Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science 293 (2013) 67–81

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Timed Modelling of Gene Networks with Arbitrarily Precise Expression Discretization

S. Van Goethem, J.-M. Jacquet

Faculty of Computer Science, University of Namur
Namur, Belgium

L. Brim, D. Šafránek [1]

Faculty of Informatics, Masaryk University
Brno, Czech Republic

Abstract

In this paper, a novel approach to discrete modeling of gene regulatory networks is presented. The approach is based on timed automata and is new in: (i) reflecting discrete abstraction of gene expression with arbitrary granularity, (ii) combining boolean logic with approximation of Hill kinetics. This is achieved by introducing delays that change dynamically with respect to current activity levels of regulating genes. The approach is implemented in UPPAAL and evaluated on benchmark models and on a biological case study.

Keywords: Discrete modeling, gene regulatory network, Boolean logic, UPAAL

1 Introduction

The central topic in systems biology is analysis of dynamics imposed by complex interaction networks. The elementary layer of these networks is formed by genes and their mutual interactions through transcriptional regulation [15]. This phenomenon is captured by *gene regulatory networks* (GRNs). These networks control gene expression dynamics running on a relatively slow time-scale while determining functional modes of the cell.

There are several conceptually different approaches for modelling of GRN dynamics. The most common approach is that of *ordinary differential equations*

[1] The work has been partially supported by the Czech Grant Agency grant no. GAP202/11/0312 and by EC OP project No. CZ.1.07/2.3.00/20.0256.

doi:10.1016/j.entcs.2013.02.019

(ODE) describing deterministic (population average) behaviour in continuous manner. Since even a simple interaction among two genes introduces a necessary non-linear term into the ODE of the affected gene, analytical solution of ODE models of GRNs is impossible, thus leaving simulation as the only practical method. Moreover, continuous models require quantitative knowledge in terms of kinetic coefficients, which are unknown and very difficult to measure *in vitro*. As a consequence, various abstraction approaches have been developed to make GRN models more convenient for analysis under data uncertainty [21,18,20,10,8].

All of these models are purely qualitative (and discrete) provided that the aspect of time (resp. velocity of the dynamics) is entirely abstracted out, thus leading to strong approximations. However, although rates of individual processes in transcription are not known, they can be estimated synthetically to achieve behaviour observed *in vitro*. To this end, it appeared important to make a step back in the abstraction by extending discrete models with the quantitative aspect of time.

The most direct approach that appeared recently is that of Siebert et al. [17] further extended by Batt et al. [3]. It relies on timed automata implemented, in the former case, by employing the UPPAAL tool [4], and in the latter case, by employing IF tool suite [7]. The approach employs multi-value discrete model which is extended by time-delays representing deadlines for exiting individual discrete configurations of genes. Every discretized gene expression level in the system is statically assigned a minimal and maximal delay determining the allowed time range of exit deadline. Such a static treatment of delays is sufficient for boolean models with minimal number of levels considered in discrete abstraction of gene expression. However, to reflect the dynamics appropriately when the model precision is allowed to be set to arbitrary number of discrete levels, time-delays defined for a regulated gene should be dynamically sensitive to actual configuration (expression values) of regulating genes (*regulators configuration*).

In [16], a general framework for abstracting continuous systems by timed automata is introduced. Under that framework, an ODE model of dynamical system is rigorously transformed into a timed automaton representing over-approximation of continuous behaviour. However, supported non-linear ODE models must fit in the class of multi-affine systems. Since GRNs are modeled by means of higher non-linear systems based on Hill kinetics, the approach cannot be directly employed.

In a different context, another piece of work [6] closely related to our treatment presents a timed automata based approach to model signalling pathways with UPPAAL. The approach is (under)approximative in terms of maximal and minimal delay being degraded to an exact delay. A significant contribution of the underlying model is that it makes delays sensitive to the actual configuration of signal states. A technical problem of the timed model employed is spurious instability coming from non-determinism in parallel processes controlling regulations. In particular, regulation processes generate discrete events driving the regulated genes to increase or decrease expression levels provided that production competes with degradation. Each event is repetitively generated with a particular non-zero rate. For example, a positively regulated gene has a high rate of production, but also a small non-

zero rate of degradation. Non-deterministic generation of increase and decrease events causes irregular gene expression profiles with oscillations over two adjacent expression levels. A detailed analysis of this problem is provided in [11] including extensions of the formalism that attempt to overcome the problem.

In this paper we present a new approach to timed modeling of GRNs that avoids problems mentioned above. In contrast to [16], our contribution lies on the computational side. In particular, we investigate construction of the timed model for Hill kinetics by using the ideas formalized in the piece-wise affine approximation. However, this paper is a preliminary step – we do not present a formal abstraction technique, but provide an experimental evaluation of our treatment supported by a prototype tool chain. From the technical viewpoint, features of [3] and [6] are unified under our approach. In particular, we support arbitrarily precise discretization of gene expression while treating delays dynamically. Conceptually, our model is based on incorporating rate control into the original discrete model [21] according to piece-wise affine model [18]. Instability problems encountered in [6] are eliminated. Additionally, our approach allows mixing of boolean operations over regulatory interactions with algebraic summation of their effects.

Many approaches to formalizing dynamics of gene networks have been identified in the domain of hybrid systems. The main idea is based on the observation that regulation function of a gene has a sigmoidal shape and therefore can be approximated by step-functions or ramp-functions [14,12]. Overall continuous dynamics of gene expression is abstracted into a finite number of local modes through which the (hybrid) system traverses according to the approximated sigmoidal regulatory control [10,2]. Mathematically, these approaches are based on piece-wise affine or piece-wise multi-affine models. Naturally, analysis methods developed for hybrid systems can be employed [9]. However, efficient quantitative analysis of hybrid systems is computationally demanding. Lesser precise models fitting the domain of timed automata can therefore present a suitable solution meeting the current biological knowledge.

2 Background

2.1 Gene Regulatory Networks

2.1.1 Topology of gene regulatory networks

Let $G = \{g_1, ..., g_n\}$ be a set of *genes*, and let $P = \{p_1, ..., p_n\}$ be a set of *proteins*. We consider that a protein p_i is a product expressed from g_i by transcription. Expression of g_i can depend on concentration of some proteins in P which are *transcription factors* (TFs) regulating expression of g_i. TFs that enhance expression of g_i are called *positive regulators (activators)*. TFs that inhibit expression of g_i are called *negative regulators (inhibitors)*.

