Series of Abstractions for Hybrid Automata

Ashish Tiwari

Tiwari@csl.sri.com

Computer Science Laboratory

SRI International

Menlo Park CA 94025

Joint work with Prof. G. Khanna, Long Island University

Outline

1. High level description
2. Formal description and algorithm
3. Examples and Properties

Motivation

- Formal analysis of systems that exhibit continuous and discrete behaviors
- Focus closely on the discrete behavior (the mode switch logic) for missing transitions, incorrect logical conditions, resets, etc
- Use minimal information required of the continuous behavior without solving or numerically approximating solutions

Some Approaches for Analysis of Hybrid Systems

- Simulation and testing
- Decidability Results: Reachability algorithms for subclasses of hybrid automata
  - Timed Automata
  - Initialized Multirate Automata
  - Initialized Rectangular Automata
  - Linear Vector Fields specified using matrix A, where A is either nilpotent, diagonalizable with rational eigenvalues, or has only imaginary eigenvalues
  - O-minimal automata
- Abstraction Techniques: Compute approximations and work on them

Abstraction: The General Idea

1. Partition the state space: each member of the partition is a state of a new system
   - defines an abstraction map from concrete states onto a smaller set of "abstract states"
   - one abstract state corresponds to a set of concrete states
2. Quotient the original system modulo the partition:
   - there is a transition from one abstract state to another if there is a possible concrete one between the corresponding sets
   - the abstract system is a conservative approximation of the original system
3. Analyze the abstract system

Specific Abstraction Based Approaches

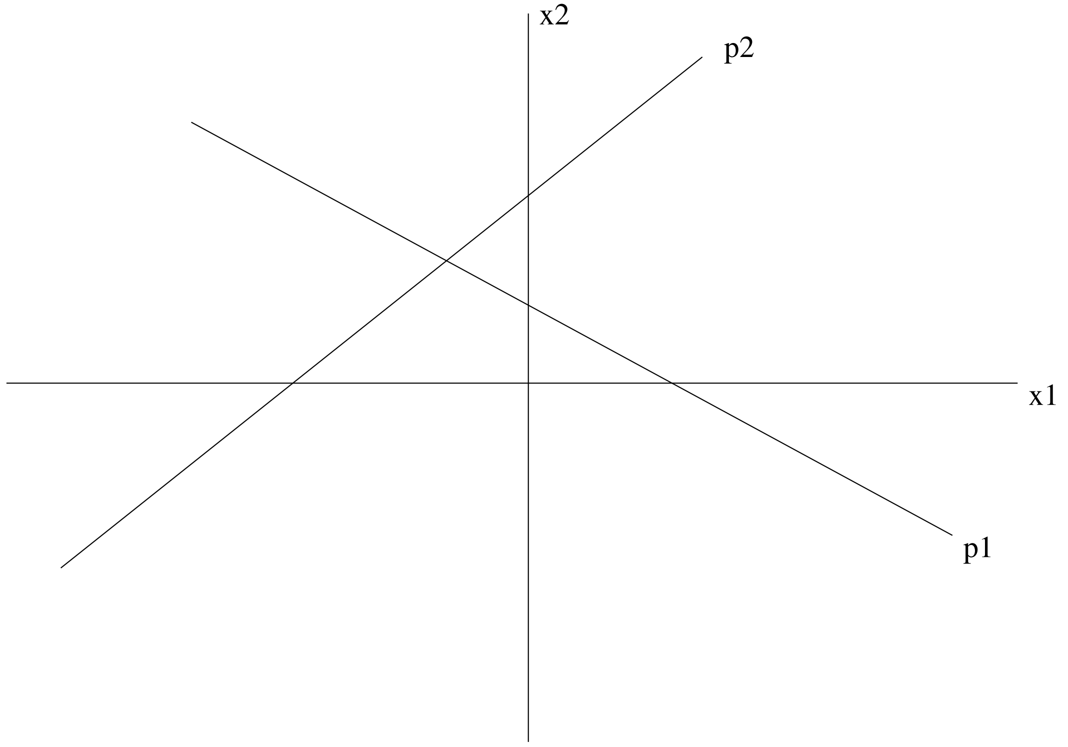
Partition	Transition Computation
Polytopes	Flowpipes
Ellipsoids	Numerical techniques
Level sets	Numerical techniques
Timed automata states	Simple quantitative reasoning
Rectangular automata states	Simple quantitative reasoning

