# Series of Abstractions for Hybrid Automata*

Ashish Tiwari[1] and Gaurav Khanna[2]

[1] SRI International,
333 Ravenswood Ave,
Menlo Park, CA, U.S.A
Tel:+1.650.859.4774
Fax:+1.650.859.2844
tiwari@csl.sri.com

[2] Theoretical and Computational Studies Group
Long Island University, Southampton
Southampton NY 11968
gkhanna@liu.edu

**Abstract.** We present a technique based on the use of the quantifier elimination decision procedure for real closed fields and simple theorem proving to construct a series of successively finer qualitative abstractions of hybrid automata. The resulting abstractions are always discrete transition systems which can then be used by any traditional analysis tool. The constructed abstractions are conservative and can be used to establish safety properties of the original system. Our technique works on linear and non-linear polynomial hybrid systems, that is, the guards on discrete transitions and the continuous flows in all modes can be specified using arbitrary polynomial expressions over the continuous variables. We have a prototype tool in the SAL environment [13] which is built over the theorem prover PVS [19]. The technique promises to scale well to large and complex hybrid systems.

## 1 Introduction

Hybrid systems describe a wide class of systems that exhibit discrete and continuous behaviors, such as a digital system embedded in an analog environment. Since hybrid systems operate in safety-critical domains, for example, inside automobiles, aircrafts, and chemical plants, analysis techniques are needed to support the design process of embedded software while maintaining safety guarantees.

The development of tools and analysis techniques for hybrid systems is faced with two challenges. It has been shown that checking reachability for very simple class of hybrid systems is undecidable [11]. Several decidable classes have been identified, see [3] for a survey, but all of these classes are too weak to represent

* The first author was supported in part by DARPA under the MoBIES program administered by AFRL under contract F33615-00-C-1700 and NASA Langley Research Center contract NAS1-00108 to Rannoch Corporation. The second author acknowledges research support from Long Island University, Southhampton.

hybrid system models that arise in practical applications. In fact, the models of physical environment in real world scenarios are usually too large and complicated even for analysis tools built on semi-decision procedures and available technologies.

Abstraction is a technique to reduce the complexity of a system design, while preserving some of its relevant behavior, so that the simplified system is more accessible to analysis tools and is still sufficient to establish certain safety properties. Two powerful abstraction techniques, called predicate abstraction and data abstraction respectively, have been used quite successfully in analyzing discrete transition systems. In this paper, we present a very simple, yet quite effective, technique based on data abstraction, to construct a series of successively finer abstractions of a given hybrid system.

Hybrid automata [1, 17] are mathematical models for representing hybrid systems. In contrast to discrete transition systems, hybrid automata can make *both* discrete and continuous transitions and hence, its semantics are given in terms of the states, which are uncountably many, reached over a continuous real time interval. However, the theory of hybrid automata can be given in terms of infinite-state transition systems [9, 11] that contain uncountably many states, but are interpreted over discrete time steps. In this paper, we map the uncountable state space into a finite state space by an abstraction function. More specifically, the $n$-dimensional real space $\mathbb{R}^n$ is partitioned into zones which are sign-invariant for all polynomials in some finite set. Increasing the number of polynomials in this set results in finer abstractions. This basic idea, although in a much simplified form, is also at the core of qualitative reasoning techniques developed in the Artificial Intelligence community [14, 20, 21].

Our work extends conventional qualitative techniques in at least two ways. We keep track of the evolution of arbitrary polynomials (over the state variables), and not just the state variables, while constructing an abstraction. Second, whereas qualitative reasoning usually uses sign of only the first derivative, we deduce based on the signs of first $n$-th derivatives. The use of these two nontrivial extensions makes the technique substantially more powerful. Similar extensions, but without any theorem proving support, have been used for behavior prediction after a fault diagnosis in a dynamic physical system [16].

One other useful feature of our approach is that it can be used to construct a series of finer abstractions. This gives an iterative methodology to prove a safety property. Starting with a crude abstraction, we check whether the property of interest holds for this abstract system. If not, we create a finer abstraction and check the property again. We can repeat this until the property is either established or no further refinements of the system can be constructed. Since the resulting abstractions are discrete transition systems, techniques such as model checking can be used. Furthermore, unlike a lot of other works on analysis of hybrid systems [5, 15], we do not use any numerical methods and techniques.

The process of construction of the abstract system requires logical reasoning in the theory of reals. The first-order theory of real closed fields is known to be decidable [23] and the first practical algorithm, based on cylindrical algebraic decomposition, was given in [6]. We use the first-order theory of reals to represent sets of continuous states and use reasoning over this theory for creating abstract transition systems.

**Preliminaries**

The *signature* of the first-order theory of reals consists of function symbols $\{+, -, \cdot\}$, constants $\mathbb{R}$, and predicate symbols $\{=, >, \geq, <, \leq\}$. In this theory, the set of terms over a set $X$ of variables corresponds to the set of polynomials $\mathbb{R}[X]$. The set $ATM(X)$, defined as $\{p \sim 0 : p \in \mathbb{R}[X] and \sim \in \{=, <, \leq, >, \geq\}\}$, is the set of all *atomic* formulas. The set $WFF(X)$ of first-order formulas (over $X$) is defined as the smallest set containing $ATM(X)$ and closed under the boolean operations (conjunction $\wedge$, disjunction $\vee$, implication $\Rightarrow$, and negation $\neg$) and quantification (existential $\exists$ and universal $\forall$). The *first-order theory* of *reals*, also denoted by $\mathbb{R}$, is defined as the set of all first-order formulas over the above signature (and a countable set of variables) that are true over the real numbers. We use the notation $\mathbb{R} \models \phi$ to denote the fact that the (first-order) formula $\phi$ is true in the theory of reals. The first-order theory of the real closed fields is a complete theory, that is every sentence in $WFF(X)$ is either true or its negation is true in this theory, and is known to be decidable [6, 23].