We formally represent a GRN by a directed graph $\langle G, E\rangle$ with edges displaying *regulations*, $E \subseteq G \times G$. Each edge is labelled with a sign $+$, resp. $-$, indicating the regulation type (activation, resp. inhibition). Considering an edge $\langle g_i, g_j\rangle$, g_i represents the *regulator* and g_j the *regulated gene*. An edge $\langle g_i, g_j\rangle \in E$ is denoted

e_{ij}, its label is denoted σ_{ij}, $\sigma_{ij} \in \{+, -\}$. An example of a GRN is depicted in Fig. 1(ab). We use the notation $P_j^+ = \{p_i \in P \mid e_{ij} \in E \wedge \sigma_{ij} =' +'\}$ and $P_j^- = \{p_i \in P \mid e_{ij} \in E \wedge \sigma_{ij} =' -'\}$ for the set of all activators (resp. inhibitors) of g_j.

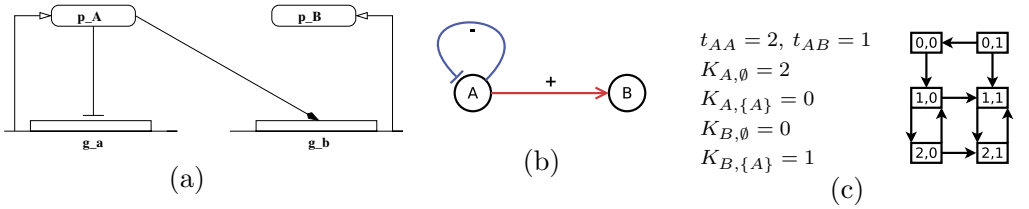


Fig. 1. (a) An example of a network where a negatively self-regulated gene g_A activates gene g_B. (b) The respective GRN as a simple graph (the +-edge represents an activation, the −-edge represents an inhibition). (c) Reachability graph generated for the given settings.

2.1.2 Discrete semantics of gene regulatory networks

For discrete (untimed) semantics of GRNs, we consider the *boolean model* of René Thomas [21]. In this model, the concentration of protein p_i is discretely abstracted by the *activity level* l_i with $l_i \in [0, r_i] \subset \mathbb{N}$ where r_i is the *maximal activity level of* p_i. Each edge $e_{ij} \in E$ is associated a *threshold* t_{ij} $(0 < t_{ij} \leq r_i)$ indicating minimal l_i such that e_{ij} is active (e_{ij} is active iff $l_i \geq t_{ij}$). For $r_i > 1$ it is assumed $\forall k \in [1, r_i]\,(\exists j(e_{ij} \in E) \wedge (t_{ij} = k))$, a convention reducing the number of considered activity levels to a necessary minimum (in our framework later we relax this constraint).

The semantics of a GRN is determined by a *regulatory logic* defined as the set $\{K_{i,\rho} \mid 1 \leq i \leq n, \rho \subseteq P_j^+ \cup P_j^-\}$ where $K_{i,\rho}$ denotes the *target activity level* of g_i when regulated by all proteins in the *regulatory context* ρ, $0 \leq K_{i,\rho} \leq r_i$. $K_{i,\rho}$ identifies a state towards which p_i converges under regulatory context ρ.

Once the regulatory logic is set, the semantics of a GRN is defined as the transition system given as a tuple $\langle S, T, s_0\rangle$ where $S = \prod_{i=1}^{n}\{0, ..., r_i\}$ is the set of states with $s_0 \in S$ being the initial state and $T \subseteq S \times S$ is the transition relation defined as follows.

First denote $l_i(s)$ the activity level of g_i in state $s \in S$. Assume $e_{ij} \in E$. We say that g_i is a *resource* for g_j in s if $\sigma(e_{ij}) =' +'$ and $l_i(s) \geq t_{ij}$, or $\sigma(e_{ij}) =' -'$ and $l_i(s) < t_{ij}$. Let $Re(s, g_j)$ denote the set of all resources for g_j in s. There is a transition $s \to s' \in T$ iff one of the following rules holds:

(i) there exists j such that $K_{j,Re(s,g_j)} > l_j$ and $l_j(s') = l_j(s) + 1$ (increase),

(ii) there exists j such that $K_{j,Re(s,g_j)} < l_j$ and $l_j(s') = l_j(s) - 1$ (decrease).

Moreover, it is required that s differs from s' in the expression value of just one gene. This gives the widely-used *asynchronous semantics* [20,5]. Example of the transition system for a particular setting of regulatory logic is given in Fig. 1(c). Asynchronous semantics results in non-deterministic transition systems where non-determinism represents choice from the set of active regulatory contexts and selection of the updated gene (asynchrony).

Non-trivial regulatory contexts represent multi-input logic operators for a regulated gene. Interpretation is boolean – a context is enabled/disabled by resources. In doing so, the *logical cooperation* of regulators is modeled.

2.1.3 Piecewise-affine model

The model based on piecewise-affine differential equations (PADE) provides a discrete abstraction with exact relation to continuous ODE models. As a base for our timed model, we use Snoussi's model [18]. Here the concentration of a protein p_j is considered continuous and denoted by $[p_j]$, $[p_j] \in [0, r_i] \subseteq \mathbb{R}$, where $r_i \in \mathbb{R}^+$ is maximal concentration. Regulation thresholds are also interpreted in $\mathbb{R}^+$. Additionally, for each regulation e_{ij}, a *rate* β_{ij} is specified, representing contribution of regulator p_i to positive change (production) of $[p_j]$. For each protein p_j, rates α_j and κ_j are defined where α_j is *dilution/degradation rate* of p_j, and κ_j is *basal production rate*. The dynamics of a protein p_j is defined by the following differential equation:

$$\frac{d[p_j]}{dt} = \kappa_j + \sum_{i \in P_j^+} \beta_{ij} \cdot \theta([p_i] > t_{ij}) + \sum_{i \in P_j^-} \beta_{ij} \cdot \theta([p_i] < t_{ij}) - \alpha_j [p_j] \qquad (1)$$

where $\theta(\cdot)$ returns 1, if the condition $(\cdot)$ is true, or 0, otherwise.

