Algebraic Invariants for Linear Hybrid Automata

Rupak Majumdar
Max Planck Institute for Software Systems, Kaiserslautern, Germany
rupak@mpi-sws.org

Joël Ouaknine
Max Planck Institute for Software Systems, Saarland Informatics Campus, Germany
Department of Computer Science, Oxford University, UK
joel@mpi-sws.org

Amaury Pouly
Université de Paris, IRIF, CNRS, F-75013 Paris, France
amaury.pouly@irif.fr

James Worrell
Department of Computer Science, Oxford University, UK
jbw@mpi-sws.org

Abstract

We exhibit an algorithm to compute the strongest algebraic (or polynomial) invariants that hold at each location of a given guard-free linear hybrid automaton (i.e., a hybrid automaton having only unguarded transitions, all of whose assignments are given by affine expressions, and all of whose continuous dynamics are given by linear differential equations). Our main tool is a control-theoretic result of independent interest: given such a linear hybrid automaton, we show how to discretise the continuous dynamics in such a way that the resulting automaton has precisely the same algebraic invariants.

2012 ACM Subject Classification Theory of computation → Timed and hybrid models; Theory of computation → Models of computation

Keywords and phrases Hybrid automata, algebraic invariants

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2020.32

Funding *Rupak Majumdar*: Supported by the European Research Council under Grant Agreement 610150 (ERC Synergy Grant ImPACT) and DFG grant 389792660 as part of TRR 248 (see https://perspicuous-computing.science).
Joël Ouaknine: Supported by ERC grant AVS-ISS (648701) and DFG grant 389792660 as part of TRR 248 (see https://perspicuous-computing.science).
Amaury Pouly: Part of this work was done at MPI-SWS.
James Worrell: Supported by EPSRC Fellowship EP/N008197/1.

Acknowledgements We thank Khalil Ghorbal for helpful discussions on the subject of this paper.

1 Introduction

Invariants are one of the most fundamental and useful notions in the quantitative sciences, appearing in a wide range of contexts, from gauge theory, dynamical systems, and control theory in physics, mathematics, and engineering to program verification, static analysis, abstract interpretation, and programming language semantics (among others) in computer science. In spite of decades of scientific work and progress, automated invariant synthesis remains a topic of active research, particularly in the fields of computer-aided verification and program analysis, and plays a central role in methods and tools seeking to establish correctness properties of computer systems; see, e.g., [8], and particularly Sec. 8 therein.

In this paper, we consider the task of computing *strongest algebraic inductive invariants* for *guard-free linear hybrid automata*. Hybrid automata are a formalism for describing systems or processes that combine discrete and continuous evolutions over their state variables (see, for example, [6]). A hybrid automaton is therefore equipped with a finite set of real-valued variables, as well as a finite set of control location (or modes). In each location, the variables evolve in continuous time according to some dynamics. Transitions between control locations may effect discrete updates (also known as *resets*) to these variables. A hybrid automaton is *guard-free* if the transitions do not have any guards, or preconditions, in order to be fired, and it is *linear* if the discrete updates on the variables consist entirely of affine transformations, and the continuous dynamics within each control location are defined by linear differential equations.[1]

An *invariant* assigns to each control location a fixed set of real values in such a way that through any trajectory of the hybrid automaton, the values of the variables always remain within the invariant. The invariant is *inductive* provided, informally speaking, that it is itself preserved by the (continuous and discrete) dynamics of the hybrid automaton. Finally, an invariant is *algebraic* (or polynomial) if it consists in a collection of varieties (or algebraic sets), i.e., positive Boolean combination of polynomial equalities. As it happens, the strongest polynomial invariant (i.e., smallest variety with respect to set inclusion) is obtained by taking the Zariski closure of the set of reachable configurations in each control location; such an invariant is always inductive provided that the dynamics are Zariski continuous.

There is a rich history of research into the computation of algebraic invariants for various classes of (discrete) computer programs; we refer the reader to our recent paper [7] and references therein. There has also been a substantial amount of work on algebraic invariant generation for hybrid systems, albeit in more recent years. One of the earliest pieces of work on this topic is by Rodrígez-Carbonell and Tiwari [15], who consider linear dynamical systems (i.e., linear hybrid automata with a single discrete location and no transition) and show how to compute strongest algebraic invariants for these. They then leverage abstract-interpretation techniques to derive algebraic invariants for linear hybrid automata, however without guarantees on the strength of the invariants. In [17], Sankaranarayanan et al. compute algebraic invariants for polynomial hybrid systems directly using constraint solving over template invariants (without however guaranteeing to obtain the strongest invariant). In subsequent work, Sankaranarayanan shows how to compute strongest algebraic invariants up to a fixed degree [16] for the same class of automata. Using different analytic techniques, Ghorbal and Platzer show in [5] how to compute algebraic invariants and differential invariants for polynomial hybrid automata; more precisely, they show that it is decidable whether a collection of algebraic sets forms an algebraic invariant; however they do not provide a procedure to guarantee that a given invariant is the strongest possible. In fact, while algebraic invariants for various classes of systems is a well-studied topic (see e.g., [2, 4, 13, 14, 1, 9] and references therein), as far as we know, none of these papers unconditionally guarantee the *strongest* algebraic invariants when applied to hybrid automata.

Main results. Our contributions in the present paper are threefold. **First,** for the class of guard-free linear hybrid automata, building on our recently developed invariant-generation techniques for affine programs [7], we show how to compute strongest algebraic invariants. Our main technical tools come from linear algebra, algebraic geometry, and Diophantine

[1] Following [6], the control locations of hybrid automata can also be equipped with 'invariant conditions' meant to enforce discrete transitions when the continuous variables reach the limit of their allowed range; however the hybrid automata considered in this paper do not feature any such invariant conditions.

geometry. **Second,** we show how one can discretise a guard-free linear hybrid automaton in such a way that the discretised version has precisely the same algebraic invariants as the original one; we are not aware of any such result in the extant control-theoretic and cyber-physical systems literature. **Third,** we show that as soon as equality guards are allowed, even for the restricted class of linear *switching systems*, there cannot exist an algorithm for computing strongest algebraic invariants, thereby establishing clearly a hard theoretical limit on how far the work presented here can be extended.

We now provide a slightly more detailed overview of our approach and results. Our main theorem uses an effective reduction from guard-free linear hybrid automata to affine programs that preserves algebraic invariants. More formally, given a hybrid automaton we show how to construct a finite (discrete-time) affine program that has the same set of variables and the same algebraic invariants as the original hybrid automaton. Thus we reduce the problem of computing strongest algebraic invariants for hybrid automata to the analogous problem for affine programs. In particular, this allow us to use a result in [7] to compute strongest algebraic invariants for hybrid automata. In fact, we show a stronger discretisation result that replaces the continuous dynamics with a finite set of discrete actions, which already preserves algebraic invariants.

In Section 7 we consider a simple but important class of hybrid systems, with purely continuous dynamics, called *switching systems*. A switching system [10] can transition arbitrarily between modes, but the variables are not reset when changing mode. It is natural to think of mode switches as being determined by an external controller which provides inputs to the system. We show that for a switching system, the Zariski closure of the set of reachable configurations is an irreducible variety. We exploit this feature to give a conceptually simple algorithm to compute the strongest algebraic invariant of a given switching system.

On the other hand, we consider the problem of computing the strongest algebraic invariant for switching systems that are augmented with the ability to test variables for zero on transitions. Here again the dynamics are exclusively continuous. We show that it is undecidable in general to compute a strongest algebraic invariant for such systems. Roughly speaking, we prove this result by defining a simulation of an arbitrary Minsky machine $\mathcal{C}$ by a hybrid automaton $\mathcal{A}$, such that we can effectively determine whether the set of reachable configurations of $\mathcal{C}$ is infinite from the strongest algebraic invariant of $\mathcal{A}$.

2 Examples

2.1 Bouncing ball

Consider a ball that bounces on horizontal slabs, as illustrated in Figure 1. The ball is moving at constant horizontal speed c and is subject to gravity along the vertical axis. We assume that there is no friction and that collisions are perfectly elastic. We do not want to make any assumption on the location of the slabs, to obtain the most general system. Thus from the system's point of the view, the positions of the slabs are "nondeterministic" and the slabs can "appear" at any moment.

