Automatic Invariant Generation For Hybrid Systems Using Ideal Fixed Points

Sriram Sankaranarayanan
University of Colorado, Boulder, CO 80305, USA.
srirams@colorado.edu

ABSTRACT

We present computational techniques for automatically generating algebraic (polynomial equality) invariants for algebraic hybrid systems. Such systems involve ordinary differential equations with multivariate polynomial right-hand sides. Our approach casts the problem of generating invariants for differential equations as the greatest fixed point of a monotone operator over the lattice of ideals in a polynomial ring. We provide an algorithm to compute this monotone operator using basic ideas from commutative algebraic geometry. However, the resulting iteration sequence does not always converge to a fixed point, since the lattice of ideals over a polynomial ring does not satisfy the descending chain condition.

We then present a bounded-degree relaxation based on the concept of "pseudo ideals", due to Colón, that restricts ideal membership using multipliers with bounded degrees. We show that the monotone operator on bounded degree pseudo ideals is convergent and generates fixed points that can be used to generate useful algebraic invariants for non-linear systems. The technique for continuous systems is then extended to consider hybrid systems with multiple modes and discrete transitions between modes.

We have implemented the exact, non-convergent iteration over ideals in combination with the bounded degree iteration over pseudo ideals to guarantee convergence. This has been applied to automatically infer useful and interesting polynomial invariants for some benchmark non-linear systems.

Categories and Subject Descriptors: F.3.1(Specifying and Verifying and Reasoning about Programs):Invariants,
C.1.m(Miscellaneous): Hybrid Systems.

General Terms: Theory, Verification.

Keywords: Ordinary Differential Equations, Hybrid Systems, Algebraic Geometry, Invariants, Verification, Conservation Laws.

1. INTRODUCTION

An invariant of a system is 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 verifying and understanding systems. As such, they are used to establish temporal properties of systems such as safety, stability, termination, progress and so on [12, 3, 18, 15]. The *invariant generation* problem consists of automatically computing useful invariants given the description of a dynamical system. The problem of generating invariants of an arbitrary form is known to be computationally intractable. However, approaches based on discovering invariants of pre-specified forms for a given class of systems have been successful in generating non-trivial invariants.

In this work, we present a technique for generating algebraic invariants for algebraic systems whose variables evolve according to nonlinear ordinary differential equations with multivariate polynomial right-hand sides. Our work attempts to synthesize polynomial invariants such as: $\bigwedge_i p_i = 0$, without *a priori* restrictions on the number of conjuncts involved. In order to discover polynomial *invariants* for a given system, we formulate the notion of an *invariant ideal* I over the ring of polynomials, such that for any polynomial $p \in I$, its *Lie derivative* also belongs to I. Further, we express an invariant ideal as a fixed point of a monotonic refinement operator over ideals. We present techniques to find an invariant ideal as a fixed point using Tarski iteration. Hence, our approach can be viewed as an abstract interpretation framework for continuous systems described by ODEs [5].

The main contributions of this work are as follows: (A) we present invariant generation over continuous vector fields as the fixed point of a monotonic refinement operator over ideals, (B) we present an algorithm for computing the refinement operator over ideals, and (C) we formulate the refinement operator over *pseudo ideals*, originally defined by Colón [4], by restricting the degree of the polynomial multipliers involved in ideal membership. The resulting iteration scheme is shown to converge in finitely many steps. The technique of generating algebraic invariants can be extended to handle continuous algebraic systems with holonomic constraints as well as hybrid systems by computing post-conditions over ideals. We have implemented a prototype system inside Mathematica (tm) that has been used to compute interesting invariants for numerous nonlinear systems.

Recently, there has been a considerable volume of work towards analyzing algebraic systems using techniques from convex optimization, commutative algebraic and semi-algebraic geometry [20, 23, 19, 14, 2, 21, 15, 16, 13]. Many of these techniques, including our previous work, synthesize invariants by computing constraints on the unknown coefficients of a single or fixed number of polynomial equalities (inequalities) of a bounded degree, that guarantee that any solution will also be inductive [20, 23, 19, 10, 13]. Bounding the degree of the invariant is a restriction, only in theory. It has been repeatedly demonstrated that useful low-degree polynomial invariants can be found that can help prove properties of complex systems [16, 15, 10]. A more subtle restriction, as pointed out by

HSCC'10, April 12–15, 2010, Stockholm, Sweden.

Platzer et al. [16] is over the number of conjuncts involved in an invariant. For instance, our previous work as well as that of Matringe et al. [20, 13] consider single polynomial equalities. This form remains very useful for systems, where we find that a vast majority of invariants that take the form of *conservation laws* can be efficiently discovered. However, useful invariants may have a more complex Boolean structure involving conjunctions of invariants such as $\bigwedge p_i = 0$, wherein each $p_i = 0$ is not an invariant by itself (Cf. Example 1.1, below). Gulwani and Tiwari consider invariants of an arbitrary but fixed Boolean structure (including disjunctions). However, their encoding restricts the domain of proofs to an essentially finite domain. In our experience, this restriction is much more meaningful to discrete programs wherein invariants with unit coefficients are the norm, than to hybrid systems where parameters may assume irregular values such as $3.1415726....$ Such parameters may require more bits to represent or extra variables (dimensions) and constraints. Nevertheless, this technique has been used successfully to analyze the stability of complex adaptive flight control system [22]. The work of Carbonell and Tiwari is remarkable in that it does not assume bounded degree or restrictions on the conjuncts [2]. A key restriction however is that it applies mostly to linear systems. Recently, Platzer et al. have proposed a powerful theorem-proving framework for verifying properties of hybrid systems through *differential logic* that extends dynamic logics using differential operators [16, 15]. The technique can also synthesize invariants by parameterizing the coefficients of an unknown polynomial to aid in its proof. Efficiency is guaranteed by using *local reasoning* involving a subset of the state variables and the dynamics. Our techniques, including most of the other invariant generation techniques discussed here, can be naturally lifted to this framework, and thus be used to discharge proof obligations for larger, more complex hybrid system involving many more variables such as collision avoidance maneuvers and automatic train control systems [17].

Example 1.1. *Consider the system*

$$\frac{dx_1}{dt} = x_2,\ \frac{dx_2}{dt} = x_3, \ldots, \frac{dx_n}{dt} = x_1\,,$$

with the initial state $x_1, \ldots, x_n = (0, \ldots, 0)$ for $n > 2$. Our approach can establish the invariant

$$x_1 = 0 \ \wedge\ x_2 = 0 \ \wedge\ \ldots \wedge x_n = 0\,,$$

without explicitly integrating the system dynamics. However, no single assertion $x_i = 0$ can be established without simultaneously asserting that $x_j = 0$ for all other $j \in [1, n]$. The same holds for any fixed number of assertions.

Our overall approach can be viewed as an abstract interpretation over the lattice of ideals, in the form of a Galois connection between the lattice of *algebraic varieties* representing a set of continuous states and ideals over a polynomial ring whose zeroes define the algebraic variety [5]. Our previous work on linear systems also used an abstract interpretation scheme to compute linear inequality invariants for such systems [21]. Therein, we considered the lattice of polyhedral cones and used heuristic *widening* operators to force convergence. Here, we consider polynomial equality invariants for algebraic systems. Further, our approach here does not employ *widening* operators. Instead, we prove that our iteration over the lattice of degree-bounded pseudo ideals converges in finitely many steps. In this respect, our work presents a lot of similarities with the seminal work of Karr for discovering affine invariants of programs [11]. This also raises the exciting possibility of a *randomized algorithm* for generating invariants efficiently using pseudo ideals along the lines of Gulwani and Necula [9].

The rest of the paper is organized as follows: Section 2 presents some basic notions from commutative algebra, Section 3 details the fixed point characterization for continuous systems and presents the algorithm for computing the refinement operator, Section 4 extends our technique to pseudo ideals, Section 5 briefly discusses an extension to hybrid system, Section 6 presents experiments. We conclude by discussing some of the related work and future extensions to our technique.

2. PRELIMINARIES

In this section, we define the basic concepts from commutative algebraic geometry that will be used throughout this work. Let $\mathcal{R}$ denote the field of real numbers and $\mathcal{C}$, the field of complex numbers obtained as the algebraic closure of $\mathcal{R}$. Many of the primitives discussed here can be specialized to any field K (of characteristic 0). Unless, otherwise mentioned, we use K to denote one of the fields $Q, \mathcal{R}$, or $\mathcal{C}$. Let $x_1, \ldots, x_n$ denote a set of variables, collectively represented as $\vec{x}$. The $K[\vec{x}]$ denotes the ring of multivariate polynomials over a given field K.

A *monomial* over $\vec{x}$ is of the form $x_1^{r_1} x_2^{r_2} \cdots x_n^{r_n}$, succinctly written as $\vec{x}^{\vec{r}}$, wherein each $r_i \in Z^{\geq 0}$. A *term* is of the form $c \cdot m$ where $c \in K,\ c \neq 0$ and m is a monomial. The degree of a monomial $\vec{x}^{\vec{r}}$ is given by $\sum_{i=1}^{n} r_i = \vec{1} \cdot \vec{r}$. The degree of a multivariate polynomial p is the maximum over the degrees of all monomials m that *occur* in p with a non-zero coefficient.