We denote formulas in $WFF(X)$ by $\phi, \psi$, possibly with subscripts and use $p$ to denote polynomials in the set $\mathbb{R}[X]$. We say a polynomial $p$ occurs in a formula $\phi$ if there is an atomic formula $p \sim 0$ in $\phi$. The rest of the notation follows the standard practice in hybrid systems literature.

## 2 Continuous Dynamical Systems

For simplicity, in this section we consider hybrid systems with no discrete components, that is, hybrid systems with exactly one mode of operation. A continuous dynamical system $CS$ is a tuple $(X, Init, Inv, f)$ where $X$ is a finite set of variables interpreted over the reals $\mathbb{R}$, $\mathbf{X} = \mathbb{R}^X$ is the set of all valuations of the variables $X$, $Init \subseteq \mathbf{X}$ is the set of initial states, $Inv \subseteq \mathbf{X}$ is the invariant set of states, and $f : \mathbf{X} \mapsto T\mathbf{X}$ is a vector field that specifies the continuous dynamics. Here $T\mathbf{X}$ denotes the tangent space of $\mathbf{X}$. We assume that $f$ satisfies the standard assumptions for existence and uniqueness of solutions to ordinary differential equations. Note that the continuous dynamical systems we consider here are autonomous, that is, they have no inputs.

The semantics, $[\![CS]\!]$, of a continuous dynamical system $CS = (X, Init, Inv, f)$ over an interval $I = [\tau_a, \tau_z] \subseteq \mathbb{R}$ is a collection of mappings $\sigma : I \mapsto \mathbf{X}$ satisfying

(a) initial condition: $\sigma(\tau_a) \in Init$,
(b) continuous evolution: for all $\tau \in (\tau_a, \tau_z)$, $\dot{\sigma}(\tau) = f(\sigma(\tau))$, and
(c) invariant: for all $\tau \in [\tau_a, \tau_z]$, $\sigma(\tau) \in Inv$.

In case the interval $I$ is left unspecified, it is assumed to be the interval $[0, \infty)$.

We assume that the flow derivative, $f$, is specified using polynomial expressions over the state variables $X$, that is $f \in (\mathbb{R}[X])^{|X|}$, where $\mathbb{R}[X]$ denotes the set of polynomials over the indeterminates $X$ and coefficients in $\mathbb{R}$, and $|X|$ denotes the cardinality of $X$. These polynomials can be nonlinear in general.

*Example 1.* The actuator module in a simple electronic throttle control system is driven by a pulse-width modulated signal and can be described as a hybrid system with two modes: when the input signal is high, the system is in the "on" mode and is described by

$$\dot{V} = \tfrac{2000}{9}(24 - 2V - I) \qquad \dot{I} = \tfrac{1000}{15}(120 - 22I)$$

and when the input is low, the system is in "off" mode and is described by

$$\dot{V} = \tfrac{-2000}{3} I \qquad \dot{I} = \tfrac{2000}{15}(5V - 16I).$$

In each mode, therefore, the actuator behaves as a continuous dynamical system with two continuous variables $V$ and $I$.

## 3 Discrete Transition Systems

A discrete state transition system $DS$ is a tuple $(Q, \mathit{Init}, t)$ where $Q$ is a finite set of variables interpreted over countable domains, $\mathbf{Q}$ denotes the (countable) set of all valuations of the variables $Q$ over the respective domains, $\mathit{Init} \subseteq \mathbf{Q}$ is a set of initial states, and $t \subseteq \mathbf{Q} \times \mathbf{Q}$ is a set of transitions. The semantics, $[\![DS]\!]$, of a discrete state transition system $DS = (Q, \mathit{Init}, t)$ is the collection of all mappings $\theta : \mathbb{N} \mapsto \mathbf{Q}$ satisfying

(a) initial condition: $\theta(0) \in \mathit{Init}$, and
(b) discrete evolution: for all $i \in \mathbb{N}$, $(\theta(i), \theta(i+1)) \in t$.

In order to define a notion of *abstraction* precisely, we need to establish a correspondence between discrete evolutions $\theta : \mathbb{N} \mapsto \mathbf{Q}$ and continuous evolutions $\sigma : [0, \infty) \mapsto \mathbf{Q}$. This is done using discrete sampling.

