Controlled Invariance of Discrete Time Systems

by René E. Vidal

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Dr. Shankar S. Sastry
Research Advisor

Date

* * * * * *

Sr. Pravin Varaiya
Second Reader

Date

Controlled Invariance of Discrete Time Hybrid Systems

To Kathia

Acknowledgments

There are many people I would like to acknowledge for the support they have given my during my career. I am afraid I will miss someone, so let me start by giving thanks to everybody!

I am particularly indebted to my advisor Dr. Shankar S. Sastry. He has not only provided me constant guidance and inspiration in my research, but also been prompt to help in many aspects of my grad life. Many thanks!

I also want to thank Dr. John Lygeros, whose course in hybrid systems introduced me to this new and exciting field of research. His valuable comments and inspiration are reflected throughout this thesis.

There are no words to acknowledge the work of Shawn Schaffert, with whom I have discovered the beauty of hybrid systems. We have jointly worked in most of the theory presented here and he has done a superb job programming all the algorithms. Also many thanks to Omid Shakernia for his great job in helping me develop the theory of Section 4.3 and for providing the corresponding numerical examples.

I am very grateful to all the members of the Berkeley Aerobot team and to my office-mates Cory Sharp, Shawn Shu and Shahid Rashid for providing an exciting work atmosphere. Special thanks to Yi Ma for answering tons of questions in this two years.

Finally I would like to thank my wife and my parents for their endless love and support. This work is dedicated to them.

Berkeley, May 19, 2000

Contents

List of Figures

Abstract

Controlled Invariance of Discrete Time Hybrid Systems

by

René Vidal

Master of Science in Electrical Engineering and Computer Sciences

University of California at Berkeley

Professor Shankar Sastry, Chair

In this thesis we study the problem of controller synthesis under reachability specifications for discrete time hybrid systems. First, we show that, without loss of generality, the synthesis problem can be solved by a memoryless controller. Then, we establish conditions on the uniquenes of the solution of the synthesis problem. Afterwards, we propose an algorithm for computing the maximal controlled invariant set and the least restrictive controller. We show how the algorithm can be encoded using quantifier elimination, which leads to a semi-decidability result for definable systems. However, the computational complexity of the algorithm is doubly exponential.

For linear discrete time systems with all sets specified by linear inequalities, a more efficient (worst case exponential) implementation is proposed using linear programming and Fourier elimination. If in addition the system is in controllable canonical form, the input is scalar and unbounded, the disturbance is scalar and bounded and the initial set is a rectangle, then the problem is decidable. To speed up the computation, we illustrate how the algorithm can be approximated using robust semidefinite programming. This implementation is quite efficient (polynomial) and is applicable to systems with ellipsoidal constraints as well.

Finally, we generalize the controlled invariance algorithm to discrete time hybrid systems, with some states and inputs taking values on the reals and others taking values on a finite set. We show that this class of hybrid systems is a special case of a discrete time system, and hence the controlled invariance problem is semi-decidable.

Chapter 1

Introduction

The design of controllers is one of the most active research topics in the area of hybrid systems. Problems that have been addressed include hierarchical control [7, 29], distributed control [27], and optimal control using dynamic programming techniques [4, 6, 30, 35] or extensions of the maximum principle [17]. A substantial research effort has also been directed towards solving control problems with reachability specifications, that is designing controllers that guarantee that the state of the system will remain in a "good" part of the state space. Such control problems turn out to be very important in applications, and are closely related to the computation of the *reachable states* of a hybrid system and to the concept of *controlled invariance*. The proposed solutions extend game theory methods for purely discrete [31, 37] and purely continuous [3, 21, 23] systems to certain classes of hybrid systems: timed automata [19, 26], rectangular hybrid automata [40] and more general hybrid automata [25, 34, 38].

All of these techniques are concerned with hybrid systems whose continuous state evolves in continuous time, according to differential equations or differential inclusions. Unlike conventional continuous dynamical systems, little attention has been devoted to systems where the continuous state evolves in discrete time, according to difference equations. Besides being interesting in its own right, this class of hybrid systems can be used to approximate hybrid systems with differential equations. Indeed, most of the techniques that have been proposed for reachability computations for general continuous dynamics involve some form of discretization of the continuous space [13, 18, 38, 8], followed by a reachability computation on the resulting discrete time system.

On the other hand, different approximated solutions to related problems have been studied. In [21] elliposidal methods are used to compute reachable sets of continuous time affine systems. These methods have been generalized to a variety of similar problems in control theory [20]. In fact, many control problems can be solved by semidefinite programming [5], which has improved the time efficiency of many algorithms. Lately, semidefinite programming has been extended to deal with uncertainty [16]. This opens an opportunity

to apply robust semidefinite programming to the design of controllers for systems subject to disturbances.

Thesis Outline

In Section 2, we formulate the problem of controller synthesis for discrete time systems under reachability specifications, introduce the concepts of maximal controlled invariant set and least restrictive controller, and propose an algorithm for computing them. In Section 3 we review some concepts of mathematical logic and show how the algorithm can be implemented using quantifier elimination. This immediately leads to a semi-decidability result for discrete time systems whose continuous dynamics can be encoded in a decidable theory of the reals. In Section 4, we implement the proposed algorithm for discrete time linear systems with all the sets defined by linear inequalities. The implementation is based on a more efficient method for performing quantifier elimination in the theory of linear constraints using linear programming and Fourier elimination. We also show that the problem is decidable when the single-input single-disturbance linear discrete time system is in controllable canonical form, the input is unbounded, and the safe set is a rectangle. Because the computational burden of exactly performing quantifier elimination, even for this class of systems is still substantial, in Section 4.3 we also propose a method for approximating the solution using semidefinite programming. This implementation is quite efficient (polynomial) and is applicable to systems with ellipsoidal constraints as well. In Section 5 we generalize the controlled invariance algorithm to discrete time hybrid systems, with some states and inputs taking values on the reals and others taking values on a finite set. We show that this class of hybrid systems is a special case of a discrete time system, and hence the controlled invariance problem is semi-decidable. Finally, in Section 6, we illustrate the proposed method with some examples.

Chapter 2

Discrete Time Systems and Safety Specifications

2.1 Basic Definitions

Let Y be a countable collection of variables and let $\mathbf{Y}$ denote its set of valuations, that is the set of all possible assignments of these variables. We refer to variables whose set of valuations is countable as *discrete* and to variables whose set of valuations is a subset of a Euclidean space $\mathbb{R}^n$ as *continuous*. For a set $\mathbf{Y}$ we use $\mathbf{Y}^c$ to denote the complement of $\mathbf{Y}$, $2^{\mathbf{Y}}$ to denote the set of all subsets of $\mathbf{Y}$, $\mathbf{Y}^*$ to denote the set of all finite sequences of elements of $\mathbf{Y}$, and $\mathbf{Y}^\omega$ to denote the set of all infinite sequences. Since the dynamical systems we will consider will be time invariant we will use $y = \{y[i]\}_{i=0}^N$ to denote sequences. We use $\wedge$ to denote conjunction, $\vee$ to denote disjunction, $\neg$ to denote negation, $\forall$ to denote the universal quantifier, and $\exists$ to denote the existential quantifier.

The dynamics of a discrete time system are characterized by a reset relation that, given the current value of the state and input, returns the possible next states of the system. More formally:

Definition 1 (Discrete Time System (DTS)) *A discrete time system H is a collection $H = (X, V, \mathrm{Init}, f)$ consisting of a finite collection of state variables, X, a finite collection of input variables, V, a set of initial states,* $\mathrm{Init} \subseteq \mathbf{X}$*, and a reset relation, $f : \mathbf{X} \times \mathbf{V} \to 2^{\mathbf{X}}$.*

A DTS naturally characterizes a subset of the set of sequences from $\mathbf{X} \times \mathbf{V}$.

Definition 2 (Execution of DTS) *A sequence $\chi = (x, v) \in (\mathbf{X} \times \mathbf{V})^* \cup (\mathbf{X} \times \mathbf{V})^\omega$ is said to be an execution of the discrete time system H if $x[0] \in \mathrm{Init}$, and for all $k \geq 0$, $x[k+1] \in f(x[k], v[k])$.*

To ensure that every finite execution can be extended to an infinite execution we assume that $f(x,v) \neq \emptyset$ for all $(x,v) \in \mathbf{X} \times \mathbf{V}$. We call such a DTS *non-blocking.*[1]

We denote the set of all executions of H starting at $x_0 \in \mathbf{X}$ as $\mathcal{E}_H(x_0)$, and the set of all executions of H by $\mathcal{E}_H$. Clearly, $\mathcal{E}_H = \bigcup_{x_0 \in \text{Init}} \mathcal{E}_H(x_0)$.

Our goal here is to design controllers for DTS. Assume that we are given a *plant*, modeled as a DTS, and we would like to "steer" it using its input variables, so that its executions satisfy certain properties. We assume that the input variables are partitioned into two classes, $V = U \cup D$. U are assumed to be *control variables*, that is variables whose valuations we can specify at will. D, on the other hand, are assumed to be *disturbance variables*, over whose valuations we have no control (we say they are determined by the *environment*) and that can potentially disrupt our plans. In this context a controller can be defined as a feedback map.

Definition 3 (Controller) *A controller, C, is a map $C : \mathbf{X}^* \to 2^{\mathbf{U}}$. A controller is called* non-blocking *if $C(x) \neq \emptyset$ for all $x \in \mathbf{X}^*$. A controller is called* memoryless *if for all $x, x' \in \mathbf{X}^*$ ending at the same state we have $C(x) = C(x')$.*

The interpretation is that, given the evolution of the plant state up to now, the controller determines the set of allowable controls for the next transition. With this interpretation in mind, we define the set of *closed loop causal executions* as

$$\mathcal{E}_{H_C} = \{(x,u,d) \in \mathcal{E}_H \mid \forall k \geq 0, u[k] \in C(x{\downarrow_k})\},$$

where $x \downarrow_k$ denotes the subsequence of x consisting of its first k elements. Notice that a memoryless controller can be characterized by a map $g : \mathbf{X} \to 2^{\mathbf{U}}$, and its set of closed loop causal executions is simply

$$\mathcal{E}_{H_g} = \{(x,u,d) \in \mathcal{E}_H \mid \forall k \geq 0, u[k] \in g(x[k])\}.$$

Our goal is to use controllers to steer the executions of the plant, so that they satisfy certain desirable properties. In this paper we will restrict our attention to a class of properties known as *safety properties*: Given a set $F \subseteq \mathbf{X}$, we would like to find a non-blocking controller that ensures that the state stays in F for ever. We will say that a controller C *solves the problem* $(H, \Box F)$, if and only if C is non-blocking and for all $(x,u,d) \in \mathcal{E}_{H_C}$, $x[k] \in F$ for all $k \geq 0$. If such a controller exists we say that the problem $(H, \Box F)$ *can be solved.*

Even though safety properties are not the only properties of interest[2], they turn out to be very useful in applications. Many important problems, such as absence of collisions

[1]The condition is only sufficient. Although it can be refined to be necessary as well, we will not pursue this direction since the emphasis of this paper is controller synthesis.

[2]Other important properties are liveness properties (ensuring that the state eventually reaches a certain set, visits a set infinitely often, etc.), stability, optimality, etc.

in transportation systems, mutual exclusion in distributed algorithms, etc., can be naturally encoded as safety properties. Fortunately, it can be shown that for this class of properties one can, without loss of generality, restrict attention to memoryless controllers.

Proposition 1 *The problem $(H, \Box F)$ can be solved if and only if it can be solved by a memoryless controller.*

Proof: The *if* part is obvious. For the *only if* part, assume, for the sake of contradiction, that there exists a controller $C : \mathbf{X}^* \to 2^{\mathbf{U}}$ that solves the problem $(H, \Box F)$, but there does not exist a memoryless controller that solves the problem. Therefore, there must exist two finite executions $\chi_i = (x_i, u_i, d_i) \in \mathcal{E}_{H_C}$, $i = 1, 2$, ending at the same state, x, at times k_1 and k_2 respectively, such that $\chi_1 \neq \chi_2$, and $C(x_1) \neq C(x_2)$. Moreover, the information about the way in which the state x is reached must be essential for subsequent control decisions. Assume that x is reached via χ_1, but we choose to ignore this fact and apply controls after k_1 as though x had been reached via χ_2. Then, there must exist a continuation, χ', such that the concatenation $\chi'_1 = (x'_1, u'_1, d'_1) = \chi_1\chi' \in \mathcal{E}_H$ leaves the set F. In particular, since $\chi_1 \in \mathcal{E}_{H_C}$ and C solves $(H, \Box F)$, there must exist $k > 0$ such that $x'_1[k_1 + k] \notin F$. Notice, however, that the concatenation $\chi'_2 = (x'_2, u'_2, d'_2) = \chi_2\chi'$ is also an element of $\mathcal{E}_H$. Moreover, $\chi_2\chi' \in \mathcal{E}_{H_C}$. But $x'_2[k_2 + k] = x'_1[k_1 + k] \notin F$. This contradicts the assumption that C solves the problem $(H, \Box F)$. ■

Motivated by Proposition 1, we restrict our attention to memoryless controllers from now on.

2.2 Controlled Invariant Sets and Least Restrictive Controllers

The concept of controlled invariance turns out to be fundamental for the design of controllers for safety specifications [25]. Roughly speaking, a set of states, W, is called controlled invariant if there exists a controller that ensures that all executions starting somewhere in W remain in W for ever. More formally:

Definition 4 (Controlled invariant set) *A set $W \subseteq \mathbf{X}$ is called a controlled invariant set of H if there exists a non-blocking controller that solves the problem $(H', \Box W)$, where $H' = (X, V, W, f)$ (the same as H, but with $\mathrm{Init}' = W$).*

We say that the controller that solves the problem $(H', \Box W)$ *renders the set W invariant.* Also, given a set $F \subseteq \mathbf{X}$, a set $W \subseteq F$ is called a *maximal controlled invariant subset*

of F, if it is controlled invariant and it is not a proper subset of any other controlled invariant subset of F. The following lemma establishes the uniqueness of the maximal controlled invariant set.

Lemma 1 *The problem* $(H, \Box F)$ *can be solved if and only if there exists a unique maximal controlled invariant set,* $\hat{W}$*, with* $\text{Init} \subseteq \hat{W} \subseteq F$*.*

Proof: For the *if* part, assume that such a $\hat{W}$ exists and consider a non-blocking controller g that renders $\hat{W}$ invariant. Then $\mathcal{E}_{H_g} \subseteq \mathcal{E}_{H'_g}$, since $\text{Init} \subseteq \hat{W}$. Therefore, all $\chi \in \mathcal{E}_{H_g}$ remain for ever in $\hat{W}$, and hence in F.

For the *only if* part, assume the problem $(H, \Box F)$ can be solved by a non-blocking controller g. We claim that there exists a controlled invariant set W with $\text{Init} \subseteq W \subseteq F$. Consider the set

$$W = \bigcup_{x_0 \in \text{Init}} \bigcup_{k \geq 0} \left\{ x[k] \mid (x, u, d) \in \mathcal{E}_{H_g}(x_0) \right\}.$$

By definition, $\text{Init} \subseteq W$. Since g solves the problem $(H, \Box F)$, $W \subseteq F$. Moreover, for any $x[0] \in W$ consider an execution (x, u, d), with arbitrary $d \in \mathbf{D}^*$ and $u[k] \in g(x[k])$. Then by definition of W, $x[k] \in W$ for all $k \geq 0$. Therefore, the controller g renders the set W invariant, which proves the claim.

