Model Checking Liveness Properties of Genetic Regulatory Networks

Grégory Batt[1], Calin Belta[1], and Ron Weiss[2]

[1] Center for Information and Systems Engineering and Center for BioDynamics, Boston University, Brookline, MA, USA
[2] Department of Electrical Engineering and Department of Molecular Biology, Princeton University, Princeton, NJ, USA
batt@bu.edu, cbelta@bu.edu, rweiss@princeton.edu

Abstract. Recent studies have demonstrated the possibility to build genetic regulatory networks that confer a desired behavior to a living organism. However, the design of these networks is difficult, notably because of uncertainties on parameter values. In previous work, we proposed an approach to analyze genetic regulatory networks with parameter uncertainties. In this approach, the models are based on piecewise-multiaffine (PMA) differential equations, the specifications are expressed in temporal logic, and uncertain parameters are given by intervals. Abstractions are used to obtain finite discrete representations of the dynamics of the system, amenable to model checking. However, the abstraction process creates spurious behaviors along which time does not progress, called time-converging behaviors. Consequently, the verification of liveness properties, expressing that something will eventually happen, and implicitly assuming progress of time, often fails. In this work, we extend our previous approach to enforce progress of time. More precisely, we define transient regions as subsets of the state space left in finite time by every solution trajectory, show how they can be used to rule out time-converging behaviors, and provide sufficient conditions for their identification in PMA systems. This approach is implemented in RoVerGeNe and applied to the analysis of a network built in the bacterium *E. coli*.

1 Introduction

The main goal of the nascent field of synthetic biology is to design and construct biological systems that present a desired behavior. The construction of networks of interregulating genes, so-called genetic regulatory networks, has demonstrated the feasibility of this approach [1]. However, most of the newly-created networks are non-functioning and need subsequent tuning [1]. One important reason is that large uncertainties on parameter values hamper the design of the networks. These uncertainties are caused by current limitations of experimental techniques but also by the fact that parameter values themselves vary with the ever-fluctuating extra- and intracellular environmental conditions.

In previous work [2, 3], we have developed a method for the verification of dynamical properties of genetic regulatory networks with *parameter uncertainty*. In this approach, models are based on piecewise-multiaffine (PMA) differential

equations, dynamical properties are specified in temporal logic, and uncertain parameters are given by intervals. Following an approach widely-used in hybrid systems theory [4], we use a time-abstracting embedding transition system in combination with discrete abstractions to obtain finite discrete representations of the dynamics of the system in state and parameter spaces, amenable to algorithmic verification by model checking [5].

In the context of gene network design, *liveness* properties, expressing that something will *eventually* happen [6], are commonly-encountered. When proving liveness properties of a dynamical system, the implicit requirement that along every behavior time progresses without upper bound plays a key role [7, 8]. However, spurious, time-converging behaviors created by the abstraction process often cause the verification on the abstract system of liveness properties to fail. This problem was early recognized but is still largely unsolved for continuous and hybrid systems [7, 9].

In this work, we address the above problem by enforcing progress of time in the abstract systems. First, we define transient regions as subsets of the state space that are left in finite time by every solution trajectory. Then, the simple observation that if the system remains in a transient region, the corresponding behavior is necessarily time-converging, provides us with a means to rule out time-converging behaviors in abstract systems. Finally, we propose sufficient conditions for the identification of transient regions of PMA systems. This approach has been implemented in a tool for Robust Verification of Gene Networks (RoVerGeNe), and applied to the verification of a non-trivial liveness property of a transcriptional cascade built in the bacterium *E. coli*. This case study demonstrates the practical applicability of the proposed approach.

The remainder of this paper is organized as follows. In Section 3, the biological problem is illustrated by means of an example: the analysis of the robustness of a transcriptional cascade. In Section 4, we present PMA models and briefly review the approach described in [2, 3] for their analysis under parameter uncertainty. Our contribution to the verification of liveness properties is detailed in Section 5 and 6. More precisely, we show in Section 5 how transient regions can be used to rule out time-converging behaviors in discrete abstractions, and in Section 6 how transient regions can be computed for uncertain PMA systems. In Section 7, we apply the proposed approach to the analysis of the transcriptional cascade. The final section discusses the proposed approach in the context of related work.

2 Preliminaries

All the notions and notations presented here are described at length in [2]. We consider Kripke structures, also called transition systems, $T = (S, \rightarrow, \Pi, \models)$, where S is a finite or infinite set of states, $\rightarrow \subseteq S \times S$ is a total transition relation, Π is a finite set of propositions, and $\models \subseteq S \times \Pi$ is a satisfaction relation [5]. An *execution* of T is an infinite sequence $e = (s_0, s_1, \ldots)$ such that for every $i \geq 0$, $s_i \in S$ and $(s_i, s_{i+1}) \in \rightarrow$. An equivalence relation $\sim \subseteq S \times S$ is proposition- preserving if $\forall s, s' \in S$ and $\pi \in \Pi$, if $s \sim s'$ and $s \models \pi$, then $s' \models \pi$. The *quotient transition system* of $T = (S, \rightarrow, \Pi, \models)$ given a proposition-preserving equivalence relation $\sim \subseteq S \times S$ is $T/\sim = (S/\sim, \rightarrow_\sim, \Pi, \models_\sim)$, where

$S/_\sim$ is the set of all equivalence classes R of S, $\rightarrow_\sim \subseteq S/_\sim \times S/_\sim$ is such that $R \rightarrow_\sim R'$ iff there exist $s \in R, s' \in R'$ such that $s \rightarrow s'$, and $\models_\sim \subseteq S/_\sim \times \Pi$ is such that $R \models_\sim \pi$ iff there exists $s \in R$ such that $s \models \pi$. The strongly connected components of a transition system $T = (S, \rightarrow, \Pi, \models)$ are the maximal strongly connected subgraphs of the graph $(S, \rightarrow)$. We refer to [5] for the syntax and semantics of LTL formulas interpreted over executions. A transition system T satisfies an LTL formula ϕ, denoted $T \models \phi$, iff every execution of T satisfies ϕ.

Let $S \subseteq \mathbb{R}^n$. $\overline{S}$ denotes its closure in $\mathbb{R}^n$, and hull(S), its convex hull. A polytope P in $\mathbb{R}^n$ is a bounded intersection of a finite number of open or closed halfspaces. P is hyperrectangular if $P = P_1 \times \ldots \times P_n$ where $P_i = \{x_i \in \mathbb{R} \mid x = (x_1, \ldots, x_n) \in P\}$, $i \in \{1, \ldots, n\}$. The set of points $v_1, \ldots, v_p \in \mathbb{R}^n$ satisfying $\overline{P} = \text{hull}(\{v_1, \ldots, v_p\})$ and $v_i \notin \text{hull}(\{v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_p\})$, $i \in \{1, \ldots, p\}$, is the set $\mathcal{V}_P$ of vertices of P. A facet of a full-dimensional polytope P is the intersection of P with one of its supporting hyperplanes. An affine function $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a polynomial of degree at most 1. A multiaffine function $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a polynomial in which the degree of f in any of its variables is at most 1. Stated differently, non-linearities are restricted to product of distinct variables.

