Formal Analysis of Piecewise Affine Systems through Formula-Guided Refinement

Boyan Yordanov, Jana Tůmová, Calin Belta, Ivana Černá, and Jiří Barnat

***Abstract*— We present a computational framework for identifying a set of initial states from which all trajectories of a piecewise affine (PWA) system satisfy a Linear Temporal Logic (LTL) formula over a set of linear predicates in its state variables. Our approach is based on the construction and refinement of finite abstractions of infinite systems (*i.e.* systems where states can take infinitely many values). We derive conditions guaranteeing the equivalence of an infinite system and its finite abstraction with respect to a specific temporal logic formula and propose methods aimed at the construction of such formula-equivalent abstractions. We show that the proposed procedure can be implemented using polyhedral operations and analysis of finite graphs. While provably correct, the overall method is conservative and expensive. The proposed algorithms have been implemented as a software tool that is available for download. Illustrative examples for the PWA models of two gene networks are included.**

I. Introduction

In control problems, trajectories of "complex" mathematical models of physical systems, such as systems of differential or difference equations, are usually checked against "simple" specifications, such as stability of equilibria and set invariance. In formal verification, "rich" specifications, such as formulas of temporal logics, are checked against "simple" models such as (finite) transition graphs and automata models of software programs and digital circuits [9]. The study of physical systems requires the development of theoretical frameworks and computational tools for bridging in this gap, and therefore allowing for specifying the properties of continuous and hybrid systems in a rich language, with automatic verification and controller synthesis. Recent results include temporal logics for systems with continuous dynamics [10], control of linear systems from temporal logic specifications [23], [18], task specification and controller synthesis in mobile robotics [20], [12], and specification and analysis of qualitative behavior of genetic circuits [2], [4].

In this paper, we focus on piecewise affine systems (PWA) that evolve along different discrete-time affine dynamics in different polytopic regions of the (continuous) state space. PWA systems are widely used as models in many areas. They can approximate nonlinear dynamics with arbitrary accuracy, and are equivalent with other classes of hybrid systems [16]. In addition, there exist techniques for the identification of such models from experimental data, which include Bayesian methods, bounded-error procedures, clustering-based methods, mixed-integer programming, and algebraic geometric methods (see [17] for a review).

We consider the following problem: given a PWA system and an arbitrary LTL formula over an arbitrary set of linear predicates in its state variables, find the largest region of initial states from which all trajectories of the system satisfy the formula. Our approach to this problem is based on the construction and iterative refinement of finite abstractions. The refinement is guided by formula equivalence, *i.e.,* at each iteration we aim at constructing a finite abstraction of the PWA system that satisfies exactly the same formula. To this goal, we use ideas from LTL model checking [9] and bisimulation - based refinement [5]. The construction of the abstractions is enabled by our previous results [25], where we showed that finite quotients of PWA systems can be constructed by using polyhedral operations only.

This work can be seen in the context of literature focused on the construction of finite quotients of infinite systems, and is related to [22], [23], [8]. The embedding of discrete-time systems into transition systems is inspired from [22], [23]. However, while the focus there is on characterizing the existence of bisimulation quotients or developing control strategies using such quotients for linear systems, in this work we consider an analysis problem and focus on the computation of finite quotients, which are equivalent to the original, infinite PWA system with respect to the satisfaction of a specific temporal logic formula.

Relying on a temporal logic formula to guide the refinement of an abstraction has been previously used in methods based on counterexample guided refinement [8] for verification purposes. Instead of performing many model checking steps, in this work we aim directly at the construction of formula equivalent finite quotients of infinite systems. In addition, our approach yields more informative results, since we obtain regions of initial conditions for which the system satisfies the specification, instead of simple Yes/No answers. The analysis of PWA systems for properties such as invariance and reachability has been previously considered in literature focused on controlling PWA systems [1], [15]. In this paper, we focus on the analysis of autonomous PWA systems but allow for greater expressivity by considering specifications given as LTL formulas.

The method presented in this paper was implemented in MATLAB and is available at http://hyness.bu.edu/software.

This work was partially supported by grants ARO W911NF-09-1-0088, NSF CNS-0834260, and AFOSR FA9550-09-1-020 at Boston University and by grant GA201/09/1389 at Masaryk University. B. Yordanov (yordanov@bu.edu) is with the Dept. of Biomedical Engineering, Boston University. J. Tůmová (xtumova@fi.muni.cz, tumova@bu.edu) is with Dept. of Informatics, Masaryk University and the Dept. of Mechanical Engineering, Boston University. C. Belta (cbelta@bu.edu) is with the Dept. of Mechanical Engineering and the Division of Systems Engineering, Boston University. I. Černá (cerna@fi.muni.cz) and J. Barnat (barnat@fi.muni.cz) are with Dept. of Informatics, Masaryk University

II. Definitions and Preliminaries

A. Transition System

Definition 1: A transition system is a tuple $T = (Q, \rightarrow, O, o)$, where Q is a (possibly infinite) set of states, $\rightarrow \subseteq Q \times Q$ is a transition relation, O is a finite set of observations, and $o : Q \rightarrow O$ is an observation map.

A transition $(x, x') \in \rightarrow$ is also denoted by $x \rightarrow x'$. The transition system T is *finite* if its set of states Q is finite and *infinite* otherwise. The transition system T is *deterministic* if, for all $x \in Q$, there exists at most one $x' \in Q$ such that $x \rightarrow x'$. Finally, T is called *non-blocking* if, for every state $x \in Q$, there exists $x' \in Q$ such that $x \rightarrow x'$. In this paper only non-blocking transition systems are considered.

A *trajectory* or *run* of T starting from x_0 is an infinite sequence $x_0x_1x_2\ldots$ with the property that $x_i \in Q$, and $x_i \rightarrow x_{i+1}$, for all $i \geq 1$. A trajectory $x_0x_1x_2\ldots$ defines a *word* $o(x_0)o(x_1)o(x_2)\ldots$. The set of all words generated by the set of all trajectories starting at $x \in Q$ is called the *language* of T originating at x and is denoted by $\mathcal{L}_T(x)$.