Def. 2.1 (Ideal). *An* ideal $I \subseteq K[x_1, \ldots, x_n]$ *is a set of polynomials with the following properties:*

- $0 \in I$,
- *If* $p_1, p_2 \in I$ *then* $p_1 + p_2 \in I$,
- *If* $p \in I$ *and* $q \in K[\vec{x}]$ *then* $pq \in I$.

The ideal generated by a set $P = \{p_1, \ldots, p_m\} \subseteq K[\vec{x}]$, $n \geq 0$, of polynomials is written as

$$\langle\!\langle P \rangle\!\rangle = \{\sum_{i=1}^{m} g_i p_i \ \mid\ p_i \in P,\ g_i \in K[\vec{x}], \text{ for } i \in [1, m]\}$$

Ideal I is finitely generated if $I = \langle\!\langle P \rangle\!\rangle$ for some finite set P, called the *basis* of I. The *Hilbert basis theorem* states that every ideal $I \subseteq K[x_1, \ldots, x_n]$ is finitely generated [6]. Informally, ideals can be viewed as representing some of the polynomial consequences of a finite set of polynomials. The *algebraic variety* (or simply the variety) corresponding to an ideal I (denoted $\mathcal{V}(I)$) consists of a set of points $\vec{x}$ such that $p(\vec{x}) = 0$ for all $p \in I$. Similarly, an algebraic variety X corresponds to the ideal $I = \mathcal{I}(X)$ containing all the polynomials $p \in K[\vec{x}]$ such that $p(\vec{x}) = 0$ for all $\vec{x} \in X$.

Theorem 2.1. *Let I be an ideal generated by $P : \{p_1, \ldots, p_m\}$. Then for any polynomial p, if $p \in I$ then $\{p_1 = 0, \ldots, p_m = 0\} \models (p = 0)$.*

Theorem 2.1 states that in order to establish the entailment $\varphi \models (p = 0)$, it is sufficient to test membership of p in the ideal generated by φ. The "converse" of Theorem 2.1 is true, provided the field K is algebraically closed: if $\varphi \models p = 0$ then $p^m \in I$ for some $m > 0$. In general, we can ensure that index $m = 1$ for all p by computing the *ideal radical* of I [6].

Def. 2.2 (Ideal Membership Problem). *Given a finite set of polynomials $P \subseteq K[\vec{x}]$ and a polynomial $p \in K[\vec{x}]$ the ideal membership problem decides whether $p \in \langle\!\langle P \rangle\!\rangle$.*

In general, there are multiple techniques for deciding the ideal membership problem. The theory of Groebner basis [6] and Wu's method [8] remain popular for deciding ideal membership. We now present the notion of *Syzygies*. Syzygies will be used in this work to define a monotonic operator over ideals.

Def. 2.3 (Syzygies). *Let $P : \{p_1, \ldots, p_m\}$ be a finite set of polynomials. A* Syzygy *of P is a vector of polynomials $(g_1, \ldots, g_m)$ such that $\sum_i g_i p_i = 0$. We denote the set of all Syzygies of P as* $\text{SYZ}(P)$.

If $\vec{g} : (g_1, \ldots, g_m)$ and $\vec{h} : (h_1, \ldots, h_m)$ are syzygies of P, then so are $\vec{g} + \vec{h}$ and $p\vec{g} : (pg_1, \ldots, pg_m)$ for any $p \in K[\vec{x}]$. As a result, the set of all syzygies form a *module* over the ring $K[\vec{x}]$. Informally, a module can be viewed as the analog of a "vector space" over a ring. Whereas the notion of a vector space is defined over a field K, modules generalize vector spaces over rings.

Theorem 2.2. *For a finite set $P : \{p_1, \ldots, p_m\}$ of polynomials, the syzygies* $\text{SYZ}(P)$ *has the following property:*

1. $\text{SYZ}(P)$ *forms a module over $K[\vec{x}]$.*
2. $\text{SYZ}(P)$ *is a finitely generated module for the ring polynomials $K[\vec{x}]$ over a field K.*
3. *The generators of* $\text{SYZ}(P)$ *can be computed using the Groebner basis G of P.*

PROOF. A proof is available from Cox et al. [6] or Adams& Loustaunau [1]. □

We note that once a Groebner basis G is computed, the generators of the Syzygy bases can be computed by modifying the standard Büchberger's algorithm for computing the Groebner basis [1].

2.1 Algebraic Templates

A template polynomial p is a polynomial whose coefficients are linear expressions over a set of unknowns A.

Def. 2.4 (Template). *Let A be a set of* template variables *and $Lin(A)$ be the domain of all* linear expressions *over variables in A of the form $c_0 + c_1 a_1 + \ldots + c_n a_n,\ c_i \in K$. A* template *over A, X is a polynomial in $Lin(A)[\vec{x}]$. An A-*environment *is a map α that assigns a value in K to each variable in A, and by extension, maps each expression in $Lin(A)$ to a value in K, and each template in $Lin(A)[\vec{x}]$ to a polynomial in $K[\vec{x}]$.*

Example 2.1. *Let $A = \{a_1, a_2, a_3\}$, hence $Lin(A) = \{c_0 + c_1 a_1 + c_2 a_2 + c_3 a_3 \mid c_0, \ldots, c_3 \in \mathcal{R}\}$. An example template is $(2a_2 + 3)x_1 x_2^2 + (3a_3)x_2 + (4a_3 + a_1 + 10)$. The environment $\alpha \equiv \langle a_1 = 0, a_2 = 1, a_3 = 2\rangle$, maps this template to the polynomial $5x_1 x_2^2 + 6x_2 + 18$.*

The *generic template polynomial* over $x_1, \ldots, x_n$ of degree $m > 0$ is formed by considering all monomial terms $\vec{x}^{\vec{r}}$ such that $\sum_i r_i \leq m$.

2.2 Vector Fields

For the remainder of this discussion, let $K = \mathcal{R}$. A *vector field* F over an (open) set $X \subseteq \mathcal{R}^n$ is a map $F : X \mapsto \mathcal{R}^n$ mapping each point $\vec{x} \in X$ to a vector $F(\vec{x}) \in \mathcal{R}^n$. A vector field F is continuous if the map F is continuous. A polynomial vector field $F : X \mapsto \mathcal{R}[\vec{x}]^n$ is specified by a map $F(\vec{x}) = \langle p_1(\vec{x}), p_2(\vec{x}), \ldots, p_n(\vec{x})\rangle$, wherein $p_1, \ldots, p_n \in \mathcal{R}[\vec{x}]$. A system of ordinary differential equations D,

$$\begin{aligned} \tfrac{dx_1}{dt} &= p_1(x_1, \ldots, x_n) \\ &\vdots \\ \tfrac{dx_n}{dt} &= p_n(x_1, \ldots, x_n) \end{aligned}$$

specifies the evolution of variables $(x_1, \ldots, x_n) \in X$ over time t. Such a system can be viewed as a vector field $F(\vec{x}) : \langle p_1(\vec{x}), \ldots, p_n(\vec{x})\rangle$.

Def. 2.5 (Lie Derivative). *Given a continuous vector field $F(\vec{x}) : \langle f_1, \ldots, f_m\rangle$, the* Lie derivative *of a continuous and differentiable function $f(\vec{x})$ is given by*

$$\mathcal{L}_F(f) = (\nabla f) \cdot F(\vec{x}) = \sum_{i=1}^{n} \left(\frac{\partial f}{\partial x_i} \cdot f_i \right)$$

Henceforth, wherever the vector field F is clear from the context, we will drop subscripts and use $\mathcal{L}(p)$ to denote the Lie derivative of p w.r.t F.

Example 2.2. *Consider a mechanical system expressed in generalized position co-ordinates (q_1, q_2) and momenta (p_1, p_2) defined using the following vector field:*

$$F(p_1, p_2, q_1, q_2) : \langle\ -2q_1 q_2^2,\ -2q_1^2 q_2,\ 2p_1,\ 2p_2\ \rangle$$

The Lie-derivative of $q_1^2 - q_2^2$ is given by $4(p_1 q_1 - p_2 q_2)$.

A continuous vector field $\mathfrak{F}$ is *locally* Lipschitz continuous [25], if for each $\vec{x} \in X$ there exists an open subset $S(\vec{x}) \subseteq X$ and a constant $L(\vec{x}) \geq 0$ such that for all $\vec{y}, \vec{y}' \in S(\vec{x})$, $||F(\vec{y}') - F(\vec{y})|| \leq L(\vec{x})||\vec{y}' - \vec{y}||$. In general, all polynomial vector fields are locally Lipschitz continuous, but not necessarily *globally* Lipschitz continuous over an unbounded domain X.

2.3 Invariants

We first define algebraic systems and their semantics.

Def. 2.6 (Algebraic System). *An algebraic system over $\vec{x}$ is a pair $A : \langle F, X_0\rangle$ such that $F : \vec{x} \mapsto \mathcal{R}[\vec{x}]^n$ is a polynomial vector field and $X_0 \subseteq R^n$ is an* algebraic variety *specifying the set of* initial states *of the system.*

Note: For algebraic system $\langle F, X_0\rangle$, we assume that X_0 is specified by the generators of its ideal $\mathcal{I}(X_0) \subseteq \mathcal{R}[\vec{x}]$.

