Generating Invariants of Hybrid Systems via Sums-Of-Squares of Polynomials with Rational Coefficients *

Min Wu and Zhengfeng Yang
Shanghai Key Laboratory of Trustworthy Computing
East China Normal University, Shanghai 200062, China
{mwu,zfyang}@sei.ecnu.edu.cn

ABSTRACT

In this paper we discuss how to generate inequality invariants for continuous dynamical systems involved in hybrid systems. A hybrid symbolic-numeric algorithm is presented to compute inequality invariants of the given systems, by transforming this problem into a parameterized polynomial optimization problem. A numerical inequality invariant of the given system can be obtained by applying polynomial Sum-of-Squares (SOS) relaxation via Semidefinite Programming (SDP). And a method based on Gauss-Newton refinement is deployed to obtain candidates of polynomials with rational coefficients, and finally we certify that this polynomial *exactly* satisfies the conditions of invariants, by use of SOS representation of polynomials with rational coefficients.

Several examples are given to show that our algorithm can successfully yield inequality invariants with rational coefficients.

Categories and Subject Descriptors: I.2.1 [Computing Methodologies]: Symbolic and Algebraic Manipulation —Algorithms; G.1.2 [Mathematics of Computing]: Numerical Analysis—Approximation

General Terms: algorithms, verification

Keywords: semidefinite programming, sum-of-squares relaxation, program verification, differential invariant

1. INTRODUCTION

Complex physical systems are systems in which the techniques of sensing, control, communication and coordination are involved and interacted with each other. Among complex physical systems, many of them are safety critical systems, such as airplanes, railway, and automotive applications. Due to the complexity, ensuring correct functioning of these systems, e.g., spatial separation, especially collision avoidance, of aircrafts during the entire flights, is among the most challenging and most important problems in computer science, mathematics and engineering.

As a common mathematical model for complex physical systems, hybrid systems [13, 3] are dynamical systems that are governed by interacting discrete and continuous dynamics [1, 13, 10]. Continuous dynamics is specified by differential equations, which is possibly subject to domain restrictions or algebraic relations resulting from physical circumstances or the interaction of continuous dynamics with discrete control. For discrete transitions, the hybrid system changes state instantaneously and possibly discontinuously, for example, the instantaneous change of control variables like the acceleration (e.g., the changing of a by setting $a := -b$ with braking force $b > 0$).

The analysis of hybrid systems is an important problem that has been studied extensively both by the control theory, and the formal verification community for over a decade. Among the most important analysis questions for hybrid systems are those of *safety*, i.e., deciding whether a given property ψ holds in all the reachable states, and the dual problem of *reachability*, i.e., deciding if a state satisfying the given property ψ is reachable. Both these problems are closely related to the problem of generating invariants of hybrid systems.

An invariant [31] of a hybrid system is a property that holds in all the reachable states of the system, or, in other words, an over-approximation of all the reachable states of the system. Invariants are useful facts about the dynamics of a given system and are widely used in numerous approaches to verify and understand systems. The invariants of hybrid systems are used to establish temporal properties of systems such as safety, stability, termination, progress and so on [19, 7, 28, 23].

The problem of generating invariants of an arbitrary form is known to be computationally hard, intractable even for the simplest classes. The usual technique for generating invariants is to generate an *inductive invariant*, i.e., an assertion that holds at the initial states of the system, and is preserved by all discrete and continuous state changes. There has been a considerable volume of work towards invariant generation for hybrid systems using techniques from convex optimization, commutative algebraic and semi-algebraic geometry [36, 4, 32, 21, 22, 33, 23, 12, 33, 26, 24, 20, 27, 31]. Many of these techniques synthesize invariants of prespecified form by computing constraints on the unknown co-

*Min Wu is supported by the Chinese National Natural Science Foundation under Grants 10801052 and 91018012.
Zhengfeng Yang is supported by the Chinese National Natural Science Foundation under Grants 10901055 and 91018012, the China National 973 Project(No.2011CB302802), and Shanghai Natural Science Foundation under Grant 09ZR1408800.

efficients of a single or fixed number of polynomial inequalities with a bounded degree, that guarantee that any solution will also be inductive. Carbonell and Tiwari [4] presented a powerful computational method for automatically generating polynomial invariants, which does not assume bounded degree, whereas their method is only applicable to linear hybrid systems. In [2], Asarin et al. employed a numerical approach based on conservative approximation to deal with nonlinear systems. However, as pointed out in [25], numerical errors resulted from numerical computation tended to cause unsoundness of the verification of hybrid systems. Recently, Platzer et al. proposed a powerful theorem-proving framework [21, 22, 23, 26, 24, 27] for verifying properties of hybrid systems through differential logic that extends dynamic logics using differential operators and quantifier elimination [6]. However, it is well known that the bottleneck of quantifier elimination algorithms lies in their high complexity, which is doubly exponential.

In this work, we study how to generate inequality invariants for nonlinear continuous systems given by polynomial vector fields. In virtue of the efficiency of numerical computation and the error-free property of symbolic computation, we present a symbolic-numeric hybrid method, based on Sum-of-Squares (SOS) relaxation via semidefinite programming (SDP) and exact SOS representation recovery, to generate inequality invariants of continuous systems. More specifically, computing inequality invariants of a continuous system is transformed into solving a semi-algebraic system with parameters, while the latter system is constructed by predetermining the template of the polynomial invariant with the given degree. It is noticed that SDP is applicable to polynomial optimization problem by use of Sum-of-Squares relaxation. Since Matlab packages, such as *SOSTOOLS* [29] and *Yalmip* [18], that deal with SDP problem, are running in the fixed precision, these algorithms yield only numerical solutions. However, numerical invariants may not be the correct invariants of the given continuous system. In [14, 11], Kaltofen et al. provided a symbolic-numeric hybrid method, combined with the SDP solvers and rational vector recovery techniques, to verify whether a multivariate polynomial is positive semidefinite by demonstrating the corresponding exact SOS representation. In this paper, we also deploy this symbolic-numeric method to compute inequality-form invariants of the given continuous systems. The idea is as follows: (a) From the conditions of invariants, we construct an SDP system to solve the associated semi-algebraic system with parameters; (b) an exact invariant is obtained by recovering the exact SOS representation from the approximate SOS representation yielded from the SDP solvers. During the process of recovery step, we need apply Gauss-Newton iteration to refine the approximate SOS representation. More details can be found in Section 3.

In comparison with other symbolic approaches of invariant generation based on qualifier elimination theory, our approach is more efficient and practical, because the SDP systems constructed from the continuous system can be solved in polynomial time. Furthermore, we apply Gauss-Newton refinement technique and rational vector recovery to obtain an exact invariant of the given system. Having the exact representation of polynomials as sums of squares with rational coefficients, the invariant obtained by our approach can be easily verified to satisfy its constraints *exactly*. Therefore, our approach is able to overcome the weakness of numerical approaches on approximate invariant computation, as claimed in [25], .