A subset $X \subseteq Q$ is called a *region* of T. The set of all trajectories originating in X is denoted by $T(X)$ and the set of all words generated by runs in $T(X)$ is called the language of T originating at X and is denoted by $\mathcal{L}_T(X) = \bigcup_{x\in X} \mathcal{L}_T(x)$. For an arbitrary region X, we define the set of states $Pre_T(X)$ that reach X in one step as

$$Pre_T(X) = \{x \in Q \,|\, \exists x' \in X,\ x \rightarrow x'\} \tag{1}$$

The observation map o of a transition system T induces an observational equivalence relation $\sim$ over the set of states Q. We say that states $x_1, x_2 \in Q$ are equivalent (written as $x_1 \sim x_2$) if and only if $o(x_1) = o(x_2)$. The equivalence relation naturally induces a *quotient transition system* $T/_\sim = (Q/_\sim, \rightarrow_\sim, O, o_\sim)$. $Q/_\sim$ is the quotient space (the set of all equivalence classes). Given an equivalence class $X \in Q/_\sim$ [1], we denote the set of all equivalent states in that class by $con(X) \subseteq Q$ (*con* stands for concretization map). Since all states $x \in Q$ in an equivalence class $X \in Q/_\sim$ have the same observation, $o_\sim(X)$ is well defined and given by $o_\sim(X) = o(x), x \in con(X)$. The transition relation $\rightarrow_\sim$ is defined as follows: for $X_1, X_2 \in Q/_\sim$, $X_1 \rightarrow_\sim X_2$ if and only if there exist $x_1 \in con(X_1)$ and $x_2 \in con(X_2)$ such that $x_1 \rightarrow x_2$. It is easy to see that

$$\forall X \in Q/_\sim, \mathcal{L}_T(con(X)) \subseteq \mathcal{L}_{T/_\sim}(X). \tag{2}$$

The quotient transition system $T/_\sim$ is said to *simulate* the original system T.

Definition 2: The equivalence relation $\sim$ induced by the observation map o is a bisimulation of a transition system $T = (Q, \rightarrow, O, o)$ if, for all states $x_1, x_2 \in Q$, if $x_1 \sim x_2$ and $x_1 \rightarrow x_1'$, then there exist $x_2' \in Q$ such that $x_2 \rightarrow x_2'$ and $x_1' \sim x_2'$.

If $\sim$ is a bisimulation, then the quotient transition system $T/_\sim$ is called a *bisimulation quotient* of T, and the transition systems T and $T/_\sim$ are called *bisimilar*, denoted as $T/_\sim \backsimeq T$. An immediate consequence of bisimulation is language equivalence, *i.e.,*

$$\forall X \in Q/_\sim, \mathcal{L}_T(con(X)) = \mathcal{L}_{T/_\sim}(X). \tag{3}$$

Using the $Pre_T()$ operator defined in Equation (1), a characterization of bisimulation can be given as follows: the equivalence relation $\sim$ is a bisimulation if and only if for all equivalence classes $X' \in Q/_\sim$, $Pre_T(con(X'))$ is either empty or a finite union of equivalence classes. Equivalently, the bisimulation property (Def. 2) is violated at $X \in Q/_\sim$ if there exists a state $X' \in Q/_\sim$, such that

$$\emptyset \subset con(X) \cap Pre_T(con(X')) \subset con(X). \tag{4}$$

This leads to an iterative procedure for the construction of the coarsest bisimulation $\sim$, known as the "bisimulation algorithm" [5], which in general does not terminate but if it does then $T/_\sim$ is a finite bisimulation quotient, and can be (equivalently) used for model checking instead of T (see Equation (7)).

B. Linear Temporal Logic and Büchi Automata

To specify temporal logic properties for system trajectories, in this paper we use Linear Temporal Logic (LTL) formulas [9]. LTL formulas are inductively defined over the set of observations O, by using the standard Boolean operators (*i.e.,* $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction)) and the temporal operators $\bigcirc$ ("next"), $\mathcal{U}$ ("until"), $\Box$ ("always"), $\Diamond$ ("eventually"). Each LTL formula ϕ over an alphabeth O defines a language $\mathcal{L}_\phi$ of words that satisfy ϕ. Given a finite transition system $T = (Q, \rightarrow, O, o)$ and an LTL formula ϕ over O, an off-the-shelf model checker, such as `NuSMV` [7] or `DiVinE` [3], can be used to check whether the language $\mathcal{L}_T(x)$ satisfies ϕ, for all $x \in Q$. For a region $X \subseteq Q$, we write $T(X) \vDash \phi$ if all the words from $\mathcal{L}_T(X)$ satisfy ϕ. Let

$$X_T^\phi = \{x \in Q \mid T(x) \vDash \phi\}. \tag{5}$$

Note that, if $x \notin X_T^\phi$, then there exists a word in $\mathcal{L}_T(x)$ that violates ϕ. Therefore, X_T^ϕ is the largest region of T satisfying ϕ.

If $T/_\sim$ is a quotient of T, then for any equivalence class $X \in Q/_\sim$ and formula ϕ, we have:

$$T/_\sim(X) \vDash \phi \Rightarrow T(con(X)) \vDash \phi \tag{6}$$

In addition, if $\sim$ is a bisimulation, then

$$T/_\sim(X) \vDash \phi \Leftrightarrow T(con(X)) \vDash \phi \tag{7}$$

Properties (6) and (7) (which follow immediately from (2) and (3)) allow one to model check finite quotients and extend the results to the (possibly infinite) original transition system.

Definition 3: A Büchi automaton is a tuple $\mathcal{B} = (S, S_0, O, \delta_\mathcal{B}, F)$ where S is a finite set of states, $S_0 \subseteq S$ is the set of initial states, O is the input alphabet, $\delta_\mathcal{B} : S \times O \rightarrow 2^S$ is a nondeterministic transition function and $F \subseteq S$ is the set of accepting (final) states.

The semantics of a Büchi automaton is defined over infinite input words. A run of $\mathcal{B}$ over a word $w = o_1o_2o_3\ldots \in O^\omega$ is a sequence $\rho = s_0s_1s_2\ldots$, where $s_0 \in S_0$ and $(s_{i-1}, o_i, s_i) \in \delta_\mathcal{B}$ for all $i \geq 1$. Let $\inf(\rho)$ denote the set of states that appear in the run ρ infinitely often. A run ρ of $\mathcal{B}$

[1] with a slight abuse of notation, we use symbol X to denote states of $T/_\sim$ (*i.e.* $X \in Q/_\sim$) and regions of T (*i.e.* $X \subseteq Q$) but the precise meaning should be clear from the context

is accepting if and only if $\inf(\rho) \cap F \neq \emptyset$. In other words, an input word w is accepted by $\mathcal{B}$ if and only if there exists at least one run over w that visits F infinitely often. We denote by $\mathcal{L}_{\mathcal{B}}$ the language accepted by $\mathcal{B}$, i.e. the set of all words accepted by $\mathcal{B}$. An LTL formula ϕ can always be translated into a Büchi automaton $\mathcal{B}_\phi$ using off-the-shelf tools such as LTL2BA [14], such that $\mathcal{L}_{\mathcal{B}_\phi} = \mathcal{L}_\phi$. A Büchi automaton is deterministic if S_0 and $\delta_{\mathcal{B}}(s, o)$ are singletons for all $s \in S$ and $o \in O$.