**Definition 1.** *A discrete evolution $\theta : \mathbb{N} \mapsto \mathbf{Q}$ is a sufficiently complete discretization of a continuous evolution $\sigma : [0, \infty) \mapsto \mathbf{Q}$ if there exists a strictly increasing sequence $\langle \tau_0, \tau_1, \tau_2, \ldots \rangle$ of reals in the interval $[0, \infty)$ such that*
*(i) $\tau_0 = 0$,*
*(ii) the function $\sigma$ does not change on the domain $(\tau_i, \tau_{i+1})$, that is, $\sigma(\tau) = \sigma(\tau')$ for all $\tau, \tau' \in (\tau, \tau_{i+1})$, and*
*(iii) for all $i$, $\theta(2i) = \sigma(\tau_i)$ and $\theta(2i+1) = \sigma(\tau)$, where $\tau_i < \tau < \tau_{i+1}$.*

Intuitively, a sufficiently complete discretization captures all the "different" (abstract) states in the given continuous evolution.

**Definition 2.** *Let $CS = (X, \mathit{InitX}, \mathit{Inv}, f)$ be a continuous dynamical system and $DS = (Q, \mathit{InitQ}, t)$ be a discrete transition system. We say $DS$ is an* abstraction *for $CS$ if there exists a mapping $abs : \mathbf{X} \mapsto \mathbf{Q}$ such that*

*(a)* $abs(\mathit{InitX}) \subseteq \mathit{InitQ}$,[1] *and*
*(b) for every* $\sigma \in [\![CS]\!]$, *if* $\sigma'$ *is a sufficiently complete discretization of* $abs(\sigma)$, *then* $\sigma' \in [\![DS]\!]$.

This definition of abstraction corresponds to the usual sense of abstraction, but applied to the infinite state transition system associated with a continuous (hybrid) system. We consider the problem of constructing discrete transition system abstractions for continuous dynamical systems in the sense of Definition 2. The definition and the procedure for constructing an abstraction naturally extends to hybrid systems, see Section 5.

## 4 Abstracting Continuous Dynamical Systems

Data abstraction refers to the idea of using a partition of the domain of interpretation as the new domain of interpretation for the state variables or expressions over the state variables. The focus in this paper is on performing data abstraction on continuous and hybrid systems. We use abstract variables that represent *polynomials* over the original continuous variables $X$ and interpret them over a three valued abstract domain $\{neg, pos, zero\}$.

Given a continuous dynamical system $CS = (X, \mathit{InitX}, \mathit{Inv}, f)$, we construct the abstract discrete state transition system $DS = (Q, \mathit{InitQ}, t)$ in two steps. The first phase creates a finite set $P \subseteq \mathbb{R}[X]$ of polynomials over the continuous variables $X$ which are used as the discrete variables $Q$. In the second phase, the initial states $\mathit{InitQ}$ and the transition relation $t$ are computed.

### Phase I: Obtaining a set of polynomials

Fixing the set $P$ of polynomials for abstraction involves starting with a small set $P_0$ of polynomials of interest and adding to this set the time derivatives of polynomials in $P_0$. The initial set $P_0$ could contain, for example, the polynomials that appear in the statement of the property of interest that we want to establish for the given continuous system, or the polynomials that occur in the guards of mode change transitions for exiting this mode, etc. The phase I saturation process involves application of the following inference rule: *if* $p \in P$, *then add* $\dot{p}$, *the derivative (with respect to time) of* $p$, *to the set* $P$ *unless* $\dot{p}$ *is a constant or a constant factor multiple of some existing polynomial in* $P$.

Since we assume that $f \in (\mathbb{R}[X])^{|X|}$, it follows that $\dot{p} \in \mathbb{R}[X]$ is a polynomial. However, note that for general flow derivatives $f$, specified using arbitrary polynomial expressions, the saturation process might not terminate. But there are special cases where this process is guaranteed to terminate.

[1] We shall use $abs$ to also denote liftings of the function $abs$ to sets and functions. Thus, $abs(\mathit{InitX}) = \{abs(\mathbf{x}) : \mathbf{x} \in \mathit{InitX}\}$. Similarly, if $\sigma : [0, \infty) \mapsto \mathbf{X}$, then $(abs(\sigma))(\tau) = abs(\sigma(\tau))$.

**Nilpotent Systems.** Consider the class of linear time invariant systems specified using a nilpotent matrix A. If we also use $X$ to denote the column vector of state variables $X$, then the flow rate $f = AX$ and hence, $\dot{X} = AX$. A polynomial $p = \sum_i a_i x_i$ can be written, in matrix notation, as $\boldsymbol{a}^T X$, where $\boldsymbol{a}^T$ denotes the transpose of $\boldsymbol{a}$. Thus, $\dot{p} = \boldsymbol{a}^T \dot{X} = \boldsymbol{a}^T AX$ and $\ddot{p} = \boldsymbol{a}^T A^2 X$. Hence, if $A^n = 0$, then the $n$-th derivative of the polynomial $p$ is $\boldsymbol{a}^T A^n \boldsymbol{x} = 0$. Thus, the saturation process is guaranteed to terminate for such systems.

**Systems such that $\boldsymbol{A^n = rA^m}$.** If the matrix $A$ used to specify the flow of the continuous dynamical system $CS$ is such that $A^n = rA^m$ for some constant $r \in \mathbb{R}$ and $n, m \in \mathbb{N}$, then again the saturation process can be shown to terminate. In particular, if $p = \boldsymbol{a}^T X$ is an arbitrary polynomial, then $\frac{d^n p}{d\tau^n} = \boldsymbol{a}^T A^n X = \boldsymbol{a}^T r A^m X = r \frac{d^m p}{d\tau^m}$. Since the $n$-th derivative of $p$ is a constant multiple of the $m$-th derivative of $p$, it does not get added to the set $P$ of polynomials in the saturation process.