The rest of the paper is organized as follows. In Section 2, we introduce the notions of hybrid systems and invariants. Section 3 is devoted to illustrating a symbolic-numeric approach for generating invariants of continuous systems. In Section 4, we show two examples of inequality invariant generation. Section 5 concludes the paper.

2. INVARIANTS

To model hybrid systems, we use hybrid automata [13, 33].

Definition 1 (Hybrid Systems). *A* hybrid system $\mathbf{H}$: $\langle V, L, \mathcal{T}, \Theta, \mathcal{D}, \psi, \ell_0\rangle$ *consists of the following components:*

- V*, a set of real-valued system* variables. *A* state *is an interpretation of* V*, assigning to each* $x \in V$ *a real value. An* assertion *is a first-order formula over* V*. A state* s *satisfies an assertion* φ*, written as* $s \models \varphi$*, if* φ *holds on* s*. We will also write* $\varphi_1 \models \varphi_2$ *for two assertions* φ_1, φ_2 *to denote that* φ_2 *is true at least in all the states in which* φ_1 *is true;*
- L*, a finite set of locations;*
- $\mathcal{T}$*, a set of (discrete) transitions. Each transition* $\tau : \langle \ell_1, \ell_2, \rho_\tau\rangle \in \mathcal{T}$ *consists of a prelocation* $\ell_1 \in L$*, a postlocation* $\ell_2 \in L$*, and an assertion* ρ_τ *over* $V \cup V'$ *representing the next-state relation, where* V' *denotes the values of* V *in the next state;*
- Θ*, an assertion specifying the* initial *condition;*
- $\mathcal{D}$*, a map that maps each location* $\ell \in L$ *to a* differential rule *(also known as a* vector field *or a* flow field*)* $\mathcal{D}(\ell)$*, of the form* $\dot{x}_i = \frac{dx_i}{dt} = f_i(V)$ *for each* $x_i \in V$*, with* t *the time variable. The differential rule at a location specifies how the system variables evolve in that location;*
- ψ*, a map that maps each location* $\ell \in L$ *to a* location condition (location invariant) $\psi(\ell)$*, an assertion over* V*;*
- $\ell_0 \in L$*, the* initial location*; we assume that the initial condition satisfies the location invariant at the initial location, that is* $\Theta \models \psi(\ell_0)$*.*

The following definition of invariants for hybrid systems comes from [33].

Definition 2 (Invariant). *[33, Definition 3] An* invariant $\mathcal{I}$ *of a hybrid system at a location* ℓ *is an assertion* $\mathcal{I}$ *such that for any reachable state* $\langle \ell, \mathbf{x}\rangle$ *of the hybrid system,* $\mathbf{x} \models \mathcal{I}$*.*

The problem of generating invariants of arbitrary form is known to be computationally hard, intractable even for the simplest classes. The usual technique for generating invariants is to generate *inductive invariants*, which are defined as follows.

Definition 3 (Inductive Invariant). *An* inductive assertion map $\mathcal{I}$ *of a hybrid system* $\mathbf{H} : \langle V, L, \mathcal{T}, \Theta, \mathcal{D}, \psi, \ell_0\rangle$ *is a map that associates with each location* $\ell \in L$ *an assertion* $\mathcal{I}(\ell)$ *that holds initially and is preserved by all discrete* *transitions and continuous flows of* $\mathbf{H}$. *More formally an inductive assertion map satisfies the following requirements:*

(i) **[Initial]** $\Theta \models \mathcal{I}(\ell_0)$.

(ii) **[Discrete Consecution]** *For each discrete transition* $\tau : \langle \ell_1, \ell_2, \rho_\tau \rangle$, *starting from a state satisfying* $\mathcal{I}(\ell_1)$, *and taking* τ *leads to a state satisfying* $\mathcal{I}(\ell_2)$. *Formally,*

$$\mathcal{I}(\ell_1) \wedge \rho_\tau \models \mathcal{I}(\ell_2)',$$

where $\mathcal{I}(\ell_2)'$ *represents the assertion* $\mathcal{I}(\ell_2)$ *with the current state variables* $x_1, \ldots, x_n$ *replaced by the next state variables* $x_1', \ldots, x_n'$, *respectively.*

(iii) **[Continuous Consecution]** *For every location* $\ell \in L$ *and states* $\langle \ell, \mathbf{x}_1 \rangle, \langle \ell, \mathbf{x}_2 \rangle$ *such that* $\mathbf{x}_2$ *evolves from* $\mathbf{x}_1$ *according to the differential rule* $\mathcal{D}(\ell)$ *at* ℓ, *if* $\mathbf{x}_1 \models \mathcal{I}(\ell)$ *then* $\mathbf{x}_2 \models \mathcal{I}(\ell)$.

Remark 1. Clearly, Definition 3 generalizes the notions of invariants for both discrete and continuous dynamical systems. An assertion $\mathcal{I}$ is called a *discrete (inductive) invariant* if $\mathcal{I}$ satisfies the conditions (i) and (ii); and $\mathcal{I}$ is called a *differential invariant* if $\mathcal{I}$ satisfies the conditions (i) and (iii).

The usual way to compute invariants of hybrid systems is to compute discrete invariants and differential invariants separately. A lot of techniques are available for computing invariants of discrete loop programs. The reader can refer to [37, 8, 15, 9, 34, 30, 16, 5] for more details.

In this work, we only concentrate on how to compute differential invariants of nonlinear continuous dynamical systems given by polynomial vector fields, i.e., systems of the form

$$\dot{x}_1 = p_1(x_1, \ldots, x_n), \quad \cdots, \quad \dot{x}_n = p_n(x_1, \ldots, x_n) \tag{1}$$

where $p_i \in \mathbb{R}[x_1, \ldots, x_n]$ for $1 \le i \le n$.

[**Convention**] In what follows, we will adopt the following notations:

(i) The boldfaced symbols $\mathbf{x}$ and $\mathbf{p}$ denote the vectors

$$(x_1, \ldots, x_n) \quad \text{and} \quad (p_1, \ldots, p_n),$$

respectively;

(ii) For a smooth function $V(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$, the notation $\mathbf{d}V$ denotes the $1 \times n$ row vector $(\frac{\partial V}{\partial x_1}, \ldots, \frac{\partial V}{\partial x_n})$ of partial derivatives of V with respect to the variables $x_1, \ldots, x_n$;

(iii) The expression $\mathbf{d}V \cdot \mathbf{p}$ represents the inner product of the vectors $\mathbf{d}V$ and $\mathbf{p}$. Actually we have $\mathbf{d}V \cdot \mathbf{p} = \frac{dV}{dt}$.

We will study how to generate inequality-form invariants of the system $\dot{\mathbf{x}} = \mathbf{p}$, i.e., invariants of the form $F \ge 0$ where F is an expression in the system variables $x_1, \ldots, x_n$ and the initial values $\mathbf{x}(0)$. The following lemma will be needed in latter discussion.

