Towards Action-Based Analysis of Discrete Regulatory Networks with Short-Term Stimuli

Christian Krause*
Software Engineering
CWI Amsterdam
c.krause@cwi.nl

Carola Krause
Molecular Cell Biology
Leiden University Medical Center
c.krause@lumc.nl

Erik P. de Vink
Formal Methods
TU Eindhoven
evink@win.tue.nl

ABSTRACT

To incorporate instant effects and different timescales within a single biological system, an extension of discrete regulatory networks with short-term stimuli is proposed. By maintaining a vector of recent changes, activities initiated by a steep increase or decrease can be captured in a qualitative setting. In order to compensate for the blow-up due to enhanced states, we focus on observable behavior. Identification of bisimilar states yields a compact system representation truthfully expressing the information relevant for deciding logical properties. The approach is implemented by means of a front-end to the mCRL2 tool set and illustrated for the switching of bacteriophage lambda and a bio-medical case study related to TGFβ driven fibrotic conditions.

Keywords

Discrete regulatory networks, bisimulation, model checking.

1. INTRODUCTION

Boolean and multi-valued regulatory networks are established qualitative modeling tools in settings with imprecise data or incomplete knowledge [16, 34]. Practical methods can be exploited to construct networks, e.g. network synthesis from gene expression data, and to analyze their dynamics, e.g. identifying sets of attractors. For simplicity, in the standard asynchronous semantics of discrete regulatory networks all agents act at the same timescale. A priori there is no scheduling of transitions between logical states, although frequently the biological evidence indicates otherwise. However, refining the computational interpretation should be taken with care. Also in qualitative approaches analysis techniques are hampered by state space explosion. In view of this, we propose to distinguish in discrete regulatory networks between so-called fast and slow transitions, while applying bisimulation minimization to keep the associated state spaces reduced.

Following [35], a discrete regulatory network is given by an interaction graph and update functions, the nodes representing the agents involved, the edges indicating the mutual activation and inhibition between them, the update functions determining the possible fluctuations. Logical states hold the abstract concentration levels. For a transition between states, an agent is selected, its stimuli, i.e. the incoming arcs, are assessed and the next concentration value is computed according to the activation function of the agent. We augment the state information for each agent with differentials, that indicate the increase or decrease since the previous reference point. For slow transitions of an agent only the concentration levels of its stimuli are taken into account, as in standard discrete regulatory networks. For fast transitions also their differential values are considered. The expressivity gained by fast transitions is two-fold: In a maximal progress interpretation where fast transitions have precedence over slow transitions, two qualitative timescales can be distinguished, separating steep increases or decreases from moderate ones. This acts at the level of transitions. Also, priority can be given to specific sets of stimuli over others, distinguishing differences in activation in the presence or absence of catalysts without affecting the target concentration level. The concepts and their application are illustrated for the well-known switch of the lambda phage building on work reported in [29].

As another contribution of our paper we advocate the use of bisimulation for discrete regulatory networks, both for standard and augmented ones [23]. We label the transitions between logical states of a regulatory network with the name and target level of the agent involved. As a consequence, in many cases, one can discard the states themselves when analyzing the system as the relevant information can be retrieved from the actions, the labels that decorate the transitions. Thus discrete regulatory networks yield labeled transition systems (LTS) as their state spaces. In such an observation centric approach quotienting of the state space modulo bisimulation is a natural option [2]. For, properties expressible in modal μ-calculus or fragments thereof equivalently hold in the original LTS as in its bisimilar minimization. The computational penalty is limited, minimization is linear. Starting from a minimal representation of the regulatory network, one can either proceed by model checking or apply dedicated algorithms, for example to find steady states or attracting cycles, at reduced computation times.

The above approach has been implemented as a front-end

*Corresponding author.

to the mCRL2 toolset [13]. From a complete description of a discrete regulatory network, the interaction graph and update functions, with or without fast transitions, a system of parallel processes in terms of the mCRL2 specification language is generated. One may subsequently analyze the system by means of symbolic model checking within the toolset or export the minimal LTS for processing by other tools. Alternatively, as proposed in [4], one may consider an interaction graph by itself without update functions given. Then one may generate all possible update functions that satisfy a specific set of properties. For this a generate-and-test cycle is set up, following the same procedure as for complete networks. In general, one has to deal with an exponential blow-up, but e.g. under monotonicity conditions [31] it can be feasibly done. In an *in silico* case study related to myofibrosis we show, by chasing all possible activation functions, the value of our approach directing further experimental investigations.

The paper is organized as follows. We recall the notion of bisimulation and its logical invariance in Section 2 to stress our focus on observability. Discrete regulatory networks are covered in Section 3, taking benefit from the formal modeling by Bernot et al. The extension with short-term stimuli, introducing slow and fast transitions, is presented in Section 4. Bisimulation as a means for state space minimization is explained in Section 5. A significant reduction in number of states, and as a consequence in number of transitions, can be obtained. In Section 6 we show in a qualitative setting how to fine-tune in an augmented regulatory network the gene interaction to line up with experimental results. A bio-medical case is discussed in Section 7, dealing with short-termed influence of one agent on another. Support by our *Arne* tool is discussed in Section 8. Finally, Section 9 covers related work, Section 10 wraps up and mentions future work.

2. BISIMULATION

Bisimulation, strong bisimulation to be precise, is a well-known notion of behavioral equivalence in concurrency theory [23]. In essence, two states of the same or in two different labeled transition systems are bisimilar if they cannot be distinguished by an external observer.

Definition 1. A labeled transition system (LTS) is a triple $\mathcal{L} = \langle S, A, \rightarrow \rangle$ with set of states S, set of actions or labels A and a set of transitions $\rightarrow \subseteq S \times A \times S$. Notation: $s \xrightarrow{a} s'$ for $(s, a, s') \in \rightarrow$. □

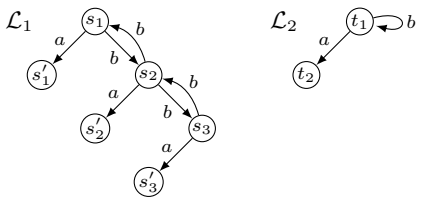


Figure 1: bisimilarity

Figure 1 shows two labeled transition systems, $\mathcal{L}_1$ and $\mathcal{L}_2$. They have actions a, b, states s_i, s_i', $i = 1, 2, 3$, and t_1, t_2, respectively, and transitions $s_i \xrightarrow{a} s_i'$, $s_1 \xrightarrow{b} s_2$, $s_2 \xrightarrow{b} s_1$, $s_2 \xrightarrow{b} s_3$, $s_3 \xrightarrow{b} s_2$ for $\mathcal{L}_1$ and $t_1 \xrightarrow{a} t_2$, $t_1 \xrightarrow{b} t_1$ for $\mathcal{L}_2$.

Definition 2. A bisimulation for two labeled transition systems $\mathcal{L}_1$ and $\mathcal{L}_2$, say $\mathcal{L}_i = \langle S_i, A, \rightarrow_i \rangle$, is a binary relation $R \subseteq S_1 \times S_2$ such that for all $s \in S_1$, $t \in S_2$ such that $R(s, t)$ it holds that

- if $s \xrightarrow{a}_1 s'$ then $t \xrightarrow{a}_2 t'$ for some $t' \in S_2$ with $R(s', t')$;
- symmetrically, if $t \xrightarrow{a}_2 t'$ then $s \xrightarrow{a}_1 s'$ for some $s' \in S_1$ satisfying $R(s', t')$.