To show that there exists a unique maximal controlled invariant set, let $\mathcal{W}$ be the family of all controlled invariant sets W, with $\text{Init} \subseteq W \subseteq F$, $\mathcal{G}$ be the family of the corresponding non-blocking memoryless controllers that render each element of $\mathcal{W}$ invariant, and $h : \mathcal{W} \to \mathcal{G}$ be the map assigning to each $W \in \mathcal{W}$ its corresponding memoryless controller $g \in \mathcal{G}$. By the above discussion, $\mathcal{W}$ (and hence $\mathcal{G}$) is non-empty. By the well-ordering theorem [32] there exists a well-ordering relation for $\mathcal{G}$. We define the memoryless controller

$$g(x) = \min_{W \in \mathcal{W}} \{ h(W) \mid x \in W \},$$

where min is taken according to the order on $\mathcal{G}$. Let

$$\hat{W} = \bigcup_{W \in \mathcal{W}} W.$$

Clearly $\text{Init} \subseteq \hat{W} \subseteq F$. If we show that $\hat{W}$ is also controlled invariant, then the class of controlled invariant sets will be closed under arbitrary unions, and hence possess a unique maximal element. Let $\chi = (x, u, d)$ be an execution of H starting at $x[0] \in \hat{W}$, with arbitrary $d \in \mathbf{D}^*$ and $u[k] \in g(x[k])$. Assume, for the sake of contradiction, that there exists $k \geq 0$ such that $x[k'] \in \hat{W}$ for all $0 \leq k' \leq k$, and $x[k+1] \notin \hat{W}$. Since $x[k] \in \hat{W}$ and $u[k] \in g(x[k])$, there exists $W \in \mathcal{W}$ such that $x[k] \in W$ and $u[k] \in h(W)$. By assumption, $h(W)$ solves the problem $(H', \Box W)$ with $Init' = W$. Therefore $x[k+1] \in W \subseteq \hat{W}$, contradicting the assumption that $x[k+1] \notin \hat{W}$. ∎

A useful and intuitive characterization of the concept of controlled invariance can be given in terms of the operator $\mathrm{Pre} : 2^{\mathbf{X}} \to 2^{\mathbf{X}}$ defined by

$$\mathrm{Pre}(W) = \{x \in W \mid \exists u \in \mathbf{U}\ \forall d \in \mathbf{D},\ f(x,u,d) \cap W^c = \emptyset\}\,.$$

The operator returns the set of states $x \in W$ for which $u \in \mathbf{U}$ can be chosen such that, for all choices of $d \in \mathbf{D}$, all the states that can be reached from x after one transition are also in W. The following properties of the operator Pre are easy to establish and will be useful in the subsequent discussion.

Proposition 2 *The operator* Pre *has the following properties:*

1. Pre *is contracting, that is for all* $W \subseteq \mathbf{X}$, $\mathrm{Pre}(W) \subseteq W$*;*
2. Pre *is monotone, that is for all* $W, W' \subseteq \mathbf{X}$ *with* $W \subseteq W'$, $\mathrm{Pre}(W) \subseteq \mathrm{Pre}(W')$*; and,*
3. *A set* $W \subseteq \mathbf{X}$ *is controlled invariant if and only if it is a fixed point of* Pre*, that is if and only if* $\mathrm{Pre}(W) = W$*.*

Proof: The first part follows from the definition. For the second part, notice that for all $x \in \mathrm{Pre}(W)$ there exists $u \in \mathbf{U}$ such that for all $d \in \mathbf{D}$, $f(x,u,d) \subseteq W$. Since $W \subseteq W'$, this means that for the same u, and for all d, $f(x,u,d) \subseteq W'$, and therefore, $x \in \mathrm{Pre}(W')$.

We now turn our attention to the third part of the proposition. For the *if* part, assume $\mathrm{Pre}(W) = W$ and consider the memoryless controller

$$g(x) = \begin{cases} \{u \in \mathbf{U} \mid \forall d \in \mathbf{D},\ f(x,u,d) \cap W^c = \emptyset\} & x \in W \\ \mathbf{U} & x \notin W. \end{cases}$$

By construction g is non-blocking. Consider an execution $(x,u,d) \in \mathcal{E}_{H_g}$ with $x[0] \in W$, and assume that for all $0 \leq k' \leq k$, $x[k'] \in W$. Then $x[k+1] \in f(x[k],u[k],d[k]) \subseteq W$ by construction of g. Therefore, $x[k] \in W$ for all $k \geq 0$ by induction. For the *only if* part, notice that by definition $\mathrm{Pre}(W) \subseteq W$. Assume there exists a non-blocking, memoryless controller, g, that solves the problem $(H', \Box W)$ with $\mathrm{Init}' = W$. Consider an arbitrary $x \in W$ and notice that by assumption there exists $u \in g(x)$ such that for all $d \in \mathbf{D}$ and for all $x' \in f(x,u,d)$, $x' \in W$. Therefore, $x \in \mathrm{Pre}(W)$ and $W \subseteq \mathrm{Pre}(W)$. ∎

Many memoryless controllers may be able to solve a particular problem. Controllers that impose less restrictions on the inputs they allow are in a sense better than controllers that impose more restrictions. For example, controllers that impose fewer restrictions allow more freedom if additional safety specifications are imposed, or if one is asked to optimize the performance of the (safe) closed loop system with respect to other objectives. To quantify this intuitive notion we introduce a partial order on the space of memoryless controllers. We write $g_1 \preceq g_2$ if for all $x \in \mathbf{X}$, $g_1(x) \subseteq g_2(x)$.

Definition 5 (Least restrictive controller) *A memoryless controller $g : \mathbf{X} \to 2^{\mathbf{U}}$ that solves the problem (H, F) is called least restrictive if it is maximal among the controllers that solve $(H, \Box F)$ in the partial order defined by $\preceq$.*

Lemma 2 *A controller that renders a set W invariant exists if and only if a unique least restrictive controller that renders W invariant exists.*

Proof: The *if* part is obvious. For the *only if* part, assume that, given a set W, there exists a controller g that renders W invariant. Let $\mathcal{G}$ be the collection of all controllers that render W invariant and define $\hat{g} : \mathbf{X} \to 2^{\mathbf{U}}$ as

$$\hat{g}(x) = \bigcup_{g \in \mathcal{G}} g(x).$$

We claim that $\hat{g}$ renders W invariant. Let $\chi = (x, u, d) \in \mathcal{E}_{H_{\hat{g}}}(x_0)$ for some $x_0 \in W$ and assume, for the sake of contradiction, that there exists $k \geq 0$ such that $x[k'] \in W$ for all $0 \leq k' \leq k$, but $x[k+1] \notin W$. Now, by definition of $\hat{g}$, there exists $g \in \mathcal{G}$ such that $u(x[k]) \in g(x[k])$. Since g renders W invariant and $x[k] \in W$, $x[k+1] \in W$, contradicting the assumption that $x[k+1] \notin W$. Therefore, the class of controllers that renders W invariant is closed under arbitrary unions, and hence it possesses a unique maximal element. ∎

Notice that the least restrictive controller that renders a set W invariant must, by definition, allow $\hat{g}(x) = \mathbf{U}$ for all $x \notin W$. Summarizing Lemmas 1 and 2 we have the following:

Theorem 1 *The problem $(H, \Box F)$ can be solved if and only if there exists:*

1. *a unique maximal controlled invariant set $\hat{W}$ with* Init $\subseteq \hat{W} \subseteq F$, *and*
2. *a unique least restrictive controller, $\hat{g}$, that renders $\hat{W}$ invariant.*

Motivated by Theorem 1 we state the controlled invariance problem more formally.

Problem 1 (Controlled Invariance Problem (CIP)) *Given a DTS and a set $F \subseteq \mathbf{X}$ compute the maximal controlled invariant subset of F, $\hat{W}$, the least restrictive controller, $\hat{g}$, that renders $\hat{W}$ invariant, and test whether* Init $\subseteq \hat{W}$.

2.3 Computation of $\hat{W}$ and $\hat{g}(x)$

We first present a conceptual algorithm for solving the CIP for general DTS. Even though there is no straightforward way of implementing this algorithm in the general case, in subsequent sections we show how this can be done for special classes of DTS.

Algorithm 1 (Controlled Invariance Algorithm)

initialization: $W^0 = F$, $W^{-1} = \mathbf{X}$, $l = 0$
while $W^{l-1} \cap (W^l)^c \neq \emptyset$ **do**
$W^{l+1} = \mathrm{Pre}(W^l)$
$l = l + 1$
end while
set $\hat{W} = \bigcap_{l \geq 0} W^l$
set $\hat{g}(x) = \begin{cases} \left\{ u \in \mathbf{U} \mid \forall d \in \mathbf{D},\ f(x,u,d) \cap (\hat{W})^c = \emptyset \right\} & x \in \hat{W} \\ \mathbf{U} & x \notin \hat{W} \end{cases}$

Theorem 2 *$\hat{W}$ is the maximal controlled invariant subset of F and $\hat{g}$ is the least restrictive controller that renders $\hat{W}$ invariant.*

Proof: To show that $\hat{W}$ is controlled invariant we show that it is a fixed point of Pre. By definition $\mathrm{Pre}(\hat{W}) \subseteq \hat{W}$. Conversely, consider $x \in \hat{W}$ and assume, for the sake of contradiction that $x \notin \mathrm{Pre}(\hat{W})$. Then for all $u \in \mathbf{U}$ there exists $d \in \mathbf{D}$ and $x' \in f(x,u,d)$ such that $x' \notin \hat{W}$. Therefore, there is an l such that $x' \notin W^l$. Hence $x \notin \mathrm{Pre}(W^l) = W^{l+1} \supseteq \hat{W}$, which is a contradiction. Therefore $\hat{W} \subseteq \mathrm{Pre}(\hat{W})$.

To show that $\hat{W}$ is maximal, consider a controlled invariant set $W \subseteq F$. Assume, for the sake of contradiction, that there exists $x[0] \in W \setminus \hat{W}$. Therefore, there exists $l \geq 0$ such that $x[0] \notin W^l$. By definition of the operator Pre this implies that either $x[0] \notin W^{l-1}$, or for all $u \in \mathbf{U}$ there exists $d \in \mathbf{D}$ and $x' \in f(x[0],u,d)$ such that $x' \notin W^{l-1}$. In the latter case set $x[1] = x'$. By induction, for all choices of u there exists a finite sequence that leaves $W^0 = F \supseteq W$ after at most l steps. This contradicts the assumption that W is controlled invariant.

Finally, to show $\hat{g}$ is least restrictive, consider another controller, g, that renders $\hat{W}$ invariant, and assume, for the sake of contradiction, that there exists $x \in \mathbf{X}$ and $u \in g(x) \setminus \hat{g}(x)$. By construction of $\hat{g}$, $x \in \hat{W}$. Since $u \notin \hat{g}(x)$, there exists $d \in \mathbf{D}$ and $x' \in f(x,u,d)$ such that $x' \notin \hat{W}$. This contradicts the assumption that g renders $\hat{W}$ invariant. ∎

To implement the controlled invariance algorithm one needs to be able to:

1. encode sets of states, perform intersection and complementation, and test for emptiness,
2. compute the Pre of a set, and
3. guarantee that a fixed point is reached after a finite number of iterations.

For classes of DTS for which 1 and 2 are satisfied we say that the CIP is *semi-decidable*; if all three conditions are satisfied we say that the CIP is *decidable*. As an example, consider *finite state machines* (FSM), that is the class of DTS for which $\mathbf{X}$, $\mathbf{U}$ and $\mathbf{D}$ are finite. In this case, one can encode sets of states, perform intersection, complementation, test for emptiness and compute Pre by enumeration (or other more efficient representations). Moreover, by the monotonicity of W^l and the fact that $\mathbf{X}$ is finite, the algorithm is guaranteed to terminate in a finite number of steps. Therefore, the CIP is decidable for finite state machines.

In subsequent sections we show how the computation can be performed for DTS with state and input taking values on a Euclidean space and transition relations given by certain classes of functions of the state and input. We then generalize this approach to discrete time hybrid systems, with some state and input variables taking values on a Euclidean space, and others taking values on finite sets. Before we can present the details, however, we need to introduce some notation from mathematical logic.

Chapter 3

Mathematical Logic and Quantifier Elimination

3.1 Languages, Models and Theories

The following discussion is based on [28]. For a more in depth treatment the reader is referred to [11, 12].

A *language* $\mathcal{L} = \{R_1, \ldots, R_n, f_1, \ldots, f_m, c_0, \ldots, c_l\}$ is a set of symbols separated into *relations*, $R_1, \ldots, R_n$, *functions*, $f_1, \ldots, f_m$, and *constants*, $c_0, \ldots, c_l$. For example, $\mathcal{P} = \{<, +, -, 0, 1\}$ and $\mathcal{R} = \{<, +, -, \cdot, 0, 1\}$ are languages with (binary) relation $<$, (binary) functions $+$, $-$ and $\cdot$, and constants 0 and 1.

Given a language $\mathcal{L}$ and a set of variables $\{x_1, x_2, \ldots, v_1, v_2, \ldots\}$, the *terms* of the language are inductively defined. All the variables and all the constants are terms, and if $t_1, \ldots, t_n$ are terms and f is an n-ary function, $f(t_1, \ldots, t_n)$ is also a term. For instance, if a, b and c are positive integer constants and x_1 and x_2 are variables, $ax_1 - bx_2 + c$ is a term of $\mathcal{P}$ and $ax_1^2 + bx_1x_2 + c$ is a term of $\mathcal{R}$[1].

An *atomic formula* of the language is of the form $t_1 = t_2$ or $R(t_1, \ldots, t_n)$, where R is a n-ary relation and t_i, $i = 1, \ldots, n$ are terms. For example, $ax_1 - bx_2 + c < 0$ is an atomic formula of $\mathcal{P}$ and $ax_1^2 + bx_1x_2 + c = 0$ is an atomic formula of $\mathcal{R}$. *First order formulas* (or simply formulas) are recursively defined from atomic formulas:

1. atomic formulas are formulas,

2. if ϕ, ψ are formulas, then so are $\phi \wedge \psi$ and $\neg\phi$[2],

[1] Integer constants are generated inductively by repeatedly adding the constant 1 to itself.

[2] Disjunction, $\phi \vee \psi$, is interpreted as $\neg(\neg\phi \wedge \neg\psi)$.

3. if ϕ is a formula and x is a variable, then $\exists x \mid \phi$ is also a formula[3].