We remark here that the termination of the saturation process is determined by *both* the initial set $P_0$ of polynomials *and* the flow derivative $f$.

**General Case.** Our abstraction technique works for general (possibly nonlinear) time invariant systems whose flow is specified using polynomials. The termination of the saturation phase is not necessary for creating an abstraction. We can stop at any point and pass on the current set $P$ to the second phase. A larger set $P$ yields a finer abstraction as it results in a larger state space in the final abstract system.

*Example 2.* Consider the "off" mode of the actuator in Example 1. If we start with the set $P_0 = \{V, I\}$ of polynomials, the phase I saturation procedure first adds the polynomial $\dot{I} = 2000/15(5V - 16I)$ and then the derivative of this polynomial $\ddot{I} = 2000^2/15(-16V/3 + 77I/5)$. Since the derivative $\dot{V}$ is a constant multiple of the polynomial $I \in P$, it is not added. Note that we need not add the exact derivatives, but only a polynomial upto some constant factor. Although we can continue the process of adding derivatives, we stop the phase I here with the final set $P = \{V, I, 5V - 16I, -80V + 231I\}$.

### Phase II: Constructing the Abstract Transitions

Let $CS = (X, \mathit{InitX}, \mathit{Inv}, f)$ be a continuous system and $P \subseteq \mathbb{R}[X]$ be a finite set of polynomials over the set $X$ of variables produced by the first phase. The state variables $Q$ in the corresponding abstract discrete system $DS = (Q, \mathit{InitQ}, t)$ contains exactly one new variable for each polynomial $p \in P$. Thus, $Q = \{q_p : p \in P\}$. These new variables are interpreted over the domain $\{\mathit{pos}, \mathit{neg}, \mathit{zero}\}$ and consequently the set $\mathbf{Q}$ of all discrete states is the set $\{\mathit{pos}, \mathit{neg}, \mathit{zero}\}^Q$ of all valuations of the variables $Q$ over this domain. We shall represent any such valuation by the corresponding conjunction of atomic formulas. For example, the valuation $\langle q_{p_1} \mapsto \mathit{pos}, q_{p_2} \mapsto \mathit{neg}, q_{p_3} \mapsto \mathit{zero} \rangle$ will be thought of as the formula

$p_1 > 0 \ \wedge \ p_2 < 0 \ \wedge \ p_3 = 0$. We shall use such conjunctions and valuations interchangeably. The set of all conjunctions representing such valuations will also be denoted by $\mathbf{Q}$. Note that these conjunctions are in the set $\mathit{WFF}(X)$ of formulas over free variables $X$.

If $\psi \in \mathbf{Q}$ is a state in the abstract system $DS$, say represented as $\wedge_{i\in J_1} p_i > 0 \ \wedge \ \wedge_{i\in J_2} p_i < 0 \ \wedge \ \wedge_{i \in J_3} p_i = 0$, then the concretization function, $\gamma$, maps abstract states to sets of concrete states and is defined by[2],

$$\begin{aligned}\gamma(\psi) = \{x \in \mathbf{X} : \mathbb{R} \models p_i(x) > 0 \ \forall i \in J_1 \ \mathit{and} \ \mathbb{R} \models p_i(x) < 0 \ \forall i \in J_2 \ \mathit{and} \\ \mathbb{R} \models p_i(x) = 0 \forall i \in J_3\}\end{aligned}$$

Conversely, if $x \in \mathbf{X}$ is a concrete state of the system $CS$, then the abstraction function, $abs$, maps a concrete state to an abstract state and is defined by,

$$abs(x) = \bigwedge_{i\in J_1} p_i > 0 \ \wedge \ \bigwedge_{i\in J_2} p_i = 0 \ \wedge \ \bigwedge_{i\in J_3} p_i < 0,$$

where $J_1 \cup J_2 \cup J_3$ is a partition of the set $\{1, 2, \ldots, |P|\}$ such that $i \in J_1$ iff $\mathbb{R} \models p_i(x) > 0$, $i \in J_2$ iff $\mathbb{R} \models p_i(x) = 0$, and $i \in J_3$ iff $\mathbb{R} \models p_i(x) < 0$.

**The Initial States.** Assume that the initial set of states $\mathit{InitX}$ for the continuous system is specified using a first-order formula $\phi_X$ over $X$. The initial set of states $\mathit{InitQ}$ consists of all valuations $\psi$ of the abstract variables such that the formulas $\psi$ and $\phi_X$ are simultaneously satisfiable. Specifically,

$$\mathit{InitQ} = \bigvee \{\psi \in \mathbf{Q} : \mathbb{R} \models \exists X : \psi \wedge \phi_X\}.$$

**Lemma 1.** *Let $CS = (X, \mathit{InitX}, \mathit{Inv}, f)$ be a continuous system with the initial states $\mathit{InitX}$ specified by the first-order formula $\phi_X$. If $DS$, $\mathit{InitQ}$, and $abs$ are as defined as above, then, $abs(\mathit{InitX}) \subseteq \mathit{InitQ}$.*[3]