The states s in $\mathcal{L}_1$ and t in $\mathcal{L}_2$ are called bisimilar if $R(s, t)$ for a bisimulation R for $\mathcal{L}_1$ and $\mathcal{L}_2$.

Two labeled transition systems $\mathcal{L}_1$ and $\mathcal{L}_2$ are bisimulation equivalent if there exists a bisimulation relation R for $\mathcal{L}_1$ and $\mathcal{L}_2$ covering all states, i.e. $S_1 = \{ s \mid \exists t \colon R(s, t) \}$ and $S_2 = \{ t \mid \exists s \colon R(s, t) \}$. □

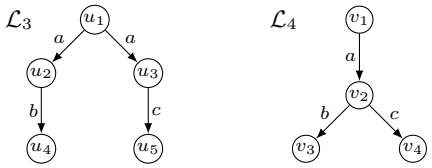


Figure 2: non-bisimilarity

Consider the labeled transition systems $\mathcal{L}_1$ and $\mathcal{L}_2$ depicted in Figure 1. The states s_1 and t_1 are bisimilar. Also s_2 and s_3, on the one hand, and t_1, on the other hand, are bisimilar as the relation $R = \{ (s_i, t_1), (s_i', t_2) \mid i = 1..3 \}$ is a bisimulation for $\mathcal{L}_1$ and $\mathcal{L}_2$. In fact, $\mathcal{L}_2$ is the *minimal* labeled transition system that is bisimilar to $\mathcal{L}_1$. As a non-example, the states u_1 and v_1 of $\mathcal{L}_3$ and $\mathcal{L}_4$ in Figure 2 are not bisimilar, as both u_2 and u_3 in $\mathcal{L}_3$ miss a transition to be bisimilar to v_2 in $\mathcal{L}_4$. However, from u_1 and v_1 the same traces are observed, viz. ab and ac.

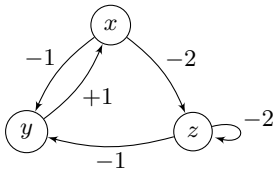
Given a labeled transition system $\mathcal{L}$, the minimal labeled transition system that is bisimilar to $\mathcal{L}$ can be efficiently constructed. The well-known Paige-Tarjan algorithm for example, has complexity $\mathcal{O}(|\rightarrow| \cdot \log |S|)$, linear in the number of transitions, logarithmic in the number of states. The following fundamental result connects bisimilarity and the modal μ-calculus. The latter is an expressive modal logic, featuring greatest and least fixed point operators and strictly subsuming CTL*. For more detail see, e.g., [7].

Theorem 1. If for two labeled transitions systems $\mathcal{L}_1$ and $\mathcal{L}_2$, two states $s_1 \in \mathcal{L}_1$ and $s_2 \in \mathcal{L}_2$ are bisimilar, then they satisfy the same formulae of the modal μ-calculus. □

From the theorem we derive that if $\mathcal{L}$ is bisimilarly minimized to $\mathcal{L}_{min}$, a property that is expressible in the modal μ-calculus and holds for $\mathcal{L}_{min}$, also holds for $\mathcal{L}$, and vice versa. Hence, to check if $\mathcal{L}$ satisfies a certain property, it may be computationally advantageous to check the property on $\mathcal{L}_{min}$ instead.

3. DISCRETE REGULATORY NETWORKS

Discrete regulatory networks or generalized logical networks as advocated by Thomas c.s. [35], constitute a discrete multi-valued abstraction of regulatory networks. For a set of agents, typically genes $\mathcal{X}$, the interaction among its species is represented by a directed graph $\mathcal{G}$, say. The



R_x	$\emptyset$	y	
K_x	0	1	

R_y	$\emptyset$	x	z	xz
K_y	1	1	1	0

R_z	$\emptyset$	x	z	xz
K_z	0	1	0	2

Figure 3: example regulatory network

nodes are the elements of $\mathcal{X}$, the edges reflect an activating or inhibiting effect of one gene, the source node of the edge, on another gene, the target node of the edge. Self-loops, representing auto-regulation, are permitted. Edges are labeled with signed integers. A positive number $+n$ on an edge $x \xrightarrow{+n} y$ indicates an activating influence of gene x on gene y provided the current value of x is at least n. Likewise, a negative number $-n$ on an edge $x \xrightarrow{-n} y$ indicates an inhibitory influence of gene x on gene y, again provided the current value of x is at least n. In addition to the interaction structure we need (i) to record the logical state of the network that holds the current gene levels, and (ii) to decide, for a given gene, what is the net effect of simultaneous influences, as well the induced change of state.

We represent a discrete regulatory network as a four-tuple

$$\mathcal{N} = \langle\, \mathcal{X}, (B_x)_{x\in\mathcal{X}}, (A_{x,y})_{x,y\in\mathcal{X}}, (K_x)_{x\in\mathcal{X}} \,\rangle$$

where, for genes $x, y \in \mathcal{X}$ with $x \xrightarrow{+n} y$ or $x \xrightarrow{-n} y$,

- B_x is a positive number called the bound of x;
- $A_{x,y} = \{\, m \mid n \le m \le B_x \,\}$ or $A_{x,y} = \{\, m \mid 0 \le m < n \,\}$, and is called the set of activating values of x for y;
- $K_x \colon 2^{\mathcal{R}_x} \to \mathcal{B}_x$, with $\mathcal{R}_x = \{\, y \mid A_{y,x} \neq \emptyset \,\}$ and $\mathcal{B}_x = \{0, 1, \ldots, B_x\}$, is the activation function of x mapping sets of stimuli of x to levels of x.

Regarding the representation as a graph $\mathcal{G}$, the bound B_x is often chosen to be the out-degree of x for $\mathcal{G}$. If an arrow $x \xrightarrow{+n} y$ connects the genes x and y, $A_{x,y} \subseteq \mathcal{B}_x$ is upward closed and includes n, if $x \xrightarrow{-n} y$ then $A_{x,y} \subseteq \mathcal{B}_x$ is downward closed and excludes n. If there is no edge $x \to y$ in $\mathcal{G}$, we have $A_{x,y} = \emptyset$. Note, by referring to $A_{x,y}$ the actual value of x is abstracted away.

For example, for the regulatory network in Figure 3, we have the gene set $\mathcal{X} = \{x, y, z\}$ with bounds $B_x = 2$, $B_y = 1$ and $B_z = 2$. The activation values are $A_{x,y} = \{0\}$ for x on y, $A_{x,z} = \{0, 1\}$ for x on z, $A_{y,x} = \{1\}$ for y on x, $A_{z,y} = \{0\}$ for z on y and $A_{z,z} = \{0, 1\}$ for z on itself. The activation function K_x for x shows that there is no expression of x in the absence of y. The activation function K_y captures that expression of y is inhibited only if both x and z have reached their threshold values. Moreover K_z is defined such that the expression of z is moderately inhibited by z reaching the threshold 2 only, but a value of 2 for x strongly inhibits z. For a compact notation the resource sets R_x, R_y and R_z are denoted as strings, e.g. xz representing the set $\{x, z\}$.