In comparison to boolean model, PADE model does not abstract from time and protein concentration, the abstraction approximates continuous Hill kinetics by regulatory logic while time and its effect on concentration dynamics is preserved in the form of rates. For a given initial state, there is a unique PADE trajectory representing time behaviour of all proteins. In contrast, from the corresponding initial state in the boolean model, there are several paths providing untimed abstractions which may represent unrealistic behaviours.

2.1.4 Timed models

To discard unrealistic paths in the boolean model, time constraints reflecting transition rates have been introduced and implemented using timed automata [17,3]. For activity level l $(0 \leq l \leq r_i)$ of p_i, two delays $\delta_{i,l}^{\downarrow}$ and $\delta_{i,l}^{\uparrow}$ are specified. $\delta_{i,l}^{\downarrow}$ (resp. $\delta_{i,l}^{\uparrow}$) is the delay necessary for p_i to switch from l to $l-1$ (resp. $l+1$). E.g., when considering the state $\langle 1, 0 \rangle$ in Fig. 1c, the decision between switching to $\langle 2, 0 \rangle$ or $\langle 1, 1 \rangle$ depends on the rate of p_A and p_B production which can be different. In this case, $\delta_{A,1}^{\uparrow}$ is compared with $\delta_{B,0}^{\uparrow}$ and transition with the shortest delay is chosen.

Nevertheless, the approach mentioned above still relies on strong abstractions. First, each pair of delays $(\delta_{i,l}^{\downarrow}, \delta_{i,l}^{\uparrow})$ is shared for all regulatory contexts of g_i. Moreover, to precisely model time behaviour we also need to distinguish delays for individual regulators configurations activating a given context. E.g., for p_B in Fig. 1 and the context $\{A\}$, we should distinguish situation $l_A = 1$ and $l_A = 2$. Both activate the regulatory context $\{A\}$ of p_B. However, the rate of p_B production will be probably faster in the latter case due to stronger influence of the activator.

Second, protein time behaviour is not exactly reflected. In particular, each level change causes the history of protein behaviour to be lost. For example, assume p_i at level l_i is associated with delays satisfying $\delta_{i,l_i}^{\downarrow} < \delta_{i,l_i}^{\uparrow}$. Then exact production time

of p_i is lost once $\delta^{\downarrow}_{i,l_i}$ is reached and p_i level decreased. This problem is overcome in [17,3] by generalizing each delay δ to interval $[0, \delta]$ at the price of adding many unrealistic paths including Zeno paths.

3 A New Timed Approach for GRNs

To overcome the problems mentioned in Section 2.1.4, we propose an approach that extends boolean model with time aspects adopted from the PADE model. In contrast to the boolean model, our approach works with arbitrary number of arbitrarily distributed activity levels discretizing the protein concentration domain. This refinement brings the abstraction of regulatory modes closer to a continuous model, namely, the effect of an activation can be considered stronger with increasing level of the respective regulator (as modelled in Hill kinetics). Each particular configuration of regulators is assigned with an individual time delay affecting the activity of regulated gene. For a given configuration of regulators, the time delay is obtained as a value reciprocal to the corresponding rate in PADE model.

Delays are generalized to intervals of the form $[\delta^{min}, \delta^{max}]$ representing delay uncertainty. Moreover, for $\delta^{min} = 0$ delay intervals ensure conservative abstraction wrt discrete approximation of continuous concentration. In the continuous system, exiting the range of a discrete level from two different concentration values under this level takes different time.

Additionally, we consider logical cooperations coming from the boolean model. In our setting, a *logical cooperation* guards a certain portion of the protein production rate. It is evaluated as a boolean condition over the Cartesian product of regulating proteins activity levels. Formally, for a gene g_j regulated cooperatively by regulators $P_{cop_j} \subseteq P_j^+ \cup P_j^-$ a *cooperation c for gene* g_j is defined as a boolean function $\Theta_{cop_{j_c}}$ given as a logical formula built over propositions of the form $(l_i \odot t_{ij})$ where $\odot \in \{<, >, \leq, \geq\}$, $p_i \in P_{cop_j}$, and $t_{ij} \in [1, r_i]$ is a threshold. A simple example of a cooperation for g_z consists of the conjunction of two activators p_x and p_y. Such cooperation is enabled only if both proteins are active at the same time. This is encoded by $\theta((l_x \geq t_{xz}) \wedge (l_y \geq t_{yz}))$. The set of all cooperations for gene g_j is denoted by C_j.

3.1 Formalization of Time Delays

For a GRN $\langle G, E\rangle$, each gene product p_j is assigned a maximal number of activity levels r_j which is now arbitrary. Timed semantics of p_j dynamics is based on delay determined for current regulators configuration and is given as a value reciprocal to the rate $\frac{d[p_j]}{dt}$ in the corresponding PADE model.

Formally, let $\omega \in \{'min', 'max'\}$ be a label denoting the minimal/maximal delay. For p_j at current activity level l_j, *delay* to next change of p_j activity level, δ^{ω}_j, is determined by the following equation ($sgn(\delta^{\omega}_j)$ indicates the direction of the expected change):

$$\frac{1}{\delta^{\omega}_{j}} = \underbrace{\frac{1}{\delta^{\omega}_{\kappa_j}}}_{\substack{\text{(i) basal}\\ \text{production}}} + \underbrace{\sum_{i\in P_j^+}\sum_{k=1}^{l_i}\frac{1}{\delta^{\omega}_{k,\beta_{ij}}}}_{\text{activation (ii)}} + \underbrace{\sum_{i\in P_j^-}\sum_{k=0}^{l_i}\frac{1}{\delta^{\omega}_{k,\beta_{ij}}}}_{\text{inhibition (ii)}} + \underbrace{\sum_{c\in C_j}\frac{\theta(\Theta_{cop_{j_c}})}{\delta^{\omega}_{cop_{j_c}}}}_{\substack{\text{cooperative}\\ \text{activation (iii)}}} + \underbrace{\sum_{k=1}^{l_j}\frac{1}{\delta^{\omega}_{k,\alpha_j}}}_{\substack{\text{degradation}\\ \text{(iv)}}} \quad (2)$$

where

(i) $\delta^{\omega}_{\kappa_j}$ is the delay implied by the (constant) basal production rate of p_j, $\delta^{\omega}_{\kappa_j} > 0$.