Given an algebraic system A, its time trajectories are defined as a vector-valued function over time whose gradient coincides with the value of the vector field F at all times.

Def. 2.7. *Given an algebraic system $A : \langle F, X_0\rangle$, a continuous and differentiable function $\tau : [0, T) \mapsto \mathcal{R}^n$ is a* time trajectory *of A upto time $T > 0$ if it satisfies the following conditions:*

1. $\tau(0) \in X_0$,
2. $\forall s \in [0, T),\ \frac{d\tau}{dt}|_{t=s} = F(\tau(s))$.

The Lipschitz continuity of the vector field F, ensures that given $\vec{x} = \vec{x}_0$, there exists a time $T > 0$ and a unique time trajectory $\tau : [0, T) \mapsto \mathcal{R}^n$ such that $\tau(t) = \vec{x}_0$.

Theorem 2.3 (Picard-Lindelöf). *Let F be a (time independent) polynomial vector field over an open subset $U \subseteq \mathcal{R}^n$. For any initial value $\vec{x}_0 \in U$ there exists a time interval $[0, T)$, $T > 0$ and a unique, continuous and differentiable (local)* time trajectory *$\tau : [0, T) \mapsto U$ of F such that $\tau(0) = \vec{x}_0$.*

The proof of this theorem involves an iteration of a *contractive operator* over the space of continuous functions that is termed the *Picard* iteration.

Def. 2.8 (Invariant Set). *A set X is an invariant of a given algebraic system A iff all time trajectories of the system lie in X.*

Example 2.3. *Consider the system from Ex. 2.2 using the initial set of initial states defined by*

$$X_0 : \{\vec{x} : (p_1, q_1, p_2, q_2) \mid p_1^2 + p_2^2 - 4 = 0,\ q^2 - 1 = 0, q_2 = 0\}\,.$$

Using our approach, we show that the following set is an invariant of the system above starting from the initial set of states X_0:

$$H(p_1, p_2, q_1, q_2) : p_1^2 + p_2^2 + q_1^2 q_2^2 - 4 = 0\,.$$

Incidentally, H is an expression for the Hamiltonian of the system in the co-ordinates $\vec{p}, \vec{q}$.

In this work, we wish to study algebraic invariants that are described by the common zeros of a (finite) set of multivariate polynomials in $\mathcal{R}[\vec{x}]$, i.e, invariant sets that are *algebraic varieties*. Naturally, algebraic invariants can be described by their corresponding ideals in $\mathcal{R}[\vec{x}]$ as follows:

Def. 2.9 (Invariant Ideal). *An ideal $I \subseteq \mathcal{R}[\vec{x}]$ is* invariant *for the algebraic system $\langle F, X_0\rangle$ iff*

1. $(\forall p \in I,\ \vec{x}_0 \in X_0)\ p(\vec{x}_0) = 0$*, alternatively,* $I \subseteq \mathcal{I}(X_0)$.
2. $(\forall p \in I),\ \mathcal{L}(p) \in I$.

Informally, an invariant ideal I is a sub-ideal of $\mathcal{I}(X_0)$ that is closed under the action of computing Lie derivatives of the polynomials in the ideal. We wish to prove that any invariant ideal is truly invariant w.r.t. to all the time trajectories of a system. Before we do so, we first prove the following key fact about the time trajectories of a system with polynomial right hand sides.

Lemma 2.1. *Let trajectory $\tau : [0, T) \mapsto \mathcal{R}^n$ be the unique solution to the algebraic ODE $\frac{d\vec{x}}{dt} = F(\vec{x})$, for initial value $\vec{x}_0 \in U$. Then τ is analytic around $t = 0$.*

PROOF. This theorem is a special case of the more general Cauchy-Kowalowskaya theorem for partial differential equations [7]. □

Using the fact that all derivatives exist for a trajectory around the initial state $\vec{x}_0$, we show the soundness of the invariant ideal.

Theorem 2.4. *Let I be an invariant ideal for the system $A : \langle F, X_0\rangle$. For any time trajectory $\tau : [0, T) \mapsto \mathcal{R}^n$ of A, $\tau(t) \in \mathcal{V}(I)$ for all $t \in [0, T)$.*

PROOF. We establish that $p(\tau(t)) = 0$ for all $p \in I$ and $t \in [0, T)$. Consider $p \in I$. It follows that since $I \subseteq \mathcal{I}(X_0)$, $p(\tau(0)) = 0$. Let $p_m : \mathcal{L}^{(m)}(p)$ denote the m^{th} Lie derivative of p, and let $s_m : p_m(\tau(0))$.

We first prove by induction that $p_m \in I$ for all $m \geq 0$, and thus $s_m = 0$ for all $m \geq 0$. This holds for $m = 0$. Assuming that $p_k \in I$, we have $p_{k+1} = \mathcal{L}(p_k) \in I$. Therefore, since $I \subseteq I_0$, it follows that $s_{k+1} = 0$.

Consider the function $f(t) = p(\tau(t))$. We have $\frac{df}{dt} = g(t) = \mathcal{L}(p)(\tau(t))$. The function $f(t)$, satisfies the differential equation $\frac{df}{dt} = g(t)$. We can show that $g(t) = \mathcal{L}(p)(\tau(t))$ is continuous (in fact it is analytic around $t = 0$). First of all, $f(0) = 0, g(0) = 0$ and all derivatives of $g(t)$ vanish at $t = 0$. Secondly, by the continuity of $g(t)$, the differential equation for $f(t)$ must have a unique solution in the entire interval $[0, T)$. We know that $f(t) = 0$ is one possible solution for $t \in [0, T)$. By uniqueness, it must be the only possible solution in the interval $[0, T)$. As a result we conclude that $p(\tau(t)) = 0$ for all $t \in [0, T)$.

Therefore, $\tau(t) \in \mathcal{V}(I)$ for each $t \in [0, T)$. □

Example 2.4. *Returning to Ex. 2.3, $\langle\langle p_1^2 + p_2^2 + q_1^2 q_2^2 - 4\rangle\rangle$ is an invariant ideal. Its value as well as that of its Lie derivative are both zero at the initial state of the system.*

3. INVARIANT IDEALS

In this section, we present techniques for computing invariant ideals using a characterizations of such ideals as the fixed point under a suitably defined *refinement operator*.

3.1 Fixed-Point Characterization

Let $A : (X_0, F)$ be an algebraic system and I be an invariant ideal for A. We define a monotonic operator over the lattice of ideals such that any invariant ideal I can be seen as a the pre fixed point of this operator.

Def. 3.1 (Refinement Operator). *Let I be an ideal and A be an algebraic system. The* refinement *of the ideal I w.r.t A is defined as:*

$$\partial_A(I) :\ \mathcal{I}(X_0) \cap \{p \in I \mid \mathcal{L}(p) \in I\}\,.$$

In other words, the refinement operator intersects $\mathcal{I}(X_0)$ with the set of those polynomials in I whose Lie derivatives w.r.t F also lie in I.

An ideal I is a *pre fixed point* of ∂_A iff $I \subseteq \partial_A(I)$. We show that the refinement operator is a monotone map over the lattice of ideals ordered by inclusion and furthermore, any pre fixed point of this operator ∂_A is an invariant of the system A.

Lemma 3.1. *For an ideal I, its refinement $\partial_A(I)$ is also an ideal.*

PROOF. It suffices to prove that the set

$$I' = \{p \in I \mid \mathcal{L}(p) \in I\}\,,$$

is an ideal and note that ideals are closed under intersections.

First of all, we note that $0 \in I'$. Secondly, if $p_1, p_2 \in I'$ then $p_1 + p_2 \in I$ and furthermore, $\mathcal{L}(p_1 + p_2) = \mathcal{L}(p_1) + \mathcal{L}(p_2) \in I$. Therefore, $p_1 + p_2 \in I'$.

Finally, for each $p \in I'$, we wish to show that $g \cdot p \in I'$ for $g \in K[\vec{x}]$. First, we observe that $g \cdot p \in I$. Using the product rule for Lie derivatives, $\mathcal{L}(gp) = p\mathcal{L}(g) + g\mathcal{L}(p)$ Note that, both the terms belong to the ideal I and therefore $\mathcal{L}(gp) \in I$. As a result, $gp \in I'$. □

We now relate invariant ideals to pre fixed points of ∂_A.

Theorem 3.1. *An ideal I is invariant for a system A iff $I \subseteq \partial_A(I)$.*

PROOF. [⇐] We will establish that (a) $I \subseteq \mathcal{I}(X_0)$, and (b) $\forall p \in I, \mathcal{L}(p) \in I$. Note that,

$$\begin{aligned} I &\subseteq \partial_A(I) \\ &\subseteq \mathcal{I}(X_0) \cap \{p \in I \mid \mathcal{L}(p) \in I\} \end{aligned}$$

We deduce that $I \subseteq \mathcal{I}(X_0)$ and also $I \subseteq \{p \in I \mid \mathcal{L}(p) \in I\}$. As a result, we conclude that I is an invariant ideal of A.