The set $\mathcal{S} = \prod_{x\in\mathcal{X}} \mathcal{B}_x$ comprises the state space associated to a network $\mathcal{N}$. A logical state $s \in \mathcal{S}$ is a vector $(s_x)_{x\in\mathcal{X}}$ of values for all the genes in $\mathcal{X}$. The network $\mathcal{N}$ induces a transition relation on S for each gene. We write

$$s \xrightarrow{y, s'_y} s' \iff s'_y = s_y + \mathrm{sgn}(\, K_y(R_{s,y}) - s_y \,) \wedge s'_x = s_x \quad \text{for } x \neq y$$

where $R_{s,y} = \{\, x \mid s_x \in A_{x,y} \,\}$ is the resource set of y in s. The activation function K_y for y determines a target level, given the resource set R in s. However, in the asynchronous set-up, the level for y in s does not jump directly to this target level in s'. Instead, the sign-function 'sgn' yields an increase $+1$ if $K_y(R) > s_y$ and a decrease -1 if $K_y(R) < s_y$. In cases where $K_y(R) = s_y$, we let sgn be undefined and allow no transition. Hence, we model only those transitions that correspond to actual level changes. Moreover, the value of y changes at most by ± 1 and the values of the other genes remain the same.

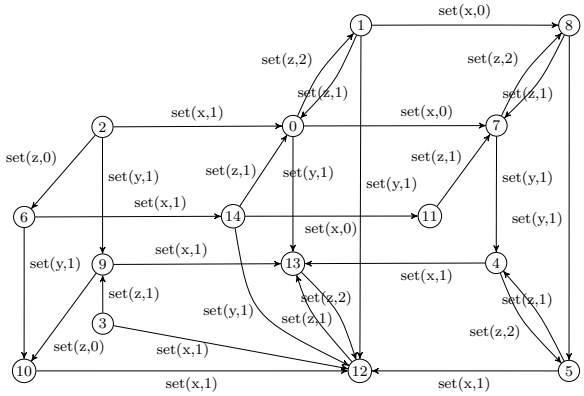


Figure 4: induced labeled transition system

The labeled transition system that is induced by the example gene regulatory network as obtained after bisimulation reduction is given in Figure 4. The figure has been generated using the tool discussed in Section 8. The transitions are labeled with an action called 'set' which has two arguments: (i) the name of the gene who's activation level changed, and (ii) the level it has changed to. Note that our approach is action-based, and not state-based as usual in modeling of discrete regulatory networks. This is witnessed by the fact that states are anonymous and that their respective activation levels can only be recovered by looking at the history, i.e., traces leading into the states. However, specific unfoldings are needed to represent different but bisimilar initial states. See Section 6. We believe that the observation-oriented approach has several advantages. For instance, we can use notions of behavioral equivalence and reduction techniques to respectively compare and minimize network behavior. We discuss this topic in more detail in Sections 5 and 8.

4. SHORT-TERM EFFECTS

In order to differentiate in timescales of transitions in discrete gene regulatory networks, we propose an extension of standard networks with so-called short-term or fast dynamics. To this end, a state consists of a pair $\langle s, \delta \rangle$, holding the levels of gene expression in the vector s as before, and additionally a vector δ as an indication of a recent increase, of a recent decrease, or of no recent change.

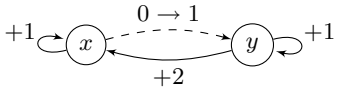
Definition 3. A discrete regulatory network with short-term effects $\mathcal{N}$ is a five tuple

$$\mathcal{N} = \langle\, \mathcal{X}, (B_x)_{x\in\mathcal{X}}, (A_{x,y})_{x,y\in\mathcal{X}}, (A'_{x,y})_{x,y\in\mathcal{X}}, (K_x)_{x\in\mathcal{X}} \,\rangle$$

with, for genes $x, y \in \mathcal{X}$,

- a bound $B_x > 0$, $\mathcal{B}_x = \{\, 0, 1, \ldots B_x \,\}$;
- a set $A_{x,y} \subseteq \mathcal{B}_x$ of regular activations of x, as before;
- a set $A'_{x,y} \subseteq \mathcal{B}_x \times \mathcal{B}_x$ of short-term activations of x;
- an activation function $K_y \colon 2^{\mathcal{R}_y} \to \mathcal{B}_y$, where $\mathcal{R}_y = \{\, x \mid A_{x,y} \cup A'_{x,y} \neq \emptyset \,\}$. □

Now, the activation function K_y for a gene y does not only take into account the current level of genes x with $A_{x,y} \neq \emptyset$, but also the *change* in level for genes x with $A'_{x,y} \neq \emptyset$. One may interpret a set $A'_{x,y} \subseteq \mathcal{B}_x \times \mathcal{B}_x$ of short-term activations as denoting a number of jumps in the level of the gene x. A pair $(n, m) \in A'_{x,y}$ indicates that the gene x is stimulating the gene y, activating or inhibiting, if x has recently jumped from level n to level m. So, if x has changed from n into m, the influence of x on y should be taken into account when evaluating the target value for y. Below, we make precise what is considered as recent.



R_x	$\emptyset$	x	y	xy
K_x	1	0	1	2

R_y	$\emptyset$	x	y	xy
K_y	0	0	2	0

Figure 5: example network with short-term effects

Consider, as an example of a discrete gene regulatory network with short-term effects, the network depicted in Figure 5. We have genes $\mathcal{X} = \{x, y\}$ with respective bounds $B_x = 2$, $B_y = 2$. Here we have the activation sets $A_{x,x} = \{1, 2\}$, $A_{y,x} = \{2\}$ and $A_{y,y} = \{1, 2\}$ and the short-term activation set $A'_{x,y} = \{\langle 0, 1\rangle\}$. The latter is indicated visually by the edge label 0→1 and a dashed line. Thus, if x increases from level 0 to level 1, the gene x will activate, apparently negatively, for a short while the gene y. The activation sets not mentioned are empty.

In the following, we define the dynamics of a regulatory network with short-term effects, again in terms of a labeled transition system. Essentially, we enrich the state information with a history of recent gene updates and introduce a new type of transitions.

Definition 4. The state space of a discrete gene regulatory network $\mathcal{N}$ with short-term effects is given by $\mathcal{S} \times \Delta$, where $\mathcal{S} = \prod_{x\in\mathcal{X}} \mathcal{B}_x$, as before, $\Delta_x = \{\, -B_x, \ldots, B_x \,\}$ for $x \in \mathcal{X}$, and $\Delta = \prod_{x\in\mathcal{X}} \Delta_x$. The network $\mathcal{N}$ induces two transition relations on $\mathcal{S} \times \Delta$, $\to_s$ and $\to_f$, so-called slow transitions and fast transitions, respectively, as follows.

$$(s, \delta) \xrightarrow{y, s'_y}_s (s', \delta') \iff$$
$$s'_y = s_y + \sigma \wedge \delta'_y = \sigma \wedge s'_x = s_x \wedge \delta'_x = 0$$
$$\text{with } \sigma = \operatorname{sgn}(\, K_y(R') - s_y \,) \text{ and } x \neq y$$

$$(s, \delta) \xrightarrow{y, s'_y}_f (s', \delta') \iff$$
$$s'_y = s_y + \sigma \wedge \delta'_y = \delta_y + \sigma \wedge s'_x = s_x \wedge \delta'_x = \delta_x$$
$$\text{with } \sigma = \operatorname{sgn}(\, K_y(R_s \cup R_f) - s_y \,),\ R_f \neq \emptyset \text{ and } x \neq y$$

where $R_s = \{\, x \mid s_x \in A_{x,y} \,\}$, $R_f = \{\, x \mid \langle s_x - \delta_x, s_x\rangle \in A'_{x,y} \,\}$. □

An element $(s, \delta) \in \mathcal{S} \times \Delta$ holds the current concentrations of the genes in $\mathcal{X}$ in s and their current changes in δ. For a gene $x \in \mathcal{X}$ its level ranges from 0 to B_x. As the level of x's concentration may increase or decrease, its change ranges between the bounds $-B_x$ and B_x. A slow transition $(s, \delta) \xrightarrow{y, s'_y}_s (s', \delta')$ updates the level and the change information of the gene y with the value σ, the outcome of the activation function K_y. The activation function is evaluated for the genes that are *on*, collected in the resource set R', according to their current concentrations and regular activation sets $A_{x,y}$ as before. The levels s_x of the other genes do not change. Their change information δ_x however, is reset to 0. Hence, a slow transition starts a new 'recent' time point, in this case with $\delta_y = \sigma$ as only change, if non-zero. In contrast, for the evaluation of the activation function, a fast transition $(s, \delta) \xrightarrow{y, s'_y}_f (s', \delta')$ not only takes the regular stimuli into account, but also the shortly activated genes. The latter are collected in the set of genes R''. Together, R' and R'' form the resource set for fast transitions. The change information δ_y of y is updated accordingly, while the change information of the other genes is maintained, rather than reset as for regular stimuli.

