Joint 48th IEEE Conference on Decision and Control and
28th Chinese Control Conference
Shanghai, P.R. China, December 16-18, 2009

Temporal Logic Control of Discrete-Time Piecewise Affine Systems

Boyan Yordanov and Calin Belta

***Abstract*— We consider the problem of controlling a discrete-time piecewise affine (PWA) system from a specification given as a Linear Temporal Logic (LTL) formula over linear predicates in its state variables. We present a computational framework for finding initial states and feedback control strategies guaranteeing the satisfaction of such a specification by all the trajectories of the closed loop system. Our solution is based on abstracting the system to a finite transition system and on controlling the abstraction from an LTL specification.**

I. Introduction

Temporal logics and model checking [6] are customarily used for specifying and verifying the correctness of digital circuits and computer programs. However, due to their resemblance to natural language, expressivity, and existence of off-the-shelf algorithms for model checking, temporal logics have the potential to impact several other areas. Examples include analysis of systems with continuous dynamics [7], control of linear systems from temporal logic specifications [22], [14], task specification and controller synthesis in mobile robotics [17], [8], [15] and specification and analysis of qualitative behavior of genetic networks [2], [4], [3].

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 proven to be equivalent with several other classes of hybrid systems [11]. In addition, there exist computationally efficient 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 [12] for a review).

We consider the following problem: given a discrete-time PWA system with polytopic control constraints, and a specification in the form of a Linear Temporal Logic (LTL) formula over linear predicates in its state variables, find initial states and feedback control strategies such that all trajectories of the closed loop system satisfy the specification. Our approach is based on partitioning both the state space and the control space of the system to construct an abstraction in the form of a finite transition system. The states and inputs of the abstraction are equivalence classes induced by state and input space partitions. A state feedback control strategy is constructed for the abstraction by using techniques inspired from LTL model checking and Büchi games, and it is then adapted to the initial PWA system. The particular partitioning of the input space guarantees language equivalence between the closed-loop abstraction and the closed-loop PWA system. Although the method is conservative, the feedback control strategy specifies a set of allowed control values at a region in the state space, and it is therefore robust with respect to uncertainty in state measurements and applied controls. The computation involves polyhedral operations, which can be performed efficiently [16], and control of transitions systems from LTL specifications, which we developed in our previous work [14], [18].

This paper can be seen in the context of literature focused on the construction of finite quotients of infinite systems (see [1] for an earlier review), and is related to [20], [22], [14]. The embedding of discrete-time systems into transition systems is inspired from [20], [22], where the existence of bisimulation quotients and control strategies under the assumption of controllability for linear systems is characterized. In this work, we focus instead on developing algorithmic procedures for the computation of quotients and control strategies for the more general class of PWA systems. The related problem of controlling Mixed Logical Dynamical (MLD) systems has been considered in [13] by representing LTL specifications as mixed-integer linear constraints but a finite horizon assumption is imposed. The main contribution of this paper is to show that, for PWA systems, finite quotients can be constructed through polyhedral operations only and can be used for designing control strategies in lieu of the original (infinite) PWA system. This paper extends recent results on formal analysis of PWA systems [9], [24], [23] to a control framework.

The method presented in this paper was implemented as the user friendly software package `conPAS` and is freely available at http://iasi.bu.edu/software. The remainder of the paper is organized as follows. In Sec. II we provide some preliminaries used throughout the paper. The problem is formulated in Sec. III. In Sec. IV we define the control transition system (Sec. IV-A), describe its computation (Sec. IV-B) and show how it can be used to generate the control strategy (Sec. IV-C). In Sec. V we briefly describe the implementation of the method and present results from its application.

II. Notation and Preliminaries

A. Polytopes

A full dimensional *polytope* $\mathcal{X}$ in $\mathbb{R}^N$ can be described in either its V-representation $\mathcal{X} = hull\{v \mid v \in \mathcal{V}(\mathcal{X})\}$ or

This work was partially supported by NSF under grants 0834260 and CAREER 0447721 at Boston University.

B. Yordanov is with the Dept. of Biomedical Engineering, Boston University `yordanov@bu.edu`

C. Belta is with the Dept. of Mechanical Engineering and the Division of Systems Engineering, Boston University `cbelta@bu.edu`

978-1-4244-3872-3/09/$25.00 ©2009 IEEE

its H-representation $\mathcal{X} = \{x \in \mathbb{R}^N \,|\, Hx \leq K\}$, where $hull$ denotes the convex hull, $\mathcal{V}(\mathcal{X})$ represents the set of vertices of $\mathcal{X}$, and H, K are matrices of appropriate dimensions. Given a full dimensional polytope $\mathcal{X}$, there exist algorithms for translation between its V- and H-representations [19]. Given a matrix $A \in \mathbb{R}^{N\times N}$ and a polytope $\mathcal{X}$ we use $A\mathcal{X}$ to denote the image of $\mathcal{X}$ through A *i.e.* $A\mathcal{X} = hull\{Av \,|\, v \in \mathcal{V}(\mathcal{X})\}$.

