Proceedings of the European Control Conference 2007
Kos, Greece, July 2-5, 2007

Model Checking Discrete-Time Piecewise Affine Systems: Application to Gene Networks

Boyan Yordanov, Gregory Batt, and Calin Belta

Abstract— **In this paper, we focus on discrete-time continuous-space Piecewise Affine (PWA) systems, and study properties of their trajectories expressed as temporal and logical statements over polyhedral regions. Specifically, given a PWA system and a Linear Temporal Logic (LTL) formula over linear predicates in its state variables, we attempt to find the largest region of initial states from which all trajectories of the system satisfy the formula. Our method is based on a classical algorithm for the iterative computation of simulation quotients augmented with model checking. We show that the determinism inherent in the problem and the particular linear structure of the invariants and of the dynamics can be exploited in a computationally attractive algorithm. We illustrate the application of our method to the computation of basins of attraction for the two equilibria of a PWA model of a two-gene network.**

Index Terms— **piecewise affine systems, transition systems, simulation quotients, LTL model checking, gene networks**

I. INTRODUCTION

Temporal logics [19] and model checking [16] were developed for specifying and verifying the correctness of digital circuits and computer programs. However, due to their resemblance to natural language, their expressivity, and the existence of off-the-shelf algorithms for model checking, temporal logics have the potential to impact several other areas of engineering. Analysis of systems with continuous dynamics based on qualitative simulations and temporal logic was proposed in [37], [11], [17]. Control of linear systems from temporal logic specifications has been considered in both discrete time [39] and continuous time [28]. The use of temporal logic for task specification and controller synthesis in mobile robotics has been advocated as far back as [3], and recent results include [32], [35], [20], [29]. In the area of systems biology, the qualitative behavior of genetic circuits can be expressed in temporal logic, and model checking can be used for analysis, as suggested in [4], [6].

In this paper, we focus on piecewise affine systems in discrete time and continuous space (called PWA for short), which can be seen as evolving along different affine dynamics (in discrete time) in different non-overlapping polytopal regions of the (continuous) state space. Such systems are particularly attractive for three main reasons. First, they are quite general, since they can approximate nonlinear dynamics with arbitrary accuracy [31], and are proven to be equivalent with several other classes of hybrid systems [24]. Second, there exist computationally attractive techniques for the identification of such systems from input-output data, which include Bayesian methods [25], bounded-error procedures [7], clustering-based methods [21], Mixed-Integer Programming [36], and algebraic geometric methods [40].

Third, PWA systems seem to be particularly suited as models for gene networks (the states are concentrations of species, such as mRNAs, proteins, and small molecule species). Indeed, for most networks, the dynamics are affine, with the exception of the gene regulation function (GRF), which captures the relation between the concentrations of the transcription factors and the activity of a gene. However, recent experimental techniques based of fluorescent reporter genes [23] allow for the collection of a large amount of input-output data relating the transcription factor concentrations and the gene activity. Using one of the above identification techniques, this data can then be used to construct a piecewise affine representation of GRFs, which leads to an overall PWA model of a gene network.

A rich spectrum of properties of gene networks are naturally expressed in Linear Temporal Logic (LTL) [19] formulas over linear predicates in the state variables. Examples include reachability, safety and invariance. However, the PWA systems have an infinite number of states, and model checking cannot be used directly. We use the polytopes from the definition of the PWA system and the linear predicates in the formula to define equivalence classes, and model check the produced quotients. Our method is iterative, and based on the notions of transition system, simulation, and bisimulation [16]. Specifically, in this paper, given a PWA system Σ in a bounded polytope $\mathcal{P}_N$ in $\mathbb{R}^n$ (assumed to be a invariant for Σ), and an LTL formula ϕ over linear predicates, we find (if possible) the largest subset P_ϕ of $\mathcal{P}_N$ with the property that all trajectories originating in P_ϕ satisfy the formula. We show the application of our method to the computation of basins of attraction for the two equilibria of a PWA model of a two-gene network.

From a theoretical and computational point of view, this work can be seen in the context of literature focused on the construction of finite quotients of infinite systems (see [2] for a review), and is closely related to [34], [39], [28]. Unlike counterexample guided refinement [15], [1], which eliminates spurious runs in the abstraction, our approach relays on the iterative construction of a bisimulation. Algorithm 3 (Section III) is an extension of the algorithm for the construction of the coarsest proposition preserving

B. Yordanov is with the Department of Biomedical Engineering, Boston University, Boston, MA 02215, USA yordanov@bu.edu

G. Batt is with the Center for Information and Systems Engineering, Boston University, Brookline, MA 02446, USA batt@bu.edu

C. Belta is with the Center for Information and Systems Engineering, Boston University, Brookline, MA 02446, USA cbelta@bu.edu

This work was partially supported by NSF 0432070 and NSF CAREER 0447721 at Boston University.

ISBN 978-3-9524173-8-6

bisimulation proposed in [9] (also Algorithm 2 from [2]). One of the main contributions of this paper is to show that all the steps of this algorithm are computable for PWA systems, and can be reduced to operations on polyhedral sets, which can be performed efficiently [30]. The embedding of discrete time systems into transition systems is inspired from [34], [39]. However, while the focus in [34], [39] is on characterizing the existence of bisimulation quotients, in this work we focus on computation of simulation and bisimulation quotients, while including the model checking process. Unlike [13], where specifications of the system are verified by model checking approximate quotient transition system, here we compute exact reachability mappings but no explicit periodic sampling is assumed as in [38], [27]. Our previous work [28] considered continuous time linear control systems, while in this paper we study discrete time affine systems.

From an application point of view, this paper relates to [5], [8], [12], [18], where temporal logics are used to specify properties of biomolecular networks. These works aim at checking whether a system satisfies dynamical properties for given (sets of) initial conditions. In contrast, we search for the largest set of initial conditions for which the given properties are satisfied. Our approach yields more informative results, since we obtain regions of initial conditions for which the system satisfies the properties, instead of simple Yes/No answers.