We fix an arbitrary coordinate system in which the ball starts at position $(0, h)$ with initial velocities $(c, 0)$. The system is modelled using a single discrete location. There are three constants c (the horizontal speed), h (the initial height), and g (approximately $9.8ms^{-2}$). Technically speaking we could also include m (the mass of the ball), but it will not feature in the final set of invariants that we derive nor in our differential equations. There are five variables: t, x, y, v_x, and v_y, where t is the time variable. The differential equations are simply Newton's laws of motion. There is a single reset with no guards modelling the

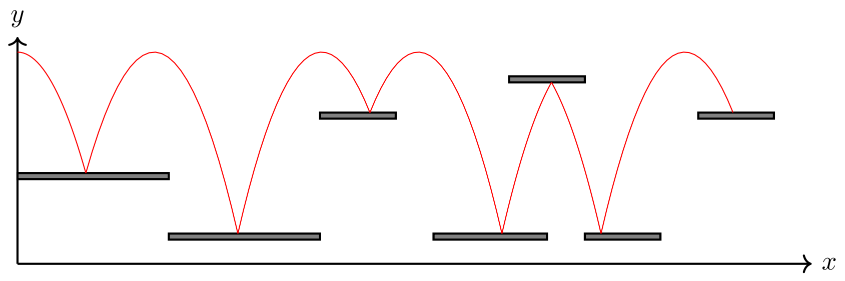


Figure 1 A ball bouncing on horizontal slabs.

action of the ball bouncing on each of the slabs (notably ensuring that the vertical velocity is instantly inverted). We obtain the hybrid system described in Figure 2a.

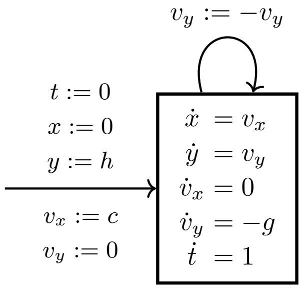
Now we come to the invariants. The most obvious is that the horizontal speed is constant: $v_x = c$, which in turn entails that $x = tc$. Next, consider an inertial coordinate system moving horizontally to the right at speed c. In that system energy must be conserved. Initially there is no kinetic energy, and all the potential energy amounts to mgh (where m is the mass of the ball). At any subsequent time t, the sum of the kinetic and potential energy must therefore sum to that value, i.e., $\frac{1}{2}mv_y^2 + mgy = mgh$, so $v_y^2 + 2g(y - h) = 0$. These must be the only invariants: the ambient space is 5-dimensional (given that our variables are t, x, y, v_x, and v_y), and the corresponding variety (with 3 equations) is two dimensional. But the system has indeed exactly two degrees of freedom, since t and v_y can be set to arbitrary values (provided $|v_y| \leq gt$), and once t and v_y are fixed, every other variable is fixed. In summary, the strongest algebraic invariant is the conjunction of the following three equations:

$$v_x = c, \qquad x = tc, \qquad v_y^2 + 2g(y - h) = 0. \tag{1}$$

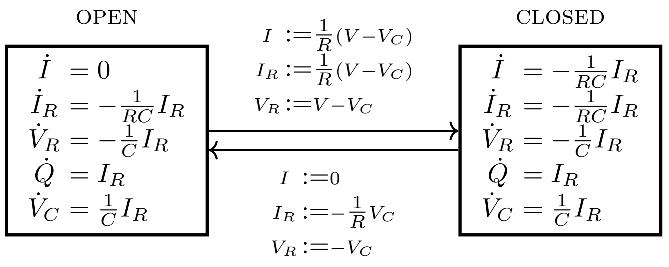
One way to obtain the invariants in (1) is to construct an affine program (i.e., a hybrid system with trivial continuous dynamics) over the same set of locations and variables as the original hybrid automaton and that moreover has the same algebraic invariants. One can then apply Theorem 2 to compute a strongest invariant of the latter. In the case at hand, such an affine program can be obtained by a direct time-discretisation construction in which the continuous flow of variables is replaced by a self-loop—see Figure 4a—which performs a simultaneous assignment $x := x + v_x$, $y := y + v_y - \frac{1}{2}g$, $v_y := v_y - g$ and $t := t + 1$. Here, essentially, we have replaced a system of linear differential equations of the form $\dot{x}(t) = Ax(t)$ with an analogous system of linear difference equations $x(t + 1) = e^A x(t)$; this is sound because the discrete semigroup $\{e^{An} : n \in \mathbb{N}\}$ is Zariski dense is the semigroup $\{e^{At} : t \geq 0\}$ (see Proposition 13). Unfortunately, due to the presence of the matrix exponential, this naive construction in general yields affine programs with transcendental constants, which precludes applying the algorithm described in Theorem 2. As it happens, in the case at hand, the matrix e^A has rational entries. We refer to Section B of the Appendix for more details of how the affine program is obtained.

2.2 RC circuit

Consider an RC circuit with a switch, as illustrated in Figure 3. When the switch is on, the capacitor is connected to a battery and charges. When the switch is off, the capacitor discharges through the resistor. The battery has constant voltage V, the resistor has resistance R and the capacitor has capacity C. There are 5 variables: the current I in the



(a) Bouncing ball.



OPEN
$\dot{I} = 0$
$\dot{I}_R = -\frac{1}{RC}I_R$
$\dot{V}_R = -\frac{1}{C}I_R$
$\dot{Q} = I_R$
$\dot{V}_C = \frac{1}{C}I_R$
$I := \frac{1}{R}(V - V_C)$
$I_R := \frac{1}{R}(V - V_C)$
$V_R := V - V_C$
$I := 0$
$I_R := -\frac{1}{R}V_C$
$V_R := -V_C$
CLOSED
$\dot{I} = -\frac{1}{RC}I_R$
$\dot{I}_R = -\frac{1}{RC}I_R$
$\dot{V}_R = -\frac{1}{C}I_R$
$\dot{Q} = I_R$
$\dot{V}_C = \frac{1}{C}I_R$

(b) RC circuit.

Figure 2 Examples of hybrid systems: (a) bouncing ball, (b) RC circuit

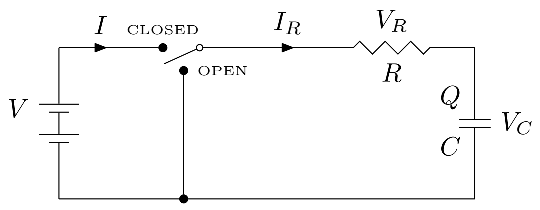


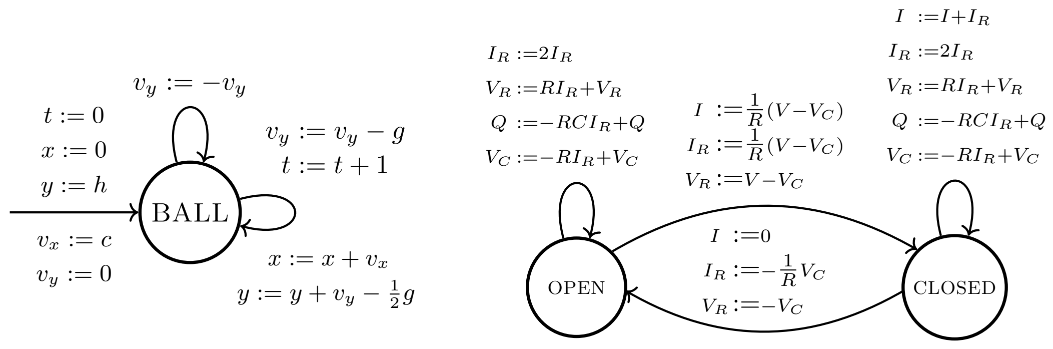
Figure 3 An RC circuit with a switch to disconnect the battery.

wire between the battery and the switch, the voltages V_R and V_C across the resistor and capacitor respectively, the current I_R flowing through the resistor, and finally the charge Q held by the capacitor. All derivatives are with respect to time, though this time we choose not to include an explicit variable for the passage of time. There are two discrete locations, OPEN and CLOSED, corresponding to the two possible positions of the switch. We assume the switch starts in the OPEN position. When switching from OPEN to CLOSED, all variables but Q and V_C experience a reset. We obtain the hybrid system described in Figure 2b.