B. Transition Systems

Definition 1: A transition system is a tuple $T = (Q, \Sigma, \delta, O, o)$, where Q and Σ are (possibly infinite) sets of states and inputs, $\delta : Q \times \Sigma \to 2^Q$ is a (nondeterministic) transition map (2^Q is the powerset of Q), O is a set of observations and $o : Q \to O$ is an observation map.

A subset of the state set $X \subseteq Q$ is called a *state region* and a subset of the input set $U \subseteq \Sigma$ is called an *input region* of T. A (nondeterministic) transition $\delta(x, u) = X'$ indicates that, while the system is in state x it can make a transition to any state x' in region X' under input u. We denote the set of inputs available at a state $x \in Q$ by Σ^x. A transition $\delta(x, u) = X'$ is *deterministic* if the set X' is a singleton and the transition system T is deterministic if all its transitions are deterministic. Transition system T is *finite* if both its set of states Q and set of inputs Σ are finite. T is *non-blocking* if, for every state $x \in Q$, there exists $X' \subseteq Q$ and $u \in \Sigma$ such that $\delta(x, u) = X'$.

An *input word* of the system is defined as an infinite sequence $u_1, u_2, u_3, \ldots$, where $u_i \in \Sigma$. A *trajectory* or *run* of T produced by input word $u_1, u_2, u_3, \ldots$ and starting from state $x_1 \in Q$ is an infinite sequence $x_1, x_2, x_3 \ldots$ with the property that $x_i \in Q$, and $x_{i+1} \in \delta(x_i, u_i)$, for all $i \geq 1$. A trajectory of the system $x_1, x_2, x_3 \ldots$ defines a *word* $o(x_1), o(x_2), o(x_3) \ldots$. The set of all words generated by the set of all trajectories starting at state $x \in Q$ is called the *language* of T originating at x and is denoted by $\mathcal{L}_T(x)$ (similarly, we use $\mathcal{L}_T(X)$ to denote the language of T originating in region X).

For an arbitrary state region X and input region U, we define the set of states $Post_T(X, U)$ that can be reached from X in one step by applying an input in U as

$$Post_T(X, U) = \{x' \in Q \,|\, \exists x \in X,\ \exists u \in U,\ x' \in \delta(x, u)\} \tag{1}$$

The observation map o of a transition system T induces an 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 $o(x_1) = o(x_2)$. Then, the equivalence relation naturally induces a *quotient* transition system $T/_\sim = (Q/_\sim, \Sigma, \delta_\sim, O, o_\sim)$, where $Q/_\sim$ is the set of all equivalence classes formed in Q and transitions are defined as $X' \in \delta_\sim(X, u)$ if and only if there exist $x \in X$ and $x' \in X'$ such that $x' \in \delta(x, u)$ (we abuse the notation and use the symbols X, X' to indicate states in $Q/_\sim$ as well as regions of Q but the precise meaning is clear from the context). The sets of inputs Σ and observations O of $T/_\sim$ are preserved from T. 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 $o_\sim(X) = o(x)$, $x \in X$.

C. LTL, Model Checking, and LTL Control

To specify temporal logic properties for trajectories of PWA systems, in this paper we use Linear Temporal Logic [6]. Informally, LTL formulas are recursively defined over the set of observations O, by using the standard Boolean operators (*e.g.,* $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction)) and temporal operators, which include $\bigcirc$ ("next"), $\mathcal{U}$ ("until"), $\Box$ ("always"), $\Diamond$ ("eventually"). LTL formulas are interpreted over infinite words, as are those generated by the transition system T from Definition 1. Given a finite transition system $T = (Q, \Sigma, \delta, O, o)$ and an LTL formula ϕ over O, checking whether the words of T starting from each state satisfy ϕ is called LTL model checking, or simply model checking in this paper. An off-the-shelf model checker, such as `NuSMV` [5], takes as input a finite transition system T and a formula ϕ and returns the states of T, at which the formula is satisfied (*i.e.,* the states for which the language originating there satisfies the formula). For the non-satisfying states, a model checker returns a non-satisfying run as a certifying counter-example. We write $T(X) \vDash \phi$ if $\mathcal{L}_T(X)$ satisfies ϕ.