II. PRELIMINARIES

A. Polytopes and Affine Functions

Let $N \in \mathbb{N}$ and consider the N - dimensional Euclidean space $\mathbb{R}^N$. A full dimensional *polytope* $\bar{\mathcal{P}}_N$ is defined as the convex hull of at least $N+1$ affinely independent points in $\mathbb{R}^N$. A set of $M \geq N+1$ points $v_1, \ldots, v_M \in \mathbb{R}^N$ whose convex hull gives $\bar{\mathcal{P}}_N$ and with the property that v_i, $i = 1, \ldots, M$ is not contained in the convex hull of $v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_M$ is called the set of *vertices* of $\bar{\mathcal{P}}_N$. A polytope is completely described by its set of vertices:

$$\bar{\mathcal{P}}_N = Conv\{v_1, \ldots, v_M\}, \tag{1}$$

where $Conv$ denotes the convex hull. Alternatively, $\bar{\mathcal{P}}_N$ can be described as the intersection of at least $N+1$ closed half spaces. In other words, there exist a $K \geq N+1$ and $a_i \in \mathbb{R}^N$, $b_i \in \mathbb{R}$, $i = 1, \ldots, K$ such that

$$\bar{\mathcal{P}}_N = \{x \in \mathbb{R}^N \mid a_i^T x + b_i \leq 0,\ i = 1, \ldots, K\} \tag{2}$$

Forms (1) and (2) are referred to as V- and H- representations of the polytope, respectively. Given a full dimensional polytope $\bar{\mathcal{P}}_N$, there exist algorithms for translation from representation (1) to representation (2) [33], [22]. A *facet* of a polytope $\bar{\mathcal{P}}_N$ is the intersection of $\bar{\mathcal{P}}_N$ with one of its supporting hyperplanes $a_i^T x + b_i = 0$, $i = 1, \ldots, K$ from equation (2).

A polytope $\bar{\mathcal{P}}_N$ without its facets is called an *open polytope*. We use the notation $\mathcal{P}_N$ for open polytopes. *i.e.,* $\mathcal{P}_N = int(\bar{\mathcal{P}}_N)$ and $\bar{\mathcal{P}}_N = cl(\mathcal{P}_N)$, where int and cl stand for interior and closure, respectively.

A function $f : \mathbb{R}^N \to \mathbb{R}^m$ is called affine if it can be written as $f(x) = Ax + b$, $A \in \mathbb{R}^{m \times N}$, $b \in \mathbb{R}^m$, for all $x \in \mathbb{R}^N$. If $\bar{\mathcal{P}}_N$ is a full dimensional polytope in $\mathbb{R}^N$ with set of vertices $\{v_1, \ldots, v_M\}$ and $f : \mathbb{R}^N \to \mathbb{R}^m$ is an affine function, then it is easy to prove that

$$f(\bar{\mathcal{P}}_N) = Conv\{f(v_1), \ldots, f(v_M)\}, \tag{3}$$

i.e., the image of a polytope through an affine function is the convex hull of the vertex images through the affine function.

In the particular case $N = m$, if matrix A is nonsingular, then the vertices, facets, and interior of the polytope map through the affine transformation to the vertices, facets, and interior of the image of the polytope, respectively. Therefore

$$f(int(\bar{\mathcal{P}}_N)) = int(f(\bar{\mathcal{P}}_N)) \tag{4}$$

B. Transition Systems, Simulations, and Bisimulations

Definition 1: A transition system is a tuple $T = (Q, \rightarrow, \Pi, \models)$, where Q is a set of states, $\rightarrow \subseteq Q \times Q$ is a transition relation, Π is a finite set of atomic propositions, and $\models \subseteq Q \times \Pi$ is a satisfaction relation.

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

For an arbitrary proposition $\pi \in \Pi$, we define $[[\pi]] = \{q \in Q | q \models \pi\}$ as the set of all states satisfying it. Conversely, for an arbitrary state $q \in Q$, let $\Pi_q = \{\pi \in \Pi \mid q \models \pi\}$, $\Pi_q \in 2^{\Pi}$, denote the set of all atomic propositions satisfied at q. A *trajectory* or *run* of T starting from q is an infinite sequence $r = r(1)r(2)r(3)\ldots$ with the property that $r(1) = q$, $r(i) \in Q$, and $(r(i), r(i+1)) \in \rightarrow$, for all $i \geq 1$. A trajectory $r = r(1)r(2)r(3)\ldots$ defines a *word* $w = w(1)w(2)w(3)\ldots$, where $w(i) = \Pi_{r(i)}$. The set of all words generated by the set of all trajectories starting at $q \in Q$ is called the *language* of T originating at q and is denoted by $L_T(q)$. A subset P of the state set Q ($P \subseteq Q$) is called a *region* of T. The set of all words generated by all runs of T originating at all states of P is called the language of T originating at P and is denoted by $L_T(P)$. The language of the transition system T is defined as $L_T(Q)$.

For an arbitrary region P, we define the set of states $Pre(P)$ that reach P in one step as

$$Pre(P) = \{q \in Q \mid \exists p \in P,\ q \rightarrow p\} \tag{5}$$

An equivalence relation $\sim \subseteq Q \times Q$ over the state space of T is *proposition preserving* if for all $q_1, q_2 \in Q$ and all $\pi \in \Pi$, if $q_1 \sim q_2$ and $q_1 \models \pi$, then $q_2 \models \pi$. Among the several proposition preservation equivalence relations that can be defined, *propositional equivalence* defined as $q_1 \sim q_2$ if and only if $\Pi_{q_1} = \Pi_{q_2}$ is of special interest. A proposition

preserving equivalence relation naturally induces a *quotient transition system* $T/_\sim = (Q/_\sim, \rightarrow_\sim, \Pi, \vDash_\sim)$. $Q/_\sim$ is the quotient space (the set of all equivalence classes). The transition relation $\rightarrow_\sim$ is defined as follows: for $P_1, P_2 \in Q/_\sim$, $P_1 \rightarrow_\sim P_2$ if and only if there exist $q_1 \in P_1$ and $q_2 \in P_2$ such that $q_1 \rightarrow q_2$. The satisfaction relation is defined as follows: for $P \in Q/_\sim$, we have $P \vDash_\sim \pi$ if and only if there exist $q \in P$ such that $q \vDash \pi$. It is easy to see that

$$L_T(P) \subseteq L_{T/_\sim}(P), \tag{6}$$

for any $P \in Q/_\sim$ (with a slight abuse of notation, we use the same symbol P to denote both a state of $T/_\sim$ and the corresponding region of equivalent states of T). The quotient transition system $T/_\sim$ is said to *simulate* the original system T, which is written as $T/_\sim \geq T$.

