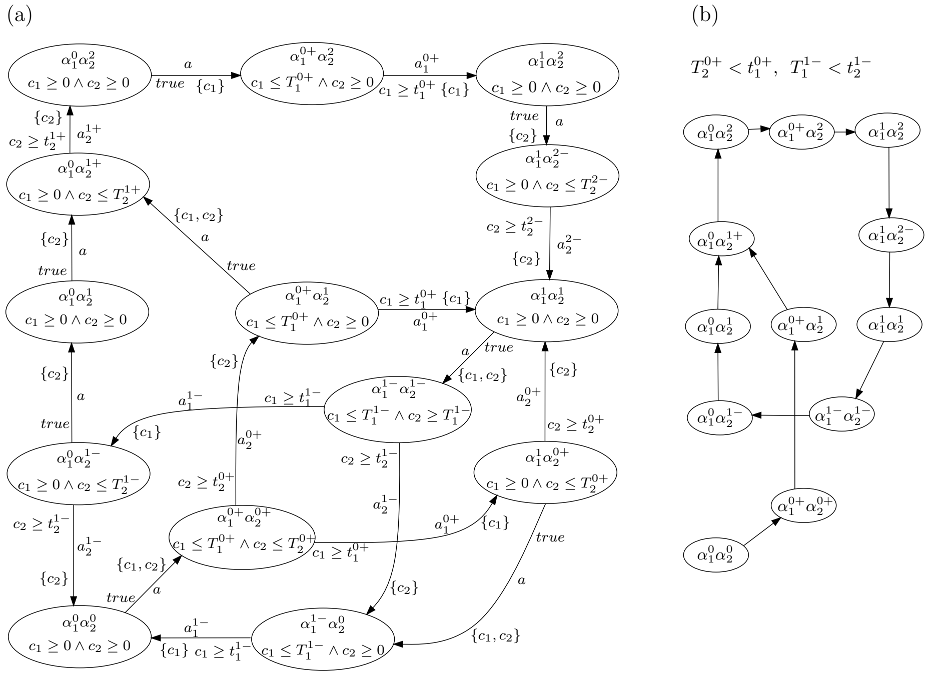


Fig. 3. In (a), part of the product automaton A derived from the components A_1 and A_2 given in Fig. 2. In (b), a path in the symbolic transition system respecting the time constraints given in the figure.

resource edges effecting a process of expression level change. If $l_i = \alpha_i^{k\varepsilon}$ is an intermediate location with true switch condition, then $l_i' = \alpha_i^k$, since the true switch condition signifies that the current state of the system does not support the process of expression level change of α_i any longer. The switches resulting from the evaluation of the switch conditions are labeled with the event a and have no guard, or rather they are labeled with the guard *true*. When executing such a switch all the clocks of the components undergoing a location change are reset. In our example we can see such a switch starting in (α_1^0, α_2^0) leading to $(\alpha_1^{0+}, \alpha_2^{0+})$, since the switch conditions of both locations of the respective component automata given in Figure 2 are true.

Transition system. The graph representing the automaton A does not represent the possible dynamical behavior of the system, since we have not yet evaluated the time constraints on switches and locations. Thus we have to derive the corresponding transition system. Basically, we follow the paths along the discrete location and switches in the graph representing A, if there exist clock values consistent with the time constraints we encounter along the way. That is, time may pass in locations as long as the maximal time delays in the invariants are not exceeded. Switches can be activated when the clock values are larger than the minimal time delays in the guards. Of course, every switch with the guard *true* can be executed regardless of the clock values. Executing switches is instantaneous and the reset commands have to be obeyed. We refine the resulting transition system in one aspect. Whenever the system is in a state allowing for the execution of a switch resulting from the evaluation of the switch conditions, i. e., a switch labeled with a, time is not allowed to elapse further in the corresponding location of A. Thus we ensure that the network interactions primarily determine the behavior of the system. For details see [9].

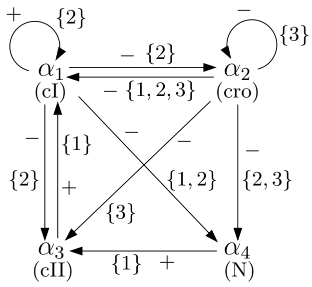
However, additional information about the time delays may lead to considerable refinement of the analysis of the system's dynamics. The state transition graph of our running example is strongly connected (see Figure 1(b)), prohibiting precise predictions of the system's behavior. This is also illustrated by Figure 3(a), which shows all the locations of the automaton A reachable from the location (α_1^0, α_2^0). However, given the additional information that the expression level increase from 0 to 1 is always faster for α_2, signifying for instance a higher production rate of a gene, and that the expression level decrease from 1 to 0 is always faster for α_1, representing for instance a high decay rate of some substance, we can obtain a much stronger understanding of the dynamics. Under those assumptions the system will reach, starting from (α_1^0, α_2^0), a cycle, that is a sustained oscillation, as shown in Figure 3(b).