Theorem 1 *[10] Let $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be an affine function and P be a polytope in $\mathbb{R}^n$. Then, $f(\overline{P}) = \text{hull}(\{f(v) \mid v \in \mathcal{V}_P\})$.*

Theorem 2 *[11] Let $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a multiaffine function and P be a hyperrectangular polytope in $\mathbb{R}^n$. Then, $f(P) \subseteq \text{hull}(\{f(v) \mid v \in \mathcal{V}_P\})$.*

3 A motivating example: tuning a transcriptional cascade

We consider the genetic regulatory network built in *E. coli* [12] and represented in Figure 1(a). It consists of 4 genes forming a cascade of transcriptional inhibitions. The network is controlled by the addition or removal of aTc that serves as controllable input. The output is the fluorescence intensity of the system, due to the fluorescent protein EYFP. The cascade is ultrasensitive: at steady-state, the output undergoes a dramatic change for a moderate change of the input in a narrow interval. The cascade is expected to present at least a 1000-fold increase of the output value for a two-fold increase of the input value, but the actual network does not meet its specifications (Figure 1(b)).

In [2, 3], we investigated the possibility to tune the network by modifying some of its parameters. To do so, we built a model of the system (Figure 2(a)), identified parameter values using experimental data available in [12], specified the expected behavior in LTL (Figure 2(b)), and searched for and found parameter values for which the system satisfies its specifications (Figure 1(b)). It is important that the network presents a robust behavior, since it should behave as expected despite environmental fluctuations. So, before actually experimentally tuning the network as suggested, we would like to use our model to evaluate the robustness of the tuned system. More specifically, we would like to verify that the tuned cascade satisfies its specification for all production and degradation rate parameters varying in $\pm 10\%$ intervals centered at their reference values.

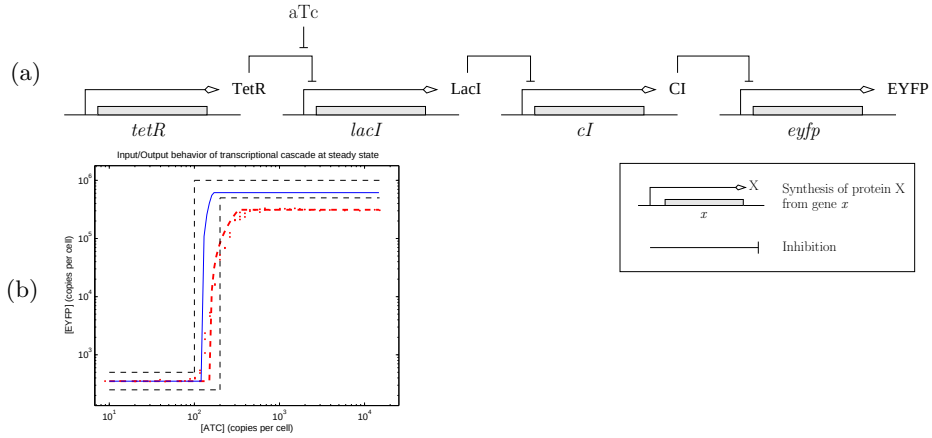


Fig. 1. (a) Synthetic transcriptional cascade. The genes *tetR*, *lacI*, *cI*, and *eyfp* code for the proteins TetR, LacI, CI, and EYFP, respectively. When a gene is expressed, the corresponding protein is produced, which inhibits the expression of a gene downstream. The input molecule, aTc, relieves the inhibition of *lacI* by TetR. (b) Steady-state I/O behavior of the cascade: measured (red dots), expected (region delimited by black dashed lines), and predicted, before (red dashed line) and after (blue solid line) tuning.

4 Model checking genetic regulatory networks with parameter uncertainty

4.1 PMA models and LTL specifications

We first present a formalism for modeling gene networks. The notations and terminology are adapted from [13]. We consider a network consisting of n genes. The state of the network is given by the vector $x = (x_1, \ldots, x_n)$, where x_i is the concentration of the protein encoded by gene i. The state space $\mathcal{X}$ is a hyperrectangular subset of $\mathbb{R}^n$: $\mathcal{X} = \prod_{i=1}^{n}[0, max_{x_i}]$, where max_{x_i} denotes a maximal concentration of the protein encoded by gene i. Some parameters may be *uncertain*: $p = (p_1, \ldots, p_m)$ is the vector of uncertain parameters, with values in the parameter space $\mathcal{P} = \prod_{j=1}^{m}[min_{p_j}, max_{p_j}]$, where min_{p_j} and max_{p_j} denotes a minimal and maximal value for p_j.

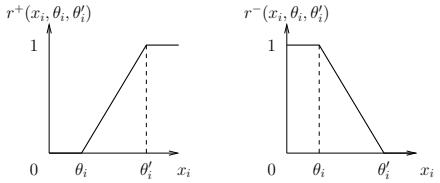
The dynamics of the network is given by a set of differential equations:

$$\dot{x}_i = f_i(x,p) = \sum\nolimits_{j \in P_i} \kappa_i^j \, r_i^j(x) - \sum\nolimits_{j \in D_i} \gamma_i^j \, r_i^j(x) \, x_i, \quad i \in \{1, \ldots, n\}, \qquad (1)$$

where P_i and D_i are sets of indices, $\kappa_i^j > 0$ and $\gamma_i^j > 0$ are (*possibly uncertain*) *production* and *degradation rate parameters*, and $r_i^j : \mathcal{X} \to [0,1]$ are continuous, piecewise-multiaffine (PMA) functions, called *regulation functions*. As seen in our example, PMA functions arise from products of ramp functions r^+ and r^- used for representing complex gene regulations or protein degradations (see Figure 2(a) Eq. (2') and Ref. [2]). The components of p are production or degradation rate parameters. With the additional assumption that r_i^j does not depend

(a)
$$\dot{x}_{tetR} = \kappa_{tetR} - \gamma_{tetR}\, x_{tetR}, \quad (1')$$
$$\dot{x}_{lacI} = \kappa^0_{lacI} + \kappa_{lacI}(1 - r^+(x_{tetR}, \theta^1_{tetR}, \theta^2_{tetR})\, r^-(u_{aTc}, \theta^1_{aTc}, \theta^2_{aTc})) - \gamma_{lacI}\, x_{lacI}, \quad (2')$$
$$\dot{x}_{cI} = \kappa^0_{cI} + \kappa_{cI}\, r^-(x_{lacI}, \theta^1_{lacI}, \theta^2_{lacI}) - \gamma_{cI}\, x_{cI}, \quad (3')$$
$$\dot{x}_{eyfp} = \kappa^0_{eyfp} + \kappa_{eyfp}\, r^-(x_{cI}, \theta^1_{cI}, \theta^2_{cI}) - \gamma_{eyfp}\, x_{eyfp}, \quad (4')$$
$(\theta^1_{aTc}, \theta^2_{aTc}) = (80, 4000)$; $(\kappa_{tetR}, \gamma_{tetR}, \theta^1_{tetR}, \theta^2_{tetR}) = (260, 0.013, 4500, 5500)$; $(\kappa^0_{lacI}, \kappa_{lacI}, \gamma_{lacI}, \theta^1_{lacI}, \theta^2_{lacI}) = (2.4, 875.6, 0.013, 500, 4500)$; $(\kappa^0_{cI}, \kappa_{cI}, \gamma_{cI}, \theta^1_{cI}, \theta^2_{cI}) = (3.9, 386, 0.013, 600, 23000)$; $(\kappa^0_{eyfp}, \kappa_{eyfp}, \gamma_{eyfp}) = (4.58, 4048, 0.013)$