As a dual to the model checking problem, one can formulate an LTL control problem: for a finite transition system T, find a set of initial states X_0 and a control strategy such that $T(X_0) \vDash \phi$. If T is deterministic, then off-the-shelf model checking can, in principle, be used to solve the control problem. Indeed, one can model check T from every initial state against $\neg\phi$. If there is a violating run at a state (*i.e.,* a run satisfying ϕ), it is returned as a counter-example by the model checker. Since the system is deterministic, a sequence of inputs producing the run can be found, which will give the control strategy. We followed this approach in [14], where we redesigned the model checking process such that the produced runs were optimal with respect to a predefined cost (this method has been implemented in a tool called `LTLCon`[1]). If T is non-deterministic, the problem is more difficult. In [18], we proposed a solution inspired from Büchi games and implemented it as the software tool `BüCon`. The control strategy takes the form of a "feedback automaton", which reads the current state of T and produces the control to be applied at that state. The solution to this control problem is complete if the specification is restricted to an LTL formula whose satisfying language is generated by a deterministic Büchi automaton.

III. Problem Formulation and Approach

Let $\mathcal{X}, \mathcal{X}_l$, $l \in L$ 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 $\bigcup_{l\in L} \bar{\mathcal{X}}_l = \bar{\mathcal{X}}$, where $\bar{\mathcal{X}}$ is the closure of $\mathcal{X}$.

[1]In [14], `LTLCon` was designed for the synthesis of control strategies for continuous time, linear systems, where the LTL_X segment of LTL was used to formulate specifications. In this work, we apply `LTLCon` directly to finite transition systems and formulate specification over the full LTL

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

$$x_{k+1} = A_l x_k + B_l u_k + c_l, \qquad (2)$$
$$x_k \in \mathcal{X}_l,\ u_k \in \mathcal{U},\ l \in L,\ k = 0, 1, 2, \ldots,$$

where $x_k \in \mathbb{R}^N$ is the state of the system, u_k is the control input restricted to a polytopic set $\mathcal{U} \subseteq \mathbb{R}^M$, and $A_l \in \mathbb{R}^{N\times N}$, $B_l \in \mathbb{R}^{N\times M}$, $c_l \in \mathbb{R}^N$ are the system parameters for each mode $l \in L$. We assume that $\mathcal{X}$ is an invariant for the trajectories of (2). In Sec. IV-B we will show that polyhedral control constraints guaranteeing this condition can be computed.

We are interested in properties of (2) specified in terms of the polytopes from its definition. Intuitively, a trajectory of the system $x_1 x_2 x_3 \ldots$ where $x_k \in \mathcal{X}_{l_k}$ produces an infinite word $l_1, l_2, l_3, \ldots$ by listing the index of the polytope visited at each step. An LTL formula can then be interpreted over trajectories of system (2).

We consider the following problem:

Problem 1: Given a PWA system (2) and an LTL formula ϕ over L, find a set $X_0 \subseteq \mathcal{X}$ and a control strategy, such that all trajectories of the closed loop system originating in X_0 satisfy formula ϕ.

Formally, we define the satisfaction of LTL formulas by the trajectories of system (2) through an embedding into a transition system:

Definition 2: The embedding transition system $T_e = (Q_e, \Sigma_e, \delta_e, O_e, o_e)$ for system (2) is defined as:

- $Q_e = \bigcup_{l\in L} \mathcal{X}_l$,
- $\Sigma_e = \mathcal{U}$,
- $\delta_e(x, u) = x'$ if and only if there exist $l \in L$ and $u \in \mathcal{U}$ such that $x \in \mathcal{X}_l$ and $x' = A_l x + B_l u + c_l$,
- $O_e = L$,
- $o_e(x) = l$ if and only if $x \in \mathcal{X}_l$.

The embedding transition system T_e is non-blocking and deterministic but both its set of states Q_e and set of inputs Σ_e are infinite.

Definition 3: Trajectories of system (2) originating in X_0 satisfy formula ϕ if and only if $T_e(X_0)$ satisfies ϕ.

Remark 1: We capture only the reachability of open full dimensional polytopes in the semantics of the embedding. 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. In addition, the specification is given over the polytopes $\mathcal{X}_l$ but arbitrary linear inequalities can be accommodated simply by refining the polytopic partition.

