Controller Synthesis of Discrete Linear Plants Using Polyhedra

MATTEO SLANINA
Stanford University
and
SRIRAM SANKARANARAYANAN
NEC Laboratories America
and
HENNY B. SIPMA and ZOHAR MANNA
Stanford University

We study techniques for synthesizing synchronous controllers for affine plants with disturbances, based on safety specifications. Our plants are modeled in terms of discrete linear systems whose variables are partitioned into system, control, and disturbance variables. We synthesize non-blocking controllers that satisfy a user-provided safety specification by means of a fixed point iteration over the control precondition state transformer. Using convex polyhedra to represent sets of states, we present both precise and approximate algorithms for computing control preconditions and discuss strategies for forcing convergence of the iteration. We present technique for automatically deriving controllers from the result of the analysis, and demonstrate our approach on examples.

Categories and Subject Descriptors: []:

General Terms:

Additional Key Words and Phrases:

1. INTRODUCTION

We propose a new method to synthesize controllers for linear discrete systems with disturbances. Given a plant description and a safety specification, the method computes a control invariant that strengthens the specification and uses this control invariant as the basis for constructing a *non-blocking* controller that guarantees that the system is *live* and satisfies the specification.

Plants are modelled as discrete-time linear transition systems, similar to the piece-wise affine (PWA) systems of Sontag [1981] and the mixed logical dynamical (MLD) system model studied by Bemporad & Morari [1999]. The (real-valued)

Author's address: Matteo Slanina, Stanford University, Computer Science Department, 353 Serra Mall, Stanford, CA 94305-9045.
e-mail: matteo@cs.stanford.edu

variables of the plants are partitioned into system, control and disturbance (drift) variables. The transitions are expressed by linear inequality constraints relating the state variables before and after the transition. The behavior of the plants is represented by infinite sequences of plant configurations that are the outcome of games played by the three principal actors of the model: a Scheduler, which picks the transition to be executed next, a Controller, which provides the control input, and a Drifter, which creates the disturbances.

The method to compute the control invariant resembles backward propagation used in invariant generation, with the exception that it uses a non-blocking control precondition as predicate transformer instead of a weakest precondition. Techniques from abstract interpretation [Cousot and Cousot 1977] are used to construct and justify a backward propagation scheme that underapproximates the computation of a greatest fixed point. An underapproximating widening operator is proposed to force convergence. We specialize the method for the domain of convex polyhedra and demonstrate how the control precondition can be computed exactly in many common cases and how it can be efficiently underapproximated in the remaining cases.

The method to synthesize the controller uses the computed control invariant to construct a plant refinement that restricts the controller to signals that are guaranteed to keep the system within its specification. An method using triangulation is proposed to extract from this restriction an efficient implementation of the controller.

We have implemented our method using PPL [Bagnara et al. 2002] and applied it to a case study consisting of a network of buffers.

Related Work. Discrete plant models have been widely used in the control theory community with difference equation models being the classical discrete analog of differential equations. Finite state systems have been studied in the framework of Ramadge and Wonham [1989]. In this framework, the plant can be controlled by selectively disabling or enabling certain actions over time. A number of recent papers have explored the implementation of the Ramadge-Wonham framework over infinite-state plants using abstract interpretation [Kumar and Garg 2005] and polyhedra-based abstraction [Le Gall et al. 2005]. Piecewise affine system models [Sontag 1981] and mixed logical dynamical systems [Bemporad and Morari 1999] integrate many aspects of finite state systems and difference equations. The safety specifications used for the controller synthesis are based mainly on the avoidance of a given unsafe region. Frequently, stability is the only liveness property.

Our plant models are *synchronous*, i.e., the plant actions including the application of the control and disturbances are all synchronized to occur at fixed time points. Synchronous modelling has been used by the computer systems community to model industrial-scale systems with widely used modelling languages such as Esterel and Lustre. The controller synthesis is presented in the setting of infinite games and alternating systems [Alur et al. 2002]. The actions of the plant are modeled as an infinite game played by the controller against an adversarial environment. The framework makes fewer assumptions on the nature of the plant or the controller. It has been applied to infinite-state systems through abstraction [Henzinger et al. 2000], interactive proof methods [Slanina et al. 2006], and inter-

face specifications for program modules [de Alfaro and Henzinger 2005]. However, the computational complexity of reasoning about these games makes it hard to apply this framework unless an abstraction is used.

Hybrid models such as timed and hybrid automata necessitate controller synthesis techniques from both communities. Recent approaches synthesize controllers for plant models used in control theory for temporal logic specifications. Asarin et al. investigate controller synthesis for plants modeled as linear hybrid automata and for safety objectives [Asarin et al. 2000]. Kloetzer and Belta use polyhedral analysis to control switched systems for LTL specifications [Kloetzer and Belta 2006]. Tabuada and Pappas [2003] discuss LTL controllability for infinite-state discrete systems that roughly correspond to our plants with a single transition, and without disturbances.

2. PRELIMINARIES

We introduce the computational model for linear plants. In this model, all relations and assertions are drawn from the first-order theory of reals without multiplication. We first recall some definitions from linear algebra [Schrijver 1986].

2.1 Polyhedra

An *affine expression* is of the form $e : \ \vec{a}^{\mathrm{T}}\vec{x} + b$, where $\vec{a}$ and $\vec{x}$ are $n \times 1$ column vectors and b is a scalar. A *linear assertion* is a finite conjunction of linear inequalities of the form $\bigwedge_i e_i \geq 0$ where each e_i is an affine expression. A linear assertion can be expressed as $A\vec{x} + \vec{b} \geq \mathbf{0}$, where A is an $m \times n$ matrix, $\vec{b}$ is an $m \times 1$ column vector, and $\geq$ is interpreted entry-wise. The set of points satisfying a linear assertion, denoted by $[\![\varphi]\!]$, is a *polyhedron*. Any polyhedron φ may also be represented by a set of *generators*: a set of *vertices* $V = \{\vec{v_1}, \ldots, \vec{v_m}\}$ and *rays* $G = \{\vec{r_1}, \ldots, \vec{r_p}\}$ such that

$$[\![\varphi]\!] = \left\{ \vec{x} = \sum_{j=1}^{m} \lambda_j \vec{v}_j + \sum_{i=1}^{p} \mu_i \vec{r}_i \;\middle|\; \lambda_j, \mu_i \geq 0 \text{ and } \sum_{j=1}^{m} \lambda_j = 1 \right\}$$

Given a linear assertion $\varphi : A\vec{x} + \vec{b} \geq \mathbf{0}$, called the *constraint representation*, the number of vertices and rays of the dual generator representation may be exponential in the number of constraints (the n-dimensional hypercube is an example) and vice versa. Nevertheless, the problems of computing the generators given the assertion and vice versa have been well studied and practically efficient algorithms are available [Fukuda and Prodon 1996].