(b)



(c)
$$\begin{aligned} \phi_1 = \quad & u_{aTc} < 100 \rightarrow \mathrm{FG}(x_{eyfp} > 2.5\,10^2 \wedge x_{eyfp} < 5\,10^2) \\ \wedge \quad & 100 < u_{aTc} < 200 \rightarrow \mathrm{FG}(x_{eyfp} > 2.5\,10^2 \wedge x_{eyfp} < 10^6) \\ \wedge \quad & u_{aTc} > 200 \rightarrow \mathrm{FG}(x_{eyfp} > 5\,10^5 \wedge x_{eyfp} < 10^6). \end{aligned}$$

Fig. 2. (a) Model of the cascade. x_{tetR}, x_{lacI}, x_{cI}, x_{eyfp} denote protein concentrations, u_{aTc}, input molecule concentration, θ's, threshold parameters, κ's, production rate parameters, and γ's, degradation rate parameters. r^+ and r^- are ramp functions represented in (b). The product of ramp functions in Eq. (2') captures the assumption that the expression of *lacI* is repressed when TetR is present and aTc absent, and causes the model to be piecewise-multiaffine. Parameter values are indicated. Tuned parameters are: $\kappa_{lacI} = 2591$, $\kappa_c I = 550$, and $\kappa_{eyfp} = 8000$. (b) Increasing (r^+) and decreasing (r^-) ramp functions. (c) LTL specification of the expected behavior of the cascade represented in Figure 1(b). FGp ("eventually, p will be always true") is used to express that the property p holds at steady state. ϕ_1 is a liveness property.

on x_i for $j \in D_i$,[3] it holds that $f = (f_1, \ldots, f_n) : \mathcal{X} \times \mathcal{P} \rightarrow \mathbb{R}^n$ is a (non-smooth) *continuous* function of x and p, a *piecewise-multiaffine* function of x, and an *affine* function of p. Note that production and degradation rate parameters may be uncertain, but regulation functions (with their threshold parameters) must be known precisely. Finally, Equation (1) is easily extended to account for constant inputs u by considering u as a new variable satisfying $\dot{u} = 0$.

A number of dynamical properties of gene networks can be specified in temporal logic by LTL formulas over atomic propositions of type $x_i < \lambda$ or $x_i > \lambda$, where $\lambda \in \mathbb{R}_{\geq 0}$ is a constant. We denote by Π the set of all such atomic propositions. A *PMA system* Σ is then defined by a piecewise-multiaffine function f defined as above and a set of atomic propositions Π: $\Sigma = (f, \Pi)$.

PMA models of gene networks were proposed in [14] (see [15] for a related, piecewise-continuous formalism). The models considered here are also related to the piecewise-affine (PA) models proposed in [16] (see also [13]). However, contrary to the step functions used in PA models, ramp functions capture the graded response of gene expression to continuous changes in effector concentrations.

[3] This assumption requires that a protein does not regulate its own degradation. In practice, this assumption is generally satisfied.

4.2 Embedding transition systems and discrete abstractions

The specific form of the PMA function f suggests a division of the state space $\mathcal{X}$ into hyperrectangular regions (see Figure 3 for our example network). Let $\Lambda_i = \{\lambda_i^j\}_{j\in\{1,\ldots,l_i\}}$ be the ordered set of all threshold constants in f, and of all atomic proposition constants in Π, associated with gene i, together with 0 and max_{x_i}, $i \in \{1,\ldots,n\}$. The cardinality of Λ_i is l_i. Then, we define $\mathcal{R}$ as the following set of n-dimensional hyperrectangular polytopes $R \subseteq \mathcal{X}$, called *rectangles*:

$$\mathcal{R} = \{R_c \mid c = (c_1,\ldots,c_n) \text{ and } \forall i \in \{1,\ldots,n\} : c_i \in \{1,\ldots,l_i - 1\}\},$$

where

$$R_c = \{x \in \mathcal{X} \mid \forall i \in \{1,\ldots,n\} : \lambda_i^{c_i} < x_i < \lambda_i^{c_i+1}\}.$$

The union of all rectangles in $\mathcal{X}$ is denoted by $\mathcal{X}_\mathcal{R}$: $\mathcal{X}_\mathcal{R} = \cup_{R\in\mathcal{R}} R$. Note that $\mathcal{X}_\mathcal{R} \neq \mathcal{X}$. Notably, threshold hyperplanes are not included in $\mathcal{X}_\mathcal{R}$. $rect : \mathcal{X}_\mathcal{R} \to \mathcal{R}$ maps every point x in $\mathcal{X}_\mathcal{R}$ to the rectangle R such that $x \in R$. Two rectangles R and R', are said *adjacent*, denoted $R \fallingdotseq R'$, if they share a facet. Figure 3(a) shows 9 rectangles in a 2-D slice of the state space of our example network. R^1 and R^2 are adjacent (*i.e.* $R^1 \fallingdotseq R^2$), whereas R^1 and R^5 are not.

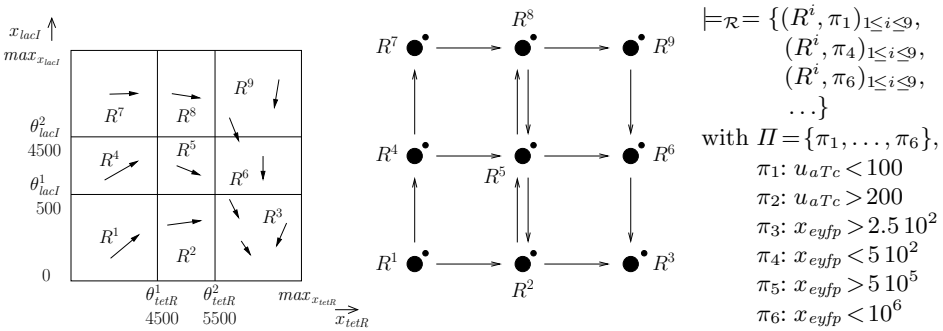


Fig. 3. Transcriptional cascade. (a) Schematic representation of the flow (arrows) in a 2-D slice of the state space. Other variables satisfy: $0 < u_{aTc} < 100$, $0 < x_{cI} < 600$, and $0 < x_{eyfp} < 250$. (b) and (c) Discrete abstraction $T_\mathcal{R}(p)$: subgraph of $(\mathcal{R}, \to_{\mathcal{R},p})$ corresponding to the region represented in (a), and satisfaction relation $\models_\mathcal{R}$. Dots denote self transitions.

Formally, we define the semantics of a PMA system Σ by means of a time-abstracting embedding transition system [4, 8].