Formulas defined in a language $\mathcal{L}$ are called $\mathcal{L}$-formulas. For example, $Mx \leq \beta$ is a $\mathcal{P}$-formula, where $M \in \mathbb{Q}^{m \times n}$ and $\beta \in \mathbb{Q}^m$ are constants, and $x = (x_1, \ldots, x_n)$ are variables. This becomes clear if we let $m_{ij} \in \mathbb{Q}$ be the i,j element of M and $\beta_i \in \mathbb{Q}$ be the i element of β and write $Mx \leq \beta$ as

$$\bigwedge_{i=1}^{m} [q_i (m_{i1}x_1 + \ldots + m_{in}x_n - \beta_i) < 0] \vee [q_i (m_{i1}x_1 + \ldots + m_{in}x_n - \beta_i) = 0]$$

where q_i is a positive common denominator of m_{ij}, $j = 1, \ldots, n$ and β_i. With a similar interpretation the following expression is also a $\mathcal{P}$-formula

$$\exists u \, \forall d \mid (Mx \leq \beta) \wedge (MAx + MBu + MCd \leq \beta) \tag{3.1}$$

where A, B, C, M and β are constant matrices with rational coefficients and x, u and d are variables.

The occurrence of a variable in a formula is *free* if it is not within the scope of a quantifier; otherwise it is *bound*. For example, x is free, and u and d are bound in (3.1). We often write $\phi(x_1, \ldots, x_n)$ to indicate that $x_1, \ldots, x_n$ are the free variables of formula ϕ. A *sentence* is a formula with no free variables.

A *model* of a language $\mathcal{L}$ consists of a non-empty set S and a semantic interpretation of the relations, functions and constants of $\mathcal{L}$. For instance, $(\mathbb{R}, <, +, -, 0, 1)$ and $(\mathbb{R}, <, +, -, \cdot, 0, 1)$ with the usual interpretation for the symbols are models of $\mathcal{P}$ and $\mathcal{R}$ respectively. Every sentence of the language is either true or false for a given model. Every formula, $\phi(x_1, \ldots, x_n)$, of the language defines a subset of S^n, namely the set of valuations of $x_1, \ldots, x_n$ for which the formula is true. Conversely, we say that a set $Y \subseteq S^n$ is *definable* in $\mathcal{L}$ if there exists a formula $\phi(x_1, \ldots, x_n)$ in $\mathcal{L}$ such that

$$Y = \{(a_1, \ldots, a_n) \in S^n \mid \phi(a_1, \ldots, a_n)\}.$$

Two formulas $\phi(x_1, \ldots, x_n)$ and $\psi(x_1, \ldots, x_n)$ are equivalent in a model, denoted by $\phi \equiv \psi$, if for every valuation $(a_1, \ldots, a_n)$ of $(x_1, \ldots, x_n)$, $\phi(a_1, \ldots, a_n)$ is true if and only if $\psi(a_1, \ldots, a_n)$ is true. Equivalent formulas define the same set.

Every model defines a *theory*, as the set of all sentences which hold in the model. For example, we denote by $\mathrm{Lin}(\mathbb{R})$ the theory defined by the formulas of $\mathcal{P}$ which are true over $(\mathbb{R}, <, +, -, 0, 1)$; in other words, $\mathrm{Lin}(\mathbb{R})$ is the theory of linear constraints. We denote by $\mathrm{OF}(\mathbb{R})$ the theory defined by the formulas of $\mathcal{R}$ which are true over $(\mathbb{R}, <, +, -, \cdot, 0, 1)$; in other words, $\mathrm{OF}(\mathbb{R})$ is the theory of the real numbers as an ordered field.

[3]Universal quantification, $\forall x \mid \phi$, is interpreted as $\neg(\exists x \mid \neg\phi)$.

3.2 Quantifier Elimination and Semi-decidability

The terminology introduced in the previous section provides a framework for defining sets of states, by using formulas in an appropriate theory. It also provides a method for performing intersection and complementation of sets, by taking conjunction and negation of the corresponding formulas. One would also like to be able to determine whether a set definable in the model is empty or not.

For some theories, it is possible to determine the sentences that belong to the theory. The Tarski-Seidenberg decision procedure provides a way of doing this for OF($\mathbb{R}$). It can be shown that OF($\mathbb{R}$) is decidable [33, 36], in other words, there exists a computational procedure that after a finite number of steps determines whether an $\mathcal{R}$-sentence belongs to OF($\mathbb{R}$) or not. The decision procedure is based on quantifier elimination, an algorithm that converts a formula $\phi(x_1, \ldots, x_n)$ to an equivalent quantifier free formula. Notice that this provides a method for testing emptiness. A set $Y = \{(x_1, \ldots, x_n) \mid \phi(x_1, \ldots, x_n)\}$ is empty if and only if the sentence $\exists x_1 \ldots \exists x_n \mid \phi(x_1, \ldots, x_n)$ is equivalent to false.

To relate this to the problem at hand, we restrict our attention to CIP which are "definable" in an appropriate theory.

Definition 6 (Definable CIP) *A CIP, $(H, \Box F)$, is definable in a theory if $\mathbf{X} = \mathbb{R}^n$, $\mathbf{U} \subseteq \mathbb{R}^{n_u}$, $\mathbf{D} \subseteq \mathbb{R}^{n_d}$ and the sets $\mathbf{U}$, $\mathbf{D}$,* Init, *$f(x, u, d)$ for all $x \in \mathbf{X}$, $u \in \mathbf{U}$ and $d \in \mathbf{D}$, and F are definable in the same theory.*

If $(H, \Box F)$ and W^l are definable in OF($\mathbb{R}$), then

$$\psi^l(x) \equiv \exists u \, \forall d \, \forall x' \mid [x \in W^l] \wedge [u \in \mathbf{U}] \wedge [(d \notin \mathbf{D}) \vee (x' \notin f(x, u, d)) \vee (x' \in W^l)] \qquad (3.2)$$

is a first order formula in the corresponding language. Therefore, each step of the controlled invariance algorithm involves eliminating the quantifiers in (3.2) to obtain a quantifier free formula defining W^{l+1}. The fact that OF($\mathbb{R}$) is decidable immediately leads to the following:

Theorem 3 *The class of CIP definable in* OF($\mathbb{R}$) *is semi-decidable.*

Moreover, if $(H, \Box F)$ is definable in OF($\mathbb{R}$) and W is a controlled invariant set also definable in OF($\mathbb{R}$), then the set

$$\{(x, u) \mid \forall d \in \mathbf{D} \, \forall x' \in f(x, u, d), \; x' \in W\}$$

describing the least restrictive controller that renders W invariant is also definable in OF($\mathbb{R}$). Furthermore, quantifier elimination can be performed in this formula, to obtain an explicit expression for the least restrictive controller. Finally, the question $W \cap \text{Init}^c = \emptyset$ can be

decided. Therefore, if the algorithm happens to terminate in a finite number of steps, the CIP can be completely solved.

The class of CIP definable in OF($\mathbb{R}$) is fairly broad. The class contains DTS with polynomial constraints on u, d and Init and reset relations encoded by constraints on the possible next states which are polynomials in x, u and d. Strictly speaking the problem remains semi-decidable even if we add polynomial state dependent input constraints, i.e. at each state, x, allow values of u and d that satisfy polynomial constraints in x, u and d. This includes for example the closed loop system obtained by coupling the least restrictive controller with the plant.

Although different methods have been proposed for performing quantifier elimination in OF($\mathbb{R}$) [2, 33, 36], and the process can be automated using symbolic tools [14], the quantifier elimination procedure is in general hard, both in theory and in practice, since the solvability may be doubly exponential [22]. For the theory Lin($\mathbb{R}$), a somewhat more efficient implementation can be derived using techniques from linear algebra and linear programming. The next section shows how quantifier elimination in the theory Lin($\mathbb{R}$) can be performed more efficiently for the formula (3.2) used in the controlled invariance algorithm.

Chapter 4

CIP for Linear Discrete Time Systems

4.1 Exact Computation of $\hat{W}$ and $\hat{g}(x)$

A *linear CIP* (LCIP) consists of

- a Linear DTS (LDTS), i.e. a DTS with $\mathbf{X} = \mathbb{R}^n$, $\mathbf{U} = \{u \in \mathbb{R}^{n_u} \mid Eu \leq \eta\} \subseteq \mathbb{R}^{n_u}$, $\mathbf{D} = \{d \in \mathbb{R}^{n_d} \mid Gd \leq \gamma\} \subseteq \mathbb{R}^{n_d}$, Init $= \{x \in \mathbf{X} \mid Jx \leq \theta\}$ and a reset relation given by $f(x,u,d) = \{Ax + Bu + Cd\}$, where $A \in \mathbb{Q}^{n \times n}$, $B \in \mathbb{Q}^{n \times n_u}$, $C \in \mathbb{Q}^{n \times n_d}$, $E \in \mathbb{Q}^{m_u \times n_u}$, $G \in \mathbb{Q}^{m_d \times n_d}$, $\eta \in \mathbb{Q}^{m_u}$, $\gamma \in \mathbb{Q}^{m_d}$, $J \in \mathbb{Q}^{n \times m_i}$ and $\theta \in \mathbb{Q}^{m_i}$ with m_u, m_d and m_i being the number of constraints on the control, disturbance and initial conditions, respectively; and,
- a set $F = \{x \in \mathbb{R}^n \mid Mx \leq \beta\}$ where $M \in \mathbb{Q}^{m \times n}$, $\beta \in \mathbb{Q}^m$ and m is the number of constraints on the state.

Notice that LDTS are non-blocking and deterministic, in the sense that for every state x and every input (u,d) there exists a unique next state. Since the sets F, $\mathbf{U}$ and $\mathbf{D}$ are all convex polygons, and the dynamics f are given by a linear map, a LCIP is definable in the theory Lin($\mathbb{R}$), and therefore, according to the discussion in Section 3, it is semi-decidable. We assume that the sets F and $\mathbf{U}$ can be either bounded or unbounded, but $\mathbf{D}$ is bounded[1].

For the LCIP it turns out that, after the l-th iteration, the set W^l can be described by m^l linear constraints as $\{x \in \mathbb{R}^n \mid M^l x \leq \beta^l\}$, that is, W^l remains a convex polygon. Obviously, $m^0 = m$, $M^0 = M$ and $\beta^0 = \beta$. Letting $\hat{A}^l = M^l A$, $\hat{B}^l = M^l B$ and $\hat{C}^l = M^l C$, (3.2) becomes

$$\begin{aligned} \psi^l(x) &\equiv \exists u \, \forall d \mid [M^l x \leq \beta^l] \wedge [Eu \leq \eta] \wedge [(Gd > \gamma) \vee (\hat{A}^l x + \hat{B}^l u + \hat{C}^l d \leq \beta^l)] \\ &\equiv [M^l x \leq \beta^l] \wedge [\exists u \mid (Eu \leq \eta) \wedge (\forall d \mid (Gd > \gamma) \vee (\hat{A}^l x + \hat{B}^l u + \hat{C}^l d \leq \beta^l))]. \end{aligned}$$

[1]The theoretical discussion can be extended to unbounded $\mathbf{D}$ sets, but the computational implementation is somewhat more involved.

Thus, in each step of the algorithm, we need to be able to eliminate variables u and d from the inner formulae, intersect the new constraints with the old ones and check if the new set is empty. Notice that not all of the new constraints generated by quantifier elimination may be necessary to define the set W^{l+1}. Also, some of the old constraints may become redundant after adding the new ones. Hence, we also need to check the redundancy of the constraints when doing the intersection.

4.1.1 Quantifier Elimination

We first perform quantifier elimination on d over the formula

$$\phi^l(x,u) \equiv \forall d \mid (Gd > \gamma) \vee (\hat{A}^l x + \hat{B}^l u + \hat{C}^l d \leq \beta^l) .$$

Let $\hat{a}_i^T$, $\hat{b}_i^T$ and $\hat{c}_i^T$ be the i-th row of $\hat{A}^l$, $\hat{B}^l$ and $\hat{C}^l$, respectively. Then, parsing ϕ^l leads to

$$\phi^l(x,u) \equiv \forall d \mid \bigwedge_{i=1}^{m^l} (Gd > \gamma) \vee (\hat{c}_i^T d \leq \beta_i^l - \hat{a}_i^T x - \hat{b}_i^T u).$$

Consider $\delta : \mathbb{R}^{m^l \times n_d} \rightarrow \mathbb{R}^{m^l}$ defined by $\delta_i(\hat{C}^l) = \max_{d:Gd\leq\gamma} (\hat{c}_i^T d)$ for $i = 1, \ldots, m^l$.

Proposition 3 *$\phi^l(x,u)$ is equivalent to $\varphi^l(x,u) \equiv \hat{A}^l x + \hat{B}^l u \leq \beta^l - \delta(\hat{C}^l)$.*

Proof: If $\phi^l(x,u)$ is false then $\exists d^* \mid (Gd^* \leq \gamma) \wedge (\hat{c}_i^T d^* > \beta_i^l - \hat{a}_i^T x - \hat{b}_i^T u)$ for some i. Since $\delta_i(\hat{C}^l) \geq \hat{c}_i^T d^*$, we conclude that $\neg\phi^l \Rightarrow \neg\varphi^l$, hence $\varphi^l \Rightarrow \phi^l$. Similarly, if $\varphi^l(x,u)$ is false, $\exists d^* \mid (Gd^* \leq \gamma) \wedge (\hat{c}_i^T d^* > \beta_i^l - \hat{a}_i^T x - \hat{b}_i^T u)$ for some i. Thus ϕ^l is false and $\phi^l \Rightarrow \varphi^l$. ■

Therefore, the elimination of the $\forall$ quantifier can be done by solving a finite collection of linear programming problems. Since we have assumed that $\mathbf{D}$ is bounded, such an optimization problem is guaranteed to have a solution, and hence $\delta(\cdot)$ is well defined. Since $\delta(\cdot)$ is applied to each row of $\hat{C}^l$, in the sequel we will use $\delta_i(\hat{C}^l)$ and $\delta(\hat{c}_i^T)$ interchangeably. Notice that, strictly speaking, $\delta(\cdot)$ is not part Lin($\mathbb{R}$), but we use it as a shorthand for the constant obtained by solving the linear programs.

Next, we perform quantifier elimination on u over the formula

$$\phi^l(x) \equiv \exists u \mid (Eu \leq \eta) \wedge (\hat{A}^l x + \hat{B}^l u \leq \beta^l - \delta(\hat{C}^l)) \equiv \exists u \mid \begin{bmatrix} \hat{A}^l & \hat{B}^l \\ 0 & E \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} \beta^l - \delta(\hat{C}^l) \\ \eta \end{bmatrix} \quad (4.1)$$

We will discuss two methods to eliminate u. The first is known as *Fourier Elimination* [15], and the second, attributed to Cernikov [9], is an application of Farkas Lemma on duality [10].

