Template-Based Unbounded Time Verification of Affine Hybrid Automata*

Thao Dang[1,**] and Thomas Martin Gawlitza[1,2]

[1] Verimag
{Thao.Dang,Thomas.Gawlitza}@imag.fr
[2] University of Sydney
TGawlitza@usyd.edu.au

Abstract. Computing over-approximations of all possible time trajectories is an important task in the analysis of hybrid systems. Sankaranarayanan et al. [20] suggested to approximate the set of reachable states using template polyhedra. In the present paper, we use a max-strategy improvement algorithm for computing an abstract semantics for affine hybrid automata that is based on template polyhedra and safely over-approximates the concrete semantics. Based on our formulation, we show that the corresponding abstract reachability problem is in $\mathsf{co-NP}$. Moreover, we obtain a polynomial-time algorithm for the time elapse operation over template polyhedra.

1 Introduction

Motivation. Hybrid systems have become widely accepted as a mathematical model appropriate for embedded systems and cyber-physical systems since they allow to describe the mixed discrete-continuous dynamics resulting from integrations of computations and physical processes. Verification is one of the most important questions in the design of such systems. For safety properties, this often leads to reachability analysis. The essential idea of many existing reachability computation techniques could be roughly described as tracking the evolution of the reachable set under the continuous flows using some set represention (such as polyhedra, ellipsoids, level sets, support functions)[1]. Since exact computation is possible only for restrictive classes of continuous dynamics, reachable sets are often approximated using time discretization. Such step-by-step tracking processes can be expensive when time steps should be small for accuracy reasons, and moreover discrete transitions can significantly increase the geometric complexity of reachable sets. This is a reason, besides undecidability of the reachability problem for general hybrid systems, why *unbounded time* reachability computation remains a challenge. Another category of techniques aim at finding approximations which might not be precise but good enough to prove a property of interest. Among such techniques, we can mention the works on barrier certificates [18],

* This work was partially funded by the ANR project VEDECY.

** VERIMAG is a joint laboratory of CNRS, Université Joseph Fourier and Grenoble INP.

[1] The hybrid systems reachability analysis literature is vast. The reader is referred to the recent proceedings of the conferences Hybrid Systems: Control and Computation.

H. Yang (Ed.): APLAS 2011, LNCS 7078, pp. 34–49, 2011.

polynomial invariants [23] and polyhedral invariants [20]), and various discrete abstraction techniques [3–5, 22]). The work we present in this paper is close to the techniques of the second category, in particular to the work by Sankaranarayanan et al. [20].

Sankaranarayanan et al. [20] suggested to approximate the set of reachable states by template polyhedra. Their work is focused on studying the *time elapse operation* for affine hybrid automata over template polyhedra, since this is a challenging problem in hybrid systems verification. They in particular adapted the min-strategy iteration approach of Costan et al. [6] in order to compute a small template polyhedron that safely over-approximates the set of states reachable by continuous evolution. at a single location. Each min-strategy improvement step can be performed in polynomial time through linear programming. The approximation of the set of reachable states their algorithm computes can be used to improve an existing flowpipe construction technique using Taylor series [20]. However, their approach for performing the time elapse operation has disadvantages: (1) Their min-strategy iteration algorithm does not guarantee minimality of the computed template polyhedron. In fact, its accuracy heavily depends on the staying conditions (also called location invariants). If there are no restrictions due to staying conditions, then their algorithm will return too conservative approximations in many cases. (2) The number of min-stratgies is double exponential and a polynomial upper bound for the number of min-strategy improvement steps their algorithm performs is not known.

Contributions. In this paper we propose a remedy for the mentioned disadvantages of the approach of Sankaranarayanan et al. [20]. Moreover, instead of only focusing on the time elapse operation, we study the more general problem of computing *abstract semantics* for affine hybrid automata w.r.t. given linear templates — a problem which is useful for *unbounded time verification*. We emphasize that we provide a max-strategy improvement algorithm that *precisely* computes these abstract semantics and not just safely over-approximates it, as it is often done when using the widening/narrowing approach of Cousot and Cousot [7].

To this end, we firstly reduce our problem to the problem of computing least solutions of systems of inequalities of the form $\mathbf{x}_i \geq f(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, where $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are variables that take values from $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ and f is an operator of a special structure that is in particular monotone and concave (cf. Gawlitza and Seidl [10, 11, 12, 14, 15]). Our max-strategy improvement algorithm for solving these systems of inequalities performs at most exponentially many strategy improvement steps, each of which can be performed in polynomial-time through linear programming. Although only an exponential upper bound is known, the hope is that only a few strategy improvement steps are required for typical examples. As a byproduct of our considerations, we show that the corresponding abstract reachability problem is in $\mathsf{co{-}NP}$. When we apply our method to perform just the time elapse operation, our max-strategy improvement algorithm will perform at most polynomially many strategy improvement steps. Hence, we provide a polynomial-time algorithm for the time elapse operation for affine hybrid automata over template polyhedra.

Related Work. The concepts we present in this paper, strictly generalize the concepts studied by Gawlitza and Seidl [10]. This is no surprise, since affine hybrid automata

are a strict generalization of the affine programs considered by Gawlitza and Seidl [10]. The additional challenge comes from the time elapse operation. The approach of Gawlitza and Seidl [10] and the approach we present in this paper are both based on max-strategy iteration. Costan et al. [6] were the first who suggested to use strategy iteration for computing numerical invariants (for instance w.r.t. to template polyhedra). Strategy iteration can be seen as an alternative to the traditional widening/narrowing approach of Cousot and Cousot [7]. For more information regarding these approaches see Adjé et al. [1, 2], Costan et al. [6], Gaubert et al. [9], Gawlitza and Seidl [10, 11, 12], Gawlitza and Monniaux [13], Gawlitza and Seidl [14, 15].

Corresponding Technical Report. Omitted proofs and reports on our proof-of-concept implementation can be found in the corresponding technical report [8].

2 Basics

Notations. The set of real numbers is denoted by $\mathbb{R}$. The complete linearly ordered set $\mathbb{R} \cup \{-\infty, \infty\}$ is denoted by $\overline{\mathbb{R}}$. The transposed of a matrix A is denoted by $A^\top$. We denote the i-th row (resp. j-th column) of a matrix A by $A_{i\cdot}$ (resp. $A_{\cdot j}$). Accordingly, $A_{i\cdot j}$ denotes the component in the i-th row and the j-th column. We also use this notation for vectors and functions $f : X \to Y^k$, i.e., $f_{i\cdot}(x) = (f(x))_{i\cdot}$ for all $x \in X$ and all $i \in \{1, \dots, k\}$. For $x, y \in \overline{\mathbb{R}}^n$, we write $x \leq y$ iff $x_{i\cdot} \leq y_{i\cdot}$ for all $i \in \{1, \dots, n\}$. $\overline{\mathbb{R}}^n$ is partially ordered by $\leq$. We write $x < y$ iff $x \leq y$ and $x \neq y$. The elements x and y are called *comparable* iff $x \leq y$ or $y \leq x$.

Let $\mathbb{D}$ be a partially ordered set. We denote the *least upper bound* and the *greatest lower bound* of a set $X \subseteq \mathbb{D}$ by $\bigvee X$ and $\bigwedge X$, respectively, provided that they exist. Their existence is in particular guaranteed if $\mathbb{D}$ is a *complete lattice*. The least element $\bigvee \emptyset$ (resp. the greatest element $\bigwedge \emptyset$) is denoted by $\bot$ (resp. $\top$), provided that it exists. We define the binary operators $\vee$ and $\wedge$ by $x \vee y := \bigvee\{x, y\}$ and $x \wedge y := \bigwedge\{x, y\}$ for all $x, y \in \mathbb{D}$, respectively. If $\mathbb{D}$ is a *linearly ordered set* (for instance $\mathbb{R}$ or $\overline{\mathbb{R}}$), then $\vee$ is the *maximum* operator and $\wedge$ the *minimum* operator.

A function $f : \mathbb{D}_1 \to \mathbb{D}_2$, where $\mathbb{D}_1$ and $\mathbb{D}_2$ are partially ordered sets, is called *monotone* iff $x \leq y$ implies $f(x) \leq f(y)$ for all $x, y \in \mathbb{D}_1$. The fixpoint theorem of Knaster/Tarski [21] states that any monotone self-map $f : \mathbb{D} \to \mathbb{D}$ on a complete lattice $\mathbb{D}$ has a least fixpoint $\mu f = \bigwedge\{x \in \mathbb{D} \mid x \geq f(x)\}$.