2.2 Linear Controlled Transition Systems

The omputational model that we use is a variant of alternating transition systems [Slanina et al. 2006; Alur et al. 2002]), specialized for modeling control systems.

Definition 1 *Linear Controlled Transition System* (LCTS).
A *linear controlled transition system (*LCTS*)* $\Pi : \langle \vec{x}, \vec{u}, \vec{d}, L, \mathcal{I}, \mathcal{T} \rangle$ consists of:

—$\vec{x} : \{x_1, \ldots, x_n\}$: a vector of *system* (state) variables; a *system state* $s \in \mathbb{R}^n$ is a valuation of all variables; $\Sigma = \mathbb{R}^n$ denotes the set of all states.

—$\vec{u}$: vector of *control* input variables;

—$\vec{d}$: vector of *disturbance* variables;
—L: finite set of *locations*;
—$\mathcal{I}$: map from L to linear assertions over $\vec{x}$;
—$\mathcal{T}$: finite set of *transitions*. Each $\tau : \langle \ell, m, \xi_u, \xi_d, f\rangle \in \mathcal{T}$ consists of
  —$\ell, m \in L$: a source and target location, respectively;
  —ξ_u: a linear assertion over $\vec{x}$ and $\vec{u}$ restricting the control variables;
  —ξ_d: a linear assertion over $\vec{x}$ and $\vec{d}$ restricting the disturbance variables;
  —f: the *transition function*, an affine transformation of the form $\vec{x}' = f(\vec{x}, \vec{u}, \vec{d}) = A\vec{x} + B\vec{u} + C\vec{d} + \vec{b}$, specifying the *action* taken by the transition, where $\vec{x}'$ denotes the value of variables $\vec{x}$ in the next state.

Example 1 *Flow Controller.* As an illustration of an LCTS consider a buffer that smooths the flow of packets. The system is modeled by the LCTS $\mathcal{B} : \langle \{x_1, x_2\}, u, d, \{\ell_0\}, \mathcal{I}, \{\tau\}\rangle$, where x_1, x_2 represent the buffer occupancy and output flowrate, respectively, u represents the adjustment in the output flowrate, and d represents the input flow rate. The system is modeled with a single location ℓ_0 with location invariant $\mathcal{I}(\ell_0) : x_1 \geq 0 \wedge x_2 \geq 0$. The system has a single transition $\tau : \langle \ell_0, \ell_0, \xi_u, \xi_d, f\rangle$ with control restriction $\xi_u : \ -1 \leq u \leq 1$, requiring that the change in output flowrate per time unit be less than 1, and disturbance restriction $\xi_d : \ 0 \leq d \leq 4$, which limits the input flowrate. The transition relation is given by

$$f : \begin{cases} x_1 := x_1 + d - x_2 & d \text{ packets added, } x_2 \text{ packets removed} \\ x_2 := x_2 + u & \text{adjustment to output flowrate} \end{cases}$$

or

$$f : \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u + \begin{pmatrix} 1 \\ 0 \end{pmatrix} d$$

Semantics. The semantics of an LCTS is defined in terms of an infinite game played between three players: a *Scheduler* (S), a *Controller* (U), and a *Drifter* (D). Starting from a *configuration* $\langle \ell, s\rangle$, with $\ell \in L$ and s a system state in $[\![\mathcal{I}(\ell)]\!]$, a *play* consists of choices in the following order:

(1) S chooses an outgoing transition $\tau = \langle \ell, m, \xi_u, \xi_d, f\rangle$;
(2) based on s and τ, U chooses a control $\vec{u}$ satisfying $\xi_u(s, \vec{u})$, and
(3) based on s, τ, and $\vec{u}$, D chooses an input $\vec{d}$ satisfying $\xi_d(s, \vec{d})$.

A play determines the next configuration $\langle m, s'\rangle$ such that s' is the result of applying f to s based on the chosen values of $\vec{u}$ and $\vec{d}$. The play is admissible only if $s' \in [\![\mathcal{I}(m)]\!]$, denoted $\langle \ell, s\rangle \rightsquigarrow \langle m, s'\rangle$. An inadmissible play is written $\langle \ell, s\rangle \rightsquigarrow \perp$.

Example 2. Revisiting Ex. 1, let $c_0 : \ \langle \ell_0, (x_1 : \ 0, x_2 : \ 2)\rangle$ be a configuration. Suppose S chooses τ, U chooses $u = -1$ and D chooses $d = 2$; the resulting play is $c_0 \rightsquigarrow \langle \ell_0, (0, 1)\rangle$, because $(0, 1) \in [\![\mathcal{I}(\ell_0)]\!]$. On the other hand, if D were to choose $d = 0$, the resulting play would be $s_0 \rightsquigarrow \perp$, because the resulting configuration is $\langle \ell_0, (-2, 1)\rangle$, and $(-2, 1) \notin [\![\mathcal{I}(\ell_0)]\!]$.

Definition 2 *Run.* A *run* of an LCTS starting from a configuration $\langle \ell_0, s_0\rangle$ is a finite or infinite sequence of plays $\langle \ell_0, s_0\rangle \rightsquigarrow \langle \ell_1, s_1\rangle \rightsquigarrow \ldots$
A run is finite only if it ends in a configuration $\langle \ell_i, s_i\rangle$ from which it is impossible for any player to make choices satisfying ξ_u and ξ_d, or if it ends in $\perp$.