In this example, there is one set of invariants per location. In both locations, we clearly have the invariants associated with the passive components: $Q = CV_C$ and $V_R = RI_R$. In the OPEN location, we further have the invariants $I = 0$ and $V_R = -V_C$. On the other hand, in the CLOSED location, we have $I = I_R$ and $V = V_R + V_C$. These must be the only invariants: once Q is fixed, all variables are uniquely determined. In summary, the invariants are

$$\begin{array}{lllll} \text{OPEN:} & Q = CV_C, & V_R = RI_R, & I = 0, & V_R = -V_C, \\ \text{CLOSED:} & Q = CV_C, & V_R = RI_R, & I = I_R, & V_R = V - V_C. \end{array}$$

The above invariants were obtained by producing an affine program with the same set of invariants as the hybrid system in Figure 2b, and applying Theorem 2 to compute a strongest invariant of the latter. However, in this case, the naive discretisation procedure that was used in Section 2.1 yields transcendental constants (for the reasons described in Section 2.1). In fact the affine program that we construct, shown in Figure 4b, is the result of applying the more abstract discretisation procedure that is described in Section 6. The core idea of the latter procedure is, given a square matrix A with rational coefficients, to produce a matrix B, with algebraic coefficients, such that the respective semigroups $\{e^{At} : t \geq 0\}$ and $\{B^n : n \in \mathbb{N}\}$ have the same Zariski closure (see Proposition 9). The algebraic nature of this construction makes it more subtle to relate the dynamics of the resulting affine program to the original hybrid system (we invite the reader to compare the automata shown respectively



(a) Affine program modelling a bouncing ball. **(b)** Affine program modelling an RC circuit.

Figure 4 Time discretisation of the hybrid systems in Figure 2.

in Figures 2b and 4b). We refer to Section B of the Appendix for more details of how the affine program is obtained.

3 Mathematical Background

Let $\mathbb{K}$ be a field. Given a set $X \subseteq \mathbb{K}^n$, we denote by $\mathbf{I}(X)$ the ideal of polynomials in $\mathbb{K}[x_1, \ldots, x_n]$ that vanish on X. Given an ideal $I \subseteq \mathbb{K}[x_1, \ldots, x_n]$, we denote by $V(I) \subseteq \mathbb{K}^n$ the set of common zeroes of the polynomials in I. A set $X \subseteq \mathbb{K}^n$ is said to be an *affine variety* (also called an *algebraic set*) if $X = V(I)$ for some ideal $I \subseteq \mathbb{K}[x_1, \ldots, x_n]$. By the Hilbert Basis Theorem, every affine variety can be described as the set of common zeroes of finitely many polynomials. We identify $\mathrm{GL}_n(\mathbb{K})$, the set of $n \times n$ invertible matrices with entries in $\mathbb{K}$, with the variety $\{(A, y) \in \mathbb{K}^{n^2+1} : \det(A) \cdot y = 1\}$.

Given an affine variety $X \subseteq \mathbb{K}^n$, the *Zariski topology* on X has as closed sets the subvarieties of X, i.e., those sets $A \subseteq X$ that are themselves affine varieties in $\mathbb{K}^n$. Given an arbitrary set $S \subseteq X$, we write $\overline{S}$ for its closure in the Zariski topology on X.

A set $S \subseteq X$ is *irreducible* if for all closed subsets $A_1, A_2 \subseteq X$ such that $S \subseteq A_1 \cup A_2$ we have either $S \subseteq A_1$ or $S \subseteq A_2$. It is well known that the Zariski topology on a variety is Noetherian. In particular, any closed subset A of X can be written as a finite union of *irreducible components*, where an irreducible component of A is a maximal irreducible closed subset of A.

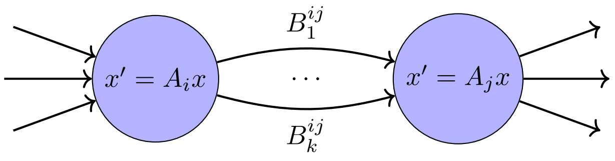
The class of *constructible* subsets of a variety X is obtained by taking all finite Boolean combinations (including complementation) of Zariski closed subsets. Suppose that the underlying field $\mathbb{K}$ is algebraically closed. Since the first-order theory of algebraically closed fields admits quantifier elimination, the constructible subsets of X are exactly the subsets of X that are first-order definable over $\mathbb{K}$.

A function $f : \mathbb{K}^m \to \mathbb{K}^n$ is said to be a *polynomial map* if there exist polynomials $p_1, \ldots, p_n \in \mathbb{K}[x_1, \ldots, x_m]$ such that $f(\boldsymbol{a}) = (p_1(\boldsymbol{a}), \ldots, p_n(\boldsymbol{a}))$ for all $\boldsymbol{a} \in \mathbb{K}^m$. Recall that polynomial maps are Zariski-continuous and thus $f(\overline{X}) \subseteq \overline{f(X)}$ for a polynomial map f. In particular matrix multiplication is a Zariski-continuous map $\mathbb{K}^{n^2} \times \mathbb{K}^{n^2} \to \mathbb{K}^{n^2}$.

Given a complex variety $V \subseteq \mathbb{C}^n$, the intersection $V \cap \mathbb{R}^n$, which is a real variety, can be computed effectively. Indeed if V is represented by the ideal $I \subseteq \mathbb{C}[x_1, \ldots, x_n]$, then $V \cap \mathbb{R}^n$ is represented by the ideal generated by $\{p_1, p_2 \in \mathbb{R}[x_1, \ldots, x_n] : p_1 + ip_2 \in I\}$ Given $S \subseteq \mathbb{R}^n$, write $\overline{S}^{\mathbb{R}}$ for its real Zariski closure and $\overline{S}$ for its complex Zariski closure (i.e., we treat the complex Zariski closure of $S \subseteq \mathbb{R}^n$ as the default). It is straightforward to verify that $\overline{S}^{\mathbb{R}} = \overline{S} \cap \mathbb{R}^n$.

4 Algebraic Invariants for Hybrid Automata

We are concerned with computing strongest algebraic invariants for the subclass of hybrid automata that has no guards, linear discrete updates, and linear continuous dynamics. Each location of such an automaton specifies a linear differential equation ($x' = Ax$), and each transition between states (nondeterministic choices are allowed) specifies a linear transformation ($x \mapsto Bx$). Such a hybrid automaton can be pictured as follows:



Formally, such an automaton $\mathcal{A}$ in dimension d is a tuple (Q, A, E, T), where Q is a finite set of locations, $A = \{A_q : q \in Q\}$ is a family of real $d \times d$ matrices, $E \subseteq Q \times \mathbb{R}^{d\times d} \times Q$ is a set of transitions labelled by real $d \times d$ matrices, and $\{T_q : q \in Q\}$ is a family of algebraic subsets of $\mathbb{R}^d$. Matrix $A_q \in \mathbb{R}^{d\times d}$ describes the continuous dynamics at location $q \in Q$ and $T_q \subseteq \mathbb{R}^d$ is the set of initial states in location q. We assume that the entries of all matrices are algebraic numbers and that the polynomials defining T_q have algebraic coefficients.

We will consider the subclasses of automata with the following restrictions:

- *affine programs*: $A_q = 0$ for all $q \in Q$, and E is finite,
- *constructible affine programs*: $A_q = 0$ for all $q \in Q$, and E is constructible,
- *switching systems*: $E = \{(p, I_n, q) : p, q \in Q\}$, i.e., every pair of locations is connected by an edge that does not update the variables,
- *linear hybrid automata*: E is finite.

In affine programs, variables are only updated on discrete edges: there is no continuous evolution within locations. At the other end of the spectrum, in switching systems variables only evolve continuously, and there are no discrete updates. The full class of linear hybrid automata accommodate both discrete and continuous updates to the variables.

The *collecting semantics* of $\mathcal{A}$ assigns to each location $q \in Q$ the set $S_q \subseteq \mathbb{R}^d$ of states that can occur in location q during a run of the automaton, starting from a configuration $(q, \boldsymbol{a})$ for some $\boldsymbol{a} \in T_q$. Formally, this the smallest family (with respect to set inclusion) such that

$$\begin{array}{lll} S_q & \supseteq & T_q \quad\text{for all } q \in Q, \\ S_q & \supseteq & BS_p \quad\text{for all } (p, B, q) \in E, \\ S_q & \supseteq & e^{A_q t} S_q \quad\text{for all } t \in \mathbb{R}_{\geqslant 0}. \end{array}$$

Equivalently, let the operator $\Phi_{\mathcal{A}} : \mathcal{P}(\mathbb{R}^d)^Q \to \mathcal{P}(\mathbb{R}^d)^Q$ be defined by

$$\left(\Phi_{\mathcal{A}}(S)\right)_q = T_q \cup \bigcup_{t \geqslant 0} e^{A_q t} S_q \cup \bigcup_{(p,B,q)\in E} BS_p.$$

Then S is the least fixed-point of $\Phi_{\mathcal{A}}$ with respect to set inclusion. Such a least fixed-point exists because $\Phi_{\mathcal{A}}$ is monotone.