Definition 2: A proposition preserving equivalence relation $\sim$ is a bisimulation of a transition system $T = (Q, \rightarrow, \Pi, \vDash)$ if for all states $p, q \in Q$, if $p \sim q$ and $p \rightarrow p'$, then there exist $q' \in Q$ such that $q \rightarrow q'$ and $p' \sim q'$.
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 by $T/_\sim \backsimeq T$. An immediate consequence of bisimulation is language equivalence, *i.e.,* $L_T(P) = L_{T/_\sim}(P)$, for all $P \in Q/_\sim$.

C. Linear Temporal Logic and Model Checking

Definition 3: [Syntax of LTL formulas] A (propositional) linear temporal logic (LTL) formula over Π is recursively defined as follows:

- Every atomic proposition $\pi \in \Pi$ is a formula, and
- If ϕ_1 and ϕ_2 are formulas, then $\phi_1 \vee \phi_2$, $\neg\phi_1$, $\bigcirc\phi_1$, $\phi_1 \mathcal{U} \phi_2$ are also formulas.

The semantics of LTL formulas are given over words in the language of the transition system T.

Definition 4: [Semantics of LTL formulas] The satisfaction of formula [1] ϕ at position $i \in \mathbb{N}$ of word w, denoted by $w(i) \vDash \phi$, is defined recursively as follows:

- $w(i) \vDash \pi$ if $\pi \in w(i)$,
- $w(i) \vDash \neg\phi$ if $w(i) \nvDash \phi$,
- $w(i) \vDash \phi_1 \vee \phi_2$ if $w(i) \vDash \phi_1$ or $w(i) \vDash \phi_2$,
- $w(i) \vDash \bigcirc\phi$ if $w(i+1) \vDash \phi$,
- $w(i) \vDash \phi_1 \mathcal{U} \phi_2$ if there exist a $j \geq i$ such that $w(j) \vDash \phi_2$ and for all $i \leq k < j$ we have $w(k) \vDash \phi_1$

A word w satisfies an LTL formula ϕ, written as $w \vDash \phi$, if $w(1) \vDash \phi$. The transition system T satisfies formula ϕ from region $P \subseteq Q$, written as $T(P) \vDash \phi$, if and only if $w \vDash \phi$ for all $w \in L_T(P)$. Note that the same symbol is used to express the satisfaction of a predicate or a formula.

The symbols $\neg$ and $\vee$ stand for negation and disjunction. The boolean constants $\top$ and $\bot$ are defined as $\top = \pi \vee \neg\pi$ and $\bot = \neg\top$. The other Boolean connectors $\wedge$ (conjunction), $\Rightarrow$ (implication), and $\Leftrightarrow$ (equivalence) are defined from $\neg$ and $\vee$ in the usual way. The *temporal operators* $\bigcirc$ and $\mathcal{U}$ is called the *next* and *until* operators. Formula $\bigcirc\phi$ holds for a word $w(1)w(2)w(3)...$ if and only if formula ϕ holds for the suffix $w(2)w(3)....$ Formula $\phi_1 \mathcal{U} \phi_2$ intuitively means that (over a word) ϕ_2 will eventually become true and ϕ_1 is true until this happens. Two useful additional temporal operators, "eventually" and "always" can be defined as $\Diamond\phi = \top\mathcal{U}\phi$ and $\Box\phi = \phi\mathcal{U}\bot$, respectively. Formula $\Diamond\phi$ means that ϕ becomes eventually true, whereas $\Box\phi$ indicates that ϕ is true at all positions of w. More expressiveness can be achieved by combining the temporal operators. Examples include $\Box\Diamond\phi$ (ϕ is true infinitely often) and $\Diamond\Box\phi$ (ϕ becomes eventually true and stays true forever).

Given a finite transition system $T = (Q, \rightarrow, \Pi, \vDash)$ and a formula ϕ over Π, checking whether the words of T starting from a region P satisfy ϕ ($T(P) \vDash \phi$) is called *model checking*. If we denote by L_ϕ the set of all words (language) satisfying ϕ, then model checking means deciding the language inclusion $L_T(P) \subseteq L_\phi$. Packages for model checking, such as 'NuSMV' [14], have been previously developed and are readily available.

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

$$T/_\sim(P) \vDash \phi \Rightarrow T(P) \vDash \phi. \tag{7}$$

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

$$T/_\sim(P) \vDash \phi \Leftrightarrow T(P) \vDash \phi \tag{8}$$

Properties (7) and (8) (which follow immediately from (6)) allow one to model check finite quotients and extend the results to the (possibly infinite) original transition system.

Definition 5: A region $P \subseteq Q$ of a transition system T is the *largest* region that satisfies a formula ϕ if and only if $q \in P \Leftrightarrow T(q) \vDash \phi$. Therefore $T(P) \vDash \phi$, and no states outside of P satisfy the formula.

III. ITERATIVE CONSTRUCTION AND VERIFICATION OF SIMULATION QUOTIENTS

Using the Pre operator defined in Equation (5), a characterization of bisimulation can be given as follows: a proposition preserving equivalence relation $\sim$ is a bisimulation if and only if $Pre(P)$ is either empty or a finite union of equivalence classes, for all equivalence classes P. This leads to the following iterative procedure for the construction of the coarsest bisimulation $\sim$ [10], [26]:

Algorithm 1 Coarsest bisimulation $\sim$ of T preserving Π

Initialize $\sim$ with propositional equivalence
while there exist $P, P' \in Q/_\sim$ such that $\emptyset \subset P \bigcap Pre(P') \subset P$ **do**
  $P_1 := P \bigcap Pre(P')$;
  $P_2 := P \setminus Pre(P')$;
  $Q/_\sim := Q/_\sim \setminus \{P\} \bigcup \{P_1, P_2\}$;
end while
return $\sim$;

In order to execute an iteration of the algorithm, one must be able to represent (possibly infinite) state sets, perform

[1] We use the same notation $\vDash$ for the satisfaction of propositions by the states of transition systems and for the satisfaction of formulas by words and transition systems. The significance should be clear from the context

Boolean operations, check emptiness, and compute the predecessor operation Pre on the representation of the state sets. In general, the bisimulation Algorithm 1 does not terminate. If the algorithm terminates, then $T/_\sim$ is a finite bisimulation quotient ($T/_\sim \trianglelefteq T$), and can be (equivalently) used for model checking instead of T (see Equation (8)).