To limit finite runs we assume the following syntactic restrictions:

—Each $\ell \in L$ has an outgoing transition τ, and therefore the scheduler can always pick a transition;

—For all configurations $\langle \ell, s \rangle$ such that $s \in [\![\mathcal{I}(\ell)]\!]$, and all transitions $\tau : \langle \ell, m, \xi_u, \xi_d, f \rangle$ there exist $\vec{u}$ and $\vec{d}$ such that $[\![\mathcal{I}(\ell)]\!] \vDash \exists \vec{u}.\, \xi_u(\vec{x}, \vec{u})$ and $[\![\mathcal{I}(\ell)]\!] \vDash \exists \vec{d}.\, \xi_d(\vec{x}, \vec{d})$, and therefore the Controller and the Drifter can always make a choice independent of the Scheduler.

These restrictions ensure that from any configuration that satisfies the location invariant, there always exists a choice for the three players. Some of these choices, however, may still lead to an inadmissible play, and thus to a finite run. It is the task of the Controller to avoid inadmissible plays. Its ability to do so depends on the existence of a control strategy.

Definition 3 *Control Strategy.* A *(control) strategy* is a function $\vec{u} = f_U(\langle \ell, s \rangle, \tau)$ that maps every configuration $\langle \ell, s \rangle$, such that $s \in [\![\mathcal{I}(\ell)]\!]$, and transition $\tau = \langle \ell, m, \xi_u, \xi_d, f \rangle \in \mathcal{T}$ to a control input $\vec{u}$ such that $\xi_u(s, \vec{u})$.

Definition 4 *Blocking Configuration.* A configuration c is called *blocking* if, starting from c, the Controller has no strategy to prevent an inadmissible play, i.e., there exists a choice of the Scheduler such that for all choices of the Controller there exists a choice of the Drifter that leads to an inadmissible play.

Objective. Given a system Π and a specification Ψ, the goal is to keep Π within Ψ without the system reaching a blocking configuration. More formally, given an LCTS Π and a *safety objective* Ψ, an assertion map that maps locations to linear assertions, the goal is to find an assertion map η such that for any location $\ell \in L$, $\eta(\ell) \vDash \Psi(\ell)$ and from any $\langle \ell, s \rangle$ such that $s \in [\![\eta(\ell)]\!]$, the controller has a strategy to ensure an admissable play $\langle \ell, s \rangle \rightsquigarrow \langle \ell', s' \rangle$ such that $s' \in [\![\eta(\ell')]\!]$.

In the next section we describe an abstract method for obtaining such an assertion map from a safety objective. In section 4 we provide a concrete instance of that method using polyhedra.

3. CONTROLLABILITY

Given a system and safety objective Ψ, the goal is to have the controller keep the system wihin Ψ. Not every safety objective, however, is controllable.

Definition 5 *Controllability for Safety.* An LCTS Π is *controllable* for a safety objective Ψ if, starting from any configuration $\langle \ell_0, s_0 \rangle$ such that $s_0 \vDash \Psi(\ell_0)$, there exists a strategy for U, playing against S and D, such that every resulting run $\langle \ell_0, s_0 \rangle \rightsquigarrow \langle \ell_1, s_1 \rangle \rightsquigarrow \ldots$ of Π is infinite and satisfies Ψ, i.e., $\forall\ i,\ s_i \vDash \Psi(\ell_i)$.

Example 3. Consider again the system from Example 1. The safety objective $\Psi(\ell_0) :\ 0 \le x_1 \le 20\ \wedge\ 0 \le x_2 \le 4$ is not controllable. For instance, at the configuration $\langle \ell_0, (x_1 : 0, x_2 : 4) \rangle$, U does not have a strategy to prevent an inadmissible play: D may ensure this simply by choosing $d < 4$. Similarly, for the configuration $\langle \ell_0, (x_1 : 20, x_2 : 0) \rangle$, U does not have a strategy to ensure $\Psi(\ell_0)$ in the next step. The assertion map

$$\Phi(\ell_0) : \begin{bmatrix} 0 \le x_2 \le 4\ \wedge\ x_2 \le x_1 \le x_2 + 16\ \wedge\ 2x_2 - 1 \le x_1 \le 2x_2 + 13 \\ 3x_2 - 6 \le x_1 \le 3x_2 + 11\ \wedge\ 4x_2 - 6 \le x_1 \le 4x_2 + 10 \end{bmatrix}.$$

is controllable and strengthens $\Psi(\ell_0)$. Such a map is derived in the next section.

3.1 Control Invariant

To construct a method for establishing controllability we introduce the notion of a *control invariant map*, the counterpart of an invariant map used in the analysis of regular transition systems [Manna and Pnueli 1995].

Definition 6 *Control Invariant Map.* Given an LCTS Π, an assertion map Ψ is a *control invariant map* of Π if Π is controllable for Ψ.

For transition systems (without control), we know that an assertion map χ can be proven invariant if it is *inductive*, i.e., if for each $\tau \in \mathcal{T}$, χ implies the weakest precondition of χ with respect to τ: $\chi \vDash wpc(\chi, \tau)$.

Analogously, for alternating systems, an assertion map Ψ is control invariant if, for each $\tau \in \mathcal{T}$, Ψ implies its *control precondition* with respect to τ:

Definition 7 *(Non-Blocking) Control Precondition.* Given an LCTS $\Pi : \langle \vec{x}, \vec{u}, \vec{d}, L, \mathcal{I}, \mathcal{T} \rangle$, let $\tau : \langle \ell, m, \xi_u, \xi_d, f \rangle$ be a transition in $\mathcal{T}$ and $\varphi(\vec{x})$ an assertion such that $\varphi \vDash \mathcal{I}(m)$. The *(non-blocking) control precondition* of φ with respect to τ, denoted $\mathrm{cpre}(\varphi, \tau)$, is

$$\mathrm{cpre}(\varphi, \tau) : \ \mathcal{I}(\ell) \ \wedge \ \exists \vec{u}. \left(\xi_u \wedge \forall \vec{d}. \, \xi_d \ \rightarrow \ \varphi(f(\vec{x}, \vec{u}, \vec{d})) \right) \ ,$$

where $\varphi(f(\vec{x}, \vec{u}, \vec{d}))$ is $\varphi(\vec{x})$ with $f(\vec{x}, \vec{u}, \vec{d})$ substituted for $\vec{x}$.