In our work:

- Partitions are defined by the signs of certain polynomial expressions over the state variables
- Transitions are computed using qualitative abstraction on the continuous behavior and theorem proving on the discrete parts

Abstraction Algorithm: 1

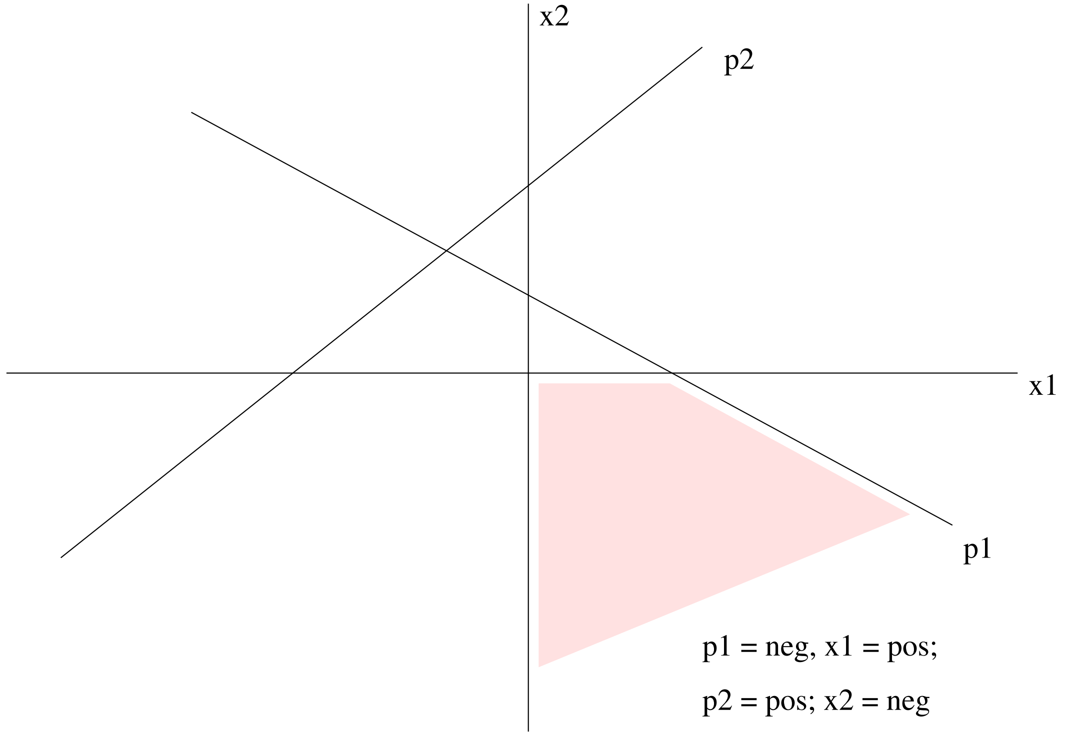
Consider a continuous dynamical system with two state variables. Concrete state space: $\Re^2$



Partitioned w.r.t signs of polynomials x_1, x_2, p_1, and p_2.