**The Transition Relation.** We add an abstract transition $(\psi_1, \psi_2) \in t$ if all of the following conditions hold (for all polynomials $p \in P$):
(a) if $p < 0$ is a conjunct in $\psi_1$, then (a1) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} < 0$, then $p < 0$ is a conjunct in $\psi_2$; (a2) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} = 0$, then $p < 0$ is a conjunct in $\psi_2$; (a3) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} > 0$, then either $p < 0$ or $p = 0$ is a conjunct in $\psi_2$; and (a4) if the valuation of $\dot{p}$ cannot be deduced from $\psi_1$, then either $p < 0$ or $p = 0$ is a conjunct in $\psi_2$;
(b) if $p = 0$ is a conjunct in $\psi_1$, then (b1) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} < 0$, then $p < 0$ is a conjunct in $\psi_2$; (b2) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} = 0$, then $p = 0$ is a conjunct in $\psi_2$; (b3) if

[2] Here, the notation $\mathbb{R} \models p(x) > 0$ means that the polynomial $p$ evaluates to a positive number on the point $x \in \mathbb{R}^{|X|}$.

[3] We use a formula and the set of valuations it represents interchangeably as the context disambiguates the intended meaning.

$\mathbb{R} \models \psi_1 \Rightarrow \dot{p} > 0$, then $p > 0$ is a conjunct in $\psi_2$; and (b4) if the valuation of $\dot{p}$ cannot be deduced from $\psi_1$, then either $p > 0$, $p = 0$, or $p < 0$ is a conjunct in $\psi_2$;
(c) if $p > 0$ is a conjunct in $\psi_1$, then (c1) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} < 0$, then either $p > 0$ or $p = 0$ is a conjunct in $\psi_2$; (c2) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} = 0$, then $p > 0$ is a conjunct in $\psi_2$; (c3) if $\mathbb{R} \models \psi_1 \Rightarrow \dot{p} > 0$, then $p > 0$ is a conjunct in $\psi_2$; and (c4) if the valuation of $\dot{p}$ cannot be deduced from $\psi_1$, then either $p > 0$ or $p = 0$ is a conjunct in $\psi_2$.

This completes the phase of adding transitions to the abstract system. Note that the sign of $\dot{p}$ can be directly read off from $\psi_1$ if $\dot{p}$ was added to $P$ in phase I. If not, then we add non-deterministic transitions from $\psi_1$ assuming all possibilities for the sign of $\dot{p}$. In the final step, we refine this abstract system to eliminate unreachable states and transitions.

**Refining the Abstraction.** We note that certain abstract states (and transition to/from those states) can be deleted because either they are infeasible or are explicitly disallowed by the given invariant set *Inv* of the concrete system. In particular, if the invariant set *Inv* is specified using a first-order formula $\phi_{Inv}$, then we can delete all abstract states $\psi$ such that $\mathbb{R} \not\models \exists X : \psi(X) \wedge \phi_{Inv}(X)$. We can also remove all transitions to/from these eliminated abstract states. Note that this process implicitly removes infeasible abstract states, that is, states $\psi(X)$ such that $\mathbb{R} \not\models \exists X : \psi(X)$.

This completes the construction of the abstract system $DS = (Q, \mathit{InitQ}, t)$ for the continuous dynamical system $CS = (X, \mathit{InitX}, \mathit{Inv}, f)$.

**Theorem 1.** *Let $CS = (X, \mathit{InitX}, \mathit{Inv}, f)$ be a continuous system and $DS = (Q, \mathit{InitQ}, t)$, be the discrete abstraction as defined above. Then, $DS$ is an abstraction (Definition 2) for $CS$.*

Note that even though the abstract transition system is a finite-state system, we need not explicitly represent the states and transitions. We can obtain the abstract system implicitly with the states and transitions specified using predicate formulas.

*Example 3.* Following up on Example 2, we can construct the abstract transition system on the set $P = \{V, I, 5V - 16I, -80V + 231I\}$ of polynomials. Assume that the initial abstract state[4] is $I > 0 \wedge V > 0 \wedge 5V - 16I < 0 \wedge -80V + 231I > 0$. Out of the $3^4 = 81$ abstract states, only 17 are feasible and the infeasible states can be identified using a theorem prover. For example, the state $I = 0 \wedge V > 0 \wedge 5V - 16I < 0 \wedge -80V + 231I > 0$ is infeasible and a decision procedure for reals can be used to deduce that this formula is unsatisfiable.

The outgoing transitions from the initial state $I > 0 \wedge V > 0 \wedge 5V - 16I < 0 \wedge -80V + 231I > 0$ are obtained as follows: (a) since $I > 0$ and $\dot{I} < 0$ (as $5V - 16I < 0$), in the successor state either $I > 0$ or $I = 0$, (b) since $V > 0$

[4] The initial abstract state is obtained from the stable states in the abstract transition system for the "on" mode of the actuator.