However, the satisfaction of an LTL formula ϕ by T might be decided even though the bisimilation algorithm does not terminate. Indeed, the equivalence relation produced at each step of Algorithm 1 can be used to construct finite simulation quotients $T/_\sim$, which can then be model checked against an LTL formula (see Equation (7)). A similar idea was used in [13] for the universal fragment ACTL of CTL.

Combining the bisimulation procedure (Algorithm 1) with model checking at each iterative step leads to Algorithm 2. The algorithm attempts to find the largest region $P_\phi \subseteq Q$ that satisfies an LTL formula ϕ. The initial partition given by propositional equivalence is iteratively refined and the produced finite quotient transition systems are model checked against both ϕ and $\neg\phi$, partitioning the set of states Q into two subsets: P_ϕ (the set of all states satisfying the formula) and $P_{\neg\phi}$ (the set of all states satisfying the negation of the formula).

If such partition exists and is constructed in a finite number of iterations, then all runs of the system satisfying the formula originate in P_ϕ and no such runs originate outside of it. Therefore, P_ϕ is the largest region satisfying ϕ. If there exists a quotient state $P \in Q/_\sim, P \notin P_\phi, P \notin P_\neg\phi$, then $\exists w_1, w_2 \in L_{T/_\sim}(P)$ such that $w_1 \vDash \phi$ and $w_2 \vDash \neg\phi$. If the states of P also happen to violate the bisimulation property of Definition 2, then they will be split by the bisimulation procedure. It is possible that, after the refinement, one of the subregions of P satisfies ϕ, while the other satisfies $\neg\phi$, in which case they can be included in P_ϕ and $P_\neg\phi$ respectively.

The possible outcomes of running Algorithm 2 are the following:

- The algorithm terminates returning the sets P_ϕ and $P_{\neg\phi}$ and $P_\phi \cup P_{\neg\phi} = Q$. In that case P_ϕ is the largest set P such that $T(P) \vDash \phi$, and there are no satisfying runs, originating outside of P_ϕ (P_ϕ can also be empty, in which case there are no states of T satisfying ϕ).
- The algorithm terminates because the maximum number of iterations is exceeded but $P_\phi \cup P_{\neg\phi} \neq Q$. Since further refinement might be possible, and there might be satisfying runs, originating outside of P_ϕ, in general, P_ϕ is not the largest satisfying set.
- The algorithm terminates, since the quotient has been refined enough to be bisimilar with the original system, but $P_\phi \cup P_{\neg\phi} \neq Q$. This can occur whenever all states of an equivalence class do not violate the bisimulation properties of Definition 2, but due to non-determinism, have multiple trajectories, some satisfying ϕ and others $\neg\phi$. Those trajectories would not be separated by additional refinement, since the quotient is already bisimilar, and so, P_ϕ cannot be expanded. Therefore, by Definition 5, it is the largest satisfying set.

Although a partition of all states in Q corresponds only to the first outcome of the algorithm, the property that all runs originating in P_ϕ and none of the runs originating in $P_\neg\phi$ satisfy the formula holds for all cases.

Algorithm 2 Determine the largest P such that $T(P) \vDash \phi$

```
Initialize ∼ with propositional equivalence;
repeat
  P_φ := ∅;
  P_¬φ := ∅;
  Construct T/∼;
  for every P_t ∈ Q/∼ do
    if T/∼(P_t) ⊨ φ then
      P_φ := P_φ ∪ P_t;
    else if T/∼(P_t) ⊨ ¬φ then
      P_¬φ := P_¬φ ∪ P_t;
    end if
  end for
  if P_φ ∪ P_¬φ = Q then
    return P_φ, P_¬φ;
  end if
  P_1 := P ⋂ Pre(P');
  P_2 := P \ Pre(P');
  Q/∼ := Q/∼ \ {P} ⋃ {P_1, P_2};
  if number of iterations exceeds limit then
    return P_φ, P_¬φ;
  end if
until there are no P, P' ∈ Q/∼ such that ∅ ⊂ P ⋂ Pre(P') ⊂ P
return P_φ, P_¬φ;
```

The bisimulation procedure (Algorithm 1) is used for the construction of Algorithm 2 and, therefore, the same operations must be computed. With the exception of model checking, which is a well studied problem when applied to finite quotients, no new operations are required in order to extend Algorithm 1 to Algorithm 2. To show this, we describe how the quotient transition system $T/_\sim$ can be constructed. The set of states $Q/_\sim$ includes all equivalence classes, as induced by the propositional equivalence relation $\sim$. The set of propositions Π is inherited from T. The satisfaction of propositions by states of the quotient follows naturally from the properties of propositional equivalence, which is used to construct it, *i.e.*, if $q \in P, P \in Q/_\sim$ then $\Pi_P = \Pi_q$. We also note that, if an equivalence class is split, all newly formed equivalence subclasses inherit the satisfaction of the parent. We have already assumed that the Pre operation and set intersections can be computed, since they were used in the bisimulation procedure. Therefore, we can use the same operations to induce the transitions of $T/_\sim$ as follows:

$$(P, P') \in \rightarrow_\sim \text{ if and only if } P \cap Pre(P') \neq \emptyset \quad (9)$$

Consequently, Algorithm 2 requires no additional operations, and can be implemented, provided that Algorithm 1 can be implemented.

Partitioning all states of a non-deterministic transition system into satisfying a formula and its negation is not

always possible, even if a finite bisimulation quotient exists. In practice, however, many transition systems of interests, such as embeddings of PWA systems, are deterministic, where finding a bisimulation is equivalent to partitioning the states with respect to a formula.

A property that follows from the system being non-blocking and deterministic, is that an equivalence class of the quotient having multiple outgoing transitions is partitioned by the Pre of its successors. This provides a strategy for refining a quotient and motivates the introduction of some changes in Algorithm 2 in order to optimize it for deterministic systems.