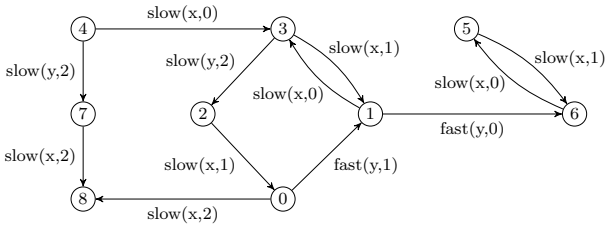


Figure 6: an LTS with fast transitions

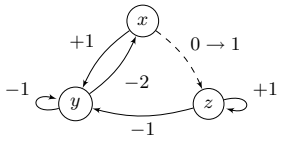
Figure 6 depicts the induced labeled transitions system with fast transitions for the network in Figure 5. An important behavioral property of this system is that x oscillates between the levels 0 and 1, using regular, i.e., slow transitions. However, there is also a larger loop $0 \to 1 \to 4 \to 7$ in which first y raises to 2 as an effect of its (slow) autoactivation, and then drops to 1 again due to an inhibition caused by x. Note further, that the system has multistationarity (state 8 and loop $3 \to 5$) and that fast transitions may have an influence on which state the system is attracted to.

The labeled transition system shown in Figure 6 is actually not directly derived using Definition 4. We did not discuss the additional minimization step, by which we reduced the state space modulo bisimulation and yields an LTS of 9 states only. The usefulness of this additional step is explained in detail in the following section.

5. STATE SPACE MINIMIZATION

An important modeling feature in our setting is the action-based view on the dynamics of a network. Instead of inspecting the concrete activation levels and their deltas of a given state, we restrict our interest to their observable change, as manifested in the actions that can be executed

from that state. This has the advantage that irrelevant details of the logical state do not have to be taken into account. Moreover, we are still in a position to apply model checking techniques, and additionally to reduce a generated labeled transition system modulo bisimulation, yielding a compact state space.



R_x	∅	y						
K_x	0	1						
R_y	∅	x	y	z	xy	xz	yz	xyz
K_y	0	2	0	1	0	2	0	0
R_z	∅	x	z	xz				
K_z	0	1	0	0				

Figure 7: example regulatory network

Consider the regulatory network with short-term effects depicted in Figure 7. Roughly, the network has the following dynamics: In absence of an influence of z on y, the genes x and y alternately reach their peak level. The gene x has a short-term activation on z when x rises to level 1. With z at level 1, z inhibits the expression of y. Once, brought down to level 0, y remains locked there.

The detailed behavior of the system is depicted in Figure 8. The picture is completely cluttered and inaccessible to human inspection. Although there are 7 different labels only, the transition system has 211 states and 246 transitions.

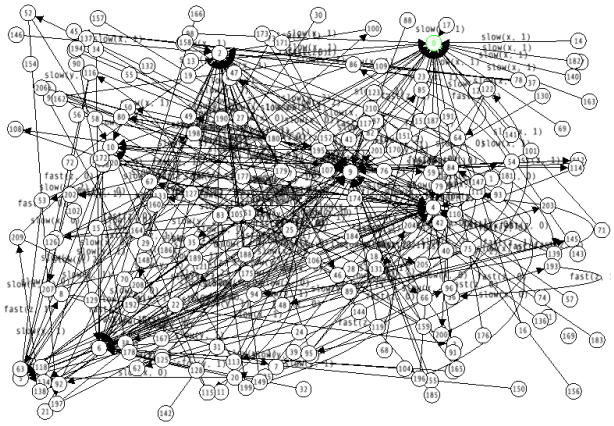


Figure 8: explicit labeled transition system

However, after application of bisimulation reduction, abstracting away from state information, the representation becomes significantly more clear, as can be seen from Figure 9. The number of states has dropped to 16, the number of transition to 24. The number of different labels remains the same, as always for bisimulation reduction. We now more easily recognize the cycling of x and y through the square $5 \to 13 \to 0 \to 6$, x flipping between level 0 and 1, y between 1 and 2 as can be read from the labels of the transitions. While z remains at level 0, the system is caught in this loop. However, in state 0 a fast transition is possible as well, based on the short-term stimulus of x reaching level 1. The cycle is maintained, viz. in the square $2 \to 14 \to 8 \to 1$, but is not stable. Because of degradation of z, lowering in level from 1 to 0, the system may return to the earlier loop. In state 1, compared to state 6, now there is also a transition reflecting the inhibiting of y by z leading to state 4. Once taken, the system moves to another component and reaches the steady cycle $9 \to 10$, where the levels of x and y stabilize at 1 and 0, respectively, and the level of z after some oscillations eventually reaches 0. Note that state 15 represents the unreachable states which apparently are all deadlock states.

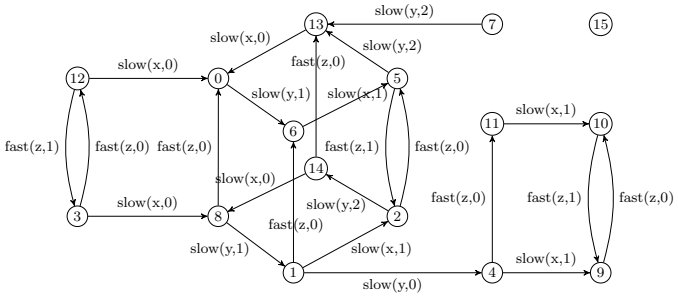


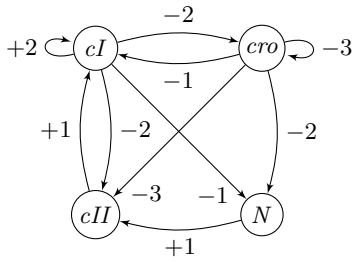
Figure 9: minimized labeled transition system

Thus, bisimulation reduction is helpful in keeping the number of states and transitions low. Naturally, this is an advantage in tool-supported settings as well, in particular because the reduction is efficient and needs to be done only once. The reduced representation of the regulatory network can be taken as starting point for subsequent analysis. Moreover, bisimulation reduction yields a normal form of the modeled system behavior and can be used to check the equivalence of two labeled transition systems. We discuss the usefulness of the latter in the section on tool support.

6. LAMBDA PHAGE

The bistable gene regulatory network controlling the toggle switch from lysogene to lytic behavior has been extensively studied [26]. When binding to a proper part of the promotor area, *CII* proteins yield the transcription of *CI*. In turn, *CI* proteins inhibit, via another agent, destruction of *CII*, thus establishing a positive feedback loop. Also, *CI* in dimerized form stimulates its own activation indirectly and represses *Cro*. The stable state reached on initiation of *CII* brings the bacteriophage into lysogenic state. However under external influences, e.g. change in nutrients or UV irradiation, a protease of the host cell, targeted at *CII* amongst others, becomes active. In absence of *CII*, *Cro* becomes expressed. Dually, *Cro* in dimerized form stimulates its own activation indirectly and represses *CI*. Another stable state is reached making the pathogen to reproduce and eventually lyse the host cell. Thus, in the lysogenic state, *CI* is dominant and *Cro* is repressed. Once Cro gets activated, the system reaches lytic state with *CI* repressed and *Cro* dominant.