Thus, $\mathrm{cpre}(\varphi, \tau)$ represents the set of all states s from which U has a (one-step) strategy to ensure that all plays starting from s are admissible and result in a state satisfying φ.

LEMMA 8. *If $\psi : \mathrm{cpre}(\varphi, \tau)$ then starting from configuration $\langle \ell, s \rangle$ with $s \in [\![\psi]\!]$, if S chooses to execute τ, then U has a strategy to ensure that every play is admissible and leads to a configuration $\langle m, s' \rangle$ such that $s' \in [\![\varphi]\!]$.*

In general, for a set of transitions $\tau_1, \ldots, \tau_k$ and assertions $\varphi_1, \ldots, \varphi_k$, the assertion $\bigwedge_{i=1}^{k} \mathrm{cpre}(\varphi_i, \tau_i)$ describes the set of states s from which U has a strategy to ensure that, if τ_i is chosen by S, all plays starting from s are admissible, and the result of executing τ_i satisfies φ_i.

THEOREM 9. *Let Π be an* LCTS *and η be an assertion map. If for all transitions $\tau : \langle \ell, m, \ldots \rangle \in \mathcal{T}$, $\eta(\ell) \vDash \mathrm{cpre}(\eta(m), \tau)$ then η is a control invariant map.*

COROLLARY 10. *Let Π be an* LCTS *and η be an assertion map. If, for all transitions $\tau : \langle \ell, m, \ldots \rangle \in \mathcal{T}$, $\eta(\ell) \vDash \mathrm{cpre}(\eta(m), \tau)$, then Π is controllable for η by means of a memoryless strategy.*

3.2 Backward Propagation

In classical backward propagation, a symbolic simulation is performed using *weakest preconditions* starting from the target assertion φ until a (greatest) fixed point χ is reached. If χ includes all initial states, χ proves that φ is invariant. For controlled systems, we perform backward propagation starting from a given safety objective Ψ. The goal, in this case, is to compute a (largest) set of states that is included in Ψ and is controllable.

Formally, backward propagation starting from a safety objective Ψ is performed using a predicate map transformer $\mathcal{B}$ consisting of a family of predicate transformers $\mathcal{B}_{\langle \ell,m\rangle}$, for each pair of locations ℓ, m such that $\tau : \langle \ell, m, \ldots\rangle \in \mathcal{T}$:

$$\mathcal{B}_{\langle \ell,m\rangle}(\eta(\ell)) \stackrel{\text{def}}{=} \Psi(\ell) \wedge \eta(\ell) \wedge \bigwedge_{\tau:\langle \ell,m,\ldots\rangle\in\mathcal{T}} \text{cpre}(\eta(m),\tau)$$

It computes the sequence $\eta^{(0)} \rightarrow \underbrace{\mathcal{B}(\eta^{(0)})}_{\eta^{(1)}} \rightarrow \underbrace{\mathcal{B}^2(\eta^{(0)})}_{\eta^{(2)}} \rightarrow \ldots$,

with $\eta^{(0)}(\ell) = \mathit{true}$ for all $\ell \in L$, until $\eta^{(i)} = \mathcal{B}(\eta^{(i)}) = \eta^{(i+1)}$. The operator $\mathcal{B}$ is monotonic and, hence, a unique greatest fixed point exists.

THEOREM 11. *Given an* LCTS Π *and a safety objective* Ψ*, the greatest fixed point of* $\mathcal{B}$ *is the weakest control invariant of* Π *that strengthens* Ψ*.*

Similar to propagation-based methods in the analysis of regular transition systems, this approach has two problems: (1) for many domains checking for convergence is not decidable, and (2) convergence may not be reached in a finite number of steps. The classical solution to the first problem is abstract interpretation [Cousot and Cousot 1977], that is, perform the propagation in an abstract domain using an abstract transformer $\mathcal{B}^A$ that guarantees that the fixed point of the abstract transformer is also a fixed point of the original (concrete) transformer.

LEMMA 12. *Let* $\mathcal{B}^A$ *be a predicate map transformer such that* $\mathcal{B}^A$ *underapproximates* $\mathcal{B}$*, i.e., for any assertion map* η*,* $\mathcal{B}^A_{\langle \ell,m\rangle}(\eta(\ell)) \vDash \mathcal{B}_{\langle \ell,m\rangle}(\eta(\ell))$ *for all* $\ell \in L$*. If* $\eta(\ell)$ *is a fixed point of* $\mathcal{B}^A$ *then* $\eta(\ell)$ *is a fixed point of* $\mathcal{B}$*. Moreover,* $\eta(\ell)$ *is a control invariant of* Π *that strengthens* Ψ*.*

The common solution to the second problem is *widening* [Cousot and Cousot 1977; Cousot and Cousot 1992].

Definition 13 *Dual Widening.* A binary operator $\widehat{\bigtriangleup}$ is a *dual widening operator* if it satisfies the following two criteria:

—*Underapproximation:* For any two assertions φ_1 and φ_2, $\varphi_1 \widehat{\bigtriangleup} \varphi_2 \vDash \varphi_1 \wedge \varphi_2$;
—*Descending chain condition:* For any sequence $\varphi_1, \varphi_2, \ldots$ of assertions such that $\varphi_{i+1} \vDash \varphi_i$, the *dual widened* sequence defined by $\psi_1 = \varphi_1$ and $\psi_{i+1} = \psi_i \widehat{\bigtriangleup} \varphi_{i+1}$ always converges.

Backward propagation with widening computes the sequence

$$\eta^{(0)} \rightarrow \underbrace{\mathcal{B}(\eta^{(0)})}_{\eta^{(1)}} \rightarrow \ldots \rightarrow \underbrace{\mathcal{B}^i(\eta^{(0)})}_{\eta^{(i)}} \rightarrow \underbrace{\widehat{\mathcal{B}}(\eta^{(i)})}_{\gamma^{(i+1)}} \rightarrow \underbrace{\widehat{\mathcal{B}}(\gamma^{(i+1)})}_{\gamma^{(i+2)}} \rightarrow \ldots$$

with

$$\widehat{\mathcal{B}}(\eta) \stackrel{\text{def}}{=} \eta \,\widehat{\bigtriangleup}\, \mathcal{B}(\eta)$$

until a fixed point is reached. The underapproximation property of the widening operator with Lemma 12 guarantees that the resulting fixed point is a control invariant, and the descending chain condition guarantees that a fixed point will be reached in a finite number of steps.