If T_e was finite and deterministic then Problem 1 can be solved by applying the software tool `LTLCon` as described in Sec. II-C. If, on the other hand, T_e was finite but nondeterministic a solution can be obtained by applying `BüCon` (see Sec. II-C). Since T_e is infinite, neither of the two approaches can be used directly. Our approach (illustrated in Fig. 1) is based on the construction of a finite control transition system T_c, the generation of a control strategy for T_c, and the adaptation of the control strategy to the original PWA system (or, equivalently, the infinite embedding T_e). The two closed loop systems will produce the same language, and therefore will be equivalent with respect to the satisfaction of the specification. The method is conservative and finding a solution is not guaranteed but if a satisfying control strategy is found, it is robust with respect to uncertainty in state measurements and applied controls.

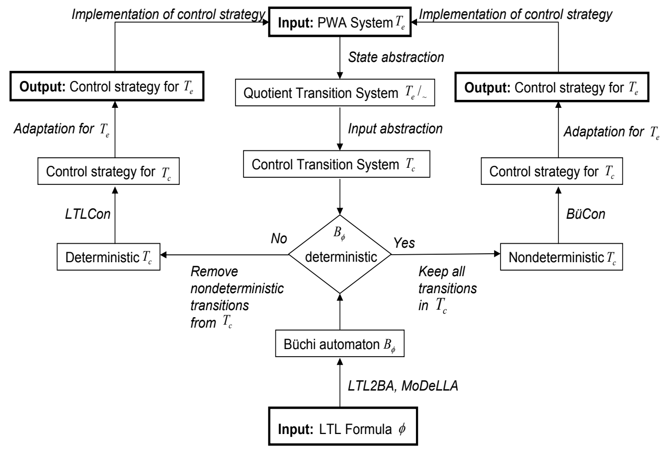


Fig. 1. Overview of our approach to Problem 1

IV. CONTROL STRATEGY

In this section, we first define the control transition system (Sec. IV-A). Then, we show that its computation reduces to polyhedral operations only (Sec. IV-B). Finally, we develop a control strategy for the control transition system and adapt it to the original PWA system (Sec. IV-C).

A. Control Transition System

Let $\sim$ denote the equivalence relation induced on Q_e by the output map o_e. Let $T_e/_\sim = (Q_e/_\sim, \Sigma_e, \delta_{e\sim}, O_e, o_{e\sim})$ denote the corresponding quotient transition system (see Sec. II-B). Note that, while the set of states $Q_e/_\sim$ of $T_e/_\sim$ is finite ($Q_e/_\sim = L$), the set of inputs Σ_e is infinite. Also note that $T_e/_\sim$ is, in general nondeterministic, even though T_e is deterministic.

The transition map $\delta_{e\sim}$ can be related to the transitions of T_e by using the $Post$ operator defined in Eqn. (1):

$$\delta_{e\sim}(l, u) = \{l' \in Q_e/_\sim \mid Post_{T_e}(\mathcal{X}_l, u) \cap \mathcal{X}_{l'} \neq \emptyset\}, \quad (3)$$

for all $l \in Q_e/_\sim$ and $u \in \Sigma_e$.

For each state $l \in Q_e/_\sim$, we define an equivalence relation $\approx_l$ over the set of inputs Σ_e as follows:

$$(u_1, u_2) \in \approx_l \text{ iff } \delta_{e\sim}(l, u_1) = \delta_{e\sim}(l, u_2) \qquad (4)$$

In other words, inputs u_1 and u_2 are equivalent at l if they produce the same set of transitions in $T_e/_\sim$. Let U_l^S, $l \in L$, $S \subseteq Q_e/_\sim$ denote the equivalence classes (regions) of Σ_e in the partition induced by the equivalence relation $\approx_l$:

$$U_l^S = \{u \in \Sigma_e \mid \delta_{e\sim}(l, u) = S\} \qquad (5)$$

We are now ready to define the control transition system:

Definition 4: The control transition system $T_c = (Q_c, \Sigma_c, \delta_c, O_c, o_c)$ is defined as:

- $Q_c = Q_e/_\sim = L$;
- $\Sigma_c \subseteq \{U_l^S \mid l \in L, S \subseteq Q_c\}$. For any $l \in Q_c$, $U_l^S \in \Sigma_c^l$ if and only if
  - U_l^S is large enough,
  - $Post_{T_e}(\mathcal{X}_l, U_l^S) \subseteq \mathcal{X}$,
  - For any S' such that $S' \subseteq S$, $U_l^{S'} \notin \Sigma_c^l$;
- $\delta_c(l, U_l^S) = S$, for $U_l^S \in \Sigma_c$;
- $O_c = O_e$;
- $o_c = o_{e\sim}$.