In general we say that a family of sets $\{S'_q\}_{q \in Q}$, with $S'_q \subseteq \mathbb{R}^d$ is an *inductive invariant* if $\Phi_{\mathcal{A}}(S') \subseteq S'$, i.e., the family is pre-fixed-point of $\Phi_{\mathcal{A}}$. If each set S'_q is algebraic then we moreover say that $\{S'_q\}_{q \in Q}$ is an *inductive algebraic invariant.*

Given $P \in \mathbb{R}[x_1, \dots, x_d]$, we say that the relation $P = 0$ holds at location q if P vanishes on S_q. We are interested in computing at each location $q \in Q$ a finite set of polynomials that generates the ideal $I_q := \mathbf{I}(S_q) \subseteq \mathbb{R}[x_1, \dots, x_d]$ of all polynomial relations that hold at location q. The real variety $V_q := V(I_q) = \overline{S_q}^{\mathbb{R}}$ corresponding to I_q is the Zariski closure of S_q viewed a subset of the affine space $\mathbb{R}^d$.

Note that the collection $V(\mathcal{A}) := \{V_q : q \in Q\}$ defines an inductive algebraic invariant. Inductiveness amounts to the following two claims:

- for every edge $(p, B, q) \in E$, we have $BV_p \subseteq V_q$,
- for every $q \in Q$ and $t \in \mathbb{R}_{\geqslant 0}$, we have $e^{A_q t} V_q \subseteq V_q$.

The first point follows from the fact that $x \mapsto Bx$ is Zariski-continuous; the second point likewise follows from the fact that for every $t \in \mathbb{R}_{\geqslant 0}$ the map $x \mapsto e^{A_q t} x$ is Zariski-continuous.

▶ **Lemma 1.** *For an automaton $\mathcal{A}$, $V(\mathcal{A})$ is the least fixpoint of the map $X \mapsto \overline{\Phi_{\mathcal{A}}(X)}$.*

Proof. Let S denote the collecting semantics of $\mathcal{A}$, then

$$\begin{aligned} V(\mathcal{A}) &= \overline{S} && \text{by definition of } V(\mathcal{A}) \\ &= \overline{\Phi_{\mathcal{A}}(S)} && \text{by definition of } S \\ &= \overline{\Phi_{\mathcal{A}}(\overline{S})} && \text{by Zariski-continuity of } \Phi_{\mathcal{A}} \\ &= \overline{\Phi_{\mathcal{A}}(V(\mathcal{A}))} \end{aligned}$$

thus $V(\mathcal{A})$ is indeed a fixed-point. Conversely, let X be such that $X = \overline{\Phi_{\mathcal{A}}(X)}$. Then X is closed and clearly $\Phi_{\mathcal{A}}(X) \subseteq \overline{\Phi_{\mathcal{A}}(X)} = X$ so it is a pre-fixpoint of $\Phi_{\mathcal{A}}$. By virtue of S being the least (pre-)fixpoint of $\Phi_{\mathcal{A}}$ we must have $S \subseteq X$. But then $\overline{S} \subseteq X$ i.e., $V(\mathcal{A}) \subseteq X$. ◀

The discussion above shows that $V(\mathcal{A})$, the Zariski closure of the collecting semantics, is the least inductive invariant of $\mathcal{A}$. Previously we have shown how to compute the Zariski closure of the collecting semantics of an affine program:

▶ **Theorem 2** ([7]). *There is an algorithm that given a constructible affine program $\mathcal{A}$ computes $V(\mathcal{A}) = \{V_q : q \in Q\}$—the real Zariski closure of its collecting semantics.*

Note that this theorem also applies to (finite) affine programs since those are particular instances of constructible affine programs.

The main result of the current paper extends the above result by accommodating the continuous dynamics of hybrid automata:

▶ **Theorem 3.** *There is an algorithm that given a guard-free linear hybrid automaton $\mathcal{A}$ computes $\{V_q : q \in Q\}$—the real Zariski closure of its collecting semantics.*

5 Proof of Theorem 3

The proof of Theorem 3 has two main ingredients. First, in Subsection 5.1, we show how to compute the Zariski closure of the orbit of a single continuous linear dynamics (i.e., the continuous evolution in a single state). Then, in Subsection 5.2, we show that computing the real Zariski closure of the collecting semantics of a linear hybrid automaton can be effectively reduced to computing the real Zariski closure of the collecting semantics of a constructible affine program. At this point, we can apply the algorithm from Theorem 2.

5.1 Linear Continuous Dynamics

The first step towards computing Zariski closure in the general case is to be able to handle the case of one differential equation. Write $\mathbb{A}$ for the field of algebraic numbers. Let $A \in \mathbb{A}^{d\times d}$ and $x_0 \in \mathbb{A}^d$, then the solution to $x(0) = x_0$, $x'(t) = Ax(t)$ is given by $x(t) = e^{At}x_0$. We are interested in computing the Zariski closure of the orbit $\{x(t) : t \in \mathbb{R}_{\geqslant 0}\}$. Since the map $\phi : M \mapsto Mx_0$ is Zariski-continuous, we have that

$$\overline{\{x(t) : t \in \mathbb{R}_{\geqslant 0}\}} = \overline{\{e^{At}x_0 : t \in \mathbb{R}_{\geqslant 0}\}} = \overline{\phi(\{e^{At} : t \in \mathbb{R}_{\geqslant 0}\})} = \overline{\phi(\overline{\{e^{At} : t \in \mathbb{R}_{\geqslant 0}\}})}$$

and thus it suffices to compute $\overline{\{e^{At} : t \in \mathbb{R}_{\geqslant 0}\}}$. Furthermore, let $\mathcal{O}_A := \{e^{At} : t \in \mathbb{R}\}$, which is a commutative group. We claim that $\overline{\{e^{At} : t \in \mathbb{R}_{\geqslant 0}\}} = \overline{\mathcal{O}_A}$. The left-to-right inclusion is clear. The converse inclusion comes from the fact that an exponential polynomial (in one variable), being an analytic function, vanishes over $\mathbb{R}_{\geqslant 0}$ if and only if it vanishes over $\mathbb{R}$. Thus it suffices to compute $\overline{\mathcal{O}_A}$.

The following lemma gives a description of the ideal of the variety $\overline{\mathcal{O}_A}$ when A is diagonal.

▶ **Lemma 4.** *Let $A = \operatorname{diag}(\lambda_1, \dots, \lambda_d) \in \mathbb{A}^{d\times d}$ be a diagonal matrix, then*

$$\overline{\mathcal{O}_A} = \{\operatorname{diag}(z_1, \dots, z_d) : \forall p \in I, p(z_1, \dots, z_d) = 0\} ,$$

where $I = \langle z^a - z^b : a - b \in L\rangle$ and $L = \{n \in \mathbb{Z}^d : n_1\lambda_1 + \cdots + n_d\lambda_d = 0\}$. Furthermore, one can compute a basis for L considered as an abelian group under addition.

Proof. Clearly $e^{At} = \operatorname{diag}(e^{\lambda_1 t}, \dots, e^{\lambda_d t})$. Since the set of diagonal matrices is closed, then the closure is of the form

$$\overline{\mathcal{O}_A} = \{\operatorname{diag}(z_1, \dots, z_n) : p_1(z) = \cdots = p_k(z) = 0\}$$

for some polynomials $p_1, \dots, p_k$. Thus, the ideal I of the closure is generated by all polynomials p such that $p(e^{\lambda_1 t}, \dots, e^{\lambda_d t}) = 0$ for all $t \in \mathbb{R}$. Let J be the ideal of all polynomials $x^a - x^b$ with $a, b \in \mathbb{N}^d$ such that $\lambda \cdot a = \lambda \cdot b$. Clearly $J \subseteq I$ since if $\lambda \cdot a = \lambda \cdot b$, then $(e^{\lambda_1 t})^{a_1} \cdots (e^{\lambda_d t})^{a_d} - (e^{\lambda_1 t})^{b_1} \cdots (e^{\lambda_d t})^{b_d} = e^{(\lambda\cdot a)t} - e^{(\lambda\cdot b)t} = 0$ for all $t \in \mathbb{R}$. Conversely, assume by contradiction that $p \in I \setminus J$ and write $p = \sum_{i=1}^{r} b_i m_i$ for some $b_i \in \mathbb{A}$ and monomials $m_1, \dots, m_r$. Further choose p so that r is minimal. For each monomial $m_i(x) = x_1^{a_1} \cdots x_d^{a_d}$, let $\mu_i := \lambda \cdot a$. Then we must have $\mu_i \neq \mu_j$ for $i \neq j$ because otherwise $m_i - m_j \in J$ and $p - b_i(m_i - m_j) \in I \setminus J$ would have fewer terms than p. Since the maps $t \mapsto e^{\mu_1 t}, \dots, t \mapsto e^{\mu_d t}$ are linearly independent, it follows that $b_1 = \cdots = b_r = 0$ which is contradiction. Thus we must have have $I = J$. It is immediate to see that J is generated by all polynomials $x^a - x^b$ such that $\lambda \cdot (a - b) = 0$, thus it suffices to compute $L = \{a \in \mathbb{Z}^d : \lambda \cdot a = 0\}$, the set of additive relations of λ. Notice that L is an additive subgroup of $\mathbb{Z}^d$ and as such must be finitely generated. An upper bound on the size of the elements of a basis of L can be found in [11] and therefore we can compute a basis for L and thus J, I and $\overline{\mathcal{O}_A}$. ◀