Definition 1 *Let $p \in \mathcal{P}$. The embedding transition system associated with the PMA system $\Sigma = (f, \Pi)$ is $T_\mathcal{X}(p) = (\mathcal{X}_\mathcal{R}, \to_{\mathcal{X},p}, \Pi, \models_\mathcal{X})$ defined such that:*

- *$\to_{\mathcal{X},p} \subseteq \mathcal{X}_\mathcal{R} \times \mathcal{X}_\mathcal{R}$ is the transition relation defined by $(x, x') \in \to_{\mathcal{X},p}$ iff there exist a solution ξ of (1) and $\tau \in \mathbb{R}_{>0}$ such that $\xi(0) = x$, $\xi(\tau) = x'$, $\forall t \in [0,\tau]$, $\xi(t) \in \overline{rect}(x) \cup \overline{rect}(x')$, and either $rect(x) = rect(x')$ or $rect(x) \fallingdotseq rect(x')$,*

– $\models_{\mathcal{X}} \subseteq \mathcal{X}_{\mathcal{R}} \times \Pi$ *is the satisfaction relation defined by* $(x, \pi) \in \models_{\mathcal{X}}$ *iff* $x = (x_1, \ldots, x_n)$ *satisfies the proposition* π *(of type* $x_i < \lambda$ *or* $x_i > \lambda$*) with the usual semantics.*

In $T_{\mathcal{X}}(p)$, a transition between two points corresponds to an evolution of the system during some time. Quantitative aspects of time are abstracted away: some time elapses, but we don't know how much. Also note that not all solution trajectories of (1) are guaranteed to be represented by our embedding. However, one can show that our embedding describes *almost all* solution trajectories of (1), which is satisfying for all practical purposes [2].

A PMA system Σ satisfies an LTL formula ϕ for a given parameter $p \in \mathcal{P}$ if $T_{\mathcal{X}}(p) \models \phi$, that is, if every execution of $T_{\mathcal{X}}(p)$ satisfies ϕ.

We use discrete abstractions [4] to obtain finite transition systems preserving dynamical properties of $T_{\mathcal{X}}(p)$ and amenable to algorithmic verification [5]. Let $\sim_{\mathcal{R}} \subseteq \mathcal{X}_{\mathcal{R}} \times \mathcal{X}_{\mathcal{R}}$ be the (proposition-preserving) equivalence relation defined by the map *rect*: $x \sim_{\mathcal{R}} x'$ iff $rect(x) = rect(x')$. $\mathcal{R}$ is the set of equivalence classes. Then, the discrete abstraction of $T_{\mathcal{X}}(p)$ is the *quotient* of $T_{\mathcal{X}}(p)$ given $\sim_{\mathcal{R}}$.

Definition 2 *Let* $p \in \mathcal{P}$*. The discrete abstraction of* $T_{\mathcal{X}}(p)$ *is the quotient of* $T_{\mathcal{X}}(p)$ *given* $\sim_{\mathcal{R}}$*, denoted by* $T_{\mathcal{R}}(p) = (\mathcal{R}, \rightarrow_{\mathcal{R},p}, \Pi, \models_{\mathcal{R}})$*.*

For the cascade, the discrete transition system $T_{\mathcal{R}}(p)$ is partially represented in Figure 3(b) and (c), with p denoting the tuned parameter values (Section 3). As suggested by the sketch of the flow in Figure 3(a), there exist solution trajectories reaching R^2 from R^1 without leaving $\overline{R}^1 \cup \overline{R}^2$. Consequently, there is a discrete transition from R^1 to R^2. Also, for example, rectangle R^1 satisfies the atomic proposition π_1: $u_{aTc} < 100$, that is, $(R^1, \pi_1) \in \models_{\mathcal{R}}$.

4.3 Model checking uncertain PMA systems

Because parameter values are often uncertain, we would like to be able to test whether a PMA system Σ satisfies an LTL formula ϕ for *every* parameter in a *set* $P \subseteq \mathcal{P}$. This problem is defined as robustness analysis in [2, 3]. Note that the problem given in Section 3 is precisely an instance of this problem.

To describe the behavior of a network for *sets* of parameters $P \subseteq \mathcal{P}$, we define the transition systems $T^{\exists}_{\mathcal{R}}(P)$ and $T^{\forall}_{\mathcal{R}}(P)$ as follows.

Definition 3 *Let* $P \subseteq \mathcal{P}$*. Then* $T^{\exists}_{\mathcal{R}}(P) = (\mathcal{R}, \rightarrow^{\exists}_{\mathcal{R},P}, \Pi, \models_{\mathcal{R}})$ *and* $T^{\forall}_{\mathcal{R}}(P) = (\mathcal{R}, \rightarrow^{\forall}_{\mathcal{R},P}, \Pi, \models_{\mathcal{R}})$*, where*
– $(R, R') \in \rightarrow^{\exists}_{\mathcal{R},P}$ *iff* $\exists p \in P$ *such that* $(R, R') \in \rightarrow_{\mathcal{R},p}$ *in* $T_{\mathcal{R}}(p)$*, and*
– $(R, R') \in \rightarrow^{\forall}_{\mathcal{R},P}$ *iff* $\forall p \in P$*,* $(R, R') \in \rightarrow_{\mathcal{R},p}$ *in* $T_{\mathcal{R}}(p)$*.*

In words, $T^{\exists}_{\mathcal{R}}(P)$ contains all the transitions present in at least one transition system $T_{\mathcal{R}}(p)$, and $T^{\forall}_{\mathcal{R}}(P)$ contains only the transitions present in all the transition systems $T_{\mathcal{R}}(p)$, $p \in P$. Informally, $T^{\exists}_{\mathcal{R}}(P)$ and $T^{\forall}_{\mathcal{R}}(P)$ can be considered as over- and under-approximations of $T_{\mathcal{R}}(p)$ when p varies, respectively.

In [2, 3], we have shown the following property.

$$\text{if } T^{\exists}_{\mathcal{R}}(P) \models \phi, \text{ then for every } p \in P,\ T_{\mathcal{X}}(p) \models \phi,$$

that is, the PMA system Σ satisfies property ϕ for every parameter in P. This property is instrumental for proving robust properties of gene networks. However, note that $T^{\exists}_{\mathcal{R}}(P) \not\models \phi$ does not imply that for some, nor for every parameter, the property is false. P might still contain parameters for which the property is true, called valid parameters. So we proposed an iterative procedure that partitions P and that tests whether $T^{\exists}_{\mathcal{R}}(P') \models \phi$ for each full-dimensional subset P'. Clearly, this approach becomes very inefficient when P does not contain valid parameters, since we keep on partitioning P. This situation can be detected by means of $T^{\forall}_{\mathcal{R}}(P)$. We have shown in [2, 3] that if $T^{\forall}_{\mathcal{R}}(P) \not\models \phi$, then we should stop partitioning P. So $T^{\exists}_{\mathcal{R}}(P)$ and $T^{\forall}_{\mathcal{R}}(P)$ are respectively used for proving robust properties of the system and for preserving the efficiency of the approach. Finally, we showed that $T^{\exists}_{\mathcal{R}}(P)$ and $T^{\forall}_{\mathcal{R}}(P)$ can be computed for polyhedral parameter sets using standard polyhedral operations. The robustness of a number of dynamical properties can be tested this way. However, because model checking results are almost always negative, this approach fails when applied to the verification of *liveness* properties. As we will see in the next section, this problem is due to the presence of spurious, time-converging executions in the abstract transition systems.