A mapping $f : \overline{\mathbb{R}}^n \to \overline{\mathbb{R}}^m$ is called *affine* iff there exist $A \in \mathbb{R}^{m \times n}$ and $b \in \overline{\mathbb{R}}^m$ such that $f(x) = Ax + b$ for all $x \in \overline{\mathbb{R}}^n$. Observe that f is monotone, if all entries of A are non-negative. Here, we use the convention $-\infty + \infty = -\infty$. A mapping $f : \overline{\mathbb{R}}^n \to \overline{\mathbb{R}}$ is called *weak-affine* iff there exist $a \in \mathbb{R}^n$ and $b \in \overline{\mathbb{R}}$ such that $f(x) = a^\top x + b$ for all $x \in \overline{\mathbb{R}}^n$ with $f(x) \neq -\infty$. Accordingly, a mapping $f : \overline{\mathbb{R}}^n \to \overline{\mathbb{R}}^m$ is called *weak-affine* iff there exist weak-affine mappings $f_1, \dots, f_m : \overline{\mathbb{R}}^n \to \overline{\mathbb{R}}$ such that $f = (f_1, \dots, f_m)$. Every affine mapping is weak-affine, but not vice-versa. In the following we are in particular interested in mappings that are point-wise minimums of finitely many monotone weak-affine mappings.

Hybrid Automata. In this paper, we study affine hybrid automata. Here, the continuous consecution at each location l is given by an affine vector field V and a staying condition

I that is a convex polyhedron. A vector field V over $\mathbb{R}^n$ is just an operator on $\mathbb{R}^n$. Hence, it can be defined by $V(x) = Ax + b$ for all $x \in \mathbb{R}^n$, where $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$. A *staying conditions* I is simply a subset of $\mathbb{R}^n$. We say that a differentiable time trajectory $\tau : [0, \delta] \to \mathbb{R}^n$ ($\delta \in \mathbb{R}_{\geq 0}$) *evolves from* $\tau(0)$ *to* $\tau(\delta)$ *according to the vector field* V *over* $\mathbb{R}^n$ *while satisfying the staying condition* $I \subseteq \mathbb{R}^n$ iff (1) $\dot{\tau}(t) = V(\tau(t))$ for all $t \in [0, \delta)$, and (2) $\tau(t) \in I$ for all $t \in [0, \delta]$.

Example 1 (Sankaranarayanan et al. [20]). We consider the affine vector field $V : \mathbb{R}^2 \to \mathbb{R}^2$ that is defined by

$$V(x) = Ax + b \qquad \text{for all } x \in \mathbb{R}^2, \text{ where } A = \begin{pmatrix} -1 & 0 \\ 0 & 0 \end{pmatrix} \text{ and } b = \begin{pmatrix} 5 \\ 1 \end{pmatrix}$$

and the staying condition $I = (-\infty, 2.5] \times \mathbb{R}$ that is a convex polyhedron. The polyhedron $P = \{x \in \mathbb{R}^2 \mid x_1 \leq 2.5,\ x_2 \leq 2.5, \text{ and } x_2 \leq x_1\}$ is an invariant in the following sense: each differentiable trajectory that starts in P and evolves according to V while satisfying I stays in P. The situation is illustrated in Figure 1. □

A *hybrid automaton* $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$ consists of the following components:

- n is the number of *continuous variables*.
- $\mathbf{L}$ is a finite set of *locations*.
- $l_0 \in \mathbf{L}$ is the *initial location*.
- $\mathcal{T}$ is a finite set of *discrete transitions*. Each transition $(l_1, \Xi, l_2) \in \mathcal{T}$ consists of a move from the location $l_1 \in \mathbf{L}$ to the location $l_2 \in \mathbf{L}$, and an assertion $\Xi \subseteq (\mathbb{R}^n)^2$.
- $\Theta \subseteq \mathbb{R}^n$ is the set of possible *initial values* of the continuous variables at l_0.
- $\mathbf{D}$ is a mapping that maps each location $l \in \mathbf{L}$ to a vector field $\mathbf{D}(l) : \mathbb{R}^n \to \mathbb{R}^n$.
- $\mathbf{I}$ is a mapping that maps each location $l \in \mathbf{L}$ to a *staying condition* $\mathbf{I}(l) \subseteq \mathbb{R}^n$.

At each location $l \in \mathbf{L}$, the values of the continuous variables evolve according to $\mathbf{D}(l)$ while satisfying $\mathbf{I}(l)$. The assertion $\Xi \subseteq (\mathbb{R}^n)^2$ of a discrete transition $(l, \Xi, l') \in \mathcal{T}$ combines a guard with an assignment.

A hybrid automaton $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$ is called *affine* iff the following statements are fulfilled: (1) The initial condition, location invariants and transition relations are all convex polyhedra.[2] (2) The dynamics $\mathbf{D}(l)$ at each location $l \in \mathbf{L}$ is an affine vector field.

We now introduce our running example. We choose a simple example without discrete transitions, since the main challenges stem from the time elapse operation on which we want to focus in this paper.

Example 2. An affine hybrid automaton is given by $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0\}$, where $n = 2$, $\mathbf{L} = \{1\}$, $\mathcal{T} = \emptyset$, $\Theta = \{x \in \mathbb{R}^2 \mid x_1, x_2 \leq 1 \text{ and } x_2 \leq x_1\}$, $\mathbf{D}(1) = V$, $\mathbf{I}(1) = I$, and $l_0 = 1$. V and I are defined in Example 1. □

A computation of a hybrid automaton is a possibly infinite sequence $(l_0, x_0), (l_1, x_1), \ldots$, where $x_0 \in \Theta$ and, for all $i \in \mathbb{N}$, one of the following statements hold: *(Discrete Consecution)* There exists a discrete transition $(l_i, \Xi, l_{i+1}) \in \mathcal{T}$ such that $(x_i, x_{i+1}) \in \Xi$. *(Continuous Consecution)* $l_i = l_{i+1}$ and there exists a $\delta \in \mathbb{R}_{>0}$ and a differentiable time trajectory $\tau : [0, \delta]$ that evolves from x_i to x_{i+1} according to the vector field $\mathbf{D}(l_i)$ while satisfying the staying condition $\mathbf{I}(l_i)$.

[2] Here, we identify $(\mathbb{R}^n)^2$ with $\mathbb{R}^{2n}$.

Template Polyhedra. As an abstract domain [7] we use template polyhedra as introduced by Sankaranarayanan et al. [19]. For that we fix a *template constraint matrix* $T \in \mathbb{R}^{m\times n}$, where we w.l.o.g. assume that $T_{i\cdot} \neq (0,\ldots,0)$ for every $i \in \{1,\ldots,m\}$. Each row of T represents a linear template. Each template relates n variables. The *concretization* $\gamma : \overline{\mathbb{R}}^m \to 2^{\mathbb{R}^n}$ and the *abstraction* $\alpha : 2^{\mathbb{R}^n} \to \overline{\mathbb{R}}^m$ are defined as follows:

$$\gamma(d) := \{x \in \mathbb{R}^n \mid Tx \leq d\}\, \forall d \in \overline{\mathbb{R}}^m,\ \alpha(X) := \bigwedge\{d \in \overline{\mathbb{R}}^m \mid \gamma(d) \supseteq X\}\, \forall X \subseteq \mathbb{R}^n,$$

As shown by Sankaranarayanan et al. [19], α and γ form a Galois connection, i.e., for all $X \subseteq \mathbb{R}^n$ and all $d \in \overline{\mathbb{R}}^m$, $\alpha(X) \leq d$ iff $X \subseteq \gamma(d)$. Hence, $\alpha \circ \gamma$ is a downwards closure operator, and $\gamma \circ \alpha$ is an upwards closure operator[3]. This in particular implies that $\alpha \circ \gamma$ and $\gamma \circ \alpha$ are monotone. In order to simplify notations, we denote $\alpha \circ \gamma$ by $\mathbf{cl}$. The abstract elements from $\alpha(2^{\mathbb{R}^n}) = \mathbf{cl}(\overline{\mathbb{R}}^m)$ are called *closed*. The convex polyhedra from the set $\gamma(\overline{\mathbb{R}}^m) = \gamma(\alpha(2^{\mathbb{R}^n}))$ are called *template polyhedra*.

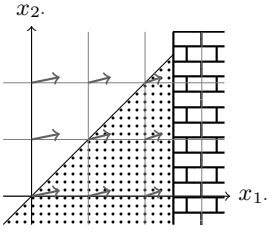


Fig. 1. Illustration for Example 1. The dotted region represents the convex polyhedron P. The wall represents the region that is not allowed, because of the staying condition I. The arrows illustrate the directions of the vector field V. Observe that any trajectory that starts in P and evolves according to V while satisfying I will stay in P.

