Temporal Logic Analysis of Gene Networks Under Parameter Uncertainty

Grégory Batt, Calin Belta, and Ron Weiss

Abstract*—The lack of precise numerical information for the values of biological parameters severely limits the development and analysis of models of genetic regulatory networks. To deal with this problem, we propose a method for the analysis of genetic regulatory networks under parameter uncertainty. We consider models based on piecewise-multiaffine differential equations, dynamical properties expressed in temporal logic, and intervals for the values of uncertain parameters. The problem is then either to guarantee that the system satisfies the expected properties for every possible parameter value—the corresponding parameter set is then called valid—or to find valid subsets of a given parameter set. The proposed method uses discrete abstractions and model checking and allows for efficient search of the parameter space. However, the abstraction process creates spurious behaviors in the abstract systems, along which time does not progress. Consequently, the verification of liveness properties, expressing that something will eventually happen, and implicitly assuming progress of time, often fails. A solution to this second problem is proposed using the notion of transient regions. This approach has been implemented in a tool for robust verification of gene networks and applied to the tuning of a synthetic network built in *E. coli*.

***Index Terms*—Discrete abstraction, model checking, piecewise-multiaffine (PMA) system, genetic regulatory network, synthetic biology.**

I. Introduction

NUMEROUS cellular processes are controlled at the molecular level by networks of interactions between genes, proteins, and small molecules, called *genetic regulatory networks*. Understanding how the cellular behavior emerges from these networks of interactions is a central problem in systems and synthetic biology [3], [4]. Arguably, the most widely used modeling frameworks for the analysis of the dynamics of these networks are based on differential equations [5]. With few exceptions [6], it is generally assumed that the numerical values of state variables and model parameters are precisely known. However, given the current limitations of experimental measurement techniques, and the fact that parameter values themselves vary with the ever-fluctuating extra- and intracellular environments, the results obtained by these techniques may be of limited validity.

In this study, we present a method for the analysis of genetic regulatory networks with *parameter uncertainty*. We consider gene network models based on *piecewise-multiaffine* (PMA) differential equations, dynamical properties expressed in *temporal logic* (LTL), and *intervals* for the values of uncertain parameters. Unlike most other classes of gene network models, PMA models present specific mathematical properties that allow the formal verification of quantitative yet uncertain models. More precisely, the problems that we consider here are either to *guarantee* that the system satisfies the expected properties for *every* possible parameter value—the corresponding parameter set is then called *valid*—or to *find* valid subsets of a given parameter set.

In the proposed approach, we use a partition of the *state space* induced by the piecewise nature of the models and specific properties of multiaffine functions [7] to define an equivalence relation on parameters. Extending an approach widely used in hybrid systems theory, we use *discrete abstractions* [8] to transpose the problem defined on (infinite) continuous state and parameter spaces into a problem defined on (finite) discrete spaces. Algorithmic analysis of the abstract system by *model-checking* [9] is then possible. Because the abstractions used are conservative approximations, we guarantee that the parameter sets returned by the procedure are valid. However, not all valid parameters are guaranteed to be found. A second consequence of the use of abstraction is that spurious behaviors are introduced in the abstract systems. In particular, some behaviors, called *time-converging* behaviors, violate the natural requirement that along every behavior of a dynamical system, time progresses without upper bound [10], [11]. This causes the verification on the abstract systems of commonly encountered properties expressing that something will *eventually* happen [12], called *liveness* properties, to fail. We propose an approach to deal with this problem by *enforcing progress of time* in the abstract systems. We introduce the notion of *transient region* and show how transient regions can be used to rule out time-converging behaviors in abstract systems. Sufficient conditions for the identification of transient regions of uncertain PMA systems are then proposed. This approach has been implemented in a tool for Robust Verification of Gene Networks (RoVerGeNe) and applied to the analysis of the tuning of a synthetic gene network, built in the bacterium *E. coli*. This case study demonstrates the practical applicability and biological relevance of the proposed approach.

This paper is organized as follows. Section II introduces preliminary notions. Our modeling framework is presented

Manuscript received December 28, 2006; revised August 10, 2007. This work was supported by the National Science Foundation under Grant 0432070. This work presents a computational framework unifying preliminary results presented at the Tenth International Workshop on Hybrid Systems: Computation and Control, Pisa, Italy, April 3–5, 2007 and at the Thirteenth International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Braga, Portugal, March 24–April 1, 2007.

G. Batt was with the Center for Information and Systems Engineering and Center for BioDynamics, Boston University, Boston, MA 02245 USA. He is now with INRIA, Paris-Rocquencourt, F-38153 Le Chesnay, France (e-mail: gregory.batt@inria.fr)

C. Belta is with the Center for Information and Systems Engineering and Center for BioDynamics, Boston University, Boston, MA 02245 USA.

R. Weiss is with the Department of Electrical Engineering and Department of Molecular Biology, Princeton University, Princeton, NJ 08544 USA.

Digital Object Identifier 10.1109/TAC.2007.911330

in Section III. In Section IV, we detail our approach for the analysis of uncertain PMA systems using discrete abstractions. In Section V, we present how to enforce progress of time in the abstract systems for the verification of liveness properties. In Section VI, we present an application to the tuning of a synthetic gene network. The final section summarizes our contributions and discusses the results in the context of related work.

II. Preliminaries

A. Transition Systems and LTL Model-Checking

We consider Kripke structures $T = (S, \rightarrow, \Pi, \models)$ defined over sets of atomic propositions Π and simply called *transition systems* [9]. S is a (finite or infinite) set of states, $\rightarrow \subseteq S \times S$, a total transition relation, and $\models \subseteq S \times \Pi$, a satisfaction relation. An *execution* of T is an infinite sequence $e = (s_0, s_1, s_2, \ldots)$ such that, for every $i \geq 0, s_i \in S$ and $(s_i, s_{i+1}) \in \rightarrow$.

To every transition system $T = (S, \rightarrow, \Pi, \models)$, we associate the directed graph $G = (S, \rightarrow)$. A *strongly connected component* (SCC) of a directed graph $G = (S, \rightarrow)$ is a maximal subgraph $G' = (S', \rightarrow')$ of G such that, for every pair $s, s' \in S'$, there exists a path in G' from s to s'. When $|S'| = 1$, the SCC is called trivial. The SCCs of a transition system are the SCCs of its associated graph.

A simulation relation between transition systems is defined as follows.

Definition 1: [13] Let $T_1 = (S_1, \rightarrow_1, \Pi, \models_1)$ and $T_2 = (S_2, \rightarrow_2, \Pi, \models_2)$ be two transition systems defined on a same set of propositions Π. $\sim \subseteq S_1 \times S_2$ is a *simulation relation* between T_1 and T_2 if and only if (iff), for every $s_1 \in S_1, s_2 \in S_2$ such that $s_1 \sim s_2$, the following conditions hold:

- for every $s_1' \in S_1$ such that $s_1 \rightarrow_1 s_1'$, there exists $s_2' \in S_2$ such that $s_2 \rightarrow_2 s_2'$ and $s_1' \sim s_2'$;
- for every $\pi \in \Pi, s_1 \models_1 \pi$ iff $s_2 \models_2 \pi$.

Then, we say that T_2 simulates T_1 (denoted by $T_1 \preceq T_2$) iff there exists a simulation relation $\sim$ between T_1 and T_2 such that, for every $s_1 \in S_1$, there exists $s_2 \in S_2$ for which $s_1 \sim s_2$.

Given a transition system $T = (S, \rightarrow, \Pi, \models)$, an equivalence relation $\sim \subseteq S \times S$ is called *proposition-preserving* iff for every $s, s' \in S$ and $\pi \in \Pi$ such that $s \sim s'$ and $s \models \pi$, it holds that $s' \models \pi$. The *quotient transition system* of the transition system $T = (S, \rightarrow, \Pi, \models)$ given the proposition-preserving equivalence relation $\sim \subseteq S \times S$ is the transition system $T/_\sim = (S/_\sim, \rightarrow_\sim, \Pi, \models_\sim)$, where $S/_\sim$ is the quotient state space, i.e., the set of all equivalence classes of S, the transition relation $\rightarrow_\sim \subseteq S/_\sim \times S/_\sim$ is defined such that, for every $R, R' \in S/_\sim, R \rightarrow_\sim R'$ iff there exists $s \in R, s' \in R'$ such that $s \rightarrow s'$, and the satisfaction relation $\models_\sim \subseteq S/_\sim \times \Pi$ is defined such that, for every $R \in S/_\sim$, and $\pi \in \Pi, R \models_\sim \pi$ iff there exists $s \in R$ such that $s \models \pi$. It holds that $T \preceq T/_\sim$.

Definition 2 (Syntax of LTL Formulas [14]): The syntax of an LTL formula over a set of atomic propositions Π is inductively defined as follows:

- if $\phi \in \Pi$, then ϕ is an LTL formula;
- if ϕ and ϕ' are LTL formulas, then $\neg\phi, \phi \wedge \phi', \mathbf{X}\phi$ and $\phi \mathbf{U} \phi'$ are LTL formulas.

We also use the standard derived operators: $\phi \vee \phi'$ $(= \neg(\neg\phi \wedge \neg\phi'))$, $\phi \rightarrow \phi' (= \neg\phi \vee \phi')$, $\mathbf{F}\phi$ $(= \top \mathbf{U} \phi)$, and $\mathbf{G}\phi$ $(= \neg\mathbf{F}\neg\phi)$.

The semantics of LTL formulas are defined over executions of transition systems.

Definition 3 (Semantics of LTL Formulas [14]): Given an execution of the transition system $T, e = (s_0, s_1 \ldots)$, and an LTL formula ϕ over Π, $e \models_{LTL} \phi$ is inductively defined as follows:

- $e \models_{LTL} \phi$, if $s_0 \models \phi$, with $\phi \in \Pi$;
- $e \models_{LTL} \neg\phi$, if it is not the case that $e \models_{LTL} \phi$;
- $e \models_{LTL} \phi \wedge \phi'$, if $e \models_{LTL} \phi$ and $e \models_{LTL} \phi'$;
- $e \models_{LTL} \mathbf{X}\phi$, if $e^1 \models_{LTL} \phi$;
- $e \models_{LTL} \phi \mathbf{U} \phi'$, if there exists $j \geq 0$ such that $e^j \models \phi'$ and for all $i < j, e^i \models_{LTL} \phi$;

where e^i denotes the suffix of e starting at state s_i.

Temporal operators have an intuitive interpretation. $\mathbf{X}$ and $\mathbf{U}$ stand for "neXt state" and "Until," respectively. Similarly, $\mathbf{F}$ and $\mathbf{G}$ can be interpreted as meaning "for some Future state" and "Globally" (i.e., for all future states). The distinction between the satisfaction relation $\models$, defined over states and atomic propositions, and the satisfaction relation $\models_{LTL}$, defined over executions and LTL formulas, is generally clear from the context. When no ambiguity is possible, we use $\models$ for both.

A (finite or infinite) transition system $T = (S, \rightarrow, \Pi, \models)$ satisfies an LTL formula ϕ, denoted $T \models \phi$, if every execution of T satisfies the formula ϕ. If T is a *finite* transition system, model-checking tools exist that can test automatically whether $T \models \phi$. If not, a counterexample for the property is returned.

Finally, we will use the following proposition and its contrapositive. Note that the converse is not generally true.

Proposition 1 (Simulation Weakly Preserves LTL [15]): Let T_1 and T_2 be two transition systems such that $T_1 \preceq T_2$ and ϕ be an LTL formula. If $T_2 \models \phi$, then $T_1 \models \phi$.