[⇒] Similarly, if $\forall p \in I, \mathcal{L}(p) \in I$, then $I \subseteq \{p \in I \mid \mathcal{L}(p) \in I\}$. This combined with the fact that $I \subseteq \mathcal{I}(X_0)$ completes the proof. □

Finally, we establish the monotonicity of ∂_A.

Lemma 3.2. *The refinement operator ∂_A is monotonic in the lattice of ideals ordered by set inclusion.*

PROOF. Let $I_1 \subseteq I_2$, we would like to establish that $\partial_A(I_1) \subseteq \partial_A(I_2)$. This follows directly from the observation that

$$\{p \in I_1 \mid \mathcal{L}(p) \in I_1\} \subseteq \{p \in I_2 \mid \mathcal{L}(p) \in I_2\}\,.$$

□

The monotonicity of the operator ∂_A allows us to apply the Tarski-Knaster fixed point theorem to deduce the existence of a maximal invariant ideal w.r.t set inclusion.

Theorem 3.2. *There exists a maximal fixed point ideal I^* such that $\partial_A I^* = I^*$.*

PROOF. The operator ∂_A is monotone over ideals I. The lattice of ideals is closed under infinite intersection. Applying Tarski-Knaster theorem we conclude the existence of a maximal fixed point I^*. □

Note that the maximal fixed point ideal is the strongest possible invariant for the given system A. We first demonstrate an algebraic technique to compute the refinement operator ∂_A. This enables us to carry out Tarski iteration $I_0 : K[\vec{x}_0]$, $I_1 : \partial_A(K[\vec{x}_0])$, ... to compute I^*. However, this iteration may not necessarily converge to a fixed point in finitely many steps.

Therefore, we present a relaxation of the refinement operator using the notion of degree-bounded *pseudo-ideals* due to M. Colón [4]. Using pseudo-ideals, we will provide an efficient as well as convergent scheme that for computing an invariant ideal $I \subseteq I^*$. In practice, we combine exact refinement with pseudo-ideal relaxation in order to obtain a powerful technique for discovering algebraic invariants.

3.2 Computing Refinement

In this section, we present the *exact* scheme for computing the refinement operator. Our approach first computes a finitely generated Syzygy module (informally, a "*vector-space*" over the ring of polynomials naturally derived from an ideal, Cf. [1, 6]). Given an ideal $I = \langle\!\langle p_1, \ldots, p_m \rangle\!\rangle$, our approach for computing $\partial_A(I)$ is as follows

1. We first characterize a *module* $G(I)$ defined as follows:

$$G(I) = \{(g_1, \ldots, g_m) \mid \sum_{j=1}^{m} g_j \mathcal{L}(p_j) \in I\}\,.$$

   We show that $G(I)$ is a finitely generated module, obtained by projecting the basis of the *Syzygy module* of the ideal $\overline{I} = \langle\!\langle \mathcal{L}(p_1), \ldots, \mathcal{L}(p_m), p_1, \ldots, p_m \rangle\!\rangle$.

2. Given the matrix of polynomials $H \in K[\vec{x}]^{k \times m}$, whose rows represent the generators of the module $G(I)$, we compute

$$\partial_A(I) : \mathcal{I}(X_0) \cap \left\langle\!\!\left\langle H. \begin{pmatrix} p_1 \\ \vdots \\ p_m \end{pmatrix} \right\rangle\!\!\right\rangle\,.$$

Derivative Module: Let I be an ideal and F a polynomial vector field. The *derivative module* of the ideal I w.r.t F is defined as follows:

$$G(I) = \{(g_1, \ldots, g_m) \mid \sum_{j=1}^{m} g_j \mathcal{L}(p_j) \in I\}\,.$$

We now show that module $G(I) \subseteq K[\vec{x}]^m$ completely characterizes $\partial_A(I)$. Furthermore, we show that it is finitely generated. For the ensuing discussion, let $I = \langle\!\langle p_1, \ldots, p_m \rangle\!\rangle$ and I' be the set

$$I' : \{p \in I \mid \mathcal{L}(p) \in I\}\,.$$

We note that $\partial_A(I) = \mathcal{I}(X_0) \cap I'$. The following lemma provides a link between the derivative module $G(I)$ and the ideal I'.

Lemma 3.3. $p \in I'$ *iff* $\exists (g_1, \ldots, g_m) \in G(I),\ p = \sum_j g_j p_j$.

PROOF. Let $p = \sum_{i=1}^{m} h_i p_i \in I'$. By definition, $p \in I$ and $\mathcal{L}(p) \in I$. Our goal is to show that $p \in I' \iff (h_1, \ldots, h_m) \in G(I)$. Note that by product rule

$$\mathcal{L}(p) = \sum_{i=1}^{m} \mathcal{L}(h_i) p_i + \sum_i h_i \mathcal{L}(p_i)\,.$$

Since $\sum_{i=1}^{m} \mathcal{L}(h_i) p_i \in I$. Therefore, $\mathcal{L}(p) \in I$ iff $\sum_i h_i \mathcal{L}(p_i) \in I$. By definition of $G(I)$, we note that this is equivalent to requiring $(h_1, \ldots, h_m) \in G(I)$. □

We now demonstrate that $G(I)$ can be computed and represented using a finite set of generators. This can be proved readily using non-constructive techniques along the lines Hilbert's basis theorem, as shown below. We provide a constructive technique for computing $G(I)$ through Syzygy modules.

Lemma 3.4. *The module $G(I)$ is finitely generated.*

PROOF. We first note that $G(I)$ is a submodule of $K[\vec{x}]^m$, since

$$\forall p \in K[\vec{x}], (pg_1, \ldots, pg_m) \in G(I)\,.$$

The result follows from the observation that $K[\vec{x}]$ is a Noetherian ring and a theorem in commutative algebra that states that any submodule of B^m for a Noetherian ring B is finitely generated (here $B = K[\vec{x}]$) [1]. □

The generators of $G(I)$ can be computed by using the standard techniques for computing *Syzygies*. Let S be the generators for the syzygy module corresponding to the ideal:

$$\overline{I} : \langle\!\langle \mathcal{L}(p_1), \ldots, \mathcal{L}(p_m), p_1, \ldots, p_m \rangle\!\rangle\,.$$

As a result,

$$S = \{(g_1, \ldots, g_{2m}) \mid \sum_{i=1}^{m} g_i \mathcal{L}(p_i) + \sum_{i=1}^{m} g_{i+m} p_i = 0\,.\}$$

Let $\overline{S}$ be obtained by projecting m components away from S:

$$\overline{S} = \{(g_1, \ldots, g_m) \mid (g_1, \ldots, g_m, g_{m+1}, \ldots, g_{2m}) \in S\}\,.$$

The generators of the module $\overline{S}$ are obtained by projecting the last m components away from the generators of S.

Theorem 3.3. $G(I) \equiv \overline{S}$.

PROOF. Let $(g_1, \ldots, g_m) \in \overline{S}$. It follows that there exists a set of polynomials $(g_1, \ldots, g_m, g_{m+1}, \ldots, g_{2m})$ such that

$$\sum_{j=1}^{m} g_i \mathcal{L}(p_i) + \sum_{j=1}^{m} g_{i+m} p_i = 0\,.$$

As a result, we conclude that $\sum_i g_i \mathcal{L}(p_i) \in I$ and thus $(g_1, \ldots, g_m) \in G(I)$. □

Input: $A : \langle \mathcal{I}(X_0), F \rangle$ (Algebraic System), $I : \langle\!\langle p_1, \ldots, p_m \rangle\!\rangle$ (Ideal)
Result: $\partial_A I$
begin
$I_1 \leftarrow (\mathcal{L}(p_1), \ldots, \mathcal{L}(p_m), p_1, \ldots, p_m)$
$M_1 \leftarrow \mathsf{SyzygyModuleGenerators}(I_1)$
$M \leftarrow \mathsf{ProjectOutColumns}(M_1, m+1, 2m)$
$I' \leftarrow \mathsf{MatrixVectorProduct}(M, (p_1, \ldots, p_m))$
$\partial_A(I) \leftarrow \mathsf{IdealIntersection}(I', \mathcal{I}(X_0))$
end

Algorithm 1: ComputeRefinement

Computational Complexity. The computational complexity of the refinement step shown in Algorithm 1 depends on the number of variables in the system A and the degree of the vector field. In practice, the computation of Syzygy bases require an expensive Gröbner basis computation, whose complexity is not well understood, in general. Similarly, the ideal intersection is also based on a Gröbner basis computation.

As mentioned earlier, the downward Tarski iteration on the lattice of ideals does not converge since ideals over $K[\vec{x}]$ do not exhibit the descending chain condition unless $K[\vec{x}]$ is an *Artinian* ring. In practice, the rings $Q[\vec{x}], R[\vec{x}], C[\vec{x}]$ of rational, reals and complex polynomials are not Artinian.

We now present a convergent technique based on a relaxation of the refinement procedure to consider degree bounded *pseudo ideals*. Such a relaxation also yields a convergence guarantee for the Tarski iteration as well as providing an efficient refinement operator. Furthermore, the techniques developed in this section prove to be quite useful as a starting point for the pseudo ideal relaxation.

4. INVARIANT PSEUDO IDEALS

In this section, we present our technique over the domain of *pseudo ideals*. The notion of pseudo ideals was originally formulated by M. Colón [4] in order to generate invariants for programs. We first recall some basic properties of pseudo ideals.