and $\dot{V} < 0$ (as $-I < 0$), in the successor state either $V > 0$ or $V = 0$, (c) since $5V - 16I < 0$ and $5\dot{V} - 16\dot{I} > 0$ (as $-80V + 231I > 0$), in the successor state either $5V - 16I < 0$ or $5V - 16I = 0$, and (d) since $-80V + 231I > 0$ and $-80\dot{V} + 231\dot{I}$ is unknown, in the successor state either $-80V + 231I > 0$ or $-80V + 231I = 0$. Out of the 16 potential successors, only 4 are feasible:

$$\begin{aligned}
q_1 &: I > 0 \ \wedge\ V > 0 \ \wedge\ 5V - 16I < 0 \ \wedge\ -80V + 231I > 0 \\
q_2 &: I > 0 \ \wedge\ V = 0 \ \wedge\ 5V - 16I < 0 \ \wedge\ -80V + 231I > 0 \\
q_3 &: I > 0 \ \wedge\ V > 0 \ \wedge\ 5V - 16I < 0 \ \wedge\ -80V + 231I = 0 \\
q_4 &: I = 0 \ \wedge\ V = 0 \ \wedge\ 5V - 16I = 0 \ \wedge\ -80V + 231I = 0.
\end{aligned}$$

Continuing this way, we can construct the complete abstract system containing 10 reachable abstract states. We also note that among these states, only the state $q_4$ is stable.

## 5 Hybrid Automata

The technique for constructing finite state abstractions of continuous systems extends naturally to hybrid systems. We skip the definitions and details here and refer to the full version of the paper. To summarize these results, the abstract system corresponding to the hybrid system $HS = (Q, X, Init, Inv, t, f)$ and a finite set $P$ of polynomials (over X) is a discrete state transition system $DS = (Q^A, Init^A, t^A)$, where $Q^A = Q \cup (Q_P = \{q_p : p \in P\})$ is the set of discrete variables, $Init^A \subseteq \mathbf{Q^A}$ is the initial states, and $t^A \subseteq \mathbf{Q^A} \times \mathbf{Q^A}$ is the set of transitions. The new discrete variables $Q_P$ are interpreted over the domain $\{pos, neg, zero\}$ as before. Thus, the set of states in the abstract system $\mathbf{Q^A}$ is $\mathbf{Q} \times \{pos, neg, zero\}^{Q_P}$.

Let $q^a = (q, \phi) \in \mathbf{Q^A}$ be a state in the abstract system, where $q \in \mathbf{Q}$ is a discrete state of the hybrid automaton $HS$ and $\phi$ is a valuation of the variables in $Q_P$ over $\{pos, neg, zero\}$. We think of $\phi$ as a formula in $WFF(X)$ as before. The transitions in the abstract system $DS$ from the state $q^a$ are obtained as a union of two kinds of transitions:

1. *Abstractions of the discrete transitions:* If $(q, Cond, q') \in t$ is a discrete transition of the hybrid automata $HS$, where $q, q' \in \mathbf{Q}$ are discrete states and $Cond \subset \mathbf{X}$ is a set of continuous states (or the guard) represented by, say, the predicate formula $\psi$ over the variables $X$, then there is an abstract transition $((q, \phi), (q', \phi)) \in t^a$ if $\mathbb{R} \models \exists X : (\phi(X) \wedge \psi(X))$.
2. *Abstractions of the continuous transitions:* The rule for constructing new abstract transitions from the continuous flows is the same as before. We note that the first component of the state is left unchanged, that is, we add a new abstract transition $((q, \phi), (q, \psi))$ in $t^a$ if $\psi$ can be obtained from $\phi$ using the rules given before in Section 4 (applied to the flow corresponding to the discrete state $q$).

Note that we can handle cases where the set $\mathbf{Q}$ of discrete states in $HS$ is infinite as long as the number of distinct "modes" (each of which can be specified as a formula over $Q$) are finite.

**Theorem 2.** *Let $HS = (Q, X, Init, Inv, t, f)$ be a hybrid automata and $P \subseteq \mathbb{R}[X]$ be a finite set of polynomials over the set $X$ of real variables. If $DS = (Q^A = Q \cup Q_P, Init^A, t^A)$ is the discrete transition system constructed by the above method, then $DS$ is an abstraction for $HS$.*

We illustrate the abstraction technique on a simple hybrid system example.

*Example 4.* Consider a thermostat that controls the heating of a room. Assume that the thermostat turns the heater on when the temperature $x$ is between 68 and 70 and it turns the heater off when the temperature is between 80 and 82. Suppose the continuous dynamics in the on and off modes is specified respectively by the equations

$$\dot{x} = -x + 100 \quad \textit{and} \quad \dot{x} = -x.$$

If we assume that the heater is initially off and the room temperature is between 70 and 80, the hybrid automaton is given by $HS = (Q, X, Init, Inv, t, f)$, where $Q = \{q_1\}$ is the set of discrete variables, $\mathbf{Q} = \{on, off\}$ is the set of discrete states (thus, $q_1 \in \{on, off\}$), $X = \{x_1\}$ is the set of continuous variables, $\mathbf{X} = \mathbb{R}$ is the set of continuous states, $Init = \{(off, x) : 70 < x < 80\}$ is the initial condition, $Inv = \{(on, x) : x < 82\} \cup \{(off, x) : x > 68\}$ is the invariant set, $t = \{(on, x, off) : x \geq 80\} \cup \{(off, x, on) : x \leq 70\}$ is the set of discrete transitions, and $f(on) = -x + 100$ and $f(off) = -x$ specifies the continuous flow rates.