B. Convex Sets and Polytopes, Affine and Multiaffine Functions

Let S be a subset of $\mathbb{R}^n$. $\bar{S}$ and $\text{hull}(S)$ denote respectively its closure in $\mathbb{R}^n$ and its convex hull. A *polytope* is a bounded intersection of finitely many open or closed halfspaces. A polytope $P \subset \mathbb{R}^n$ is *hyperrectangular* if it is the Cartesian product of n (possibly degenerate) intervals of the real line. A *face* of a polytope P is the intersection of $\bar{P}$ with one of its supporting hyperplanes. A *facet* is an $N-1$-dimensional face, with $N \leq n$, the dimension of P. The sets of points $v_1, \ldots, v_p \in \mathbb{R}^n$ satisfying $\bar{P} = \text{hull}(\{v_1, \ldots, v_p\})$ 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. The Minkowski sum of two sets P and Q is $P \bigoplus Q = \{p + q \,|\, p \in P \text{ and } q \in Q\}$.

An *affine* function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, with $n, m \in \mathbb{N}$, is a polynomial of degree at most 1. A *multiaffine* function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, with $n, m \in \mathbb{N}$, is a polynomial in the variables $x_1, \ldots, x_n$ with the property that the degree of f in any of the variables x_i is at most 1. Stated differently, nonlinearities are restricted to products of distinct variables. Theorem 1 [respectively, 2] states that the value of an affine (multiaffine) function in a polytope (respectively, hyperrectangular polytope) is a convex combination of the values of the function at the vertices of the polytope (respectively, hyperrectangular polytope).

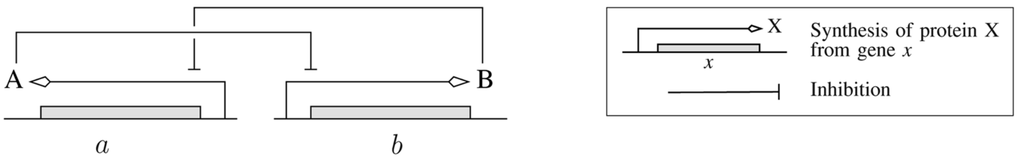


Fig. 1. Cross-inhibition network: gene network comprising two genes a and b and coding for two repressor proteins, A and B. Each protein represses the expression of the other gene, that is, the synthesis of the other protein.

Theorem 1: [16] Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be an affine function and P be a polytope in $\mathbb{R}^n$. Then,

$$f(\bar{P}) = \mathrm{hull}(\{f(v) \,|\, v \in \mathcal{V}_P\}).$$

Theorem 2: [7] Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a multiaffine function and P be a hyperrectangular polytope in $\mathbb{R}^n$. Then

$$f(\bar{P}) \subseteq \mathrm{hull}(\{f(v) \,|\, v \in \mathcal{V}_P\}).$$

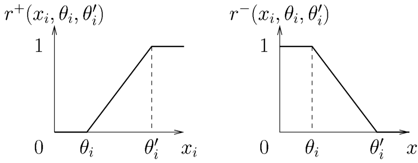


Fig. 2. Ramp functions r^+ and r^-. Ramp functions are used to capture the effect on gene expression of a regulatory protein (activator or inhibitor). θ_i and θ_i' are *threshold* parameters.

III. Uncertain PMA Models of Genetic Regulatory Networks

A. PMA Systems and LTL Specifications

Here, we present a formalism for modeling gene networks and use a cross-inhibition network (Fig. 1) as an illustrative example. Some notations and terminology are adapted from [17].[1] We consider a gene 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} denote a minimal and a maximal value for p_j.

The dynamics of the network are given by the differential equations

$$\dot{x}_i = f_i(x, p) = \sum_{j \in P_i} \kappa_i^j \, r_i^j(x) - \sum_{j \in D_i} \gamma_i^j \, r_i^j(x) \, x_i \qquad (1)$$

where $i \in \{1, \ldots, n\}$, 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, PMA functions called *regulation functions*. PMA functions arise from products of *ramp functions* r^+ and r^- (Fig. 2), used to capture the combined effect of several regulatory proteins on the control of gene expression or protein degradation (see Fig. 6(b) (2′) and [18]). With the additional assumption that r_i^j does not depend on x_i for $j \in D_i$,[2] it holds that $f = (f_1, \ldots, f_n) : \mathcal{X} \times \mathcal{P} \to \mathbb{R}^n$ is a (nonsmooth) *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. Each component of the vector p of uncertain parameters is either a production or a degradation rate parameter. Finally, (1) is easily extended to account for constant inputs u by considering u as new variables satisfying $\dot{u} = 0$.

The cross-inhibition network in Fig. 1 can be represented by the following PMA differential equations:

$$\dot{x}_a = \kappa_a r^- \left(x_b, \theta_b^1, \theta_b^2\right) - \gamma_a \, x_a$$
$$\dot{x}_b = \kappa_b r^- \left(x_a, \theta_a^1, \theta_a^2\right) - \gamma_b \, x_b.$$

For example, the first equation states that protein A synthesis is inhibited by protein B (r^- function) and that its degradation is not regulated. Note that, because of its simplicity (the expression of each gene is controlled by a single protein), this model is actually piecewise-affine. We assume the following values for known and uncertain parameters:

	κ_i	γ_i	θ_i^1	θ_i^2
x_a	$[0, 40]$	2	8	12
x_b	$[0, 20]$	1	8	12

Synthesis parameters are unknown: $p = (\kappa_a, \kappa_b) \in \mathcal{P} = [0, 40] \times [0, 20]$. For illustrating our purpose, we also consider a particular parameter $p^1 \in \mathcal{P}$, with $p^1 = (36, 17)$.

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 PMA function f defined as above and a set of atomic propositions Π: $\Sigma = (f, \Pi)$.

The cross-inhibition network is known to be *bistable*. If the system is in a state in which the concentration of protein A is low and the concentration of protein B is high, then it will remain in such a state for all time. A symmetrical property holds with the

[1] Note, however, that, unlike [17], we consider here *quantitative* piecewise-*multiaffine* models.

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

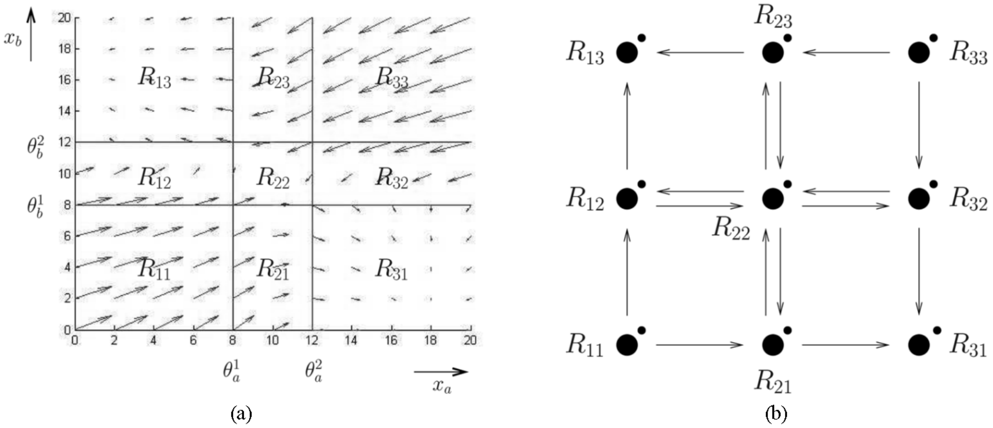


Fig. 3. (a) Vector field describing the continuous dynamics in the state space of the cross-inhibition network for parameter $p^1 = (\kappa_a, \kappa_b) = (36, 17)$. (b) Discrete abstraction of the dynamics in (a). Dots denote self transitions.

concentrations of A and B being high and low, respectively. This property can be expressed in LTL by the formula ϕ_1

$$\phi_1 = \left(x_a < \theta_a^1 \wedge x_b > \theta_b^2 \rightarrow \mathbf{G}\left(x_a < \theta_a^1 \wedge x_b > \theta_b^2\right)\right) \\ \wedge \left(x_b < \theta_b^1 \wedge x_a > \theta_a^2 \rightarrow \mathbf{G}\left(x_b < \theta_b^1 \wedge x_a > \theta_a^2\right)\right).$$

For example, the first part of the property expresses that, if the concentrations of proteins A and B are respectively low $(x_a < \theta_a^1)$ and high $(x_b > \theta_b^2)$, then the system will always $(\mathbf{G})$ remain in such a state. This states that the region $\{x = (x_a, x_b) \in \mathcal{X} \mid x_a < \theta_a^1 \wedge x_b > \theta_b^2\}$ is invariant. We refer the reader to [19] for a discussion on the use of invariants to express stability in biology. Another well-known property of the cross-inhibition network is *mutual exclusion*. The proteins cannot both remain present at high concentrations. Stated differently, it should hold that, irrespective of the initial state, eventually $(\mathbf{F})$ at least one protein is present at low or average concentration (i.e., $x_i < \theta_i^2$)

$$\phi_2 = \mathbf{F}\left(x_a < \theta_a^2 \vee x_b < \theta_b^2\right). \tag{2}$$

The use of PMA models for gene networks has been proposed in [20] (see also [21] for a related piecewise-continuous approach). The class of uncertain PMA models considered here is also closely related to the class of piecewise-affine (PA) differential equation models proposed by Glass and Kauffman [22] (see [17] and [21] for further developments and [23] and [24] for alternative, discrete formalisms). In PA models, step functions are used instead of ramp functions. Both are simplifications of the sigmoids (e.g., Hill functions) traditionally used for representing genetic regulations [25], [26]. However, contrary to step functions, ramp functions capture the graded response of gene expression to continuous changes in effector (activator or inhibitor) concentrations and allow the development of finer models.

B. Embedding Transition Systems

The specific form of the PMA function f suggests a division of the state space $\mathcal{X}$ into hyperrectangular regions (see Fig. 3(a) for our example). For every $i \in \{1, \ldots, n\}$, 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}. Then, we define $\mathcal{R}$ as the following set of n-dimensional hyperrectangular polytopes $R \subseteq \mathcal{X}$, simply 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

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

The n-tuple c is called the *coordinate* of rectangle R_c. 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 $\mathcal{X}_\mathcal{R}$. Two rectangles R and R' are said to be *adjacent*, denoted $R \Bumpeq R'$, if they share a facet. coord : $\mathcal{R} \rightarrow \prod_{i=1}^{n}\{1, \ldots, l_i - 1\}$ maps every rectangle $R \in \mathcal{R}$ to its coordinate, and rect : $\mathcal{X}_\mathcal{R} \rightarrow \mathcal{R}$ maps every point x in $\mathcal{X}_\mathcal{R}$ to the rectangle R such that $x \in R$. For the cross-inhibition network, the set $\mathcal{R} = \{R_{11}, \ldots, R_{33}\}$ of all rectangles is represented in Fig. 3(a). R_{11} and R_{21} are adjacent $(R_{11} \Bumpeq R_{21})$, whereas R_{11} and R_{22} are not.

Formally, the semantics of a PMA system Σ is defined by means of an *embedding transition system*.

Definition 4: Let $p \in \mathcal{P}$. The embedding transition system associated with the PMA system $\Sigma = (f, \Pi)$ is $T_\mathcal{X}(p) = (\mathcal{X}_\mathcal{R}, \rightarrow_{\mathcal{X},p}, \Pi, \models_\mathcal{X})$ defined such that:

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

Remark: Not all solution trajectories of (1) are represented by executions of the embedding transition system. First, due to our restricted notion of adjacency $(\Bumpeq)$, solution trajectories of (1) that go from a rectangle to another by passing through a face of low $(< n - 1)$ dimension are not represented in the embedding. Second, the dynamics of the system in $\mathcal{X} \setminus \mathcal{X}_\mathcal{R}$ (including threshold hyperplanes) is not described by the embedding. However, since the vector field is continuous everywhere, trajectories originating in full-dimensional rectangles can not "disappear" in a facet by sliding along the supporting hyperplane. Consequently, the embedding describes *almost* all solution trajectories of (1), which is satisfying for all practical purposes.