THEOREM 14. *Given an* LCTS Π *and an objective* Ψ*, backward propagation with widening computes a control invariant strengthening* Ψ *in finitely many steps.*

4. POLYHEDRAL CONTROLLABILITY

We describe how the generic backward propagation scheme described previously can be implemented efficiently in the domain of polyhedra. Recall that backward propagation requires the repeated computation of the operation $\Psi(\ell) \wedge \eta(\ell) \wedge \bigwedge_{\tau:\langle \ell,m,\ldots\rangle \in \mathcal{T}} \mathrm{cpre}(\eta(m), \tau)$ with control precondition (Def. 5):

$$\mathrm{cpre}(\varphi, \tau) \;\equiv\; \mathcal{I}(\ell) \;\wedge\; \exists \vec{u}.\, (\xi_u \wedge \underbrace{\forall \vec{d}.\, \xi_d \;\rightarrow\; \varphi(f(\vec{x}, \vec{u}, \vec{d}))}_{\varphi_1}) \;,$$

in which we now assume all assertions are linear. In general, the result of this computation may contain disjunctions of linear assertions. Particularly, the elimination of the universal quantifier $\forall \vec{d}$ may result in a union of polyhedra. To compute in the domain of polyhedra we underapproximate φ_1 by a single polyhedron, resulting in a backward propagation operator $\mathcal{B}_P$ that underapproximates $\mathcal{B}$. By Lemma 12, $\mathcal{B}_P$ will still lead to a control invariant. We begin by considering restrictions to the model, so that $\mathcal{B}$ itself results in a single polyhedron.

In many systems the restrictions ξ_d are independent of the actual system state, i.e., ξ_d does not contain variables from $\vec{x}$. In such cases, the assertion $\forall \vec{d}.\, \xi_d \rightarrow \varphi(f(\vec{x}, \vec{u}, \vec{d}))$ can be represented by a linear assertion over $\vec{x}$ and $\vec{u}$.

THEOREM 15. *Let* $\varphi : G\vec{x} + H\vec{u} + J\vec{d} + \vec{c} \geq \mathbf{0}$ *be a linear assertion over* $\vec{x}$*,* $\vec{u}$*, and* $\vec{d}$ *and let* ξ_d *be a linear assertion over* $\vec{d}$ *represented (in generator representation) by vertices* $\vec{v}_1, \ldots, \vec{v}_m$ *and rays* $\vec{r}_1, \ldots, \vec{r}_k$*. Then* $\forall \vec{d}.\, \xi_d \rightarrow \varphi$ *is equivalent to the linear assertion* $\bigwedge_{i=1}^{m} G\vec{x} + H\vec{u} + J\vec{v}_i + \vec{c} \geq \mathbf{0} \quad \wedge \quad \bigwedge_{j=1}^{k} J\vec{r}_j \geq \mathbf{0}$.

Notice that having ξ_d be independent of $\vec{x}$ and $\vec{u}$ does not compromise the ability of the Drifter to choose a disturbance based on $\vec{x}$ and $\vec{u}$.

Example 4. Reconsider the system from Example 1 with safety objective $\varphi : 0 \leq x_1 \leq 20 \wedge 0 \leq x_2 \leq 4$. For transition τ, $\varphi(f(\vec{x}, \vec{u}, \vec{d})) \equiv 0 \leq x_1 + d - x_2 \leq 20 \;\wedge\; 0 \leq x_2 + u \leq 4$. Restriction ξ_d is dependent only on $\vec{d}$, with vertices $\{(d:0),\ (d:4)\}$. Application of Theorem 15 yields (after simplification) $\xi_d \rightarrow \varphi :\ 0 \leq x_1 - x_2 \leq 16 \;\wedge\; 0 \leq x_2 + u \leq 4$. Finally, eliminating u and conjoining with $\mathcal{I}(\ell)$ yields $\mathrm{cpre}(\varphi, \tau) :\ 0 \leq x_1 - x_2 \leq 16 \;\wedge\; 0 \leq x_2 \leq 5$.

As presented above, the universal quantifier elimination requires a vertex enumeration of ξ_d which can be exponential in the number of drift variables. This can be improved using linear programming.

LEMMA 16. *Let* $\varphi : G\vec{x} + H\vec{u} + J\vec{d} + \vec{c} \geq \mathbf{0}$*, let* J_i *denote the* i^{th} *row of* J*, and let* δ_i *be the result of the linear program (LP):* δ_i : *minimize* $J_i\vec{d}$ *subject to* ξ_d *.*
Let $\vec{\delta}$ *represent the vector of values of* δ_i*. If* $\delta_i \neq -\infty$ *for all* i*, then* $\forall \vec{d}.\, \xi_d \rightarrow \varphi$ *is equivalent to* $G\vec{x} + H\vec{u} + \vec{\delta} + \vec{c} \geq \mathbf{0}$*. If some* $\delta_i = -\infty$*, then* $\forall \vec{d}.\, \xi_d \rightarrow \varphi$ *is false (the empty polyhedron).*

Thus, the universal elimination of $\forall \vec{d}.\, \xi_d \rightarrow \varphi$ can be performed in polynomial time and results in an assertion that has the same number of conjuncts as φ.

Table I. Polyhedra encountered in the fixed point iteration for Example 1.

Iter	$\gamma_i :\ \mathrm{cpre}(\eta_i, \tau)$	$\eta^{(i+1)} :\ \eta^{(i)} \wedge \gamma_i$
0	$x_1 - x_2 \in [0,16] \wedge x_2 \in [0,5]$	$x_1 - x_2 \in [0,16] \wedge \eta^{(0)}$
1	$x_1 - 2x_2 \in [-1,13] \ \wedge\ \gamma_0$	$x_1 - 2x_2 \in [-1,13] \wedge \eta^{(1)}$
2	$x_1 - 3x_2 \in [-3,11] \ \wedge\ \gamma_1$	$x_1 - 3x_3 \in [-3,11] \wedge \eta^{(2)}$
3	$x_1 - 4x_2 \in [-6,10] \ \wedge \gamma_2$	$x_1 - 4x_2 \in [-6,10] \wedge \eta^{(3)}$
4	$\equiv \gamma_3$	$\equiv \eta^{(4)}$

If a transition τ does not satisfy the syntactic restriction above, we may still use the theorem by over-approximating ξ_d and thus under-approximating the control precondition operator. A simple overapproximation of ξ_d is given by $\xi'_d \equiv \exists \vec{x}, \vec{u}.\ \mathcal{I}(\vec{x}) \wedge \xi_u(\vec{x}, \vec{u}) \wedge \xi_d(\vec{x}, \vec{d})$.