Example 3. Let the template constraint matrix $T \in \mathbb{R}^{3\times 2}$ and $d \in \overline{\mathbb{R}}^3$ be defined by

$$T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 1 \end{pmatrix}, \text{ and } \qquad d = \begin{pmatrix} 2.5 \\ 2.5 \\ 0 \end{pmatrix}.$$

Then $\gamma(d) = P$, where P is defined in Example 1 (see Figure 1). □

The following properties of the operator $\mathbf{cl}$ will be crucial for the algorithms we present in this paper:

Lemma 1. *For all $i \in \{1,\ldots,m\}$ and all $d \in \overline{\mathbb{R}}^m$, we have:*

1. $\mathbf{cl}_{i\cdot}(d) = \sup\{T_{i\cdot}x \mid x \in \mathbb{R}^n \text{ and } Tx \leq d\}$
2. $\mathbf{cl}$ *is a point-wise minimum of finitely many monotone weak-affine mappings.*

[3] An operator $f : \mathbb{D} \to \mathbb{D}$ on a partially ordered set $\mathbb{D}$ is called *downwards (resp. upwards) closure operator* iff (1) f is monotone, (2) f is idempotent (i.e. $f^2 = f$), and (3) $f(x) \subseteq x$ (resp. $f(x) \supseteq x$) for all $x \in \mathbb{D}$.

Proof. For the first statement see Sankaranarayanan et al. [19]. In order to show that $\mathbf{cl}$ is a point-wise minimum of finitely many monotone weak-affine mappings, we use the strong duality theorem for linear programming as follows: $\mathbf{cl}_{i\cdot}(d) = \sup\{T_{i\cdot}x \mid x \in \mathbb{R}^n \text{ and } Tx \leq d\} = \inf\{d^\top y \mid y \in \mathbb{R}^m_{\geq 0},\ T^\top y = T_{i\cdot}^\top\}$ for all d with $\gamma(d) \neq \emptyset$. This gives us the statement. □

Invariants and Positive Invariants. Let $V : \mathbb{R}^n \to \mathbb{R}^n$ be a vector field and $I \subseteq \mathbb{R}^n$ a staying condition. A set $X \subseteq \mathbb{R}^n$ is called an *invariant* of (V, I) iff every trajectory that starts in X and evolves according to V while satisfying I stays in X. Before going further, we introduce the following notation: For all $d \in \overline{\mathbb{R}}^m$ and all $R \subseteq \{1, \ldots, m\}$, we define $d|_R \in \overline{\mathbb{R}}^m$ by

$$(d|_R)_{i\cdot} = \begin{cases} d_{i\cdot} & \text{if } i \in R \\ \infty & \text{if } i \notin R \end{cases} \qquad \text{for all } i \in \{1, \ldots, m\}.$$

Assume now that the affine vector field V is affine, and the staying condition I is a template polyhedron, i.e., $I \in \gamma(\overline{\mathbb{R}}^m)$. A template polyhedron $P \in \gamma(\overline{\mathbb{R}}^m)$ is called a *positive invariant* of (V, I) iff there exists some $R \subseteq \{1, \ldots, m\}$ such that the following statements hold:

1. $T_{i\cdot}V(x) \leq 0$ for all $x \in P$ with $T_{i\cdot}x = \alpha_{i\cdot}(P)$ and all $i \in R$ with $\alpha_{i\cdot}(P) < \alpha_{i\cdot}(I)$.
2. $P = \gamma(\alpha(P)|_R)$.

Our notion of positive invariants slightly differs from the notion of positive invariants of Sankaranarayanan et al. [20]. However, observe that every positive invariant in the sense of Sankaranarayanan et al. [20] is a positive invariant in our sense.

Every positive invariant is an invariant. However, there exist template polyhedra that are invariants without being positive invariants. Indeed due to the presence of staying conditions, for a template polyhedron P to be an invariant, the above condition $T_{i\cdot}V(x) \leq 0$ does not need to be satisfied at all the points $x \in P$ on the face corresponding to $T_{i\cdot}x = \alpha_{i\cdot}(P)$, when $\alpha_{i\cdot}(P) < \alpha_{i\cdot}(I)$. However, because of lack of space and additionally for clarity of presentation, we do not consider this in the present paper.

Example 4. We continue our running example (see Figure 1), i.e., the affine vector field V and the staying condition I are defined in Example 1, and the template constraint matrix T is defined in Example 3. The staying condition I is a template polyhedron, since $I = \gamma((2.5, \infty, \infty)^\top)$. The template polyhedron $P = \gamma(d)$ is an invariant as well as a positive invariant of (V, I). If we choose $R = \{1, 3\}$, then the requirements of the definition can be verified easily (cf. Figure 1). □

Our Goals: Time Elapse Operations and Abstract Semantics. We are interested in computing abstract semantics for affine hybrid automata w.r.t. template polyhedra. Performing the time elapse operation w.r.t. to template polyhedra is just the special case, where the affine hybrid automaton has no discrete transitions.

The *abstract semantics* for the affine hybrid automaton $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$ (w.r.t. the template polyhedra domain) is the point-wise minimal mapping $V^\sharp_\square$ that maps

every location $l \in \mathbf{L}$ to a template polyhedron $V^\sharp_\square[l] \in \gamma(\overline{\mathbb{R}}^m)$ and fulfills the following constraints: (1) $V^\sharp_\square[l_0] \supseteq \Theta$. (2) $V^\sharp_\square[l]$ is a positive invariant of $(\mathbf{D}(l), \mathbf{I}(l))$ for every location $l \in \mathbf{L}$. (3) $x' \in V^\sharp_\square[l']$ for all discrete transitions $(l, \Xi, l') \in \mathcal{T}$ and all $(x, x') \in \Xi$ with $x \in V^\sharp_\square[l]$. The existence of such a point-wise minimal mapping will be ensured by our findings. The abstract semantics safely over-approximates the concrete semantics.

In order to verify safety properties, a problem one is interested in is *abstract reachability*, which is the following decision problem: Decide whether or not, for a given template constraint matrix $T \in \mathbb{R}^{m \times n}$, a given affine hybrid automaton $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$, and a given location $l \in L$, the statement $V^\sharp[l] \neq \emptyset$ holds.

In this paper, we will adapt the max-strategy improvement algorithm of Gawlitza and Seidl [10, 11, 12, 14, 15] for computing $V^\sharp_\square$. We will find that abstract reachability is in $\mathsf{co{-}NP}$. Whether or not it is also in P is an open question. However, we at least know that it is a hard problem in the following sense: a polynomial-time algorithm for abstract reachability would give us a polynomial-time algorithm for computing the winning regions of mean-payoff games (see [12]). The latter problem is in $\mathsf{UP} \cap \mathsf{co{-}UP}$ (see Jurdzinski [16]) and it is a long outstanding and fundamental question whether or not it is in P.

The problem of performing the time elapse operation over template polyhedra is the following computational problem: Compute, for a given template constraint matrix T, a given affine vector field $V : \mathbb{R}^n \to \mathbb{R}^n$ and given template polyhedra Θ and I with $\Theta \subseteq I$, the least positive invariant of (V, I) which is a superset of Θ. We will show that the latter computational problem can be solved in polynomial time.

3 Our Approach: Getting into the Corset of the Monotone Framework

We aim at adapting the max-strategy improvement algorithms of Gawlitza and Seidl [10, 11, 12, 14, 15] in order to obtain an algorithm for computing abstract semantics. For that we have to formulate the problem as a problem of finding the least fixpoint of a self-map that is a maximum of finitely many monotone and concave self-maps (cf. Gawlitza and Seidl [10, 14]). The challenge is to get the time elapse operation into the corset of this monotone framework.

The Time Elapse Operation. Let $V : \mathbb{R}^n \to \mathbb{R}^n$ be an affine vector field. Firstly, we define the operator Δ^V on $\overline{\mathbb{R}}^m$ by

$$\Delta^V_{k\cdot}(d) := \sup \{T_{k\cdot} V(x) \mid x \in \mathbb{R}^n,\ Tx \leq d,\ T_{k\cdot} x \geq d_{k\cdot}\}$$

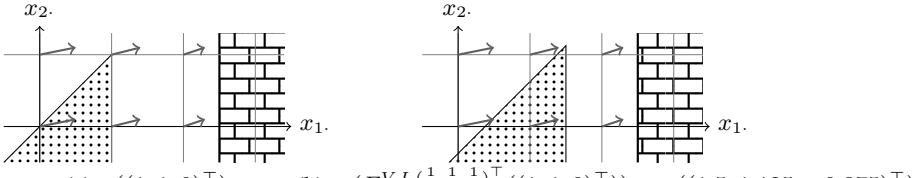
for all $k \in \{1, \ldots, m\}$ and all $d \in \overline{\mathbb{R}}^m$ with $d_{k\cdot} < \infty$. Note that $\Delta^V_{k\cdot}(d) = -\infty$, whenever $\{x \in \mathbb{R}^n \mid Tx \leq d, T_{k\cdot} x \geq d_{k\cdot}\} = \emptyset$. This is in particular fulfilled, if there exists some $i \in \{1, \ldots, m\}$ with $d_{i\cdot} = -\infty$. Moreover, we set $\Delta^V_{k\cdot}(d) := 0$ for all $k \in \{1, \ldots, m\}$ and $d \in \overline{\mathbb{R}}^m$ with $d_{k\cdot} = \infty$. Intuitively, $\Delta^V_{k\cdot}(d) > 0$ iff there exists some point x on the face $\mathcal{F} := \{x \in \mathbb{R}^n \mid Tx \leq d, T_{k\cdot} x \geq d_{k\cdot}\}$ such that $V(x)$ points to the outside.