5 Transient regions and liveness checking

The analysis of counter-examples returned by model-checkers reveals why the verification of liveness properties generally fails. For example, the execution $e_{\mathcal{R}1} = (R^1, R^1, R^1, \ldots)$ of $T_{\mathcal{R}}(p)$ (Figure 3(b)), is a counter-example of the liveness property ϕ_1 given in Figure 2(c). However, from the sketch of the flow in Figure 3(a), it is intuitively clear that the system leaves R^1 in finite time. Consequently, the execution $e_{\mathcal{R}1}$ that describes a system remaining always in R^1 conflicts with the requirement that time progresses without upper bound. Such executions are called *time-converging* [7, 9][4]. Because they do not represent genuine behaviors of the system, these executions should be excluded when checking the properties of the system.

5.1 Time-diverging executions and transient regions

Definition 4 *Let $p \in \mathcal{P}$.*

An execution $e_{\mathcal{X}} = (x_0, x_1, \ldots)$ of $T_{\mathcal{X}}(p)$ is time-diverging iff there exists a solution ξ of (1) and a sequence of time instants $\tau = (\tau_0, \tau_1, \ldots)$ such that $\xi(\tau_i) = x_i$, for all $i \geq 0$, and $\lim_{i\to\infty} \tau_i = \infty$.

An execution $e_{\mathcal{R}} = (R_0, R_1, \ldots)$ of $T_{\mathcal{R}}(p)$ is time-diverging iff there exists a time-diverging execution $e_{\mathcal{X}} = (x_0, x_1, \ldots)$ of $T_{\mathcal{X}}(p)$ such that $x_i \in R_i$, for all $i \geq 0$.

[4] Time-converging executions are sometimes called Zeno executions [7, 9]. However, we prefer the former term since the latter is also used in a more restricted sense [17].

Intuitively, an execution of the embedding transition system $T_{\mathcal{X}}(p)$ is time-diverging if it represents at least one solution on the time interval $[0, \infty)$. Also, an execution of the discrete transition system $T_{\mathcal{R}}(p)$ is time-diverging if it is the abstraction of at least one time-diverging execution of $T_{\mathcal{X}}(p)$. Here, we identify two causes for the absence of progress in the abstract system $T_{\mathcal{R}}(p)$. The first one is due to the time-abstracting semantics used. The time-elapse corresponding to a transition in $T_{\mathcal{X}}(p)$ can be infinitesimal such that the sum of all time-elapses of the transitions of an execution of $T_{\mathcal{X}}(p)$ can be finite. The second one is due to the discrete abstraction, since the abstraction process introduces the possibility to iterate infinitely on discrete states of $T_{\mathcal{R}}(p)$. While the first problem appears only for dense-time systems, the second problem is also present in untimed systems and has been studied in the model checking community [18, 19]. Examples of time-converging executions of $T_{\mathcal{R}}(p)$ for our example network include $e_{\mathcal{R}1} = (R^1, R^1, R^1, \ldots)$, and $e_{\mathcal{R}2} = (R^2, R^5, R^2, R^5, R^2, \ldots)$ (Figure 3(a) and (b)).

The notion of time-diverging executions can be extended to $T_{\mathcal{R}}^{\exists}(P)$ and $T_{\mathcal{R}}^{\forall}(P)$ as follows.

Definition 5 *Let $P \subseteq \mathcal{P}$.*

An execution $e_{\mathcal{R}}$ of $T_{\mathcal{R}}^{\exists}(P)$ is time-diverging, if for some $p \in P$, $e_{\mathcal{R}}$ is an execution of $T_{\mathcal{R}}(p)$ and is time-diverging.

An execution $e_{\mathcal{R}}$ of $T_{\mathcal{R}}^{\forall}(P)$ is time-diverging, if for all $p \in P$, $e_{\mathcal{R}}$ is a time-diverging execution of $T_{\mathcal{R}}(p)$.

Finally, we define *transient regions* as subsets of the state space $\mathcal{X}$ that are left in finite time by every solution. For a reason that will become clear later, we focus on regions corresponding to unions of rectangles. As suggested by the sketch of the flow in Figure 3(a) and proved later, R^1 and $\cup_{j\in\{2,5,8\}} R^j$ are transient regions.

Definition 6 *Let $p \in \mathcal{P}$ and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. U is transient for parameter p if for every solution ξ of (1) such that $\xi(0) \in U$, there exists $\tau > 0$ such that $\xi(\tau) \notin \overline{U}$.*

5.2 Ruling out time-converging executions

From the maximality of strongly connected components (SCCs), it follows that an infinite execution of a finite transition system remains eventually always in a unique SCC. With $T_{\mathcal{R}}$ being either $T_{\mathcal{R}}(p)$, $T_{\mathcal{R}}^{\exists}(P)$, or $T_{\mathcal{R}}^{\forall}(P)$, and $e_{\mathcal{R}}$ being an execution of $T_{\mathcal{R}}$, we denote by $SCC(e_{\mathcal{R}}) \subseteq \mathcal{X}$ the union of the rectangles of the strongly connected component of $T_{\mathcal{R}}$ in which $e_{\mathcal{R}}$ remains eventually always. Then, it is clear that if an execution $e_{\mathcal{R}}$ of $T_{\mathcal{R}}(p)$ is time-diverging, that is, represents at least a solution trajectory on a time interval $[0, \infty)$ (Definition 4), then $SCC(e_{\mathcal{R}})$ can not be a transient region. Proposition 1 captures this intuition and establishes a link between time-diverging executions and transient regions.

Proposition 1 *Let $p \in \mathcal{P}$. If an execution $e_{\mathcal{R}}$ of $T_{\mathcal{R}}(p)$ is time-diverging, then $SCC(e_{\mathcal{R}})$ is not transient for p.*

Proof. Let $p \in \mathcal{P}$ and $e_{\mathcal{R}} = (R_0, R_1, \ldots)$ be a time-diverging execution of $T_{\mathcal{R}}(p)$. By definition of $SCC(e_{\mathcal{R}})$, there exists $i \geq 0$ such that for every $j \geq i$, $R_j \subseteq SCC(e_{\mathcal{R}})$. Let

$e'_{\mathcal{R}} = (R_i, R_{i+1}, \ldots)$ be a suffix of $e_{\mathcal{R}}$ and $U = \cup_{j \geq i} R_j \subseteq SCC(e_{\mathcal{R}})$. It holds that $e'_{\mathcal{R}}$ is a time-diverging execution of $T_{\mathcal{R}}(p)$. By Definition 4, there exists a time-diverging execution $e'_{\mathcal{X}} = (x_0, x_1, \ldots)$ of $T_{\mathcal{X}}(p)$ such that for all $j \geq 0$, $x_j \in R^{i+j} \subseteq U$. Then by Definition 4, this implies the existence of a solution ξ of (1) such that $\forall t \geq 0$, $\exists \tau \geq t$ such that $\xi(\tau) \in U$. Also, $\forall t \geq 0$, $\xi(t) \in \overline{U}$ because every rectangle visited by $\xi(t)$ is necessarily in U (Definitions 1 and 2). Consequently U is not transient for p (Definition 6). Because $U \subseteq SCC(e_{\mathcal{R}})$, the same necessarily holds for $SCC(e_{\mathcal{R}})$.