4.1 Pseudo Ideals

Let $K_d[\vec{x}] \subseteq K[\vec{x}]$ denote the set of polynomials p such that $\mathsf{degree}(p) \leq d$. The set $K_d[\vec{x}]$ can be viewed as a vector space generated by a basis set consisting of all monomials over $\vec{x}$ whose degrees are at most d.

Def. 4.1 (Pseudo Ideal). *The* pseudo ideal *generated by a finite set of polynomials $P = \{p_1, \ldots, p_m\}$ using multipliers with degree bound d is given by:*

$$\text{PSEUDO-IDEAL}_d(P) = \{\sum_i g_i p_i \mid g_i \in K_d[\vec{x}]\}\,.$$

Thus $\text{PSEUDO-IDEAL}_d(P)$ *consists of polynomial combinations of the elements in P using* multipliers *drawn from $K_d[\vec{x}]$.*

A pseudo ideal differs from an ideal as follows: whereas an ideal generated by P considers polynomial combinations of the elements of P using arbitrary polynomial multipliers drawn from $K[\vec{x}]$, a pseudo ideal restricts the multipliers to a degree bounded set $K_d[\vec{x}]$. The basic properties of pseudo ideals can be formulated clearly once we establish that pseudo ideals form a *vector space* over K.

Lemma 4.1. *For any finite set of polynomials $P = \{p_1, \ldots, p_m\}$, the pseudo ideal* $\text{PSEUDO-IDEAL}_d(P)$ *is a vector space over K, whose dimension is at most $\binom{n+d}{d}^m$, where $n = |\vec{x}|$.*

PROOF. Note that if $p_1, \ldots, p_l \in \text{PSEUDO-IDEAL}_d(P)$ then for any $\lambda_1, \ldots, \lambda_l \in K$, $\sum_i \lambda_i p_i \in \text{PSEUDO-IDEAL}_d(P)$. As a result, $\text{PSEUDO-IDEAL}_d(P)$ forms a vector space over k.

Each element p of $\text{PSEUDO-IDEAL}_d(P)$ is represented by some tuple $\langle g_1, \ldots, g_m \rangle \in K_d[\vec{x}]^m$, such that $\sum_{i=1}^m g_i p_i = p$. The dimension of the vector space is bounded by $(\mathsf{dim}(K_d[\vec{x}]))^m$, which yields the required upper bound. □

Example 4.1. *Consider the set of polynomials $P = \{x_1^2 - 1, x_2^2 - 1, x_3^2 - 1\}$. The parametric form (template polynomial) $a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + b_i$ for $i = 1, 2, 3$ represents the unknown multipliers from $k_1[x_1, x_2, x_3]$. As a result, any polynomial in* $\text{PSEUDO-IDEAL}_1(P)$ *can be written as*

$$\begin{array}{rcl} p & = & \sum_{i=1}^3 (a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + a_{i4})(x_i^2 - 1) \\ & = & a_{11}x_1^3 + a_{12}x_1^2x_2 + a_{13}x_1^2x_3 + a_{14}x_1^2 \\ & + & a_{21}x_1x_2^2 + a_{22}x_2^3 + a_{23}x_2^2x_3 + a_{24}x_2^2 \\ & + & a_{31}x_1x_3^2 + a_{32}x_2x_3^2 + a_{33}x_3^3 + a_{34}x_3^2 \\ & - & (a_{11} + a_{21} + a_{31})x_1 - (a_{12} + a_{22} + a_{32})x_2 \\ & - & (a_{13} + a_{23} + a_{33})x_3 \; - \; (a_{14} + a_{24} + a_{34}) \end{array}$$

Def. 4.2 (Pseudo-Ideal Degree). *The* pseudo ideal degree *of I :* $\text{PSEUDO-IDEAL}_d(p_1, \ldots, p_m)$ *is defined as*

$$\mathsf{degree}(I) : \; d + \max\{\mathsf{degree}(p_1), \ldots, \mathsf{degree}(p_m)\}\,.$$

The degree of a pseudo ideal places an upper limit on the degrees of the polynomials in it. Also, $\mathsf{degree}(\text{PSEUDO-IDEAL}_d(P)) > d$, unless all polynomials in P are of degree 0.

Representation: Any pseudo ideal $I :$ $\text{PSEUDO-IDEAL}_d(P)$ can be represented implicitly as a parametric polynomial form $\pi[\vec{c}]$ whose degree coincides with that of I, along with linear constraints on the coefficients of the parametric form:

$$\pi[\vec{c}] : \; \left\{ \sum_\alpha c_\alpha \vec{x}^\alpha \mid M\vec{c} = 0 \right\}\,.$$

In practice, many of the constraints on c_α are of the form $c_\alpha = 0$. We, therefore, optimize the representation of pseudo ideals by removing the corresponding monomials from the parametric polynomials, thus retaining a list of monomials that are part of the parametric polynomial and linear constraints involving their coefficients.

Lemma 4.2. *A set of polynomials I is a pseudo ideal iff there exists parametric polynomial $\pi[\vec{c}]$ and a matrix M such that:*

$$I : \; \{\pi[\vec{c}] \mid M\vec{c} = 0\}\,.$$

PROOF. Let $\vec{c}_1, \ldots, \vec{c}_N$ generate the kernel of the matrix M and $p_i = \pi[\vec{c}_i]$ be a polynomial. Thus, $I = \text{PSEUDO-IDEAL}_0(p_1, \ldots, p_N)$. Conversely, let $I = \text{PSEUDO-IDEAL}_d(P)$ and $D = \mathsf{degree}(I)$. Since I is a vector space, we may conceive of I as a subspace of $K_D[\vec{x}]$. Let $\pi[\vec{c}]$ be the generic polynomial form of degree at most D with coefficients $\vec{c}$. We can express the pseudo ideal I, a subspace of $K_D[\vec{x}]$, as the kernel of a matrix M. □

We provide an example of a parametric representation.

Example 4.2. $\text{PSEUDO-IDEAL}_1(P)$ *from Example 4.1 can be represented using a parametric polynomial represented implicitly here for lack of space:*

$$\pi : \; \sum \; c_{i,j,k}\vec{x}^{(i,j,k)} \; \textit{s.t.} \; \left[\begin{array}{l} i,j,k \geq 0, i+j+k \leq 3, \\ (i,j,k) \neq (1,1,1) \end{array} \right]\,.$$

By convention, let the coefficient c_α in a template correspond to the monomial $\vec{x}^\alpha$. Note that there is no term in π corresponding

to $x_1x_2x_3$ (i.e., $c_{1,1,1} = 0$). The following constraints on $\vec{c}$ define PSEUDO-IDEAL$_1(P)$*:*

$$\begin{array}{rcl} c_{0,0,0} + c_{2,0,0} + c_{0,2,0} + c_{0,0,2} & = & 0 \\ c_{1,0,0} + c_{3,0,0} + c_{2,1,0} + c_{2,0,1} & = & 0 \\ c_{0,1,0} + c_{1,2,0} + c_{0,3,0} + c_{0,2,1} & = & 0 \\ c_{0,0,1} + c_{1,0,2} + c_{0,1,2} + c_{0,0,3} & = & 0 \end{array}$$

In practice, our data structures currently list the set of terms of the parametric polynomials explicitly, storing the constraints in a matrix.

Membership Testing: Testing whether polynomial p belongs to J : PSEUDO-IDEAL$_d(P)$ is performed by checking if the coefficients of p satisfy the linear constraints $M\vec{c} = 0$, corresponding to J. Note that if $\mathsf{degree}(p) > \mathsf{degree}(J)$, then we may conclude that $p \notin J$.

Intersection: Informally, intersection of two pseudo ideals P_1, P_2 is performed by conjoining their constraints $M_1\vec{c} = 0$ and $M_2\vec{c} = 0$. In general, however, the monomial terms involved in the parametric representations of P_1 and P_2 differ. Therefore, (A) we first compute a common set of monomials that occur both in P_1 and P_2's parametric form. (B) For $P_1(P_2)$ we drop the coefficients corresponding to monomials that do not appear in P_2(resp. P_1) by setting them to zero. This corresponds to removing the corresponding columns from the matrix M_1(resp. M_2) to obtain a matrix M_1' (resp. M_2'). Furthermore, the monomial terms corresponding to these columns are eliminated. (C) Finally, we conjoin the two matrices M_1' and M_2' as

$$M_{P_1 \cap P_2} : \left[\begin{array}{c} M_1' \\ M_2' \end{array} \right] .$$

Example 4.3. *We recall the pseudo ideal* PSEUDO-IDEAL$_1(P)$ *from Exs. 4.1 and 4.2. The degree* 1 *pseudo ideal* $Q = (x_1 - 1, x_2 - 1, x_3 + 1)$ *is represented by a generic degree two polynomial* $\sum_{i+j+k\leq 2} d_{i,j,k}x_1^i x_2^j x_3^k$ *with the following constraint on its coefficients:*

$$\begin{array}{rcl} d_{1,0,0} + d_{0,1,0} + d_{0,0,1} & = & d_{0,0,0} + d_{2,0,0} + d_{0,2,0} + \\ & & d_{0,0,2} + d_{1,1,0} + d_{0,1,1} - d_{1,0,1} \end{array} .$$

PSEUDO-IDEAL$_1(P) \cap$PSEUDO-IDEAL$_1(Q)$ *is computed as follows:* PSEUDO-IDEAL$_1(P)$ *has a higher degree template polynomial. We therefore zero away all variables with degree* 3 *or above. This corresponds to removing the corresponding coefficient variables from the constraints for* PSEUDO-IDEAL$_1(P)$*. Finally, we conjoin the two sets of constraints and obtain a pseudo ideal represented by a generic template of degree two*

$$\pi_{12} = e_{0,0,0} + e_{2,0,0}x_1^2 + e_{0,2,0}x_2^2 + e_{0,0,2}x_3^2 .$$

with the following constraint on its coefficients:

$$e_{0,0,0} + e_{2,0,0} + e_{0,2,0} + e_{0,0,2} = 0$$

This is, in fact, a representation of PSEUDO-IDEAL$_0(P)$.