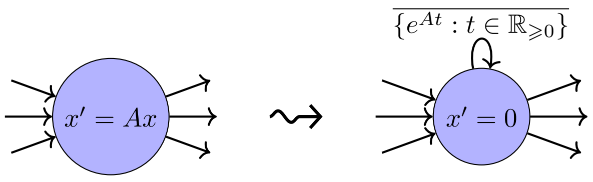
A corollary of this result is that we can compute $\overline{\mathcal{O}_A}$ by separating the diagonal and nilpotent parts of A. The fact that this closure is computable is well known (see, e.g., [15]), however the particular form of the polynomials determining the Zariski closure is the important part of Lemma 4 for our purposes. In particular, Lemma 4 is instrumental in the proof of Proposition 6, which reduces the problem of computing the strongest algebraic invariants of hybrid automata to the analogous problem for affine programs.

▶ **Proposition 5.** *There is an algorithm that given $A \in \mathbb{A}^{d\times d}$, computes $\overline{\{e^{At} : t \in \mathbb{R}_{\geqslant 0}\}}$.*

Proof. We can write $A = P(D+N)P^{-1}$ where P is invertible, D is diagonal, N is nilpotent, and D and N commute. Notice that $\mathcal{O}_D$ only consists of diagonal matrices and $\mathcal{O}_N$ of unipotent matrices. Since $\mathcal{O}_A$ is a commutative group, and D and N commute, we have that $\mathcal{O}_A = P(\mathcal{O}_D \cdot \mathcal{O}_N)P^{-1}$, and thus $\overline{\mathcal{O}_A} = P\overline{\overline{\mathcal{O}_D} \cdot \overline{\mathcal{O}_N}}P^{-1}$. All the operations in this equation are effective thus it suffices to compute $\overline{\mathcal{O}_D}$ as explained above, and $\overline{\mathcal{O}_N}$. But since N is nilpotent, $q_n(t) := e^{Nt}$ is really a polynomial in Nt and thus in t. It follows that $\overline{\mathcal{O}_N} = \overline{q_n(\mathbb{R})} = \overline{q_n(\overline{\mathbb{R}})} = \overline{q_n(\mathbb{C})}$ which we know how to compute. ◀

5.2 From Continuous Dynamics to Constructible Discrete Dynamics

In the previous section, we saw that given a linear continuous system, one can compute its strongest inductive algebraic invariant. The main obstacle to generalize this approach to linear hybrid system is the mixture of discrete and continuous dynamics. In particular, the invariants are not irreducible. The idea to work around this problem is to replace the continuous evolution of the variables in each location of a hybrid automaton with an infinite set of discrete transitions. Graphically this corresponds to rewriting the automaton as follows:



The key point is that this infinite set is very special: it is constructible (and in fact algebraic), and hence falls into the scope of our previous paper [7].

▶ **Proposition 6.** *Given a linear hybrid automaton $\mathcal{A}$, one can compute a constructible affine program $\mathcal{A}'$ that has the same algebraic invariants, i.e., $V(\mathcal{A}) = V(\mathcal{A}')$.*

Proof. The idea is to replace each continuous dynamics $x' = A_q x$ by a closed (and thus constructible) set of discrete transitions: $\overline{\{e^{A_q t} : t \in \mathbb{R}_{\geqslant 0}\}}$, which we know how to compute thanks to Proposition 5.

Formally, let $\mathcal{A} = (Q, A, E, T)$ be a linear hybrid automaton. Define the constructible affine program $\mathcal{A}' = (Q, A', E', T)$ where $A'_q = 0$ for all $q \in Q$ and

$$E' = E \cup \left\{(q, X, q) : q \in Q, X \in \overline{\mathcal{O}_{A_q}}\right\}$$

where $\mathcal{O}_A := \left\{e^{At} : t \in \mathbb{R}_{\geqslant 0}\right\}$. Note that it is constructible because $\overline{\mathcal{O}_{A_q}}$ is closed (and thus constructible).

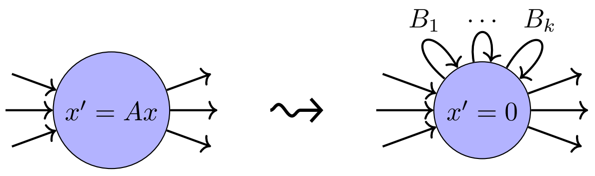
We will now relate the operators $\Phi_{\mathcal{A}}$ and $\Phi_{\mathcal{A}'}$. Given $X \in \mathcal{P}(\mathbb{C}^d)^Q$ we have

$$\begin{aligned}
\overline{\Phi_{\mathcal{A}}(X)_q} &= \overline{T_q \cup (\mathcal{O}_{A_q} \cdot X_q) \cup \bigcup_{(p,B,q)\in E} BX_p} \\
&= \overline{T_q \cup \overline{\mathcal{O}_{A_q} \cdot X_q} \cup \bigcup_{(p,B,q)\in E} BX_p} \\
&= \overline{T_q \cup (\overline{\mathcal{O}_{A_q}} \cdot X_q) \cup \bigcup_{(p,B,q)\in E} BX_p} \\
&= \overline{\Phi_{\mathcal{A}'}(X)_q}\,.
\end{aligned}$$

Thus the maps $X \mapsto \overline{\Phi_{\mathcal{A}}(X)}$ and $X \mapsto \overline{\Phi_{\mathcal{A}'}(X)}$ are identical. But, by Lemma 1, $V(\mathcal{A})$ is the least fixpoint of $X \mapsto \overline{\Phi_{\mathcal{A}}(X)}$ and $V(\mathcal{A}')$ is the least fixpoint of $X \mapsto \overline{\Phi_{\mathcal{A}'}(X)}$. We conclude that $V(\mathcal{A}') = V(\mathcal{A})$. ◀

6 From Continuous Dynamics to Finite Discrete Dynamics

The proof of Theorem 3 used an effective reduction from a linear hybrid automaton to a constructible affine program with the same set of algebraic invariants. In this section, we show a stronger result—of independent control-theoretic interest—that one can in fact reduce a linear hybrid automaton to a *finite* affine program with the same set of algebraic invariants. The idea is to replace the continuous evolution of the variables in each location of a hybrid automaton with a finite set of discrete transitions. Graphically this corresponds to rewriting the automaton as follows:



The result is stronger because finite affine programs are also constructible.

Mathematically, the task is as follows: *Given $A \in \mathbb{A}^{d\times d}$, find $B_1,\dots,B_k \in \mathbb{A}^{d\times d}$ such that $\overline{\{e^{At} : t\in\mathbb{R}\}} = \overline{\langle B_1,\dots,B_k\rangle}$.* There is a conceptually simple approach to this problem: namely for every matrix $A \in \mathbb{A}^{d\times d}$ and rational number τ we have $\overline{\{e^{At} : t\in\mathbb{R}\}} = \overline{\langle e^{A\tau}\rangle}$ (see Section A). But this does not fulfil our desiderata, since it is not possible in general to find $\tau\in\mathbb{R}$ such that $e^{A\tau}$ has exclusively algebraic entries. Nevertheless given $A \in \mathbb{A}^{d\times d}$ it is possible to find $B \in \mathbb{A}^{d\times d}$ such that $\overline{\{e^{At} : t\in\mathbb{R}\}} = \overline{\langle B\rangle}$. The idea is to construct B such that there is a correspondence between the set of additive relations satisfied by the eigenvalues of A and the multiplicative relations satisfied by the eigenvalues of B.