Lemma 1. *Let* $\dot{\mathbf{x}} = \mathbf{p}$ *be a nonlinear dynamical system with initial states* Θ *and state invariants* ψ. *For a polynomial* $V(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, *define*

$$d_0 = \min\{V(\mathbf{x}(0)) : \mathbf{x}(0) \in \Theta\},$$

and set $r(\mathbf{x}) = \frac{dV}{dt} = \mathbf{d}V \cdot \mathbf{p}$. *If the polynomial* $r(\mathbf{x})$ *is positive semidefinite within the state invariants* ψ, *meaning that* $r(\mathbf{x})$ *is positive semidefinite for all* $\mathbf{x}$ *such that* ψ *hold, then the formula* $V(\mathbf{x}) - d_0 \ge 0$ *is a differential invariant of the dynamical system* $\dot{\mathbf{x}} = \mathbf{p}$.

Remark 2. In order to compute the differential invariant $V(\mathbf{x}) - d_0 \ge 0$ as in Lemma 1, we need compute both the polynomial $V(\mathbf{x})$ and the constant d_0. Actually, once $V(\mathbf{x})$ is obtained, there are several approaches to obtain the minimum d_0 of $V(\mathbf{x})$ in Θ. Hereafter, we will omit the discussion of d_0, and focus on computing polynomials $V(\mathbf{x})$ such that $r(\mathbf{x}) = \mathbf{d}V \cdot \mathbf{p}$ is positive semidefinite within the state invariants ψ. For simplicity, in the sequel, by saying $V(\mathbf{x})$ is a polynomial invariant we mean that the polynomial inequality $V(\mathbf{x}) - d_0 \ge 0$ is an invariant of the given nonlinear dynamical system, where $d_0 = \min\{V(\mathbf{x}(0)) : \mathbf{x}(0) \in \Theta\}$.

3. POLYNOMIAL INEQUALITY INVARIANT GENERATION

According to Lemma 1 and Remark 2, in order to compute invariants of a nonlinear system $\dot{\mathbf{x}} = \mathbf{p}$ with the state invariants ψ, one may obtain a polynomial $V(\mathbf{x})$ such that $r(\mathbf{x}) = \mathbf{d}V \cdot \mathbf{p}$ is positive semidefinite within ψ. In general, we consider the state invariants ψ with the following form:

$$\psi_1 \ge 0 \wedge \cdots \wedge \psi_k \ge 0, \tag{2}$$

where $\psi_1, \cdots, \psi_k \in \mathbb{R}[\mathbf{x}]$. The problem of computing polynomial invariants can be transformed into the following problem

$$\begin{array}{ll} \text{find} & V \in \mathbb{R}[\mathbf{x}] \\ \text{s.t.} & r = \mathbf{d}V \cdot \mathbf{p} \ge 0 \text{ if } \psi_1 \ge 0 \wedge \cdots \wedge \psi_k \ge 0. \end{array} \tag{3}$$

Our idea of computing $V(\mathbf{x})$, based on SOS relaxation and rational vector recovery, is as follows.

Step 1: Predetermine a template of polynomial invariants with the given degree and convert the problem of computing polynomial invariants to the associated (parametric) polynomial optimization problem. SOS relaxation method is then applied to obtain an approximate polynomial invariant with floating point coefficients.

Step 2: Apply Gauss-Newton refinement and rational vector recovery on the approximate polynomial invariant to get one polynomial with rational coefficients, which satisfies exactly the conditions of invariants of the given dynamical system.

Steps 1 and 2 will be explained in more details in Sections 3.1 and 3.2, respectively.

3.1 Sum of Squares Relaxation

To solve the problem (3), let us predetermine a template of polynomial invariants with the given degree d, that is, we assume

$$V(\mathbf{x}) = \sum_{\alpha} \mathbf{c}_\alpha \mathbf{x}^\alpha, \tag{4}$$

where $\mathbf{c}_\alpha \in \mathbb{R}$ are parameters, $\mathbf{x}^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ and $\alpha \in \mathbb{Z}^n_{\ge 0}$ with $\sum_{i=1}^n \alpha_i \le d$.

We first consider a simpler case where there exists no state invariants. The problem then becomes computing a polynomial $V \in \mathbb{R}[\mathbf{x}]$ such that

$$r(\mathbf{x}) = \mathbf{d}V \cdot \mathbf{p} \geq 0, \qquad \forall\, \mathbf{x} \in \mathbb{R}^n.$$

It is obvious that a sufficient condition for $r(\mathbf{x})$ to be positive semidefinite for arbitrary value $\mathbf{x} \in \mathbb{R}^n$ is that there exists an SOS decomposition of $r(\mathbf{x})$:

$$r(\mathbf{x}) = \sum_i {f_i}^2(\mathbf{x}), \quad \text{with } f_i(\mathbf{x}) \in \mathbb{R}[\mathbf{x}], \tag{5}$$

or, equivalently, there exists a following representation of $r(\mathbf{x})$:

$$r(\mathbf{x}) = m(\mathbf{x})^T \cdot W \cdot m(\mathbf{x}),$$

where W is a real symmetric and positive semidefinite matrix and $m(\mathbf{x})$ is a vector of terms in $\mathbb{R}[\mathbf{x}]$ with degree $\leq d/2$. The SOS problem (5) can be further converted into the following SDP program

$$\left.\begin{array}{ll} \inf\limits_{W} & \mathrm{Trace}(W) \\ \text{s. t.} & r(\mathbf{x}) = m(\mathbf{x})^T \cdot W \cdot m(\mathbf{x}) \\ & W \succeq 0, W^T = W. \end{array}\right\} \tag{6}$$

(here $\mathrm{Trace}(W)$ acts as a dummy objective function that is commonly used in SDP for optimization problem with no objective functions.)

Now let us consider the case where the state invariants are given as in (2). Similar to the case with no state invariants, we predetermine the template (4) of $V(\mathbf{x})$ with the given degree d, where the coefficients c_α are parameters. One can certainly apply quantifier elimination methods to solve the corresponding parametric semi-algebraic system. Some Maple packages such as DISCOVERER [38] are available to solve this kind of problems. For the given template, quantifier elimination methods yield the necessary and sufficient conditions for the existence of invariants with given degree. However, quantifier elimination methods are of high complexity since they rely on the cylindrical algebraic decomposition (CAD) algorithm. Instead in this paper, we will explore the SOS relaxation techniques based on semidefinite programming to obtain polynomial invariants. These techniques supply a sufficient condition for the existence of $V(\mathbf{x})$.