Example 5 *Flow Controller.* Returning to the flow controller example, we now carry out the fixed point iteration starting from $\eta^{(0)}(\ell) \equiv x_1 \in [0,20] \ \wedge\ x_2 \in [0,4]$. Table I shows the polyhedra encountered during the iteration.

The backward propagation for the example converged in four steps. The domain of polyhedra, however, does not have the finite descending chain property, and thus convergence is not guaranteed in general. We need a widening operator to force convergence.

Definition 17 *Polyhedral Dual Widening Operator.* Given two polyhedra φ_1, φ_2, the dual widening $\varphi \equiv \varphi_1 \widehat{\bigtriangleup}_P \varphi_2$ is defined as follows:

(1) If either polyhedron is empty, then $\varphi \equiv \mathit{false}$.
(2) The generator representation of φ contains those generators of φ_1 that are also generators of φ_2. In other words, we *drop* those generators of φ_1 that are not generators of φ_2.

THEOREM 18. *The operator $\widehat{\bigtriangleup}_P$ over polyhedra is a dual widening operator.*

5. CONTROLLER SYNTHESIS AND IMPLEMENTATION

Given a system Π and safety objective Ψ, the controller may not have a strategy to keep the system within Ψ starting from any configuration in Ψ. It may have a strategy, however, to keep the system within a smaller set of configurations, represented by the control invariant map η. In the previous two sections we presented a method to compute η from Ψ. In this section we show how to use η to refine Π into a new system Π' that is controllable for η and propose an implementation for the controller.

5.1 Refinement

Given an LCTS $\Pi :\ \langle \vec{x}, \vec{u}, \vec{d}, L, \mathcal{I}, \mathcal{T} \rangle$ and a safety objective Ψ, let η be a control invariant map of Π that strengthens Ψ. If $\eta(\ell) = \mathit{false}$ for all $\ell \in L$ then we fail: no refinement is possible based on η. Otherwise we construct a refinement $\Pi' :\ \langle \vec{x}, \vec{u}, \vec{d}, L', \mathcal{I}', \mathcal{T}' \rangle$ as follows:

—$L' \ =\ L - \{\ell \mid \eta(\ell) = \mathit{false}\}$;
—$\mathcal{I}'(\ell) \ =\ \mathcal{I}(\ell) \wedge \eta(\ell)$ for all $\ell \in L'$;

—$\mathcal{T}' = \{\tau : \langle \ell, m, \xi'_u, \xi_d, f\rangle \mid \tau : \langle \ell, m, \xi_u, \xi_d, f\rangle \in \mathcal{T}$ and $\ell, m \in L'\}$ with

$$\xi'_u : \xi_u \wedge \forall \vec{d}.\ \xi_d \rightarrow \eta(m)[\vec{x} \mapsto f(\vec{x}, \vec{u}, \vec{d})] \quad (1)$$

that is, we restrict ξ'_u such that every choice of $\vec{u}$ by the controller leads to a state that satisfies $\eta(m)$.

THEOREM 19. *Let Π' be the refinement of an* LCTS Π *with respect to a control invariant map η. Every run of Π' is (a) a run of Π, (b) non-blocking, and (c) satisfies η.*

System refinement is computationally inexpensive, and yields a refined plant that is guaranteed to keep the system within the original safety objective.

5.2 Controller Implementation

In this section, we synthesize a non-blocking strategy for the controller U. Following the discussion on refining plant specifications, we restrict our attention to the locations in L' and transitions between them. Let $\tau = \langle \ell, m, \xi_u, \xi_d, f\rangle$ be a transition, $\varphi_\ell \equiv \eta(\ell)$, and $\varphi_m \equiv \eta(m)$. By the nature of the set L', both φ_m and φ_ℓ are non-empty.

Specifically, we seek a function $\vec{u} = f_\tau(\vec{x})$ that, given a point $\vec{x} \in [\![\varphi_\ell]\!]$, chooses a control input $\vec{u}$ ensuring that the resulting play satisfies φ_m. The available choices for the control variable $\vec{u}$ are given by ξ'_u from (1).

The construction of an actual controller requires us to choose a single $\vec{u}$ from the available choices given a state $\vec{x}$. Given a state $\vec{x}$ it is possible to obtain constraints over $\vec{u}$ by substituting in ξ'_u. In the polyhedral domain, these constraints are linear inequality and solving them requires solving a linear feasibility problem. Although LP solvers are efficient enough for off-line computation, using them in real time to compute controllers is usually not possible. A more efficient solution can be obtained by storing some precomputed data, as we show next.

Control Using Triangulations. We may choose control inputs at the vertices of $[\![\varphi_\ell]\!]$ and handle interior points based on the choices at the vertices. Assume, for ease of presentation, that $[\![\varphi_\ell]\!]$ is bounded, i.e.,its generator representation does not contain any rays. Let $\vec{v}_1, \ldots, \vec{v}_m$ be the vertices of $[\![\varphi_\ell]\!]$. Any point $\vec{x} \in [\![\varphi_\ell]\!]$ can be written as $\vec{x} = \lambda_1\vec{v}_1 + \cdots + \lambda_m\vec{v}_m$, with $\lambda_i \geq 0$ and $\lambda_1 + \cdots + \lambda_m = 1$. The values of the multipliers λ_i uniquely identify the state $\vec{x}$. If we fix control inputs $\vec{u}_1, \ldots, \vec{u}_m$ at the vertices $\vec{v}_1, \ldots, \vec{v}_m$ of $[\![\varphi_\ell]\!]$, a valid control for the interior point $\vec{x}$ is given by $\vec{u} = f(\vec{x}) = \lambda_1\vec{u}_1 + \cdots + \lambda_m\vec{u}_m$.