5 Context Sensitivity of Time Delays

The current state of the system may not only influence the character of the network interactions but also the time delays associated with an expression level change. In this section we motivate why and illustrate how to incorporate context sensitivity of time delays into our modeling approach by considering the genetic switch of bacteriophage λ.

Phage λ is a virus that can act in two different ways upon infection of a bacterium. If they display the lytic response, the virus multiplies and lyses the cell. In other cases the viral DNA integrates into the bacterial chromosome, rendering the viral genome harmless for the so-called lysogenic bacterium. In [12] the authors propose a logical model of the genetic network underlying the behavior described above. It comprises the genes *cI*, *cro*, *cII* and N, the choice of parameter values and thresholds is based on experimental data. The resulting model is given in Figure 4. Here the inhibiting influence of *cI* on itself mentioned in Section 2 is not incorporated since it only takes place under conditions not of interest for the analysis of the behavioral switch (see again [12]). The lytic and lysogenic behavior can be identified in the state transition graph. The former is represented by the steady state $(2, 0, 0, 0)$ and the latter by a cycle comprising the states $(0, 2, 0, 0)$ and $(0, 3, 0, 0)$.

We have modeled this system as a timed automaton and implemented it in UPPAAL (see `http://www.uppaal.com`), a software that allows for verification and analysis via a model checking engine. In our model we exploited available temporal data to refine the results concerning the dynamical behavior (see [10]



$K_{1,\{e_{21}\}}$	$= 2$	$K_{2,\{e_{12}\}}$	$= 2$
$K_{1,\{e_{31}\}}$	$= 2$	$K_{2,\{e_{12},e_{22}\}}$	$= 3$
$K_{1,\{e_{11},e_{21}\}}$	$= 2$		
$K_{1,\{e_{11},e_{31}\}}$	$= 2$	$K_{3,\{e_{13},e_{23},e_{43}\}}$	$= 1$
$K_{1,\{e_{21},e_{31}\}}$	$= 2$		
$K_{1,\{e_{11},e_{21},e_{31}\}}$	$= 2$	$K_{4,\{e_{14},e_{24}\}}$	$= 1$

Fig. 4. Model of the phage λ network. Edges are denoted by e_{ij} with α_i the tail and α_j the head of the edge. Only non-zero parameter values are given.

for details). However, certain aspects of the system are not correctly captured in the model as we will explain in the following. While genes *cI* and *cro* define the lysogenic and lytic states respectively, studies have shown the importance of *cII* in the switching process. The time delay values associated with accumulation and decay of the product of *cII* can be linked to environmental conditions such as richness of the medium (see [6]). Furthermore, it has been shown that the influence of *cII* leads to rapid synthesis of the *cI* product. When considering the parameter values given in Figure 4 we see that both the presence of *cII* and the absence of *cro* product are sufficient for *cI* to obtain its highest expression level. We lack the means to express the different time delays associated with the *cI* expression level change with respect to *cII* activity. However, since our modeling approach is highly suited for context sensitive systems, we can easily extend the model to capture the addressed properties.

Enhancing the framework

The process of changing the expression level is represented by the intermediate location of the component automaton. If we want to associate different time delays with such a process we simply introduce two (or more) intermediate locations for the same process. We indicate the difference in the location name, e. g. with $\varepsilon \in \{+, -\}$ we denote by $\alpha_i^{k\varepsilon,s}$ (resp. $\alpha_i^{k\varepsilon,f}$) the slow (resp. fast) process of expression level change. For each location we choose a maximal time delay $T_i^{k\varepsilon,s}$ (resp. $T_i^{k\varepsilon,f}$) for the invariant. Each location is connected to the location α_i^l, with $l = k \pm 1$ depending on ε, via an edge labeled with the guard representing the corresponding minimal time delay $t_i^{k\varepsilon,s}$ (resp. $t_i^{k\varepsilon,f}$), an event $a_i^{k\varepsilon,s}$ (resp. $a_i^{k\varepsilon,f}$), and a reset for the component clock. In Figure 5 we see a part of the timed automaton representing *cI* with two intermediate locations representing the expression level change from 0 to 1.