▶ **Proposition 7.** *Let $a_1,\dots,a_d\in\mathbb{C}$ be algebraic numbers. Then we can compute rational numbers $\lambda_1,\dots,\lambda_d$ such that $a_1n_1+\cdots+a_dn_d=0$ iff $\lambda_1^{n_1}\cdots\lambda_d^{n_d}=1$ for all $n_1,\dots,n_d\in\mathbb{Z}$.*

Proof. Let s be the dimension of the $\mathbb{Q}$-vector space spanned by $a_1,\dots,a_d$. By computing a basis over $\mathbb{Q}$ for the number field $\mathbb{Q}(a_1,\dots,a_d)$ and the respective rational coordinates of $a_1,\dots,a_d$ with respect to this basis, we obtain an $s\times d$ integer matrix A such that for every integer vector $\boldsymbol{x}=(n_1,\dots,n_d)\in\mathbb{Z}^d$ we have $n_1a_1+\cdots+n_da_d=0$ iff $A\boldsymbol{x}=0$.

Now write $A=PBQ$, where B is an $s\times d$ matrix in Smith normal form and P,Q are unimodular square matrices. Since B has rank s it has the form $B=\begin{pmatrix}D & 0\end{pmatrix}$ for D an $s\times s$ diagonal matrix of full rank.

We define positive integers $\mu_1,\dots,\mu_d$ as follows. Choose $\mu_1,\dots,\mu_s$ to be the first s prime numbers and let $\mu_{s+1}=\dots=\mu_d=1$. Write $Q=(q_{ij})$ and define $\lambda_i=\mu_1^{q_{1i}}\cdots\mu_d^{q_{di}}$ for $i\in\{1,\dots,d\}$.

Then for all $n_1,\dots,n_d\in\mathbb{Z}$ we have

$$\begin{aligned}
\lambda_1^{n_1}\cdots\lambda_d^{n_d}=1 \quad&\Leftrightarrow\quad \mu_1^{(Q\boldsymbol{x})_1}\cdots\mu_d^{(Q\boldsymbol{x})_d}=1\\
&\Leftrightarrow\quad (Q\boldsymbol{x})_1=0,\dots,(Q\boldsymbol{x})_s=0\\
&\Leftrightarrow\quad BQ\boldsymbol{x}=0 \quad (\text{since } B=\begin{pmatrix}D & 0\end{pmatrix})\\
&\Leftrightarrow\quad PBQ\boldsymbol{x}=0 \quad (\text{since } P \text{ is invertible})\\
&\Leftrightarrow\quad A\boldsymbol{x}=0\\
&\Leftrightarrow\quad a_1n_1+\cdots+a_dn_d=0\,.
\end{aligned}$$

◀

▶ **Corollary 8.** *Let D be a $d \times d$ diagonal matrix with algebraic entries. Then there exists a diagonal matrix D', of the same dimension and with rational entries, such that $\overline{\langle e^D \rangle} = \overline{\langle D' \rangle}$.*

Proof. Write $D = \mathrm{diag}(a_1, \dots, a_d)$ and let rational numbers $\lambda_1, \dots, \lambda_d$ be chosen as in Proposition 7, i.e., such that $a_1 n_1 + \cdots + a_d n_d = 0$ iff $\lambda_1^{n_1} \cdots \lambda_d^{n_d} = 1$ for all $n_1, \dots, n_d \in \mathbb{Z}$. Define $D' = \mathrm{diag}(\lambda_1, \dots, \lambda_d)$. By Lemma 4, the ideal of the variety $\overline{\langle e^D \rangle}$ is generated by $z^n - z^m$ such that $(n_1 - m_1)a_1 + \cdots + (n_d - m_d)a_d = 0$. On the other hand, it follows from [3, Lemma 6] that the ideal of the variety $\overline{\langle D' \rangle}$ is generated by $z^n - z^m$ such that $\lambda_1^{n_1 - m_1} \cdots \lambda_d^{n_d - m_d} = 1$. But by construction the additive relations of a are the same as the multiplicative relations of λ, therefore the ideals are the same. It follows $\overline{\langle e^D \rangle} = \overline{\langle D' \rangle}$. ◀

▶ **Proposition 9.** *Let $A \in \mathbb{Q}^{d \times d}$ be a rational matrix. Then there exists an algebraic matrix B such that $\overline{\langle B \rangle} = \overline{\langle e^A \rangle} = \overline{\{e^{At} : t \in \mathbb{R}\}}$.*

Proof. Let P be an invertible matrix such that $A = P^{-1}(D+N)P$ with $D = \mathrm{diag}(a_1, \dots, a_d)$ diagonal and N a nilpotent Jordan matrix. By Corollary 8 there exists a rational diagonal matrix D' such that $\overline{\langle D' \rangle} = \overline{\langle e^D \rangle}$. We now define $B := P^{-1}(D' e^N)P$ where $D' = \mathrm{diag}(\lambda_1, \dots, \lambda_d)$. Note that e^N is a matrix of rational numbers. Then we have:

$$\overline{\langle e^A \rangle} = P^{-1}\overline{\langle e^D e^N \rangle}P = P^{-1}\overline{\overline{\langle e^D \rangle} \cdot \overline{\langle e^N \rangle}}P = P^{-1}\overline{\overline{\langle D' \rangle} \cdot \overline{\langle e^N \rangle}}P = P^{-1}\overline{\langle D' e^N \rangle}P \quad = \overline{\langle B \rangle}\,.$$

◀

▶ **Proposition 10.** *Given a linear hybrid automaton $\mathcal{A}$, one can compute a finite affine program $\mathcal{A}'$ that has the same algebraic invariants, i.e., $V(\mathcal{A}) = V(\mathcal{A}')$.*

Proof. Suppose that $\mathcal{A} = (Q, A, E, T)$. We define $\mathcal{A}' = (Q, A', E', T)$, where $A'_q = 0$ for all $q \in Q$ and

$$E' = E \cup \{(q, B_q, q) : q \in Q\}\,,$$

with B_q is an algebraic matrix such that $\overline{\langle e^{A_q} \rangle} = \overline{\langle B_q \rangle}$ for all $q \in Q$. The existence of the matrices B_q is guaranteed by Proposition 9. In other words, we obtain $\mathcal{A}'$ from $\mathcal{A}$ by setting the derivative of all variables to 0 in every location and by adding a compensatory selfloop edge to every location.

The reasoning that $V(\mathcal{A}) = V(\mathcal{A}')$ is entirely analogous to that in the proof of Proposition 6. ◀

7 Switching Systems

Finally, we consider the special case of *switching systems*, that is, hybrid systems in which every pair of locations is connected by an edge and such that variables are not updated on discrete edges. It is known that reachability is undecidable even for this restricted class of systems [12]. We show two results: first, we give a simple algorithm to compute strongest algebraic invariants for this class, benefitting from the fact that the strongest algebraic invariant is an irreducible variety (Subsection 7.1); second, we show undecidability of computing a strongest algebraic invariant if guards are introduced (Subsection 7.2).

7.1 Computing Algebraic Invariants

We compute strongest algebraic invariants for switching systems building on Proposition 5.

```
Procedure Semigroup-Closure($A_1, \dots, A_k$)
   input : $A_1, \dots, A_k \in \mathbb{A}^{d \times d}$
1  $H := \{I_d\}$
2  $G_1 := \overline{\{e^{A_1 t} : t \geq 0\}}; \dots ; G_k := \overline{\{e^{A_k t} : t \geq 0\}}$
3  repeat
4      $H_{old} := H$
5      for $i \in \{1, \dots, k\}$ do
6          $H := \overline{H \cdot G_i}$
7  until $H_{old} = H$
   output : $H$
```

▶ **Proposition 11.** *Algorithm Semigroup-Closure terminates and outputs the Zariski closure of the sub-semigroup of* $\mathrm{GL}_d(\mathbb{C})$ *generated by* $\{e^{A_1 t}, \dots, e^{A_k t} : t \geq 0\}$.

Proof. First note that the effectiveness of Line 2 relies on Proposition 5.