For all $\epsilon \in \mathbb{R}^m_{>0}$, we define the operator $f^{V,\epsilon}$ on $\overline{\mathbb{R}}^m$ by

$$f^{V,\epsilon}(d) := d + \epsilon^\top \Delta^V(d) \qquad \text{for all } d \in \overline{\mathbb{R}}^m.$$

An application of the operator $f^{V,\epsilon}$ corrects the bounds to the templates according to the vector field V, ignoring the staying condition I. In order to take the staying condition I into account, we assume w.l.o.g. that I is a template polyhedron, i.e., $I \in \gamma(\overline{\mathbb{R}}^m)$. For all $\epsilon \in \mathbb{R}^m_{>0}$, we define the operator $F^{V,I,\epsilon}$ on $\overline{\mathbb{R}}^m$ as follows:

$$F^{V,I,\epsilon}(d) := f^{V,\epsilon}(d) \wedge \alpha(I) \qquad \text{for all } d \in \overline{\mathbb{R}}^m$$



(a) $\gamma((1,1,0)^\top)$ (b) $\gamma(F^{V,I,(\frac{1}{8},\frac{1}{8},\frac{1}{8})^\top}((1,1,0)^\top)) = \gamma((1.5, 1.125, -0.375)^\top)$

Fig. 2. The Running Example: An Application of $F^{V,I,\epsilon}$ for $\epsilon = (\frac{1}{8}, \frac{1}{8}, \frac{1}{8})^\top$

How the operator $F^{V,I,\epsilon}$ modifies a template polyhedron is shown in Figure 2 for our running example. Positive invariants can now be characterized as follows:

Lemma 2. *Let $\epsilon \in \mathbb{R}^m_{>0}$. For all $d \in \overline{\mathbb{R}}^m$ the following holds: The template polyhedron $\gamma(d)$ is a positive invariant of (V, I) iff $d \geq \mathbf{cl}(\mathbf{cl}(d) \vee F^{V,I,\epsilon}(\mathbf{cl}(d)))$.* □

In order to use the above lemma within a monotone framework, we have to ensure that $F^{V,I,\epsilon} \circ \mathbf{cl}$ is monotone. Then $\mathcal{F} := \mathbf{cl} \circ (\mathbf{cl} \vee F^{V,I,\epsilon} \circ \mathbf{cl})$ is monotone, too, and the fixpoint theorem of Knaster/Tarski [21] can be applied.[4] Observe that by construction $F^{V,I,\epsilon} \circ \mathbf{cl}$ is monotone, whenever $f^{V,\epsilon} \circ \mathbf{cl}$ is monotone. The operator $f^{V,\epsilon} \circ \mathbf{cl}$ is monotone on $\overline{\mathbb{R}}^m$, whenever the operator $f^{V,\epsilon}$ is monotone on $\mathbf{cl}(\overline{\mathbb{R}}^m)$.[5] If we choose ϵ small enough, then we enforce the monotonicity of $f^{V,\epsilon}$ on $\mathbf{cl}(\overline{\mathbb{R}}^m)$ and thus finally the monotonicity of $F^{V,I,\epsilon} \circ \mathbf{cl}$ and $\mathcal{F}$:

Lemma 3 (Monotonicity of $f^{V,\epsilon}$). *Assume $V(x) = Ax + b$ for all $x \in \mathbb{R}^n$. From A and T, we can compute an $\epsilon^{(0)} \in \mathbb{R}^m_{>0}$ in polynomial time such that $f^{V,\epsilon}$ is monotone on $\mathbf{cl}(\overline{\mathbb{R}}^m)$, whenever $\epsilon \leq \epsilon^{(0)}$. For every $\epsilon \leq \epsilon^{(0)}$, $f^{V,\epsilon} \circ \mathbf{cl}$ is a point-wise minimum of finitely many monotone weak-affine self-maps.* □

Because of the above lemma, we from now on assume that we have chosen an $\epsilon \in \mathbb{R}^m_{>0}$ such that $f^{V,\epsilon} \circ \mathbf{cl}$ and thus finally $\mathbf{cl} \circ (\mathbf{cl} \vee F^{V,I,\epsilon} \circ \mathbf{cl}) = \mathbf{cl} \circ (\mathrm{id} \vee F^{V,I,\epsilon}) \circ \mathbf{cl}$ is monotone. Therefore, for all sets $\Theta \subseteq \mathbb{R}^n$ of values, there exists a least positive invariant P of (V, I) which is a superset of Θ. It is given by

$$\gamma(\mu(\alpha(\Theta) \vee \mathbf{cl} \circ (\mathbf{cl} \vee F^{V,I,\epsilon} \circ \mathbf{cl})))$$

[4] For mappings $f, g : X \to \mathbb{D}$, $f \vee g$ denotes the mapping that is defined by $(f \vee g)(x) := f(x) \vee g(x)$ for all $x \in X$.

[5] It is not always possible to choose an ϵ such that $f^{V,\epsilon}$ is monotone on $\overline{\mathbb{R}}^m$.

However, we do not use this formulation. We want to have a simpler formulation that will allow us to perform the time elapse operation in polynomial time. For that we use the following fixpoint transfer lemma:

Lemma 4. *Let $\epsilon \in \mathbb{R}^m_{>0}$ be chosen such that $F^{V,I,\epsilon} \circ \mathbf{cl}$ is monotone. For all closed $\theta \in \mathbf{cl}(\overline{\mathbb{R}}^m)$, we have $\gamma(\mu(\theta \vee \mathbf{cl} \circ (\mathbf{cl} \vee F^{V,I,\epsilon} \circ \mathbf{cl}))) = \gamma(\mu(\theta \vee F^{V,I,\epsilon} \circ \mathbf{cl}))$.*[6] □

Putting everything together, we obtain our main result for the time elapse operation:

Theorem 1 (The Time Elapse Operation). *Let $V : \mathbb{R}^n \to \mathbb{R}^n$ be an affine vector field, and $\Theta, I \in \gamma(\overline{\mathbb{R}}^m)$ template polyhedra. Assume that $\epsilon \in \mathbb{R}^m_{>0}$ is chosen such that $F^{V,I,\epsilon} \circ \mathbf{cl}$ is monotone. The template polyhedron $\gamma(\mu(\alpha(\Theta) \vee F^{V,I,\epsilon} \circ \mathbf{cl}))$ is the least positive invariant of (V, I) which is a superset of Θ.*

Proof. The existence of ϵ is ensured by Lemma 3. The existence of the least fixpoint is ensured by the fixpoint theorem of Knaster/Tarski. Lemmata 2 gives us that $P := \gamma(\mu(\alpha(\Theta) \vee \mathbf{cl} \circ (\mathbf{cl} \vee F^{V,I,\epsilon} \circ \mathbf{cl})))$ is the least positive invariant of (V, I) which is a superset of Θ. Lemma 4 finally gives us $P = \gamma(\mu(\alpha(\Theta) \vee F^{V,I,\epsilon} \circ \mathbf{cl}))$. □

Abstract Semantics. So far, we have ignored the discrete transitions. In order to take them into account, we define an abstract semantics for discrete transitions $(l, \Xi, l') \in \mathcal{T}$. Recall that the assertion $\Xi \subseteq \mathbb{R}^{2n}$ is a convex polyhedron. In the following we will always assume that the convex polyhedron Ξ is given by a matrix $A_\Xi \in \mathbb{R}^{l \times 2n}$ and a vector $b_\Xi \in \mathbb{R}^l$ such that $\Xi = \{x \in \mathbb{R}^{2n} \mid A_\Xi x \leq b_\Xi\}$. The collecting semantics $[\![\Xi]\!]$ of Ξ is defined by $[\![\Xi]\!](X) := \{y \in \mathbb{R}^n \mid \exists x \in X \,.\, (x, y) \in \Xi\}$ for all $X \subseteq \mathbb{R}^n$. The abstract semantics $[\![\Xi]\!]^\sharp$ of Ξ is defined by $[\![\Xi]\!]^\sharp := \alpha \circ [\![\Xi]\!] \circ \gamma$. The abstract semantics safely over-approximates the collecting semantics and the concrete semantics. For all $k \in \{1, \ldots, m\}$ and all $d \in \overline{\mathbb{R}}^m$, we have:

$$[\![\Xi]\!]^\sharp_k(d) := \sup \left\{ T_{k\cdot} y \;\middle|\; x \in \gamma(d),\ \begin{pmatrix} x \\ y \end{pmatrix} \in \Xi \right\} \tag{1}$$

$$= \sup \left\{ T_{k\cdot} y \;\middle|\; \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^{2n},\ Tx \leq d,\ A_\Xi \begin{pmatrix} x \\ y \end{pmatrix} \leq b_\Xi \right\} \tag{2}$$