Theorem 1. *Let $\dot{\mathbf{x}} = \mathbf{p}$ be a nonlinear dynamical system with initial states Θ and state invariants*

$$\psi := \psi_1 \geq 0 \wedge \cdots \wedge \psi_k \geq 0.$$

Suppose there exists a polynomial $V(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ such that $r(\mathbf{x}) = \mathbf{d}V \cdot \mathbf{p}$ can be written as

$$r(\mathbf{x}) = \sum_{i=0}^{k} \sigma_i \psi_i, \tag{7}$$
$$\text{where } \sigma_0, \ldots, \sigma_k \text{ are SOSs and } \psi_0 = 1,$$

then the formula $V(\mathbf{x}) - d_0 \geq 0$ with

$$d_0 = \min\{V(\mathbf{x}(0)) : \mathbf{x}(0) \in \Theta\},$$

is a differential invariant of the dynamical system $\dot{\mathbf{x}} = \mathbf{p}$.

Proof. It is easy to verify that if $r(\mathbf{x})$ has the form (7) then $r(\mathbf{x}) \geq 0$ holds for any $\mathbf{x} \in \mathbb{R}^n$ such that ψ holds. The conclusion then follows from Lemma 1. □

A weaker sufficient condition than (7) can be presented using cross products of ψ_i's, as described in the following corollary:

Corollary 1. *Under the assumptions of Theorem 1, if the polynomial $r(\mathbf{x})$ can be written as*

$$r(\mathbf{x}) = \sum_{\mu \in \{0,1\}^k} \sigma_\mu \psi_\mu \quad \text{where } \psi_\mu = \psi_1^{\mu_1} \cdots \psi_k^{\mu_k} \tag{8}$$
$$\sigma_\mu \text{ are all SOSs},$$

then the formula $V(\mathbf{x}) - d_0 \geq 0$ with

$$d_0 = \min\{V(\mathbf{x}(0)) : \mathbf{x}(0) \in \Theta\},$$

is a different invariant of the given dynamical system.

Given the degree bound $2N$ of $\sigma_i(\mathbf{x})$ with $N \in \mathbb{Z}^+$, one can easily construct an SOS program of the form (7), in which the decision variables are the coefficients of $V(\mathbf{x})$ and $\sigma_i(\mathbf{x})$, for $0 \leq i \leq k$. Similar to (6), the SOS program (7) is equivalent to the following SDP problem with a block form:

$$\left.\begin{array}{ll} \inf\limits_{W^{[0]},\ldots,W^{[k]}} & \sum_{i=0}^{k} \mathrm{Trace}(W^{[i]}) \\ \text{s. t.} & r(\mathbf{x}) = \sum_{i=0}^{k} m_i(\mathbf{x})^T \cdot W^{[i]} \cdot m_i(\mathbf{x}) \psi_i \\ & W^{[i]} \succeq 0, (W^{[i]})^T = W^{[i]}, 0 \leq i \leq k. \end{array}\right\} \tag{9}$$

The SOS program corresponding to (8) would be obtained likewise. Many Matlab packages of SDP solvers, such as SOSTOOLS [29], YALMIP [18], and SeDuMi [35], are available to solve efficiently the problems (6) and (9).

3.2 Exact Certificate of Sum of Squares Decomposition

Since the SDP solvers in Matlab is running in fixed precision, the techniques in Section 3.1 will yield numerical solutions to the associated SDP problems of (5, 7, 8), and therefore yield numerical solutions to (5, 7, 8), respectively. For instance, applying an SDP solver to (9), we obtain a numerical polynomial $V^*(\mathbf{x})$ with floating point coefficients and some numerical positive semidefinite matrices $W^{[i]}$, which satisfy approximately

$$r^*(\mathbf{x}) \approx \sum_{i=0}^{k} m_i(\mathbf{x})^T \cdot W^{[i]} \cdot m(\mathbf{x}) \psi_i \quad \text{and } W^{[i]} \succsim 0, \tag{10}$$

where $r^*(\mathbf{x}) = \mathbf{d}V^*(\mathbf{x}) \cdot \mathbf{p}$. However, due to round-off errors, $V^*(\mathbf{x}) - d_0 \geq 0$ may not necessarily be an invariant of the given dynamical system because $r^*(\mathbf{x})$ may not be positive definite within ψ exactly. Therefore in the next step, from the numerical polynomial $V^*(\mathbf{x})$ and the numerical positive semidefinite matrices $W^{[i]}$, we will try to recover a polynomial $V(\mathbf{x})$ with rational coefficients, which satisfies (7) *exactly*.

In [14, 11], the authors proposed a method to certify a lower bound of global optimum of a polynomial via SOS representation. In this paper, we will employ this method to compute invariants of continuous dynamical systems. We first recall the idea of this method. Given a multivariate polynomial $f \in \mathbb{R}[\mathbf{x}]$, they first constructed the associated SOS program and solve the SDP system to obtain an approximate lower bound r^* and an approximate positive semidefinite matrix W such that

$$f(\mathbf{x}) - r^* \approx m(\mathbf{x})^T \cdot W \cdot m(\mathbf{x}) \quad \text{with} \quad W \succsim 0.$$

Next, they rounded r^* to a nearby rational number r and, for a given tolerance τ, applied Gauss-Newton iteration to refine W such that

$$\|f(\mathbf{x}) - r - m(\mathbf{x})^T \cdot W \cdot m(\mathbf{x})\| \le \tau \text{ and } W \succeq 0.$$

In the end, a positive semidefinite matrix $\widetilde{W}$ with rational entries would be obtained by orthogonal projection method or rational vector recovery method such that

$$f(\mathbf{x}) - r = m(\mathbf{x})^T \cdot \widetilde{W} \cdot m(\mathbf{x}), \quad \text{with } \widetilde{W} \succeq 0. \tag{11}$$

Therefore, r is a certified lower bound of the global optimum of f because r and $\widetilde{W}$ exactly satisfy (11).

In comparison with the problem of the lower bound verification in [14], finding a polynomial invariant $V(\mathbf{x})$ of nonlinear continuous dynamical systems is a bit different. To obtain a certified polynomial invariant $V(\mathbf{x})$, we need consider the following two problems:

P.1: Instead of obtaining a single rational number r, we need find a rational vector $\mathbf{v}$ denoting the coefficient vector of $V(\mathbf{x})$.

P.2: $V(\mathbf{x})$ should exactly satisfy (7) or (8), i.e., not only the σ_i's and σ_μ's satisfy the equalities (7) and (8) respectively, but also they have exact SOS representations simultaneously.

In Section 3.1, a numerical vector $\mathbf{v}^*$ that denotes the (numerical) coefficient vector of $V^*(\mathbf{x})$ is obtained by solving some SDP system. For the given bound of the common denominator of $\mathbf{v}^*$, we can recover the numerical vector $\mathbf{v}^*$ to a vector $\mathbf{v}$ with rational entries by a simultaneous Diophantine approximation algorithm [17]. This solves the first problem.