Siebert and Bockmayr present in [29] an approach based on timed automata to study gene regulatory networks. Nondeterminism of the system is restricted by adding quantitative constraints. We cast their analysis of the switch of the lambda phage to our qualitative setting. Starting point for their four gene model of the switch is the discrete regulatory network proposed in [33]. By tuning the time requirements on transitions a number of experimental observations reported in [25] could be incorporated. Figure 10 presents the interaction graph and associated activation functions.



R_{cI}	$\emptyset$	cI	cro	cII	cI, cro	cI, cII	cro, cII	cI, cro, cII
K_{cI}	0	0	2	2	2	2	2	2
R_{cro}	$\emptyset$	cI	cro		cI, cro			
K_{cro}	0	2	0		3			
R_{cII}	$\emptyset$	cI	cro	N	cI, cro	cI, N	cro, N	cI, cro, N
K_{cII}	0	0	0	0	0	0	0	1
R_N	$\emptyset$	cI	cro		cI, cro			
K_N	0	0	0		1			

Figure 10: lambda phage regulatory network

Figure 11 depicts the relevant part of the LTS, the node labeled 0 represents the state $(0, 0, 0, 0)$ for cI, cro, cII and N.

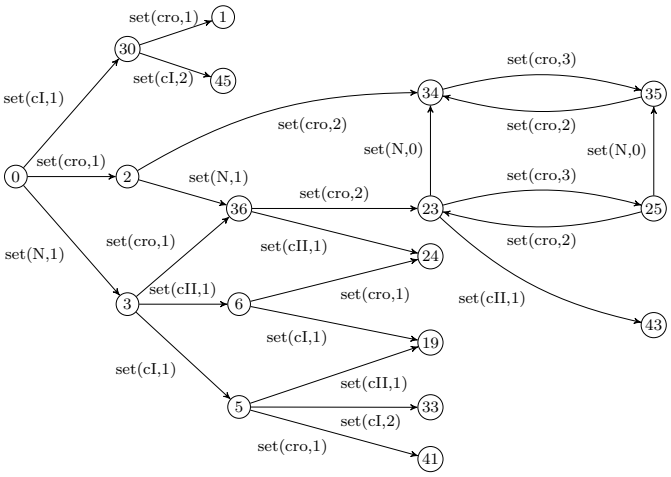


Figure 11: partial LTS of the lambda phage model

Steady state analysis techniques can be exploited to detect attractors [12, 37]. Model checking techniques can be applied too. By marking individual states, e.g. tagging the state $(2, 0, 0, 0)$ with the action *aux2000*, i.e. adding the auxiliary transition $(2, 0, 0, 0) \xrightarrow{aux2000} (2, 0, 0, 0)$ to the LTS generated for the interaction graph and activation functions of Figure 10, the modal formula

$$[true* \,.\, aux2000] \langle\, true \,\rangle false$$

verifies that $(2, 0, 0, 0)$, node 45 in Figure 11, is a stable state. Similarly, the transitions that mark the states $(0, 2, 0, 0)$ and $(0, 3, 0, 0)$, viz. transitions $(0, 2, 0, 0) \xrightarrow{aux0200} (0, 2, 0, 0)$ and $(0, 3, 0, 0) \xrightarrow{aux0300} (0, 3, 0, 0)$, can be used to show that $(0, 2, 0, 0)$ and $(0, 3, 0, 0)$ form a stable cycle, nodes 34 and 35,

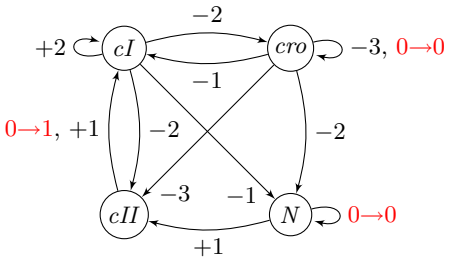


Figure 12: refined model for the lambda phage

via the formula

$$[true^* \,.\, aux0200] \langle\, true \,.\, aux0300 \,\rangle true \wedge$$
$$[true^* \,.\, aux0200 \,.\, \neg aux0200] \langle\, aux0300 \,\rangle true \;\wedge$$
$$[true^* \,.\, aux0300] \langle\, true \,.\, aux0200 \,\rangle true \wedge$$
$$[true^* \,.\, aux0300 \,.\, \neg aux0300] \langle\, aux0200 \,\rangle true$$

expressing that after reaching state $(0, 2, 0, 0)$ state $(0, 3, 0, 0)$ can be reached in one step, and any transition different from the loop tag *aux0200* will lead to $(0, 3, 0, 0)$. Similar for state $(0, 3, 0, 0)$ with respect to state $(0, 2, 0, 0)$.

As advocated in [29], the timed automata model can be tuned further by adjusting the time constraints for the invariants and guards of the timed automata involved. This way, observations reported in [25] for the case of the lambda phage lead to a reduced state transition graph. This is because the genes Cro and N are initially more abundantly expressed then the genes CI and CII. In [29] priority is achieved by adapting the parameters for the time window of specific transitions. In our qualitative approach the same can be achieved by adding the label 0→0 to the existing self-loop for *cro* and to a new loop for N. See Figure 12. Now, we have $A'_{cro,cro} = \{\langle 0, 0 \rangle\}$ as short-term activation set for *cro* on itself. The activation function K_{cro} remains unchanged as *cro* was already a self-stimulus. For N, the short-term activation set is now the singleton $\{\langle 0, 0 \rangle\}$ too. For the activation function we have $K_N(\{cI, cro, N\}) = 1$, taking into account the short-term self-stimulus of N, besides the previous $K_N(\{cI, cro\}) = 1$, while K_N yields zero otherwise as before. Note, no transition is added; the net effect is that the transitions for *cro* and N rising from level 0 to 1 become fast transitions.

Next, we assume fast transitions to have priority over slow ones. One may call this a *maximal progress* interpretation of the generated LTS: slow transitions in a state are discarded when a fast transition is available. Formally, we write

$$s \xrightarrow{a}_{mp} s' \iff s \xrightarrow{a}_{f} s' \vee (s \not\rightarrow_{f} \wedge s \xrightarrow{a}_{s} s')$$

reading, a maximal progress transition is either a fast transition or a slow transition provided there is no fast transition from that state (indicated by the notation $\not\rightarrow_f$). Thus, priority is given to fast transitions. Part of the reduced LTS is displayed in Figure 13. For example, in comparison to Figure 11, slow transitions from state $(0, 0, 0, 0)$ to state $(1, 0, 0, 0)$, nodes 0 and 30, and from state $(0, 1, 0, 0)$ to state $(0, 2, 0, 0)$, nodes 2 and 34, are omitted now.