If we consider the dual of the above linear programming problem, we get that also the operator $[\![\Xi]\!]^\sharp$ on $\overline{\mathbb{R}}^m$ has nice properties (cf. Gawlitza and Seidl [10, 14]):

Lemma 5 (The Abstract Semantics $[\![\Xi]\!]^\sharp$). *The following holds for every convex polyhedron $\Xi \subseteq \mathbb{R}^{2n}$: (1) $[\![\Xi]\!]^\sharp = [\![\Xi]\!]^\sharp \circ \mathbf{cl} = \mathbf{cl} \circ [\![\Xi]\!]^\sharp$. (2) $[\![\Xi]\!]^\sharp$ is the point-wise minimum of finitely many monotone weak-affine operators.* □

We are now going to define an abstract semantics $V^\sharp$ for an *affine* hybrid automata $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$ that corresponds to the abstract semantics $V^\sharp_\square$ of Ψ (cf. Section 2). W.o.l.g. we assume that the initial condition Θ, the location invariants $\mathbf{I}(l)$, $l \in \mathbf{L}$, and the transition relations are all template polyhedra. The abstract semantics $V^\sharp$ of Ψ is the least solution of the following constraint system:

$$\mathbf{V}^\sharp[l_0] \geq \alpha(\Theta) \tag{3}$$

[6] Here, θ denotes the function that returns θ for every argument.

$$\mathbf{V}^\sharp[l] \geq F^{\mathbf{D}(l),\mathbf{I}(l),\epsilon(l)}(\mathbf{cl}(\mathbf{V}^\sharp[l])) \qquad \text{for all } l \in \mathbf{L} \tag{4}$$

$$\mathbf{V}^\sharp[l'] \geq [\![\Xi]\!]^\sharp(\mathbf{V}^\sharp[l]) \qquad \text{for all } (l, \Xi, l') \in \mathcal{T} \tag{5}$$

The variables $\mathbf{V}^\sharp[l]$, $l \in \mathbf{L}$ take values from $\overline{\mathbb{R}}^m$. The existence of the least solution is ensured by the fixpoint theorem of Knaster/Tarski, since we assume that, for all locations $l \in \mathbf{L}$, $\epsilon(l) \in \mathbb{R}^m_{>0}$ is chosen such that $F^{\mathbf{D}(l),\mathbf{I}(l),\epsilon(l)}$ is monotone. The existence of such an $\epsilon(l)$ is again ensured by Lemma 3.

Constraint (3) takes all possible initial values of the continuous variables at the initial location l_0 into account. Constraint (4) ensures that the template polyhedron $\gamma(V^\sharp[l])$ is a positive invariant of $(\mathbf{D}(l), \mathbf{I}(l))$ (cf. Lemma 2). Constraint (5) ensures that the template polyhedron $\gamma(V^\sharp[l'])$ contains at least all values that can come through the discrete transition (l, Ξ, l'). By construction, we have:

Theorem 2. *$V^\sharp_\square[l] = \gamma(V^\sharp[l])$ for all locations $l \in \mathbf{L}$.* □

Because of the above theorem, we should now aim at computing $V^\sharp$.

4 Adapting the Max-Strategy Approach

Notations. In this section, we consider systems $\mathcal{C}$ of inequalities of the form $\mathbf{x} \geq e$ (resp. $\mathbf{x} \leq e$), where $\mathbf{x}$ is a variable that takes values form $\overline{\mathbb{R}}$ and e is an expression over $\overline{\mathbb{R}}$. The set of variables of $\mathcal{C}$ is denoted by $\mathbf{X}_\mathcal{C}$, where we omit the subscript, whenever it is clear from the context. The semantics $[\![e]\!] : (\mathbf{X} \to \overline{\mathbb{R}}) \to \overline{\mathbb{R}}$ of an expression e is defined by $[\![\mathbf{x}]\!](\rho) := \rho(\mathbf{x})$ and $[\![f(e_1, \ldots, e_k)]\!](\rho) := f([\![e_1]\!](\rho), \ldots, [\![e_k]\!](\rho))$, where $\mathbf{x} \in \mathbf{X}$, f is a k-ary operator on $\overline{\mathbb{R}}$, $e_1, \ldots, e_k$ are expressions, and $\rho : \mathbf{X} \to \overline{\mathbb{R}}$ is a variable assignment.

For a system $\mathcal{C}$ of constraints of the form $\mathbf{x} \geq e$ (resp. $\mathbf{x} \leq e$), we define the operator $[\![\mathcal{C}]\!] : (\mathbf{X} \to \overline{\mathbb{R}}) \to \mathbf{X} \to \overline{\mathbb{R}}$ by

$$[\![\mathcal{C}]\!](\rho)(\mathbf{x}) := \bigvee \{[\![e]\!]\rho \mid \mathbf{x} \geq e \text{ belongs to } \mathcal{C}\}$$

(resp. $[\![\mathcal{C}]\!](\rho)(\mathbf{x}) := \bigwedge \{[\![e]\!]\rho \mid \mathbf{x} \geq e \text{ belongs to } \mathcal{C}\}$) for all variable assignments $\rho : \mathbf{X} \to \overline{\mathbb{R}}$ and all variables $\mathbf{x} \in \mathbf{X}$. Hence, ρ is a solution of $\mathcal{C}$ iff $\rho \geq [\![\mathcal{C}]\!](\rho)$ (resp. $\rho \leq [\![\mathcal{C}]\!](\rho)$). The least (resp. the greatest) solution of $\mathcal{C}$ is $\mu[\![\mathcal{C}]\!]$ (resp. $\nu[\![\mathcal{C}]\!]$). For a system $\mathcal{C}$ of inequalities of the form $\mathbf{x} \geq e$ and a pre-fixpoint ρ of the operator $[\![\mathcal{C}]\!]$ (i.e., $\rho \leq [\![\mathcal{C}]\!](\rho)$), $\mu_{\geq\rho}[\![\mathcal{C}]\!]$ denotes the least solution of $\mathcal{C}$ that is greater than or equal to ρ.

Rewriting the Abstract Semantic In-Equations. We now rewrite the abstract semantic in-equations (3) - (5) into a system $\mathcal{C}(\Psi)$ of in-equations of the form $\mathbf{x} \geq f(\mathbf{x}_1, \ldots, \mathbf{x}_k)$, where the variables take values from $\overline{\mathbb{R}}$ and the operator f is a maximum of finitely many monotone weak-affine operators. The set $\mathbf{X}$ of variables of the system $\mathcal{C}(\Psi)$ of in-equations we are going to construct is $\mathbf{X} = \{\mathbf{d}_{l,i} \mid l \in \mathbf{L} \text{ and } i \in \{1, \ldots, m\}\}$. Corresponding to constraint (3) we add the following in-equations:

$$\mathbf{d}_{l_0,i} \geq \alpha_i.(\Theta) \qquad \text{for all } i \in \{1, \ldots, m\} \tag{6}$$

Corresponding to constraint (4), for every location $l \in \mathbf{L}$, we add the following in-equations that will deal with the time elapse operation:

$$\mathbf{d}_{l,i} \geq F_{i\cdot}^{\mathbf{D}(l),\mathbf{I}(l),\epsilon(l)}(\mathbf{cl}((\mathbf{d}_{l,1},\ldots,\mathbf{d}_{l,m})^\top)) \qquad \text{for all } i \in \{1,\ldots,m\} \tag{7}$$

Corresponding to constraint (5), for every discrete transition $(l, \Xi, l') \in \mathcal{T}$, we add the following in-equations that will deal with the discrete transition (l, Ξ, l'):

$$\mathbf{d}_{l',i} \geq [\![\Xi]\!]^\sharp_{i\cdot}\left((\mathbf{d}_{l,1},\ldots,\mathbf{d}_{l,m})^\top\right) \qquad \text{for all } i \in \{1,\ldots,m\} \tag{8}$$

By construction we have:

Lemma 6. *Let $\rho^* : \mathbf{X} \to \overline{\mathbb{R}}$ be the least solution of $\mathcal{C}(\Psi)$. Then, for all locations $l \in \mathbf{L}$ and all $i \in \{1,\ldots,m\}$, we have $(V^\sharp[l])_{i\cdot} = \rho^*(\mathbf{d}_{l,i})$.* □

Example 5. We continue our running example, i.e., we aim at computing the abstract semantics $V^\sharp$ of the hybrid automaton Ψ from Example 2, where we use the template constraint matrix T introduced in Example 3. For that we choose $\epsilon(1) = (1,\ldots,1)^\top$. Then $f^{\mathbf{D}(1),\epsilon(1)}$ and thus $F^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}$ are monotone. The system $\mathcal{C}(\Psi)$ consists of the following in-equations:

$$\begin{aligned}
&\mathbf{d}_{1,1} \geq 1 && \mathbf{d}_{1,1} \geq F_{1\cdot}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1},\mathbf{d}_{1,2},\mathbf{d}_{1,3})^\top))\\
&\mathbf{d}_{1,2} \geq 1 && \mathbf{d}_{1,2} \geq F_{2\cdot}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1},\mathbf{d}_{1,2},\mathbf{d}_{1,3})^\top))\\
&\mathbf{d}_{1,3} \geq 0 && \mathbf{d}_{1,3} \geq F_{3\cdot}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1},\mathbf{d}_{1,2},\mathbf{d}_{1,3})^\top))
\end{aligned}$$

Example 6 shows how we can compute the least solution of this constraint system. □

The Max-Strategy Improvement Algorithm. Let $\mathcal{C}$ be a system of inequalities of the form $\mathbf{x} \geq e$, where $[\![e]\!]$ is a point-wise minimum of finitely many monotone weak-affine operators. We aim at computing the least solution $\mu[\![\mathcal{C}]\!]$ of $\mathcal{C}$ that exists due to monotonicity.