Let us focus on the second problem. Here we only consider the case (7), and the case (8) can be handled likewise. From (9), we will construct another SDP system given by one big matrix $W = \mathrm{diag}(\mathrm{W}^{[0]}, \ldots, \mathrm{W}^{[k]})$, and, by use of SDP solvers, some numerical positive semidefinite matrices $W^{[0]}, \ldots, W^{[k]}$ and an approximate form of (9) will be obtained. The recovery of $W^{[0]}, \ldots, W^{[k]}$ into rational positive semidefinite matrices is split into two steps: we first recover $\widetilde{W}^{[1]}, \ldots, \widetilde{W}^{[k]}$ and then recover $\widetilde{W}^{[0]}$.

Step 1. Given the numerical positive definite matrices $W^{[i]}$, $1 \le i \le k$, we find the nearby rational positive semidefinite matrices $\widetilde{W}^{[i]}$. In practice, all $W^{[i]}$ are very simple, and by setting the small entries of $W^{[i]}$ to be zeros we easily get the nearby rational positive semidefinite matrices $\widetilde{W}^{[i]}$ for $i = 1, \ldots, k$.

Step 2. Having $\mathbf{v}$ and $\widetilde{W}^{[1]}, \ldots, \widetilde{W}^{[k]}$, (9) is converted to

$$\left.\begin{array}{r} r(\mathbf{x}) - \sum_{i=1}^{k} m_i(\mathbf{x})^T \cdot \widetilde{W}^{[i]} \cdot m_i(\mathbf{x})\psi_i \\ \approx m_0(\mathbf{x})^T \cdot W^{[0]} \cdot m_0(\mathbf{x}), \\ W^{[i]} \succeq 0,\ 1 \le i \le k,\ W^{[0]} \underset{\approx}{\succ} 0, \end{array}\right\} \tag{12}$$

where $r(\mathbf{x}) = \mathbf{d}V(\mathbf{x}) \cdot \mathbf{p}$ with $V(\mathbf{x})$ the polynomial corresponding to the coefficient vector $\mathbf{v}$. Observing in (12), the matrix $W^{[0]}$, obtained from the computation in Step 1, has floating point entries, while the $\widetilde{W}^{[i]}$, $1 \le i \le k$, are rational positive semidefinite matrices. Therefore, the remaining task of (**P.2**) is to find a nearby rational positive semidefinite matrix $\widetilde{W}^{[0]}$ such that the equation in (12) holds exactly, which is exactly the same problem as described in [14]. Therefore, we can apply Gauss-Newton iteration to refine $W^{[0]}$, and recover a rational positive definite matrix $\widetilde{W}^{[0]}$ from the refined $W^{[0]}$ by orthogonal projection if $W^{[0]}$ is of full rank, or by rational vector recovery method otherwise.

Remark 3. In practice, we will do as follows to fulfil the above task of finding a rational positive semidefinite matrix $\widetilde{W}^{[0]}$ that satisfies (12) exactly. First, using the rational polynomial $V(\mathbf{x})$ determined by the coefficient vector $\mathbf{v}$, and rational positive semidefinite matrices $\widetilde{W}^{[1]}, \ldots, \widetilde{W}^{[k]}$, we will reconstruct an SDP system of the form (12), which gives us a better initial point for Gauss-Newton iteration. Then, we will compute numerical solutions for $W^{[0]}$ of (12) using SDP solvers, and finally apply the method in [14] to recover $\widetilde{W}^{[0]}$.

The above discussion leads to an algorithm of computing the certified polynomial inequality invariants of the dynamical system $\dot{\mathbf{x}} = \mathbf{p}$ with state invariants ψ. As stated above, we only present the case where the invariant $V(\mathbf{x})$ satisfies (7), and the case of (8) is similar.

3.3 Algorithm

Algorithm *Polynomial Inequality Invariant Generation*

```
Input:   ▸ $\dot{\mathbf{x}} = \mathbf{p}$: a continuous dynamical system.
         ▸ $d \in \mathbb{Z}_{>0}$: the degree bound of the candidate
           polynomial invariant.
         ▸ $D \in \mathbb{Z}_{>0}$: the bound of the common denom-
           inator of the coefficient vector of the polyno-
           mial invariant.
         ▸ $N \in \mathbb{Z}_{\ge 0}$: the degree bound $\deg(\sigma_i) \le 2N$
           used to construct the SDP system.
         ▸ $\tau \in \mathbb{R}_{>0}$: the given tolerance.
Output:  ▸ $V(\mathbf{x})$: a verified polynomial invariant.

1. Compute the candidates of polynomial invariants
   (a) Case 1: the state invariant is empty:
       (i) Predetermine the template of $V(\mathbf{x})$ with degree $d$ and construct an SDP system of form (6).
           • If the SDP system (6) has no feasible solutions,
             return "we can't find polynomial invariants with degree $\le d$;"
           • Otherwise,
             obtain a numerical vector $\mathbf{v}^*$ and a numerical positive semidefinite matrix $W$.
       (ii) For the common denominator bound $D$, compute from $\mathbf{v}^*$ a rational vector $\mathbf{v}$ by Diophantine approximation algorithm, and then the associated rational polynomial $V(\mathbf{x})$.
   (b) Case 2: the state invariant is given as
             $\psi : \psi_1 \ge 0 \wedge \ldots \wedge \psi_k \ge 0.$     (13)
       (i) Set up the SDP system (9) by predetermining the template of $V(\mathbf{x})$ with $\deg(V) = d$ and predetermining the template of $\sigma_j$ with $\deg(\sigma_j) \le 2N$.
           • If the SDP system (9) has no feasible solutions,
             return "we can't find polynomial invariants with degree $\le d$;"
```

- Otherwise,
obtain a numerical vector $\mathbf{v}$ and numerical positive semidefinite matrices $W^{[0]}$ and $W^{[i]}, 1 \leq i \leq k$.

(ii) For the common denominator bound D, compute a rational vector $\mathbf{v}$ using Diophantine approximation algorithm and get the associate rational polynomial $V(\mathbf{x})$.

(iii) Convert all the $W^{[i]}$, $1 \leq i \leq k$, into rational matrices $\widetilde{W}^{[i]}$ that are positive semidefinite.

2. Compute the exact SOS decomposition

(a) **Case 1:** the state invariant is empty:

(i) Use $V(\mathbf{x})$ to reconstruct an SDP system of form (6) and get an approximate SOS decomposition:

$$r(\mathbf{x}) \approx m(\mathbf{x})^T \cdot W^{[0]} \cdot m(\mathbf{x}) \qquad (14)$$

with $r(\mathbf{x}) = \mathbf{d}V(\mathbf{x}) \cdot \mathbf{p}$.

(ii) Apply Gauss-Newton iteration to refine $W^{[0]}$ in (14).