Now, the set of polynomials that appear in the guards are $\{x - 70, x - 80\}$, and polynomials in the invariant specification are $\{x - 68, x - 82\}$. The derivative of each of these four polynomials is $\dot{x}$. In the mode when the heater is on, this evaluates to $-x + 100$ and in the mode when the heater is off, this is $-x$. Hence, we have two more polynomials, $\{x, x - 100\}$, in the set $P$. Note that further saturation of the set $P$ of these six polynomials under time derivative yields no new polynomials.

Using the saturated set $P$ of six polynomials, we can construct an abstraction for the thermostat hybrid model and we show the final result Figure 1. In the figure, transitions arising from the continuous and discrete evolutions of $H$ are drawn in different colors. Furthermore, the representation of abstract states has been simplified. For example, the expression $70 < x < 80$ denotes the conjunction $70 < x \wedge x < 80 \wedge 68 < x \wedge x < 82 \wedge -x + 100 > 0 \wedge x > 0$. This conjunction is logically equivalent to $70 < x \wedge x < 80$.

## 6 Implementation and Related Work

The SAL tool set provides interfaces that can be used to construct discrete abstractions of hybrid systems as described in this paper [13]. The quantifier

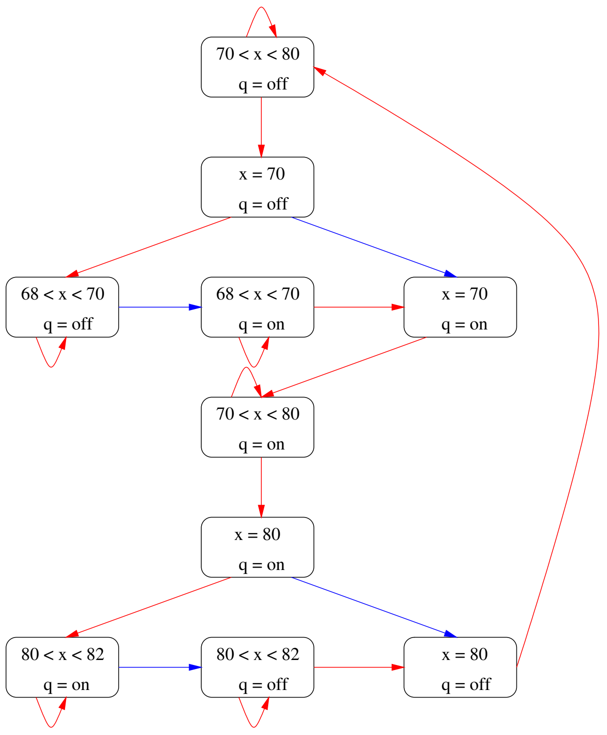

**Fig. 1.** Abstract transition system for the thermostat hybrid automata

elimination decision procedure for the real closed fields is implicitly used to decide the implications over the real numbers. The tool QEPCAD [12], which is built over the symbolic algebra library SACLIB [4], is integrated to the theorem prover PVS [19] for this purpose. We are working on adding more explicit interfaces into SAL to directly construct such abstractions for hybrid systems.

The discrete abstractions we construct do not store information about the duration of a continuous run. However, our technique extends, quite easily, to time variant systems by simply explicitly considering time as another continuous variable. Note that we can get some timing information in the abstractions if we include polynomials containing this variable for time in the set $P$.

Qualitative reasoning has been used by researchers in the AI community for modeling and analyzing physical systems in the face of incomplete knowledge of the system dynamics [20]. The idea is to interpret a continuous variable, say $x$, over an abstract domain of the form $\{(-\infty, c_0), c_0, (c_0, c_1), c_1, (c_1, c_2), c_2, \ldots, c_n, (c_n, \infty)\}$, where $c_0, c_1, \ldots, c_n \in \mathbb{R}$ are constants. Model construction involves keeping track of the sign of the derivative of $x$. In [20], the authors give a method for proving temporal properties about systems specified (incompletely) using *qualitative* differential equations. In [14] and [21], the authors assume a (more) completely specified input model (using differential equations, for example) and construct an abstraction either incrementally [14] or directly [21]. The latter approach has been implemented in Charon [2]. Our paper substantially extends the idea of qualitative reasoning by allowing for arbitrary *polynomials*, and not just state *variables*, for defining the qualitative state space. Additionally, we also use the signs of higher order derivatives in the procedure. As a result, the abstractions we obtain have more information and are more useful from an analysis point of view.

There has been a lot of work on constructing abstractions for hybrid systems. These works can be categorized based on the semantics of the hybrid system considered, the class of formulas preserved, the class of hybrid systems considered, the class of abstract systems generated, and whether the abstractions are conservative or accurate. Accurate abstractions, or bisimulations, lead to decidability results [3]. In [18], the interest is in abstracting certain restricted classes of linear hybrid systems into another simpler class of hybrid systems called *timed* automata. The paper [10] abstracts a nonlinear hybrid by a linear hybrid automata.