Lemma 4.3. *Let I_1, I_2 be two pseudo ideals. It follows that* $\mathsf{degree}(I_1 \cap I_2) \leq \min(\mathsf{degree}(I_1), \mathsf{degree}(I_2))$.

Refinement: We discuss the computation of a refinement operator over pseudo ideals. Let I be a pseudo-ideal of degree d_I. Let $\pi[\vec{c}]$ be the parametric form associated with I and $M\vec{c} = 0$ be the constraints on the coefficients of $\pi[\vec{c}]$. We seek to compute

$$\partial_F(I) : \{p \in I \mid \mathcal{L}_F(p) \in I\} ,$$

wherein the operator $\partial_F(I)$ is now interpreted over sets of polynomials that are pseudo-ideals instead of ideals. Refinement proceeds as follows:

1. We first compute the parametric form $\pi' : \mathcal{L}_F(\pi[\vec{c}])$. This will result in a polynomial whose coefficients are linear expressions over $\vec{c}$.
2. We derive constraints over $\vec{c}$ to ensure that $\mathcal{L}_F(\pi) \in I$. Let $M'\vec{c} = 0$ be the constraints thus derived.
3. We conjoin $M'\vec{c} = 0$ with the original constraints $M\vec{c} = 0$ and simplify based on these new constraints.

Example 4.4. *Consider the pseudo ideal* PSEUDO-IDEAL$_0(x_1^2 - 1, x_2^2 + 2x_1 - 2x_1^2)$ *represented using parametric polynomial*

$$\pi[\vec{c}] : c_{2,0}x_1^2 + c_{0,2}x_2^2 + c_{1,0}x_1 + c_{0,0}$$

with constraints

$$M\vec{c} : \left[\begin{array}{rcl} c_{2,0} + c_{1,0} + c_{0,0} & = & 0 \\ c_{1,0} - 2c_{0,2} & = & 0 \end{array} \right.$$

Let F be the vector field $(x_2x_1, -x_1)$. The Lie derivative of π is

$$\begin{array}{rcl} \pi' & = & 2c_{2,0}x_1(x_2x_1) + 2c_{0,2}x_2(-x_1) + c_{1,0}(x_2x_1) \\ & = & 2c_{2,0}x_1^2x_2 + (c_{1,0} - 2c_{0,2})x_1x_2 \end{array}$$

We would like $\pi' \in I$. Note that the terms $x_1^2x_2$ and x_1x_2 do not exist in $\pi[\vec{c}]$. As a result, we obtain the constraints:

$$c_{2,0} = 0 \ \wedge \ c_{1,0} - 2c_{0,2} = 0 .$$

The refinement can be expressed using the form π, satisfying these constraints as well as the original constraints over $\pi[\vec{c}]$. After simplification, we obtain

$$\pi_1 : \ c_{0,2}x_2^2 + c_{1,0}x_1 + c_{0,0} .$$

Note that the term for x_1^2 is no longer present due to the constraint $c_{2,0} = 0$. The constraints on the coefficients after simplification are:

$$c_{1,0} - 2c_{0,2} = 0 \ \wedge \ c_{1,0} + c_{0,0} = 0 .$$

This is in fact, PSEUDO-IDEAL$_0(x_2^2 + 2x_1 - 2)$.

Lemma 4.4. *If I is a pseudo ideal, then the set*

$$\partial_F(I) : \{p \in I \mid \mathcal{L}_F(p) \in I\}$$

is also a pseudo ideal.

PROOF. After computing the parametric representation of I, the procedure for refinement yields a parametric form π_1 with constraints on its coefficients. This is a pseudo ideal following Lemma 4.2. □

We now state the theorem guaranteeing convergence of the overall Tarski iteration over pseudo ideals.

Lemma 4.5. *Consider an infinite sequence of pseudo ideals $I_1 \supseteq I_2 \supseteq I_3 \cdots$. There exists a limit $N > 0$ s.t. $I_N = I_{N+1} = \cdots$. In other words, any descending chain of pseudo ideals converges to a limit.*

PROOF. Pseudo ideals are in fact finite dimensional vector spaces. If $I_j \supset I_{j+1}$ then the dimension of I_{j+1} is strictly lower than that of I_j. As a result the descending chain of pseudo ideals converges in finitely many steps. □

Finally, we show that the fixed point obtained is an invariant ideal. Before doing so, we first relate pseudo ideals to ideals.

Lemma 4.6. *Given finite set $P \subseteq K[\vec{x}]$,* PSEUDO-IDEAL$_d(P) \subseteq \langle\langle P \rangle\rangle$.

Theorem 4.1. *Let* PSEUDO-IDEAL$_d(P)$ *be a pre-fixed point over pseudo ideals. It follows that $\langle\langle P \rangle\rangle$ is an invariant ideal.*

PROOF. Let $P = \{p_1, \ldots, p_k\}$. We note that each of the generators of $\mathcal{I}(X_0)$ belong to PSEUDO-IDEAL$_d(P)$ and thus to $\langle\langle P \rangle\rangle$. As a result, $\mathcal{I}(X_0) \subseteq \langle\langle P \rangle\rangle$. By the property of pseudo ideal refinement, for each $p \in P$, $\mathcal{L}(p) \in$ PSEUDO-IDEAL$_d(P)$. As a result, $\mathcal{L}(p_1), \ldots, \mathcal{L}(p_k) \in \langle\langle P \rangle\rangle$. Therefore, for every $p = \sum_i g_i p_i \in \langle\langle P \rangle\rangle$, we have $\mathcal{L}(p) = \sum_i g_i \mathcal{L}(p_i) + \sum_i \mathcal{L}(g_i) p_i \in \langle\langle P \rangle\rangle$. □

Complexity: Let N be the dimension of PSEUDO-IDEAL$_d(I_0)$ at the start of the iteration. Each pseudo ideal encountered is, in general, a subspace of N represented by a $N \times N$ matrix. As a result, each iteration requires a refinement followed by intersection, requiring time $O(N^2)$. We may iterate for at most N steps, leading to a $O(N^3)$ complexity. On the other hand, checking for convergence is also a $O(N^3)$ operation. If we repeatedly check for convergence, we have a $O(N^4)$ worst case, where $N = O(\binom{n+d}{d}^m)$. In practice, we maintain d to be small $d = 0, 1, 2$, and observe convergence in number of steps much smaller than N.

5. EXTENSIONS TO HYBRID SYSTEMS

We have thus far presented a technique for generating invariants of continuous systems with polynomial ODEs. In this section, we extend our discussion to hybrid systems with discrete modes and transitions between them.

Continuous Systems with Constraints: We first consider an extension to continuous algebraic systems $\langle X_0, F \rangle$ in order to consider the effect of *holonomic constraints* on the state-space. In other words, the evolution of the system is constrained to remain inside a domain X. We assume that $X \subseteq \mathcal{R}^n$ is an algebraic variety whose corresponding ideal is $\mathcal{I}(X)$.

Def. 5.1 (Constrained Algebraic System). *A constrained algebraic system consists of $\langle F, X_0, X \rangle$ wherein F is a polynomial vector field, X_0 is a variety describing possible initial states and X is an algebraic* constraint *(commonly termed an invariant).*

The semantics of the system are modified to ensure that all time trajectories $\tau : [0, T) \mapsto \mathcal{R}^n$ satisfy the condition $\tau(t) \in X$ for all $t \in [0, T)$. Naturally, any invariant of such a system will also, in general, be subsumed by X. Let ∂_F be a refinement operator over ideals (resp. pseudo ideals) defined for the system $\langle F, X_0 \rangle$ in the absence of any constraints. In the presence of constraint set X represented by ideal $\mathcal{I}(X)$, the iteration scheme is modified as follows:

$$I_{n+1} = \mathcal{F}(I_n) = (\mathcal{I}(X_0) \cap \partial_F(I_n)) \oplus \mathcal{I}(X) \qquad (1)$$

wherein $\oplus$ denotes the ideal addition (i.e, the union of generators of the ideal), representing the intersection of the corresponding algebraic varieties. Over pseudo ideals, we may interpret $\oplus$ as the vector space addition of the two subspaces represented by the arguments.