Case 2: the state invariant is given as (13)

(i) Reconstruct an SDP system of the form (12) to get an approximate positive semidefinite matrix $W^{[0]}$ satisfying

$$r(\mathbf{x}) - \sum_{i=1}^{k} m_i(\mathbf{x})^T \cdot \widetilde{W}^{[i]} \cdot m_i(\mathbf{x})\psi_i \approx m_0(\mathbf{x})^T \cdot W^{[0]} \cdot m_0(\mathbf{x}) \qquad (15)$$

with $r(\mathbf{x}) = \mathbf{d}V(\mathbf{x}) \cdot \mathbf{p}$.

(ii) Apply Gauss-Newton iteration to refine $W^{[0]}$ in (15).

(b) From the refined $W^{[0]}$, compute a rational matrix $\widetilde{W}^{[0]}$ satisfying (15) exactly by orthogonal projection method if $W^{[0]}$ is of full rank, or by rational vector recovery if $W^{[0]}$ is singular.

(c) Check whether $\widetilde{W}^{[0]}$ is positive semidefinite.

- If so, return $V(\mathbf{x})$;
- Otherwise,
return "we can't find polynomial invariants with degree $\leq d$."

Remark 4. Our algorithm cannot guarantee a rational solution will always be found since there are some limitations of the above algorithm such as choosing the degree bound N and the common denominator bound D. Furthermore, it is difficult to determine in advance whether there exists differential invariants with rational coefficients or not. Therefore, even if our algorithm cannot find differential invariants, it does not mean that the given dynamical system has no differential invariants.

4. EXPERIMENTS

In this section, two examples are given to illustrate our algorithm for computing polynomial inequality invariants.

Example 1. Consider the nonlinear system

$$\dot{x_1} = x_1 + x_2^2, \qquad \dot{x_2} = x_1^2 + x_2.$$

Assume that $\deg(V) = 2k+1$ for $k = 0, 1, 2, \ldots$. For $\deg(V) = 1$, there exists no feasible solutions of the corresponding SDP system. Now set $\deg(V(x_1, x_2)) = 3$ and let the coefficients of $V(x_1, x_2)$ be parameters. By solving the associated SDP system, we obtain a numerical polynomial $V^*(x_1, x_2)$ as follows

$$\begin{aligned} V^*(x_1, x_2) =& 0.0778x_1^3 + 0.1287x_1^2x_2 + 0.3406x_1^2 + 0.1287x_1x_2^2 \\ & - 0.06165x_1x_2 - 1.75 \times 10^{-6}x_1 + 0.0778x_2^3 \\ & + 0.3406x_2^2 - 1.75 \times 10^{-6}x_2. \end{aligned}$$

Let $\tau = 10^{-2}$, and round the coefficients of $V^*(x_1, x_2)$ to be rational numbers with the common denominator bound 10^4. We obtain a rational polynomial

$$\begin{aligned} V(x_1, x_2) =& \frac{33}{424}x_1^3 + \frac{191}{1484}x_1^2x_2 + \frac{1011}{2968}x_1^2 + \frac{191}{1484}x_1x_2^2 \\ & - \frac{183}{2968}x_1x_2 + \frac{33}{424}x_2^3 + \frac{1011}{2968}x_2^2. \end{aligned}$$

Therefore,

$$\begin{aligned} r(x_1, x_2) = \frac{\mathbf{d}V}{dt} =& \frac{99}{212}x_1^2x_2^2 + \frac{255}{1484}x_1^3 + \frac{191}{742}x_1x_2^3 + \frac{396}{371}x_1^2x_2 \\ & + \frac{396}{371}x_1x_2^2 + \frac{1011}{1484}x_1^2 + \frac{191}{1484}x_2^4 + \frac{255}{1484}x_2^3 \\ & - \frac{183}{1484}x_1x_2 + \frac{191}{1484}x_1^4 + \frac{191}{742}x_1^3x_2 + \frac{1011}{1084}x_2^2. \end{aligned}$$

By running the algorithm in Section 3.3, we find that $r(x_1, x_2)$ can be written as an SOS of 4 polynomials with rational coefficients. The SOS representation of $r(x_1, x_2)$ is given in Appendix. Therefore,

$$V(x_1, x_2) - d_0 \geq 0,$$

with $d_0 = \min\{V(x_1, x_2) : (x_1, x_2) \in \Theta\}$, is an invariant of the given nonlinear system.

The next example shows how to compute polynomial inequality invariants for nonlinear systems with state invariants.

Example 2. Consider the following nonlinear system

$$\dot{x_1} = x_2 + x_1x_2^2, \qquad \dot{x_2} = -x_1 + x_1^2x_2.$$

Assume that the state invariant is $x_1 \geq 0 \wedge x_2 \geq 0$. Let us assume that $\deg(V) = 2k, k = 1, 2, \ldots$. According to Corollary 1, we construct the associated SDP system as (8), that is, the polynomial

$$\mathbf{d}V \cdot \mathbf{p} - \sigma_1(x_1, x_2)x_1 - \sigma_2(x_1, x_2)x_2 - \sigma_3(x_1, x_2)x_1x_2$$

can be written as an SOS, and $\sigma_i(x_1, x_2), i = 1, 2, 3$ can also be written as SOSs.

Suppose the degree bounds of $V(x_1, x_2)$ and of $\sigma_i(x_1, x_2)$ are all 2.

Solve the associated SDP system and find numerical solutions:

$$V^*(x_1, x_2) = 0.424\,x_1^2 + 0.424\,x_2^2 + \cdots + 2.286 \times 10^{-13}x_2,$$

and

$$\begin{aligned} \sigma_1^*(x_1, x_2) =& -1.171 \times 10^{-12}x_1^2 + \cdots + 2.121 \times 10^{-5}x_2^2, \\ \sigma_2^*(x_1, x_2) =& 1.426 \times 10^{-5}x_1^2 + \cdots + 8.87 \times 10^{-13}x_2^2, \\ \sigma_3^*(x_1, x_2) =& 2.917 \times 10^{-11}x_1^2 + \cdots + 3.367 \times 10^{-11}x_2^2. \end{aligned}$$

Set $\tau = 10^{-2}$, and round the coefficients of $V^*(x_1, x_2)$ as rational numbers with the common denominator bound 100. We obtain a rational polynomial

$$V(x_1, x_2) = \frac{14}{33}x_1^2 + \frac{14}{33}x_2^2,$$

and

$$\sigma_i(x_1, x_2) = 0, i = 1, 2, 3.$$

It is easy to verify that

$$\frac{\mathbf{d}V}{dt} - \sigma_1(x_1, x_2)x_1 - \sigma_2(x_1, x_2)x_2 - \sigma_3(x_1, x_2)x_1x_2 = \frac{56}{33}x_1^2x_2^2,$$

and is thus positive semidefinite. So,

$$V(x_1, x_2) - d_0 \geq 0, \quad \text{or equivalently,} \quad x_1^2 + x_2^2 - d_0 \geq 0,$$

with $d_0 = \min\{V(x_1, x_2) : (x_1, x_2) \in \Theta\}$, is an invariant of the given nonlinear system with the state invariants $x_1 \geq 0$ and $x_2 \geq 0$.