LEMMA 20. *If $\vec{u}_1, \ldots, \vec{u}_m$ are valid control inputs at vertices $\vec{v}_1, \ldots, \vec{v}_m$ satisfying $\xi'_u(\vec{v}_i, \vec{u}_i)$ for all i, then $\vec{u} = \lambda_1\vec{u}_1 + \cdots + \lambda_m\vec{u}_m$ is a valid control input at any interior point $\vec{x} = \lambda_1\vec{v}_1 + \cdots + \lambda_m\vec{v}_m$ (i.e., it satisfies $\xi'_u(\vec{x}, \vec{u})$).*

One efficient way to obtain the multipliers $\lambda_1, \ldots, \lambda_m$ is to compute a *triangulation* [Lee 1997] of the polyhedron $[\![\varphi_\ell]\!]$. This divides the polyhedron φ_ℓ into many simplices $\varphi_1, \ldots, \varphi_k$. The design of a controller for the transition τ on the entire polyhedron φ_ℓ is thus reduced to designing acontroller for each simplex in the triangulation. The resulting controller can then use a search tree to locate the simplex to which a given state $\vec{x}$ belongs in the triangulation and compute the

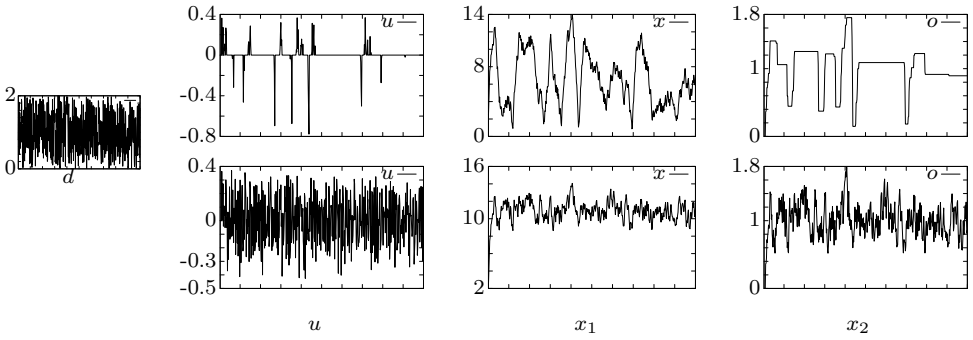


Fig. 1. Values of the control, buffer occupancy x_1 and output flow rate x_2 for same drift pattern (shown left). The controller on the top row minimizes $|u|$ while the bottom row controller minimizes u.

control based on this simplex, with a total time at most quadratic in the number of system variables:

Let $\vec{v}_0, \ldots, \vec{v}_d$ be the (affinely independent) vertices of a d-simplex φ_i in the triangulation. Any point $\vec{x} \in [\![\varphi_i]\!]$ can be expressed using the barycentric coordinates $\vec{\lambda}$ as $\vec{x} = V\vec{\lambda}$, such that $\vec{\lambda} \geq \mathbf{0}$ and $\mathbf{1}^{\mathrm{T}}\vec{\lambda} = 1$. By the affine independence of the columns of V, we may write $\vec{\lambda} = (V^{\mathrm{T}}V + 1)^{-1}(V^{\mathrm{T}}\vec{x} + \mathbf{1})$ to directly express the barycentric coordinates in terms of the cartesian coordinates. (1 is a matrix of all ones, $\mathbf{1}$ is a vector of all ones.) Let $\vec{u}_0, \ldots, \vec{u}_d$ be the control inputs chosen at the vertices and arranged as the columns of a matrix U. The control inputs at $\vec{x}$ are then given by

$$\vec{u} = U\vec{\lambda} = U(V^{\mathrm{T}}V + 1)^{-1}(V^{\mathrm{T}}\vec{x} + \mathbf{1}) = \Lambda\vec{x} + \vec{x}_0 \ .$$

In other words, the control $\vec{u}$ inside a simplex can be expressed as a (closed form) affine expression over the system variables. The controller for a transition $\tau : \ell \to m$ is obtained by performing a triangulation of φ_ℓ and using the affine control law $\vec{u} = \Lambda_i\vec{x} + x_{0,i}$ in the i-th simplex of the triangulation to compute the control input. Note that if the fixed point iteration computation converges quickly, we expect the resulting polyhedron to have a compact generator representation. As a result, we expect the simplicial decomposition to be tractable.

Example 6. Recall that the result of the fixed point iteration starting from the initial specification is shown in Example 5. Given a particular system state $\langle x_1, x_2 \rangle$, the conditions $\xi'_u(x_1, x_2, u)$ on the single control u form an interval. Any value from this interval is a valid input that maintains safety. We consider two "control strategies": (a) Low flow-rate input: Simply apply the least possible control input. (b) Low "rate-change" control: Minimize $|u|$ to keep the change in x_2 small. Figure 1 contrasts the two control strategies by plotting the control input, x_1 and x_2 observed for the same drift input pattern.

6. APPLICATIONS AND CONCLUSIONS

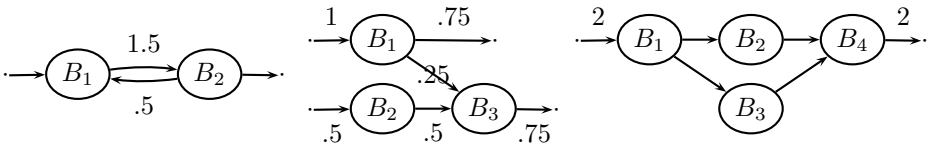
Our prototype PLANTER implements the fixed point iteration over polyhedra using the polyhedral library PPL [Bagnara et al. 2002]. It implements the dual widening

operator described earlier, along with a heuristic extrapolation operator that under-approximates the conjunction but does not guarantee convergence.

Example 7 *Parametric Flow Controller.* We modify the flow controller in Example 1 to include a parameter $\alpha \geq 0$ which restricts the range of control inputs, and thus the rate of change of the ouput flow rate x_2. The control condition for τ is replaced by $\xi_u : (-\alpha \leq u \leq \alpha)$. We wish to determine the smallest value of α for which the system is controllable.