In other words, the control transition system T_c is the same as the quotient transition system $T_e/_\sim$, with the exception of the set of inputs and the set of transitions, which are both finite for T_c. The set of inputs Σ_c^l available at a state $l \in Q_c$ is a subset of $\{U_l^S, S \subset Q_c\}$, which is a finite set. Sets U_l^S that are empty or too "small" (see Sec. IV-B) are excluded to reduce the size of the system and to make it more robust. In addition, each control u from an allowed set U_l^S at state l should keep T_e inside $\mathcal{X}$. This guarantees that $\mathcal{X}$ is an invariant for the trajectories of the PWA system, which was the assumption we made at the beginning of Sec. III. Finally, we do not include in T_c a nondeterministic transition to a set of states S for which a transition to a subset $S' \subseteq S$ exists. This assumption is motivated by the `BüCon` algorithm for the control of a nondeterministic transition system. Roughly, more "nondeterminism" available at a state does not result into more winning strategies for the game algorithm that we described in [18].

B. Computation of the Control Transition System

The states of the control transition system T_c are simply the labels $l \in L$ of the polytopes from the definition of the PWA system (Eqn. (2)). To complete its construction, we need to be able to compute the set of controls Σ_c^l available at each state $l \in Q_c$. In this section, we show that this can be achieved through polyhedral operations. Given two polytopes $\mathcal{X}_l$ and $\mathcal{X}'$, where $\mathcal{X}_l$ is a polytope from the definition of the PWA system (Eqn. (2)) and $\mathcal{X}'$ is an arbitrary polytope, let

$$U^{\mathcal{X}_l \Rightarrow \mathcal{X}'} = \{u \in \Sigma_e \mid Post_{T_e}(\mathcal{X}_l, u) \subseteq \mathcal{X}'\} \qquad (6)$$

be the set of all inputs guaranteeing that all states from $\mathcal{X}_l$ transit inside $\mathcal{X}'$. In other words, regardless which $u \in U^{\mathcal{X}_l \Rightarrow \mathcal{X}'}$ and $x \in \mathcal{X}_l$ are selected, x will transit inside $\mathcal{X}'$ under u in T_e. Similarly, let

$$U^{\mathcal{X}_l \rightarrow \mathcal{X}'} = \{u \in \Sigma_e \mid Post_{T_e}(\mathcal{X}_l, u) \cap \mathcal{X}' \neq \emptyset\} \qquad (7)$$

denote the set of all inputs under which T_e can make a transition from a state in $\mathcal{X}_l$ to a state inside $\mathcal{X}'$. Equivalently, applying any input $u \in \mathcal{U}, u \notin U^{\mathcal{X}_l \rightarrow \mathcal{X}'}$ guarantees that T_e will not make a transition inside $\mathcal{X}'$, from any state in $\mathcal{X}_l$.

Let $\mathcal{X}' = \{x \in \mathbb{R}^N \mid Hx \leq K\}$. Then, $U^{\mathcal{X}_l \Rightarrow \mathcal{X}'}$ is a polytope with the following H-representation:

$$U^{\mathcal{X}_l \Rightarrow \mathcal{X}'} = \{u \in \mathcal{U} \mid HB_l u \leq K - H(A_l v + c_l), \forall v \in \mathcal{V}(\mathcal{X}_l)\} \qquad (8)$$

Let H and K be the matrices in the H-representation of the following polytope:

$$\{\hat{x} \in \mathbb{R}^N \mid \exists x \in \mathcal{X}_l, A_l x + \hat{x} + c_l \in \mathcal{X}'\} \qquad (9)$$

Then $U^{\mathcal{X}_l \rightarrow \mathcal{X}'}$ is a polytope with the following H-representation:

$$U^{\mathcal{X}_l \rightarrow \mathcal{X}'} = \{u \in \mathcal{U} \mid HB_l u \leq K\} \qquad (10)$$

Proposition 1: Given a state $l \in Q_c$ and a set of states $S \subseteq Q_c$, the set U_l^S from Eqn. (5) can be computed as follows:

$$U_l^S = \bigcap_{l' \in S} U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l'}} \setminus \bigcup_{l'' \notin S} U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l''}} \qquad (11)$$