III. Problem Formulation and Approach

Let $\mathcal{X}_l, l \in L$ be a set of open polytopes in $\mathbb{R}^N$, where L is a finite index set, such that $\mathcal{X}_{l_1} \bigcap \mathcal{X}_{l_2} = \emptyset$ for all $l_1, l_2 \in L$, $l_1 \neq l_2$ and $\mathcal{X} = \cup_{l \in L} cl(\mathcal{X}_l)$ is a closed full-dimensional polytope in $\mathbb{R}^N$ ($cl(\mathcal{X}_l)$ denotes the closure of set $\mathcal{X}_l$). A discrete-time piecewise affine (PWA) system is defined as:

$$x_{k+1} = A_l x_k + b_l,\ x_k \in \mathcal{X}_l,\ l \in L,\ k = 0, 1, 2, \ldots. \quad (8)$$

We assume that $\mathcal{X}$ is an invariant for all trajectories of the system and matrix A_l is nonsingular for all $l \in L$. We are interested in properties of (8) specified in terms of the polytopes from its definition. Informally, the semantics of system (8) can be understood in the following sense: a trajectory of the system $x_0 x_1 x_2 \ldots$ produces an infinite word $l_0 l_1 l_2 \ldots$, where $l_i \in L$ is the index of the polytope visited at step i (*i.e.* $x_i \in \mathcal{X}_{l_i}$). An LTL formula over L can then be interpreted over trajectories of the system (see Sec. II). In the following, we give a formal definition of the semantics through an embedding into a transition system.

Definition 4: The embedding transition system $T_e = (Q_e, \rightarrow_e, O_e, o_e)$ for the PWA system from Eqn. (8) is defined as:

- $Q_e = \cup_{l \in L} \mathcal{X}_l$,
- $x \rightarrow_e x'$ if and only if there exist $l \in L$ such that $x \in \mathcal{X}_l$ and $x' = A_l x + b_l$,
- $O_e = L$,
- $o_e(x) = l$ if and only if $x \in \mathcal{X}_l$.

Note that the embedding T_e has an infinite number of states and is always deterministic and non-blocking.

Definition 5: Given a subset $X \subseteq Q_e$, we say that all trajectories of system (8) originating in X satisfy formula ϕ if and only if $T_e(X)$ satisfies ϕ.

Problem 1: Given a PWA system (8) and an LTL formula ϕ over L, find the largest region of initial states, from which all trajectories of the system satisfy ϕ.

The solution to Problem 1 amounts to the computation of $X^\phi_{T_e}$ (see Eqn. (5)). Since T_e has an infinite number of states, it cannot be analyzed directly. Our approach is based on the construction of a finite abstraction that is equivalent to the initial system with respect to the satisfaction of a specific temporal logic formula. We develop the notion of formula equivalent quotients in Sec. IV and describe an algorithm for their computation in Sec. V, where the results are valid for general deterministic infinite transition systems. We describe the construction of a formula equivalent finite quotient for T_e in Sec. VI, which leads to the solution of Problem 1. As it will become clear later, our approach is conservative, in the sense that, we can only "try" to find the formula equivalent

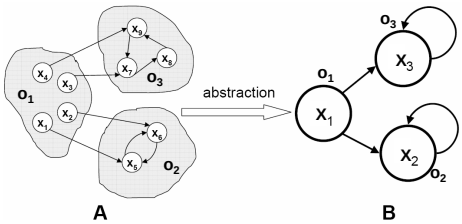


Fig. 1: The transition system shown in **A** forms three equivalence classes under observational equivalence. The resulting finite quotient is shown in **B**. The finite quotient is clearly not a bisimulation quotient and the bisimulation property (Eqn. (4)) is violated at state X_1. However, the quotient is ϕ-equivalent for LTL formula $\phi = \bigcirc(X_2 \vee X_3)$ and can be equivalently used instead of the original system for model checking against the formula.

quotient of T_e and the satisfying region $X^\phi_{T_e}$ but, in general, we can only guarantee to obtain subsets of the latter.

There are several simplifying assumptions in the formulation of Problem 1 that may seem restrictive. First, $\mathcal{X}$ is assumed to be an invariant for all trajectories of (8). The method proposed here can be easily extended to capture trajectories of (8) leaving $\mathcal{X}$ through transitions to states in T_e labeled as being "outside" the defined space. Then, the LTL specification must be augmented to specify that "outside" is never visited. Second, we capture only the reachability of open full dimensional polytopes in the semantics of the embedding. Arguably, this is enough for practical purposes, since only sets of measure zero are disregarded, and it is unreasonable to assume that equality constraints can be detected in real-world applications. There are two situations in which the boundaries can affect the semantics of the trajectories non-trivially: (1) when trajectories originate and remain in such sets for all times, and (2) when trajectories start in open polytopes and then "vanish" in the boundaries. For both these situations, the system dynamics and the polytopes need to satisfy special conditions, which can be easily derived, but are omitted due to space constraints. Third, the specification is given over the indexes $l \in L$ of the polytopes $\mathcal{X}_l$ from the system definition. However, arbitrary linear inequalities can be accommodated simply by refining the polytopic partition - the resulting PWA will have some polytopes with identical dynamics. Fourth, the assumption that $A_l, l \in L$ are nonsingular is usually satisfied in practice, when the discrete-time PWA description is obtained through identification from experimental data [17], or through discretization of continuous linear dynamics, when $A_l, l \in L$ are matrix exponentials.

IV. Formal Analysis of Infinite Transition Systems

In this section, we consider the following problem:

Problem 2: Given an infinite transition system T (Def. 1) and an LTL formula ϕ over its set of observations O, find X^ϕ_T (Eqn. (5)).

We assume that, given the observational equivalence relation $\sim$ (see Sec. II-A), the finite quotient $T/_\sim$ is computable (its computation for T_e is discussed in Sec. VI). Then, $X^\phi_{T/_\sim}$ can be computed by model checking $T/_\sim$ from each state

$X \in Q/_\sim$. From Eqn. (6) it follows that $con(X^\phi_{T/_\sim}) \subseteq Q$ is a satisfying region in T but, in general, $con(X^\phi_{T/_\sim}) \subseteq X^\phi_T$, so only a subset of the largest satisfying region is obtained.

If $\sim$ is a bisimulation of T then from Eqn. (7) it follows that for any LTL formula ϕ

$$con(X^\phi_{T/_\sim}) = X^\phi_T. \tag{9}$$

A solution to Problem 2 can then be obtained by computing the coarsest bisimulation $\sim$ of T using the bisimulation algorithm (see Sec. II-A)and model checking the bisimulation quotient $T/_\sim$ from each state to compute $X^\phi_{T/_\sim}$. However, such a procedure would only work for the particular case when T admits a finite bisimulation quotient $T/_\sim$. Alternatively, the equivalence relation produced at each step of the bisimulation algorithm can be used to construct finite simulation quotients, which can then be model checked against an LTL formula. A similar idea was used in [6] for the universal fragment ACTL of CTL. We followed this approach in [25], where we combined state refinement, inspired by the bisimulation algorithm, with model checking in an iterative procedure (see Remark 1 for additional details). At each step, the set $con(X^\phi_{T/_\sim}) \subseteq X^\phi_T$ provided an under-approximation of the solution to Problem 2. This under-approximation could be improved by performing additional iterations but the termination of the algorithm with an exact solution could not be guaranteed. In the following, we consider conditions guaranteeing that Eqn. (9) holds and therefore an exact solution to Problem 2 can be computed. As already stated, bisimulation is one such condition, but as it will become clear later, it is unnecessarily strong.