Abstraction Algorithm: 2

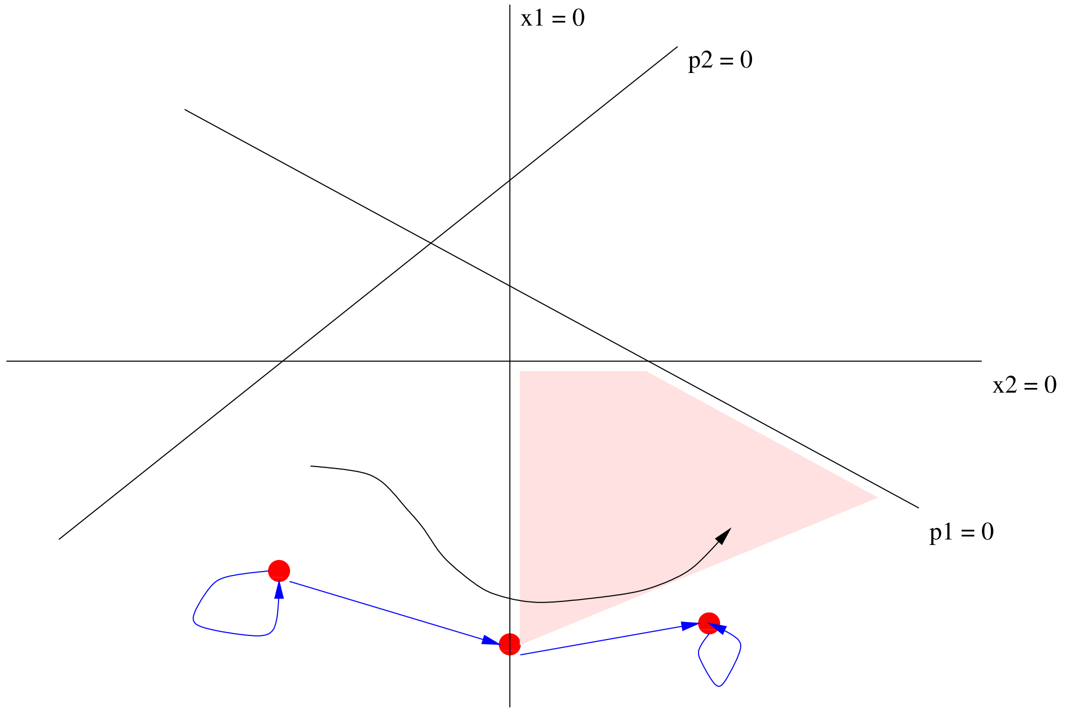
Abstract states correspond to subsets of concrete states.



More polynomials would mean more abstract states.

Abstraction Algorithm: 3

Abstract transitions overapproximate concrete transitions.



Total number of abstract states = $3^4 = 81$, but feasible abstract states = $11 + 16 + 6 = 33$

More polynomials would mean more abstract states.

Outline

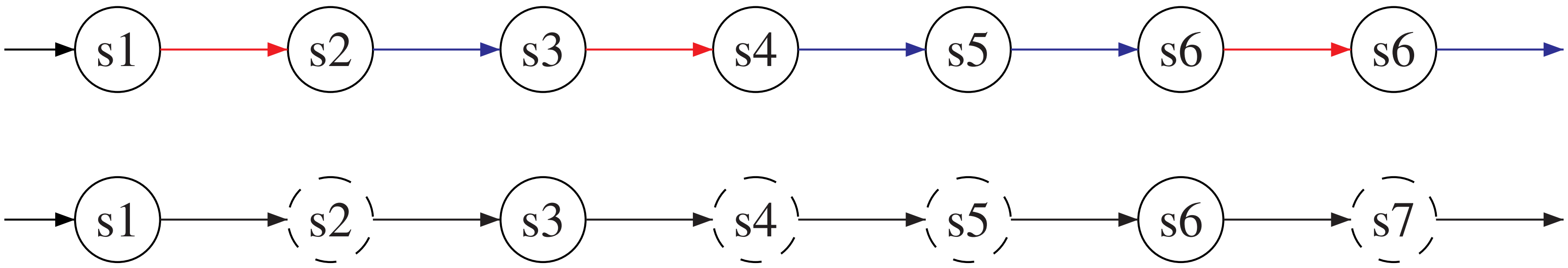
1. High level description
2. Formal description and algorithm
3. Examples and Properties

Hybrid Automata

A tuple $(Q, X, \mathbf{S}_0, F, Inv, R)$:

- Q: finite set of discrete variables
- X: finite set of continuous variables
- $\mathbf{X} = \mathbb{R}^{|X|}$, $\mathbf{Q}$ = set of all valuations for Q
- $\mathbf{S} = \mathbf{Q} \times \mathbf{X}$
- $\mathbf{S}_0 \subseteq \mathbf{S}$ is the set of initial states
- $F : \mathbf{Q} \mapsto (\mathbf{S} \mapsto \mathbb{R}^{|X|})$ specifies the rate of flow
- $Inv : \mathbf{Q} \mapsto 2^{\mathbb{R}^{|X|}}$ gives the invariant set
- $R \subseteq \mathbf{S} \times \mathbf{S}$ captures discontinuous state changes