Proposition 1: If a transition system T is non-blocking and its quotient $T/_\sim$ is deterministic, then the equivalence relation $\sim$ is a bisimulation.

Proof: Assume by contradiction that $\sim$ is not a bisimulation. Then, there exist $P, P' \in Q/_\sim$, $q_1, q_2 \in P$, and $q_1' \in P'$ such that $q_1 \to q_1'$ and there does not exist $q_2' \in P'$ such that $q_2 \to q_2'$. However, since T is nonblocking, there exists $q_2'' \in Q$ and $P'' \in Q/_\sim$, $P'' \neq P'$ such that $q_2 \to q_2''$, $q_2'' \in P''$. In the quotient $T_\sim$, this induces $P \to P'$ and $P \to P''$, which implies that $T/_\sim$ is non-deterministic. ■

Proposition 2: An equivalence relation defined on a deterministic, non-blocking transition system T is a bisimulation if and only if the quotient $T/_\sim$ is deterministic.

Proof: From Proposition 1 it follows that if the quotient is deterministic then the equivalence relation is a bisimulation. Assume by contradiction that $T/_\sim$ is not deterministic. Then, there exist $P, P', P'' \in Q/_\sim$ such that $P \to P'$ and $P \to P''$. However, since $\sim$ is a bisimulation, there exists $q, q', q'' \in Q, q \in P, q' \in P', q'' \in P''$ such that $q \to q'$ and $q \to q''$, which implies that T is non-deterministic. ■

If a quotient $T/_\sim$ is deterministic, then for a given formula ϕ and an equivalence class P, $T/_\sim(P) \vDash \phi$ or $T/_\sim(P) \vDash \neg\phi$. This follows from the fact that in a deterministic system only one trajectory and therefore one word is possible starting at any given state.

Following the results described above we can propose a modification of Algorithm 2 that applies to non-blocking, deterministic transition systems and labels all states as either satisfying a formula ϕ or its negation $\neg\phi$. Similar to Algorithm 2, this algorithm refines a quotient and performs model checking with a formula of interest at each step. In this case, however, refinement is done as iterative elimination of nondeterminism. If the quotient system is refined enough to be deterministic, we have shown in Proposition 1 that it is bisimilar with the original transition system (and, in fact, from Proposition 2 it follows that it is bisimilar only when it becomes deterministic).

As in Algorithm 2, if $P_\phi \cup P_{\neg\phi} = Q$ then P_ϕ is the largest set satisfying the formula. However, if $P_\phi \cup P_{\neg\phi} \neq Q$, because of the deterministic property, we know that bisimulation was not achieved and the preset number of iterations was exceeded. This property also allows us to have an infinite loop in the algorithm, since achieving a bisimulation is equivalent to partitioning all states into satisfying ϕ or $\neg\phi$.

The changes that were introduced for the construction of Algorithm 3 do not require the computation of any new operations. As for general transition systems, the Pre operator is used for refinement, and it can also be applied to the construction of the quotient transition systems.

Algorithm 3 Determine the largest P of a deterministic system such that $T(P) \vDash \phi$

```
Initialize ∼ with propositional equivalence;
Initialize P_φ := ∅;
Initialize P_¬φ := ∅;
while TRUE do
  Construct T/∼;
  if number of iterations exceeds limit then
    return P_φ, P_¬φ;
  end if
  for every P_t ∈ Q/∼, P_t ∉ P_φ, P_t ∉ P_¬φ do
    if T/∼(P_t) ⊨ φ then
      P_φ := P_φ ∪ P_t;
    else if T/∼(P_t) ⊨ ¬φ then
      P_¬φ := P_¬φ ∪ P_t;
    end if
  end for
  if P_φ ∪ P_¬φ = Q then
    return P_φ, P_¬φ;
  end if
  for every P ∈ Q/∼, P ∉ P_φ, P ∉ P_¬φ do
    OUT := {P'|P → P'};
    if |OUT| > 1 then
      P_refined := ∅
      for every P' ∈ OUT do
        P_refined := P_refined ⋃{P ⋂ Pre(P')};
      end for
      Q/∼ := Q/∼ \ {P} ⋃{P_refined};
    end if
  end for
end while
```

In this algorithm, once an equivalence class is labeled it is memorized and no longer considered for refinement or model checking. We do not need to refine a state that has been found to satisfy the formula or its negation, since all or no trajectories, respectively, originating at that state satisfy the formula. We do not need to model check that state any more, because additional refining anywhere else in the system would not change the satisfaction of a formula by the state. This optimization limits the explosion of states that have to be considered as refinement progresses.

Another useful feature of the algorithm is that equivalence classes are refined in parallel, *i.e.*, all classes that need refinement are split once at every iteration. This guarantees that, if the execution of Algorithm 3 is terminated, because a preset number of iterations is exceeded, the refinement would not be concentrated in only one region.

IV. FINITE QUOTIENTS OF PWA SYSTEMS

Let $\mathcal{P}_N, \mathcal{P}_N^l$, $l \in L$ be a set of open polytopes in $\mathbb{R}^N$, where L is a finite index set, such that $\mathcal{P}_N^{l_1} \bigcap \mathcal{P}_N^{l_2} = \emptyset$ for

all $l_1, l_2 \in L$, $l_1 \neq l_2$ and $\bigcup_{l\in L} \bar{\mathcal{P}}_N^l = \bar{\mathcal{P}}_N$.

A discrete-time continuous-space piecewise affine (PWA) system is defined as:

$$\Sigma: \quad x_{k+1} = A_l x_k + b_l,\ x_k \in \mathcal{P}_N^l,\ l \in L,\ k = 0, 1, 2, \ldots, \tag{10}$$

where $A_l \in \mathbb{R}^{N\times N}$, $b_l \in \mathbb{R}^N$, for all $l \in L$. We assume that the matrix A_l is nonsingular for any $l \in L$ and that $\mathcal{P}_N$ is an invariant for the trajectories of (10), *i.e.,* if $x \in \mathcal{P}_N$, then $A_l x + b_l \in \mathcal{P}_N$, for all $l \in L$.