Definition 6: Given an (infinite) transition system T and an LTL formula ϕ, an observation preserving equivalence relation $\sim$ is a ϕ-equivalence of T if and only if for all states $x_1, x_2 \in Q$ such that $x_1 \sim x_2$, $T(x_1) \vDash \phi \Leftrightarrow T(x_2) \vDash \phi$

We denote a ϕ-equivalence relation as $\sim_\phi$ and refer to the quotient $T/_{\sim_\phi}$ as ϕ-equivalent quotient. From Eqn. (7) it follows that a bisimulation relation $\sim$ is a ϕ-equivalence for all LTL formulas ϕ. Bisimulation is a sufficient condition guaranteeing that Eqn. (9) holds but since we are interested in the analysis of T for a specific LTL formula ϕ it can be too restrictive. An example where a formula equivalent quotient is not a bisimulation quotient is shown in Fig. 1.

Proposition 1: Given a transition system T and an LTL formula ϕ, Eqn. (9) is satisfied if and only if $\sim$ is a ϕ-equivalence of T.

Proof: Assume that $\sim$ is a ϕ-equivalence. From Def. 6 it follows that $\forall x \in Q$ such that $T(x) \vDash \phi, x \in con(X), X \in T/_\sim$ we have $T/_\sim(X) \vDash \phi$. Then, $\forall x \in Q$, $x \in X^\phi_T \Leftrightarrow x \in con(X^\phi_{T/_\sim})$ and therefore $X^\phi_T = con(X^\phi_{T/_\sim})$. Assume that $\sim$ is not a ϕ-equivalence. Then, $\exists x_1, x_2 \in Q$ such that $x_1 \sim x_2$, $T(x_1) \vDash \phi$ and $T(x_2) \nvDash \phi$. Considering the equivalence class $X \in Q/_\sim$ such that $x_1, x_2 \in con(X)$ we have $T/_\sim(X) \nvDash \phi$. Then $x_1 \in X^\phi_T$ but $x_1 \notin con(X^\phi_{T/_\sim})$ and therefore $X^\phi_T \neq con(X^\phi_{T/_\sim})$. ∎

Prop. 1 shows that ϕ-equivalence is a necessary and sufficient condition for Eqn. (9). Then, Problem 2 reduces to the computation of $con(X^\phi_{T/_{\sim_\phi}})$, where $T/_{\sim_\phi}$ is a finite, ϕ-equivalent quotient for T. We discuss the computation of formula equivalent quotients in Sec. V.

Remark 1: The method we presented in [25] involved the iterative model checking and refinement of simulation quotients of an infinite transition system. By model checking with both an LTL formula and its negation we were able to target refinement to the specific set of states from which some but not all runs satisfied the formula. Although our method was originally inspired by the bisimulation algorithm, this optimization led to the construction of formula equivalent quotients in the cases when the algorithm terminated. However, a large number of model checking and refinement steps was required. As it will become clear in Sec. V our current approach aims directly at the construction of formula equivalent quotients and is more efficient.

V. Formula Guided Refinement

In this section we develop an algorithm for the computation of ϕ-equivalent quotients of (possibly infinite) transitions systems, leveraging ideas from the bisimulation algorithm and automata-based model checking. We assume that given a deterministic transition system T and the observational equivalence relation $\sim$, the finite quotient $T/_\sim$ is computable (as will be the case for T_e). For simplicity, we also assume that LTL formula ϕ is translated to a deterministic Büchi automaton $\mathcal{B}_\phi$ over the set of observations O. As discussed in Remark 2, this assumption can be relaxed by noting that any LTL formula can be translated to a deterministic Rabin automaton. Since the computation of ϕ-equivalent quotients is guided by formula ϕ, it is most natural to perform the computation in the product automata $P = T/_\sim \otimes \mathcal{B}_\phi$ (Def. 7), where both the structure of the system ($T/_\sim$) and the specification ($\mathcal{B}_\phi$) is captured.

Definition 7: The *product automaton* $P = T/_\sim \otimes \mathcal{B}_\phi$ of a finite transition system $T/_\sim = (Q/_\sim, \rightarrow_\sim, O, o_\sim)$ and a Büchi automaton $\mathcal{B}_\phi = (S, S_0, O, \delta_{\mathcal{B}_\phi}, F)$ accepting the language $\mathcal{L}_\phi$ for some LTL formula ϕ is defined as $P = (S_P, S_{P0}, \delta_P, F_P)$, where

- $S_P = Q/_\sim \times S$ is the set of states,
- $S_{P0} = Q/_\sim \times S_0$ is the set of initial states,
- δ_P is the transition function, where for a $(X, s) \in S_P$, $\delta_P((X, s)) = \{(X', s') \in S_P \mid X \rightarrow_\sim X' \text{ and } s' = \delta_{\mathcal{B}_\phi}(s, o(X))\}$,
- $F_P = Q/_\sim \times F$ is the set of accepting states.

The product automaton is a nondeterministic Büchi automaton with input alphabet containing only one element, which is therefore omitted. An accepting run $r_P = (X_1, s_1)(X_2, s_2)\ldots$ in P can be projected into a run $r_{T/_\sim} = X_1 X_2 \ldots$ of $T/_\sim$, such that $o(X_1)o(X_2)\ldots$ is accepted by $\mathcal{B}_\phi$ [24] and therefore satisfies ϕ. Let us denote by $\alpha : S_P \rightarrow Q/_\sim$, $\alpha(X, s) = X$, the projection of states of product automaton P to the states of $T/_\sim$.

The set $X^\phi_{T/_\sim}$ can be computed as the projection $\alpha(S_Y \cap S_{P0}) \subseteq Q/_\sim$ where $S_Y \subseteq S_P$ is the set of states in P from which all runs are accepting (see Sec. II-B). S_Y can be efficiently computed following the method that we described in [21]. Specifically, we first identify a subset $F_Y \subseteq F$ of

accepting states, from which infinitely many revisits to F are guaranteed. S_Y is then a set of states from which a visit to F_Y is guaranteed in zero or more steps.

We can also easily identify a set of states $S_N \subseteq S_P$ of P from which no runs are accepting. The projection $\alpha(S_N \cap S_{P0}) \subseteq Q/_\sim$ corresponds to $X^{\neg\phi}_{T/_\sim}$ (*i.e.* the largest set of states of $T/_\sim$ from which no runs satisfy ϕ).