Definition 5: 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.

Then, valid parameter sets are defined as follows.

Definition 6: Let Σ be a PMA system and ϕ an LTL formula. A parameter set $P \subseteq \mathcal{P}$ is *valid* for ϕ iff Σ satisfies ϕ for almost all $p \in P$.

Again, the use of *almost all* is motivated by the fact that this criterion is sufficient for all practical purposes and allows us to avoid tedious technicalities. Finally, we consider the following problems.

Problem 1: Let $\Sigma = (f, \Pi)$ be a PMA system, let $\mathcal{P}$ be a hyperrectangular parameter space, and let ϕ be an LTL formula over Π.

A) Robustness Analysis: Check whether $\mathcal{P}$ is valid for ϕ.

B) Synthesis: Find a set $P \subseteq \mathcal{P}$ such that P is valid for ϕ.

Our focus on hyperectangular parameter spaces comes from our hypothesis that uncertain parameters are given by intervals. However, in full generality, the proposed method can deal with bounded polyhedral parameter spaces.

IV. Analysis of PMA Systems With Parameter Uncertainty

A. Discrete Abstraction

We use discrete abstractions [8] to obtain finite transition systems preserving dynamical properties of $T_{\mathcal{X}}(p)$ and amenable to algorithmic verification [9]. Let $\sim_{\mathcal{R}} \subseteq \mathcal{X}_{\mathcal{R}} \times \mathcal{X}_{\mathcal{R}}$ be the (proposition-preserving) equivalence relation defined by the surjective map $\mathrm{rect} : x \sim_{\mathcal{R}} x'$ iff $\mathrm{rect}(x) = \mathrm{rect}(x')$. $\mathcal{R}$ is the set of equivalence classes. Then, we define the *discrete abstraction* of $T_{\mathcal{X}}(p)$ as follows.

Definition 7: Let $p \in \mathcal{P}$. The discrete abstraction of $T_{\mathcal{X}}(p)$ is $T_{\mathcal{R}}(p) = (\mathcal{R}, \rightarrow_{\mathcal{R},p}, \Pi, \models_{\mathcal{R}})$, the quotient of $T_{\mathcal{X}}(p)$ given the equivalence relation $\sim_{\mathcal{R}}$.

For our example network, $T_{\mathcal{R}}(p^1)$ is represented in Fig. 3(b). From the definition of quotient transition systems (Section II), the following is true.

Proposition 2: For every $p \in \mathcal{P}$

$$T_{\mathcal{X}}(p) \preceq T_{\mathcal{R}}(p).$$

In words, the discrete transition system $T_{\mathcal{R}}(p)$ is a *conservative approximation* of the continuous dynamics of the PMA system described by $T_{\mathcal{X}}(p)$, in the sense that, for every execution $e_{\mathcal{X}} = (x^0, x^1, \ldots)$ of $T_{\mathcal{X}}(p)$, there exists a corresponding execution in $T_{\mathcal{R}}(p)$: $e_{\mathcal{R}} = (\mathrm{rect}\ (x^0), \mathrm{rect}\ (x^1), \ldots)$. Because simulation relations *weakly preserve* LTL (Proposition 1), if we can prove that an LTL property ϕ holds for $T_{\mathcal{R}}(p)$, then it also holds for $T_{\mathcal{X}}(p)$. Note that the converse does not necessarily hold.

To compute $T_{\mathcal{R}}(p)$, we provide the following characterization of its transition and satisfaction relations. These results exploit specific properties of multiaffine functions defined over hyperrectangular polytopes [7].

Proposition 3: Let $p \in \mathcal{P}$. $T_{\mathcal{R}}(p) = (\mathcal{R}, \rightarrow_{\mathcal{R},p}, \Pi, \models_{\mathcal{R}})$, where:

- $\rightarrow_{\mathcal{R},p} \subseteq \mathcal{R} \times \mathcal{R}$ is such that $(R, R') \in \rightarrow_{\mathcal{R},p}$ iff $R = R'$, or $R \Bumpeq R'$ and there exists $v \in \mathcal{V}_R \cap \mathcal{V}_{R'}$ such that
$$f_i(v,p)(c'_i - c_i) > 0$$
with $c = \mathrm{coord}(R), c' = \mathrm{coord}(R')$ and $i \in \{1, \ldots, n\}$ such that $c_i \neq c'_i$.
- $\models_{\mathcal{R}} \subseteq \mathcal{R} \times \Pi$ is such that $(R, \pi) \in \models_{\mathcal{R}}$ iff for every $x \in R$, $(x, \pi) \in \models_{\mathcal{X}}$.

Proof: Let $R, R' \in \mathcal{R}$. By Definitions 4 and 7, it is clear that, if neither $R = R'$ nor $R \Bumpeq R'$, there cannot exist a transition from R to R'. If $R = R'$, then, since it exists a solution of (1) that remains in R on $[0, \tau]$ for some $\tau > 0$, there exists a (self) transition from R to R' (Definitions 4 and 7). The last case is when $R \Bumpeq R'$. Then, let $c = \mathrm{coord}(R), c' = \mathrm{coord}(R')$ and $i \in \{1, \ldots, n\}$ such that $c_i \neq c'_i$ and let F be the facet shared by R and R'. We assume without loss of generality that $c'_i - c_i = 1$, the other case $(= -1)$ being symmetrical.

$\Rightarrow$ (by contradiction): Suppose that, for every $v \in \mathcal{V}_R \cap \mathcal{V}_{R'} = \mathcal{V}_F, f_i(v,p) \leq 0$. Using Theorem 2, it holds that, for every $x \in F, f_i(x,p) \leq 0$. Consequently, no solution can enter R' from R and $(R, R') \notin \rightarrow_{\mathcal{R},p}$.

$\Leftarrow$: Assume that there exists $v \in \mathcal{V}_F$ such that $f_i(v,p) > 0$. By continuity of f, there exists a ball $B_{v,\epsilon}$ of center v and radius ϵ such that $\forall x \in B_{v,\epsilon}, f_i(x,p) > 0$. In particular, there exist $x_f \in F, x_f \neq v$, such that $f_i(x_f, p) > 0$. Then, there exists a solution entering R' from R without leaving $\bar{R} \cup \bar{R}'$, and, by Definitions 4 and 7, $(R, R') \in \rightarrow_{\mathcal{R},p}$.

The characterization of $\models_{\mathcal{R}}$ follows immediately from the fact that the equivalence relation $\sim_{\mathcal{R}}$ preserves the atomic propositions in Π. ■

Informally, Proposition 3 simply states that there is a transition between two adjacent rectangles iff there exists at least one common vertex at which the direction of the vector field $(f_i(v,p))$ is in agreement with the relative position of the two rectangles $(c'_i - c_i)$. An approach using similar intuitions has been proposed for reachability analysis of multiaffine systems in [27].

For exemplification, consider the two rectangles R_{11} and R_{21} in Fig. 3(a). They share two vertices: $v^1 = (\theta_a^1, 0)$ and $v^2 = (\theta_a^1, \theta_b^1)$. From Proposition 3, there is a transition from R_{11} to R_{21}, because $f_a(v^1, p^1) > 0$, and there is no transition from R_{21} to R_{11}, because neither $f_a(v^1, p^1) < 0$ nor $f_a(v^2, p^1) < 0$ [check with Fig. 3(b)].

In summary, Proposition 3 provides a means to compute the relation $\rightarrow_{\mathcal{R},p}$ for a given parameter p by evaluating f at all the vertices of the rectangles. The computation of the set of states $\mathcal{R}$ and of the satisfaction relation $\models_{\mathcal{R}}$ are trivial. So $T_{\mathcal{R}}(p)$ can be computed, and because $T_{\mathcal{R}}(p)$ is a finite transition system, one can use model checking for testing whether $T_{\mathcal{R}}(p) \models \phi$. If the abstract system $T_{\mathcal{R}}(p)$ satisfies ϕ, then so does the original system $T_{\mathcal{X}}(p)$ (Propositions 1 and 2), and p is valid for ϕ.

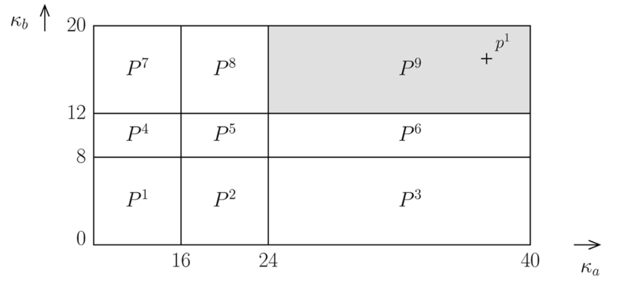


Fig. 4. Parameter space in the dimensions of κ_a and κ_b. Parameter $p^1 = (36, 17)$ is represented. The shaded region is a set of valid parameters for property ϕ_1.

Conversely, if $T_{\mathcal{R}}(p)$ does not satisfy ϕ, no conclusion on the validity of p can be obtained. If some parameters are unknown, we will use Proposition 3 to define an equivalence relation on parameters and reason for parameter sets.

B. Parameter Equivalence Classes

Consider a vertex $v \in \mathcal{V}_R, R \in \mathcal{R}$. Because f is an affine function of $p, f_i(v,p)$ is an affine expression in p: $f_i(v,p) = a_{i,v}^T p + b_{i,v}$, with $a_{i,v} \in \mathbb{R}^m$ and $b_{i,v} \in \mathbb{R}$. Let Ψ be the set of all such nonconstant $(a_{i,v} \neq 0)$ affine expressions

$$\Psi = \big\{ f_i(v,p) = a_{i,v}^T p + b_{i,v} \,|\, i \in \{1,\ldots,n\}, \\ v \in \mathcal{V}_R, R \in \mathcal{R} \text{ and } a_{i,v} \neq 0 \big\}.$$

After having removed repeated elements, we denote by n_Ψ the cardinality of Ψ and order the elements in Ψ: $\Psi = \{\psi_1, \ldots, \psi_{n_\Psi}\}$. For our example network, with uncertain parameters κ_a and κ_b, out of the 32 affine expressions only four different nonconstant expressions exist $(n_\Psi = 4)$

$$\begin{aligned} \Psi =& \{\psi_1, \psi_2, \psi_3, \psi_4\}, \qquad \text{with} \\ & \psi_1(p) = \kappa_a - 24; \quad \psi_2(p) = \kappa_b - 12 \\ & \psi_3(p) = \kappa_b - 8; \quad \psi_4(p) = \kappa_a - 16. \end{aligned}$$

The affine predicates $\psi_i(p) = 0, \psi_i \in \Psi$, divide the *parameter space* into polyhedral regions (Fig. 4[3]). These regions can be represented by a Boolean encoding. Let $\mathcal{B}^l$ be the set of Boolean numbers of length l: $\mathcal{B}^l = \{0,1\}^l$. We denote by ϵ the Boolean of length 0. Then, to every Boolean $b \in \mathcal{B}^l, l \in \{0,\ldots,n_\Psi\}$, we associate the parameter set P_b such that $P_\epsilon = \mathcal{P}$ and, if $b \neq \epsilon$, then

$$P_b = \{p \in \mathcal{P} \,|\, \forall i \in \{1,\ldots,l\} : \psi_i(p) < 0, \text{ if } b_i = 0, \text{and} \\ \psi_i(p) > 0, \text{ if } b_i = 1\}.$$

The sets P_b are subsets of $\mathcal{P}$ obtained by adding constraints of type $\psi_i(p) < 0$ or $\psi_i(p) > 0$, with $\psi_i \in \Psi$. If b is a prefix of b', then $P_{b'} \subseteq P_b$. The hierarchy between the sets P_b induced by the set-inclusion partial order is represented in Fig. 5 for the cross-inhibition network (see [28] and [29] for similar ideas in the context of predicate abstraction).