For degree bounds 4 and 6 of the polynomial $V(x_1, x_2)$, the associated SDP systems have no feasible solutions.

Now suppose the degree bound of $V(x_1, x_2)$ is 8. By solving the associated SDP system, we get numerical solutions

$$V^* = 0.228x_1^4x_2^4 + 0.006x_1^4x_2^3 + 0.197x_1^4x_2^2 + \cdots + 0.251x_2^4 + 0.829x_2^2,$$

and

$$\begin{aligned}
\sigma_1^* &= 8.80 \times 10^{-13}\, x_1^5x_2 + 0.741x_1^2x_2^4 + \cdots + 2.27 \times 10^{-6}x_2^6,\\
\sigma_2^* &= 1.80x_1^6 + 0.765\, x_1^4x_2^2 + \cdots - 2.95 \times 10^{-12}\, x_2^6,\\
\sigma_3^* &= -1.763 \times 10^{-12}x_1^6 + 0.602x_1^4x_2^2 + \cdots - 5.15 \times 10^{-13}\, x_2^6.
\end{aligned}$$

Set $\tau = 10^{-2}$, and round the coefficients of V^* as rational numbers with the common denominator bound 100. We obtain the corresponding rational polynomial $V(x_1, x_2)$ and $\sigma_i(x_1, x_2), i = 1, 2, 3$:

$$\begin{aligned}
V =& \frac{17}{76}x_1^4x_2^4 + \frac{15}{76}x_1^4x_2^2 + \frac{27}{76}x_1^4 + \frac{4}{19}x_1^3x_2^3 - \frac{15}{38}x_1^3x_2\\
&+ \frac{15}{76}x_1^2x_2^4 + \frac{11}{19}x_1^2x_2^2 + \frac{63}{76}x_1^2 + \frac{15}{38}x_1x_2^3 + \frac{1}{4}x_2^4 + \frac{63}{76}x_2^2,\\
\sigma_1 =& \frac{14}{19}x_1^2x_2^4, \quad \sigma_2 = \frac{38}{29}x_1^4x_2^2, \quad \sigma_3 = \frac{23}{38}x_1^4x_2^2 + \frac{45}{38}x_1^2x_2^4.
\end{aligned}$$

Now let us verify whether the polynomial

$$\frac{\mathbf{d}V}{dt} - \sigma_1(x_1, x_2)x_1 - \sigma_2(x_1, x_2)x_2 - \sigma_3(x_1, x_2)x_1x_2$$

is positive semidefinite. By running the algorithm in Section 3.3, we find that the above polynomial can be written as an SOS of 8 polynomials, and its SOS representation is listed in Appendix. With the state invariants $x_1 \geq 0 \wedge x_2 \geq 0$, it is obvious that $\frac{\mathbf{d}V}{dt}$ is always positive semidefinite. Therefore,

$$V(x_1, x_2) - d_0 \geq 0$$

with $d_0 = \min\{V(x_1, x_2) : (x_1, x_2) \in \Theta\}$, is an invariant of the given nonlinear system with the state invariants $x_1 \geq 0$ and $x_2 \geq 0$.

5. CONCLUSION

In this paper, we present a symbolic-numeric approach to compute inequality invariants of nonlinear continuous systems in hybrid systems. Employing SOS relaxation and rational vector recovery techniques, it can be guaranteed that an exact invariant, rather than a numerical one, can be obtained efficiently and practically. This approach avoids both the high complexity of symbolic invariant generation methods based on quantifier elimination, and the weakness of applying numerical approaches to verify correctness of hybrid systems.

Acknowledgments

We wish to thank Professor Lu Yang for his valuable suggestions on improving our two examples, and we also thank the reviewers for their helpful comments.

6. REFERENCES

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Their. Compute. Sci.*, 138:3–34, 1995.

[2] E. Asarin, T. Dang, and A. Girard. Reachability analysis of nonlinear systems using conservative approximation. In O. Maler and A. Pnueli, editors, *Hybrid Systems: Computation and Control*, volume 2623 of *Lecture Notes in Computer Science*, pages 20–35. Springer Berlin/Heidelberg, 2003.

[3] M. Branicky. General hybrid dynamical systems: Modeling, analysis, and control. In *Alur, R., Henzinger, T.A. Sontag, E.D., eds: Hybrid Systems, Volume 1066 of LNCS*, pages 186–200, 1995.

[4] E. R. Carbonell and A. Tiwari. Generating polynomial invariants for hybrid systems. In *HSCC, volume 3414 of LNCS*, pages 590–605, 2005.

[5] Y. Chen, B. Xia, L. Yang, and N. Zhan. Generating polynomial invariants with discoverer and qepcad. *LNCS*, 4700:67–82, 2007.

[6] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *J. Symb. Compute.*, 12(3):299–328, 1991.

[7] M. Colón and H. Sipma. Synthesis of linear ranking functions. In *TACAS, volume 2031 of LNCS*, pages 67–81, 2001.

[8] M. A. Colón, S. Sankaranarayanan, and H. B. Sipma. Linear invariant generation using non-linear constraint solving. *Lecture Notes in Computer Science*, 2725:420–432, 2003.

[9] P. Cousot. Proving program invariance and termination by parametric abstraction, lagrangian relaxation and semidefinite programming. *LNCS*, 3385:1–24, 2005.

[10] J. Davoren and A. Nerode. Logics for hybrid systems. In *Proceedings of the IEEE*, volume 88, pages 985–1010, 2000.

[11] Z. Y. Erich Kaltofen, Bin Li and L. Zhi. Exact certification in global polynomial optimization via sums-of-squares of rational functions with rational coefficients. Accepted by Journal of Symbolic Computation, 2009.

[12] S. Gulwani and A. Tiwari. Constraint-based approach for hybrid systems. *LNCS*, 5123:190–203, 2008.

[13] T. Henzinger. The theory of hybrid automata. In *LICS, Los Alamitos*, pages 278–292. IEEE Computer Society, 1996.

[14] E. Kaltofen, B. Li, Z. Yang, and L. Zhi. Exact certification of global optimality of approximate

factorizations via rationalizing sums-of-squares with floating point scalars. In *Proceedings of the twenty-first international symposium on Symbolic and algebraic computation*, ISSAC '08, pages 155–164, New York, NY, USA, 2008. ACM.

[15] D. Kapur. Automatically generating loop invariants using quantifier elimination preliminary report-. In *IMACS Intl. Conf. on Applications of Computer Algebra*, 2004.

[16] L. Kovács. Reasoning algebraically about p-solvable loops. *Lecture Notes in Computer Science*, 4963:249–264, 2008.

[17] J. C. Lagarias. The computational complexity of simultaneous diophantine approximation problems. *SIAM J. Comp.*, 14:196–209, 1985.

