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 feedback control strategies (Sec. IV-C). In Sec. V we briefly describe the implementation of the method and present results from its application.

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

II. NOTATION AND PRELIMINARIES

A. Polytopes

A full dimensional polytope $\mathcal{X}$ in $\mathbb{R}^N$ can be described in either its V-representation $\mathcal{X} = \text{hull}\{v \mid v \in \mathcal{V}(\mathcal{X})\}$ or 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.
its \( H \)-representation \( \mathcal{X} = \{ x \in \mathbb{R}^N \mid Hx \leq K \} \), 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., \( A\mathcal{X} = \text{hull}\{Av \mid v \in \mathcal{V}(\mathcal{X})\} \).

**B. Transition Systems**

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

A subset of the state set \( X \subseteq Q \) is called a state region \( X \). 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 \( \Sigma_x \). 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 \( \Sigma \) 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 \( x_1 \in Q \), 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 \( \text{Post}_T(X, U) \) that can be reached from \( X \) in one step by applying an input in \( U \) as

\[
\text{Post}_T(X, U) = \{ x' \in Q \mid \exists x \in X, \exists u \in U, x' \in \delta(x, u) \}
\]

(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 \( \Sigma \) 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), \( \lor \) (disjunction), \( \land \) (conjunction)) and temporal operators, which include \( \bigcirc \) ("next"), \( [\text{until}] \) ("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 \( \phi \) over \( O \), checking whether the words of \( T \) starting from each state satisfy \( \phi \) 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 \( \phi \) 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 \( T(X) \models \phi \) if \( \mathcal{L}_T(X) \) satisfies \( \phi \).

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 \( T(X_0) \models \phi \). 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 \( \phi \)), 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 \( \mathcal{X}_l \cap \mathcal{X}_l = \emptyset \) for all \( l_1, l_2 \in L \), \( l_1 \neq l_2 \) and \( \bigcup_{l \in L} \mathcal{X}_l = \mathcal{X} \), where \( \mathcal{X} \) 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:

\[ x_{k+1} = A_l x_k + B_l u_k + c_l, \]  

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 system (2) through an embedding into a dimensional polytopes in the semantics of the embedding.

Remark 1: We consider the following problem:

**Problem 1:** Given a PWA system (2) and an LTL formula \( \phi \) 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 \( \phi \).

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 \( u \in U \) 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 \( \Sigma_e \) are infinite.

**Definition 3:** Trajectories of system (2) originating in \( X_0 \) satisfy formula \( \phi \) if and only if \( T_e(X_0) \), the generation of a control strategy for \( T_e \), 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.

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

The transition map \( \delta_e/\approx_e \) can be related to the transitions of \( T_e \) by using the Post operator defined in Eqn. (1):

\[ \delta_e/\approx_e(l, u) = \{ l' \in Q_e/\approx_e \mid \text{Post}_{T_e}(X_l, u) \cap X_{l'} \neq \emptyset \}, \]  

for all \( l \in Q_e/\approx_e \) and \( u \in \Sigma_e \).

For each state \( l \in Q_e/\approx_e \), we define an equivalence relation \( \approx_l \) over the set of inputs \( \Sigma_e \) as follows:

\[ (u_1, u_2) \in \approx_l \text{ iff } \delta_e/\approx_e(l, u_1) = \delta_e/\approx_e(l, u_2) \]  

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

\[ U^S_l = \{ u \in \Sigma_e \mid \delta_e/\approx_e(l, u) = S \} \]

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:
- $Q_c = Q_e/\sim = L$;
- $\Sigma_c \subseteq \{U^S_l \mid l \in L, S \subseteq Q_e\}$. For any $l \in Q_c$, $U^S_l \in \Sigma_c^l$ if and only if
  - $U^S_l$ is large enough,
  - $Post_{T_c}(X_l, U^S_l) \subseteq X'$,
- For any $S'$ such that $S' \subseteq S$, $U^S_l \notin \Sigma_c^l$;
- $\delta_c(l, U^S_l) = S$, for $U^S_l \in \Sigma_c$;
- $O_c = O_e$;
- $o_c = o_e$.

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 $\Sigma_c^l$ available at a state $l \in Q_c$ is a subset of $\{U^S_l, S \subseteq Q_e\}$, which is a finite set. Sets $U^S_l$ that are empty or too “small” are excluded to reduce the size of the system and to make it more robust. In addition, each control $u$ from an allowed set $U^S_l$ at state $l$ should keep $T_c$ inside $X$. This guarantees that $X$ 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 Büchi 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 $\Sigma_c^l$ available at each state $l \in Q_c$. In this section, we show that this can be achieved through polyhedral operations. Given two polytopes $X_l$ and $X'$, where $X_l$ is a polytope from the definition of the PWA system (Eqn. (2)) and $X'$ is an arbitrary polytope, let

$$U^X_l \Rightarrow X' = \{u \in \Sigma_c \mid Post_{T_c}(X_l, u) \subseteq X'\}$$

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

$$U^X_l \Rightarrow X' = \{u \in \Sigma_c \mid Post_{T_c}(X_l, u) \cap X' \neq \emptyset\}$$

(7)

denote the set of all inputs under which $T_c$ can make a transition from a state in $X_l$ to a state inside $X'$. Equivalently, applying any input $u \in U, u \notin U^X_l \Rightarrow X'$ guarantees that $T_c$ will not make a transition inside $X'$, from any state in $X_l$.