We say that two parameters p and p' are *equivalent* if their associated discrete transition systems $T_{\mathcal{R}}(p)$ and $T_{\mathcal{R}}(p')$ are isomorphic. A similar definition is used in [17] and [30]. Naturally, a PMA system satisfies the same LTL properties for two equivalent parameters.

Definition 8: Let $\sim_{\mathcal{P}} \subseteq \mathcal{P} \times \mathcal{P}$ be the equivalence relation defined by $p \sim_{\mathcal{P}} p'$ iff $T_{\mathcal{R}}(p) = T_{\mathcal{R}}(p')$.

Then, it holds that the set of all predicates $\psi_i(p) = 0, \psi_i \in \Psi$, divide the parameter space in equivalence classes.

Proposition 4: Let $b_\Psi \in \mathcal{B}^{n_\Psi}$. For all $p, p' \in P_{b_\Psi}, p \sim_{\mathcal{P}} p'$.

Proof: Let $b_\Psi \in \mathcal{B}^{n_\Psi}$ and $p, p' \in P_{b_\Psi}$. Then, $\forall i \in \{1,\ldots,n\}, R \in \mathcal{R}$ and $v \in \mathcal{V}_R, f_i(v,p) \# 0$ iff $f_i(v,p') \# 0$, with $\# \in \{<,>\}$. So, by Proposition 3, $T_{\mathcal{R}}(p) = T_{\mathcal{R}}(p')$ and $p \sim_{\mathcal{P}} p'$. ■

With $b_\Psi \in \mathcal{B}^{n_\Psi}$, if for *some* $p \in P_{b_\Psi}, T_{\mathcal{R}}(p) \models \phi$, then using Propositions 1, 2, and 4, it holds that, for *all* $p \in P_{b_\Psi}, T_{\mathcal{X}}(p) \models \phi$: P_{b_Ψ} is a valid parameter set. Since we can compute $T_{\mathcal{R}}(p)$ for any given p (Proposition 3), solutions to Problems 1.A and 1.B can be obtained by testing for every equivalence class $P_{b_\Psi} \subseteq \mathcal{P}$ whether $T_{\mathcal{R}}(p) \models \phi$ for some (randomly chosen) $p \in P_{b_\Psi}$. Note, however, that, if $T_{\mathcal{R}}(p) \not\models \phi$, no conclusion can be obtained on the validity of P_{b_Ψ}, since $T_{\mathcal{R}}(p) \not\models \phi$ does not imply that $T_{\mathcal{X}}(p) \not\models \phi$. On our example network, only two equivalence classes, P_{1110} and P_{1111}, both corresponding to P^9, are found to be valid for the bistability property ϕ_1 (Figs. 5 and 4). However, this naive approach is impractical since the number of equivalence classes (i.e., the leaves of the tree in Fig. 5) increases exponentially with the number of affine predicates in Ψ, the latter increasing exponentially with the number of variables and uncertain parameters. A more efficient approach is proposed in the next section.

C. Hierarchical Parameter Space Exploration

Our goal is to describe the behavior of the network for *sets* of parameters $P \subseteq \mathcal{P}$. To do so, we introduce two transition systems, $T_{\mathcal{R}}^{\exists}(P)$ and $T_{\mathcal{R}}^{\forall}(P)$. Then, we show how they can be computed and how they can be used to prove properties for sets of parameters.

Definition 9: Let $P \subseteq \mathcal{P}$. Then, $T_{\mathcal{R}}^{\exists}(P) = (\mathcal{R}, \rightarrow_{\mathcal{R},P}^{\exists}, \Pi, \models_{\mathcal{R}})$ and $T_{\mathcal{R}}^{\forall}(P) = (\mathcal{R}, \rightarrow_{\mathcal{R},P}^{\forall}, \Pi, \models_{\mathcal{R}})$, where

- $(R, R') \in \rightarrow_{\mathcal{R},P}^{\exists}$ iff $\exists p \in P$ such that $(R, R') \in \rightarrow_{\mathcal{R},p}$ in $T_{\mathcal{R}}(p)$, and
- $(R, R') \in \rightarrow_{\mathcal{R},P}^{\forall}$ iff $\forall p \in P, (R, R') \in \rightarrow_{\mathcal{R},p}$ in $T_{\mathcal{R}}(p)$.

By Definition 9, $T_{\mathcal{R}}^{\exists}(P)$ contains all the transitions present in at least one transition system $T_{\mathcal{R}}(p)$, and $T_{\mathcal{R}}^{\forall}(P)$ contains only the transitions present in all the transition systems $T_{\mathcal{R}}(p)$. For every $p \in P, T_{\mathcal{R}}^{\exists}(P)$ simulates $T_{\mathcal{R}}(p)$, which simulates $T_{\mathcal{R}}^{\forall}(P)$. This follows immediately from the definition of simulation between transition systems, using the fact that $\rightarrow_{\mathcal{R},P}^{\forall} \subseteq \rightarrow_{\mathcal{R},p} \subseteq \rightarrow_{\mathcal{R},P}^{\exists}$. Informally, $T_{\mathcal{R}}^{\exists}(P)$ and $T_{\mathcal{R}}^{\forall}(P)$ can be, respectively, considered as over- and under-approximations of the possible behaviors of $T_{\mathcal{R}}(p)$, when p varies in P.

Proposition 5: For every $p \in P$

$$T_{\mathcal{R}}^{\forall}(P) \preceq T_{\mathcal{R}}(p) \preceq T_{\mathcal{R}}^{\exists}(P).$$

In order to compute $T_{\mathcal{R}}^{\exists}(P)$ and $T_{\mathcal{R}}^{\forall}(P)$, we first introduce the function g that associates to every pair of adjacent rectangles R

[3] Note that, in general, the partition of the parameter space is not hyperrectangular.

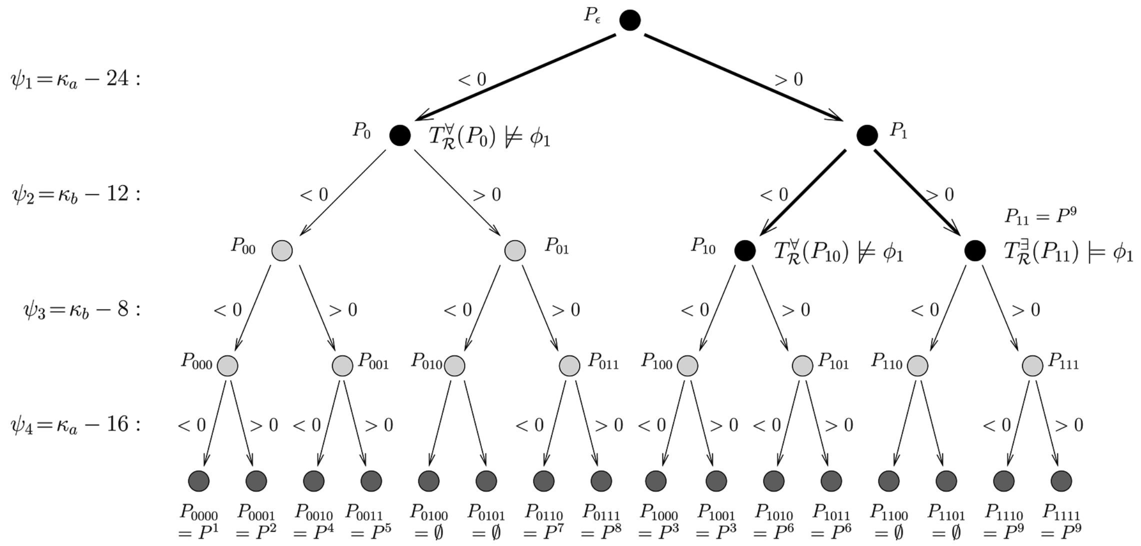


Fig. 5. Hierarchy between the parameter sets P_b, represented as a binary tree. Arrows indicate set inclusion: $P \rightarrow P'$ means $P' \subseteq P$. Leaves (dark gray) correspond to parameter equivalence classes. $P^1, \ldots, P^9$ refer to regions in Fig. 4. The fragment of the tree actually computed during hierarchical parameter space exploration for the analysis of property ϕ_1 is emphasized. Model checking results used for backtracking are shown at the nodes where the recursive search stops.

and R' the set of parameters $p \in \mathcal{P}$ for which there is a transition from R to R' in $T_{\mathcal{R}}(p)$

$$g(R, R') = \{p \in \mathcal{P} \mid \exists v \in \mathcal{V}_R \cap \mathcal{V}_{R'} \text{ such that } f_i(v, p)(c'_i - c_i) > 0\}$$

with $c = \text{coord}(R), c' = \text{coord}(R')$ and $i \in \{1, \ldots, n\}$ such that $c_i \neq c'_i$.

Because for all $i \in \{1, \ldots, n\}$ and $v \in \mathcal{V}_R, R \in \mathcal{R}$, the $f_i(v, p)$ are affine expressions in p, the sets $g(R, R')$ correspond to unions of polytopes in $\mathcal{P}$. Then, using the following proposition, the computation of the transition systems $T^{\exists}_{\mathcal{R}}(P)$ and $T^{\forall}_{\mathcal{R}}(P)$ for polyhedral sets P simply amounts to compute intersections and decide inclusions of unions of polytopes, which are standard polyhedral operations efficiently implemented in toolboxes.

Proposition 6: Let $P \subseteq \mathcal{P}$

$$-(R, R') \in \rightarrow^{\exists}_{\mathcal{R},P} \quad \text{iff either } R = R', \quad \text{or } R \mathrel{\diamond} R' \text{ and } P \cap g(R, R') \neq \emptyset,$$

$$-(R, R') \in \rightarrow^{\forall}_{\mathcal{R},P} \quad \text{iff either} R = R', \quad \text{or } R \mathrel{\diamond} R' \text{ and } P \subseteq g(R, R').$$

Proof: Let $P \subseteq \mathcal{P}$ and $R, R' \in \mathcal{R}$ be such that $R \mathrel{\diamond} R'$ (the other cases being trivial). From Proposition 3, it is easy to see that $g(R, R')$ is the set of parameters $p \in \mathcal{P}$ for which there is a transition from R to R' in $T_{\mathcal{R}}(p)$. Then, the result follows from the definition of the transition relations $\rightarrow^{\exists}_{\mathcal{R},P}$ and $\rightarrow^{\forall}_{\mathcal{R},P}$ (Definition 9). ■

Using Propositions 1, 2, and 5, it holds that, for *any* $P \in \mathcal{P}$, if $T^{\exists}_{\mathcal{R}}(P) \models \phi$, then $\forall p \in P, T_{\mathcal{X}}(p) \models \phi$: P is a valid parameter set. Alternatively, using the contrapositive of Proposition 1 and Proposition 5, it also holds that if $T^{\forall}_{\mathcal{R}}(P) \not\models \phi$, then $\forall p \in P, T_{\mathcal{R}}(p) \not\models \phi$: no valid parameter can be found in P using our approach, either because P contains no valid parameter, or because the discrete abstraction is overly conservative. Otherwise ($T^{\exists}_{\mathcal{R}}(P) \not\models \phi$ and $T^{\forall}_{\mathcal{R}}(P) \models \phi$), it is worth inspecting subsets of P, that may contain valid parameter sets. Accordingly, we propose an algorithm that explores $\mathcal{P}$ in a hierarchical manner by considering parameter sets P_b associated with Booleans of increasing length, starting from P_ϵ. The main procedure is Algorithm 1, which essentially calls COMPUTE_PARAMETER_CONSTRAINTS and TEST_PARAMETER_SET. The function COMPUTE_PARAMETER_CONSTRAINTS (Algorithm 2) is a preprocessing step, in which the set Ψ and the function g are computed. Finally, the procedure TEST_PARAMETER_SET (Algorithm 3) recursively explores a binary tree, represented in Fig. 5 for our example and, for each node, computes $P_b, T^{\exists}_{\mathcal{R}}(P_b)$ and $T^{\forall}_{\mathcal{R}}(P_b)$, and test whether $T^{\exists}_{\mathcal{R}}(P_b) \models \phi$ and whether $T^{\forall}_{\mathcal{R}}(P_b) \models \phi$. As explained above, Proposition 5 is used to stop the recursive search as soon as possible (either because $T^{\exists}_{\mathcal{R}}(P_b) \models \phi$, or because $T^{\forall}_{\mathcal{R}}(P_b) \not\models \phi$). For the leaves of the search tree (i.e., the parameter equivalence classes), $T^{\exists}_{\mathcal{R}}(P_b) = T^{\forall}_{\mathcal{R}}(P_b)$ and the search necessarily terminates. The maximum recursion depth is n_Ψ.