Consider again the executions $e_{\mathcal{R}1} = (R^1, R^1, R^1, \ldots)$ and $e_{\mathcal{R}2} = (R^2, R^5, R^2, R^5, R^2, \ldots)$ of $T_{\mathcal{R}}(p)$ (Figure 3(b)). Then, as mentioned earlier, $SCC(e_{\mathcal{R}1}) = R^1$ and $SCC(e_{\mathcal{R}2}) = \cup_{j \in \{2,5,8\}} R^j$ are transient regions for parameter p. By Proposition 1, $e_{\mathcal{R}1}$ and $e_{\mathcal{R}2}$ are consequently time-converging for p.

The following property is a generalization of Proposition 1.

Proposition 2 *Let $P \subseteq \mathcal{P}$.*

(a) If an execution $e_{\mathcal{R}}$ of $T^{\exists}_{\mathcal{R}}(P)$ is time-diverging, then for some $p \in P$, $SCC(e_{\mathcal{R}})$ is not transient for p.

(b) If an execution $e_{\mathcal{R}}$ of $T^{\forall}_{\mathcal{R}}(P)$ is time-diverging, then for all $p \in P$, $SCC(e_{\mathcal{R}})$ is not transient for p.

Proof. First note that we can not use directly Proposition 1, since by definition, $SCC(e_{\mathcal{R}})$ differs depending on whether $e_{\mathcal{R}}$ is an execution of $T^{\exists}_{\mathcal{R}}(P)$, $T^{\forall}_{\mathcal{R}}(P)$ or $T_{\mathcal{R}}(p)$, $p \in P$. However, with $e_{\mathcal{R}}$ an execution of $T^{\exists}_{\mathcal{R}}(P)$ (resp. of $T^{\forall}_{\mathcal{R}}(P)$), we can show exactly as in the proof of Proposition 1, the existence of a set U included in $SCC(e_{\mathcal{R}})$ and non-transient for some (resp. every) parameter $p \in P$. The conclusion follows immediately.

To summarize, let us denote by $T_{\mathcal{R}}$ either $T_{\mathcal{R}}(p)$, $T^{\exists}_{\mathcal{R}}(P)$ or $T^{\forall}_{\mathcal{R}}(P)$ and interpret "transient" as transient for p, for every $p \in P$ or for some $p \in P$, respectively. Then, using the contrapositive of Proposition 1 or 2, we obtain that given a strongly connected component of $T_{\mathcal{R}}$, if the corresponding region $U \subseteq \mathcal{X}$ is transient then every execution of $T_{\mathcal{R}}$ remaining in U (*i.e.* being eventually always in U) is time-converging and should not be taken into account when checking the properties of the system. Provided that transient regions can be identified, this suggests a method to rule out time-converging executions. To do so, we define a new atomic proposition $\pi =$ 'transient' in Π and label as 'transient' all and only rectangles R in transient SCCs. Then, instead of testing whether

$$T_{\mathcal{R}} \models \phi,$$

we test whether

$$T_{\mathcal{R}} \models \phi', \quad \text{with } \phi' = \neg \mathrm{FG}(\text{'transient'}) \rightarrow \phi.$$

The executions of $T_{\mathcal{R}}$ satisfying FG('transient') necessarily remain in a transient SCC, and are consequently time-converging (Proposition 1 or 2). So, only time-converging executions are ruled out this way. However, because Propositions 1 and 2 give only necessary conditions for an execution to be time-diverging, not all time-converging executions are guaranteed to be ruled out.

Consider again our example network. As said earlier, R^1 is a transient region. Because R^1 forms a (single-state) SCC, it is labeled 'transient' in $T_{\mathcal{R}}(p)$. Then, the execution $e_{\mathcal{R}1} = (R^1, R^1, R^1, \ldots)$, satisfying FG('transient'), is not a counter-example of ϕ'_1, and will not cause the property to be falsely invalidated anymore.

6 Transient region computation for PMA systems

The approach presented in the previous section is rather general in the sense that it solely requires the capacity to characterize transient regions. In this section, we provide sufficient conditions for their identification in PMA systems. More precisely, we provide conditions for proving that regions corresponding to SCCs in the discrete abstractions are transient for a given parameter (Proposition 3), for some parameter (Proposition 5), or for all parameters in a polyhedral set (Proposition 4). Using sufficient conditions, not all transient regions are guaranteed to be identified. However, only time-converging executions will be ruled out using the approach presented in Section 5. More precisely, Propositions 3, 4 and 5 are used in combination with (the contrapositive of) Propositions 1, 2(a) and 2(b), respectively. These properties rely on the fact that in a rectangle R the function f is multiaffine and hence is a convex combination of its value at the vertices of R (Theorem 2). Our focus on PMA systems is motivated by biological applications. However, Theorem 1 for affine functions on polytopes is similar to, and in fact stronger than Theorem 2 for multiaffine functions on rectangles, such that the results in this section also hold for similarly-defined continuous, piecewise-affine systems on polytopes.

Proposition 3 *Let $p \in \mathcal{P}$ and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. If*

$$0 \notin \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\}),$$

then U is transient for parameter p.

Proof. Let $p \in \mathcal{P}$ and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. Assume $0 \notin \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\})$. Using the separating hyperplane theorem, there exists $\alpha \in \mathbb{R}^n$ such that for all $z \in \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\})$, $\alpha^T z > 0$. For every rectangle $R \subseteq U$, $f(x,p)$ is a multiaffine function of x on $\overline{R}$, so it holds that for every $x \in \overline{R}$, $f(x,p) \in \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R\})$ (Theorem 2). Then, for every $x \in \overline{U}$, $f(x,p) \in \cup_{R \subseteq U} \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R\})$, which is included in $\text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\})$. Consequently $\alpha^T f(x,p) > 0$. Since $\overline{U}$ is compact (union of compact sets $\overline{R}$) and f is continuous, $\alpha^T f(\overline{U},p)$ is compact, which implies that there exists $c > 0$ such that the velocity in the direction of α^T is always larger than c. Consequently, $\overline{U}$ is left in finite time.

The conditions of the above property are satisfied by R^1 and $\cup_{j \in \{2,5,8\}} R^j$, which proves that these regions are transient, as hypothesized earlier. Propositions 4 and 5 are generalizations of Proposition 3 to polyhedral parameter sets.