One can naturally associate an infinite state transition system, with uncountably many states, with a continuous dynamical system or a hybrid system. However, different abstractions preserve different behaviors of this infinite transition system. In [11], certain discrete transitions of the hybrid system are marked *observables*, and the abstraction preserves the observable behavior. In other cases [8], the discrete states in a run of the system are observed and the abstraction preserves this sequence of discrete states. In [5], the constructed abstraction captures all the discrete transition made by the system. In our work, the overall behavior of the hybrid system is abstracted with respect to a finite set of polynomials and the original discrete states. In particular, the behavior inside a continuous evolution is captured too. However, our method for computing the abstract transitions is more approximate (and consequently much simpler computationally) than some of these other methods as we only use information contained in the signs of the $n$-th derivatives (for fixed $n$) of some expressions. It should, however, be noted that as in [5], our abstractions do not retain any timing information apart from the temporal ordering of abstract states. However, timing information can be introduced either by treating $t$ as another state variable with $\dot{t} = 1$, or by incorporating quantitative timing information in the process of constructing an abstraction as in [22].

In the future we plan to further mechanize our technique and investigate its use for doing test vector generation for hybrid systems that would cover all regions of the state space, where a region is defined as the subspace which is sign-invariant for a set of polynomials. We also plan to explore further the integration with methods that employ additional quantitative information to create an abstraction.

**Acknowledgements**. We would like to thank Dr. J. Rushby, Dr. N. Shankar, and the anonymous reviewers for their helpful comments.

## References

1. R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems* [7], pages 209–229.
2. R. Alur, R. Grosu, Y. Hur, V. Kumar, and I. Lee. Modular specifications of hybrid systems in CHARON. In *Proc of 3rd Intl Workshop on Hybrid Systems: Computation and Control*, volume 1790 of *LNCS*, pages 6–19, 2000.
3. R. Alur, T. A Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(2):971–984, July 2000.
4. B. Buchberger, G. E. Collins, M. J. Encarnacion, H. Hong, J. R. Johnson, W. Krandick, R. Loos, A. M. Mandache, A. Neubacher, and H. Vielhaber. SACLIB 1.1 user's guide. In *RISC-Linz Report Series, Tech Report No 93-19*. Kurt Gödel Institute, 1993. `www.eecis.udel.edu/~saclib/`.
5. A. Chutinam and B. H. Krogh. Verification of polyhedral-invariant hybrid automata using polygonal flow pipe approximations. In *Hybrid Systems: Computation and Control*, volume 1569 of *LNCS*, pages 76–90. Springer-Verlag, 1999.
6. G. E. Collins. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. In *Proc. Second GI Conf. Automata Theory and Formal Languages*, volume 33 of *LNCS*, pages 134–183, 1975.
7. R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel (eds.). *Hybrid Systems*, volume 736 of *LNCS*. Springer-Verlag, Berlin, 1993.
8. M. R. Henzinger, T. A. Henzinger, and P. W. Kopke. Computing simulations on finite and infinite graphs. In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science FOCS*, pages 453–462, 1995.
9. T. A. Henzinger. Hybrid automata with finite bisimulations. In *Proc. 22nd ICALP*, volume 944 of *LNCS*, pages 324–335. Springer-Verlag, 1995.
10. T. A. Henzinger, P-H. Ho, and H. Wong-Toi. Algorithmic analysis of nonlinear hybrid systems. *IEEE Transactions on Automatic Control*, 43:540–554, 1998.
11. T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57:94–124, 1998.
12. H. Hong. Quantifier elimination in elementary algebra and geometry by partial cylindrical algebraic decomposition version 13. In *The world wide web*, 1995. `www.eecis.udel.edu/~saclib/`.
13. Computer Science Laboratory. SAL: Symbolic analysis laboratory. `http://www.csl.sri.com/projects/sal/`.
14. T. Loeser, Y. Iwasaki, and R. Fikes. Safety verification proofs for physical systems. In *Proc. of the 12th Intl. Workshop on Qualitative Reasoning*, pages 88–95. AAAI Press, 1998.
15. I. Mitchell, A. Bayen, and C. Tomlin. Validating a hamilton-jacobi approximation to hybrid system reachable sets. In M. D. Benedetto and A. L. Sangiovanni-Vincentelli, editors, *HSCC 4th Intl. Workshop*, volume 2034 of *LNCS*, 2001.
16. P. J. Mosterman and G. Biswas. Monitoring, prediction, and fault isolation in dynamic physical systems. *AAAI-97*, pages 100–105, 1997.
17. X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. An approach to the description and analysis of hybrid systems. In *Hybrid Systems* [7], pages 149–178.
18. A. Olivero, J. Sifakis, and S. Yovine. Using abstractions for the verification of linear hybrid systems. In *Proc. of the 6th Computer-Aided Verification CAV*, volume 818 of *LNCS*, pages 81–94, 1994.

19. N. Shankar, S. Owre, and J. M. Rushby. *The PVS Proof Checker: A Reference Manual.* Computer Science Lab, SRI International, 1993.
20. B. Shults and B. J. Kuipers. Proving properties of continuous systems: qualitative simulation and temporal logic. *AI Journal*, 92:91–129, 1997.
21. O. Sokolsky and H. S. Hong. Qualitative modeling of hybrid systems. In *Proc. of the Montreal Workshop*, 2001. Available from `http://www.cis.upenn.edu/~rtg/rtg_papers.htm`.
22. O. Stursberg, S. Kowalewski, I. Hoffmann, and Preußig. Comparing timed and hybrid automata as approximations of continuous systems. In P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, editors, *Hybrid Systems IV*, volume 1273 of *LNCS*, pages 361–377. Springer-Verlag, 1997.
23. A. Tarski. *A Decision Method for Elementary Algebra and Geometry.* University of California Press, 1948. Second edition.