[18] J. Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD*, Taipei, Taiwan, 2004. Available at `http://control.ee.ethz.ch/~joloef/yalmip.php`.

[19] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer, New York, 1995.

[20] N. Matringe, A. V. Moura, and R. Rebiha. Morphisms for non-trivial non-linear invariant generation for algebraic hybrid systems. In *HSCC, volume 5469 of LNCS*, pages 445–449, 2009.

[21] A. Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *Journal of Logic and Computation*, 20:309–352, 2007.

[22] A. Platzer. Differential dynamic logic for verifying parametric hybrid systems. *LNAI*, 4548:216–232, 2007.

[23] A. Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reasoning Manuscript*, 41:143–189, 2008.

[24] A. Platzer and E. Clarke. Formal verification of curved flight collision avoidance maneuvers: A case study. In *Proceedings of Formal Methods*, pages 547–562, 2009.

[25] A. Platzer and E. M. Clarke. The image computation problem in hybrid systems model checking. In *Bemporad et al*, pages 473–486. Springer, 2007.

[26] A. Platzer and E. M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. *Form. Methods Syst. Des.*, 35:98–120, August 2009.

[27] A. Platzer and J. Quesel. European train control system: A case study in formal verification. 5885:246–265, 2009.

[28] A. Podelski and S. Wagner. A sound and complete proof rule for region and stability of hybrid systems. In *HSCC, volume 4416 of LNCS*, pages 750–753, 2007.

[29] S. Prajna, A. Papachristodoulou, and P. Parrilo. SOSTOOLS: Sum of squares optimization toolbox for MATLAB, 2002. Available at `http://www.cds.caltech.edu/sostools`.

[30] E. Rodríguez-Carbonella and D. Kapur. Generating all polynomial invariants in simple loops. *Journal of Symbolic Computation*, 42:443–476, 2007.

[31] S. Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In *ProcHSCC'10*, 2010.

[32] S. Sankaranarayanan, H. Sipma, and Z. Manna. Fixed point iteration for computing the time-elapse operator. In *HSCC, LNCS*. Springer, 2006.

[33] S. Sankaranarayanan, H. Sipma, and Z. Manna. Constructing invariants for hybrid systems. *Formal Methods in System Design*, 32:25–55, 2008.

[34] S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Non-linear loop invariant generation using Gröbner bases, 2004.

[35] J. F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11/12:625–653, 1999.

[36] A. Tiwari and G. Khanna. Nonlinear systems: Approximating reach sets. *Lecture Notes of Computer Science*, 2993:600–614, 2004.

[37] A. Tiwari, H. Ruess, H. Saïdi, and N. Shankar. A technique for invariant generation. In *TACAS 2001, vol. 2031 of LNCS*, pages 113–127. Springer-Verlag, 2001.

[38] B. Xia. Discoverer: a tool for solving semi-algebraic systems. *ACM Commun. Compute. Algebra*, 41(3):102–103, 2007.

Appendix

A solution to Example 1,

$$\mathbf{r} = \frac{1484}{1011}f_1^2 + \frac{285776}{193095}f_2^2 + \frac{11143727}{1016838}f_3^2 + \frac{1508987592}{30774371}f_4^2$$

where

$$f_1 = \tfrac{1011}{1484}x_2 - \tfrac{183}{2968}x_1 + \tfrac{255}{2968}x_2^2 + \tfrac{1209}{2968}x_1x_2 + \tfrac{375}{2968}x_1^2,$$
$$f_2 = \tfrac{193095}{285776}x_1 + \tfrac{268305}{2000432}x_2^2 + \tfrac{126945}{285776}x_1x_2 + \tfrac{194745}{2000432}x_1^2,$$
$$f_3 = \tfrac{1016838}{11143727}x_2^2 - \tfrac{4931}{454846}x_1x_2 - \tfrac{1792057}{22287454}x_1^2$$
$$f_4 = \tfrac{30774371}{1508987592}x_1x_2 - \tfrac{30774371}{1508987592}x_1^2.$$

□

A solution to Example 2,

$$\mathbf{r} = \frac{38}{15}f_1^2 + \frac{190}{147}f_2^2 + \frac{2793}{1010}f_3^2 + \frac{1919}{4172}f_4^2 + \frac{4756080}{7066109}f_5^2 + \frac{1074048568}{1371482325}f_6^2 + \frac{104232656700}{92227612763}f_7^2 + \frac{28037194279952}{24255149844713}f_8^2,$$

where

$$f_1 = \tfrac{15}{38}x_2^2 + \tfrac{3}{38}x_1x_2 + \tfrac{3}{38}x_1^2 - \tfrac{2}{19}x_1^2x_2 + \tfrac{15}{38}x_1x_2^3 + \tfrac{4}{19}x_1^2x_2^2 - \tfrac{3}{38}x_1^3x_2 - \tfrac{3}{38}x_1^3x_2^2,$$
$$f_2 = \tfrac{147}{190}x_1x_2 + \tfrac{11}{95}x_1^2 + \tfrac{2}{19}x_1x_2^2 - \tfrac{8}{95}x_1^2x_2 - \tfrac{3}{190}x_1^2x_2^2 - \tfrac{11}{95}x_1^3x_2 + \tfrac{1}{38}x_1^2x_2^3 + \tfrac{21}{380}x_1^3x_2^2,$$
$$f_3 = \tfrac{1010}{2793}x_1^2 + \tfrac{250}{2793}x_1x_2^2 + \tfrac{94}{2793}x_1^2x_2 + \tfrac{563}{1862}x_1^2x_2^2 - \tfrac{1010}{2793}x_1^3x_2 - \tfrac{11}{2793}x_1^2x_2^3 + \tfrac{1}{133}x_1^3x_2^2,$$
$$f_4 = \tfrac{4172}{1919}x_1x_2^2 - \tfrac{89}{3838}x_1^2x_2 - \tfrac{1087}{3838}x_1^2x_2^2 - \tfrac{17}{404}x_1^2x_2^3 - \tfrac{119}{1919}x_1^3x_2^2,$$
$$f_5 = \tfrac{7066109}{4756080}x_1^2x_2 - \tfrac{1391293}{4756080}x_1^2x_2^2 - \tfrac{175701}{3170720}x_1^2x_2^3 - \tfrac{12287}{113240}x_1^3x_2^2,$$
$$f_6 = \tfrac{1371482325}{1074048568}x_1^2x_2^2 + \tfrac{23684501}{268512142}x_1^2x_2^3 - \tfrac{4029727}{537024284}x_1^3x_2^2,$$
$$f_7 = \tfrac{92227612763}{104232656700}x_1^2x_2^3 - \tfrac{110608051}{208465313400}x_1^3x_2^2,$$
$$f_8 = \tfrac{24255149844713}{28037194279952}x_1^3x_2^2.$$

□