Because the efficiency of the computations may significantly depend on the order in which the affine predicates $\psi_i(p) = 0, \ \psi_i \in \Psi$, are considered during the search, we implemented a simple heuristic (in function REMOVE_REPEATED_ELEMENT_AND_SORT, Algorithm 2) that orders first the predicates splitting the parameter space the more evenly (i.e., yielding two polytopes of similar volumes).

The fragment of the tree actually computed for the analysis of property ϕ_1 is represented in Fig. 5. The same result is obtained as previously (P^9 is a valid parameter set), but in much fewer tests. (Please see Algorithms 1–3).

Note that, in general, $T^{\forall}_{\mathcal{R}}(P)$ does *not* provide information on the *original* system $T_{\mathcal{X}}(p)$, since no simulation relation exists between $T^{\forall}_{\mathcal{R}}(P)$ and $T_{\mathcal{X}}(p)$. Nevertheless, it makes it possible to identify (potentially large) regions of the parameter space in which no valid parameter can be found. Consequently, it plays

Algorithm 1 RoVerGeNe

```
// input: $(f, \Pi)$ a PMA system, $\phi$ an LTL formula over $\Pi$, and $\mathcal{P}$ a hyperrectangular set of uncertain parameters
// output: V a list of valid parameter sets
global $f, \Pi, \phi, \mathcal{P}, V, g, \Psi$
$V := \emptyset$
$(f, \Pi, \phi, \mathcal{P}) :=$ READ_DATA()
$(\Psi, g) :=$ COMPUTE_PARAMETER_CONSTRAINTS()
TEST_PARAMETER_SET($\epsilon$)                                    // $\epsilon$: Boolean of length 0
DISPLAY_RESULTS($V$)
```

Algorithm 2 Preprocessing step: computation of Ψ and g

```
function COMPUTE_PARAMETER_CONSTRAINTS()
// input: the global variables $f, \Pi$ and $\mathcal{P}$
// output: $\Psi$ a list of parameter constraints and $g$ a function giving for every pair of adjacent rectangles the set of parameters
// in $\mathcal{P}$ for which there is a transition between the rectangles
    $\Psi := \emptyset$; $g := \emptyset$
    for all $R \in \mathcal{R}$, $v \in \mathcal{V}_R$, and $i \in \{1, \ldots, n\}$ do
        $\Psi := \Psi \cup f_i(v, p)$
    end for
    REMOVE_REPEATED_ELEMENT_AND_SORT()                                // modifies elements in $\Psi$
    for all $R, R' \in \mathcal{R}$ such that $R \Bumpeq R'$ do                  // for all pairs of adjacent rectangles
        $g(R, R') := \{p \in \mathcal{P} \mid \exists v \in \mathcal{V}_R \cap \mathcal{V}_{R'}$ such that $f_i(v, p)(c'_i - c_i) > 0\}$   // with $c = coord(R)$, $c' = coord(R')$
    end for
    return $\Psi$ and $g$
end function
```

Algorithm 3 Recursive search for valid parameter sets

```
procedure TEST_PARAMETER_SET($b$)
// input: $b$ a Boolean and the global variables $f, \Pi, \phi, \mathcal{P}, g$ and $\Psi$
// modifies $V$ the list of valid parameter sets
    $P_b$:= COMPUTE_PARAMETER_SET($b$)
    $(T^{\exists}_{\mathcal{R}}(P_b), T^{\forall}_{\mathcal{R}}(P_b))$:= COMPUTE_TRANSITION_SYSTEMS($P_b$)
    if $T^{\exists}_{\mathcal{R}}(P_b) \models \phi$ then
        $V := V \cup P_b$;  return                          // $P_b$ is a valid parameter set; backtracking
    else if $T^{\forall}_{\mathcal{R}}(P_b) \not\models \phi$ then
        return                                                          // backtracking
    else                                                                // recursive analysis
        TEST_PARAMETER_SET($b.0$)                                // dot denotes concatenation
        TEST_PARAMETER_SET($b.1$)
    end if
end procedure
```

a key role when exploring large parameter spaces where only small regions are valid sets.

A number of dynamical properties can be tested this way for robustness analysis or tuning of gene networks. 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 Section V, this problem is due to the presence of spurious, time-converging executions in the abstract transition systems.

V. Enforcing Progress of Time in Discrete Abstractions

The analysis of counterexamples returned by model checkers reveals why the verification of liveness properties generally fails. For example, the execution $e_{\mathcal{R}1} = (R_{33}, R_{33}, R_{33}, \ldots)$ of $T_{\mathcal{R}}(p^1)$ [Fig. 3(b)], is a counterexample of the liveness property ϕ_2 [Equation (2)] expressing mutual exclusion. However, from the sketch of the flow in Fig. 3(a), it is intuitively clear that the system leaves R_{33} in finite time. Consequently, the execution $e_{\mathcal{R}1}$ that describes a system remaining always in R_{33} conflicts with the requirement that time progresses without upper bound. Such executions are called *time-converging* [10], [31].[4] Because they do not represent genuine behaviors of the system, these executions should be excluded when checking the properties of the system.

A. Ruling Out Time-Converging Executions Using Transient Regions

First, we define time-diverging executions in the transition systems $T_{\mathcal{X}}(p)$ and $T_{\mathcal{R}}(p)$.

Definition 10: 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 $\lim_{i\to\infty} \tau_i = \infty$.

[4] Time-converging executions are sometimes called Zeno executions [10], [31]. However, we prefer the former term since the latter is also used in a more restricted sense [32].

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 $i \geq 0$.

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 [33], [34]. $e_{\mathcal{R}1} = (R_{33}, R_{33}, R_{33}, \ldots)$ is an example of a time-converging execution of $T_{\mathcal{R}}(p^1)$ [Fig. 3(a) and (b)].

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

Definition 11: Let $P \subseteq \mathcal{P}$.

An execution $e_{\mathcal{R}}$ of $T^{\exists}_{\mathcal{R}}(P)$ 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 $T^{\forall}_{\mathcal{R}}(P)$ 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. For example, as suggested by the sketch of the flow in Fig. 3(a) and proved later, R_{33} is a transient region for parameter p^1.

Definition 12: 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 $\xi(\tau) \notin \bar{U}$.

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 $T_{\mathcal{R}}(p), T^{\exists}_{\mathcal{R}}(P)$, or $T^{\forall}_{\mathcal{R}}(P)$, and $e_{\mathcal{R}}$ being an execution of $T_{\mathcal{R}}$, we denote by $\mathrm{SCC}(e_{\mathcal{R}}) \subseteq \mathcal{X}$ 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, it represents at least a solution trajectory on a time interval $[0, \infty)$ (Definition 10), then $\mathrm{SCC}(e_{\mathcal{R}})$ can not be a transient region. Proposition 7 captures this intuition and establishes a link between time-diverging executions and transient regions.

Proposition 7: Let $p \in \mathcal{P}$. If an execution $e_{\mathcal{R}}$ of $T_{\mathcal{R}}(p)$ is time-diverging, then $\mathrm{SCC}(e_{\mathcal{R}})$ 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 $\mathrm{SCC}(e_{\mathcal{R}})$, there exists $i \geq 0$ such that, for every $j \geq i, R^j \subseteq \mathrm{SCC}(e_{\mathcal{R}})$. Let $e'_{\mathcal{R}} = (R^i, R^{i+1}, \ldots)$ be a suffix of $e_{\mathcal{R}}$ and $U = \cup_{j \geq i} R^j \subseteq \mathrm{SCC}(e_{\mathcal{R}})$. It holds that $e'_{\mathcal{R}}$ is a time-diverging execution of $T_{\mathcal{R}}(p)$. By Definition 10, 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 10, 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, $\forall t \geq 0, \xi(t) \in \bar{U}$ because every rectangle visited by $\xi(t)$ is necessarily in U (Definitions 4 and 7). Consequently U is not transient for p (Definition 12). Because $U \subseteq \mathrm{SCC}(e_{\mathcal{R}})$, the same necessarily holds for $\mathrm{SCC}(e_{\mathcal{R}})$. ■

Consider again the execution $e_{\mathcal{R}1} = (R_{33}, R_{33}, R_{33}, \ldots)$ of $T_{\mathcal{R}}(p^1)$ [Fig. 3(b)]. As mentioned earlier, $\mathrm{SCC}(e_{\mathcal{R}1}) = R_{33}$ is a transient region for parameter p^1. By Proposition 7, $e_{\mathcal{R}1}$ is consequently time-converging for p^1.

The following property is a generalization of Proposition 7.

Proposition 8: Let $P \subseteq \mathcal{P}$.

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

Proof: First, note that we cannot use directly Proposition 7, since by definition $\mathrm{SCC}(e_{\mathcal{R}})$ 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 7, the existence of a set U included in $\mathrm{SCC}(e_{\mathcal{R}})$ and nontransient for some (respectively, 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 Propositions 7 or 8, 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 predicate $\pi =$"transient" that is true for all and only rectangles R in transient SCCs. Then, instead of testing whether

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

we test whether

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

The executions of $T_{\mathcal{R}}$ satisfying **FG** ("transient") necessarily remain in a transient SCC and are consequently time-converging (Proposition 7 or 8). So, only time-converging executions are ruled out this way. However, because Propositions 7 and 8 give only necessary conditions for an execution to be time-diverging, not all time-converging executions are guaranteed to be ruled out. Using these results, we propose a modified version of Algorithm 3 (Algorithm 4).

Consider again our example network. As said earlier, R_{33} is a transient region. Because R_{33} forms a (trivial) SCC, it is labeled "transient" in $T_{\mathcal{R}}(p^1)$. Then, the execution $e_{\mathcal{R}1} = (R_{33}, R_{33}, R_{33}, \ldots)$, satisfying **FG** ("transient"), is not a counterexample of ϕ'_2 and will not cause the property to be falsely invalidated anymore.