Let $S_? \subseteq S_P, S_? = S_P \setminus (S_Y \cup S_N)$ be the set of states from which some but not all runs are accepting in P. The projection $\alpha(S_? \cap S_{P0}) \subseteq Q/_\sim$ corresponds to states of $T/_\sim$ where both runs satisfying ϕ and $\neg\phi$ originate and, therefore, the ϕ-equivalence property (Def. 6) is violated at those states.

Proposition 2: The equivalence relation $\sim$ is a ϕ- equivalence of a deterministic transition system T if and only if $(S_? \cap S_{P0}) = \emptyset$. Then, $S_? = \emptyset$ guarantees that $\sim$ is a ϕ-equivalence.

Proof: Let $(S_? \cap S_{P0}) \neq \emptyset, (X, s) \in (S_? \cap S_{P0})$. Then $X = \alpha((X, s))$ is a state of $T/_\sim$ such that $\exists x_1, x_2 \in con(X), T(x_1) \vDash \phi, T(x_2) \vDash \neg\phi$ and therefore $\sim$ is not a ϕ-equivalence. Let $(S_? \cap S_{P0}) = \emptyset$. Then $\forall X \in Q/_\sim$ we have $\forall x \in con(X), T(x) \vDash \phi$ or $\forall x \in con(X), T(x) \vDash \neg\phi$ and therefore $\sim$ is a ϕ-equivalence. ∎

In general, the set $S_?$ is nonempty but can be made empty if accepting and non-accepting runs from each state $(X, s) \in S_?$ are separated through refinement. Following from Prop. 2 and the discussion presented in Sec. IV this provides a solution to Problem 2. Since the structure of P is completely determined by $\mathcal{B}_\phi$ and $T/_\sim$ and $\mathcal{B}_\phi$ is fixed, the only way to refine states in P is through refinement of $T/_\sim$. We refine a state $(X, s) \in S_?$ by applying the procedure REFINE $(T/_\sim, \alpha(X, s))$, followed by UPDATE $(P, (X, s))$, which simply projects changes made during REFINE $(T/_\sim, \alpha(X, s))$ into the product P.

Algorithm 1 REFINE$(T/_\sim, X)$

Input: finite quotient $T/_\sim$ of a deterministic T, state $X \in Q/_\sim$
Output: finite quotient $\hat{T}/_\sim$ refined at state X

initialize $\hat{Q}/_\sim = Q/_\sim \setminus X$
for each state X' such that $X \rightarrow_\sim X'$ **do**
  construct state X_{new} such that:
  $con(X_{new}) = con(X) \cap Pre_T(con(X'))$
  add state X_{new} to $\hat{Q}/_\sim$
end for
update $\hat{\rightarrow}_\sim$;
update $\hat{o}_\sim$;
return $\hat{T}/_\sim = (\hat{Q}/_\sim, \hat{\rightarrow}_\sim, O, \hat{o}_\sim)$

The refinement procedure REFINE$(T/_\sim, X)$ (Algorithm 1) is inspired by the bisimulation algorithm (see Sec. II-A). Unlike the bisimulation algorithm, which refines the equivalence relation $\sim$ globally, REFINE$(T/_\sim, X)$ refines the quotient $T/_\sim$ locally at a state $X \in Q/_\sim$. By considering all its successors, state $X \in Q/_\sim$ is partitioned so that each resulting subset X_{new} can make a transition to only one of the original successor states X'. An outgoing transition $X_{new} \hat{\rightarrow}_\sim X'$ of a newly formed state X_{new}, where $con(X_{new}) = con(X) \cap Pre_T(con(X'))$ is thus implicitly induced. The incomming transitions of X_{new} are updated as follows: each transition $X' \rightarrow_\sim X$ is replaced with $X' \hat{\rightarrow}_\sim X_{new}$ if $con(X') \cap Pre_T(con(X_{new})) \neq \emptyset$. All subsets of a refined state inherit the observation of the parent and, therefore, $\hat{o}_\sim$ is easily updated.

Algorithm 2 COMPUTE $X^\phi_{\hat{T}/_\sim}$

Input: deterministic infinite TS T and LTL formula ϕ
Output: $X^\phi_{\hat{T}/_\sim} \subseteq X^\phi_T$

Construct $T/_\sim$, such that $\sim$ is observational equivalence
Construct deterministic BA $\mathcal{B}_\phi$, such that $\mathcal{L}_{\mathcal{B}_\phi} = \mathcal{L}_\phi$
Build product automaton $P = T/_\sim \otimes \mathcal{B}_\phi$
Initialize $\hat{T}/_\sim = T/_\sim$, $\hat{P} = P$
repeat
  Compute S_Y and S_N in $\hat{P}$
  $S_? = S_{\hat{P}} \setminus (S_Y \cup S_N)$
  for all $(X, s) \in S_?$ **do**
    if X not refined in $\hat{T}/_\sim$ and X is large enough **then**
      $\hat{T}/_\sim$ =REFINE$(\hat{T}/_\sim, X)$
    end if
    if X refined in $\hat{T}/_\sim$ **then**
      UPDATE$(\hat{P}, (X, s))$
    end if
  end for
until $\hat{P}$ not updated during previous iteration
return $X^\phi_{\hat{T}/_\sim} = \alpha(S_Y \cap S_{P0})$

The overall method discussed in this section is summarized in Algorithm 2. Since the regions of T contain, in general, an infinite number of states, the algorithm might perform an infinite number of refinement steps. To ensure the algorithm terminates, we refine a state only if it corresponds to a "large enough" region of T (see Sec. VI for such a measure for T_e). This means that $S_?$ might be nonempty when we force the algorithm to terminate, and we cannot guarantee an exact solution to Problem 2.

It is important to note that if a state X is refined in $\hat{T}/_\sim$, not every state (X, s) is necessarily refined in $\hat{P}$. There is a one-to-many correspondence between the refined product $\hat{P}$ and the refined quotient $\hat{T}/_\sim$, and the projection α is of the type $\alpha : S_{\hat{P}} \rightarrow 2^{\hat{Q}/_{\sim_\phi}}$. This might lead to a major reduction in the computational complexity of the solution: the product automaton $\hat{P}$ after refinement might be significantly smaller than the product automaton $\hat{T}/_\sim \otimes \mathcal{B}$, which we used for model checking in our old approach [25].