Proposition 4 *Let $P \subseteq \mathcal{P}$ be a polytope and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. If $0 \notin \text{hull}(\{f(v,w) \mid v \in \mathcal{V}_R, R \subseteq U, w \in \mathcal{V}_P\})$, then U is transient for all parameters $p \in P$.*

Proof. Using Proposition 3 we only have to prove that if $0 \notin \text{hull}(\{f(v,w) \mid v \in \mathcal{V}_R, R \subseteq U, w \in \mathcal{V}_P\})$ then $\forall p \in P$, $0 \notin \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\})$. We prove its contrapositive. Let $p \in P$ be such that $0 \in \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\})$. Then since f is affine in p, by Theorem 1 it holds that $0 \in \text{hull}(\{\text{hull}(\{f(v,w) \mid w \in \mathcal{V}_P\}) \mid v \in \mathcal{V}_R, R \subseteq U\})$, or more simply $0 \in \text{hull}(\{f(v,w) \mid v \in \mathcal{V}_R, R \subseteq U, w \in \mathcal{V}_P\})$.

Proposition 5 *Let $P \subseteq \mathcal{P}$ be a polytope and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. If for some $w \in \mathcal{V}_P$, $0 \notin \text{hull}(\{f(v,w) \mid v \in \mathcal{V}_R, R \subseteq U\})$, then U is transient for some parameters $p \in P$.*

By Proposition 3, Proposition 5 is obviously sufficient for proving that a region is transient for some parameter in a polyhedral set. However, it may seem very conservative to test whether $0 \notin \text{hull}(\{f(v,w) \mid v \in \mathcal{V}_R, R \subseteq U\})$ is true only at the vertices of P instead of testing whether this is true for every parameter in P. The following proposition states that this is in fact equivalent.

Proposition 6 *Let $P \subseteq \mathcal{P}$ be a polytope and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. $\exists p \in P$ such that $0 \notin \text{hull}(\{f(v,p) \mid v \in \mathcal{V}_R, R \subseteq U\})$ iff $\exists w \in \mathcal{V}_P$ such that $0 \notin \text{hull}(\{f(v,w) \mid v \in \mathcal{V}_R, R \subseteq U\})$.*

Proof. The necessity is trivial. We prove sufficiency by contradiction. Let $p \in P$ and let I and J be two sets of indices labeling the vertices in $\cup_{R \subseteq U} \mathcal{V}_R$ and $\mathcal{V}_P$: $\cup_{R \subseteq U} \mathcal{V}_R = \{v_i\}_{i \in I}$ and $\mathcal{V}_P = \{w_j\}_{j \in J}$. Then, there exists $\{\mu_j\}_{j \in J}$ such that $\sum_{j \in J} \mu_j w_j = p$, with $\mu_j \geq 0$, $\forall j \in J$, and $\sum_{j \in J} \mu_j = 1$. Also, it holds that

$$\begin{aligned} \text{hull}(\{f(v_i,p)\}_{i \in I}) &= \text{hull}(\{\textstyle\sum_{j \in J} \mu_j f(v_i,w_j)\}_{i \in I}) \quad //f \text{ is affine in } p \\ &= \textstyle\bigoplus_{j \in J} \text{hull}(\{\mu_j f(v_i,w_j)\}_{i \in I}) \quad //\text{Minkowski sum of convex hulls} \end{aligned}$$

Then, for every $w_j \in \mathcal{V}_P$, $0 \in \text{hull}(\{f(v_i,w_j)\}_{i \in I})$ implies that $0 \in \text{hull}(\{\mu_j f(v_i,w_j)\}_{i \in I})$. So, by definition of Minkowski sum, we have $0 \notin \text{hull}(\{f(v_i,p)\}_{i \in I})$. Contradiction.

From a computational point of view, it is important to note that the conditions in Propositions 3, 4 and 5, can be simply evaluated by solving a linear optimization problem. The implementation of the approach described in Sections 5 and 6 resulted in a new version of a publicly-available tool for Robust Verification of Gene Networks (RoVerGeNe) (`http://iasi.bu.edu/~batt/rovergene/rovergene.htm`). RoVerGeNe is written in Matlab and uses MPT (polyhedral operations and linear optimization), MatlabBGL (SCC computation) and NuSMV (model-checking).

7 Analysis of the tuned transcriptional cascade

As explained in Section 3, in previous work we have predicted a way to tune the transcriptional cascade such that it satisfies its specifications, using a PMA model of the system. Before tuning the cascade experimentally, it is important to evaluate its robustness. To do so, we have tested whether the system satisfies the liveness property ϕ_1 (Figure 2(c)) for all of the 11 production and degradation rate parameters varying in $\pm 10\%$ intervals centered at their reference values. Because the network has no feedback loops, it is not difficult to show that oscillatory behaviors are not possible. Consequently, every (time-diverging) execution necessarily eventually remains in a single (non-transient) *rectangle*, instead of *SCC* in the general case (see Proposition 2). We have consequently applied Propositions 4 and 5 to rectangles only, to obtain tighter predictions.

Using RoVerGeNe, we have been able to prove this property in <4 hours (PC, 3.4 GHz processor, 1 Gb RAM). Given that the problem was to prove that a non-

computational time (in minutes)		number of uncertain parameters				
		0	2	5	8	11
number of continuous variables	3	0.03	0.04	0.07	-	-
	4	0.20	0.27	0.59	2.66	-
	5	2.60	3.28	6.46	29.11	207.76

Fig. 4. Computational time for the verification of a liveness property as a function of the number of variables and uncertain parameters. The 3- and 4-dimensional systems correspond to similar but shorter transcriptional cascades (see [12]).

trivial property holds for every initial condition in a 5-dimensional state space (1 input and 4 state variables) and for every parameter in an 11-dimensional parameter set, this example illustrates the applicability of the proposed approach to the analysis of networks of realistic size and complexity. Computational times for smaller instances of this problem are given in Figure 4.

The same test has been performed for $\pm 20\%$ parameter variations and a negative answer has been obtained (<4 hours). We recall that from negative answers, one can not conclude that the property is false for some parameters in the set. Nevertheless, the analysis of the counter-example given by the model checker has revealed that the system can remain in a (non-transient) rectangle in which the concentration of EYFP is below the minimal value allowed by the specifications ($5\ 10^5$), when the production rate constants κ^0_{eyfp} and κ_{eyfp} are minimal and the degradation rate constant γ_{eyfp} is maximal, in the $\pm 20\%$ intervals. As a consequence, the property is not robustly satisfied by the system for $\pm 20\%$ parameter variations. This analysis illustrates that relevant constraints on parameters are identified by this approach.

8 Discussion

This work addresses the problem of the verification of liveness properties of genetic regulatory networks modeled as PMA systems. It extends previous work on the verification of PMA systems with parameter uncertainty [2, 3]. Abstractions are used to obtain discrete representations of the dynamics of the system in state and parameter spaces, amenable to model checking. However, the abstractions introduce spurious behaviors along which time does not progress, called time-converging behaviors. The presence of these behaviors in the abstract systems generally causes the verification of liveness properties, expressing that something will eventually happen, to fail.

In this work, we proposed an approach to identify and rule out these behaviors, thus enforcing the progress of time in the abstract systems. We introduce the notion of transient regions as subsets of the state space that are eventually left by every solution trajectory, and established a simple relation between time-converging executions and regions corresponding to SCCs of the abstract discrete transition systems: executions that remain in a transient SCC are necessarily time-converging. Then, we provide sufficient conditions for characterizing transient regions in PMA systems. The approach is described for fixed parameters and systematically extended to deal with (polyhedral) sets of parameters.