For the first method, assume we want to eliminate u_1 first. Let e_i be the i-th unit vector in $\mathbb{R}^{m^l+m_u}$,

$$H^l = \begin{bmatrix} \hat{B}^l \\ E \end{bmatrix} \quad \text{and} \quad \xi^l(x) = \begin{bmatrix} \beta^l - \delta(\hat{C}^l) - \hat{A}^l x \\ \eta \end{bmatrix}.$$

Thus $\phi^l(x)$ is equivalent to $\exists u \mid H^l u \leq \xi^l(x)$. Also define $P^l = \{p \mid H^l_{p1} > 0\}$, $Q^l = \{q \mid H^l_{q1} < 0\}$ and $R^l = \{r \mid H^l_{r1} = 0\}$, where H^l_{ij} refers to the i,j element of the matrix H^l. Then $\phi^l(x)$ is equivalent to

$$\exists u \mid \bigwedge_{p\in P^l} \bigwedge_{q\in Q^l} \left[\frac{1}{H^l_{q1}}(\xi^l_q(x) - \sum_{j=2}^{m} H^l_{qj}u_j) \leq u_1 \leq \frac{1}{H^l_{p1}}(\xi^l_p(x) - \sum_{j=2}^{m} H^l_{pj}u_j) \right]$$
$$\wedge \bigwedge_{r\in R^l} \left[0 \leq (\xi^l_r(x) - \sum_{j=2}^{m} H^l_{rj}u_j) \right].$$

Hence, after the elimination of u_1 we obtain

$$\exists u \mid \bigwedge_{p\in P^l} \bigwedge_{q\in Q^l\cup R^l} [H^l_{p1} \ -H^l_{q1}] \begin{bmatrix} \hat{e}_q^T \\ \hat{e}_p^T \end{bmatrix} \begin{bmatrix} \hat{A}^l x \\ 0 \end{bmatrix} \leq [H^l_{p1} \ -H^l_{q1}] \begin{bmatrix} \hat{e}_q^T \\ \hat{e}_p^T \end{bmatrix} \begin{bmatrix} \beta^l - \delta(\hat{C}^l) \\ \eta \end{bmatrix}$$
$$- [H^l_{p1} \ -H^l_{q1}] \begin{bmatrix} \sum_{j=2}^m H^l_{qj}u_j \\ \sum_{j=2}^m H^l_{pj}u_j \end{bmatrix}. \qquad (4.2)$$

Therefore, the elimination of the $\exists$ quantifier is performed by taking nonnegative linear combinations of all pairs of constraints so as to cancel the quantified variable. Note that if all the coefficients of the quantified variable are positive (negative), then ϕ^l is true, and we need not to eliminate the remaining variables. Otherwise, after u_1 has been eliminated, we apply the same procedure to the constraints in (4.2), so as to eliminate $u_2, \ldots, u_{n_u}$. Since the procedure is based on nonnegative row operations, it is clear that

$$\phi^l(x) \equiv \Lambda^l \begin{bmatrix} \hat{A}^l x \\ 0 \end{bmatrix} \leq \Lambda^l \begin{bmatrix} \beta^l - \delta(\hat{C}^l) \\ \eta \end{bmatrix} \equiv (\tilde{M}^l x \leq \tilde{\beta}^l) \wedge (0 \leq \Lambda^l_2 \eta), \qquad (4.3)$$

where $\Lambda^l = [\Lambda^l_1 \ \Lambda^l_2] \in \mathbb{Q}^{\tilde{m}^l \times (m^l+m_u)}$ is a matrix with nonnegative entries such that $\Lambda^l H^l = 0$, $\tilde{m}^l$ is the number of new constraints obtained through quantifier elimination, $\tilde{M}^l = \Lambda^l_1 \hat{A}^l \in \mathbb{Q}^{\tilde{m}^l \times n}$ and $\tilde{\beta}^l = \Lambda^l_1(\beta^l - \delta(\hat{C}^l)) \in \mathbb{Q}^{\tilde{m}^l}$. Notice that if the condition $\Lambda^l_2 \eta \geq 0$ is violated, then $\hat{W} = \emptyset$. Otherwise, we just need to add the new constraints $\tilde{M}^l x \leq \tilde{\beta}^l$ to the original set W^l.

Although *Fourier Elimination* is attractive because of its simplicity, it is quite inefficient. In general, it generates many new constraints in the intermediate steps, and in the worst case the method is exponential. This difficulty can be partially remedied since many of the inequalities are likely to be redundant [10]. An alternative method [9] computes the rows of Λ^l directly as the extreme points of the set $\{\lambda^l \in \mathbb{R}^{m+m_u} \mid {\lambda^l}^T H^l = 0 \wedge \lambda^l \geq 0 \wedge \sum \lambda^l_i = 1\}$,

where the last constraint is added to ensure that the set is a polytope. Although the extreme points method is better than Fourier elimination, because it eliminates the costly intermediate steps, the computation of the extreme points is still costly and also generates a lot of redundant constraints. A more efficient method [22] uses a generalized linear programming formulation and an on-line convex hull construction to obtain an incremental inner approximation of the set defined by ϕ^l. The method considerably reduces the number of constraints defining the resulting set.

4.1.2 Intersection, Emptiness and Redundancy

Provided that $\Lambda_2^l \eta \geq 0$, the quantifier elimination procedure presented above computes the set of states $\tilde{W}^l \equiv \{x \,|\, \tilde{M}^l x \leq \tilde{\beta}^l\}$ that can be forced by u to transition into W^l. To obtain W^{l+1}, such a set must be intersected with W^l. Since both sets are convex, the intersection can be carried out by simply appending $\tilde{M}^l$ and $\tilde{\beta}^l$ to M^l and β^l, respectively. However, this method of performing the intersection is likely to lead to a description of the set which is larger than necessary since many of the constraints may be redundant. Algorithm 2 is aimed at checking the emptiness of the intersection and then eliminate redundant constraints. In the algorithm, [] denotes an empty matrix, $\mathbf{1} = (1 \ldots 1)^T \in \mathbb{Q}^{\tilde{m}^l + m^l}$, and ${m'_i}^T$ and β'_i are the i-th rows of $M'_0 = \begin{bmatrix} \tilde{M}^l \\ M^l \end{bmatrix}$ and $\beta'_0 = \begin{bmatrix} \tilde{\beta}^l \\ \beta^l \end{bmatrix}$, respectively. Initially, $M' = M'_0$ and $\beta' = \beta'_0$.

The idea behind the algorithm is that $W^l \cap \tilde{W}^l \neq \emptyset$ if and only if $\exists x \mid M'x \leq \beta'$, which is equivalent to saying that $\min\{t \mid M'x \leq \beta' + \mathbf{1}t\} \leq 0$. Afterwards, if the problem $\max\{{m'_i}^T x \mid M'x \leq \beta'\}$ is feasible, and the constraint ${m'_i}^T x \leq \beta'_i$ is not redundant, then the optimal value of the problem is β'_i. Moreover, if the non-redundant constraint ${m'_i}^T x \leq \beta'_i$ is removed from the optimization problem, then the new optimal value m^* satisfies $m^* > \beta'_i$.

Algorithm 2 (Emptiness and Redundancy Algorithm)

initialization $M' = M'_0$, $\beta' = \beta'_0$, $M^{l+1} = []$, $\beta^{l+1} = []$.
$m^* = \min\{t \mid M'x \leq \beta' + \mathbf{1}t\}$
if $m^* > 0$ **or** $\Lambda_2^l \eta \not\geq 0$ **then**
    $\hat{W} = \emptyset$, **terminate** controlled invariance algorithm
else
    for $i = 1$ **to** $\tilde{m}^l + m^l$ **do**
        remove ${m'_i}^T$ from M' and β'_i from β'
        $m^* = \max\{{m'_i}^T x \mid M'x \leq \beta'\}$
        if $m^* > \beta'_i$ **then**
            add ${m'_i}^T$ to M^{l+1} and M',
            add β'_i to β^{l+1} and β'
        end if
    end for
end if

if $M^{l+1} = M^l$ and $\beta^{l+1} = \beta^l$ **then**
$\hat{W} = W^l$, **terminate** controlled invariance algorithm
end if

The controlled invariance algorithm terminates if the redundancy algorithm concludes that either $\Lambda_2^l \eta \not\geq 0$ or $W^l \cap \tilde{W}^l = \emptyset$ (in which case $\hat{W} = \emptyset$), or if all the new constraints are redundant (in which case $W^l = W^{l+1} = \hat{W}$)[2]. Otherwise, upon termination of the redundancy algorithm, the process is repeated for W^{l+1}. An obvious optimization of the code involves terminating both algorithms if after all new constraints in $\tilde{M}^l x \leq \tilde{\beta}^l$ have been tested, M^{l+1} and β^{l+1} are still empty. Notice that for all l the set W^l is a convex polygon as claimed.

In the next section, we study situations where the algorithm is guaranteed to terminate in a finite number of steps. In Section 6, we will provide an example which actually converges after an infinite number of iterations. This will prove that:

Theorem 4 *The LCIP is semi-decidable.*

4.2 Decidable Special Cases

We first summarize some of the observations made so far about situations where the algorithm terminates in a finite number of steps.

Proposition 4 *For an LCIP with* $\mathbf{U} = \mathbb{R}^{n_u}$*, if either one of the columns of* MB *is componentwise positive (negative), or if* $rank(MB) = \min\{m, n\}$*, the algorithm terminates in a finite number of steps.*

Proof: If one of the columns of MB is positive (negative), there is no $\lambda \geq 0$ other than $\lambda = 0$ such that $\lambda^T MB = 0$. Thus $\hat{W} = F$, $g(x) = \mathbf{U}$ for all $x \in \mathbf{X}$, and the algorithm terminates in the first iteration. If $rank(MB) = m$, the only solution of $\lambda^T MB = 0$ is $\lambda = 0$, thus we obtain the same result as in the previous case. Finally, if $rank(MB) = n$, then $\lambda^T MB = 0 \Rightarrow \lambda^T MA = 0$. Thus, if $\lambda^T(\beta - \delta(MC)) \geq 0$, we have $\hat{W} = F$ and $\hat{g}(x) = \mathbf{U}$ for all $x \in \mathbf{X}$; otherwise, we have $\hat{W} = \emptyset$ and $\hat{g}(x) = \mathbf{U}$ for all $x \in \mathbf{X}$. Again, the algorithm terminates in the first iteration. ∎

Next, we limit our attention to the case $F = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset \mathbb{R}^n$ with $\alpha_i \leq \beta_i$ and $[\alpha_i, \beta_i] \subset \mathbb{R}, i = 1 \ldots n$, $u \in \mathbb{R}$, and $d \in [d_1, d_2] \subset \mathbb{R}$. To remind ourselves of the fact

[2]Note that any redundant constraint in the original description of F will be eliminated the first time the redundancy algorithm is invoked by the controlled invariance algorithm.

that u and d are scalar, we use b and c instead of B and C. We also assume that (A, b) is in controllable canonical form, that is

$$x[k+1] = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & & & & & 1 \\ a_{n1} & a_{n2} & & \cdots & & a_{nn} \end{bmatrix} x[k] + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u[k] + \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} d[k]. \qquad (4.4)$$

In this case $\psi^1(x)$ is equivalent to

$$\exists u \mid \quad \bigwedge_{j=1}^{n} (\alpha_j \le x_j \le \beta_j) \wedge \bigwedge_{j=2}^{n} (\alpha_{j-1} - \delta(-c_{j-1}) \le x_j \le \beta_{j-1} - \delta(c_{j-1})) \wedge$$
$$\left(\alpha_n - \sum_{j=1}^{n} a_{nj} x_j - \delta(-c_n) \le u \le \beta_n - \sum_{j=1}^{n} a_{nj} x_j - \delta(c_n) \right). \qquad (4.5)$$

From the last expression, it is clear that given $x_1 \in [\alpha_1, \beta_1]$, $x_j, j = 2 \ldots n$ exists if and only if

$$\alpha_j^1 = \max(\alpha_j, \alpha_{j-1} - \delta(-c_{j-1})) \quad \le \quad \min(\beta_j, \beta_{j-1} - \delta(c_{j-1})) = \beta_j^1, \qquad j = 2 \ldots n$$

and u exists if and only if

$$\alpha_n - \delta(-c_n) \le \beta_n - \delta(c_n).$$

It is straightforward to see that in the l-th iteration ($0 \le l \le n$) W^l is defined by:

$$W^l \quad = \quad [\alpha_1^0, \beta_1^0] \times [\alpha_2^1, \beta_2^1] \times \ldots \times [\alpha_{l+1}^l, \beta_{l+1}^l] \times [\alpha_{l+2}^l, \beta_{l+2}^l] \times [\alpha_{l+3}^l, \beta_{l+3}^l] \times \ldots \times [\alpha_n^l, \beta_n^l],$$

where

$$\alpha_j^0 = \alpha_j, \qquad \alpha_j^l = \max(\alpha_j^{l-1}, \alpha_{j-1}^{l-1} - \delta(c_{j-1})) \qquad l+1 \le j \le n$$
$$\beta_j^0 = \beta_j, \qquad \beta_j^l = \min\,(\beta_j^{l-1}, \beta_{j-1}^{l-1} - \delta(c_{j-1})) \qquad l+1 \le j \le n$$

This means that after n iterations, the maximal controlled invariant set remains unchanged, and the least restrictive controller is given by the last constraint in (4.5), but with α_n and β_n replaced by α_n^{n-1} and β_n^{n-1}, respectively. This result can be summarized as follows:

Lemma 3 *Given system (4.4) with $F = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset \mathbb{R}^n$, $\mathbf{U} = \mathbb{R}$ and $\mathbf{D} = [d_1, d_2] \subset \mathbb{R}$, the solution to the CIP, obtained after at most n iterations of the algorithm, is given by:*

$$\hat{W} \quad = \quad \begin{cases} \left\{ x \mid \bigwedge_{j=1}^{n} \alpha_j^{j-1} \le x_j \le \beta_j^{j-1} \right\} & \text{if } \bigwedge_{j=2}^{n} \left(\alpha_j^{j-1} \le \beta_j^{j-1} \right) \wedge \left(|c_n| \le \frac{\beta_n^{n-1} - \alpha_n^{n-1}}{d_2 - d_1} \right) \\ \emptyset & \text{otherwise} \end{cases}$$

$$\hat{g}(x) \quad = \quad \begin{cases} \left\{ u \mid \alpha_n^{n-1} - \delta(-c_n) \le u + \sum_{j=1}^{n} a_{nj} x_j \le \beta_n^{n-1} - \delta(c_n) \right\} & \text{if } x \in \hat{W} \\ \mathbf{U} & \text{otherwise} \end{cases}$$

Note that if at the first iteration, we compute

$$\begin{aligned} \alpha_1^1 &= \alpha_1, \qquad \alpha_j^1 = \max(\alpha_j, \alpha_{j-1}^1 - \delta(c_{j-1})) \qquad 2 \le j \le n \\ \beta_1^1 &= \beta_1, \qquad \beta_j^1 = \min\,(\beta_j, \beta_{j-1}^1 - \delta(c_{j-1})) \qquad 2 \le j \le n \end{aligned}$$

then the problem can be solved in one iteration.

Theorem 5 *For systems of the form (4.4) with* $F = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset \mathbb{R}^n$, $\mathbf{U} = \mathbb{R}$ *and* $\mathbf{D} = [d_1, d_2] \subset \mathbb{R}$, *the LCIP is decidable.*

The conditions of Theorem 5 for decidability are somewhat demanding. If, for example, u is bounded, that is, $\mathbf{U} = [u_1, u_2] \subset \mathbb{R}$, then the new constraints added to x during each iteration may change the bounds on x to a non-rectangular polyhedron. For example, in the first iteration, the following constraints are added to x:

$$\left(\alpha_n - \sum_{j=1}^{n} a_{nj}x_j - \delta(-c_n) \le u_2\right) \bigwedge \left(u_1 \le \beta_n - \sum_{j=1}^{n} a_{nj}x_j - \delta(c_n)\right).$$

In this case, the CIP is no longer decidable, and the system falls into the more general class of systems described at the beginning of the section. We conjecture that the LCIP is decidable in a much more general setting, using a completely different algorithm that exploits the stabilizability of the pairs (A, B) and (A, C) and the observability of the pair (A, M).