Theorem 5.1. *Let $\mathcal{F}$ denote the operator in Eq. 1. The following facts hold about $\mathcal{F}$ both over ideals as well as pseudo ideals.*

Monotonicity: *$I \subseteq J$ then $\mathcal{F}(I) \subseteq \mathcal{F}(J)$.*

Inclusion: *$\mathcal{F}(J) \subseteq \mathcal{I}(X)$.*

5.1 Hybrid Systems

We now extend our notions to hybrid systems.

Def. 5.2 (Algebraic Hybrid System). *An algebraic hybrid system is a tuple $\langle \mathcal{S}, \mathcal{T} \rangle$, wherein $\mathcal{S} = \{S_1, \ldots, S_k\}$ consists of k discrete modes and $\mathcal{T}$ denotes discrete transitions between the modes. Each mode $S_i \in \mathcal{S}$ consists of an algebraic system $\langle X_{0,i}, F_i, X_i \rangle$.*

Each transition $\tau : \langle S_i, S_j, P_{ij} \rangle \in \mathcal{T}$ consists of an edge $S_i \rightarrow S_j$ along with an algebraic transition relation *$P_{i,j}[\vec{x}, \vec{x}']$ specifying the next state $\vec{x}'$ in relation to the previous state $\vec{x}$. Note that the transition is* guarded *by the assertion $\exists \vec{x}' P_{ij}[\vec{x}, \vec{x}']$.*

Discrete transitions are treated in the process of generating invariants using the post-condition operator.

Def. 5.3 (Post-Conditions). *The post condition of a (pseudo) ideal I_i over a transition $\tau : \langle S_i \rightarrow S_j, P_{ij}[\vec{x}, \vec{x}'] \rangle$ is defined as:*

$$post(I_i, \tau) : \ (\exists \vec{x}) \ [I_i[\vec{x}] \oplus P_{ij}[\vec{x}, \vec{x}']] \, .$$

We recall that the operation $\oplus$ over ideals represents the intersection of the associated variety and is computed by combining the generators of I_i and P_{ij}. The elimination of the variables $\vec{x}$ from the resulting ideal is performed by computing the Groebner basis using an *elimination ideal* [6]. The computation of post conditions over pseudo ideals is described in detail elsewhere [4].

Thus far, we have computed invariant ideals for continuous algebraic systems as fixed points over ideals as well as pseudo ideals. For the case of hybrid systems with multiple modes, our goal is to compute multiple invariant ideals, one for each mode of the system. Therefore, we lift our notions from invariant (pseudo) ideals to a map that associates (pseudo) ideals to each mode $S \in \mathcal{S}$: $\eta : \mathcal{S} \mapsto \mathcal{P}(K[\vec{x}])$, s.t. $\eta(S_i) \subseteq K[\vec{x}]$. The notion of an invariant (pseudo) ideal for an algebraic system is extended to an invariant (pseudo) ideal map.

Def. 5.4 (Invariant Ideal Map). *A map $\eta : \mathcal{S} \mapsto \mathcal{P}(K[\vec{x}])$ is an invariant map iff the following facts hold:*

Initiation and Mode Constraints: *$\forall S_i : \langle X_{0,i}, F_i, X_i \rangle$, we have $\mathcal{I}(X_i) \subseteq \eta(S_i) \subseteq \mathcal{I}(X_{0,i})$. In other words, the invariant associated with mode S_i must respect the initial condition and constraints at S_i.*

Continuous Sub-system: *$\forall S_i : \langle X_{0,i}, F_i, X_i \rangle$, the ideal $\eta(S_i)$ is a fixed point w.r.t F_i: $\eta(S_i) \subseteq [\mathcal{I}(X_{0,i}) \cap \partial_{F_i} \eta(S_i)] \oplus \mathcal{I}(X_i)$.*

Discrete Transitions: *$\forall \tau : \langle S_i, S_j, P_{ij} \rangle$, the ideals $\eta(S_i)$ and $\eta(S_j)$ must satisfy* consecution*: $\eta(S_j) \subseteq post(\eta(S_i), \tau)$.*

In order to compute the invariant map, we start with an initial map $\eta^{(0)}$ such that $\eta^{(0)}(S_i) = K[\vec{x}]$, and update using the rule $\eta^{(i+1)} = \mathcal{G}(\eta^{(i)})$ such that

$$\eta^{(i+1)}(S) = \mathcal{I}(X_i) \oplus \left[\begin{array}{l} \bigcap_{\tau : S' \rightarrow S} \underbrace{post(\eta^{(i)}(S'), \tau)}_{\text{Discrete Transition}} \\ \cap \underbrace{\mathcal{I}(X_{0,i}) \cap \partial_{F_i}(\eta^{(i)}(S))}_{\text{Continuous System}} \end{array} \right]$$

The extensions to pseudo ideals proceeds along similar lines. The initial map is set to $\eta^{(0)}(S_i) = K_d[\vec{x}]$ for the case of iteration over pseudo ideals.

6. EXPERIMENTS

In this section, we describe our prototype implementation and some results on some interesting non-linear systems.

Table 1: Results of various runs of our technique. Note: Ideal Iter: number of initial refinement steps, Pseudo Degree: degree of pseudo ideal, Steps: number of steps taken to converge, # Inv: number of generators in invariant ideal.

System	Var	Ideal Iter	Pseudo Degree	Steps	Time (sec)	# Inv
Volterra-3D	3	1	1	3	1.1	4
Coup-Spring	5	1	2	23	21	4
Collision2	12	1	1	3	570	10
Collision3	16	1	0	2	4	4
Collision3	16	2	0	3	196	14
Collision3	16	1	1	6	372	15
Collision3	16	3	0	3	12900	13

6.1 Implementation.

The techniques described here have been implemented inside Mathematica(tm) for finding invariants of continuous systems, using Singular package interface to Mathematica(tm). We have implemented the Syzygy-based refinement procedure as well as the one based on pseudo ideals. In practice, our iteration scheme consists of running a small and fixed number of iterations of the exact refinement operator. The resulting generators form the generators of a pseudo ideal for some fixed degree d. The iteration is then carried out further over the lattice of pseudo ideals until convergence. The soundness of this scheme follows from the fact that $\partial_A^m(I) \subseteq I(X_0)$ for all I and from the soundness of the pseudo ideal fixed point. The Mathematica(tm) implementation along with the systems analyzed and invariants computed will be made available on-line[1].

6.2 Experiments

We now present the results obtained over some interesting benchmark systems. Table 1 summarizes the parameters used in our executions and the performance of our technique at a glance.

3D-Lotka-Volterra System: We considered the following $3D$ Lotka Volterra System over variables x, y, z:

$$\mathfrak{F}(x,y,z) : \ (xy - xz, yz - yx, zx - zy) \, .$$

The initial states lie over the vertices of an unit cube:

$$X_0 : \ (x^2 - 1, y^2 - 1, z^2 - 1) \, .$$

The initial pseudo ideal involved 18 unknown parameters in our initial template. We obtained the following invariants:

$$\begin{array}{ll} p_1: & -1 - 2yz - y^2z^2 + y^4z^2 + 2y^3z^3 + y^2z^4, \\ p_2: & -x - y + y^2z + xz^2 + 3yz^2 - y^3z^2 + z^3 - 2y^2z^3 - yz^4, \\ p_3: & -x + xy^2 + y^3 - z + 3y^2z - y^4z + yz^2 - 2y^3z^2 - y^2z^3, \\ p_4: & -3 + x^2 - y^2 - 4yz + 2y^3z - z^2 + 4y^2z^2 + 2yz^3 \end{array}$$

All of them are mutually dependent on each other. Further, attempting to obtain these using generic templates of degree 4 requires 126×4 unknowns as opposed to 18 unknowns that were used.

Coupled Spring-Mass System: Consider a mechanical system modeling the undamped/unforced oscillation of two masses coupled using springs with constants k_1, k_2 and masses m_1, m_2 tuned such that $\frac{k_1}{m_1} = \frac{k_2}{m_2} = k$. Furthermore, we assume that $m_1 = 5m_2$.