A subset σ of $\mathcal{C}$ is called a max-strategy of $\mathcal{C}$ iff it contains exactly one constraint $\mathbf{x} \geq e$ for every variable $\mathbf{x}$ occurring in $\mathcal{C}$. For simplicity, we assume that in $\mathcal{C}$ there exists a constraint $\mathbf{x} \geq -\infty$ for every variable $\mathbf{x}$ occurring in $\mathcal{C}$. Then $\{\mathbf{x} \geq -\infty \mid \mathbf{x} \in \mathbf{X}\}$ is a max-strategy. This will be the max-strategy the algorithm starts with.

The max-strategy improvement algorithm maintains a current max-strategy σ and a current approximate $\rho : \mathbf{X} \to \overline{\mathbb{R}}$ to the least solution $\mu[\![\mathcal{C}]\!]$ of $\mathcal{C}$. The max-strategy algorithm can be written as follows:

Algorithm 1. The Max-Strategy Improvement Algorithm

$\sigma \leftarrow \{\mathbf{x} \geq -\infty \mid \mathbf{x} \in \mathbf{X}\}$; $\rho \leftarrow \{\mathbf{x} \mapsto -\infty \mid \mathbf{x} \in \mathbf{X}\}$;

while (ρ is not a solution of $\mathcal{C}$) {$\sigma \leftarrow$ improvement of σ w.r.t. ρ; $\rho \leftarrow \mu_{\geq\rho}[\![\sigma]\!]$; }

return ρ;

We have to define the term *improvement*. Let σ be a max-strategy of $\mathcal{C}$ and ρ be a pre-solution of $[\![\sigma]\!]$, i.e., $\rho \leq [\![\sigma]\!]\rho$. A max-strategy σ' of $\mathcal{C}$ is called an *improvement of σ w.r.t. ρ* iff the following conditions hold:

1. $[\![\sigma']\!]\rho \geq [\![\sigma]\!]\rho$.
2. If $\mathbf{x} \geq e$ belongs to σ and $\mathbf{x} \geq e'$ belongs to σ' with $e \neq e'$, then $[\![e']\!]\rho > [\![e]\!]\rho$.

The second condition ensured that a max-strategy is only changed at variables where we have a strict improvement. This is important for the correctness of the algorithm (cf. Gawlitza and Seidl [10, 11, 12, 14, 15]).

It is obvious that the algorithm returns the least solution of $\mathcal{C}$, whenever it terminates. From the considerations in the next subsection, it will follow that it terminates at the latest after considering every max-strategy at most once. In the next subsection, we will also explain how we can compute $\mu_{\geq\rho}[\![\sigma]\!]$ for a max-strategy σ and a variable assignment ρ that occurs during the run of the algorithm. Before doing so, we will use our algorithm for computing the abstract semantics of our running example.

Example 6. We apply the max-strategy improvement algorithm to the system $\mathcal{C}$ of constraints defined in Example 5. After the first max-strategy improvement step we may get the max-strategy σ_1 that consists of the following constraints:

$$\mathbf{d}_{1,1} \geq 1 \qquad \mathbf{d}_{1,2} \geq 1 \qquad \mathbf{d}_{1,3} \geq 0$$

We have to find the least solution ρ_1 of σ_1 that is greater than of equal to $\rho_0 = \{\mathbf{x} \mapsto -\infty \mid \mathbf{x} \in \mathbf{X}\}$. Hence, obviously $\rho_1 = \mu_{\geq\rho_0}[\![\sigma_1]\!] = \{\mathbf{d}_{1,1} \mapsto 1, \mathbf{d}_{1,2} \mapsto 1, \mathbf{d}_{1,3} \mapsto 0\}$. However, ρ_1 is not a solution of $\mathcal{C}$. Hence, we can improve the current max-strategy σ_1 w.r.t. ρ_1. We may obtain the max-strategy σ_2 that consists of the following constraints:

$$\mathbf{d}_{1,1} \geq F_{1.}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1}, \mathbf{d}_{1,2}, \mathbf{d}_{1,3})^\top))$$
$$\mathbf{d}_{1,2} \geq F_{2.}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1}, \mathbf{d}_{1,2}, \mathbf{d}_{1,3})^\top)) \qquad \mathbf{d}_{1,3} \geq 0$$

We get $\rho_2 = \mu_{\geq\rho_1}[\![\sigma_1]\!] = \{\mathbf{d}_{1,1} \mapsto 2.5, \mathbf{d}_{1,2} \mapsto 3.5, \mathbf{d}_{1,3} \mapsto 0\}$. How we can compute ρ_2 will be explained in Example 7. ρ_2 solves the constraint system $\mathcal{C}$. Hence, the algorithm terminates and returns ρ_2, which is the correct least solution of $\mathcal{C}$. Thus, we have $V^\sharp[1] = (2.5, 3.5, 0)^\top$. By Theorem 2, we get $V_\square^\sharp[1] = \gamma((2.5, 3.5, 0)^\top) = \{(x_1, x_2)^\top \in \mathbb{R}^2 \mid x_1 \leq 2.5, x_2 \leq 2.5, x_2 \leq x_1\}$ (cf. Figure 1). □

In the above example, we have 3 inequality constraints for each variable after introducing the constraints $\mathbf{d}_{1,i} \geq -\infty$, $i = \{1, 2, 3\}$. Hence, we have $3^3 = 9$ max-strategies. However, since the sequence of approximates is strictly increasing until it stabilizes, the constraint $\mathbf{d}_{1,1} \geq -\infty$ will not be considered after considering the constraint $\mathbf{d}_{1,1} \geq 1$. Similar, the constraint $\mathbf{d}_{1,1} \geq 1$ will not be considered after considering the constraint $\mathbf{d}_{1,1} \geq F_{1.}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1}, \mathbf{d}_{1,2}, \mathbf{d}_{1,3})^\top))$. Hence, the maximal number of max-strategies considered by our max-strategy improvement algorithm is $1 + 2 \cdot 3 = 7$.

This is not by accident. Whenever the affine hybrid system has exactly one location and no discrete transitions, the number of max-strategies the algorithm considers is at most $1 + 2m$. If we start the algorithm with the max-strategy that corresponds to the set Θ of all possible initial values, then we can reduce this number to $1 + m$. Thus, we have:

Lemma 7. *If we apply our max-strategy improvement algorithm for performing the time elapse operation, then the number of max-strategy improvement steps is bounded by m, where m is number of templates.* □

Evaluating a Single Max-Strategy. Let σ be a max-strategy for $\mathcal{C}(\Psi)$ and ρ be a variable assignment that occurs during a run of the max-strategy improvement algorithm (the constraint system $\mathcal{C}(\Psi)$ is defined in Subsection 4). We are aiming at computing $\mu_{\geq\rho}[\![\sigma]\!]$. For that, we firstly remove all constraints $\mathbf{x} \geq -\infty$ from σ and replace the corresponding variables with the constant $-\infty$. For simplicity, we denote the resulting system again by σ. Since the algorithm only improves max-strategies at positions where there are strict improvements, we have $\mu_{\geq\rho}[\![\sigma]\!](\mathbf{x}) > -\infty$ for all variables $\mathbf{x} \in \mathbf{X}$.

From the results of Gawlitza and Seidl [10, 14] it follows that $\mu_{\geq\rho}[\![\sigma]\!]$ equals the variable assignment $\rho^\sigma : \mathbf{X} \to \overline{\mathbb{R}}$ which is defined as follows:

$$\rho^\sigma(\mathbf{z}) := \sup\{\rho(\mathbf{z}) \mid \rho : \mathbf{X} \to \mathbb{R},\ \rho(\mathbf{x}) \leq [\![e]\!](\rho) \text{ for all constraints } \mathbf{x} \geq e \text{ of } \sigma\} \quad (9)$$

for all $\mathbf{z} \in \mathbf{X}$. Observe that the variable assignment $\rho^\sigma = \mu_{\geq\rho}(\sigma)$ only depends on the max-strategy σ and not on the variable assignment ρ. This is an important observation. Since the max-strategy improvement algorithm generates a strictly increasing sequence of variable assignments, each of which only depends on the current max-strategy, it follows that the max-strategy algorithm terminates at the latest after considering each max-strategy at most once.