We present an intuitive illustration of this idea in Fig. 2. Fig. 2-A shows an infinite transition system T and Fig. 2-B shows its observational equivalence quotient $T/_\sim$. The task is to find the states of T from which an LTL formula $\phi = \Diamond(a \wedge \Diamond(a \wedge \Diamond a))$, saying that a will hold at least three times in future, is satisfied. The formula can be translated into the deterministic Büchi automaton $\mathcal{B}_\phi$ given in Fig. 2-C. Fig. 2-D and 2-E show an application of the new approach presented in this paper. First, the product P is created and the set $S_?$ is computed (Fig. 2-D). Then, refinement is applied

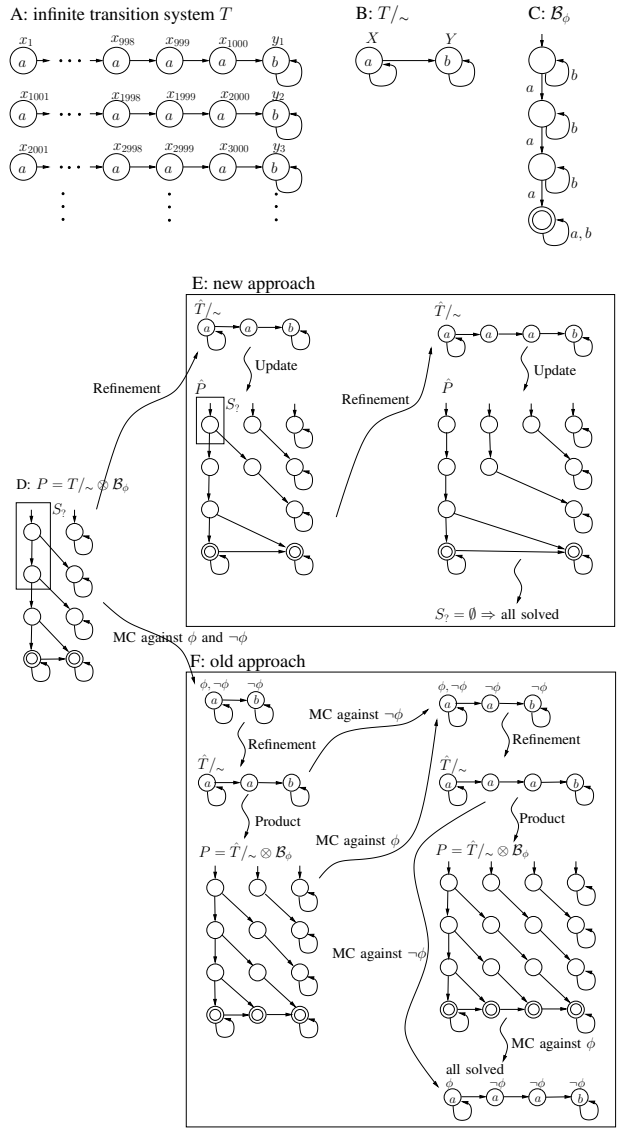


Fig. 2: A case study illustrating the difference between the approach proposed here and the method developed in [25]. **A:** Infinite transition system T. **B:** Observational equivalence quotient $T/_\sim$. **C:** Deterministic Büchi automaton $\mathcal{B}_\phi$ **D:** Product $P = T/_\sim \otimes \mathcal{B}_\phi$. **E:** New approach illustration. **F:** Old approach illustration. MC stands for model checking.

in $\hat{T}/_\sim$ on the states corresponding to $S_?$ and the product $\hat{P}$ is updated, analyzed, and a new $S_?$ is found. We again apply refinement in $\hat{T}/_\sim$ and update $\hat{P}$. We can see that $S_?$ is empty after the second application of refinement and thus, for each state of $\hat{T}/_\sim$ we can decide whether ϕ holds or not. Fig. 2-D and 2-F illustrate the old approach presented in our previous work. This approach involves model checking against ϕ and $\neg\phi$ after each refinement step applied in $\hat{T}/_\sim$. The model checking procedure involves the construction and analysis of a product automaton $P = \hat{T}/_\sim \otimes \mathcal{B}_\phi$ (and $\hat{T}/_\sim \otimes \mathcal{B}_{\neg_\phi}$, respectively), although this is transparent to the user of a model checking tool. Fig. 2-E and 2-F demonstrate that the product structure is smaller in the case of new approach, because states in the product are refined only where really needed. As analysis in the product automaton is the core of the both approaches, we can see that the computational complexity might be reduced significantly. This is also a good example demonstrating the difference between the ϕ-equivalence and bisimulation algorithms: if we applied the bisimulation algorithm, we would have ended with a $\hat{T}/_\sim$ with 1001 states.

Remark 2: The approach presented above is restricted to LTL formulas that can be translated to deterministic Büchi automata. It is well known that there exist LTL formulas that can only be translated to non-deterministic Büchi automata. However, any LTL formula can be translated into a language-equivalent deterministic Rabin automaton $\mathcal{R} = (S, S_0, O, \delta_\mathcal{R}, F)$, which is identical to a Büchi automaton, except for its acceptance condition, which is defined as $F = \{(G_1, B_1), \ldots, (G_n, B_n)\}$, where $G_i, B_i \subseteq S$ for all $i \in \{1, \ldots, n\}$. A run ρ is accepting if $inf(\rho) \cap G_i \neq \emptyset \wedge inf(\rho) \cap B_i = \emptyset$ for some $i \in \{1, \ldots, n\}$.

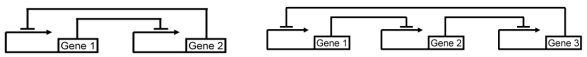
Our solution can be easily adapted for deterministic Rabin automata. The construction of the product automaton is similar to the Büchi case. The only difference is in the computation of S_Y, which in this case is given as the set of states from which a visit to G_Y is guaranteed in zero or more steps. G_Y is computed as a set of states from which infinitely many visits to some G_i without any visit to the corresponding B_i is guaranteed.

Remark 3: The overall complexity of the solution can be further reduced by enforcing a specific order in which states of the product are refined. This optimization is based on the decomposition of the product automaton into maximal strongly connected components (SCCs). It can be shown that all states from a SCC belong to the same set S_Y, S_N, or $S_?$. In addition, if all states from a SCC belongs to set S_Y or S_N then all states in other SCCs reachable from it must also belong to S_Y or S_N, respectively. Converting the product automaton to a SCC quotient graph provides two major improvements. First, it allows for a more efficient computation of sets S_Y, S_N, and $S_?$. Second, refinement in the product automaton can be performed more efficiently in a bottom up manner, while unnecessary refinement is avoided. Note that the current implementation of the methods described in this paper does not include the suggested optimizations.

VI. Finite quotients of PWA systems

Through the embedding of the PWA system (Eqn. (8)) into an infinite transition system T_e (Def. 4), we reduced Problem 1 to Problem 2. In Sec. IV we developed the notion of formula equivalent finite quotients of infinite systems and showed that such quotients can be used to provide the solution to Problem 2. Based on the assumptions that finite quotients can be constructed and each step of the refinement procedure can be implemented, we proposed an algorithm for the refinement of such quotients in order to obtain formula equivalence in Sec. V. In this section, we discuss the construction of the quotients, which completes the solution to Problem 1. Note that this is a summary of results from [25].