[1] Cf.\url{http://www.cs.colorado.edu/~srirams/algebraic-invariants}.

$$\begin{array}{lll} p_1 & : & 576 + 1200v_1^2 + 625v_1^4 + 2880v_1v_2 + 3000v_1^3v_2 + \\ & & 528v_2^2 + 4150v_1^2v_2^2 + 1320v_1v_2^3 + 121v_2^4 \\ & & -1860kx_2^2 + 2750kv_1^2x_2^2 + 1600kv_1v_2x_2^2 \\ & & +710kv_2^2x_2^2 + 525k^2x_2^4, \\ p_2 & : & 240x_1 + 250v_1^2x_1 + 600v_1v_2x_1 + 110v_2^2x_1 \\ & & +396x_2 - 525v_1^2x_2 - 260v_1v_2x_2 - 131v_2^2x_2 - 105kx_2^3, \\ p_3 & : & 24 + 25v_1^2 + 60v_1v_2 + 11v_2^2 + 50kx_1x_2 + 5kx_2^2 \\ p_4 & & -21 + 25v_1^2 + 10v_1v_2 + 6v_2^2 + 25kx_1^2 + 5kx_2^2 \end{array}$$

Figure 1: Invariant obtained for coupled mass spring system.

$$\begin{array}{l} \mathbf{p_1} : e_1^2 + e_2^2 - b^2, \quad \mathbf{p_2} : d_1^2 + d_2^2 - a^2 \\ \mathbf{p_3} : e_1 - r_2\theta + \theta y_2, \quad \mathbf{p_4} : -a + d_1 - r_2\omega + \omega x_2 \\ \mathbf{p_5} : b - e_2 - r_1\theta + \theta y_1, \\ \mathbf{p_6} : -br_1 + by_1 + e_1r_2 - e_1y_2 - e_2r_1 + e_2y_1 \\ \mathbf{p_7} : br_2 - by_2 - e_1r_1 + e_1y_1 - e_2r_2 + e_2y_2 \\ \mathbf{p_8} : -d_2 - r_1\omega + \omega x_1 \\ \mathbf{p_9} : ad_2r_2 - ad_2x_2 + d_1d_2r_2 - d_1d_2x_2 - r_1d_2^2r_1 + d_2^2x_1 \\ \mathbf{p_{10}} : ar_1 - ax_1 - d_1r_1 + d_1x_1 - d_2r_2 + d_2x_2 \end{array}$$

Figure 2: Invariants obtained for the two aircraft collision avoidance system.

The resulting system consists of variables $\vec{x} : (x_1, x_2, v_1, v_2, k)$ representing the displacements, velocities and the spring constant. The evolution is specified using the vector field:

$$\mathfrak{F}(\vec{x}) : \left(\ v_1, \ v_2, -kx_1 - \tfrac{k}{5}(x_1 - x_2), \ k(x_1 - x_2), \ 0 \ \right) .$$

The initial condition is set to $x_1 = x_2 = 0, v_1 = 1, v_2 = -1$. This resulted in the invariant ideal shown in Fig. 1. p_3 and p_4 are seen to be conservation laws satisfying $\frac{dp_3}{dt} = \frac{dp_4}{dt} = 0$. However, p_1, p_2 are mutually dependent on themselves as well as p_3, p_4. Finding these invariants parametrically require a degree 6 template with 4×462 unknowns. Our initial pseudo ideals involves an initial parametric form with 84 unknowns. The choice of $m_2 = 5m_1$ in this example was arbitrary. We found interesting invariants of a similar form for all other choices we experimented with including $m_1 = 2m_2, 3m_2, 11m_2, \ldots$.

Collision Avoidance Maneuvers: Finally, we consider the algebraic abstraction of the collision avoidance system analyzed recently by Platzer and Clarke [16] and much earlier by Tomlin et al. [24]. The two airplane collision avoidance system consists of the variables (x_1, x_2) denoting the position of the first aircraft, (y_1, y_2) for the second aircraft, (d_1, d_2) representing the velocity vector for aircraft 1 and (e_1, e_2) for aircraft 2. In addition, the parameters $\omega, \theta, a, b, r_1, r_2$ are also represented as system variables. The dynamics are modeled by the following differential equations:

$$\begin{array}{cccc} x_1' = d_1 & x_2' = d_2 & d_1' = -\omega d_2 & d_2' = \omega d_1 \\ y_1' = e_1 & y_2' = e_2 & e_1' = -\theta e_2 & e_2' = \theta e_1 \\ a' = 0 & b' = 0 & r_1' = 0 & r_2' = 0 \end{array}$$

We note that the form of the equations are invariant under time reversal $t \mapsto -t$. The initial set

$$\left[\begin{array}{c} x_1 = y_1 = r_1 \ \wedge \ x_2 = y_2 = r_2 \ \wedge \ d_1 = a \ \wedge \\ d_2 = 0 \ \wedge \ e_1 = b \ \wedge \ e_2 = 0 \end{array} \right]$$

represents a collision. Fig. 2 presents the invariants. The initial parametric form had 252 unknowns. $p_1, \ldots, p_5$ and p_8 are conservation laws. The remaining invariants are dependent on $p_1, \ldots, p_5$

and p_8. Our tool also was run on a larger system with 3 aircrafts consisting of 16 variables in all (removed three parameters from the model). Table 1 shows the behavior of our implementation under various values for the number of initial iterations and the starting degree of the pseudo ideal. The blowup involved in going from 2 initial iterations to 3 is interesting. The overall time required to compute three iterations of the exact refinement remains roughly 5 seconds for this case. However, the result has roughly 230 polynomials. Most of the time is spent parameterizing the initial pseudo ideal, computing its derivatives and so on. We hope to reimplement parts of our system inside C++/Java to avoid such a slowdown in the future.

Conclusions: Thus far, we have presented an invariant generation technique using a fixed point iteration over ideals and pseudo ideals. A prototype implementation of our technique has been shown to compute interesting and non-trivial invariants for systems that would currently be considered non trivial. In the future, we hope to explore extensions to compute inequality invariants as well as to integrate these techniques inside a theorem proving environment such as KeYmaera [17].

Acknowledgements: We gratefully acknowledge Prof. André Platzer for his detailed and patient answers to our queries and the anonymous reviewers for their comments and suggestions.

References

[1] William W. Adams and Philippe Loustaunau. *An Introduction to Gröbner Bases*. American Mathematical Society, 1991.

[2] Enric Rodriguez Carbonell and Ashish Tiwari. Generating polynomial invariants for hybrid systems. In *HSCC*, volume 3414 of *LNCS*, pages 590–605, 2005.

[3] Michael Colón and Henny Sipma. Synthesis of linear ranking functions. In *TACAS*, volume 2031 of *LNCS*, pages 67–81. Springer, April 2001.

[4] Michael A. Colón. Polynomial approximations of the relational semantics of imperative programs. *Science of Computer Programming*, 64(1):76 – 96, 2007. Special issue on the 11th Static Analysis Symposium - SAS 2004.

[5] Patrick Cousot and Rhadia Cousot. Abstract Interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *ACM Principles of Programming Languages*, pages 238–252, 1977.

[6] David Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction To Commutative Algebraic Geometry and Commutative Algebra*. Springer, 1993.

[7] Bruce Driver. PDE lecture notes, 2003. Cf. `http://www.math.ucsd.edu/~bdriver/231-02-03/Lecture_Notes/pde4.pdf`.

[8] Giovanni Gallo and Bud Mishra. Wu-Ritt characteristic sets and their complexity. *Dimacs series in discrete mathematics and theoretical computer science*, 6:111–131, 1991.

[9] Sumit Gulwani and George Necula. Discovering affine equalities through random interpretation. In *ACM Principles of Prog. Lang. (POPL 2003)*, pages 74–84. ACM Press, 2003.

[10] Sumit Gulwani and Ashish Tiwari. Constraint-based approach for hybrid systems. In *Computer-Aided Verification*, volume 5123 of *Lecture Notes in Computer Science*, pages 190–203, 2008.

[11] M. Karr. Affine relationships among variables of a program. *Acta Informatica*, 6:133–151, 1976.

[12] Zohar Manna and Amir Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer, New York, 1995.

[13] Nadir Matringe, Arnoldo Viera Moura, and Rachid Rebiha. Morphisms for non-trivial non-linear invariant generation for algebraic hybrid systems. In *HSCC*, volume 5469 of *LNCS*, pages 445–449, 2009.

[14] Venkatesh Mysore, Carla Piazza, and Bud Mishra. Algorithmic algebraic model checking II: Decidability of semi-algebraic model checking and its applications to systems biology. In *ATVA*, volume 3707 of *LNCS*, pages 217–233. Springer, 2005.

[15] André Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reasoning*, 41(2):143–189, 2008.

[16] André Platzer and Edmund Clarke. Computing differential invariants of hybrid systems as fixedpoints. *Formal Methods in Systems Design*, 35(1):98–120, 2009.

[17] André Platzer and Jan-David Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008.

[18] Andreas Podelski and Silke Wagner. A sound and complete proof rule for region stability of hybrid systems. In *HSCC*, volume 4416 of *Lecture Notes in Computer Science*, pages 750–753. Springer, 2007.

[19] Stephen Prajna and Ali Jadbabaie. Safety verification using barrier certificates. In *HSCC*, volume 2993 of *LNCS*, pages 477–492. Springer, 2004.

[20] Sriram Sankaranarayanan, Henny B. Sipma, and Zohar Manna. Constructing invariants for hybrid systems. In *HSCC*, volume 2993 of *LNCS*, pages 539–555. Springer, 2004.

[21] Sriram Sankaranarayanan, Henny B. Sipma, and Zohar Manna. Fixed point iteration for computing the time-elapse operator. In *HSCC*, LNCS. Springer, 2006.

[22] Ashish Tiwari. Formally analyzing adaptive flight control, 2009. Proc. of workshop on Numerical Software Verification (NSVII).

[23] Ashish Tiwari and Gaurav Khanna. Non-linear systems: Approximating reach sets. In *HSCC*, volume 2993 of *LNCS*, pages 477–492. Springer, 2004.

[24] Claire J. Tomlin, George J. Pappas, and Shankar Sastry. Conflict resolution for air traffic management: A study in multi-agent hybrid systems. *IEEE Trans. on Aut. Control*, 43(4): 509–521, April 1998.

[25] Wolfgang Walter. *Ordinary Differential Equations*. Springer, 1998.