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

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

(8)

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

$$\{x \in \mathbb{R}^N \mid \exists x \in X_l, Ax + \hat{x} + c_l \in X'\}$$

(9)

Then $U^X_l \Rightarrow X'$ is a polytope with the following H-representation:

$$U^X_l \Rightarrow X' = \{u \in U \mid HBu \leq K\}$$

(10)

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

$$U^S_l = \bigcap_{v \in S} U^X_l \Rightarrow X' \setminus \bigcup_{v' \notin S} U^X_l \Rightarrow X'$$

(11)

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

$$U^l_{\text{post}} = U^X_l \Rightarrow X'$$

(12)

In order to guarantee that $X$ 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:

$$\Sigma_c^l = U^X_l \Rightarrow X$$

(13)

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

$$\text{Post}_{T_c/l}(l, \Sigma_c^l) = \{l' \in Q_e/\sim \mid \text{Post}_{T_c}(X_l, \Sigma_c^l) \cap X_l' \neq \emptyset\}$$

(14)

and can be computed using polyhedral operations, since

$$\text{Post}_{T_c}(X_l, \Sigma_c^l) = A_l X_l + B_l U^X_l \Rightarrow X + c_l$$

(15)

We can guarantee that if a state $l'$ is not reachable from state $l$ (i.e., $l' \notin \text{Post}_{T_c/l}(l, \Sigma_c^l)$) then $U^X_l \Rightarrow X' = \emptyset$ and therefore, $U^S_l = \emptyset$ if $S \subseteq \text{Post}_{T_c/l}(l, \Sigma_c^l)$ and otherwise the computation in Eqn. (11) reduces to

$$U^S_l = \bigcap_{v \in S} U^X_l \Rightarrow X' \setminus \bigcup_{v' \notin \text{Post}_{T_c/l}(l, \Sigma_c^l)} U^X_l \Rightarrow X'$$

(16)

All results presented in this section are summarized in Algorithm 1, which constructs the set of inputs $\Sigma_c$ 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 Algorithm 1:

- On line 2 we ensure that only inputs guaranteeing the invariance of polytope $X$ 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 $\Sigma_c$ input regions that are large enough. A non-empty input region $U^S_l$ can always be represented as a union of polytopes (see Eqn. (16)). Then, only the polytopes from $U^S_l$ with radii of inscribed spheres larger than a certain, predefined limit are kept. If no polytopes satisfy this condition, the region $U^S_l$ is considered empty and excluded from $\Sigma_c$.
- Otherwise, applying the center of the inscribed sphere of any polytope in $U^S_l$ in a control strategy (as it will become clear in Sec. IV-C) guarantees its robustness.
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 $\phi$ 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 $\Sigma_c$ can be efficiently performed and therefore a deterministic $T_c$ can be constructed (alternatively, a nondeterministic $T_c$ can always be determined 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 $X_0 = \bigcup_{l \in L_0} X_l$. The control strategy generated by BÜCon and LTLCon will take the form of a feedback control automaton $C()$, which reads the current state of $T_c$ and provides the next input to be applied (i.e. $C(l) \in \Sigma_c$). 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 $U_k \in \Sigma^{k}_c$ is guaranteed to satisfy the LTL specification $\phi$, as long as $l_1 \in L_0$ and $U_k = C(l_k)$ for each $k = 1, 2, \ldots$.

From the definition of $T_c$ it follows that

$$\forall x \in X_l, \forall u \in U^S_1, \delta_e(x, u) \in X_p, l' \in S$$

and therefore a trajectory $x_1, x_2, x_3, \ldots$, where $x_k \in Q_c$, generated by input word $u_1, u_2, u_3, \ldots$, where $u_k \in \Sigma_c$, will produce a word $l_1, l_2, l_3, \ldots$ that satisfies $\phi$ as long as $x_1 \in X_0$, where $X_0$ is the set of satisfying initial regions and $u_k \in C(o_k(x_k))$. As discussed in Sec. IV-B, only input regions with radii of inscribed spheres larger than a predefined limit have been included in $\Sigma_c$. Then, taking $u_k$ to be the center of the inscribed sphere of the input region specified by $C(o_k(x_k))$ (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://usi.bu.edu/software.

A PWA system of dimensions $N = M = 2$, defined on polytopes $X_1, \ldots, X_{36}$ (shown in Fig. 2) was analyzed with conPAS.

The specification of interest was

$$\phi = \diamond X_1 \land \diamond X_{10} \land \diamond X_{27} \land \diamond X_{36}$$

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
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 ($X_0 = X$) when nondeterministic transitions were also included.

Starting from random initial conditions $x_1 \in X_{23}$, 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 $x_k \in X_{10}$ an input region $U_{i_k}$ 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 $U_{i_k}$ 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