From the definitions of the observational equivalence relation $\sim$, induced by the observation map o (Sec. II-A) and T_e (Def. 4), the initial set of states $Q_e/_\sim$, of the finite quotient



A B

Fig. 3: Schematic representations of: **A** the genetic toggle switch [13]; **B** the Repressilator [11].

$T_e/\sim$, is simply the set of observations $Q_e/\sim = O_e = L$ and the observation map is identity. Given a state $l \in Q_e/\sim$, $con(l) = \mathcal{X}_l$ is a polytope from the system definition (Eqn. (8)). In order to finish the construction of the quotient, we need to find the set of transitions $\rightarrow_{e,\sim}$. By the definition of $\rightarrow_\sim$ (Sec.II-A) and Eqn. (1), given $l, l' \in Q_e/\sim$, we have:

$$\begin{aligned} l \rightarrow_{e,\sim} l' \quad &\Leftrightarrow \quad con(l) \cap Pre_{T_e}(con(l')) \neq \emptyset \Leftrightarrow \\ &\Leftrightarrow \quad \mathcal{X}_l \cap Pre_{T_e}(\mathcal{X}_{l'}) \neq \emptyset \end{aligned} \tag{10}$$

The transition relation $\rightarrow_{e,\sim}$ can then be constructed using polyhedral operations only, since

$$\mathcal{X}_l \cap Pre_{T_e}(\mathcal{X}_{l'}) = \mathcal{X}_l \cap A_l^{-1}(\mathcal{X}_{l'} - b_l). \tag{11}$$

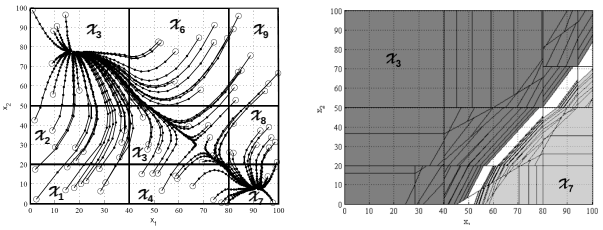
In order to implement REFINE$(T_e/\sim, l)$ we need to be able to construct a state l_{new}, such that given $l', \in Q_e/\sim$ where $l \rightarrow_\sim l'$, $con(l_{new}) = con(l) \cap Pre_{T_e}(con(l'))$ (see Algorithm 1), which is computable using Eqn. 11. Outgoing transitions of a refined state are implicitly induced as discussed in Sec. V and the computation of Eqn. (10) can be used to recompute incoming transitions whenever refinement is applied. In order to decide if the region $con(l)$ is large enough to undergo additional refinement (as discussed in Sec. V), we compute the radius of its inscribed sphere and apply refinement if it is larger than a certain predefined limit ϵ.

VII. Implementation and Case Study

The method described in this paper was implemented in MATLAB, where all polyhedral operations were performed using routines from the MPT toolbox [19]. The current version of the implementation does not include the optimization described in Remark 3. The tool takes as input a PWA system (as defined in Eqn. (8)) and an LTL formula and produces a set of satisfying initial regions. The tool can also simulate runs of the system and is made public and freely downloadable at http://hyness.bu.edu/software.

To demonstrate the approach described in this paper and compare it to the method we presented in [25] we developed two PWA models inspired by synthetic networks of repressor genes (Fig. 3) that have been previously constructed [13], [11]. Gene regulation is modeled by ramp functions, which are piecewise affine functions defined by two threshold values, inducing three regions of different dynamics. At low repressor concentrations (below threshold 1) the regulated gene is fully expressed, at high repressor concentrations (above threshold 2) expression is only basal and the response between the two thresholds is graded. The PWA models capture the characteristic behavior of the two systems (*i.e.* bistability (Fig. 4-A) and oscillations (Fig. 5-A)).

The first network we considered (Fig. 3-A) is inspired by the genetic toggle switch [13]. It includes two mutually inhibiting genes and acts as a switch, allowing only one of the genes to be expressed depending on initial conditions.

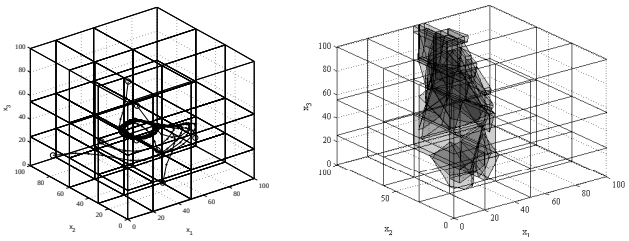


A B

Fig. 4: A: Simulated trajectories of the toggle switch PWA model go towards one of two stable equilibria located in regions $\mathcal{X}_3$ and $\mathcal{X}_7$ (initial states are marked by open circles). **B:** Results from the analysis of the toggle switch PWA model indicate that trajectories originating in the dark gray region are guaranteed to satisfy specification $\phi_1 = \Diamond 3$, while trajectories originating in the light gray region are guaranteed to satisfy $\phi_2 = \Diamond 7$.

Since there are two repressors, two ramp functions are used in a two dimensional ($N = 2$) PWA model and, therefore, the system has a total of nine rectangular regions (denoted $\mathcal{X}_1, \ldots, \mathcal{X}_9$ with the set of labels $L = \{1, \ldots, 9\}$). Dynamics 3 and 7 have unique, asymptotically stable equilibria inside rectangles $\mathcal{X}_3$ and $\mathcal{X}_7$ (see Fig. 4-A). Biologically, the equilibria correspond to the two modes of the system (each gene can be fully expressed, while the other is expressed only basally). An interesting problem is finding the regions of attraction for the two equilibria. By exploiting convexity properties of affine functions on polytopes, it can be shown that $\mathcal{X}_3$ and $\mathcal{X}_7$ are invariants for dynamics 3 and 7, respectively. From this, we can immediately conclude that $\mathcal{X}_3$ and $\mathcal{X}_7$ are regions of attraction for the two equilibria. Therefore, our problem reduces to finding maximal regions satisfying LTL formulas $\phi_1 = \Diamond 3$ and $\phi_2 = \Diamond 7$. In other words, we want to find maximal sets of initial conditions guaranteeing that all trajectories of the system eventually reach regions regions $\mathcal{X}_3$ or $\mathcal{X}_7$, respectively. The initial finite quotient included 9 states and its computation required under 1 sec. on a 3.4GHz machine with 1GB of memory. A limit $\epsilon = 2$ was imposed on the size of regions that can undergo refinement as described in Sec. VI. The refinement procedure for both specifications required under 20 sec. and terminated without returning a formula equivalent quotient but satisfying regions were identified for both ϕ_1 and ϕ_2 and are given in Fig. 4-B. Analyzing the same PWA model with the method described in [25] required a similar amount of time and produced results where the solution sets were the same (as discussed in Remark 1) but small differences in refinement were due to the order in which regions are refined.