Algorithm 4 Recursive search for valid parameter sets with detection of transient regions

```
procedure TEST_PARAMETER_SET($b$)
// input: $b$ a Boolean and the global variables $f, \Pi, \phi, \mathcal{P}, g$ and $\Psi$
// modifies $V$ the list of valid parameter sets
    $P_b$:= COMPUTE_PARAMETER_SET($b$)
    $(T^{\exists}_{\mathcal{R}}(P_b), T^{\forall}_{\mathcal{R}}(P_b))$ := COMPUTE_TRANSITION_SYSTEMS($P_b$)
    $(scc\_list^{\exists}, scc\_list^{\forall})$ := COMPUTE_SCCS($T^{\exists}_{\mathcal{R}}(P_b), T^{\forall}_{\mathcal{R}}(P_b)$)      // $scc\_list$: list of strongly connected components
    $(\phi'^{\exists}, \phi'^{\forall})$ := UPDATE_WITH_TRANSIENT_REGIONS($P_b, scc\_list^{\exists}, scc\_list^{\forall}$)      // tests each SCC for being transient
                                                                      // and modifies LTL property accordingly
    if $T^{\exists}_{\mathcal{R}}(P_b) \models \phi'^{\exists}$ then
        $V := V \cup P_b$; return                                      // $P_b$ is a valid parameter set; backtracking
    else if $T^{\forall}_{\mathcal{R}}(P_b) \not\models \phi'^{\forall}$ then
        return                                                          // backtracking
    else                                                                // recursive analysis
        TEST_PARAMETER_SET($b$.0)
        TEST_PARAMETER_SET($b$.1)
    end if
end procedure
```

B. Transient Region Computation for Uncertain 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 uncertain 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 9), for some parameter (Proposition 11), or for all parameters in a bounded polyhedral set (Proposition 10). 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 the previous section. More precisely, Propositions 9, 10 and 11 are used in combination with (the contrapositive of) Propositions 7, 8(1), and 8(2), 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 9: Let $p \in \mathcal{P}$ and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. If

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

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 $0 \notin \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$. Using the separating hyperplane theorem, there exists $\alpha \in \mathbb{R}^n$ such that for all $z \in \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R, R \subseteq U\}), \alpha^T z > 0$. For every rectangle $R \subseteq U$, $f(x,p)$ is a multiaffine function of x on $\bar{R}$, so it holds that, for every $x \in \bar{R}$, $f(x,p) \in \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R\})$ (Theorem 2). Then, for every $x \in \bar{U}, f(x,p) \in \cup_{R \subseteq U} \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R\})$, which is included in $\mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$. Consequently, $\alpha^T f(x,p) > 0$. Since $\bar{U}$ is compact (union of compact sets $\bar{R}$) and f is continuous, $\alpha^T f(\bar{U},p)$ 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, $\bar{U}$ is left in finite time. ■

The conditions of the above property are satisfied by R_{33}, which proves that this region is transient, as hypothesized earlier. We illustrated our approach on a trivial SCC (i.e., containing a single rectangle, R_{33}), but note that Proposition 9 (and 10 and 11) applies also to nontrivial SCCs.

Propositions 10 and 11 are generalizations of Proposition 9 to polyhedral parameter sets.

Proposition 10: Let $P \subseteq \mathcal{P}$ be a polytope and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. If $0 \notin \mathrm{hull}(\{f(v,w) \,|\, v \in \mathcal{V}_R, R \subseteq U, w \in \mathcal{V}_P\})$, then U is transient for all parameters $p \in P$.

Proof: Using Proposition 9, we only have to prove that, if $0 \notin \mathrm{hull}(\{f(v,w) \,|\, v \in \mathcal{V}_R, R \subseteq U, w \in \mathcal{V}_P\})$, then $\forall p \in P, 0 \notin \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$. We prove its contrapositive. Let $p \in P$ be such that $0 \in \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$. Then, since f is affine in p, by Theorem 1 it holds that $0 \in \mathrm{hull}(\{\mathrm{hull}(\{f(v,w) \,|\, w \in \mathcal{V}_P\}) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$ or, more simply $0 \in \mathrm{hull}(\{f(v,w) \,|\, v \in \mathcal{V}_R, R \subseteq U, w \in \mathcal{V}_P\})$. ■

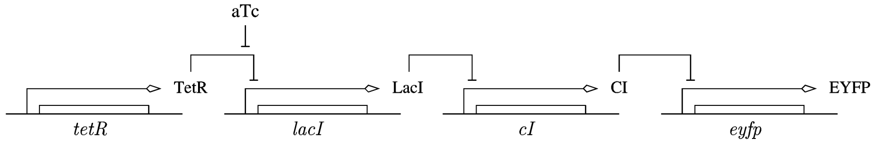
Proposition 11: Let $P \subseteq \mathcal{P}$ be a polytope and $U \subseteq \mathcal{X}$ be a union of rectangles $R \in \mathcal{R}$. If, for some $w \in \mathcal{V}_P$, $0 \notin \mathrm{hull}(\{f(v,w) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$, then U is transient for some parameters $p \in P$.

By Proposition 9, Proposition 11 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 $0 \notin \mathrm{hull}(\{f(v,w) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$ 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 12: 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 $0 \notin \mathrm{hull}(\{f(v,p) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$ iff $\exists w \in \mathcal{V}_P$ such that $0 \notin \mathrm{hull}(\{f(v,w) \,|\, v \in \mathcal{V}_R, R \subseteq U\})$.

Proof: The necessity is trivial. We prove sufficiency by contradiction. Let $p \in P$ and let I and J be two sets of indexes labeling the vertices in $\cup_{R \subseteq U} \mathcal{V}_R$ and $\mathcal{V}_P$: $\cup_{R \subseteq U} \mathcal{V}_R =$

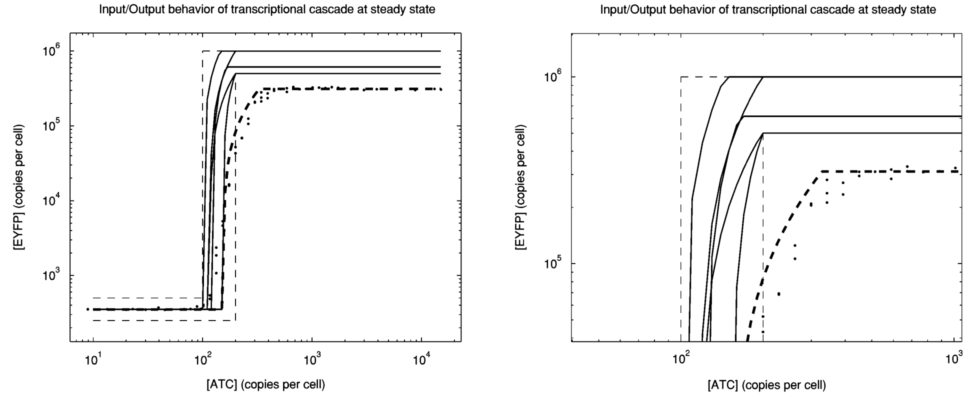
(a)



(b)

$$\begin{aligned}
&\dot{x}_{tetR} = \kappa_{tetR} - \gamma_{tetR}\, x_{tetR}, && (1')\\
&\dot{x}_{lacI} = \kappa^0_{lacI} + \kappa_{lacI}(1 - r^+(x_{tetR}, \theta^1_{tetR}, \theta^2_{tetR})\, r^-(u_{aTc}, \theta^1_{aTc}, \theta^2_{aTc})) - \gamma_{lacI}\, x_{lacI}, && (2')\\
&\dot{x}_{cI} = \kappa^0_{cI} + \kappa_{cI}\; r^-(x_{lacI}, \theta^1_{lacI}, \theta^2_{lacI}) - \gamma_{cI}\, x_{cI}, && (3')\\
&\dot{x}_{eyfp} = \kappa^0_{eyfp} + \kappa_{eyfp}\; r^-(x_{cI}, \theta^1_{cI}, \theta^2_{cI}) - \gamma_{eyfp}\, x_{eyfp}, && (4')
\end{aligned}$$

$(\theta^1_{aTc}, \theta^2_{aTc}) = (80, 4000)$; $(\kappa_{tetR}, \gamma_{tetR}, \theta^1_{tetR}, \theta^2_{tetR}) = (260, 0.013, 4500, 5500)$;
$(\kappa^0_{lacI}, \kappa_{lacI}, \gamma_{lacI}, \theta^1_{lacI}, \theta^2_{lacI}) = (2.40, 875.6, 0.013, 500, 4500)$; $(\kappa^0_{cI}, \kappa_{cI}, \gamma_{cI}, \theta^1_{cI}, \theta^2_{cI}) = (3.90, 386, 0.013, 600, 23000)$;
$(\kappa^0_{eyfp}, \kappa_{eyfp}, \gamma_{eyfp}) = (4.58, 4048, 0.013)$



(c) (d)

Fig. 6. (a) Synthetic transcriptional cascade made of four genes. $tetR$ inhibits $lacI$, $lacI$ inhibits cI, and cI inhibits $eyfp$. The input aTc relieves the inhibition of $lacI$ expression by TetR. The fluorescence of the protein EYFP is the output. (b) PMA model. The notations follow those introduced in Section III. Equation ($2'$) states that $lacI$ is repressed when the protein TetR is present and aTc absent. Parameter values estimated for the actual cascade from experimental data in [40] are indicated. (c) I/O response of the cascade at steady state [zoomed in (d)]. Measured (dots), predicted (thick dashed line), and desired (region delimited by dashed lines) behaviors of the actual network. Predicted (solid lines) behaviors for different parameters in the set P^*.

$\{v_i\}_{i\in I}$ 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

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

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

From a computational point of view, it is important to note that the conditions in Propositions 9–11, can be simply evaluated by polyhedral operations. Moreover, as noted in [35], this problem can be reformulated as an equivalent linear optimization problem and solved more efficiently using linear programming.

The methods described in Sections IV and V have been implemented in a tool for RoVerGeNe. The tool is freely available and can be downloaded from [36]. It consists of approximatively 2000 lines of Matlab (The MathWorks, Inc.) code and exploits the Multi-Parametric Toolbox [37] for polyhedral operations and linear programming, the Matlab Boost Graph Library [38] for SCC computations, and the CTL/LTL model-checker NuSMV [39]. Given a PMA model, an LTL specification of the property and an initial parameter set, the tool can either test whether the given parameter set is valid (Problem 1.A), or recursively search for valid parameter subsets (Problem 1.B). Valid parameter sets are returned as a list of polytopes.

VI. Tuning of a Transcriptional Cascade

A. Modeling and Specification

The method presented in the previous section is illustrated by the analysis of the steady-state input/output (I/O) behavior of a synthetic transcriptional cascade built and analyzed in [40] [Fig. 6(a)]. [5] Because of the topology of the network (cascade of inhibitions), an ultrasensitive response might be achieved: the output (EYFP) at steady state undergoes a dramatic change for a moderate change of the input (aTc) in a transition region. More

[5] An improved version of this model and additional computational results can be found in [41].

precisely, the desired behavior is to obtain at least a 1000-fold increase of the output value for a twofold increase of the input value [Fig. 6(c)]. Using $\mathbf{FG}p$ ("eventually, p will be always true") to express that property p holds at equilibrium, the specifications in Fig. 6(c) can be translated to LTL as follows:

$$\begin{aligned}\phi_3 = u_{aTc} < 100 &\rightarrow \mathbf{FG}(2.5\,10^2 < x_{eyfp} < 5\,10^2)\\ \wedge\, 100 < u_{aTc} < 200 &\rightarrow \mathbf{FG}(2.5\,10^2 < x_{eyfp} < 10^6)\\ \wedge\, u_{aTc} > 200 &\rightarrow \mathbf{FG}(5\,10^5 < x_{eyfp} < 10^6).\end{aligned}$$

The actual network does not meet these specifications. So, our goal is to tune the network by finding valid parameter sets for property ϕ_3 (Problem 1.B). Additionally, because it is important that the network behaves as expected despite environmental fluctuations, we would like to test using our model if the tuned network will present a robust behavior, before actually experimentally tuning it. More specifically, we would like to verify that the tuned cascade satisfies specifications ϕ_3 for *all* production and degradation rate parameters varying in $\pm 10\%$ (or $\pm 20\%$) intervals centered at their reference values (Problem 1.A).