For $l, l' \in Q_c$, the computation of the set of inputs $U_l^{l'}$, inducing a deterministic transition from l to l', reduces to

$$U_l^{l'} = U^{\mathcal{X}_l \Rightarrow \mathcal{X}_{l'}} \qquad (12)$$

In order to guarantee that $\mathcal{X}$ is an invariant for all trajectories of the system it is sufficient to restrict the inputs available at each state $l \in Q_c$ as follows:

$$\Sigma_c^l \subseteq U^{\mathcal{X}_l \Rightarrow \mathcal{X}} \qquad (13)$$

Then, the set of states reachable from state l under the allowed inputs is

$$Post_{T_e/_\sim}(l, \Sigma_c^l) = \{l' \in Q_e/_\sim \mid Post_{T_e}(\mathcal{X}_l, \Sigma_c^l) \cap \mathcal{X}_{l'} \neq \emptyset\} \qquad (14)$$

and can be computed using polyhedral operations, since

$$Post_{T_e}(\mathcal{X}_l, \Sigma_c^l) = A_l \mathcal{X}_l + B_l U^{\mathcal{X}_l \Rightarrow \mathcal{X}} + c_l \qquad (15)$$

We can guarantee that if a state l' is not reachable from state l (*i.e.* $l' \notin Post_{T_e/_\sim}(l, \Sigma_c^l)$) then $U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l'}} = \emptyset$ and therefore, $U_l^S = \emptyset$ if $S \nsubseteq Post_{T_e/_\sim}(l, \Sigma_c^l)$ and otherwise the computation in Eqn. (11) reduces to

$$U_l^S = \bigcap_{l' \in S} U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l'}} \setminus \bigcup_{l'' \in Post_{T_e/_\sim}(l, \Sigma_c^l) \setminus S} U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l''}} \qquad (16)$$

All results presented in this section are summarized in Algorithm 1, which constructs the set of inputs Σ_c of the control transition system T_c and can be implemented using polyhedral operations.

In order to construct the control transition system T_c in accordance with Definition 4, we include the following operations in Algorithm1:

- On line 2 we ensure that only inputs guaranteeing the invariance of polytope $\mathcal{X}$ are allowed for each state.
- On line 3 we ignore input regions which are known to be empty or do not lead to additional satisfying strategies.
- On line 11 we only include in Σ_c input regions that are large enough. A non-empty input region U_l^S can always be represented as a union of polytopes (see Eqn. (16)). Then, only the polytopes from U_l^S with radii of inscribed spheres larger than a certain, predefined limit are kept. If no polytopes satisfy this condition, the region U_l^S is considered empty and excluded from Σ_c. Otherwise, applying the center of the inscribed sphere of any polytope in U_l^S in a control strategy (as it will become clear in Sec. IV-C) guarantees its robustness.

Algorithm 1 Construct Σ_c

for each $l \in Q_c$ **do**
  Initialize $\Sigma_c^l = U^{\mathcal{X}_l \Rightarrow \mathcal{X}}$
  for each $S \subseteq Post_{T_e/\sim}(l, \Sigma_c^l)$ such that $U_l^{S'} \notin \Sigma_c$ for any $S' \subseteq S$ **do**
    if $|S| == 1$ (*i.e.* $S = l'$) **then**
      $U_l^S = \Sigma_c^l \cap U^{\mathcal{X}_l \Rightarrow \mathcal{X}_{l'}}$
    else
      $U_1 = \bigcap_{l' \in S} U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l'}}$
      $U_2 = \bigcup_{l'' \in Post_{T_e/\sim}(l, \Sigma_c^l) \setminus S} U^{\mathcal{X}_l \rightarrow \mathcal{X}_{l''}}$
      $U_l^S = \Sigma_c^l \cap (U_1 \setminus U_2)$
    end if
    if U_l^S is large enough **then**
      Include U_l^S in Σ_c
    end if
  end for
end for

C. Control

So far in Sec. IV we have defined the control transition system T_c (Sec. IV-A) and showed that it can be efficiently constructed using only polyhedral operations (Sec. IV-B). In this section, we formulate a solution to Problem 1 by generating a control strategy for T_c and adapting it to the embedding T_e or, equivalently, the original PWA system.

The control transition system T_c is always finite but, in general, nondeterministic. Then, the software tool `BüCon` (see Sec. II-C) can be used to generate a control strategy for T_c, as long as the LTL specification ϕ can be translated into a deterministic Büchi automaton (possibly by using tools such as `LTL2BA` [10] or `MoDeLLA` [21]). Since the existence of a deterministic Büchi automaton cannot be guaranteed, the application of `BüCon` might be restrictive with respect to the specifications that can be formulated. In order to handle specifications resulting in nondeterministic Büchi automata, `LTLCon` can be applied, but only if T_c is deterministic. In Sec. IV-B (Eqn. (12)), we showed that the computation of deterministic inputs in Σ_c can be efficiently performed and therefore a deterministic T_c can be constructed (alternatively, a nondeterministic T_c can always be determinized by removing all nondeterministic inputs and transitions). Then, `LTLCon` can be applied to a deterministic T_c to generate a control strategy when the specification can only be translated into a nondeterministic Büchi automaton, but the overall method becomes more conservative.