We are interested in studying properties of trajectories of system (10) specified in terms of a set of linear predicates of the form

$$\Pi = \{\pi_i \,|\, \pi_i :\ c_i^T x + d_i < 0,\ i = 1, \ldots, K\}, \tag{11}$$

where $x, c_i \in \mathbb{R}^N$ and $d_i \in \mathbb{R}$. Informally, the semantics of system (10) will be given in terms of satisfaction of propositions from (11) in the following sense: a trajectory $x_0 x_1 x_2 \ldots$ produces a word $w_0 w_1 w_2 \ldots$, where each $w_i \in 2^\Pi$ lists the propositions from Π which are satisfied by x_i. Then such words can be checked against satisfaction of LTL formulas ϕ over Π.

Specifically, we consider the following problem:

Problem 1: Given a discrete-time continuous-space piecewise affine system (10) and an LTL formula ϕ over a set of linear predicates (11), find the largest set $P_\phi \subseteq \mathcal{P}_N$ such that all trajectories of (10) originating in P_ϕ satisfy formula ϕ.

To formally define the satisfaction of a formula ϕ over Π by system Σ, we embed Σ into a transition system:

Definition 6: The embedding transition system for system Σ and the set of predicates Π is defined as $T_{emb} = (Q_{emb}, \rightarrow_{emb}, \Pi_{emb}, \vDash_{emb})$, where

- $Q_{emb} = \bigcup_{l\in L} \mathcal{P}_N^l$,
- $(x, x') \in \rightarrow_{emb}$ if and only if there exists $l \in L$ such that $x \in \mathcal{P}_N^l$ and $x' = A_l x + b_l$,
- $\Pi_{emb} = L \bigcup \Pi$,
- $\vDash_{emb}$ is defined as follows: if $\pi = l \in L$, then $x \vDash_{emb} \pi$ if and only if $x \in \mathcal{P}_N^l$; if $\pi = \pi_i \in \Pi$, then $x \vDash_{emb} \pi$ if and only if $c_i^T x + d_i < 0$,

Given a subset $P \subseteq Q_{emb}$, we say that all trajectories of Σ originating in P satisfy formula ϕ if and only if $T_{emb}(P)$ satisfies ϕ.

The embedding transition system T_{emb} has infinitely many states and cannot be model checked. We note that, since regions with different dynamics never overlap, the transition system T_{emb} is deterministic. Therefore we can apply Algorithm 3 to T_{emb} in order to solve Problem 1. In the rest of this section, we show that all the steps of Algorithm 3 applied to T_{emb} are computable, and reduce to operations on polyhedral sets.

The computation of the (initial) propositional equivalence relation $\sim$ amounts to checking the non-emptiness of the open polytopes given by the intersection of each $\mathcal{P}_N^l$ with all subsets of Π (recall that $\mathcal{P}_N^l$ are pairwise disjoint). The equivalence classes formed by all such nonempty sets will be the states of the first quotient $Q_{emb}/\sim$ and all operations with those states are polyhedral operations.

The Pre operator is used to determine transitions of the quotient (Equation (9)), as well as to refine the equivalence relation. We note that, in both cases, Pre is used only to find the intersection $P \bigcap Pre(P')$, where P has a transition to P'. If P satisfies l, $P' = int(Conv\{v_1, \ldots, v_M\})$, since the matrix A_l is non-singular, using Equations (3) and (4) we can compute the intersection as:

$$P \bigcap Pre(P') = int(P \bigcap Conv\{A_l^{-1}(v_i - b_l), \\ i = 1, \ldots, M\}) \tag{12}$$

Therefore, the quotient transition system $T/\sim$ can be constructed, and all computation involved in the execution of Algorithm 3 is reduced to polyhedral set operations and model checking.

The regions P_ϕ and $P_{\neg\phi}$ returned by the algorithm correspond to regions of initial conditions of the system with the property that that all runs originating at P_ϕ and no runs originating at $P_{\neg\phi}$ satisfy the formula. If $P_\phi \bigcup P_{\neg\phi} = \mathcal{P}_N$, then P_ϕ is the largest region of initial conditions for the system satisfying the formula.

Remark 1: There are several simplifying assumptions that we make in the formulation and solution of Problem 1. First, in equation (10) we assume that the matrices A_l are non-singular and the polytope $\mathcal{P}_N$ is an invariant for al trajectories of Σ. However, this assumption is not very restrictive since discrete affine dynamics are usually non-singular (or arise by discretizing continuous dynamics which involves matrix exponentiation, which produces non-singular matrices) and $\mathcal{P}_N$ can be assumed large enough to contain all possible state values in a particular process. Second, we assume that the predicates (equation (11) are given over strict inequalities, and only the reachability of open full dimensional polytopes is captured in the semantics of the embedding and of the quotients. However, this seems to be 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 a real world application.

The algorithm was implemented as a software package for MATLAB. All operations on polytopes were handled using the MPT toolbox [30]. NuSMV was used for model checking [14].

V. ANALYSIS OF A TWO GENE NETWORK

We applied the proposed methods to the analysis of a small genetic network. The system (shown in Figure 1) includes two mutually inhibiting genes and acts as a genetic switch, allowing only one of the genes to be expressed depending on initial conditions.

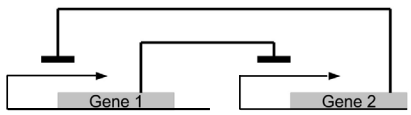

Fig. 1. A genetic switch, consisting of two mutual repressors. High levels of one of the products shut down the expression of the other gene.

To analyze the system, we developed a two dimensional ($N = 2$) discrete time PWA model using ramp functions to represent gene regulation. A ramp function is defined by two threshold values, which induce three regions of different dynamics. At low concentrations of repressor (below threshold 1) the regulated gene is fully expressed, while at high repressor concentrations (above threshold 2) expression is only basal. For concentrations between the two thresholds expression is graded. Since there are two repressors, two ramp functions are used and, therefore, the system has a total of nine rectangular invariants. We use $L = \{1, 2, \ldots, 9\}$ as set of labels and denote the rectangles by $\mathcal{P}^1, \ldots, \mathcal{P}^9$.