In order to compute ρ^σ, we set up the system σ' of linear inequalities as follows: We start with an empty system of linear inequalities. For every inequality of the form $\mathbf{d} \geq c$, where $c \in \mathbb{R} \cup \{\infty\}$, we add the linear constraint $\mathbf{d} \leq c$. For every bunch of inequalities of the form

$$\mathbf{d}_1 \geq F^{V,I,\epsilon}_{1\cdot}(\mathbf{cl}((\mathbf{d}_1,\dots,\mathbf{d}_m)^\top)) \quad \cdots \quad \mathbf{d}_m \geq F^{V,I,\epsilon}_{m\cdot}(\mathbf{cl}((\mathbf{d}_1,\dots,\mathbf{d}_m)^\top)),$$

where $V(x) = Ax + b$ for all $x \in \mathbb{R}^n$ and $I \in \gamma(\overline{\mathbb{R}}^m)$, we add (according to Lemma 1 and the definition of $F^{V,I,\epsilon}$) the following linear constraints

$$\begin{aligned}
\mathbf{d}_i &\leq \mathbf{d}'_i + \epsilon_i . T_i . (A(\mathbf{x}_{i,1},\dots,\mathbf{x}_{i,n})^\top + b) \wedge \alpha_k . (I) && \forall i \in \{1,\dots,m\}\\
T_i . (\mathbf{x}_{i,1},\dots,\mathbf{x}_{i,n}) &\geq \mathbf{d}'_i && \forall i \in \{1,\dots,m\}\\
T_j . (\mathbf{x}_{i,1},\dots,\mathbf{x}_{i,n}) &\leq \mathbf{d}'_j && \forall i,j \in \{1,\dots,m\}\\
\mathbf{d}'_i &\leq T_i . (\mathbf{x}'_{i,1},\dots,\mathbf{x}'_{i,m})^\top && \forall i \in \{1,\dots,m\}\\
T_j . (\mathbf{x}'_{i,1},\dots,\mathbf{x}'_{i,n}) &\leq \mathbf{d}_j && \forall i,j \in \{1,\dots,m\}
\end{aligned}$$

where $\mathbf{d}'_i$, $\mathbf{x}_{i,j}$, $\mathbf{x}'_{i,j}$ are fresh variables for all $i,j \in \{1,\dots,m\}$.

For every inequality constraint $\mathbf{d} \leq [\![\Xi]\!]^\sharp_{k\cdot}(\mathbf{d}_1,\dots,\mathbf{d}_m)$, where $\Xi = \{(x,y)^\top \in \mathbb{R}^{2n} \mid Ax \leq b\}$ with $A \in \mathbb{R}^{l\times 2n}$ and $b \in \mathbb{R}^l$, we add (according to Equation (2)) the following linear constraints:

$$\begin{aligned}
\mathbf{d} \leq T_{k\cdot}(\mathbf{y}_1,\dots,\mathbf{y}_n)^\top \qquad & T_i . (\mathbf{x}_1,\dots,\mathbf{x}_n)^\top \leq \mathbf{d}_i \quad \forall i \in \{1,\dots,m\}\\
& A_i . (\mathbf{x}_1,\dots,\mathbf{x}_n,\mathbf{y}_1,\dots,\mathbf{y}_n)^\top \leq b_i . \quad \forall i \in \{1,\dots,l\}
\end{aligned}$$

Here, $\mathbf{x}_1,\dots,\mathbf{x}_n$ and $\mathbf{y}_1,\dots,\mathbf{y}_n$ are fresh variables. Finally, we get

$$\rho^\sigma(\mathbf{z}) = \sup\{\rho(\mathbf{z}) \mid \rho : \mathbf{X} \to \mathbb{R} \text{ and } \rho(\mathbf{x}) \leq [\![e]\!]\rho \text{ for all constraints } \mathbf{x} \leq e \text{ of } \sigma'\}$$

for all $\mathbf{z} \in \mathbf{X}$. Hence, for all $\mathbf{z} \in \mathbf{X}$, $\rho^\sigma(\mathbf{z})$ can be computed by solving a linear programming problem that can be constructed in polynomial time:

Lemma 8. *For every max-strategy σ of $\mathcal{C}(\Psi)$, the variable assignment ρ^σ defined by Equation* (9) *can be computed in polynomial time through linear programming. Whenever the max-strategy algorithm must compute $\mu_{\geq\rho}[\![\sigma]\!]$, we have $\mu_{\geq\rho}[\![\sigma]\!] = \rho^\sigma$.* □

Example 7. We consider the max-strategy σ_2 from Example 6, i.e., we aim at computing $\mu_{\geq\rho_1}[\![\sigma_2]\!]$ which equals ρ^{σ_2} as defined in Equation (9). Therefore, for all $i \in \{1,\dots,m\}$, we aim at solving the following optimization problem:

$$\begin{aligned} \sup\ \mathbf{d}_{1,i} \qquad & \mathbf{d}_{1,1} \leq F_{1.}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1},\mathbf{d}_{1,2},\mathbf{d}_{1,3})^\top)) \\ & \mathbf{d}_{1,2} \leq F_{2.}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}(\mathbf{cl}((\mathbf{d}_{1,1},\mathbf{d}_{1,2},\mathbf{d}_{1,3})^\top)) \qquad \mathbf{d}_{1,3} \leq 0 \end{aligned}$$

We are now going to simplify the constraints that define the feasible space. By unfolding the definition of $\mathbf{cl}$ and the definition of $F_{i.}^{\mathbf{D}(1),\mathbf{I}(1),\epsilon(1)}$ for all $i \in \{1,2\}$, we obtain:

$$\begin{aligned} &\mathbf{d}_{1,1} \leq \mathbf{d}'_{1,1} + 5 + \sup\{-x_1 \mid x_1 = \mathbf{d}'_{1,1}, x_2 \leq \mathbf{d}'_{1,2}, x_2 - x_1 \leq \mathbf{d}'_{1,3}\} \\ &\mathbf{d}_{1,1} \leq 2.5 \\ &\mathbf{d}_{1,2} \leq \mathbf{d}'_{1,2} + 1 + \sup\{0 \mid x_1 \leq \mathbf{d}'_{1,1}, x_2 = \mathbf{d}'_{1,2}, x_2 - x_1 \leq \mathbf{d}'_{1,3}\} \\ &\mathbf{d}_{1,3} \leq 0 \\ &\mathbf{d}'_{1,1} \leq \sup\{x_1 \mid x_1 \leq \mathbf{d}_{1,1}, x_2 \leq \mathbf{d}_{1,2}, x_2 - x_1 \leq \mathbf{d}'_{1,3}\} \\ &\mathbf{d}'_{1,2} \leq \sup\{x_2 \mid x_1 \leq \mathbf{d}_{1,1}, x_2 \leq \mathbf{d}_{1,2}, x_2 - x_1 \leq \mathbf{d}'_{1,3}\} \\ &\mathbf{d}'_{1,3} \leq \sup\{x_2 - x_1 \mid x_1 \leq \mathbf{d}_{1,1}, x_2 \leq \mathbf{d}_{1,2}, x_2 - x_1 \leq \mathbf{d}'_{1,3}\} \end{aligned}$$

After eliminating the supremums in the right-hand sides, we get:

$$\begin{array}{llll} \mathbf{d}_{1,1} \leq \mathbf{d}'_{1,1} + 5 - \mathbf{x}_{1,1,1} & \mathbf{x}_{1,1,1} = \mathbf{d}'_{1,1} & \mathbf{x}_{1,1,2} \leq \mathbf{d}'_{1,2} & \mathbf{x}_{1,1,2} - \mathbf{x}_{1,1,1} \leq \mathbf{d}'_{1,3} \\ \mathbf{d}_{1,1} \leq 2.5 & & & \\ \mathbf{d}_{1,2} \leq \mathbf{d}'_{1,2} + 1 & \mathbf{x}_{1,2,1} \leq \mathbf{d}'_{1,1} & \mathbf{x}_{1,2,2} = \mathbf{d}'_{1,2} & \mathbf{x}_{1,2,2} - \mathbf{x}_{1,2,1} \leq \mathbf{d}'_{1,3} \\ \mathbf{d}_{1,3} \leq 0 & & & \\ \mathbf{d}'_{1,1} \leq \mathbf{x}'_{1,1,1} & \mathbf{x}'_{1,1,1} \leq \mathbf{d}_{1,1} & \mathbf{x}'_{1,1,2} \leq \mathbf{d}_{1,2} & \mathbf{x}'_{1,1,2} - \mathbf{x}'_{1,1,1} \leq \mathbf{d}_{1,3} \\ \mathbf{d}'_{1,2} \leq \mathbf{x}'_{1,2,2} & \mathbf{x}'_{1,2,1} \leq \mathbf{d}_{1,1} & \mathbf{x}'_{1,2,2} \leq \mathbf{d}_{1,2} & \mathbf{x}'_{1,2,2} - \mathbf{x}'_{1,2,1} \leq \mathbf{d}_{1,3} \\ \mathbf{d}'_{1,3} \leq \mathbf{x}'_{1,3,2} - \mathbf{x}'_{1,3,1} & \mathbf{x}'_{1,3,1} \leq \mathbf{d}_{1,1} & \mathbf{x}'_{1,3,2} \leq \mathbf{d}_{1,2} & \mathbf{x}'_{1,3,2} - \mathbf{x}'_{1,3,1} \leq \mathbf{d}_{1,3} \end{array}$$