Applying `BüCon` or `LTLCon` will result in a set of initial states $L_0 \subseteq Q_c$ ($L_0 \subseteq L$) and a control strategy for T_c. We can translate L_0 into a set of initial regions in T_e directly as $X_0 = \bigcup_{l \in L_0} \mathcal{X}_l$. The control strategy generated by `BüCon` and `LTLCon` will take the form of a feedback control automaton $\mathcal{C}()$, which reads the current state of T_c and provides the next input to be applied (*i.e.* $\mathcal{C}(l) \in \Sigma_c^l$). An infinite word $l_1, l_2, l_3, \ldots$, where $l_k \in Q_c$, generated by input word $U_1, U_2, U_3 \ldots$, where $U_k \in \Sigma_c^{l_k}$ is guaranteed to satisfy the LTL specification ϕ, as long as $l_1 \in L_0$ and $U_k = \mathcal{C}(l_k)$ for each $k = 1, 2, \ldots$.

From the definition of T_c it follows that

$$\forall x \in \mathcal{X}_l, \forall u \in U_l^S, \delta_e(x, u) \in \mathcal{X}_{l'}, l' \in S \qquad (17)$$

and therefore a trajectory $x_1, x_2, x_3, \ldots$, where $x_k \in Q_e$, generated by input word $u_1, u_2, u_3, \ldots$, where $u_k \in \Sigma_e$, will produce a word $l_1, l_2, l_3, \ldots$ that satisfies ϕ as long as $x_1 \in X_0$, where X_0 is the set of satisfying initial regions and $u_k \in \mathcal{C}(o_e(x_k))$. As discussed in Sec. IV-B, only input regions with radii of inscribed spheres larger than a predefined limit have been included in Σ_c. Then, taking u_k to be the center of the inscribed sphere of the input region specified by $\mathcal{C}(o_e(x_k))$ (or any of the centers in the case when the control region is given as a union of polytopes) defines the PWA control strategy and guarantees its robustness, which solves Problem 1. Note that under the control strategy defined above, the closed loop control transition system and the closed loop PWA system have the same language.

Remark 2: In order to understand the complexity of the proposed method we focus only on the complexity of constructing the control transition system (for a discussion on the complexities associated with `LTLCon` and `BüCon` see [14] and [18]). Although, the theoretical worst case complexity for the construction of T_c is $O(2^{|Q_c|})$, on average this can be significantly reduced through the optimizations described in Sec. IV-B. Even so, the method is computationally intensive and might not be applicable to PWA systems defined over many polytopes (when $|Q_c|$ is large). In the case study described in Sec. V the construction of T_c required less than 15 minutes on a 2.66GHz Intel Xeon Quad-Core machine with 3 GB memory. Once T_c was constructed, generating the control strategy required an additional 20 seconds.

V. Implementation and Case Study

The method described in this paper was implemented in MATLAB as the user friendly software package `conPAS`, where all polyhedral operations were performed by the MPT toolbox [16]. The tool takes as input a PWA system (as defined in Eqn. (2)) and an LTL formula, produces a set of satisfying initial regions and a feedback control strategy for the system and can simulate runs in the closed loop system. Since inputs in T_c are constructed locally at each state, the computation can be distributed efficiently on multi-processor platforms. The tool is made public and freely downloadable at http://iasi.bu.edu/software.

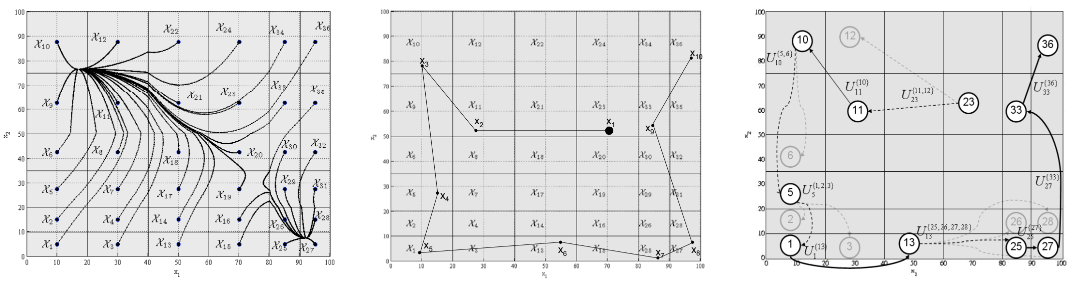
A PWA system of dimensions $N = M = 2$, defined on polytopes $\mathcal{X}_1, \ldots, \mathcal{X}_{36}$ (shown in Fig. 2) was analyzed with `conPAS`.