This approach is implemented in a tool called RoVerGeNe. Its capacity to provide meaningful results for non-trivial problems on networks of biological interest is illustrated on the analysis of a transcriptional cascade.

The use of model checking for the analysis of biological networks has attracted much attention [20–24]. The verification of true (*i.e.* unbounded) liveness properties is not possible when the semantics is based on a set of necessarily time-bounded solution trajectories obtained by numerical simulation of ordinary differential equation models [20, 23]. For discrete [22, 23] or hybrid [21] models, fairness properties can be added in an *ad hoc* manner for the system at hand. So, although liveness properties are commonly encountered in biological applications, no systematic approach has been proposed yet for their verification.

More generally, this work addresses the problem of the verification of liveness properties of continuous or hybrid systems having dense-time semantics. In comparison with the amount of work done for the verification of safety properties of these systems, not much work has been done for liveness properties [9]. It has been proposed that the difficulty to enforce progress of time in dense-time systems makes liveness properties comparatively more difficult to analyze [9]. Tools supporting the verification of true (*i.e.* unbounded) liveness properties of dense-time systems are Uppaal [25], TReX [18] and RED [9]. However, their applicability is limited to timed automata, which have very restricted continuous dynamics. In contrast, our approach applies to any discrete abstraction provided that transient regions can be characterized. As mentioned in Section 5, a similar problem arise in untimed systems for the verification of liveness properties when abstractions are used [18, 19]. Progress of the abstract system is then enforced by the addition of fairness constraints, expressing that the system can not always remain in a given set of states. Because $\neg$FG('transient') (= GF($\neg$'transient'), Section 5) is a fairness constraint, our approach precisely amounts to deduce fairness constraints from the computation of transient regions. Consequently, our work can be regarded as an extension of an approach previously proposed for untimed systems and as a first step in the direction of the verification of liveness properties for general classes of continuous or hybrid systems. We envision that the notion of transient set can play for liveness properties a role symmetrical to the well-established role of positive invariant sets for safety properties.

Acknowledgements We would like to thank Boyan Yordanov for contributions to model development and acknowledge financial support by NSF 0432070.

References

1. Andrianantoandro, E., Basu, S., Karig, D., Weiss, R.: Synthetic biology: New engineering rules for an emerging discipline. Mol. Syst. Biol. (2006)
2. Batt, G., Belta, C.: Model checking genetic regulatory networks with applications to synthetic biology. CISE Tech. Rep. 2006-IR-0030, Boston University (2006)
3. Batt, G., Belta, C., Weiss, R.: Model checking genetic regulatory networks with parameter uncertainty. To appear in Bemporad, A., Bicchi, A., Buttazzo, G., eds.:Proc. HSCC'07. LNCS, Springer (2007)
4. Alur, R., Henzinger, T.A., Lafferriere, G.J., Pappas, G.: Discrete abstractions of hybrid systems. Proc. IEEE **88**(7) (2000) 971–984

5. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press (1999)
6. Alpern, B., Schneider, F.B.: Recognizing safety and liveness. Distrib. Comput. **2**(3) (1986) 117–126
7. Henzinger, T.A., Nicollin, X., Sifakis, J., Yovine, S.: Symbolic model checking for real-time systems. Inform. and Comput. **111** (1994) 193–244
8. Tripakis, S., Yovine, S.: Analysis of timed systems using time-abstracting bisimulations. Formal Methods System Design **18**(1) (2001) 25–68
9. Wang, F., Huang, G.D., Yu, F.: TCTL inevitability analysis of dense-time systems: From theory to engineering. IEEE Trans. Softw. Eng. (2006) In press.
10. Habets, L.C.G.J.M., Collins, P.J., van Schuppen, J.H.: Reachability and control synthesis for piecewise-affine hybrid systems on simplices. IEEE Trans. Aut. Control **51**(6) (2006) 938–948
11. Belta, C., Habets, L.C.G.J.M.: Controlling a class of nonlinear systems on rectangles. IEEE Trans. Aut. Control **51**(11) (2006) 1749–1759
12. Hooshangi, S., Thiberge, S., Weiss, R.: Ultrasensitivity and noise propagation in a synthetic transcriptional cascade. Proc. Natl. Acad. Sci. USA **102**(10) (2005) 3581–3586
13. de Jong, H., Gouzé, J.L., Hernandez, C., Page, M., Sari, T., Geiselmann, J.: Qualitative simulation of genetic regulatory networks using piecewise-linear models. Bull. Math. Biol. **66**(2) (2004) 301–340
14. Belta, C., Habets, L.C.G.J.M., Kumar, V.: Control of multi-affine systems on rectangles with applications to hybrid biomolecular networks. In: Proc. CDC'02. (2002)
15. Mestl, T., Plahte, E., Omholt, S.: A mathematical framework for describing and analysing gene regulatory networks. J. Theor. Biol. **176** (1995) 291–300
16. Glass, L., Kauffman, S.A.: The logical analysis of continuous non-linear biochemical control networks. J. Theor. Biol. **39**(1) (1973) 103–129
17. Lygeros, J., Johansson, K.H., Simič, S.N., Zhang, J., Sastry, S.S.: Dynamical properties of hybrid automata. IEEE Trans. Aut. Control **48**(1) (2003) 2–17
18. Bouajjani, A., Collomb-Annichini, A., Lacknech, Y., Sighireanu, M.: Analysis of fair parametric extended automata. In Cousot, P., ed.: Proc. SAS'01. LNCS 2126, Springer (2001) 335–355
19. Dams, D., Gerth, R., Grumberg, O.: A heuristic for the automatic generation of ranking functions. In: Proc. WAVe'00. (2000) 1–8
20. Antoniotti, M., Piazza, C., Policriti, A., Simeoni, M., Mishra, B.: Taming the complexity of biochemical models through bisimulation and collapsing: Theory and practice. Theor. Comput. Sci. **325**(1) (2004) 45–67
21. Batt, G., Ropers, D., de Jong, H., Geiselmann, J., Mateescu, R., Page, M., Schneider, D.: Validation of qualitative models of genetic regulatory networks by model checking : Analysis of the nutritional stress response in *E. coli*. Bioinformatics **21**(Suppl.1) (2005) i19–i28
22. Bernot, G., Comet, J.P., Richard, A., Guespin, J.: Application of formal methods to biological regulatory networks: Extending Thomas' asynchronous logical approach with temporal logic. J. Theor. Biol. **229**(3) (2004) 339–347
23. Calzone, L., Chabrier-Rivier, N., Fages, F., Soliman, S.: Machine learning biochemical networks from temporal logic properties. In Priami, C., Plotkin, G., eds: Trans. Comput. Syst. Biol. VI. LNBI 4220, Springer (2006) 68–94
24. Eker, S., Knapp, M., Laderoute, K., Lincoln, P., Talcott, C.L.: Pathway logic: Executable models of biological networks. In Gadducci, F., Montanari, U., eds.: Proc. WRLA'02. ENTCS 71, Elsevier (2002)
25. Bengtsson, J., Larsen, K.G., Larsson, F., Pettersson, P., Wang, Y., Weise, C.: New generation of UPPAAL. In: Proc. STTT'98. (1998)