Now we argue that H in the algorithm is always an irreducible variety. For this it suffices to show that if $X \subseteq \mathrm{GL}_d(\mathbb{C})$ is an irreducible variety then so is $Y := \overline{X \cdot \overline{\{e^{At} : t \geq 0\}}}$ for any matrix $A \in \mathbb{C}^{d \times d}$. First observe that if X and Z are irreducible sets then so is $\overline{X \cdot Z}$, thus is it enough to show that $G = \overline{S}$ is irreducible, where $S = \{e^{At} : t \geq 0\}$. But S is a semigroup, thus G is a group. This makes G a linear algebraic group, which is therefore irreducible if and only if it is (Zariski-)connected. But G being the closure of S means it is enough to show that S is Zariski-connected, and hence enough to show that it is Euclidean-connected. The latter is trivial since every element of S is path-connected to I_d.

Now a strictly increasing chain $H_1 \subsetneq H_2 \subsetneq \cdots$ of irreducible sub-varieties of $\mathrm{GL}_d(\mathbb{C})$ has length at most the dimension of $\mathrm{GL}_d(\mathbb{C})$, which is d^2. Thus Algorithm Semigroup-Closure terminates after at most d^2 iterations of the outer loop. It is clear that the terminating value of the algorithm is the Zariski closure of the sub-semigroup of $\mathrm{GL}_d(\mathbb{C})$ generated by $\{e^{A_1 t}, \dots, e^{A_k t} : t \geq 0\}$. ◀

Now consider a switching system $\mathcal{A} = (Q, A, E, T)$. Let G be the Zariski closure of the semigroup generated by the matrices $e^{A_q t}$, for $q \in Q$ and $t \geq 0$, which can be computed by Proposition 11. Then $V(\mathcal{A}) = \{V_q : q \in Q\}$, the real Zariski closure of the collecting semantics of $\mathcal{A}$, is such that $V_q = \overline{G \cdot X} \cap \mathrm{GL}_d(\mathbb{R})$ for every $q \in Q$, where $X = \bigcup_{q \in Q} T_q$. But then $V(\mathcal{A})$ is computable from G and T.

7.2 Undecidability for Switching Systems with Guards

Finally, we show that computing the strongest algebraic inductive invariant becomes undecidable if we introduce guards to switching systems. Specifically, we consider switching systems with no discrete updates on the variables but with equality guards on the discrete mode changes. For such systems there is a smallest algebraic inductive invariant, which can be obtained as the (location-wise) intersection of the family of all algebraic inductive invariants. However, as we show in this section, this invariant is no longer computable. In other words, in the presence of equality guards the analog of Theorem 3 fails. (We remark also that the discrete mode changes no longer induce Zariski continuous maps on configurations if there are equality guards. Hence we cannot necessarily recover the smallest algebraic inductive invariant as the Zariski closure of the collecting semantics).

▶ **Theorem 12.** *There is no algorithm that computes the strongest algebraic inductive invariant for the class of switching systems with equality guards.*

Proof Sketch (see full proof in appendix). The idea is to simulate a 2-counter machine in such a way that if the machine has an infinite run then the strongest invariant has dimension 2, and otherwise it has dimension 1. Since the dimension of an algebraic set can be effectively determined; this concludes the sketch. ◀

References

1 Hirokazu Anai and Volker Weispfenning. Reach set computations using real quantifier elimination. In *Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001, Proceedings*, volume 2034 of *Lecture Notes in Computer Science*, pages 63–76. Springer, 2001.

2 Michele Boreale. Complete algorithms for algebraic strongest postconditions and weakest preconditions in polynomial odes. *Sci. Comput. Program.*, 193:102441, 2020.

3 H. Derksen, E. Jeandel, and P. Koiran. Quantum automata and algebraic groups. *J. Symb. Comput.*, 39(3-4):357–371, 2005.

4 Stephan Falke and Deepak Kapur. When is a formula a loop invariant? In *Logic, Rewriting, and Concurrency - Essays dedicated to José Meseguer on the Occasion of His 65th Birthday*, volume 9200 of *Lecture Notes in Computer Science*, pages 264–286. Springer, 2015.

5 Khalil Ghorbal and André Platzer. Characterizing algebraic invariants by differential radical invariants. In *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8413 of *Lecture Notes in Computer Science*, pages 279–294. Springer, 2014.

6 Thomas A. Henzinger. The theory of hybrid automata. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, July 27-30, 1996*, pages 278–292. IEEE Computer Society, 1996.

7 Ehud Hrushovski, Joël Ouaknine, Amaury Pouly, and James Worrell. Polynomial invariants for affine programs. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 530–539, 2018.

8 Z. Kincaid, J. Cyphert, J. Breck, and T. W. Reps. Non-linear reasoning for invariant synthesis. *PACMPL*, 2(POPL):54:1–54:33, 2018.

9 Gerardo Lafferriere, George J. Pappas, and Sergio Yovine. Symbolic reachability computation for families of linear vector fields. *J. Symb. Comput.*, 32(3):231–253, 2001. URL: `https://doi.org/10.1006/jsco.2001.0472`, `doi:10.1006/jsco.2001.0472`.

10 Daniel Liberzon. *Switching in Systems and Control.* Birkhauser/Springer, 2003.

11 D. W. Masser. Linear relations on algebraic groups. In *New Advances in Transcendence Theory.* Cambridge University Press, 1988.

12 Joël Ouaknine, Amaury Pouly, João Sousa Pinto, and James Worrell. Solvability of matrix-exponential equations. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 798–806, 2016.

13 Enric Rodríguez-Carbonell and Deepak Kapur. Automatic generation of polynomial invariants of bounded degree using abstract interpretation. *Sci. Comput. Program.*, 64(1):54–75, 2007. URL: `https://doi.org/10.1016/j.scico.2006.03.003`, `doi:10.1016/j.scico.2006.03.003`.

14 Enric Rodríguez-Carbonell and Deepak Kapur. Generating all polynomial invariants in simple loops. *J. Symb. Comput.*, 42(4):443–476, 2007. URL: `https://doi.org/10.1016/j.jsc.2007.01.002`, `doi:10.1016/j.jsc.2007.01.002`.

15 Enric Rodríguez-Carbonell and Ashish Tiwari. Generating polynomial invariants for hybrid systems. In Manfred Morari and Lothar Thiele, editors, *Hybrid Systems: Computation and Control*, pages 590–605, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

16 Sriram Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2010, Stockholm, Sweden, April 12-15, 2010*, pages 221–230. ACM, 2010.

17 Sriram Sankaranarayanan, Henny B. Sipma, and Zohar Manna. Constructing invariants for hybrid systems. *Formal Methods in System Design*, 32(1):25–55, 2008.

A Time Discretisation

▶ **Proposition 13.** *For a rational matrix $A \in \mathbb{Q}^{d\times d}$ we have*

$$\overline{\{e^{At} : t \in \mathbb{R}\}} = \overline{\langle e^A\rangle}\,.$$

Proof. Suppose that A is diagonalisable—say $A = U^{-1}DU$ for some invertible matrix U and $D = \operatorname{diag}(a_1, \dots, a_d)$. It suffices to prove that $\overline{\{e^{Dt} : t \in \mathbb{R}\}} = \overline{\langle e^D\rangle}$.

Consider a multiplicative relationship among the eigenvalues of e^D—say $(e^{a_1})^{n_1} \cdots (e^{a_d})^{n_d} = 1$, where $n_1, \dots, n_d \in \mathbb{Z}$. Then $a_1n_1 + \cdots + a_dn_d \in (2\pi i)\mathbb{Z}$. But since $a_1, \dots, a_d$ are algebraic numbers, we must in fact have $a_dn_d + \cdots + a_dn_d = 0$. It follows that $a_1tn_1 + \cdots + a_dtn_d = 0$ for all $t \in \mathbb{R}$ and hence $(e^{a_1t})^{n_1} \cdots (e^{a_dt})^{n_d} = 1$ for all $t \in \mathbb{R}$, i.e., the same multiplicative relation also holds among the eigenvalues of e^{Dt}.

Since the ideal of all polynomial relations satisfied by $\langle e^D\rangle$ is generated by the multiplicative relations satisfied by the eigenvalues of e^D, we have that for any $t \in \mathbb{R}$, matrix e^{Dt} satisfies all polynomial relations satisfied by $\langle e^D\rangle$. This proves the proposition in case A is diagonalisable.

Next, suppose that A is nilpotent. The fact that $\overline{\{e^{At} : t \in \mathbb{R}\}} = \overline{\langle e^A\rangle}$ is already shown in Section 3.3 of Derksen, Jeandel, and Koiran.