4.3 Approximate Computation of $\hat{W}$ and $\hat{g}(x)$

One of the disadvantages of the exact calculation proposed in Section 4.1 is that the computation of Pre is still worst case exponential. In this section, we present an algorithm based on semidefinite programming that computes an approximated solution to the LCIP. As a result, we also obtain an algorithm that approximates the solution of the CIP for LDTS with ellipsoidal constraints, i.e. when the sets F, $\mathbf{U}$ and $\mathbf{D}$ are ellipsoids.

First, we show how to compute an ellipsoidal approximation of $W^1 = \mathrm{Pre}(W^0)$, and then generalize the procedure to get an ellipsoidal approximation of W^l. As a result, we obtain a polynomial time algorithm for approximating the computation of Pre at each iteration of the controlled invariance algorithm. As we will see in Section 6, the proposed method is very fast and gives a good estimation of the controlled invariance set.

Ellipsoids are represented as $\mathcal{E}_1(P, \hat{x}) = \{x \mid (x - \hat{x})^T P^{-1}(x - \hat{x}) \le 1\}$ where $P \succ 0$ is a positive definite *shape matrix*, and $\hat{x}$ is the *center* of the ellipsoid. An equivalent representation, obtained by using Schur complements, is given by the linear matrix inequality

(LMI):

$$\left[\begin{array}{cc} P & x-\hat{x} \\ (x-\hat{x})^T & 1 \end{array} \right] \succeq 0.$$

This second representation allows the ellipsoids to be *flat* whenever P is positive semidefinite ($P \succeq 0$). Unless otherwise stated, we will assume that ellipsoids are non flat. A third equivalent representation is $\mathcal{E}_2(E,\hat{x}) = \{x \mid x = \hat{x} + Ez, ||z|| \leq 1\}$ where $E = P^{1/2}$. To make the subsequent LMIs more readable, we may use a $*$ to denote elements in the lower triangular part of a symmetric matrix. Further, for a positive semidefinite matrix E, we will use the trace $tr(E)$ as a measure of the size of the ellipsoid $\mathcal{E}_2(E,\hat{x})$.

Returning to the problem at hand, we assume that sets F, $\mathbf{U}$ and $\mathbf{D}$ are given by the ellipsoids $F = W^0 = \mathcal{E}_1(\Omega_0,\hat{x}_0), \mathbf{U} = \mathcal{E}_1(\Gamma,\hat{u}), \mathbf{D} = \mathcal{E}_1(\Delta,\hat{d})$, respectively, where $\Omega_0 \succ 0 \in \mathbb{Q}^{n\times n}$, $\Gamma \succ 0 \in \mathbb{Q}^{n_u\times n_u}$, $\Delta \succ 0 \in \mathbb{Q}^{n_d\times n_d}$, $\hat{x}_0 \in \mathbb{Q}^n$, $\hat{u} \in \mathbb{Q}^{n_u}$, and $\hat{d} \in \mathbb{Q}^{n_d}$.

At each iteration of the controlled invariance algorithm we need to compute

$$W^{l+1} = \mathrm{Pre}(W^l) = \left\{ x \in W^l \mid \exists u_l \in \mathbf{U}\ \forall d_l \in \mathbf{D},\ Ax + Bu_l + Cd_l \in W^l \right\}.$$

For the first iteration, the inner formula is equivalent to

$$\forall d_0 \ \mid\ ||\Delta^{-1/2}(d_0 - \hat{d})|| \leq 1 \Rightarrow \left[\begin{array}{cc} \Omega_0 & Ax + Bu_0 + Cd_0 - \hat{x}_0 \\ * & 1 \end{array} \right] \succeq 0. \tag{4.6}$$

The following lemma [16] will allow us to replace the universal quantifier in (4.6) by an existential quantifier.

Lemma 4 *Let $\mathcal{F} = F^T$, $\mathcal{L} = L^T$ and $\mathcal{R} = R^T$ be given matrices of appropriate size. Then*

$$\left(\forall Z \mid ||Z|| \leq 1 \ \Rightarrow\ \mathcal{F} + LZ\mathcal{R} + (\mathcal{L}Z\mathcal{R})^T \succeq 0\right) \iff \left(\exists \tau \geq 0 \mid \left[\begin{array}{cc} \mathcal{F} - \tau\mathcal{R}^T\mathcal{R} & \mathcal{L} \\ \mathcal{L}^T & \tau I \end{array} \right] \succeq 0\right).$$

Applying Lemma 4 to $Z = \Delta^{-1/2}(d_0 - \hat{d})$,

$$\mathcal{F} = \left[\begin{array}{cc} \Omega_0 & Ax + Bu_0 + C\hat{d} - \hat{x}_0 \\ * & 1 \end{array} \right], \ \mathcal{L} = \left[\begin{array}{c} C\Delta^{1/2} \\ 0 \end{array} \right] \text{ and } \mathcal{R} = \left[\begin{array}{cc} 0 & 1 \end{array} \right],$$

we obtain the equivalent formula

$$\exists \tau_0 \mid \left[\begin{array}{ccc} \Omega_0 & Ax + Bu_0 + C\hat{d} - \hat{x}_0 & C\Delta^{1/2} \\ * & 1 - \tau_0 & 0 \\ * & 0 & \tau_0 I \end{array} \right] \succeq 0.$$

Note that we do not need to keep the constraint $\tau_0 \geq 0$, since it is implicitly implied by the LMI whenever $C \neq 0$. Indeed, the LMI implies that $0 < \tau_0 \leq 1$. Consequently,

$$\mathrm{Pre}(W^0) = \{x \mid \exists u_0 \in \mathbf{U}\ \exists \tau_0\ , \begin{bmatrix} \Omega_0 & x-\hat{x}_0 & 0 & 0 & 0 \\ * & 1 & 0 & 0 & 0 \\ 0 & 0 & \Omega_0 & v_{01} & C\Delta^{1/2} \\ 0 & 0 & * & 1-\tau_0 & 0 \\ 0 & 0 & * & 0 & \tau_0 I \end{bmatrix} \succeq 0\}, \tag{4.7}$$

where $v_{01} = Ax + Bu_0 + C\hat{d} - \hat{x}_0$.

4.3.1 Inner approximations of $\hat{W}$ and $\hat{g}(x)$

As a first stage towards computing an inner approximation of $\hat{W}$ and $\hat{g}(x)$, we compute the maximal inner approximation $\mathcal{E}_2(E_1, \hat{x}_1)$ of $\mathrm{Pre}(W^0)$. Such approximation must satisfy:

$$\forall x \in \mathcal{E}_2(E_1, \hat{x}_1)\ \exists u_0 \in \mathbf{U}\ \exists \tau_0 \mid \begin{bmatrix} \Omega_0 & x-\hat{x}_0 & 0 & 0 & 0 \\ * & 1 & 0 & 0 & 0 \\ 0 & 0 & \Omega_0 & v_{01} & C\Delta^{1/2} \\ 0 & 0 & * & 1-\tau_0 & 0 \\ 0 & 0 & * & 0 & \tau_0 I \end{bmatrix} \succeq 0.$$

This formula is equivalent to:

$$\forall z, ||z|| \leq 1,\ \exists u_0 \in \mathbf{U}\ \exists \tau_0 \mid \begin{bmatrix} \Omega_0 & \hat{x}_1 - \hat{x}_0 + E_1 z & 0 & 0 & 0 \\ * & 1 & 0 & 0 & 0 \\ 0 & 0 & \Omega_0 & \hat{v}_{01} + AE_1 z & C\Delta^{1/2} \\ 0 & 0 & * & 1-\tau_0 & 0 \\ 0 & 0 & * & 0 & \tau_0 I \end{bmatrix} \succeq 0,$$

where $\hat{v}_{01} = A\hat{x}_1 + Bu_0 + C\hat{d} - \hat{x}_0$. In order to apply Lemma 4 to

$$\mathcal{F} = \begin{bmatrix} \Omega_0 & \hat{x}_1-\hat{x}_0 & 0 & 0 & 0 \\ * & 1 & 0 & 0 & 0 \\ 0 & 0 & \Omega_0 & \hat{v}_{01} & C\Delta^{1/2} \\ 0 & 0 & * & 1-\tau_0 & 0 \\ 0 & 0 & * & 0 & \tau_0 I \end{bmatrix},\ \mathcal{L} = \begin{bmatrix} E_1 & 0 \\ 0 & 0 \\ 0 & AE_1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \mathcal{R} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \text{ and } Z = \begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix},$$

we first exchange the quantifiers from $\forall z, ||z|| \leq 1,\ \exists u_0 \in \mathbf{U}\ \exists \tau_0$ to $\exists u_0 \in \mathbf{U}\ \exists \tau_0\ \forall z, ||z|| \leq 1$. Notice that exchanging the quantifiers may reduce the size of the set we want to approximate, and hence the inner ellipsoid we are seeking may be conservative.

Since $||z|| \leq 1$ is equivalent to $||Z|| \leq 1$, Lemma 4 can be applied. After appropriate column and row permutations, the ellipsoidal inner approximation of $\mathrm{Pre}(W^0)$ can be

obtained by solving the SDP

$$
\begin{array}{ll}
\text{maximize} & tr(E_1) \\
\text{subject to} & E_1 \succ 0, \\
& \left[\begin{array}{ccc} \Omega_0 & \hat{x}_1 - \hat{x}_0 & E_1 \\ * & 1-\gamma_0 & 0 \\ * & 0 & \gamma_0 I \end{array}\right] \succeq 0, \left[\begin{array}{cc} \Gamma & u_0 - \hat{u} \\ * & 1 \end{array}\right] \succeq 0, \\
& \left[\begin{array}{cccc} \Omega_0 & \hat{v}_{01} & C\Delta^{1/2} & AE_1 \\ * & 1-\tau_0-\gamma_0 & 0 & 0 \\ * & 0 & \tau_0 I & 0 \\ * & 0 & 0 & \gamma_0 I \end{array}\right] \succeq 0.
\end{array}
$$

Now, we seek to compute an inner approximation of $\mathrm{Pre}^m(W^0)$. For such an approximation to be as close as possible to the true one, instead of using the previously computed ellipsoidal approximation, we keep the exact representation of $\mathrm{Pre}(W^0)$ as in (4.7), then we compute an LMI inner approximation[3] of $\mathrm{Pre}^2(W^0)$ and so on. It is straightforward to see that this procedure leads to the following inner approximation of $\mathrm{Pre}^m(W^0)$:

$$
\begin{array}{l}
\{x \mid \exists u_0 \dots u_{m-1} \in \mathbf{U}\ \exists \tau_0 \dots \tau_{m-1} \mid \left[\begin{array}{cc} \Omega_0 & x - \hat{x}_0 \\ * & 1 \end{array}\right] \succeq 0 \\
\qquad \wedge \left[\begin{array}{ccccc} \Omega_0 & v_{j-1,m} & C\Delta^{1/2} & \cdots & A^{j-1}C\Delta^{1/2} \\ * & 1-\sum_{i=m-j}^{m-1}\tau_i & 0 & \cdots & 0 \\ * & 0 & \tau_{m-j}I & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ * & 0 & 0 & \cdots & \tau_{m-1}I \end{array}\right]_{j=1}^{m} \succeq 0\},
\end{array}
$$

with $v_{j-1,m} = A^j x + \sum_{i=0}^{j-1} A^i(Bu_{m+i-j} + C\hat{d}) - \hat{x}_0$, for all $j = 1 \dots m$. Then the ellipsoidal inner approximation of $\mathrm{Pre}^m(W^0)$ is obtained as:

$$
\begin{array}{ll}
\text{maximize} & tr(E_m) \\
\text{suject to} & E_m \succ 0, \\
& \left[\begin{array}{ccc} \Omega_0 & \hat{x}_m - \hat{x}_0 & E_m \\ * & 1-\gamma_{m-1} & 0 \\ * & 0 & \gamma_{m-1} I \end{array}\right] \succeq 0, \left[\begin{array}{cc} \Gamma & u_i - \hat{u} \\ * & 1 \end{array}\right]_{i=0}^{m-1} \succeq 0, \\
& \left[\begin{array}{cccccc} \Omega_0 & \hat{v}_{j-1,m} & C\Delta^{1/2} & \cdots & A^{j-1}C\Delta^{1/2} & A^j E_m \\ * & \delta_{j,m} & 0 & \cdots & 0 & 0 \\ * & 0 & \tau_{m-j}I & & 0 & 0 \\ \vdots & \vdots & & \ddots & \vdots & \vdots \\ * & 0 & 0 & \cdots & \tau_{m-1}I & 0 \\ * & 0 & 0 & \cdots & 0 & \gamma_{m-1}I \end{array}\right]_{j=1}^{m} \succeq 0,
\end{array}
$$

with $\hat{v}_{j-1,m} = A^j\hat{x}_m + \sum_{i=0}^{j-1} A^i(Bu_{m+i-j} + C\hat{d}) - \hat{x}_0$, for all $j = 1 \dots m$, and $\delta_{j,m} = 1 - \sum_{i=m-j}^{m-1}\tau_i - \gamma_{m-1}$.

[3] In general we do not obtain an exact representation because we exchange the quantifiers to apply Lemma 4.

The controlled invariance algorithm stops if at some iteration m, we have $E_m = E_{m-1}$ and $\hat{x}_m = \hat{x}_{m-1}$. In such a case, an ellipsoidal approximation $\mathcal{E}_2(\Gamma_m, \hat{u}_m) \subseteq \hat{g}(x)$ of the least restrictive controller at state x is obtained as:

$$
\begin{array}{ll}
\text{maximize} & tr(\Gamma_m) \\
\text{subject to} & \Gamma_m \succ 0, \left[\begin{array}{cc} \Gamma & u_i - \hat{u} \\ (u_i - \hat{u})^T & 1 \end{array} \right]_{i=0}^{m-1} \succeq 0, \\
& \left[\begin{array}{cccccc} \Omega_0 & \hat{w}_{j-1,m} & C\Delta^{1/2} & \cdots & A^{j-1}C\Delta^{1/2} & A^{j-1}B\Gamma_m \\ * & \delta_{j,m} & 0 & \cdots & 0 & 0 \\ * & 0 & \tau_{m-j}I & & 0 & 0 \\ \vdots & \vdots & & \ddots & \vdots & \vdots \\ * & 0 & 0 & \cdots & \tau_{m-1}I & 0 \\ * & 0 & 0 & \cdots & 0 & \gamma_{m-1}I \end{array} \right]_{j=1}^{m} \succeq 0,
\end{array}
$$

where $\hat{w}_{j-1,m} = A^j x + \sum_{i=0}^{j-2} A^i(Bu_{m+i-j} + C\hat{d}) + A^{j-1}(B\hat{u}_m + C\hat{d}) - \hat{x}_0$, for all $j = 1 \ldots m$.