When we solve the corresponding linear programming problems that aim at maximizing $\mathbf{d}_{1,1}$, $\mathbf{d}_{1,2}$, and $\mathbf{d}_{1,3}$, respectively, we get $\mathbf{d}_{1,1} = 2.5$, $\mathbf{d}_{1,2} = 3.5$, and $\mathbf{d}_{1,3} = 0$. Observe that, instead of solving three linear programming problems, we could solve just one, where we aim at maximizing the sum $\mathbf{d}_{1,1} + \mathbf{d}_{1,2} + \mathbf{d}_{1,3}$. Any optimal solution then gives us the values for $\mathbf{d}_{1,1}$, $\mathbf{d}_{1,2}$, and $\mathbf{d}_{1,3}$. □

The Final Results. Putting everything together, we finally get:

Theorem 3. *The abstract semantics of an affine hybrid automaton $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$ can be computed through the presented max-strategy improvement algorithm. This algorithm performs at most exponentially many strategy improvement steps, each*

of which can be performed in polynomial time through linear programming. The number of max-strategy improvement steps is bounded by

$$m \cdot |\mathbf{L}| \cdot \prod_{l \in \mathbf{L}} \max\{1, |\{(l_1, \rho, l_2) \in \mathcal{T} \mid l_2 = l\}|^m\},$$

where m denotes the number of linear templates. The time elapse operation for affine hybrid automata w.r.t. template polyhedra can be performed in polynomial time. □

Theorem 4. *Abstract reachability for affine hybrid systems w.r.t. template polyhedra is in* co−NP.

Proof. Let $\Psi = (n, \mathbf{L}, \mathcal{T}, \Theta, \mathbf{D}, \mathbf{I}, l_0)$ be an affine hybrid system, $l \in \mathbf{L}$, and $T \in \mathbb{R}^{m \times n}$ a template constraint matrix. We have to provide an non-deterministic algorithm that has an accepting run iff $\mu[\![\mathcal{C}(\Psi)]\!](\mathbf{d}_{l,i}) = -\infty$ for all $i \in \{1, \ldots, m\}$, i.e., l is unreachable.

The algorithm firstly chooses a max-strategy σ for $\mathcal{C}(\Psi)$ non-deterministically. Note that there exists a max-strategy σ' for $\mathcal{C}(\Psi)$ such that $\rho^{\sigma'} = \mu[\![\mathcal{C}(\Psi)]\!]$. According to Lemma 8, we then compute ρ^σ as defined by Equation (9) in polynomial time. If ρ^σ solves $\mathcal{C}(\Psi)$ (this can be checked in polynomial time), then we know that $\rho^\sigma \geq \mu[\![\mathcal{C}(\Psi)]\!]$. Hence, the algorithm accepts iff $\rho^\sigma(\mathbf{d}_{l,i}) = -\infty$ for all $i \in \{1, \ldots, m\}$. □

5 Conclusion

In this paper we showed how the max-strategy improvement algorithm of Gawlitza and Seidl [10, 11, 12, 14, 15] can be utilized to compute abstract semantics of affine hybrid automata w.r.t. template polyhedra — an problem that can be used for *unbounded time verification*. This gives us a polynomial-time algorithm for the time elapse operation over template polyhedra. Moreover, we showed that the corresponding abstract reachability problem is in co−NP. For future work, it would be interesting to see in how far this approach can be generalized to non-linear templates and non-linear dynamics (cf. Gawlitza and Seidl [14]). It also remains to evaluate the proposed approach in practice. We report on our proof-of-concept implementation in the corresponding technical report Dang and Gawlitza [8].

References

[1] Adjé, A., Gaubert, S., Goubault, E.: Computing the smallest fixed point of nonexpansive mappings arising in game theory and static analysis of programs. In: MTNS (2008)
[2] Adjé, A., Gaubert, S., Goubault, E.: Coupling Policy Iteration with Semi-Definite Relaxation to Compute Accurate Numerical Invariants in Static Analysis. In: Gordon, A.D. (ed.) ESOP 2010. LNCS, vol. 6012, pp. 23–42. Springer, Heidelberg (2010)
[3] Alur, R., Dang, T., Ivancic, F.: Counter-example guided predicate abstraction of hybrid systems. Theoretical Computer Science (TCS) 354(2), 250–271 (2006)
[4] Asarin, E., Dang, T., Girard, A.: Hybridization methods for the analysis of nonlinear systems. Acta Inf. 43(7), 451–476 (2007)
[5] Clarke, E.M., Fehnker, A., Han, Z., Krogh, B.H., Ouaknine, J., Stursberg, O., Theobald, M.: Abstraction and counterexample-guided refinement in model checking of hybrid systems. Int. J. Found. Comput. Sci. 14(4), 583–604 (2003)

[6] Costan, A., Gaubert, S., Goubault, É., Martel, M., Putot, S.: A Policy Iteration Algorithm for Computing Fixed Points in Static Analysis of Programs. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 462–475. Springer, Heidelberg (2005)
[7] Cousot, P., Cousot, R.: Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: POPL (1977)
[8] Dang, T., Gawlitza, T.M.: Template-based unbounded time verification of affine hybrid automata. Technical report, VERIMAG, Grenoble, France (2011)
[9] Gaubert, S., Goubault, E., Taly, A., Zennou, S.: Static analysis by policy iteration on relational domains. In: Nicola [17]
[10] Gawlitza, T., Seidl, H.: Precise Relational Invariants through Strategy Iteration. In: Duparc, J., Henzinger, T.A. (eds.) CSL 2007. LNCS, vol. 4646, pp. 23–40. Springer, Heidelberg (2007)
[11] Gawlitza, T., Seidl, H.: Precise fixpoint computation through strategy iteration. In: Nicola [17]
[12] Gawlitza, T., Seidl, H.: Precise Interval Analysis vs. Parity Games. In: Cuellar, J., Sere, K. (eds.) FM 2008. LNCS, vol. 5014, pp. 342–357. Springer, Heidelberg (2008)
[13] Gawlitza, T.M., Monniaux, D.: Improving Strategies via SMT Solving. In: Barthe, G. (ed.) ESOP 2011. LNCS, vol. 6602, pp. 236–255. Springer, Heidelberg (2011)
[14] Gawlitza, T.M., Seidl, H.: Computing Relaxed Abstract Semantics w.r.t. Quadratic Zones Precisely. In: Cousot, R., Martel, M. (eds.) SAS 2010. LNCS, vol. 6337, pp. 271–286. Springer, Heidelberg (2010)
[15] Gawlitza, T.M., Seidl, H.: Solving systems of rational equations through strategy iteration. TOPLAS (accepted, to appear)
[16] Jurdzinski, M.: Deciding the winner in parity games is in up ∩ co-up. Inf. Process. Lett. 68(3), 119–124 (1998)
[17] De Nicola, R. (ed.): ESOP 2007. LNCS, vol. 4421, pp. 157–172. Springer, Heidelberg (2007)
[18] Prajna, S., Jadbabaie, A.: Safety Verification of Hybrid Systems using Barrier Certificates. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 477–492. Springer, Heidelberg (2004)
[19] Sankaranarayanan, S., Sipma, H.B., Manna, Z.: Scalable Analysis of Linear Systems using Mathematical Programming. In: Cousot, R. (ed.) VMCAI 2005. LNCS, vol. 3385, pp. 25–41. Springer, Heidelberg (2005)
[20] Sankaranarayanan, S., Dang, T., Ivančić, F.: A Policy Iteration Technique for Time Elapse over Template Polyhedra. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 654–657. Springer, Heidelberg (2008)
[21] Tarski, A.: A lattice-theoretical fixpoint theorem and its appications. Pac. J. Math. 5, 285–309 (1955)
[22] Tiwari, A., Khanna, G.: Series of Abstractions for Hybrid Automata. In: Tomlin, C.J., Greenstreet, M.R. (eds.) HSCC 2002. LNCS, vol. 2289, pp. 465–478. Springer, Heidelberg (2002)
[23] Tiwari, A., Khanna, G.: Nonlinear Systems: Approximating Reach Sets. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 600–614. Springer, Heidelberg (2004)