To do so, we have developed a PMA model of this system, represented in Fig. 6(b). Parameter values were estimated based on experimental data available in [40]. Note that ϕ_3 is a liveness property. The use of the approach described in Section V for enforcing progress of time was necessary in this case, since no conclusion could have been obtained otherwise. Also, 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 (nontransient) *rectangle*, instead of *SCC* in the general case (see Proposition 8). We have consequently applied Propositions 10 and 11 to rectangles only, to obtain tighter predictions.

B. Tuning and Robustness Analysis

Using RoVerGeNe, we found a valid set P^* by tuning three production rate parameters, $\kappa_{lacI}, \kappa_{cI}$ and κ_{eyfp}

$$\begin{aligned}P^* = \{p = (\kappa_{lacI}, \kappa_{cI}, \kappa_{eyfp}) \,|\, 1832.43 < \kappa_{lacI} < 3350.62,\\ 393.46 < \kappa_{cI}, \text{ and } 6495.42 < \kappa_{eyfp} < 12995.42\}.\end{aligned}$$

These results essentially suggest to increase the production rates of LacI and EYFP by a factor 2 to 3. This could be achieved for example by tuning promoter or ribosome binding site efficiencies (see, e.g., [42] and [43]). In order to evaluate the significance of the above constraints, we computed by numerical simulation the steady-state I/O behavior of the system for different parameters in P^*, notably using extreme values [Fig. 6(c)]. This clearly revealed that relevant constraints on the parameters have been identified by our method.

With a partition of the state space having 1500 rectangles, 18 affine predicates on parameters were found, defining $> 200\,000$ parameter equivalence classes. The computation lasted < 2 hours (PC, 3.4-GHz processor, and 1-Gb RAM) and only 350 different parameter sets were analyzed using the hierarchical approach. This computational time can be considered as very reasonable, given the difficulty of the problem: we systematically explored a three-dimensional (3-D) parameter space, testing a nontrivial dynamical property for any initial condition in a five-dimensional (5-D) (one input and four state variables) state-space.

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

Fig. 7. Computational time for the verification of a liveness property as a function of the number of variables and uncertain parameters. The 3-D and 4-D systems correspond to similar but shorter transcriptional cascades (see [40]). NA: not applicable.

Any parameter in P^* is valid. However, it is not guaranteed that the behavior of the network is *robust* for any parameter in P^*. To test the robustness of the tuned network, we have selected a valid parameter p^* in P^* and tested whether the system satisfies the expected property ϕ_3 for all of the 11 production and degradation rate parameters varying in $\pm 10\%$ intervals centered at their reference values. We used $p^* = (\kappa_{lacI}, \kappa_{cI}, \kappa_{eyfp}) = (2591, 550, 8000)$, and the values given in Fig. 6(b) for the other parameters.

Using RoVerGeNe, we have been able to prove the robustness of the property in < 4 h. Given that the problem was to prove that a non-trivial property holds for every initial condition in a 5-D state space 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 Fig. 7.

The same test has been performed for $\pm 20\%$ parameter variations, and a negative answer was obtained (< 4 h). We recall that, from negative answers, one cannot conclude that the property is false for some parameters in the set. Nevertheless, the analysis of the counterexample given by the model checker has revealed that the system can remain in a (nontransient) 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. Again, this illustrates that relevant constraints on parameters were identified by our approach.

VII. Discussion

We have presented a method for the analysis of genetic regulatory networks under parameter uncertainty. Given a PMA model, a dynamical property expressed in temporal logic and a bounded polyhedral parameter set, the proposed approach can be used to test whether the property is satisfied for every parameter in the parameter set—the set is then called valid—or to find valid subsets of the given parameter set. The practical applicability and biological relevance of our approach has been demonstrated on the analysis of the tuning of a synthetic network built in *E. coli*. Network tuning is a central problem in synthetic biology, since most initial attempts at constructing gene networks do not result in a system exhibiting the desired behavior [4].

In comparison with other modeling frameworks proposed to describe gene networks (reviewed in [44]), uncertain PMA models can be considered as an intermediate formalism between exact, quantitative differential equation models such as mass-action kinetics or Hill-type models [45], and coarse-grained, qualitative models such as qualitative PA differential equation models [17]. The analysis of exact quantitative models, essentially by numerical simulation since analytical solutions seldom exist, yield precise predictions on the network behavior. However because of large parameter uncertainties, these results are often of limited validity. In contrast, qualitative models yield coarse predictions on the network behavior, but that hold for large sets of parameters. The class of uncertain PMA models combines advantages from both formalisms in the sense that one obtains predictions that are finer than using qualitative models and more robust than using exact quantitative models.

We briefly review the two main methodological contributions of this work, dealing with the analysis of uncertain PMA systems (Section IV) and with the verification of liveness properties of dynamical systems with dense-time semantics (Section V). First, we use a discrete abstraction $T_{\mathcal{R}}(p)$ of the continuous dynamical system $T_{\mathcal{X}}(p)$ to define equivalence classes on parameters, in the sense that two equivalent parameters are associated to the same discrete abstraction. Then, we define discrete transition systems, $T_{\mathcal{R}}^{\exists}(P)$ and $T_{\mathcal{R}}^{\forall}(P)$, that over- and under-approximate the discrete abstraction $T_{\mathcal{R}}(p)$ for every $p \in P$ and show how they can be used to search the parameter space efficiently by model checking. The proposed approach is conservative: if a parameter set is found, it is guaranteed to be valid. However, not all valid parameter sets are guaranteed to be found. Second, we propose a method to rule out time-converging behaviors. These behaviors along which time does not progress are introduced in the discrete abstractions by the abstraction process, and cause the verification of liveness properties to fail. 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 strongly connected components of the discrete abstractions: executions that remain in a transient SCC are necessarily time-converging. Then, we provide sufficient conditions for the identification of transient regions in uncertain PMA systems. These two methods are integrated in a unique algorithm, implemented in the publicly available tool for robust verification of gene networks, RoVerGeNe.

Other approaches have been proposed for the verification of continuous or hybrid systems with parameter uncertainty in the hybrid systems community. In most approaches, unknown parameters are represented as symbolic constants, and symbolic operations are used to manipulate sets of states and compute (approximations of) sets of predecessors or successors [30], [46]–[49]. A major limitation is that the computational techniques supporting these symbolic operations currently apply only to systems having rather simple continuous dynamics, such as timed automata [46], [47], linear hybrid automata [48], piecewise-affine systems [17], [30] (see also [50], [51] for related, purely discrete approaches), or affine hybrid automata [49]. In particular, these methods cannot deal with the multiaffine nonlinearities that appear when modeling complex genetic regulations with graded response, as we do in the transcriptional cascade example. Alternatively, numerical approaches have been proposed in which parameter uncertainties are captured by means of differential inclusions (e.g., $\dot{x} \in \mathrm{hull}(\{f(x,p) \,|\, p \in P\})$) [52]. For large parameter sets, these approaches can be very conservative. In this paper, we propose an approach which is successively symbolic (synthesis of parameter constraints) and numerical (computation of transition systems). The results of the first step are used to refine the parameter sets considered in the second step, in order to limit (though not eliminate) overconservatism, while preserving efficiency.

In comparison with the amount of work done for the verification of safety properties of continuous or hybrid systems having dense-time semantics, not much work has been done for liveness properties [31]. It has been proposed that the difficulty to enforce progress of time in dense-time systems makes liveness properties comparatively more difficult to analyze [31]. Tools supporting the verification of true (i.e., unbounded) liveness properties of dense-time systems are Uppaal [53], TReX [33] and RED [31]. However, the applicability of existing methods is again limited to systems that have very restricted continuous dynamics, namely timed automata. In contrast, our approach applies to any discrete abstraction provided that transient regions can be characterized. As mentioned in Section V-A, a similar problem arise in untimed systems for the verification of liveness properties when abstractions are used [33], [34]. 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 $\neg\mathbf{FG}$ ("transient") $(= \mathbf{GF}(\neg$"transient"$)$, Section V.A) 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.

Acknowledgment

The authors would like to thank B. Yordanov for contributions to the development of earlier versions of the mathematical model of the transcriptional cascade.

References

[1] G. Batt, C. Belta, and R. Weiss, "Model checking genetic regulatory networks with parameter uncertainty," in *Proceedings of the Tenth International Workshop Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, A. Bemporad, A. Bicchi, and G. Buttazzo, Eds. New York: Springer, 2007, vol. 4416, pp. 61–75.

[2] G. Batt, C. Belta, and R. Weiss, "Model checking liveness properties of genetic regulatory networks," in *Proceedings of the Thirteenth International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, ser. Lecture Notes in Computer Science, O. Grumberg and M. Huth, Eds. New York: Springer, 2007, vol. 4424, pp. 323–338.

[3] H. Kitano, "Systems biology: A brief overview," *Science*, vol. 295, no. 5560, pp. 1662–1664, 2002.

[4] E. Andrianantoandro, S. Basu, D. Karig, and R. Weiss, "Synthetic biology: New engineering rules for an emerging discipline," *Mol. Syst. Biol.*, vol. 2, p. 28, 2006.

[5] Z. Szallasi, J. Stelling, and V. Periwal, Eds., *System Modeling in Cellular Biology: From Concepts to Nuts and Bolts*. Cambridge, MA: MIT Press, 2007.

[6] H. de Jong and D. Ropers, "Qualitative approaches to the analysis of genetic regulatory networks," in *System Modeling in Cellular Biology: From Concepts to Nuts and Bolts*, Z. Szallasi, J. Stelling, and V. Periwal, Eds. Cambridge, MA: MIT Press, 2006, pp. 125–148.

[7] C. Belta and L. C. G. J. M. Habets, "Controlling a class of nonlinear systems on rectangles," *IEEE Trans. Autom. Control*, vol. 51, no. 11, pp. 1749–1759, Nov. 2006.

[8] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid systems," *Proc. IEEE*, vol. 88, no. 7, pp. 971–984, Jul. 2000.

[9] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model Checking*. Cambridge, MA: MIT Press, 1999.

[10] T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine, "Symbolic model checking for real-time systems," *Inf. Comput.*, vol. 111, pp. 193–244, 1994.

[11] S. Tripakis and S. Yovine, "Analysis of timed systems using time-abstracting bisimulations," *Form. Method Syst. Des.*, vol. 18, no. 1, pp. 25–68, 2001.

[12] B. Alpern and F. B. Schneider, "Recognizing safety and liveness," *Distrib. Computing*, vol. 2, no. 3, pp. 117–126, 1986.

[13] D. Park, "Concurrency and automata on infinite sequences," in *Proceedings of the Fifth GI Conference on Theoretical Computer Science*, ser. Lecture Notes in Computer Science, P. Deussen, Ed. New York: Springer, 1981, vol. 104, pp. 167–183.

[14] E. A. Emerson, "Temporal and modal logic," in *Handbook of Theoretical Computer Science*, J. van Leeuwen, Ed. Cambridge, MA: MIT Press, 1990, vol. B, Formal Models and Sematics, pp. 995–1072.

[15] M. C. Browne, E. M. Clarke, and O. Grumberg, "Characterizing finite Kripke structures in propositional temporal logic," *Theoret. Comput. Sci.*, vol. 59, pp. 115–131, 1988.

[16] L. C. G. J. M. Habets, P. J. Collins, and J. H. van Schuppen, "Reachability and control synthesis for piecewise-affine hybrid systems on simplices," *IEEE Trans. Autom. Control*, vol. 51, no. 6, pp. 938–948, Jun. 2006.