In the context of the automaton representing the network, we decide via evaluation of the switch conditions if a system component executes a location change ending in an intermediate location. Now we have to classify the switch

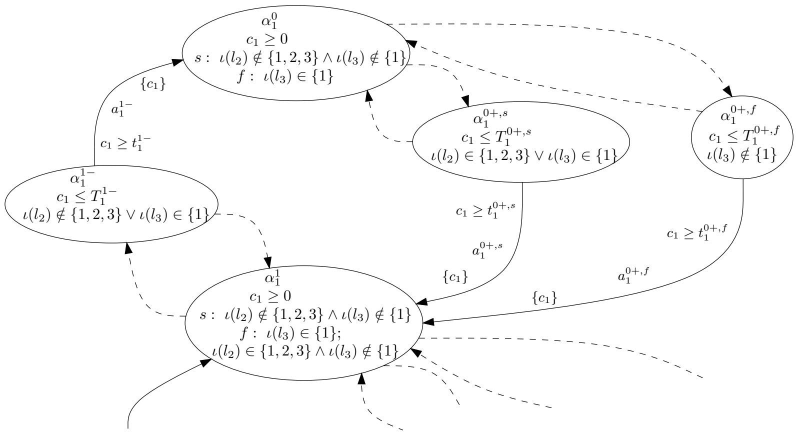


Fig. 5. Part of the timed automaton representing *cI*. Dashed arrows signify location changes due to evaluation of switch conditions.

conditions such that we can distinguish between switches leading to different locations that represent the same process of expression level change. To do so we consider again the locations $\alpha_i^{k\varepsilon,s}$ and $\alpha_i^{k\varepsilon,f}$. If $\varepsilon = +$, we need to classify the switch conditions in location α_i^k. As a first step to formulating the switch conditions we determine the sets of resource edges $\omega_1, \ldots, \omega_{m_i^1} \subset \mathcal{H}(\alpha_i)$ that lead to the process of increasing the expression level k (see Section 4). Now we group the sets ω_h that lead to a fast change of expression level in a set Ω^f and the others in a set Ω^s. Then we derive a switch condition for the sets in Ω^f just as described in Section 4. If the system is in a state that the condition is true the component changes to the location $\alpha_i^{k+,f}$. We derive the switch condition for Ω^s also as described in Section 4, but additionally we demand that the switch condition of Ω^f is false. If this condition is met, then the component executes a location change to $\alpha_i^{k+,s}$. The switch conditions for both intermediate locations are, as usual, the negation of the switch conditions leading to the corresponding switch. If $\varepsilon = -$, we proceed analogously using the sets $\upsilon_1, \ldots, \upsilon_{m_i^2}$.

In order to model the more rapid expression level increase of *cI* in the presence of *cII* product, we first consider the sets of resource edges leading to the increase from 0 to 1. They are given in the left column of parameter values in Figure 4. A fast change is effected whenever e_{31} is a resource edge. Thus, we have $\{e_{31}\}$, $\{e_{11}, e_{31}\}$, $\{e_{21}, e_{31}\}$ and $\{e_{11}, e_{21}, e_{31}\}$ in the set Ω^f. After the simplification of the switch condition as introduced in Section 4, we obtain $\iota(l_3) \in \{1\}$ as switch condition leading to fast expression level change. The switch condition for Ω^s is $\iota(l_2) \notin \{1, 2, 3\}$, and thus satisfying the condition $(\iota(l_2) \notin \{1, 2, 3, \}) \wedge (\iota(l_3) \notin \{1\})$ leads to the intermediate location $\alpha_1^{0+,s}$. The same considerations can be applied for the expression level change from 1 to

2. Part of the automaton is given in Figure 5. Here dashed arrows signify the location changes governed by the switch conditions in the network context.

Again we have implemented the model in UPPAAL and analyzed the behavior of the system. In a model with basically the same time delays for all location changes, we find that both the steady state representing the lysogenic behavior and the cycle representing lysis are reachable in the non-deterministic transition system. However, a faster expression level increase from 0 to 1 and a slower decrease from 1 to 0 of *cII* (both can be effected by a slower degradation rate) renders the cycle representing lysogeny unreachable. Thus, the system will always display lytic behavior. In contrast, smaller time delays for the expression level decrease and slower expression level increase of *cII* ensure that the system displays lysogenic behavior.