The model is modified to include α as a system variable without any constraints on its values. We obtain a fixed point polyhedron φ, such that $(\min \alpha \text{ s.t. } \varphi) = \frac{4}{9}$. This demonstrates the sufficiency of $\alpha \geq \frac{4}{9}$ for controlling the system. A simple hand calculation demonstrates the necessity of $\alpha \geq \frac{4}{9}$.

Example 8 *Flow network.* Consider networks of buffers as shown below:



Each network has $N > 0$ buffers $B_1, \ldots, B_N$, and weighted edges $i \to j$ signifying that the output flow of B_i is input to B_j. The weight on each edge limits the rate of flow along the edge as a fraction of the maximum possible. Edges without sources model inputs while edges without targets model outputs.

The controller visits each node from $B_1, \ldots, B_N$ in a round-robin fashion. At a visit to a node B_i, it uses the available controls to set the flow rate at each outgoing edge of B_i. After this visit, the flow rates are held constant until the next visit. To adjust the flow rates, the controller requires as many control inputs as the maximum out-degree Δ of the network nodes. Inputs are modelled using drifts. The system variables include $x_1, \ldots, x_N$, to model the buffer occupancies, and y_{ij}, for each edge $i \to j$, to model the flow rates. All buffers are assumed to be of the same size. The system description can be derived from the network specification.

The specification requires us to operate each buffer occupancy without causing overflow/underflow, and restrict flow along edges to satisfy the capacity constraints. The table below shows the run times for controller synthesis on the three networks shown above, measured on a 3 GHz pentium processor with 4 GB RAM:

Network	(#Sys,#Ctrl,#Drift)	#Iter	#Widen	Time (sec)	Controllable?
NET2	$\langle 5, 2, 1 \rangle$	2	0	.2	Yes
NET3	$\langle 8, 2, 2 \rangle$	5	0	114	Yes
NET4	$\langle 9, 2, 2 \rangle$	–time out 30 min–			No

Note that the control problem presented for networks of buffers can be naturally decomposed into instances of the single buffer flow control problem of Example 1 with extensions for multiple output lines. In fact, each well-balanced flow network can be shown to be controllable using such a decomposition. The liberal assumptions made on the drift inputs, in particular their non determinism makes such a decomposition possible.

In the future, we wish to explore domains such as *zonotopes*, which have been used successfully in the analysis of large continuous systems [Girard 2005]. We wish to extend our scheme to incorporate liveness properties in the controller specification. Using the theoretical scheme presented here, we intend to explore ways of realizing real-time controllers for larger real-life plant models.

REFERENCES

Alur, R., Henzinger, T. A., and Kupferman, O. 2002. Alternating-time temporal logic. *Journal of the ACM 49,* 5 (Sept.), 672–713.

Asarin, E., Bournez, O., Dang, T., Maler, O., and Pnueli, A. 2000. Effective synthesis of switching linear controllers for linear systems. *Proceedings of the IEEE 88,* 7 (July).

Bagnara, R., Ricci, E., Zaffanella, E., and Hill, P. M. 2002. Possibly not closed convex polyhedra and the Parma Polyhedra Library. In *SAS*. LNCS, vol. 2477. Springer, 213–229.

Bemporad, A. and Morari, M. 1999. Control of systems integrating logic, dynamics, and constraints. *Automatica 35*, 407–427.

Cousot, P. and Cousot, R. 1977. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL*. ACM Press, 238–252.

Cousot, P. and Cousot, R. 1992. Comparing the Galois connection and widening/narrowing approaches to abstract interpretation. In *PLILP '92*. LNCS, vol. 631. Springer, 269–295.

de Alfaro, L. and Henzinger, T. A. 2005. Interface-based design. In *Engineering Theories of Software-Intensive Systems*. NATO Science Series: Mathematics, Physics, and Chemistry, vol. 195. Springer, 83–104.

Fukuda, K. and Prodon, A. 1996. Double description method revisited. In *Combinatorics and Computer Science*. LNCS, vol. 1120. Springer, 91–111.

Girard, A. 2005. Reachability of uncertain linear systems using zonotopes. In *Hybrid Systems: Computation and Control*. LNCS, vol. 3414. Springer, 291–305.

Henzinger, T. A., Majumdar, R., Mang, F., and Raskin, J.-F. 2000. Abstract interpretation of game properties. In *SAS*. LNCS, vol. 1824. Springer, 220–239.

Kloetzer, M. and Belta, C. 2006. A fully automated framework for control of linear systems from LTL specifications. In *HSCC*, J. Hespanha and A. Tiwari, Eds. LNCS, vol. 3927. Springer, 333–347.

Kumar, R. and Garg, V. K. 2005. On computation of state avoidance control for infinite state systems in assignment program framework. *IEEE Transactions on Automation Science and Engineering 2,* 1 (Jan.), 87–91.

Le Gall, T., Jeannet, B., and Marchand, H. 2005. Supervisory control of infinite symbolic systems using abstract interpretation. In *Proceedings of IEEE CDC-ECC'05*. IEEE Press, 30–35.

Lee, C. W. 1997. Subdivisions and triangulations of polytopes. In *Handbook of Discrete and Computational Geometry*. CRC Press, 271–290.

Manna, Z. and Pnueli, A. 1995. *Temporal Verification of Reactive Systems: Safety*. Springer-Verlag, New York.

Ramadge, P. and Wonham, W. 1989. The control of discrete event systems. *Proceedings of the IEEE: Special Issue on Dynamics of Discrete Event Systems 77,* 1, 81–98.

Schrijver, A. 1986. *Theory of Linear and Integer Programming*. Wiley.

Slanina, M., Sipma, H. B., and Manna, Z. 2006. Proving ATL* properties of infinite-state systems. In *3rd International Colloquium on Theoretical Aspects of Computing (ICTAC2006)*, K. Barkaui, A. Cavalcanti, and A. Cerone, Eds. LNCS, vol. 4281. Springer, 242–256.

Sontag, E. D. 1981. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control 26,* 2 (April), 346 – 357.

Tabuada, P. and Pappas, G. 2003. Model checking LTL over controllable linear systems is decidable. In *HSCC*, O. Maler and A. Pnueli, Eds. LNCS, vol. 2623. Springer, 498–513.

CHANGE LOG

January 5, 2007. Submitted as technical report.