[17] H. de Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann, "Qualitative simulation of genetic regulatory networks using piecewise-linear models," *Bull. Math. Biol.*, vol. 66, no. 2, pp. 301–340, 2004.

[18] G. Batt and C. Belta, "Model checking genetic regulatory networks with applications to synthetic biology," Boston Univ., Boston, MA, Tech. Rep. 2006-IR-0030, 2006.

[19] A. Abate and A. Tiwari, "Box invariance of hybrid and switched systems," in *Proceedings of the Second Conference on Analysis and Design of Hybrid Systems*. New York: Elsevier, 2006, pp. 359–364.

[20] C. Belta, L. C. G. J. M. Habets, and V. Kumar, "Control of multiaffine systems on rectangles with applications to hybrid biomolecular networks," in *Proceedings of the 41st IEEE Conference on Decision Control*, 2002, pp. 534–539.

[21] T. Mestl, E. Plahte, and S. W. Omholt, "A mathematical framework for describing and analysing gene regulatory networks," *J. Theor. Biol.*, vol. 176, no. 2, pp. 291–300, 1995.

[22] L. Glass and S. A. Kauffman, "The logical analysis of continuous nonlinear biochemical control networks," *J. Theor. Biol.*, vol. 39, no. 1, pp. 103–129, 1973.

[23] S. A. Kauffman, "Homeostasis and differentiation in random genetic control networks," *Nature*, vol. 224, pp. 177–178, 1969.

[24] R. Thomas, D. Thieffry, and M. Kaufman, "Dynamical behaviour of biological regulatory networks: I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state," *Bull. Math. Biol.*, vol. 57, no. 2, pp. 247–276, 1995.

[25] B. C. Goodwin, "Oscillatory behavior in enzymatic control processes," in *Advances in Enzyme Regulation*, G. Weber, Ed. Oxford, U.K.: Pergamon, 1965, pp. 425–438.

[26] G. Yagil and E. Yagil, "On the relation between effector concentration and the rate of induced enzyme synthesis," *Biophys. J.*, vol. 11, no. 1, pp. 11–27, 1971.

[27] M. Kloetzer and C. Belta, "Reachability analysis of multiaffine systems," in *Proceedings of the Ninth International Workshop on Hybrid Systems : Computational Control*, ser. Lecture Notes in Computer Science, J. P. Hespanha and A. Tiwari, Eds. New York: Springer, 2006, vol. 3927, pp. 348–362.

[28] R. Alur, T. Dang, and F. Ivancic, "Progress on reachability analysis of hybrid systems using predicate abstraction," in *Proceedings of the Sixth International Workshop on Hybrid Systems : Computational Control*, ser. Lecture Notes in Computer Science, A. Pnueli and O. Maler, Eds. New York: Springer, 2003, vol. 2623, pp. 4–19.

[29] X. D. Koutsoukos and P. J. Antsaklis, "Safety and reachability of piecewise linear hybrid dynamical systems based on discrete abstractions," *J. Discr. Event Dynam. Syst.*, vol. 13, no. 3, pp. 203–243, 2003.

[30] G. Batt, D. Ropers, H. de Jong, J. Geiselmann, R. Mateescu, M. Page, and D. Schneider, "Validation of qualitative models of genetic regulatory networks by model checking: Analysis of the nutritional stress response in *Escherichia coli*," *Bioinformatics*, vol. 21, no. suppl. 1, pp. i19–i28, 2005.

[31] F. Wang, G.-D. Huang, and F. Yu, "TCTL inevitability analysis of dense-time systems: From theory to engineering," *IEEE Trans. Softw. Eng.*, vol. 32, no. 7, pp. 510–526, 2006.

[32] J. Lygeros, K. H. Johansson, S. N. Simić, J. Zhang, and S. S. Sastry, "Dynamical properties of hybrid automata," *IEEE Trans. Autom. Control*, vol. 48, no. 1, pp. 2–17, Jan. 2003.

[33] A. Bouajjani, A. Collomb-Annichini, Y. Lacknech, and M. Sighireanu, "Analysis of fair parametric extended automata," in *Proceedings of the Eighth International Symposium on Static Analysis*, ser. Lecture Notes in Computer Science, P. Cousot, Ed. New York: Springer, 2001, vol. 2126, pp. 335–355.

[34] D. Dams, R. Gerth, and O. Grumberg, "A heuristic for the automatic generation of ranking functions," in *Proceedings of the Workshop on Advances in Verification*, ser. Lecture Notes in Computer Science, G. Gopalakrishnan, Ed. New York: Springer, 2000, pp. 1–8.

[35] K. Fukuda, "Frequently asked questions in polyhedral computation," ETH, Zurich, Switzerland, 2004 [Online]. Available: http://www.ifor.math.ethz.ch/~fukuda/polyfaq/polyfaq.html

[36] G. Batt and C. Belta, "Robust Verification of Gene Networks (RoVerGeNe)," Boston Univ., Boston, MA, 2006 [Online]. Available: http://iasi.bu.edu/~batt/rovergene/rovergene.htm

[37] M. Kvasnica, P. Grieder, and M. Baotić, "Multi-Parametric Toolbox (MPT)," ETH, Zurich, Switzerland, 2004 [Online]. Available: http://control.ee.ethz.ch/~mpt/

[38] D. Gleich, "Matlab Boost Graph Library (MatlabBGL)," Stanford Univ., Stanford, CA, 2006 [Online]. Available: http://www.stanford.edu/~dgleich/programs/matlab_bgl/

[39] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "NuSMV2: An opensource tool for symbolic model checking," in *Proceedings of the Fourteenth International Conference on Computer-Aided Verification*, ser. Lecture Notes in Computer Science, E. Brinksma and K. Larsen, Eds. New York: Springer, 2002, vol. 2404, pp. 359–364.

[40] S. Hooshangi, S. Thiberge, and R. Weiss, "Ultrasensitivity and noise propagation in a synthetic transcriptional cascade," *Proc. Natl. Acad. Sci. USA*, vol. 102, no. 10, pp. 3581–3586, 2005.

[41] G. Batt, B. Yordanov, R. Weiss, and C. Belta, "Robustness analysis and tuning of synthetic gene networks," *Bioinformatics*, vol. 23, no. 18, pp. 2415–2422, 2007.

[42] K. Hammer, I. Mijakovic, and P. R. Jensen, "Synthetic promoter libraries—Tuning of gene expression," *Trends Biotechnol.*, vol. 24, no. 2, pp. 53–55, 2006.

[43] S. Basu, R. Mehreja, S. Thiberge, M.-T. Chen, and R. Weiss, "Spatiotemporal control of gene expression with pulse-generating networks," *Proc. Natl. Acad. Sci. USA*, vol. 101, no. 17, pp. 6355–6360, 2004.

[44] H. de Jong, "Modeling and simulation of genetic regulatory systems: A literature review," *J. Computat. Biol.*, vol. 9, no. 1, pp. 69–105, 2002.

[45] A. Cornish-Bowden, *Fundamentals of Enzyme Kinetics*. London, U.K.: Portland Press, 1995.

[46] A. Annichini, E. Asarin, and A. Bouajjani, "Symbolic techniques for parametric reasoning about counter and clock systems," in *Proceedings of the Twelfth International Conference on Computer-Aided Verification*, ser. Lecture Notes in Computer Science, E. A. Emerson and A. P. Sistla, Eds. New York: Springer, 2000, vol. 1855, pp. 419–434.

[47] F. Wang, "Symbolic parametric safety analysis of linear hybrid systems with BDD-like data-structures," *IEEE Trans. Softw. Eng.*, vol. 31, no. 1, pp. 38–51, Jan. 2005.

[48] T. A. Henzinger, P. Ho, and H. Wong-Toi, "HYTECH: A model checker for hybrid systems," *Int. J. Softw. Tools Technol. Transfer*, vol. 1, no. 1–2, pp. 110–122, 1997.

[49] R. Ghosh and C. J. Tomlin, "Symbolic reachable set computation of piecewise affine hybrid automata and its application to biological modelling: Delta-Notch protein signalling," *Proc. IEE Syst. Biol.*, vol. 1, no. 1, pp. 170–183, 2004.

[50] G. Bernot, J.-P. Comet, A. Richard, and J. Guespin, "Application of formal methods to biological regulatory networks: Extending Thomas' asynchronous logical approach with temporal logic," *J. Theor. Biol.*, vol. 229, no. 3, pp. 339–347, 2004.

[51] L. Calzone, N. Chabrier-Rivier, F. Fages, and S. Soliman, "Machine learning biochemical networks from temporal logic properties," in *Transactions in Computer Systems Biology VI*, ser. Lecture Notes in Bioinformatics, C. Priami and G. Plotkin, Eds. New York: Springer, 2006, vol. 4220, pp. 68–94.

[52] H. Lin and P. J. Antsaklis, "Robust regulation of polytopic uncertain linear hybrid systems with networked control system applications," in *Stability and Control of Dynamical Systems with Applications*, P. J. Antsaklis and D. Liu, Eds. Basel, Switzerland: Birkhauser, 2003.

[53] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, Y. Wang, and C. Weise, "New generation of Uppaal," in *Proceedings of the International Workshop on Software Tools for Technology Transfer*, 1998, pp. 43–52.

Grégory Batt received the B.Sc. degree in molecular and cellular biology from Ecole Normale Supérieure, Lyon, France, the M.Sc. degree in computer science from the Joseph Fourier University (UJF), Grenoble, France, and the Ph.D. degree in computer science from UJF and INRIA Rhône-Alpes, Grenoble, France, in 2005.

He has continued his research on the interface of hybrid systems theory and systems biology with postdoctoral fellowships with Boston University, Boston, MA, and Verimag (CNRS, UJF), Grenoble, France. He is currently a Research Scientist with INRIA Paris, Rocquencourt, France.

Calin Belta received the B.Sc. and M.Sc. degrees in control and computer science from the Technical University of Iasi, Iasi, Romania, the M.Sc. degree in electrical engineering from Louisiana State University, Baton Rouge, and the M.Sc. and Ph.D. degrees in mechanical engineering from the University of Pennsylvania, Philadelphia.

He is currently an Assistant Professor with Boston University, Boston, MA. His research interests include verification and control of dynamical systems, motion planning and control, hybrid systems, and biomolecular networks.

Dr. Belta was the recipient of a National Science Foundation CAREER Award in 2005 and a Fulbright Study Award in 1997 and was the Valedictorian of his class in 1995. He was the recipient of the Best Paper Poster at the International Conference on Systems Biology in 2004 and was a finalist for the ASME Design Engineering Technical Conference Best Paper Award in 2002 and for the Anton Philips Best Student Paper Award at the IEEE International Conference on Robotics and Automation in 2001.

Ron Weiss received the Ph.D. degree from the Massachusetts Institute of Technology (MIT), Cambridge, in computer science and electrical engineering in 2001.

He is an Associate Professor of Electrical Engineering with Princeton University, Princeton, NJ, and holds a faculty appointment with the Department of Molecular Biology. His research focuses primarily on synthetic biology, where he programs cell behavior by constructing and modeling artificial genetic circuits.

Dr. Weiss was the recipient of MIT's Technology Review Magazine's TR100 Award ("Top 100 Young Innovators") in 2003, was selected as a speaker for the National Academy of Engineering's Frontiers of Engineering Symposium in 2003, received the E. Lawrence Keyes, Jr./Emerson Electric Company Faculty Advancement Award at Princeton University in 2003, was chosen as a finalist for the World Technology Network's Biotechnology Award (2004), was the recipient of the National Science Foundation CAREER Award in 2004, and was selected as a speaker for the National Academy of Sciences Frontiers of Science Symposium in 2005. His research in synthetic biology was named by MIT's Technology Review Magazine as one of "10 emerging technologies that will change your world" in 2004.