The second network we considered (Fig. 3-B) is inspired by the Repressilator circuit [11]. It includes three inhibiting genes and has been shown to produce oscillations in the protein concentrations. A three dimensional ($N = 3$) PWA model was constructed using three ramp functions to capture the effects of gene regulation and, therefore, the system has a total of 27 hyper-rectangular regions, denoted by $\mathcal{X}_1, \ldots \mathcal{X}_{27}$. Biologically, region $\mathcal{X}_1$ is the mode of the system where all three genes are fully activated (*i.e* concentrations of all three repressors are small) and region $\mathcal{X}_{27}$ is the mode of the system where all three genes are only basally expressed (*i.e*



A B

Fig. 5: A: Simulated trajectories of the repressilator PWA model oscillate, visiting region $\mathcal{X}_{14}$ (initial states are marked by open circles). **B:** Result from the analysis of the Repressilator PWA model indicate that trajectories originating in the shaded region are guaranteed to satisfy specification ϕ_3

concentrations of all three repressors are large). The PWA model captures the characteristic oscillatory behavior of the system as demonstrated by simulated trajectories (Fig. 5-A). One of the regions of the system visited by oscillating trajectories is $\mathcal{X}_{14}$, which biologically corresponds to the mode of the system where all three genes are neither fully activated nor only basally expressed (*i.e* in region $\mathcal{X}_{14}$, concentrations of all three repressors fall between the threshold values of the three ramp functions defining the system). We therefore analyzed the Repressilator PWA model for the specification $\phi_3 = \Diamond 14$. The initial finite quotient included 27 states and its computation required under 1 sec. A limit of $\epsilon = 10$ was imposed on the size of regions that can undergo refinement as described in Sec. VI. The refinement procedure required under 30 sec. and terminated without returning a formula equivalent quotient but a set of initial conditions from which the satisfaction of specification ϕ_3 was identified and is shown in Fig. 5-B. Analyzing the Repressilator PWA model with the method described in [25] produced similar results (see Remark 1) but required over 7 min.

VIII. Conclusion

We described a computational framework for the identification of initial sets from which all trajectories of discrete-time piecewise affine (PWA) systems satisfy specifications expressed as Linear Temporal Logic (LTL) formulas over linear predicates. Our approach is based on the iterative construction and refinement of abstractions. We showed that existing methods for the refinement of such abstractions might be too restrictive and proposed conditions guaranteeing the equivalence of an infinite system and its finite abstraction with respect to a specific temporal logic formula. We developed methods for the refinement of finite abstractions aiming at the construction of formula equivalent systems. We demonstrated that our approach can be applied to the analysis of small genetic networks and provides improvements in complexity over previously developed methods.

References

[1] A.Bemporad, L. Giovanardi, and F. Torrisi, "Performance driven reachability analysis for optimal scheduling and control of hybrid systems," in *Proceedings of the 39th IEEE Conference on Decision and Control*, vol. 1, 2000, pp. 969–974.

[2] M. Antoniotti, F. Park, A. Policriti, N. Ugel, and B. Mishra, "Foundations of a query and simulation system for the modeling of biochemical and biological processes," ser. Proceedings of the Pacific Symposium on Biocomputing, R. Altman, A. Dunker, L. Hunter, and T. Klein, Eds., 2003, pp. 116–127.

[3] J. Barnat, L. Brim, and P. Ročkai, "DiVinE 2.0: High-Performance Model Checking," in *2009 International Workshop on High Performance Computational Systems Biology (HiBi 2009)*. IEEE Computer Society Press, 2009, pp. 31–32.

[4] 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.

[5] A. Bouajjani, J.-C. Fernandez, and N. Halbwachs, "Minimal model generation," in *CAV 90: Computer-Aided Verification*, ser. Lecture Notes in Computer Science, R. P. Kurshan and E. M. Clarke, Eds., vol. 531. Springer, 1990, pp. 197–203.

[6] A. Chutinan and B. H. Krogh, "Verification of infinite-state dynamic systems using approximate quotient transition systems," *IEEE Transactions on automatic control*, vol. 46, no. 9, pp. 1401–1410, 2001.

[7] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "NuSMV Version 2: An OpenSource Tool for Symbolic Model Checking," in *Proc. International Conference on Computer-Aided Verification (CAV 2002)*, ser. LNCS, vol. 2404. Copenhagen, Denmark: Springer, July 2002.

[8] E. Clarke, A. Fehnker, Z. Han, B. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald, "Abstraction and counterexample-guided refinement in model checking of hybrid systems," *International Journal of Foundations of Computer Science*, vol. 14, no. 4, pp. 583–604, 2003.

[9] E. M. Clarke, D. Peled, and O. Grumberg, *Model checking*. MIT Press, 1999.

[10] J. Davoren, V. Coulthard, N. Markey, and T. Moor, "Non-deterministic temporal logics for general flow systems," in *HSCC: 7th International Workshop*, 2004, pp. 280–295.

[11] M. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators," *Nature*, vol. 403, pp. 335–338, 2000.

[12] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, "Hybrid controllers for path planning: a temporal logic approach," in *Proceedings of the 2005 IEEE Conference on Decision and Control*, 2005.

[13] T. Gardner, C. Cantor, and J. Collins, "Construction of a genetic toggle switch in escherichia coli." *Nature*, vol. 403, no. 6767, pp. 339–342, 2000.

[14] P. Gastin and D. Oddoux, "LTL with past and two-way very-weak alternating automata," in *Proceedings of MFCS'03*, B. Rovan and P. Vojtáš, Eds., vol. 2747. Springer, 2003, pp. 439–448.

[15] P. Grieder, "Efficient computation of feedback controllers for constrained systems," PhD Thesis, ETH Zrich, 2004.

[16] W. P. M. H. Heemels, B. D. Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, 2001.

[17] A. L. Juloski, W. Heemels, G. Ferrari-Trecate, R. Vidal, S. Paoletti, and J. Niessen, "Comparison of four procedures for the identification of hybrid systems," *LNCS*, vol. 3414/2005, pp. 354–369, 1993.

[18] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

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

[20] S. G. Loizou and K. J. Kyriakopoulos, "Automatic synthesis of multiagent motion tasks based on LTL specifications," in *43rd IEEE Conference on Decision and Control*, 2004.

[21] M.Kloetzer and C. Belta, "Dealing with non-determinism in symbolic control," in *HSCC: 11th International Workshop*, ser. LNCS. Springer Berlin / Heidelberg, 2008, pp. 287–300.

[22] G. J. Pappas, "Bisimilar linear systems," *Automatica*, vol. 39, no. 12, pp. 2035–2047, 2003.

[23] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 1862–1877, 2006.

[24] M. Y. Vardi and P. Wolper, "An automata-theoretic approach to automatic program verification," in *Proc. of Logic in Computer Science*, 1986.

[25] B. Yordanov, C. Belta, and G. Batt, "Model checking discrete time piesewise affine systems: application to gene networks," in *European Control Conference*, 2007.