Timed automata in general, hence also in [29], make use of clock sets that are bound to a location. Therefore, as pointed out by Siebert and Bockmayr, one cannot express with timed automata that synthesis of *CI* has a higher rate

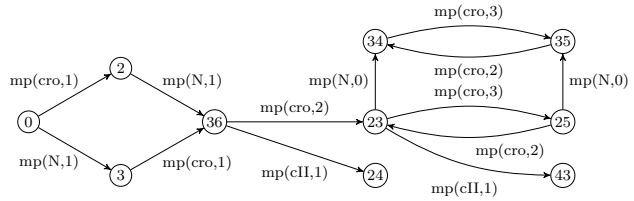


Figure 13: partial LTS under maximal progress

in the presence of *CII* than in its absence (a fact that is crucial for explaining the switch). In our set-up, such a difference can be modeled though. E.g., with the short-term label 0→1 for the stimulus of *cII* on *cI* in the interaction graph in Figure 10, we have different maximal progress for state $(0,1,0,1)$ with *cII* at level 0, as compared to state $(0,1,1,1)$ with *cII* at level 1. Figure 14 highlights the situation.

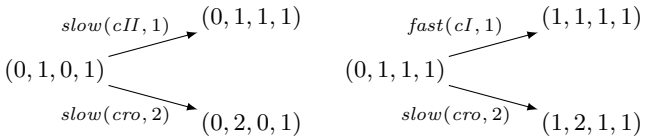


Figure 14: *cII* influence under maximal progress

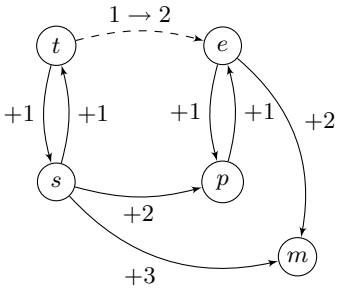
Thus, the third state component *cII* at level 1 indeed makes a difference. In state $(0,1,0,1)$ the system may develop into a lytic direction on expression of *cro* (dependent on the resolution with respect to the alternative transition), in state $(0,1,1,1)$ the system does develop in lysogenic direction on expression of *cI*.

7. TGFβ DRIVEN FIBROSIS

Fibrotic diseases, such as systemic sclerosis, organ fibrosis and Dupuytren's disease, are characterized by excessive production, deposition, and contraction of extracellular matrix which can lead to organ dysfunction. Tissue fibrosis is generally considered to arise due to a failure of the normal wound healing response to terminate. After injury, new connective tissue needs to be synthesized. During this process, fibroblasts proliferate, migrate into inflammatory sites and transform into so-called *myofibroblasts*, which synthesize matrix proteins [36]. The key to unravel tissue fibrosis lies most likely in understanding the growth factors that control the proliferation and differentiation of the myofibroblasts.

Many growth factors have been implied in fibrosis. Of these, transforming growth factor β (TGFβ) and platelet-derived growth factor (PDGF) are considered to be pivotal factors. After wounding or inflammation the release and activation of these cytokines stimulate the production of extracellular matrix proteins which contributes to tissue repair and ideally to the restoration of normal tissue architecture. In many diseases, excessive TGFβ and PDGF signaling contributes to a pathologic excess of tissue fibrosis that compromises normal organ function [5, 20].

TGFβ and PDGF ligands bind extracellularly to their distinct subset of receptors, which transduce their activation signal through phosphorylation to cytosolic proteins. TGFβ predominantly initiates the activation of receptor-regulated Smad proteins. However, it also has been shown to trigger short-term activation of the extracellular signal-regulated kinases1/2 (ERK1/2) [21]. PDGF induces sustained activation of ERK1/2. The activated cytosolic proteins travel to and accumulate in the nucleus where they act as transcription factors and participate in the regulation of target gene expression. Moreover, activated Smad complexes govern the read out of *TGFβ* whereas elevated ERK1/2 signaling leads, via other proteins, to increased *PDGF* expression. This way, two autocrine loops are sustained. Moreover, paracrine signaling through Smad controlled *PDGF* expression leads to an additional level of signal regulation. See [5, 20, 17, 22].



R_t	$\emptyset$	s		
K_t	1	2		
R_s	$\emptyset$	t		
K_s	0	2		
R_p	$\emptyset$	s	e	se
K_p	0	1	1	1
R_e	$\emptyset$	t	p	tp
K_e	0	2	0	2
R_m	$\emptyset$	s	e	se
K_m	0	1	1	2

Figure 15: network for TGFβ/PDGF pathways

TGFβ and PDGF driven fibrosis can be modeled using the discrete regulatory network with short-term effects in Figure 15. We have bounds $B_t = 3$ for TGFβ, $B_s = 4$ for Smad, $B_p = 2$ for PDGF, $B_e = 3$ for ERK1/2, and $B_m = 3$ for myofibroblasts. The activation of myofibroblasts is promoted by Smad and ERK1/2 only at their highest activation levels. All arcs in the network represent activating effects. The arc between t and e models TGFβ induced short-term activation of ERK1/2. In our model, the autocrine loops between respectively TGFβ and Smad, and PDGF and ERK1/2 are the crucial factors in the production of myofibroblasts. We therefore seek to investigate whether the short-term activation of ERK1/2 can have a long-term and potentially irreversible effect on the network state.

Figure 16 shows the corresponding (minimized) LTS. For more clarity we left out the effects on m. State 0 represents an initial activation level: all genes and proteins are at level 0 here, except t which is at level 1. This low initial level is sufficient to start the autocrine loop between t and s; they both increase to 2 in the subsequent steps. Moreover, when t reaches 2, the short-term effect on e comes into play. As can be seen from the path 4→3→6→15→14→16, e shows a peak, i.e., it rises to 2 via fast transitions and then drops again to 0 via slow transitions. This is essentially the formal equivalent of the experimentally observed short-term activation of ERK1/2 induced by TGFβ. However, we were interested in potential long-term effects of short-term activations. Indeed, we can see in our model that if p was already activated by s *before* the short-term effect appeared, e rises to 2 but does not drop again. Intuitively, if there is already a (medium) activation of p, the short-term effect on e is enough to start the autocrine loop between p and e. Hence, the network can shift into two different stable states, viz. 20 and 22. In the former e is at its lowest, and in the latter it is at its highest level. Follow-up experiments need to confirm

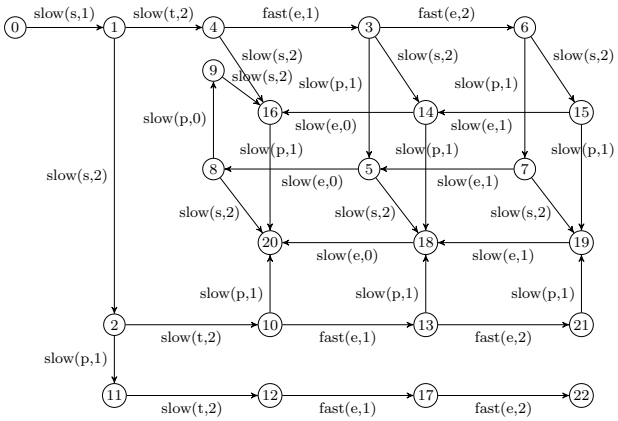


Figure 16: LTS for TGFβ/PDGF network

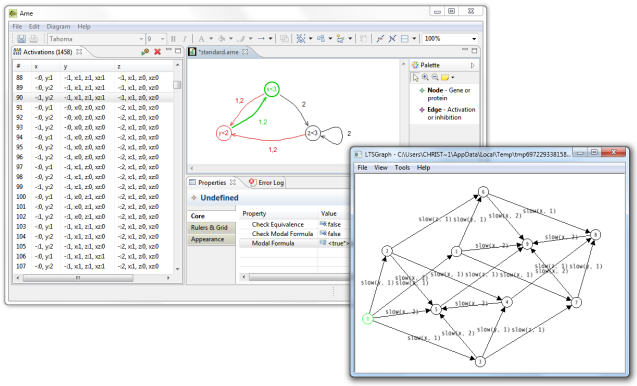


Figure 17: a screenshot of *Arne* and `ltsgraph`

if, for example synthetically, influencing the behavior in 1 and 2 is possible.