(ii) $\left|\delta^{\omega}_{k,\beta_{ij}}\right|$ is the delay implied by the regulator p_i satisfying $l_i = k$. If $p_i \in P_j^+$ then $\forall k \in [1, r_i]$: $\delta^{\omega}_{k,\beta_{ij}} > 0$ and $\sum_{k=1}^{r_i} 1/\delta^{\omega}_{k,\beta_{ij}} > 0$. If $p_i \in P_j^-$ then $\delta^{\omega}_{0,\beta_{ij}} > 0$, $\forall k \in [1, r_i] : \delta^{\omega}_{k,\beta_{ij}} < 0$.

(iii) effect of cooperation c is determined by delay $\delta^{\omega}_{cop_{j_c}}$ satisfying $\delta^{\omega}_{cop_{j_c}} > 0$.

(iv) $|\delta^{\omega}_{k,\alpha_j}|$ is the delay implied by the dilution/degradation of p_j at level $l_j = k$, $\forall k \in [1, r_i] : \delta^{\omega}_{k,\alpha_j} < 0$.

For any neglected delay in a particular configuration, the value is considered diverging ($+\infty$ or $-\infty$). It is also supposed that every delay satisfies $\delta^{min} \leq \delta^{max}$.

3.2 Implementation in Extended Timed Automata Framework

The timed model is represented compositionally in terms of UPPAAL processes (communicating timed automata) provided that there is an individual process reflecting the behaviour of each protein. For protein p_j, the respective process is associated with clocks c_j^{down} and c_j^{up}, controlling level decrease and increase, respectively. Clocks are employed to stopwatch the delay to the next change of p_j activity level. We employ two strategies in modeling clocks – stopwatch and non-stopwatch, both described in more detail in Section 3.2.1. Time delay is specified by the interval $[\delta_j^{min}, \delta_j^{max}]$. Current values of $\delta_j^{min}, \delta_j^{max}$ are atomically updated whenever some protein changes its activity level. There are two strategies for updating $\delta_j^{min}, \delta_j^{max}$ – selecting from a precomputed array or computing delays on-the-fly. We have implemented both strategies in our UPPAAL timed automata framework.

A general process scheme is displayed in Fig. 2(left). The scheme is displayed for 3-level protein. It can be trivially generalized to n levels. A concrete instance is shown in Fig. 2(right). The process is assigned an output broadcast channel to which information about each level change of p_j is sent. For each regulator in $P_j^+ \cup P_j^-$, the process is assigned an input channel for receiving messages notifying (potential) changes of regulators configuration. Self-regulations are managed by self-communication through local process variables.

For each activity level k, $k \in [0, r_i]$, there is a *level location* L_k representing p_j at level $l_j = k$, and an *update location* U_k, where regulators configuration changes are processed. Level changes are represented by red (increasing level) and blue (decreasing level) transitions in Fig. 2. Under constant regulators configuration, level updates are achieved directly by transitions among level locations. When the regulatory configuration is changed, level update is obtained indirectly by switching to the respective update location (where delays are updated) and then to a particular level location. Red and blue transitions inform other processes through

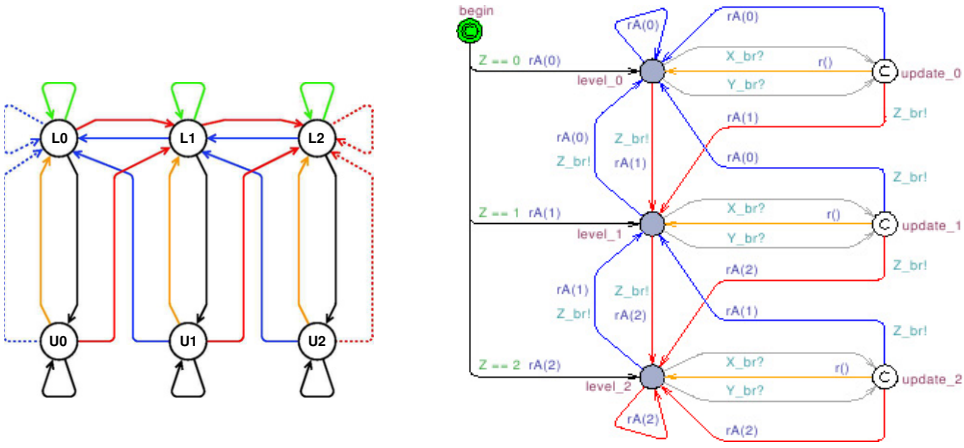


Fig. 2. (left) Automata template for 3-level protein processes. (right) Instance for a protein p_Z regulated by p_X, p_Y; self-transitions are ignored for the sake of lucidity. Level changing locations are denoted blue (decrease level) and red (increase level). Regulators configuration information is received on broadcast channels X_{br}, Y_{br}. Grey and orange transitions manage the received information and implement update of the process delay variables.

the output channel. After any level change, time delays $\delta_j^{max}, \delta_j^{min}$ are updated to reflect the newly achieved regulators configuration. Therefore each such a transition is accompanied by procedure $rA()$ that performs all necessary updates and clock resets (depending on the clock strategy). Considering update locations committed ensures delays to be updated atomically.

A level location is switched to the respective update location if a message is received on some input channel (black transitions). Effect of these transitions is immediate update of δ_j^{min} and δ_j^{max} according to Formula 2. In update state U_k it is decided whether the changed regulators configuration will lead to a level change or not. In the negative case, a regret transition (orange) leads back to L_k causing clock manipulation managed by procedure $r()$ (depending on the clock strategy).

Update locations have self-transitions (black) that ensure reupdating of delays whenever the regulators configuration changes. Self-transitions on level locations (green) are employed only in the case of stopwatch clock strategy. Their meaning is explained in Section 3.2.1. Finally, dotted transitions treat "blind" level updates ensuring the system never exceeds the range $[0, r_j]$.