Regardless of the satisfactory result in the example considered above, the modeling of the context sensitive time delays may still be improved. Given the situation that the process of expression level change is nearly completed in the slow intermediate location, a change in the system's state that satisfies the conditions for a fast expression level change would lead to a repetition of the process of expression level change. All the progress made in the slow location would be lost. Although we could introduce switches leading directly from the slow to the fast intermediate location, we still have to decide which value to assign the clock upon execution of such a switch. The framework of timed automata only allows for two possibilities. Either the clock keeps its current value or it is reset to some constant. Obviously both choices do not reflect the desired behavior.

6 Perspectives

In this paper, we introduced a rigorous framework for the logical modeling of context sensitive systems. It extends our work on a hybrid formalism based on the classical Thomas approach and the theory of timed automata (see [9], [10]) in two directions. We are now not only able to model systems displaying context sensitivity regarding network interactions as described in Section 2, but can also deal with the context sensitivity of time delays, cf. Section 5. In many cases this allows for a more realistic representation of biological systems and a refined analysis of the resulting dynamics.

Generally, many interesting questions regarding modeling and dynamical analysis in this framework remain to be considered. For example, concepts like stability should now be phrased in a hybrid way, taking into account the time constraints associated with a certain behavior. This would allow for a more precise evaluation of asymptotical behavior, thus leading to more reliable predictions in case of system simulation as well as a better basis for model comparison.

In the current framework progress achieved in an intermediate location towards an expression level change, e. g. an increase in some substance concentration nearly up to the threshold, is completely negated if a location change signifying the abortion of the expression level change occurs. A more realistic representation should allow for a time delay, depending on how much time has passed in the intermediate location, associated with the loss of the progress made. This difficulty was already addressed at the end of the preceding section. It has to be considered whether the use of a more general class of hybrid automata would resolve this problem. However, powerful results concerning analysis and verification of models by means of model checking techniques exist in the theory of timed automata. Since effective methods to analyze large transition systems are needed in the context of biological systems, we should ensure that we do not loose advantages in that area. Rather, suitability and possibilities of applying model checking techniques for analyzing the behavior of biological networks need to be studied further.

References

1. R. Alur. Timed Automata. In *Proceedings of the 11th International Conference on Computer Aided Verification*, volume 1633 of *LNCS*, pages 8–22. Springer, 1999.
2. 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.*, 229:339–347, 2004.
3. C. Chaouiya, É. Remy, B. Mossé, and D. Thieffry. Qualitative analysis of regulatory graphs: a computational tool based on a discrete formal framework. In *First Multidisciplinary International Symposium on Positive Systems: Theory and Applications, POSTA 2003*, volume 294 of *LNCIS*, pages 119–126. Springer, 2003.
4. L. Glass and S. A. Kauffman. The logical analysis of continuous, non-linear biochemical control networks. *J. Theor. Biol.*, 39:103–129, 1973.
5. S. A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, 22:437–467, 1969.
6. A. Oppenheim, O. Kobiler, J. Stavans, D. Court, and S. Adhya. Switches in bacteriophage lambda development. *Annu. Rev. Genet.*, 39:409–429, 2005.
7. É. Remy, P. Ruet, and D. Thieffry. Graphic requirements for multistability and attractive cycles in a boolean dynamical framework. Prépublication, 2005.
8. A. Richard and J.-P. Comet. Necessary conditions for multistationarity in discrete dynamical systems. Rapport de Recherche 123, 2005.
9. H. Siebert and A. Bockmayr. Incorporating time delays into the logical analysis of gene regulatory networks. In *Computational Methods in Systems Biology, CMSB 2006, Trento, Italy*, volume 4210 of *LNCS*, pages 169–183. Springer, 2006.
10. H. Siebert and A. Bockmayr. Temporal constraints in the logical analysis of regulatory networks. Matheon Preprint 385, 2007.
11. M. Sugita. Functional analysis of chemical systems in vivo using a logical circuit equivalent. *J. Theor. Biol.*, 1:415–430, 1961.
12. D. Thieffry and R. Thomas. Dynamical behaviour of biological regulatory networks - II. Immunity control in bacteriophage lambda. *Bull. Math. Biol.*, 57:277–297, 1995.
13. R. Thomas. Boolean formalisation of genetic control circuits. *J. Theor. Biol.*, 42:565–583, 1973.
14. R. Thomas and R. d'Ari. *Biological Feedback*. CRC Press, 1990.
15. 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*, 11:180–195, 2001.