We have verified using the Arne tool, discussed in the next section, that the basic model without short-term effects cannot produce such a behavior. The modal formula[1]

$$\langle true^*.slow(t,2).true^*.fast(e,2).true^*.slow(e,0)\rangle true$$

finds all activation functions that produce a peak for e after t has raised. In our model with short-term effects we found 1016 out of 7200 possible activations satisfying this formula. For a standard version of the network with a basic arc between t and e, no such activation was found.

8. TOOL SUPPORT

As the parameters of a regulatory network are usually not known all beforehand, tool support for finding interesting settings is crucial. In the present set-up, model checking techniques and bisimulation reduction are vital for an efficient and automated analysis of network behaviors. For this purpose we have developed the tool *Arne*[2] (Analysis of regulatory networks), depicted in Figure 17. The tool consists of a graphical editor for regulatory networks and an activation function generator. It further integrates with the `mCRL2` [13] tool suite for generating labeled transitions systems, bisimulation reduction, model checking and visualization. From the graphical descriptions of regulatory networks, our tool generates an equivalent description in the `mCRL2` specification language, which is based on the process algebra ACP, extended with data and time.

Since the number of possible activation functions is in general exponential in the size of the network, the tool supports various mechanisms for generating those activations that lead to a specific behavior. In particular, Arne supports filtering of activations based on (i) monotonicity assumptions for certain genes, (ii) model checking and (iii) bisimulation equivalence of the resulting LTS, and (iv) a general requirement for multi-stationarity / homeostasy.

The monotonicity property [31] ensures that an enabled activator or a disabled inhibitor cannot cause a decrease of the activation level. Formally, monotonicity for a gene x reads

$$R_x \subseteq R'_x \Rightarrow K_x(R) \leq K_x(R')$$

In our tool, monotonicity constraints can be associated to individual nodes separately. Although other means exist to restrict activation functions, monotonicity constraints are particularly interesting as they reduce the search space significantly, and do not require to compute the corresponding LTS for selecting an activation function.

Given a regulatory network together with an activation function, our tool transparently generates input for the `mCRL2` model checker and analysis tools. For instance, we use the tool `ltsgraph` of the `mCRL2` tool suite for displaying the corresponding LTS, as shown in the lower right part of Figure 17. Moreover, we can use `mCRL2` to verify temporal properties symbolically. The use of the model checker is also helpful for filtering of activation functions, which is done by Arne automatically. As discussed earlier, bisimulation reduction is not only useful to minimize an LTS, but also to check whether two networks are behavioral equivalent. Our tool uses the `mCRL2` utility `ltscompare` to check whether the labeled transition systems generated from two different activation functions are equivalent, i.e. strongly bisimilar. If this is the case, one of them can be discarded.

Another important kind of constraint is a general requirement for multi-stationarity. In principle, such a filtering constraint can be added using a modal μ-calculus formula. However, in practice for each individual network a separate formula has to be generated and checked for each LTS. Instead, we implemented multi-stationarity checks directly, using an efficient graph algorithm on the labeled transition system.

We have applied the filtering techniques for generating activation functions for the TGFβ/PDGF network in Figure 15, for now excluding the output node m. Without any constraints Arne generates a total number of $3^2 \times 3^4 \times 4^2 \times 2^4 = 209.664$ different activations. By imposing monotonicity on all nodes, the number reduces to 7200. Using the modal formula (7) only 1016 possible activations remain. Additional equivalence checks yields a number of 477 activations, which can be analyzed further.

Finally, in Arne several options can be used to influence the tool output, e.g. bisimulation reduction and self-loop transitions corresponding to zero-changes can be switched on or off, and initial activation levels for agents can be set.

[1]For the basic model, *slow* and *fast* are replaced by *set*.
[2]Freely available at `http://code.google.com/p/arne`.

9. RELATED WORK

In [29] Siebert and Bockmayr exploit timed automata to refine discrete regulatory networks. In their quantitative set-up pseudo-states are introduced that record all the asynchronous changes that may happen from a certain state. Timing constraints are incorporated by combining state invariants and guards to set the window for firing. This way non-determinism is reduced: if the time windows of two transitions do not overlap, one will have priority over the other when both are enabled. Our analysis of the switch of bacteriophage λ has been inspired by the approach of [29], providing a qualitative alternative that applies if exact timing constraints and/or threshold values for activation and inhibition are not available. Siebert and Bockmayr call upon the Uppaal modeling environment. For timed model checking, limited state space reduction is achieved implicitly by considering region graphs.

In [1] a mix of hybrid automata and discrete regulatory networks is proposed for the modeling of delays. Also here, it is argued that time passes for an agent to reach a threshold as well as to express its activating/inhibiting influence. To incorporate such, real-valued variables are added to a system state. The variables evolve over time as governed by a specific slope reflecting a first order approximation of the actual trajectory. Again, the relevant quantitative information, or a sufficiently precise estimation, needs to be available for an exhaustive analysis. However, the approach can also be applied reversely to detect which variables are influenced directly by specific regulators, yielding a system of linear constraints for the slope variables involved. Ahmad c.s. exploit the HyTech tool for reachability analysis and develop a specific algorithm to find all paths between two states under the timing constraints given.

There is a large number of applications of discrete regulatory networks in bio-medical settings, in particular using simulation and attractor finding as analysis techniques. For example, in [11] non-Boolean discrete regulatory networks are used for the analysis of the Hedgehog signaling for the wing disc development in *D. melanogaster* with modeling and simulation support by the GINsim toolset [12]. Dedicated graph reduction techniques to mitigate state space explosion are reported in [24]. The approach is more focused and preserves a number of stability conditions, compared to the general approach based on bisimulation that respects modal μ-calculus.

Petri nets have been widely exploited as a qualitative modeling and analysis framework for biological systems. See [27] for an early reference, [15] for an introduction. We mention [30], where Petri nets are used for a qualitative modeling of tryptophan production in *E. coli*, identifying attractors by means of reachability analysis. Steggles and co-workers [32] seek to optimize by propositional simplification to reduce the Petri net representation of a regulatory network. In a sporulation case study for *B. subtilis* the Petri net model of a Boolean regulatory network is examined using unfolding based model checking techniques. The powerful technique of invariant analysis, both for places and for transitions, is pivotal in the Petri net approach of [28] modeling human iron homeostasis. Extensions of qualitative Petri nets e.g. with localities and range arcs are proposed in [18] for the representations of dynamic membrane structures. In general, for these approaches it is hard to maintain explicit state spaces of reduced size. In [6] the vulva development of *C. elegans* is studied under maximal parallelism assumptions, to reduce the state space by restricting the number of interleaving points.

Various qualitative and quantitative process oriented modeling techniques for biological systems have been combined with model checking. See, e.g. [8, 14, 15, 9]. In the setting of process calculi and membrane computing a number of notions of equivalences have been studied, e.g. for the looping calculus [3], for `bio-`κ [19], for Bio-PEPA [10] as well as [2] for XS-systems. The focus on minimization when employing bisimulation, in particular for discrete regulatory networks, is distinguishing for the present paper.

10. CONCLUSIONS

We presented an extension of discrete regulatory networks for modeling short-term effects. Activation of an agent not only depends on the current activation levels, but also on their discrete differentials, modeling a recent change. Additional expressiveness is gained by considering the network behavior under a maximal progress condition. Using this approach, two qualitative timescales, i.e., steep and moderate changes can be distinguished. Also, priority can be given to transitions triggered by activation sets including or excluding specific stimuli.