3.2.1 Time Constraints

Next we describe the mechanism of time constraints reasoning about clocks c_j^{up}, c_j^{down} wrt actual delays $\delta_j^{min}, \delta_j^{max}$. Under *stopwatch strategy*, for each p_j both clocks c_j^{up}, c_j^{down} are considered as stopwatches satisfying that just one is active in any point in time provided that c_j^{up} (resp. c_j^{down}) keeps the total time p_j spent increasing (resp. decreasing) since the initial state. This strategy precisely reflects timed behaviour at the extent of limiting the analysis tasks to simulation. Under this treatment, clock manipulation procedures $r(), rA()$ ensure activation of the clock appropriate for current values of delays while freezing the second clock. In *non-stopwatch strategy*, the respective clocks are reset by procedures $r(), rA()$ invoked

constraint	stopwatch	non-stopwatch
L_k invar.	$(c_j^{up} - c_j^{down} \leq \Delta_{up_j}^{max}) \wedge$ $(c_j^{down} - c_j^{up} \leq \Delta_{down_j}^{max})$	$(c_j^{up} \leq \Delta_{up_j}^{max}) \wedge$ $(c_j^{down} \leq \Delta_{down_j}^{max})$
red trans.	$c_j^{up} - c_j^{down} \geq \Delta_{up_j}^{min}$	$c_j^{up} \geq \Delta_{up_j}^{min}$
blue trans.	$c_j^{down} - c_j^{up} \geq \Delta_{down_j}^{min}$	$c_j^{down} \geq \Delta_{down_j}^{min}$

	$down_j$	up_j	$\Delta_{down_j}^{min}$	$\Delta_{up_j}^{min}$	$\Delta_{down_j}^{max}$	$\Delta_{up_j}^{max}$
(1)	`ff`	`tt`	∞	δ_j^{min}	∞	δ_j^{max}
(2)	`tt`	`ff`	$\lvert\delta_j^{min}\rvert$	∞	$\lvert\delta_j^{max}\rvert$	∞
(3)	`tt`	`tt`	$\lvert\delta_j^{min}\rvert$	$\lvert\delta_j^{min}\rvert$	δ_j^{max}	δ_j^{max}
(4)	`ff`	`ff`	∞	∞	∞	∞

Table 1
Settings of level state invariants and transition time constraints (left). Settings of delay variables for individual situations of delay values (right).

with any level update and thus measure only time from the last level change. Within this strategy timed behaviour is strongly approximated but entirely fits the basic timed automata framework which allows formal analysis, e.g., model checking. In our UPPAAL framework we have developed a model supporting the application of both strategies. Figure 3 shows the difference between the two strategies.

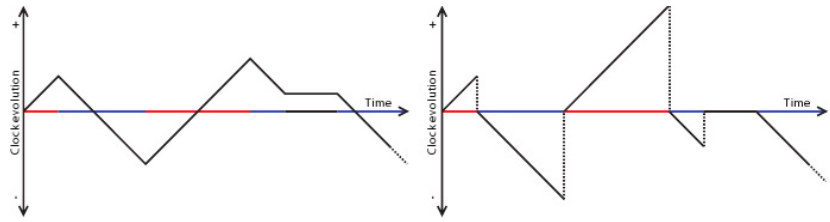


Fig. 3. Evolution of the difference $c^{up} - c^{down}$ in the stopwatch strategy (left) compared with the corresponding approximation in the non-stopwatch strategy (right). Red and blue segments of X-axis correspond to times when just a single clock is active. Concurrent activity is denoted by black segments.

In any system configuration, $\delta_j^{min}, \delta_j^{max}$ satisfy either $\delta_j^{max} \geq \delta_j^{min} > 0$ (1), $\delta_j^{min} \leq \delta_j^{max} < 0$ (2), $\delta_j^{min} < 0 < \delta_j^{max}$ (3), or $\delta_j^{min} \to -\infty, \delta_j^{max} \to \infty$ (4). To detect these situations, we supply the p_j process with local boolean variables $up_j, down_j$ such that up_j is true iff $\infty > \delta_j^{max} > 0$ and $down_j$ is true iff $-\infty < \delta_j^{min} < 0$. Additionally, we introduce delay variables $\Delta_{up_j}^{min}$, $\Delta_{up_j}^{max}$, $\Delta_{down_j}^{min}$, $\Delta_{down_j}^{max}$ to allow easy setting of time constraints reflecting each of the situations. Definition of time invariants and constraints for states/edges of p_j process is given in Table 1(left). Table 1(right) describes settings of delay variables for all mentioned situations.

Constraints for situations (1), (2), (4) directly implement expected behaviour. Situation (3) is more intricate. It happens just when delay uncertainty allows both degradation and production. In non-stopwatch strategy, the behaviour is naturally modelled by non-determinism. In stopwatch case, concurrent activity of both clocks is solved by non-deterministically selecting just one of the clocks c_j^{up}, c_j^{down} to be active. This is achieved by the effect of green self-transitions placed on level locations and guarded by condition $up_j \wedge down_j$.

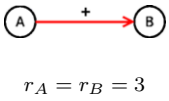
Since a level change in any process may affect the current regulators configuration, a particular clock manipulation procedure is invoked in both clock strategies immediately after the respective update of delays.