In each polytope, the system has dynamics described by Equation (10), with parameters given below.

$$A_1 = \begin{bmatrix} 0.741 & 0 \\ 0 & 0.549 \end{bmatrix}, b_1 = \begin{bmatrix} 23.845 \\ 34.967 \end{bmatrix}$$
$$A_2 = \begin{bmatrix} 0.741 & -0.480 \\ 0 & 0.549 \end{bmatrix}, b_2 = \begin{bmatrix} 23.789 \\ 34.967 \end{bmatrix}$$
$$A_3 = \begin{bmatrix} 0.741 & 0 \\ 0 & 0.549 \end{bmatrix}, b_3 = \begin{bmatrix} 4.406 \\ 34.967 \end{bmatrix}$$
$$A_4 = \begin{bmatrix} 0.741 & 0 \\ -0.672 & 0.549 \end{bmatrix}, b_4 = \begin{bmatrix} 23.845 \\ 55.735 \end{bmatrix}$$
$$A_5 = \begin{bmatrix} 1.023 & -0.545 \\ -0.764 & 0.805 \end{bmatrix}, b_5 = \begin{bmatrix} 14.787 \\ 57.149 \end{bmatrix}$$
$$A_6 = \begin{bmatrix} 0.741 & 0 \\ -0.672 & 0.549 \end{bmatrix}, b_6 = \begin{bmatrix} 4.406 \\ 64.552 \end{bmatrix}$$
$$A_7 = \begin{bmatrix} 0.741 & 0 \\ 0 & 0.549 \end{bmatrix}, b_7 = \begin{bmatrix} 23.845 \\ 3.384 \end{bmatrix}$$
$$A_8 = \begin{bmatrix} 0.741 & 0.480 \\ 0 & 0.549 \end{bmatrix}, b_8 = \begin{bmatrix} 35.544 \\ 3.384 \end{bmatrix}$$
$$A_9 = \begin{bmatrix} 0.741 & 0 \\ 0 & 0.549 \end{bmatrix}, b_9 = \begin{bmatrix} 4.406 \\ 3.384 \end{bmatrix}$$

It is easy to see that dynamics 3 and 7 have unique, asymptotically stable equilibria inside rectangles $\mathcal{P}^3$ and $\mathcal{P}^7$. 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. For this, the introduction of additional propositions is not necessary, *i.e.,* $\Pi = L$. By exploiting convexity properties of affine functions on polytopes, it can be easily proved that $\mathcal{P}^3$ and $\mathcal{P}^7$ are invariants for dynamics 3 and 7, respectively. From this, we can immediately conclude that $\mathcal{P}^3$ and $\mathcal{P}^7$ are regions of attraction for the two equilibria. Therefore, our problem reduces to finding maximal regions satisfying LTL formulas $\phi_1 =$ "$\Diamond\Box\mathcal{P}^3$" and $\phi_2 =$ "$\Diamond\Box\mathcal{P}^7$". In other words, we want to find maximal sets of initial conditions, that eventually reach regions $\mathcal{P}^3$ or $\mathcal{P}^7$ and stay there forever.

The first quotient of the embedding T_{emb} (Definition 6) determined by propositional equivalence has nine states. The transitions in the quotient are shown in Figure 2B. For both formulas, execution of the algorithm was limited to 30 iterations, which required less than 30 minutes on a 3.4 GHz, P4 machine with 1GB RAM.

Performing the analysis with both LTL formulas demonstrated the ability of the algorithm to extract an under-approximation of the attractor regions for the two stable equilibria. The region satisfying ϕ_1 is shown shaded in Figure 2C, while the region satisfying ϕ_2 is shown shaded in Figure 2E. It is interesting to note that the region satisfying ϕ_1 (P_{ϕ_1}, Figure 2C) coincides with the one satisfying $\neg\phi_2$ ($P_{\neg\phi_2}$, Figure 2F) and vice versa. This is expected as the equilibrium states in regions $\mathcal{P}^3$ and $\mathcal{P}^7$ are the only stable equilibria for the system and trajectories tend towards one of those states. Any differences between the regions P_{ϕ_1} and $P_{\neg\phi_2}$ are therefore the result of limiting the number of iterations for the algorithm.

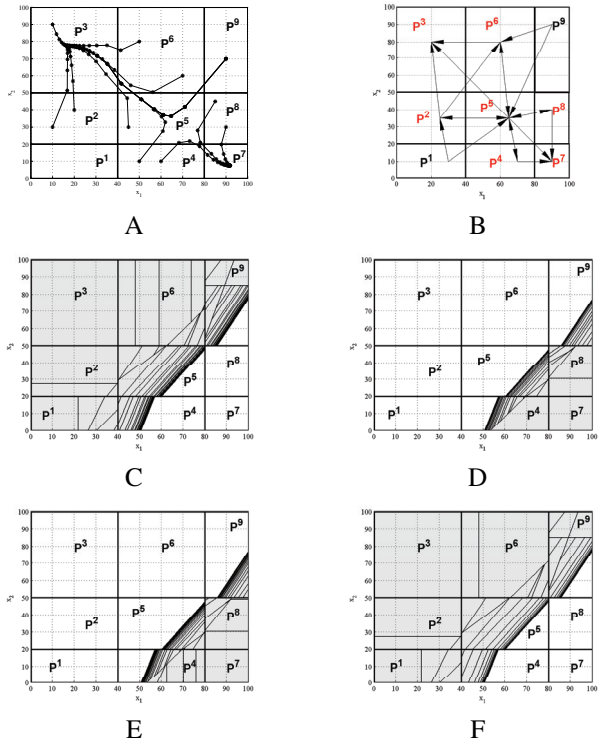

Fig. 2. (A) Sample trajectories of the PWA system show convergence towards one of the stable equilibria in $\mathcal{P}^3$ and $\mathcal{P}^7$. (B) A graphical representation of the first quotient transition system. Regions with labels shown in red contain a self loop. (C-F) Regions P_{ϕ_1} (C), $P_{\neg\phi_1}$ (D), P_{ϕ_2} (E), and $P_{\neg\phi_2}$ (F) satisfying the LTL formulas ϕ_1, $\neg\phi_1$, ϕ_2, and $\neg\phi_2$, respectively.