The specification of interest was

$$\phi = \Diamond \mathcal{X}_1 \wedge \Diamond \mathcal{X}_{10} \wedge \Diamond \mathcal{X}_{27} \wedge \Diamond \mathcal{X}_{36} \qquad (18)$$

or "*eventually visit all corner regions in any order*" and was translated into a 16 state deterministic Büchi automaton.

A control transition system with 36 states was constructed. Out of the total 4115 nonempty input regions found, 3182 were large enough (the radii of their inscribed spheres were



A B C

Fig. 2. A) Trajectories of the uncontrolled system go towards regions $\mathcal{X}_{10}$ and $\mathcal{X}_{27}$. B) A simulated trajectory $(x_1, \ldots, x_{10})$ of the closed loop system. C) The control strategy used to generate the trajectory shown in B relied on 4 nondeterministic (dotted lines) and 5 deterministic (solid lines) transitions. States that were not visited by the trajectory but were targets of the nondeterministic transitions are also shown.

larger 0.05) but only 691 were included in T_c (due to the availability of "more deterministic" transitions as described in Sec. IV-A). Considering only the subset of 260 deterministic transitions in T_c did not lead to a satisfying control strategy from any initial region ($X_0 = \emptyset$), while satisfying control strategies were found for all regions ($X_0 = \mathcal{X}$) when nondeterministic transitions were also included.

Starting from random initial conditions $x_1 \in \mathcal{X}_{23}$, a trajectory of the closed loop system was simulated until the specification was satisfied (Fig. 2-B). At each step $k = 1, \ldots, 10$ where $x_k \in \mathcal{X}_{l_k}$ an input region $U^S_{l_k}$ was specified by the control automaton. Then, the next state x_{k+1} was generated by applying the center of the inscribed sphere of any polytope in $U^S_{l_k}$ as the input u_k. The trajectory of the control transition system corresponding to the simulation and the inputs applied at each step are shown in Fig. 2-C.

REFERENCES

[1] 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.

[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] G. Batt, C. Belta, and R. Weiss, "Temporal logic analysis of gene networks under parameter uncertainty," *IEEE Trans. on Circuits and Systems and IEEE Trans. on Automatic Control, joint special issue on Systems Biology*, vol. 53, pp. 215–229, 2008.

[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. 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.

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

[7] 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.

[8] 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.

[9] G. Frehse, S. K. Jha, and B. Krogh, "A counter-example giuded approach to parameter synthesis for linear hybrid automata," in *Hybrid Systems: Computation and Control: 11th International Workshop*, ser. Lecture Notes in Computer Science, M. Egerstedt and B. Mishra, Eds. Springer Berlin / Heidelberg, 2008, pp. 187–200.

[10] 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.

[11] 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.

[12] 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," *Lecture Notes in Computer Science*, vol. 3414/2005, pp. 354–369, 1993.

[13] S. Karaman, R. G. Sanfelice, and E. Frazzoli, "Optimal control of mixed logical dynamical systems with linear temporal logic specifications," in *Proceedings of the 47th IEEE Conference on Decision and Control*, 2008, pp. 2117–2122.

[14] 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.

[15] H. Kress-Gazit, D. Conner, H. Choset, A. Rizzi, and G. Pappas, "Courteous cars," *IEEE Robotics and Automation Magazine*, vol. 15, pp. 30–38, March 2008.

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

[17] 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.

[18] M.Kloetzer and C. Belta, "Dealing with non-determinism in symbolic control," in *Hybrid Systems: Computation and Control: 11th International Workshop*, ser. Lecture Notes in Computer Science, M. Egerstedt and B. Mishra, Eds. Springer Berlin / Heidelberg, 2008, pp. 287–300.

[19] 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.

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

[21] R. Sebastiani and S. Tonetta, ""more deterministic" vs. "smaller" büchi automata for efficient ltl model checking," in *Correct Hardware Design and Verification Methods*, ser. Lecture Notes in Computer Science, G. Goos, J. Hartmanis, , and J. van Leeuwen, Eds., vol. 2860. Springer, 2003, pp. 126–140.

[22] 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.

[23] B. Yordanov and C. Belta, "Parameter synthesis for piecewise affine systems from temporal logic specifications," in *Hybrid Systems: Computation and Control: 11th International Workshop*, ser. Lecture Notes in Computer Science, M. Egerstedt and B. Mishra, Eds. Springer Berlin / Heidelberg, 2008, pp. 542–555.

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