3.3 Example

In Fig. 4, a simple gene interaction model is presented. Table in the middle gives values for degradation delays of p_B and production delays guarded by the activation e_{AB}. Minimal and maximal delays are assumed to coincide. Actual delay δ_B is computed from the equation (according to Formula 2):

$$\frac{1}{\delta_Y} = \frac{1}{\delta_{\beta_{1,XY}}}\Theta(l_X \geq 1) + \frac{1}{\delta_{\beta_{2,XY}}}\Theta(l_X \geq 2) - \frac{1}{\delta_{\alpha_{1,Y}}}\Theta(l_Y \geq 1) - \frac{1}{\delta_{\alpha_{2,Y}}}\Theta(l_Y \geq 2)$$

Table on the right shows δ_Y evaluated for each system configuration.



k	α_{kB}	β_{kAB}
0	/	/
1	-15	13
2	-6	4

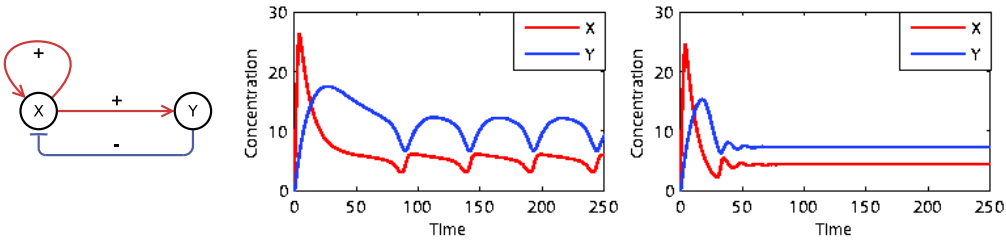
$p_A \backslash p_B$	0	1	2
0	$\pm\infty$	-15	-6
1	13	97.5	-6.39
2	4	3.84	10.68

Fig. 4. (left) Delays for the example model. (right) Delay δ_B computed for particular configurations of p_B levels (columns) and p_A levels (rows).

4 Evaluation and Case Study

To automatize the process of translating models into UPPAAL language, we have developed a tool *model-builder* [11]. The model-builder takes as input an XML file describing the GRN together with the settings of the timed model. Output of model-builder is a UPPAAL timed automaton representing the GRN timed model. In addition to this, the tool can parse the traces returned by UPPAAL simulation/model-checking in order to visualize the respective trajectories.

In order to evaluate expressiveness of our timed model, we have conducted simulations of a two-component regulatory motif [13] producing a nontrivial oscillatory behaviour. The motif, depicted in Fig. 5, has the advantage of being very sensitive to perturbations in rate coefficients, therefore, it has been chosen as a challenging model to test our methodology.



With X self-activation. Without X self-activation.

Fig. 5. Two-component oscillatory motif and its continuous simulations.

In the continuous framework, the following ODEs describe this model:

$$\frac{d[p_X]}{dt} = \underbrace{\frac{0.6 \cdot [p_X]^{10}}{4.6^{10} + [p_X]^{10}}}_{\text{self-activation}} + \underbrace{\frac{8}{1 + \frac{[p_Y]^{10}}{5.5^{10}}}}_{\text{inhibition by Y}} - \underbrace{0.1 \cdot [p_X]}_{\text{degradation}} \qquad \frac{d[p_Y]}{dt} = \underbrace{\frac{2 \cdot [p_X]^4}{5^4 + [p_X]^4}}_{\text{activation by X}} - \underbrace{0.1 \cdot [p_Y]}_{\text{degradation}}$$

	Production rates			Degradation rates		Hill constants		
	β_{XX}	β_{YX}	β_{XY}	α_X	α_Y	K_{XX}	K_{YX}	K_{XY}
Original model	0.6	8	2	0.1	0.1	4.6	5.5	5
Normalized model	0.023	0.308	0.077	0.1	0.1	0.178	0.21	0.192
Scaled model	$5.75 \cdot 10^{-4}$	$0.77 \cdot 10^{-3}$	$1.9 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$	0.178	0.21	0.192

Table 2
Parameter values of the original continuous model and its rescaled and normalized version used as a base for the timed model.

Table 2 (first row) shows the original parameters. Respective simulated trajectories are given in Fig. 5 (presented in two variants – with and without the X self-activation). In order to prepare the original continuous model for discrete approximation by a timed automaton, we first normalize (second row) the concentration domain by dividing the values by the maximal concentration value reached ($26M \cdot s^{-1}$). Values of degradation parameters remain unchanged, because they are independent of molar volume. Since we implement our timed model with on-the-fly delay computation in UPPAAL where time is approximated in integer domain, we upscale the time domain by a certain integer factor which will reduce the imprecision caused by integer rounding of rational rates when evaluating the equation (2). For our particular example we choose the factor 40 which implies that a single time unit ($1s$) in the original model is rescaled to $40s$ thus making the dynamics 40 times slower. This implies that rate parameters are divided by 40 (third row in Table 2).

After uniformly discretizing the normalized concentration domains of both proteins to 11 discrete levels, we proceed to transforming rate constants to delays. We consider the rate parameters of the normalized model and try to identify for each particular discrete level of p_X, p_Y the delays for the next level increase/decrease. Table 3(left) summarizes delays for our model. Note that delays are always specified to mimic the normalized rates. Upscaling of time is done implicitly by model-builder and is not included in the timed model specification.

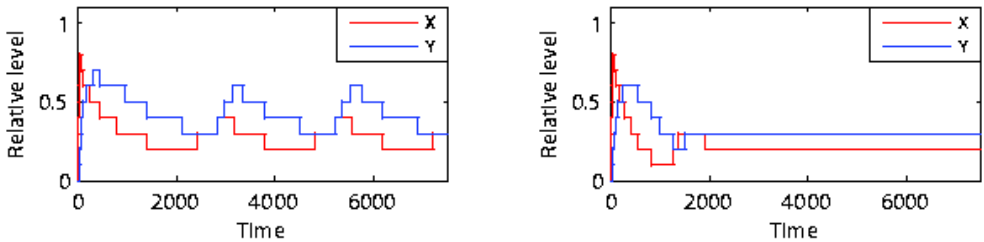
k	$\delta_{\beta_{k,YX}}$	$\delta_{\beta_{k,XX}}$	$\delta_{\alpha_{k,X}}$ & $\delta_{\alpha_{kY}}$	$\delta_{\beta_{kXY}}$
0	5	/	/	/
1	10	300	500	350
2	75	200	250	150
3	250	200	150	125
4	∞	200	125	63
5	∞	200	100	50
6	∞	200	75	38
7	∞	200	38	25
8	∞	200	25	14
9	∞	200	19	11
10	∞	200	10	9

k	$\delta_{\beta_{k,YX}}$	$\delta_{\beta_{k,XX}}$	$\delta_{\alpha_{k,X}}$	$\delta_{\alpha_{k,Y}}$	$\delta_{\beta_{k,XY}}$
0	5	/	/	/	/
1	14	20	14	20	22
2	∞	15	10	9	5

Table 3
Parameters for the oscillatory pattern with 11 levels (left) and 3 levels (right).