Comment 1 *Recall that* Pre *is monotone, hence* $\mathcal{E}_1(\Omega_m, \hat{x}_m) \subseteq \mathrm{Pre}^m(W^0)$. *Therefore, if the sequence of ellipsoids converges, then* $\hat{W}$ *exists and the limit of the sequence is an inner approximation of* $\hat{W}$. *On the other hand, for fixed* m, $\mathrm{Pre}^m(W^0)$ *is an outer approximation of* $\hat{W}$, *hence* $\mathcal{E}_1(\Omega_m, \hat{x}_m)$ *is not guaranteed to be either an inner or an outer approximation of* $\hat{W}$ *if it exists.*

Because we exchanged the order of quantifiers, the above procedure may lead to very conservative inner approximations of both the controlled invariant set and the least restrictive controller. Hence we propose an alternative heuristic algorithm that mixes both inner and outer approximations of Pre.

4.3.2 A less conservative "heuristic" approach

We first compute a polytope containing $\mathrm{Pre}(W^0)$ in (4.7) by solving a series of SDPs of the type

$$
\begin{array}{ll}
\text{maximize} & w_i^T x \\
\text{subject to} & \left[\begin{array}{cc} \Omega_0 & x - \hat{x}_0 \\ * & 1 \end{array} \right] \succeq 0, \left[\begin{array}{cc} \Gamma & u_0 - \hat{u} \\ * & 1 \end{array} \right] \succeq 0, \\
& \left[\begin{array}{ccc} \Omega_0 & Ax + Bu_0 + C\hat{d} - \hat{x}_0 & C\Delta^{1/2} \\ * & 1 - \tau_0 & 0 \\ * & 0 & \tau_0 I \end{array} \right] \succeq 0,
\end{array}
$$

where the vectors $w_i \in \mathbb{R}^n, i = 1 \ldots p$ are chosen arbitrarily. Then an outer approximation of $\mathrm{Pre}(W^0)$ is given by $\{x \mid Mx \leq \beta\}$ where w_i^T is the i-th row of M and β_i is the optimal value of the i-th semidefinite program, $i = 1 \ldots p$.

Then we compute an ellipsoidal inner approximation $\mathcal{E}_2(E_1, \hat{x}_1)$ of the obtained polytope. We want $\forall z ||z|| \leq 1, M(\hat{x}_1 + E_1 z) \leq \beta$. This is equivalent to $\bigwedge_{i=1}^{p} w_i^T \hat{x}_1 + ||E_1 w_i|| \leq \beta_i$. Notice that $E_1 \succ 0$ implies $\beta_i - w_i^T \hat{x}_1 > 0$. Hence, we can use Schur complements to rewrite these constraints as a set of LMIs. Therefore, the ellipsoidal approximation of $\mathrm{Pre}(W^0)$ is obtained as the solution of the SDP:

$$\begin{array}{ll} \text{maximize} & tr(E_1) \\ \text{subject to} & E_1 \succ 0, \left[\begin{array}{cc} (\beta_i - w_i^T \hat{x}_1) I & E_1 w_i \\ (E_1 w_i)^T & \beta_i - w_i^T \hat{x}_1 \end{array} \right]_{i=1}^{p} \succeq 0. \end{array} \tag{4.8}$$

Letting $\Omega_1 = E_1^2$, we can compute an approximation of $\mathrm{Pre}^2(W^0)$ by using the same two step procedure. Generalizing, we get an ellipsoidal approximation $\mathcal{E}_1(\Omega_m, \hat{x}_m)$ of $\mathrm{Pre}^m(W^0)$.

Finally, if the algorithm stops at iteration m, an ellipsoidal approximation $\mathcal{E}_2(\Gamma_m, \hat{u}_m)$ of the least restrictive controller at x can be obtained by solving the SDP:

$$\begin{array}{ll} \text{maximize} & tr(\Gamma_m) \\ \text{subject to} & \Gamma_m \succ 0, \left[\begin{array}{cc} (\beta_i - z_i^T \hat{u}_m) I & \Gamma_m z_i \\ (\Gamma_m z_i)^T & \beta_i - z_i^T \hat{u}_m \end{array} \right]_{i=1}^{p} \succeq 0, \end{array}$$

where β_i, for $i = 1 \ldots p$, is the solution of the SDP

$$\begin{array}{ll} \text{maximize} & z_i^T u \\ \text{subject to} & \left[\begin{array}{cc} \Gamma & u - \hat{u} \\ (u - \hat{u})^T & 1 \end{array} \right] \succeq 0, \\ & \left[\begin{array}{ccc} \Omega_{m-1} & Ax + Bu + C\hat{d} - \hat{x}_0 & C\Delta^{1/2} \\ * & 1 - \tau_{m-1} & 0 \\ * & 0 & \tau_{m-1} I \end{array} \right] \succeq 0. \end{array}$$

Comment 2 *Notice that, $\mathcal{E}_1(\Omega_m, \hat{x}_m)$ is, in general, neither an inner nor an outer approximation of* $\mathrm{Pre}^m(W^0)$. *Hence, the limit of the sequence of ellipsoids, if exists, is not guarranted to be an inner approximation of $\hat{W}$ in this case.*

Chapter 5

CIP for Discrete Time Hybrid Systems

As we mentioned in Section 2 the CIP is decidable for finite state machines, that is DTS whose states and inputs take on a finite number of values. In Section 3, we also argued that the CIP is semi-decidable for a wide class of DTS whose states and inputs are real valued. In this section we bring these two observations together and discuss the CIP for discrete time hybrid systems, that is DTS with some states and inputs taking values in finite sets and others taking values on the reals.

5.1 Discrete Time Hybrid Systems

The following discussion is based on a discrete time version of the hybrid automata of [25, 24]. State variables are partitioned as $S = Q \cup X$, with $\mathbf{Q}$ being a finite set. We use $s = (q, x) \in \mathbf{S}$ to denote the state of the system, with $q \in \mathbf{Q}$ and $x \in \mathbf{X}$ denoting the discrete and continuous state, respectively. Similarly, input variables are partitioned as $V = \Upsilon \cup U \cup \Delta \cup D$. We use (v, u) to denote the control inputs of the system, with $v \in \boldsymbol{\Upsilon}$ and $u \in \mathbf{U}$ denoting the discrete and continuous control inputs, respectively. Finally, we use (δ, d) to denote the disturbance inputs of the system, with $\delta \in \boldsymbol{\Delta}$ and $d \in \mathbf{D}$ denoting the discrete and continuous disturbance inputs, respectively.

Definition 7 (Discrete Time Hybrid System (DTHS)) *A discrete time hybrid system is a collection* $H = (S, V, \mathrm{Init}, \mathrm{Inv}, r, R)$ *consisting of a finite collection of state variables,* S*, a finite collection of input variables,* V*, a set of initial states,* $\mathrm{Init} \subseteq \mathbf{S}$*, an invariant set* $\mathrm{Inv} \subseteq \mathbf{S} \times \mathbf{V}$*, a continuous reset relation,* $r : \mathbf{S} \times \mathbf{V} \to 2^{\mathbf{X}}$ *and a discrete reset relation* $R : \mathbf{S} \times \mathbf{V} \to 2^{\mathbf{S}}$.

Definition 8 (Execution of DTHS) *A sequence $\chi = (s, v) \in (\mathbf{S} \times \mathbf{V})^* \cup (\mathbf{S} \times \mathbf{V})^\omega$ is said to be an execution of the discrete time hybrid system H if $s[0] \in Init$, and for all $k \geq 0$,*

- $s[k+1] \in R(s[k], v[k])$, *or*
- $(s[k], v[k]) \in \text{Inv}$, $q[k] = q[k+1]$ *and* $x[k+1] \in r(s[k], v[k])$.

One can check that a DTHS is non-blocking if for all (s, v) either $R(s, v) \neq \emptyset$ or $(s, v) \in \text{Inv}$ and $r(s, v) \neq \emptyset$.

Like their continuous time counterparts, DTHS can be thought of as directed graphs, with nodes $\mathbf{Q}$ and edges (q, q') for all $q, q' \in \mathbf{Q}$ such that

$$\exists x, \exists x' \in \mathbf{X}, \exists v \in \mathbf{V} \text{ such that } (q', x') \in R(q, x, v).$$

With each node, $q \in \mathbf{Q}$, of the graph we associate a set of initial conditions, an invariance relation and a transition relation given by

$$\begin{aligned} \text{Init}_q &= \{x \in \mathbf{X} \mid (q, x) \in \text{Init}\}, \\ \text{Inv}_q(v) &= \{x \in \mathbf{X} \mid (q, x, v) \in \text{Inv}\}, \\ r_q(x, v) &= \{x' \in \mathbf{X} \mid x' \in r(q, x, v)\}. \end{aligned}$$

With each edge, (q, q') of the graph we associate a guard relation and a reset relation given by

$$\begin{aligned} G_{qq'}(v) &= \{x \in \mathbf{X} \mid \exists x' \in \mathbf{X}, (q', x') \in R(q, x, v)\}, \\ R_{qq'}(x, v) &= \{x' \in \mathbf{X} \mid (q', x') \in R(q, x, v)\}. \end{aligned}$$

For pairs (q, q') which are not edges of the graph, we can set $G_{qq'}(v) = R_{qq'}(x, v) = \emptyset$ for all $x \in \mathbf{X}$ and $v \in \mathbf{V}$. Notice that a guard relation $G_{qq'}(v)$ can be explicitly included in the definition of a DTHS. These two definitions of a DTHS are equivalent if $\forall x \in G_{qq'}(v)$ we have $R(q, x, v) \cap (\{q'\} \times \mathbf{X}) \neq \emptyset$ and empty otherwise. We will assume that a guard relation is explicitly defined from now on.

Theorem 6 *DTHS are a special case of DTS. Therefore all the properties in Section 2 about general DTS (existence of maximal controlled invariant sets, least restrictive controllers, etc.) are inherited by DTHS.*

Proof: A DTHS $H = (S, V, \text{Init}, \text{Inv}, r, R)$ can be viewed as a DTS, $\widehat{H} = (S, V, \text{Init}, f)$ with

$$f(q, x, v) = R(q, x, v) \cup (\{q\} \times \{x' \in \mathbf{X} \mid (q, x, v) \in \text{Inv} \wedge x' \in r(q, x, v)\})$$

■

5.2 CIP for Definable DTHS

As before, a CIP $(H, \Box F)$ consists of a DTHS H and a set $F \subseteq \mathbf{S}$. F can be decomposed into a collection of subsets $F_q \subseteq \mathbf{X}$, $q \in \mathbf{Q}$, by setting

$$F_q = \{x \in \mathbf{X} \mid (q, x) \in F\}.$$

We say that a CIP $(H, \Box F)$ is *definable in a theory of the reals*, if $\mathbf{X} = \mathbb{R}^n$, $\mathbf{U} \subseteq \mathbb{R}^{n_u}$, $\mathbf{D} \subseteq \mathbb{R}^{n_d}$ and the sets $\mathbf{U}$, $\mathbf{D}$, Init_q, $\mathrm{Inv}_q(v)$, $r_q(x, v)$, $G_{qq'}(v)$, $R_{qq'}(x, v)$, and F_q are definable in the same theory, for all $q, q' \in \mathbf{Q}$, $x \in \mathbf{X}$ and $v \in \mathbf{V}$.

Recall that, given a set $W \subseteq \mathbf{S}$, the solution of the CIP is related to the computation of the set $\mathrm{Pre}(W)$. For a DTHS the formula $\psi(s)$ defining $\mathrm{Pre}(W)$ can be expressed as

$$\begin{array}{l} \exists v \in \boldsymbol{\Upsilon}\ \exists u \in \mathbf{U}\ \forall \delta \in \boldsymbol{\Delta}\ \forall d \in \mathbf{D}\ \forall q' \in \mathbf{Q}, \\ [x \in \mathrm{Inv}_q(v) \Rightarrow r_q(x, v, u, \delta, d) \subseteq W_q] \wedge [x \in G_{qq'}(v) \Rightarrow R_{qq'}(x, v, u, \delta, d) \subseteq W_{q'}]\,, \end{array} \tag{5.1}$$

where $W_q = \{x \in \mathbf{X} \mid (q, x) \in W\}$. For a CIP with W_q definable in $\mathrm{OF}(\mathbb{R})$, the quantifiers in $\psi(s)$ can be eliminated from right to left using the standard elimination procedure for $\mathrm{OF}(\mathbb{R})$ [33, 36] as follows:

1. The formula after all the quantifiers is a first order formula in $\mathrm{OF}(\mathbb{R})$, since it can be rewritten as

$$\begin{array}{rcl} \psi_1(q, x, v, u, \delta, d, q') & = & [\neg(x \in \mathrm{Inv}_q) \vee (\forall x'\ \neg(x' \in r_q(x, v, u, \delta, d)) \vee x' \in W_q)] \wedge \\ & & [\neg(x \in G_{qq'}) \vee (\forall x'\ \neg(x' \in R_{qq'}(x, v, u, \delta, d)) \vee x' \in W_{q'})]. \end{array}$$

   The two universal quantifiers over x' can be eliminated in the standard way, so we can assume that ψ_1 is in quantifier free form.

2. The universal quantifier over q' can be eliminated by noting that

$$\psi_2(q, x, v, u, \delta, d) = \forall q' \in \mathbf{Q},\ \ \psi_1(q, x, v, u, \delta, d, q') = \textstyle\bigwedge_{q' \in \mathbf{Q}} \psi_1(q, x, v, u, \delta, d, q').$$

3. The universal quantifier over d can be eliminated by noting that

$$\psi_3(q, x, v, u, \delta) = \forall d \in \mathbf{D},\ \ \psi_2(q, x, v, u, \delta, d) = \forall d\ \ \neg(d \in \mathbf{D}) \vee \psi_2(q, x, v, u, \delta, d),$$

   on which the standard universal quantifier elimination can be performed.

4. The universal quantifier over δ can be eliminated by taking another conjunction

$$\psi_4(q, x, v, u, \delta, d) = \forall \delta \in \boldsymbol{\Delta},\ \ \psi_3(q, x, v, u, \delta) = \textstyle\bigwedge_{\delta \in \boldsymbol{\Delta}} \psi_3(q, x, v, u, \delta).$$

5. The existential quantifier over u can be eliminated by noting that

$$\psi_5(q, x, v) = \exists u \in \mathbf{U},\ \ \psi_4(q, x, v, u) = \exists u\ \ (u \in \mathbf{U}) \vee \psi_4(q, x, v, u),$$

   on which the standard existential quantifier elimination can be performed.

6. The existential quantifier over v can be eliminated by taking a disjunction

$$\psi(q,x) = \exists v \in \boldsymbol{\Upsilon},\ \ \psi_5(q,x,v) = \bigvee_{v\in\boldsymbol{\Upsilon}} \psi_5(q,x,v).$$

An induction argument over this procedure, together with the fact that $\mathrm{OF}(\mathbb{R})$ is decidable, suggests the following generalization of Theorem 3.

Theorem 7 *The class of CIP $(H, \Box F)$ definable in $\mathrm{OF}(\mathbb{R})$, where H is a DTHS, is semi-decidable.*

5.3 CIP for Linear Discrete Time Hybrid Systems

Linear discrete time hybrid systems (LDTHS) are strictly more general than LDTS, even if all reset relations are restricted to be linear maps. The full hierarchy of the classes of systems considered in this paper is shown in Figure 5.1.