From the results, it can be concluded that the majority of the computation was performed in improving the refinement between the two regions of attraction. The level of detail in that region is higher if more iterations of the algorithm are performed. This is the result of targeting the refinement to specific states and refining states in parallel at each iteration. For example, it is obvious that partitioning region $\mathcal{P}^3$ in Figure 2C is unnecessary, since all subregions satisfy the formula. Similarly, concentrating refinement only to region $\mathcal{P}^4$ until it is completely refined, might not allow refinement of regions $\mathcal{P}^5$ or $\mathcal{P}^9$ before the iteration limit is reached.

VI. CONCLUSION

We showed that simulation quotients of PWA systems can be computed efficiently, and model checked against LTL for-

mulas to analyze temporal logic properties for trajectories of the initial system. Our approach is inspired by mathematical models and queries arising in analysis of gene networks. We showed the application of our method to the computation of regions of attraction for a PWA model of a genetic switch. In the future, we will focus on analysis of mathematical models of synthetic gene networks constructed from experimental data, analysis under parameter uncertainty, and more complicated dynamics.

REFERENCES

[1] R. Alur, T. Dang, and F. Ivancic, "Counter-example guided predicate abstraction of hybrid systems," in *The 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Warsaw, Poland, 2003.

[2] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, pp. 971–984, 2000.

[3] M. Antoniotti and B. Mishra, "Discrete event models + temporal logic = supervisory controller: Automatic synthesis of locomotion controllers," in *IEEE International Conference on Robotics and Automation*, 1995.

[4] 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.

[5] 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.

[6] ——, "Validation of qualitative models of genetic regulatory networks by model checking: analysis of the nutritional stress response in *E. coli*," in *Thirteen International Conference on Intelligent Systems for Molecular Biology*, 2005.

[7] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino, "A greedy approach to identification of piecewise affine models," in *Hybrid Systems : Computation and Control*, ser. Lecture Notes in Computer Science, O. M. O and A. Pnueli, Eds. Springer, 2003, pp. 97–112.

[8] 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," *Journal of Theoretical Biology*, vol. 229, no. 3, pp. 339–347, 2004.

[9] 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. Springer, 1990, vol. 531, pp. 197–203.

[10] 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.

[11] G. Brajnik and D. Clancy, "Focusing qualitative simulation using temporal logic: theoretical foundations," *Annals of Mathematics and Artificial Intelligence*, vol. 22, no. 1-2, pp. 59–86, 1998.

[12] N. Chabrier-Rivier, M. Chiaverini, V. Danos, F. Fages, and V. Schächter, "Modeling and querying biomolecular interaction networks," *Theoretical Computer Science*, vol. 325, no. 1, pp. 25–44, 2004.

[13] 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.

[14] 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.

[15] 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.

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

[17] J. Davoren, V. Coulthard, N. Markey, and T. Moor, "Non-deterministic temporal logics for general flow systems," in *The 7th International Workshop on Hybrid Systems: Computation and Control*, 2004, pp. 280–295.

[18] S. Eker, M. Knapp, K. Laderoute, P. Lincoln, and C. Talcott, "Pathway logic: Executable models of biological networks," *Electronic Notes in Theoretical Computer Science*, vol. 71, 2002.

[19] E. A. Emerson, "Temporal and modal logic," in *Handbook of Theoretical Computer Science: Formal Models and Semantics*, J. van Leeuwen, Ed. North-Holland Pub. Co./MIT Press, 1990, vol. B, pp. 995–1072.

[20] 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.

[21] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari, "A clustering technique for the identification of piecewise affine systems," *Automatica*, vol. 39, no. 2, pp. 205–217, 2003.

[22] K. Fukuda, "cdd/cdd+ package," URL http://www.cs.mcgill.ca/∼fukuda/soft/cdd_home/cdd.html.

[23] T. Gardner, Personal communication, Boston University, 2006.

[24] 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.

[25] A. Juloski, S. Wieland, and W. P. M. H. Heemels, "A bayesian approach to identification of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 50, no. 10, pp. 1520–1533, 2005.

[26] P. C. Kanellakis and S. A. Smolka, "CCS expressions, finite-state processes, and three problems of equivalence," *Inform. Computat.*, vol. 86, pp. 43–68, 1990.

[27] J. Kapinski and B. Krogh, "Verifying switched-mode computer controlled systems," in *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, 2002, pp. 98 – 103.

[28] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from LTL specifications," in *Hybrid Systems: Computation and Control: 9th International Workshop*, ser. Lecture Notes in Computer Science, J. Hespanha and A. Tiwari, Eds. Springer Berlin / Heidelberg, 2006, vol. 3927, pp. 333 – 347.

[29] ——, "LTL planning for groups of robots," in *IEEE International Conference on Networking, Sensing, and Control*, Ft. Lauderdale, FL, 2006.

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

[31] J. N. Lin and R. Unbehauen, "Canonical piecewise-linear approximaions," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 39, no. 8, pp. 697–699, 1992.

[32] 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.

[33] T. Motzkin, H.Raiffa, G. Thompson, and R.M.Thrall, "The double description method," in *Contributions to Theory of Games*, H. Kuhn and A. Tucker, Eds. Princeton University Press, 1953, vol. 2.

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

[35] M. M. Quottrup, T. Bak, and R. Izadi-Zamanabadi, "Multi-robot motion planning: A timed automata approach," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, 2004, pp. 4417–4422.

[36] J. Roll, A. Bemporad, and L. Ljung, "Identification of piecewise affine systems via mixed-integer programming," *Automatica*, vol. 40, no. 1, pp. 37–50, 2004.

[37] B. Shults and B. Kuipers, "Proving properties of continuous systems: Qualitative simulation and temporal logic," *Artificial Intelligence*, vol. 92, no. 1-2, pp. 91–130, 1997.

[38] B. Silva and B. Krogh, "Modeling and verification of hybrid systems with clocked and unclocked events," in *Proceedings of the 40th IEEE Conference on Decision and Control, 2001*, vol. 1, 2001, pp. 762–767.

[39] P. Tabuada and G. Pappas, "Model checking LTL over controllable linear systems is decidable," ser. Lecture Notes in Computer Science, O. Maler and A. Pnueli, Eds. Springer, 2003, vol. 2623.

[40] R. Vidal, S. Soatto, Y. Ma, and S. Sastry, "An algebraic geometric approach to the identification of a class of linear hybrid systems," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, 2003.