After running model-builder and UPPAAL we have obtained simulations de-

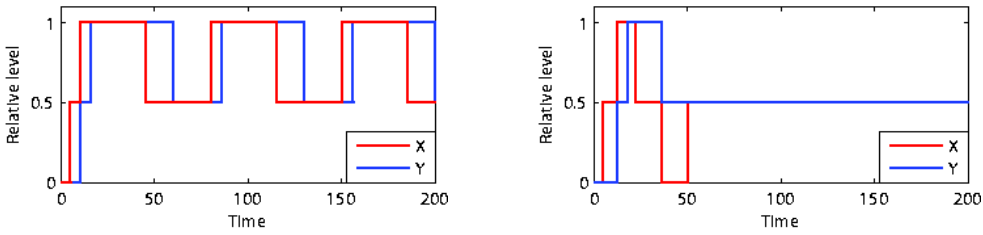
picted in Fig. 6. The results qualitatively reflect simulations achieved with the continuous model as can be seen by comparing with Fig. 5. We have also considered a variant discretizing the concentration domain by 3 levels. The timed model is given by delays specified in Table 3(right). Simulation results are depicted in Fig. 7. When comparing the 3-level model with the 11-level model, it can be seen that increasing granularity of discretization naturally leads to a more precise reproduction of the original continuous model behaviour. However, when comparing the two tables in Table 3, the 3-level model requires significantly less parameters (delays) to be specified.



With self-activation of X Without self-activation of X

Fig. 6. Simulation results using our model builder and UPPAAL. Non-stopwatch strategy has been employed. The time axis is upscaled by factor 40.

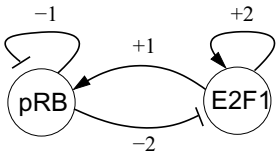
Please note that a comprehensive set of further case studies as well as performance benchmarks results is available in [11]. As regards the performance of UPPAAL reachability analysis for both model variants, the analysis of the 3-level model took $0.14s$ with 2491 states visited. For the 11-level model, the computation time was $0.73s$ with 6059 states visited. Results were performed on a computer based on Intel T7200 (dual core) 2GHz CPU with 2GB RAM.



With the self-activation of X Without the self-activation of X

Fig. 7. Simulation of the oscillatory pattern. Each concentration level l_j is expressed as relative concentration level $l'_j = l_j/r_j$.

To show a biological case study, we have investigated a model of a central GRN module governing the G_1/S cell cycle transition in mammalian cells [19]. The model considers a two-gene network describing interaction of the tumor suppressor protein pRB and the central transcription factor $E2F1$ (see Fig. 8 (left)). The model demonstrates the feature of bistability, i.e., the occurrence of two stable states characterized by different concentration of $E2F1$. The first stable state represents the low level of the $E2F1$ protein concentration. In this state, the cell stays in G_1 phase. When increased enough, the concentration of $E2F1$ grows higher, to the level of the second stable state, which causes the cell approaches to S phase.



$$t_{pRB,pRB} = t_{E2F1,pRB} \sim 0.5$$
$$t_{pRB,E2F1} \sim 5,\ t_{E2F1,E2F1} \sim 4$$

$$\frac{d[pRB]}{dt} = \beta_{pRB_1}\theta([pRB] < 0.5 \wedge [E2F1] > 0.5) + \beta_{pRB_2}\theta([pRB] < 5 \wedge [E2F1] > 0.5) - \alpha_{pRB}[pRB]$$
$$\frac{d[E2F1]}{dt} = \beta_{E2F1_1}\theta([pRB] < 5 \wedge [E2F1] > 0.5) + \beta_{pRB_2}\theta([pRB] < 5 \wedge [E2F1] > 4) - \alpha_{E2F1}[E2F1]$$

$$\beta_{pRB_1} = 0.9,\ \beta_{pRB_2} = 0.1,\ \alpha_{pRB} = 0.05$$
$$\beta_{E2F1_1} = 0.1,\ \beta_{E2F1_2} = 1.5,\ \alpha_{E2F1} = 0.1$$

Fig. 8. Genetic regulatory network controlling the G_1/S transition and the respective PADE model used for constructing the timed model.

k	$\delta_{\alpha_{k,pRB}}$ low mode	$\delta_{\alpha_{k,pRB}}$ bist. mode	$\delta_{\alpha_{k,E2F1}}$
1	4000	40	200
2	400	7	25

Model	φ_1	φ_2	φ_3
low mode	tt	ff	tt
bist. mode	tt	tt	ff

Table 4
(left) Degradation delays set for the G_1/S timed model. (right) Model checking results.

We reformulated the model in the timed automata framework and we employed UPPAAL model checking algorithm to identify bistability. By abstracting the original continuous model [19], we obtained PADE model depicted in Fig. 8. Domains of both proteins were naturally discretized into 3 levels by the thresholds. The considered rate coefficients as shown in [19] impose bistability of $E2F1$ (converging either to concentration $E2F1 = 8$ or $E2F1 = 0$ with the bistable switch at $E2F1 = 1.5$). This bistable switch is sensitive to the degradation coefficient α_{pRB}.

We built the timed model for the considered GRN by directly transforming the rates to delays as defined in Section 3. We set the respective delays to reciprocal values of rates ($\delta_{\beta_{pRB_1}} = 11$, $\delta_{E2F1_2} = 6$, $\delta_{\beta_{pRB_2}} = \delta_{\beta_{E2F1_1}} = 100$). All delay values are multiplied by 10 to increase precision of their integer representation in UPPAAL. We considered two sets of pRB degradation delays reflecting the bistable/non-bistable situation.

To detect bistability, we have formulated the following properties:

- $\varphi_1 = E2F1 \leq 1 \implies \mathbf{AG}(E2F1 \leq 1)$ stating that when $E2F1$ is initially lower than $t_{E2F1,pRB}$ then it stabilizes below $t_{E2F1,pRB}$.
- $\varphi_2 = E2F1 \geq 1 \implies \mathbf{AG}(E2F1 \geq 1)$ stating that when $E2F1$ is initially higher than $t_{E2F1,pRB}$ then it stabilizes above $t_{E2F1,pRB}$.
- $\varphi_3 = E2F1 \geq 1 \implies \mathbf{AF}(E2F1 \leq 1)$ stating that when $E2F1$ is initially higher than $t_{E2F1,pRB}$ then it always eventually reaches the level below $t_{E2F1,pRB}$.