The general case can by handled by reduction to the diagonalisable and nilpotent cases as in Proposition 5. ◀

Proposition 13 crucially relies on the fact that π does not appear in the description of A. Indeed, consider the case that $A = \big(2\pi i\big) \in \mathbb{C}^{1\times 1}$. Then $\{e^{At} : t \in \mathbb{R}\} = \{z \in \mathbb{C} : |z| = 1\}$ is the unit circle. However $\{e^{An} : n \in \mathbb{Z}\} = \{1\}$ is a singleton. Such an example is possible because the map $z \in \mathbb{C} \mapsto e^{Az}$ is not Zariski-continuous in general.

B Examples

We explain how to discretise the hybrid system in Section 2.1 into a corresponding affine program. The first step is to get rid of the continuous dynamics entirely. Let $X = (x, y, v_x, v_y, t, 1)$ be the state, where we add 1 to make it linear, then the continuous behaviour can be rewritten as $\dot{X} = AX$. Intuitively, we can replace the continuous dynamics by infinitely many discrete transitions e^{At} for $t \geqslant 0$. The key observation (Proposition 13) is that we can in fact replace this infinite set with just one matrix, e^A, without changing the smallest algebraic invariant of the system. The strongest algebraic invariant is then obtained by our previous result on affine programs (Theorem 2). In this example, we have

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & -g\\ 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad e^A = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 1 & 0 & -\frac{1}{2}g\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & -g\\ 0 & 0 & 0 & 0 & 1 & 1\\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

We can now rephrase the discrete transition $X := e^A X$ as $x := x + v_x$, $y := y + v_y - \frac{1}{2}g$, $v_y := v_y - g$ and $t := t + 1$. By construction, this exactly corresponds to a time-discretisation with 1 unit of time. We then obtain the equivalent, for algebraic invariant, affine program depicted in Figure 4a. For instance, one can check that (1) is indeed invariant under this new transition ($x', y', \ldots$ denotes the value after the transition):

$$\begin{aligned} x' - t'c &= x + v_x - (t+1)c = x - tc - c + v_x = 0 \quad \text{since } x = tc \text{ and } v_x = c, \\ {v'_y}^2 + 2g(y' - h) &= (v_y - g)^2 + 2g(y + v_y - \tfrac{1}{2}g - h) \\ &= v_y^2 + 2g(y - h) = 0. \end{aligned}$$

We next describe how to use the procedure in Section 6 to transform the hybrid system shown in Figure 2b to the equivalent (with respect to algebraic invariants) affine program in Figure 4b. Let $X = (I, I_R, V_R, Q, V_C)$ be the state, then the continuous behaviour in location OPEN (resp. CLOSED) can be rewritten as $\dot{X} = AX$ (resp. $\dot{X} = BX$). We proceed as in the previous example and replace the continuous dynamics by two discrete transitions e^A and e^B. Unfortunately in this case, the resulting matrices

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{RC} & 0 & 0 & 0 \\ 0 & -\frac{1}{C} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{1}{C} & 0 & 0 & 0 \end{bmatrix}, \qquad e^A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & e^{-\frac{1}{RC}} & 0 & 0 & 0 \\ 0 & (e^{-\frac{1}{RC}} - 1)R & 1 & 0 & 0 \\ 0 & (1 - e^{-\frac{1}{RC}})RC & 0 & 1 & 0 \\ 0 & (1 - e^{-\frac{1}{RC}})R & 0 & 0 & 1 \end{bmatrix}$$

have non-algebraic entries in general. Indeed, since RC is algebraic, $e^{-\frac{1}{RC}}$ is never algebraic and this prevents us from computing the strongest algebraic invariant using Theorem 2. We circumvent this issue by constructing another matrix, call it $\hat{A}$, with algebraic coefficients such that if we replace e^A by $\hat{A}$ then the two affine programs have the same algebraic invariants. We show in Section 6 how to compute such a matrix, which is this example produces

$$\hat{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & R & 1 & 0 & 0 \\ 0 & -RC & 0 & 1 & 0 \\ 0 & -R & 0 & 0 & 1 \end{bmatrix}, \qquad \hat{B} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & R & 1 & 0 & 0 \\ 0 & -RC & 0 & 1 & 0 \\ 0 & -R & 0 & 0 & 1 \end{bmatrix}.$$

We then obtain the equivalent, for algebraic invariant, affine program depicted in Figure 4b. For instance, one can check that the OPEN invariant is indeed invariant by $\hat{A}$:

$$\begin{aligned} Q' - CV'_C &= (-RCI_R + Q) - C(-RI_R + V_C) = Q - CV_C = 0 \\ V'_R - RI'_R &= (RI_R + V_R) - R(2I_R) = V_R - RI_R = 0 \\ V'_R + V'_C &= (RI_R + V_R) + (-RI_R + V_C) = V_R + V_C = 0. \end{aligned}$$

C Proof of Theorem 12

Recall that a non-deterministic 2-counter machine M consists of two counters C and D and a list of n instructions. Each instruction increments one of the counters, decrements one of the counters, or tests one of the counters for zero. After executing a counter update or a successful test, the machine proceeds nondeterministically to one of two specified instructions. The machine halts if it executes a test instruction whose condition is false. Given an instruction

i, if j is one of the two possible successors of i then we call the pair (i, j) a *transition* of M. Initially M starts with both counters zero and instruction 1 is the first to be executed. A configuration of M is a triple consisting of the current instruction and the current counter values. The problem of whether such a machine M can reach infinitely many configurations from its initial configuration is undecidable.

Corresponding to such a 2-counter machine M we define a linear hybrid automaton $\mathcal{A} = (Q, A, E, q)$ in dimension 3. We think of $\mathcal{A}$ as having continuous variables c, d, t, where c and d respectively correspond to the counters of M. Each variable has constant derivative in each location, which is zero unless otherwise specified. For each instruction i of M we postulate a location q_i of $\mathcal{A}$ and for each transition (i, j) of M we postulate a location $q_{i,j}$ of $\mathcal{A}$. Variable t has slope 1 in each location q_i and slope -1 in each location $q_{i,j}$. For every transition (i, j) of M, automaton $\mathcal{A}$ has an edge from q_i to $q_{i,j}$ with guard $t = 1$ and an edge from $q_{i,j}$ to q_j with guard $t = 0$. Intuitively, if an execution of $\mathcal{A}$ correctly simulates a run of M then $\mathcal{A}$ spends one time unit in each location, alternating between locations q_i that correspond to instructions of M and locations $q_{i,j}$ that correspond to transitions of M.

Suppose that the i-th instruction of M performs an incrementation $C := C + 1$. Then variable C has slope 1 in location q_i. Likewise if the i-th instruction if $C := C - 1$, then variable c has slope -1 in location q_i. If the i-th instruction of M is the zero test $C = 0$, then the edge from location q_i to $q_{i,j}$ in A has guard $c = 0$. There are corresponding constructions for counter operations and tests on counter D.

This completes the description of $\mathcal{A}$. We now claim that:

1. If M can only reach finitely many configurations from the initial configuration then $V(\mathcal{A}) = \{V_q : q \in Q\}$, the Zariski closure of the collecting semantics, is an inductive invariant that has dimension one.
2. If infinitely many configurations are reachable from the initial configuration of M then the smallest inductive invariant has dimension strictly greater than one.

To prove the claim, note that for each reachable configuration (i, z_1, z_2) of M, the collecting semantics $\Phi(\mathcal{A})_{q_i}$ contains a half-line L containing the point $(z_1, z_2, 0)$, whose direction is determined by the slopes of the respective variables of $\mathcal{A}$ in location q_i. The Zariski closure of L is the affine hull of L, i.e., the corresponding full line containing L. Crucially, the points added to L to obtain the full line are all predecessors of L under the flow relation of $\mathcal{A}$. In particular the Zariski closure is inductive: it remains closed under the transition relation of $\mathcal{A}$. In particular, if M can only reach finitely many configurations, then the Zariski closure of the collecting semantics consists of finitely many lines in each location (and so has dimension one) and is moreover an inductive invariant.

Suppose M can reach infinitely many configurations. Since any algebraic inductive invariant must in particular contain the Zariski closure of the collecting semantics, it follows that any algebraic inductive invariant must contain infinitely many lines in some location and thus must have dimension strictly greater than one.

Since the dimension of an algebraic set can be effectively determined, we conclude that it is not possible to compute the smallest algebraic invariant of a linear hybrid automaton with equality guards (even with a no discrete updates of the variables).