As before, a *linear CIP* (LCIP) consists of

- a LDTHS, i.e. a DTHS with $\mathbf{X}$, $\mathbf{U}$ and $\mathbf{D}$ the same as those of a LDTS, $\mathrm{Init}_q = \{x \in \mathbf{X} \mid J_q x \leq \theta_q\}$, $\mathrm{Inv}_q = \{x \in \mathbf{X} \mid K_q x \leq \kappa_q\}$, $G_{qq'} = \{x \in \mathbf{X} \mid L_{qq'} x \leq \xi_{qq'}\}$, $r_q(x,v) = \{A_q x + B_q u + C_q d\}$ if $x \in \mathrm{Inv}_q$ and empty otherwise, $R_{qq'}(x,v) = \{A_{qq'} x + B_{qq'} u + C_{qq'} d\}$ if $x \in G_{qq'}$ and empty otherwise, $A_q \in \mathbb{Q}^{n\times n}$, $B_q \in \mathbb{Q}^{n\times n_u}$, $C_q \in \mathbb{Q}^{n\times n_d}$, $A_{qq'} \in \mathbb{Q}^{n\times n}$, $B_{qq'} \in \mathbb{Q}^{n\times n_u}$, $C_{qq'} \in \mathbb{Q}^{n\times n_d}$, $J_q \in \mathbb{Q}^{m_{ini,q}\times n}$, $\theta_q \in \mathbb{Q}^{m_{ini,q}}$, $K_q \in \mathbb{Q}^{m_{inv,q}\times n}$, $\kappa_q \in \mathbb{Q}^{m_{inv,q}}$, $L_{qq'} \in \mathbb{Q}^{m_{g,qq'}\times n}$, $\xi_{qq'} \in \mathbb{Q}^{m_{g,qq'}}$, with $m_{ini,q}$, $m_{inv,q}$ and $m_{g,qq'}$ being the number of constraints on the initial conditions, invariant and guard relations, respectively; and
- a collection of sets $F_q = \{x \in \mathbb{R}^n \mid M_q x \leq \beta_q\}$ where $M_q \in \mathbb{Q}^{m_q\times n}$, $\beta_q \in \mathbb{Q}^{m_q}$ and m_q is the number of constraints on the continuous state.

Notice that we have assumed $\boldsymbol{\Upsilon} = \boldsymbol{\Delta} = \emptyset$, without loss of generality.

Provided that $\mathrm{Inv}_q \cap G_{qq'} = \emptyset$, LDTHS are non-blocking and deterministic, in the sense that for every state s and every input v there exists a unique next state. Since the sets F, $\mathbf{U}$, $\mathbf{D}$, Inv_q, $G_{qq'}$ are all convex polygons, and the dynamics r and R are given by linear maps, the LCIP is definable in the theory $\mathrm{Lin}(\mathbb{R})$, and therefore, according to the discussion in Section 3, it is semi-decidable. We assume that the sets F, $\mathbf{U}$, Inv_q, $G_{qq'}$ can be either bounded or unbounded, but $\mathbf{D}$ is bounded.

For a LDTHS (5.1) can be simplified to

$$\begin{array}{l}\{x \in W_q \cap \mathrm{Inv}_q \mid \exists u \in \mathbf{U}\ \forall d \in \mathbf{D},\ r_q(x,u,d) \in W_q\} \cup \\ \{x \in W_q \cap G_{qq'} \mid \exists u \in \mathbf{U}\ \forall d \in \mathbf{D},\ R_{qq'}(x,u,d) \in W_{q'}\}.\end{array} \tag{5.2}$$

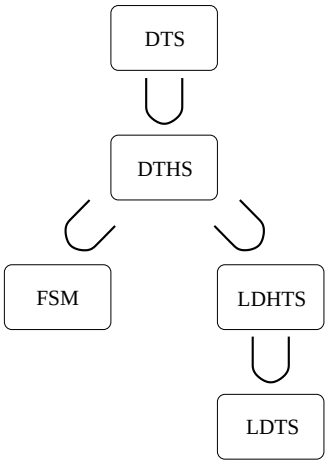


Figure 5.1: A classification of DTS

To compute Pre(W^0) we can use the implementation of Algorithm 1 proposed in Section 4. For subsequent iterations, each W_q usually consists of many disjoint convex polygons. Hence, the method of Section 4 is no longer applicable, because the formula describing W_q has not only conjunctions but also disjunctions. To overcome this limitation, assume that at certain iteration we need to compute the formula $\phi(x) = \exists u \in \mathbf{U}\ \forall d \in \mathbf{D}, \bigvee_{i=1}^{m} \wedge_{j=1}^{k_i} \psi_j^i(x,u,d)$, where $\psi_j^i(x,u,d)$ is an atomic formula in Lin($\mathbb{R}$). By the DeMorgan's law, ϕ is equivalent to

$$\exists u \in \mathbf{U} \neg \left[\exists d \mid d \in \mathbf{D} \wedge \bigwedge_{i=1}^{m} \vee_{j=1}^{k_i} \neg \psi_j^i(x,u,d) \right] = \exists u \in \mathbf{U} \neg \left[\bigvee_{j_1=1}^{k_1} \cdots \bigvee_{j_m=1}^{k_m} \exists d \mid d \in \mathbf{D} \wedge \bigwedge_{i=1}^{m} \neg \psi_{j_i}^i \right].$$

Notice that d can now be eliminated by Fourier elimination. Then ϕ is equivalent to

$$\exists u \in \mathbf{U}\ \textstyle\bigwedge_{j_1=1}^{k_1} \cdots \bigwedge_{j_m=1}^{k_m} \vee_{i=1}^{m} \neg \Phi_{j_1 \ldots j_m}(x,u)$$

where $\Phi_{j_1 \ldots j_m}(x,u) = \exists d \mid d \in \mathbf{D} \wedge \bigwedge_{i=1}^{m} \neg \psi_{j_i}^i(x,u,d)$. Therefore, by further applying DeMorgan's law and Fourier elimination, u can also be eliminated to obtain a first order formula describing Pre(W). However, it is important to mention that applying DeMorgan's law generates a combinatorial number of constraints, and hence the method is not computationally attractive. Also notice that there is no advantage on restricting F_q to be defined by conjunctions only, unless Algorithm 1 converges in the first iteration.

In [39] an alternative method is proposed for performing quantifier elimination on linear inequalities with both conjunctions and disjunctions. The method reduces the number of terms in a formula by using geometrical and logical considerations and by an early detection of superfluous branches. The method is also worst case exponential and therefore the complexity of the CIP problem is the same for LDTS and LDTHS.

Chapter 6

Experimental Results

The algorithms proposed in Sections 4.1, 4.3 and 5.3 were implemented in MATLAB and LMITOOL. In this section, we present four examples that were solved using these implementations. The first example is worked out analytically to show the way the algorithm works. The second example completes the prove of Theorem 4. The third example compares the ellipsoidal approximation against the exact solution. To initialize the SDP based algorithm, we take inner ellipsoidal approximations of the original safe set F and the set of feasible controls $\mathbf{U}$, and an outer ellipsoidal approximation of the set of feasible disturbances $\mathbf{D}$. The last example illustrates the algorithm for discrete time hybrid systems.

6.1 Example 1

The LDTS is defined by $\mathbf{U} = \mathbb{R}$, $\mathbf{D} = [-1, 1]$,

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, M = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ 1 & -1 \\ -1 & 1 \end{bmatrix} \text{ and } \beta = \begin{bmatrix} 80 \\ 40 \\ 80 \\ 70 \end{bmatrix}.$$

1. Initialization $M^0 = M$, $\beta^0 = \beta$.
2. Iteration 1
   (a) Computing $\hat{A}$, $\hat{b}$, $\hat{c}$

$$\hat{A} = \begin{bmatrix} 1 & 2 \\ -1 & -2 \\ -1 & 0 \\ 1 & 0 \end{bmatrix} \qquad \hat{b} = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \qquad \hat{c} = \begin{bmatrix} 2 \\ -2 \\ 0 \\ 0 \end{bmatrix}$$

(b) Computing possible new constraints. Here $P^1 = \{1, 4\}$, $Q^1 = \{2, 3\}$ and $R^1 = \emptyset$

$$\begin{aligned}
\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & -2 \\ 1 & 2 \end{bmatrix} x &\leq \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 40 \\ 80 \end{bmatrix} - \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \\
&\Rightarrow \quad 0 \leq 116 \Rightarrow \text{Redundant} \\
\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 1 & 2 \end{bmatrix} x &\leq \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 80 \\ 80 \end{bmatrix} - \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \\
&\Rightarrow \quad 2x_2 \leq 158 \Rightarrow \text{Redundant} \\
\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & -2 \\ 1 & 0 \end{bmatrix} x &\leq \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 40 \\ 70 \end{bmatrix} - \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} \\
&\Rightarrow \quad -2x_2 \leq 108 \Rightarrow \text{Not Redundant} \\
\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} x &\leq \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 80 \\ 70 \end{bmatrix} - \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
&\Rightarrow \quad 0 \leq 150 \Rightarrow \text{Redundant}
\end{aligned}$$

(c) Computing new M and β

$$M^1 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ 1 & -1 \\ -1 & 1 \\ 0 & -2 \end{bmatrix} \qquad \beta^1 = \begin{bmatrix} 80 \\ 40 \\ 80 \\ 70 \\ 108 \end{bmatrix}$$

3. Iteration 2

(a) Computing $\hat{A}$, $\hat{b}$, $\hat{c}$

$$\hat{A} = \begin{bmatrix} 1 & 2 \\ -1 & -2 \\ -1 & 0 \\ 1 & 0 \\ -2 & -2 \end{bmatrix} \qquad \hat{b} = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ -2 \end{bmatrix} \qquad \hat{c} = \begin{bmatrix} 2 \\ -2 \\ 0 \\ 0 \\ -2 \end{bmatrix}$$

(b) Computing possible new constraints. Here $P^2 = \{1, 4\}$, $Q^2 = \{5\}$ and $R^2 = \emptyset$

$$\begin{aligned}
\begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} -2 & -2 \\ 1 & 2 \end{bmatrix} x &\leq \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 108 \\ 80 \end{bmatrix} - \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \\
&\Rightarrow \quad 2x_2 \leq 262 \Rightarrow \text{Redundant} \\
\begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} -2 & -2 \\ 1 & 0 \end{bmatrix} x &\leq \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 108 \\ 70 \end{bmatrix} - \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} \\
&\Rightarrow \quad -2x_2 \leq 246 \Rightarrow \text{Redundant}
\end{aligned}$$

(c) Therefore $\hat{W}$ and $\hat{g}(x)$ converge to

$$\hat{W} = \left\{ x \mid \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ 1 & -1 \\ -1 & 1 \\ 0 & -2 \end{bmatrix} x \leq \begin{bmatrix} 80 \\ 40 \\ 80 \\ 70 \\ 108 \end{bmatrix} \right\}$$

$$\hat{g}(x) = \begin{cases} \{u \in \mathbf{U} \mid u \geq \max(-38 - x_1 - 2x_2, -80 - x_1, -52 - x_1 - x_2) \\ \qquad u \leq \min(78 - x_1 - 2x_2, 70 - x_1)\} & \text{if } x \in \hat{W} \\ \mathbf{U} & \text{otherwise} \end{cases}$$

Figure 6.1 shows the plot obtained by MATLAB.

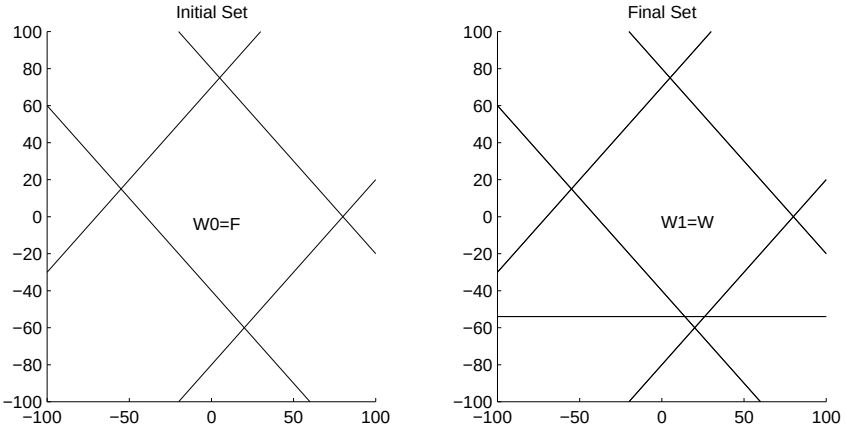


Figure 6.1: Iterations of the algorithm for Example 1

6.2 Example 2

In this example, we use the system defined in Example 1, but with

$$M = \begin{bmatrix} 1 & 1 \\ -1 & -3 \\ 1 & -1 \\ -3 & 1 \end{bmatrix} \text{ and } \beta = \begin{bmatrix} 100 \\ -50 \\ 100 \\ -50 \end{bmatrix}.$$

It is straightforward to see that the only new constraint added in the l-th iteration is $[0\ m_l]x \leq \beta_l$, where $m_l = -10 \cdot 3^{l-1}$, and $\beta_l = -210 - 265(3^{l-1} - 1)$. Therefore after an

infinite number of iterations, $\hat{W}$ and $\hat{g}(x)$ converge to

$$\hat{W} = \left\{ x \mid \begin{pmatrix} M \\ 0 \quad -2 \end{pmatrix} x \leq \begin{pmatrix} \beta \\ -53 \end{pmatrix} \right\}$$

$$\hat{g}(x) = \begin{cases} \{u \in \mathbf{U} \mid u \geq \max(18 - x_1 - \frac{4x_2}{3}, -100 - x_1, -\frac{55}{2} - x_1 - x_2) \\ \qquad\qquad u \leq \min(98 - x_1 - 2x_2, -52 - x_1 + 2x_2)\} & \text{if } x \in \hat{W} \\ \mathbf{U} & \text{else} \end{cases}$$

Figure 6.2 shows the plot obtained by MATLAB for the first three iterations.