Properties have been set to be expressible in UPPAAL temporal logic. Since UPPAAL does not support nesting of temporal operators, the initial concentration proposition has been represented by appropriate setting of initial states. Model checking of all the properties has been performed using the reachability analysis bounded by $5{\cdot}10^8$ time units. This is an acceptable restriction since the longest delay considered in the model is $4 \cdot 10^3$ time units. Results are presented in Table 4(right).

The results confirm the presence of bistability and its neglection by decreasing the pRB degradation rate.

5 Conclusions

We presented a new approach to incorporate time into discrete models of regulatory networks. Employing our prototype tool chain, we transformed the timed model into UPPAAL language. In consequence, we have successfully evaluated the model on simulation and model checking experiments.

Evaluation showed that the more activity levels are distinguished, the more precise are the results. However, increasing resolution of discrete levels arises in technical difficulty of delineating proper values for time delays. The results were achieved on automata without stopwatches, the entailed approximation did not qualitatively affect the expected (approximate) results.

For future work we plan to develop a method automatizing setting of delay values wrt rates in PADE model. This can be achieved easily for exponential decays in the degradation term. However, for delays of simple and even cooperated regulations the task is non-trivial and remains to be solved. An interesting question is to employ the inverse problems method [1] to determine constraints on delays compatible wrt time-series measurements on micro-arrays. On the theoretical side, we plan to study rigorously the relation of timed automata to the approximated ODE model. The abstraction has been only informally defined in this paper.

References

[1] Andréa, E., T. Chatain, L. Fribourg and E. Encrenaz, *An inverse method for parametric timed automata*, Electron. Notes Theor. Comput. Sci. **223** (2008), pp. 29–46.

[2] Batt, G., C. Belta and R. Weiss, *Model checking liveness properties of genetic regulatory networks*, in: *TACAS 2007*, LNCS **4424** (2007), pp. 323–338.

[3] Batt, G., R. B. Salah and O. Maler, *On timed models of gene networks*, in: *Formal Modeling and Analysis of Timed Systems*, LNCS **4763** (2007), pp. 38–52.

[4] Behrmann, G., A. David, K. G. Larsen, O. Müller, P. Pettersson and W. Yi, *Uppaal - Present and Future*, in: *Proc. of 40th IEEE Conference on Decision and Control*, IEEE **3** (2001), pp. 2881–2886.

[5] Bernot, G., J.-P. Comet, A. Richard and J. Guespin, *Application of formal methods to biological regulatory networks: Extending thomas' asynchronous logical approach with temporal logic*, Journal of Theoretical Biology **229** (2004), pp. 339–347.

[6] Bos, W., *Interactive signaling network analysis tool* (2009), Master's thesis, University of Twente. URL http://essay.utwente.nl/59184/

[7] Bozga, M., S. Graf, I. Ober, I. Ober and J. Sifakis, *The if toolset*, in: M. Bernardo and F. Corradini, editors, *Formal Methods for the Design of Real-Time Systems*, Lecture Notes in Computer Science **3185**, Springer Berlin / Heidelberg, 2004 pp. 131–132.

[8] Chaouiya et al., C., *Qualitative analysis of regulatory graphs: A computational tool based on a discrete formal framework*, in: *Positive Systems*, Lecture Notes in Control and Information Sciences **294**, Springer Berlin, 2003 pp. 830–832.

[9] Dang, T., C. L. Guernic and O. Maler, *Computing reachable states for nonlinear biological models*, in: *CMSB'09*, LNCS **5688** (2009), pp. 126–141.

[10] de Jong, H., J. Geiselmann, C. Hernandez and M. Page, *Genetic network analyzer: qualitative simulation of genetic regulatory networks*, Bioinformatics **19** (2003), pp. 336–344.

[11] Goethem, S. V., *Modelling gene regulatory networks with timed automata* (2011), Master's thesis, University of Namur.
URL http://anna.fi.muni.cz/~xsafran1/Work/cs2bio/

[12] Grosu, R., G. Batt, F. Fenton, J. Glimm, C. L. Guernic, S. Smolka and E. Bartocci, *From cardiac cells to genetic regulatory networks*, in: *Proceedings of the 23rd international conference on Computer aided verification*, CAV'11 (2011), pp. 396–411.

[13] Guantes, R. and J. Poyatos, *Dynamical principles of two-component genetic oscillators*, PLoS Comput Biol **2** (2006).

[14] Halász, A., V. Kumar, M. Imielinski, C. Belta, O. Sokolsky, S. Pathak and H. Rubin, *Analysis of lactose metabolism in e.coli using reachability analysis of hybrid systems*, Systems Biology, IET **1** (2007), pp. 130–148.

[15] Jacob, F. and J. Monod, *Genetic regulatory mechanisms in the synthesis of proteins*, Journal of Molecular Biology **3** (1961), pp. 318–356.

[16] Maler, O. and G. Batt, *Approximating continuous systems by timed automata*, in: *Proceedings of the 1st international workshop on Formal Methods in Systems Biology*, FMSB '08 (2008), pp. 77–89.

[17] Siebert, H. and A. Bockmayr, *Incorporating time delays into the logical analysis of gene regulatory networks*, in: *CMSB*, 2006, pp. 169–183.

[18] Snoussi, E. H., *Qualitative dynamics of piecewise-linear differential equations: A discrete mapping approach. dynamics and stability of systems*, Dynamics and Stability of Systems **4** (1989).

[19] Swat, M., A. Kel and H. Herzel, *Bifurcation Analysis of the Regulatory Modules of the Mammalian G1/S Transition*, Bioinformatics **20** (2004), pp. 1506–1511.

[20] Thieffry, D. and R. Thomas, *Dynamical Behavior of Biological Regulatory Networks*, Bulletin of Mathematical Biology **57** (1995), pp. 277–297.

[21] Thomas, R., *Boolean formalization of genetic control circuits.*, Journal of Theoretical Biology **42** (1973).