Bisimulation minimization helps to deal with the enlarged state space. The reduction in state and transitions is substantial as states are anonymous in our approach. Bisimulation equivalence is also applied when scanning over all possible activation functions for a given interaction graph. This limits the number of candidate activations that needs to be considered in further detail. We implemented our approach in a prototype with graphical interface that uses the `mCRL2` tool suite as a back-end. Either the model checking facilities of `mCRL2` and native algorithms for dedicated tasks such as attractor finding can be applied, or the generated label transition system or systems can be exported to other analysis software. We discussed the switch of the lambda phage and a bio-medical case related to myofibrosis to illustrate the approach.

As future work we plan to implement additional mechanisms to steer the generation of activation functions, e.g. using a constraint language. Minimization modulo maximal progress equivalence, possibly for more than two timescales as considered here, is a natural next step. As being close to branching bisimulation, an equivalence supported by the `mCRL2` tool set, we are confident that technically this can be achieved. Also, filtering based on maximal progress reduction is planned. However, further validation by other biological or bio-medical case studies, with a larger number of agents and more complex interaction, is essential. Therefore, techniques to trace back from a reduced LTS to the explicit state-transition model, based on the interaction graph and associated activation function, are to be worked out.

Acknowledgments First and second author supported by the Dutch Organization for Scientific Research, grants NWO GLANCE project WoMaLaPaDiA and NWO 918.66.606.

11. REFERENCES

[1] J. Ahmad et al. Analysing formal models of genetic regulatory networks with delay. *International Journal of Bioinformatics Research and Applications*, 4:240–262, 2008.

[2] M. Antoniotti et al. Taming the complexity of biochemical models through bisimulation and collapsing: theory and practice. *Theoretical Computer Science*, 325:45–67, 2004.

[3] R. Barbuti et al. Bisimulation congruences in the calculus of looping sequences. In K. Barkaoui, A. Cavalcanti, and A. Cerone, editors, *Proc. ICTAC 2006*, pages 93–107. LNCS 4281, 2006.

[4] G. Bernot et al. Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology*, 229(3):339–347, 2004.

[5] J. C. Bonner. Regulation of PDGF and its receptors in fibrotic diseases. *Cytokine & Growth Factor Reviews*, 15(4):255–273, 2004.

[6] N. Bonzanni et al. Executing multicellular differentiation: quantitative predictive modelling of *C. elegans* vulval development. *Bioinformatics*, 25:2049–2056, 2009.

[7] J. Bradfield and C. Stirling. Modal mu-calculi. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *The Handbook of Modal Logic*, pages 721–756. Elsevier, 2006.

[8] N. Chabrier-Rivier et al. Modeling and querying biochemical interaction networks. *Theoretical Computer Science*, 325:25–44, 2004.

[9] F. Ciocchetta and J. Hillston. Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science*, 410:3065–3084, 2009.

[10] V. Galpin and J. Hillston. Equivalence and discretazation in Bio-PEPA. In P. Degano and R. Gorrieri, editors, *Proc. CMSB 2009*, pages 189–204. LNBI 5688, 2009.

[11] A. González, C. Chaouiya, and D. Thieffry. Logical modelling of the role of the Hh pathway in patterning of the *Drosophila* wing disc. *Bioinformatics*, 24:i234–i240, 2008.

[12] A. González et al. GINsim: A software suite for the qualitative modelling, simulation and analysis of regulatory networks. *Biosystems*, 84:91–100, 2006.

[13] J.F. Groote et al. The formal specification language `mCRL2`. In E. Brinksma et al., editor, *Methods for Modelling Software Systems*. IBFI, Schloss Dagstuhl, 2007.

[14] J. Heath et al. Probabilistic model checking of complex biological pathways. *Theoretical Computer Science*, 391:239–257, 2008.

[15] M. Heiner, D. Gilbert, and R. Donaldson. Petri nets for systems and synthetic biology. In M. Bernardo, P. Degano, and G. Zavattaro, editors, *Formal Methods for Computational Systems Biology*, pages 215–264. LNCS 5016, 2008.

[16] S.A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22:437–467, 1969.

[17] S.J. Kim et al. Autoinduction of transforming growth factor β1 is mediated by the AP-1 complex. *Molecular and Cellular Biology*, 4:1492–7, 1990.

[18] J. Kleijn and M. Koutny. A Petri net model for membrane systems with dynamics structure. *Natural Computing*, 8:781–796, 2009.

[19] C. Laneve and F. Tarissan. A simple calculus for proteins and cells. *Theoretical Computer Science*, 404:127–141, 2008.

[20] A. Leask and D.J. Abraham. TGFβ signaling and the fibrotic response. *FASEB Journal*, 18:816–27, 2004.

[21] M. K. Lee et al. TGFβ activates Erk MAP kinase signalling through direct phosphorylation of ShcA. *EMBO Journal*, 26:3957–3967, 2007.

[22] N.T. Liberati et al. Smads bind directly to the jun family of AP-1 transcription factors. *PNAS*, 96:4844–9, 1999.

[23] R. Milner. *Communication and Concurrency*. Prentice Hall International, 1989.

[24] A. Naldi et al. A reduction of logical regulatory graphs preserving essential dynamical properties. In P. Degano and R. Gorrieri, editors, *Proc. CMSB 2009*, pages 266–280. LNBI 5688, 2006.

[25] A. Oppenheim et al. Switches in bacteriophage lambda development. *Annual Reviews of Genetics*, 39:409–429, 2005.

[26] M. Ptashne. *A Genetic Switch: Phage Lambda Revisited*. Cold Spring Harbor, 3rd edition, 2004.

[27] V.N. Reddy, M.N. Liebman, and M.L. Mavrovouniotis. Petri net representations in metabolic pathways. In L. Hunter, D.B. Searls, and J.W. Shavlik, editors, *Proc. ISMB*, pages 328–336. AAAI, 1993.

[28] A. Sackmann et al. An analysis of the Petri net based model of the human body iron homeostatis process. *Computational Biology and Chemistry*, 31:1–10, 2007.

[29] H. Siebert and A. Bockmayr. Temporal constraints inthe logical analysis of regulatory network. *Theoretical Computer Science*, 391:258–275, 2008.

[30] E. Simão et al. Qualitative modelling of regulated metabolic patyways: application to the tryptophan biosynthesis in *E. coli*. *Bioinformatics*, 21:ii90–ii96, 2005.

[31] H. Snoussi and R. Thomas. Logical identification of all steady states: The concept of feedback loop characteristic states. *Bulletin of Mathematical Biology*, 55:973–991, 1993.

[32] L.J. Steggles, R. Banks, O. Shaw, and A. Wipat. Qualitative modelling and analysing genetic regulatory networks: A Petri net approach. *Bioinformatics*, 23:336–343, 2007.

[33] D. Thieffry and R. Thomas. Dynamical behavior of biological regulatory networks II: Immunity control in bacteriophage lambda. *Bulletin of Mathematical Biology*, 57:277–297, 1995.

[34] R. Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42:563–585, 1973.

[35] R. Thomas and R. D'Ari. *Biologial feedback*. CRC Press, 1990.

[36] T.A. Wynn. Cellular and molecular mechanisms of fibrosis. *Journal of Pathology*, 214:199–210, 2008.

[37] S.-Q. Zhang et al. Algorithms for finding small attractors in Boolean networks. *Journal on Bioinformatics and Systems Biology*, page 20180, 2007.