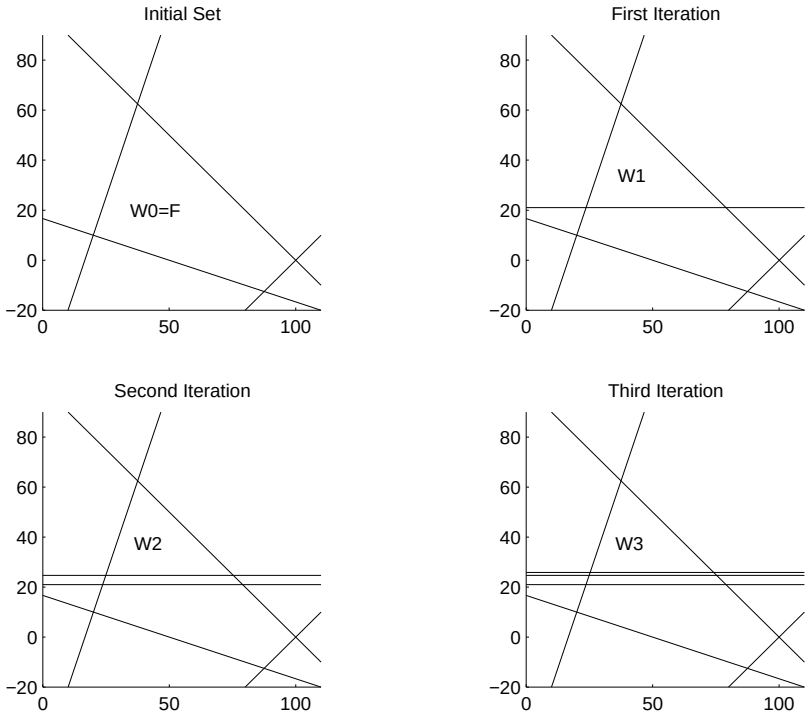


Figure 6.2: Iterations of the algorithm for Example 2

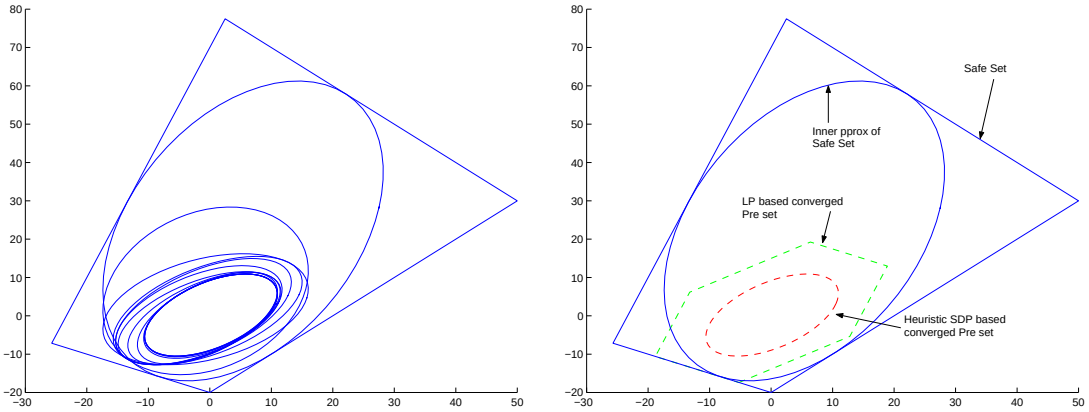
6.3 Example 3

In this example, we consider the LDTS defined by

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix},\ B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ C = \begin{bmatrix} 1 \\ 1 \end{bmatrix},\ \mathbf{D} = [-0.1, 0.1],$$

$$M = \begin{bmatrix} 1 & 1 \\ -1 & -2 \\ 1 & -1 \\ -3 & 1 \end{bmatrix},\ \beta = \begin{bmatrix} 80 \\ 40 \\ 20 \\ 70 \end{bmatrix}, E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \text{ and } \eta = \begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.5 \end{bmatrix}.$$

Figure 6.3 shows a comparison of the exact method (the dashed polytope) and the heuristic SDP approach. In this case the ellipsoidal approximation is a proper subset of the exact maximal controlled invariant set. It is also worth noting that the exact LP method took several hours to compute the safe set, while the heuristic approach took less than a minute of computation time.



Showing the algorithm iterations. Showing the converged result.

Figure 6.3: Comparison of heuristic SDP approach and LP approach for example 3.

6.4 Example 4

Consider the water tank system [1] shown in Figure 6.4. For $i = 1, 2$, let x_i denote the volume of water in Tank i, and d_i denote the flow of water out of Tank i. Let u denote the flow of water into the system, dedicated exclusively to either Tank 1 or Tank 2 at each point in time. The control task is to keep the water volumes above levels l_1 and l_2, respectively. This is to be achieved by a switched control strategy that switches the inflow to Tank 1 whenever $x_1 < r_1$ and to Tank 2 whenever $x_2 < r_2$. We assume that $x_1[0] \geq r_1 > l_1 > 0$ and $x_2[0] \geq r_2 > l_2 > 0$. The continuous dynamics of the water tank are discretized with period τ, so that it can be modeled as the following LDTHS:

- $\mathbf{Q}=\{1,2\}$, $\mathbf{X}=\mathbb{R}^2$, $\mathbf{U}=[u_m, u_M]$ and $\mathbf{D}=[d_m, d_M]^2$;
- $\text{Init} = \mathbf{Q} \times \{x \in \mathbf{X} : (x_1 \geq r_1) \wedge (x_2 \geq r_2)\}$;
- $\text{Inv}_1=\{x \in \mathbf{X} : x_2 \geq r_2\}$, $\text{Inv}_2=\{x \in \mathbf{X} : x_1 \geq r_1\}$;
- $G_{12}=\{x \in \mathbf{X} : x_2 < r_2\}$, $G_{21}=\{x \in \mathbf{X} : x_1 > r_1\}$;
- $r_q(x, u, d) = x + \tau(b_q^T u - d)$, $b_1 = (1, 0)$, $b_2 = (0, 1)$;
- $R_{12}(x, u, d) = R_{21}(x, u, d) = x - \tau d$.

We apply the controlled invariance algorithm to a water tank system with the following parameters: $u_m = 0$, $u_M = 12$, $d_m = 0$, $d_M = 1$, $\tau = 1$, $r_1 = r_2 = 20$ and $l_1 = l_2 = 10$. The controlled invariance algorithm converges after 11 iterations to the following solution:

$$
\begin{aligned}
\hat{W}_1 &= \{x \in \mathbf{X} \mid x \geq (10, 20) \vee x \geq (21, 11)\} \\
\hat{W}_2 &= \{x \in \mathbf{X} \mid x \geq (20, 10) \vee x \geq (11, 21)\} \\
\hat{g}(1, x) &= \begin{cases} [\max(11 - x_1, 0), 12] & x \geq (10, 21) \\ [\max(22 - x_1, 0), 12] & x \geq (10, 20) \wedge x_2 < 21 \\ [0, 12] & \text{otherwise} \end{cases} \\
\hat{g}(2, x) &= \begin{cases} [\max(11 - x_2, 0), 12] & x \geq (21, 10) \\ [\max(22 - x_2, 0), 12] & x \geq (20, 10) \wedge x_1 < 21 \\ [0, 12] & \text{otherwise} \end{cases} .
\end{aligned}
$$

Finally, for each $(q, x) \in \mathbf{Q} \times \mathbf{X}$ we apply a control that minimizes the flow of water into the system, namely $u^*(q, x) = \min\{u \mid u \in \hat{g}(q, x)\}$. Figure 6.5 shows the maximal controlled invariant set plotted in white, as well as one execution of the system starting at $s[0] = (1, 11, 21)$. Solid trajectories correspond to transitions within the corresponding discrete state, while dashed trajectories correspoond to transitions within the other discrete state or from one state to the other. Notice that the controller applies the minimum possible control to keep the state in the safe set, and hence the execution is always in the boundary of the maximal controlled invariant set.

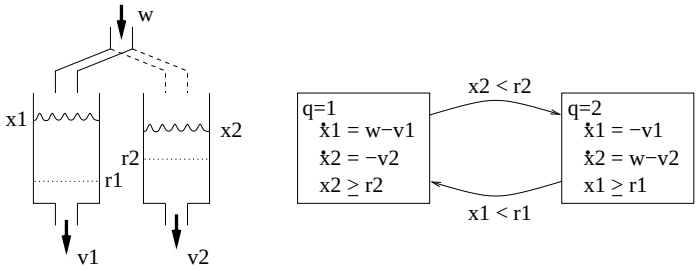


Figure 6.4: The water tank system

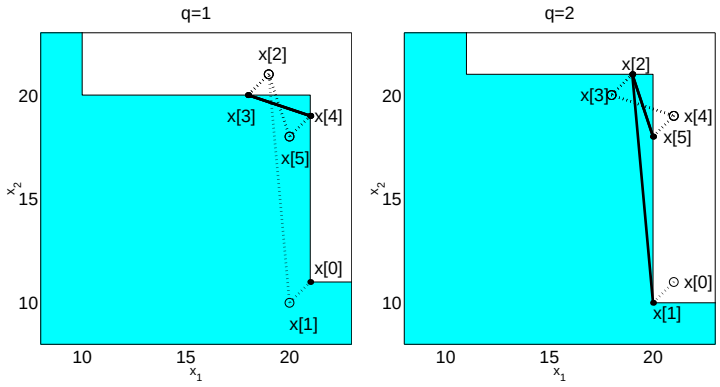


Figure 6.5: Maximal controlled invariant set

Chapter 7

Conclusions and Future Work

We showed that the problem of computing the maximal controlled invariant set and the least restrictive controller for discrete time systems is well posed and proposed a general algorithm for carrying out the computation. We then specialized the algorithm to linear discrete time systems with convex polygonal constraints, and showed how it can be implemented using linear programming and Fourier elimination. The decidability of the problem was also analyzed, and some simple, but interesting cases were found to be decidable. Then, we proposed a robust semidefinite programming based algorithm to approximate the solution of the CIP for systems with ellipsoidal constraints. The algorithm is quite efficient as compared to the exact solution for linear constraints and gives good approximations of the maximal controlled invariant set.

We then extended the proposed method to discrete time hybrid systems. It was shown that the CIP can also be posed as a quantifier elimination problem, and can be solved using Fourier elimination.

We are currently working on sufficient conditions under which the problem is decidable. So far, it seems that the decidability property is not only dependent on the system itself, but also on the initial set, as shown by Example 6.2. It would also be very useful to find a general procedure to eliminate the existential quantifier in a LMI. Such a procedure will avoid the exchange of quantifiers, and hence keep an exact representation of Pre at each iteration of the controlled invariance algorithm. Then a tight ellipsoidal approximation of Pre can be computed and the controlled invariance algorithm can be repeated.

Bibliography

[1] R. Alur and T. Henzinger. Modularity for timed and hybrid systems. In *Concurrency Theory*, volume 1243 of *LNCS*, pages 74–88. Springer Verlag, 1997.

[2] D. Arnon, G. Collins, and S. McCallum. Cylindrical algebraic decomposition I: the basic algorithm. *SIAM Journal on Computing*, 13(4):865–877, 1984.

[3] T. Başar and G. Olsder. *Dynamic Non-cooperative Game Theory.* Academic Press, 2nd edition, 1995.

[4] A. Bensoussan and J. Menaldi. Hybrid control and dynamic programming. *Dynamics of Continuous, Discrete and Impulsive Systems*, (3):395–442, 1997.

[5] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory.* Philadelphia: SIAM, 1994.

[6] M. Branicky, V. Borkar, and S. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998.

[7] P. Caines and Y. Wei. Hierarchical hybrid control systems: A lattice theoretic formulation. *IEEE Transactions on Automatic Control*, 43(4):501–508, April 1998.

[8] P. Cardaliaguet, M. Quincampoix, and P. Saint-Pierre. Set-valued numerical analysis for optimal control and differential games. In *Stochastic and Differential Games: Theory and Numerical Methods*, pages 177–247. Birkhäuser, 1999.

[9] R. Cernikov. The solution to linear programming problems by elimination of unknowns. *Soviet Mathematics Doklady*, 2:1099–1103, 1961.

[10] V. Chandru. Variable elimination in linear constraints. *The Computer Journal*, 36(5):463–472, 1993.

[11] C. Chang and H. Keisler. *Model Theory.* North-Holland, 3rd edition, 1990.

[12] D. Van Dalen. *Logic and Structure.* Springer Verlag, 3rd edition, 1994.

[13] T. Dang and O. Maler. Reachability analysis via face lifting. In *Hybrid Systems: Computation and Control*, volume 1386 of *LNCS*, pages 96–109. Springer Verlag, 1998.

[14] A. Dolzmann and T. Sturm. REDLOG: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, 1997.

[15] L. Fourier. Analyse des travaux de l'Academie Royale des Sciences, pendant l'annee 1824, Partie matematique. Histoire de l'Academie Royale des Sciences de l'Institut de France 7, 1827.

[16] L. El Ghaoui and G. Calafiore. *Robustness in Identification and Control*, chapter Worst-Case Simulation of Uncertain Systems. Springer Verlag, 1999.

[17] G. Grammel. Maximum principle for a hybrid system via singular perturbations. *SIAM Journal of Control and Optimization*, 37(4):1162–1175, 1999.

[18] M. Greenstreet and I. Mitchell. Integrating projections. In *Hybrid Systems: Computation and Control*, volume 1386 of *LNCS*, pages 159–174. Springer Verlag, 1998.

[19] M. Heymann, F. Lin, and G. Meyer. Control synthesis for a class of hybrid systems subject to configuration-based safety constraints. In *Hybrid and Real Time Systems*, volume 1201 of *LNCS*, pages 376–391. Springer Verlag, 1997.

[20] A. Kurzhanski and I. Vályi. *Ellipsoidal Calculus for Estimation and Control.* Boston: Birkhäuser, 1997.

[21] A. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis. In *Hybrid Systems: Computation and Contro*, volume 1790 of *LNCS*, pages 202–214. Springer Verlag, 2000.

[22] C. Lassez and J. Lassez. Quantifier elimination for conjunctions of linear constraints via a convex hull algorithm. In *Symbolic and Numeric Computation for Artificial Intelligence*, pages 103–122. Academic Press, 1992.

[23] J. Lewin. *Differential Games.* Springer Verlag, 1994.

[24] J. Lygeros, K. Johansson, S. Sastry, and M. Egerstedt. On the existence of executions of hybrid automata. In *IEEE Conference on Decision and Control*, pages 2249–2254, 1999.

[25] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, pages 349–370, March 1999.

[26] O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Theoretical Aspects of Computer Science*, volume 900 of *LNCS*, pages 229–242. Springer Verlag, 1995.

[27] A. Nerode and W. Kohn. Multiple agent hybrid control architecture. In *Hybrid Systems*, volume 736 of *LNCS*, pages 297–316. Springer Verlag, 1993.

[28] G. Pappas. *Hybrid Systems: Computations and Abstractions.* PhD thesis, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, 1998.

[29] G. Pappas, G. Lafferriere, and S. Sastry. Hierarchically consistent control systems. In *IEEE Conference on Decision and Control*, pages 4336–4341, 1998.

[30] B. Piccoli. Necessary conditions for hybrid optimization. In *IEEE Conference on Decision and Control*, pages 410–415, 1999.

[31] P. Ramadge and W. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, Vol.77(1):81–98, 1989.

[32] H. Royden. *Real Analysis*. MacMillan, 3rd edition, 1988.

[33] A. Seidenberg. A new decision method for elementary algebra. *Annals of Mathematics*, 60:387–374, 1954.

[34] O. Shakernia, G. Pappas, and S. Sastry. Decidable controller synthesis for classes of linear systems. In *Hybrid Systems: Computation and Control*, volume 1790 of *LNCS*, pages 407–420. Springer Verlag, 2000.

[35] H. Sussmann. A maximum principle for hybrid optimal control problems. In *IEEE Conference on Decision and Control*, pages 425–430, 1999.

[36] A. Tarski. *A decision method for elementary algebra and geometry*. University of California Press, 1951.

[37] W. Thomas. On the synthesis of strategies in infinite games. In *Proceedings of STACS 95, Volume 900 of LNCS*, pages 1–13. Springer Verlag, Munich, 1995.

[38] C. Tomlin, J. Lygeros, and S. Sastry. Computing controllers for nonlinear hybrid systems. In *Hybrid Systems: Computation and Control*, volume 1569 of *LNCS*, pages 238–255. Springer Verlag, 1999.

[39] V. Weispfenning. Simulation and optimization by quantifier elimination. *Journal of Symbolic Computation*, 24:189–208, 1997.

[40] H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *IEEE Conference on Decision and Control*, pages 4607–4613, 1997.