Semantics of Hybrid Systems



- $s1 \in \mathbf{S}_0$ is an initial state
- Discrete Evolution: $s_i \rightarrow s_{i+1}$ iff $R(s_i, s_{i+1})$
- Continuous Evolution: $s_i = (l, x_i) \rightarrow s_{i+1} = (l, x_{i+1})$ iff there exists a $f : \mathbb{R}^{|X|} \mapsto \mathbb{R}^{|X|}$ and $\delta > 0$ such that

$$\begin{array}{rclcrcl} x_{i+1} & = & f(\delta) & \quad & x_i & = & f(0) \\ \dot{f} & = & F(l) & & f(t) & \in & Inv(l) \text{ } \mathit{for} \text{ } 0 \leq t \leq \delta \end{array}$$

The Abstraction Algorithm

Consists of three steps:

- Seed: Start with a set $P = P_0$ of polynomials of interest
  - Polynomials specifying the initial region $\mathbf{S_0}$
  - Polynomials in the transition conditions (specification of $\mathbf{R}$)
  - Polynomials in the resets (specification of $\mathbf{R}$)
- Saturate: Close the above set under derivative computation
  - For each polynomial $p \in P$, and each location $l \in \mathbf{Q}$, compute $\dot{p}$, given $\dot{X} = F(l)(X)$.
  - Add $\dot{p}$ to P and recurse.
- Add Transitions

The Abstract System State Space

- Abstract Variables:
  - Old discrete variables: $q \in Q$
  - New variables: $\{q_{pi} : p_i \in P\}$.
  - Type of new variables: $\{pos, zero, neg\}$
- Abstract State Space: $\mathbf{Q} \times 3^{|P|}$
- Abstraction Map:

$q_{pi} = pos$	$\mapsto$	$p_i > 0$
$q_{pi} = zero$	$\mapsto$	$p_i = 0$
$q_{pi} = neg$	$\mapsto$	$p_i < 0$
$q_{p_{i \in J_1}} = neg; q_{p_{i \in J_2}} = zero;$ $q_{p_{i \in J_3}} = pos$	$\mapsto$	$\bigwedge_{i \in J_1} p_i < 0 \;\wedge\; \bigwedge_{i \in J_2} p_i = 0$ $\wedge \bigwedge_{i \in J_3} p_i > 0$

Abstracting Continuous Dynamics

For each mode $l \in \mathbf{Q}$:

if q_{pi}, q_{pj} are abstract variables s.t. $\dot{p_i} = p_j$ in mode l, then apply rules of the form:

- if $q_{pi} = pos$ and $q_{pj} = pos$, then new value q'_{pi} is pos.
- if $q_{pi} = pos$ and $q_{pj} = zero$, then new value q'_{pi} is pos.
- if $q_{pi} = pos$ and $q_{pj} = neg$, then new value q'_{pi} is either pos or $zero$.
- $\ldots$

If $q_{p0}, q_{p1}, q_{p2}, \ldots, q_{pn}$ is s.t. $\dot{p_i} = p_{i+1}$ in mode l, then

- if $q_{p0} = q_{p1} = q_{p2}, \ldots q_{pn-1} = zero$ and $q_{\dot{p}n} = pos$, then new values $q'_{p0} = q'_{p1} = q'_{p2}, \ldots q'_{pn-1} = pos$.

Abstracting Discrete Transitions

Discrete Transition: $(q, \psi(X), q', New(X))$, where

- q, q': modes,
- $\psi(X)$: enabling condition, and
- $New(X)$: assignments to continuous variables.

Abstract Discrete Transition: $((q, \phi_1), (q', \phi_2))$ if

- The formula $\phi_1 \wedge \psi$ is satisfiable and
- The formula $\exists X^o : \psi(X^o) \wedge X = New(X^o) \wedge \phi_2(X)$ is satisfiable.

Discharging Proof Obligations

- Preprocess the formula using logical equivalences (in the theorem prover PVS)
- Use decision procedure for real closed fields, QEPCAD, on the resulting formula (if possible)

This gives an automated abstraction algorithm for hybrid automata that are specified using only expressions in the theory of real closed fields.

We can also detect infeasible abstract states to generate a global invariant of the resulting abstract model.

Features of the Abstraction Algorithm

- *Theorem*[Correctness] The abstractions constructed by the algorithm are sound with respect to the hybrid automata semantics.
- *Theorem*[Relative Completeness] Let ϕ be a QF fmla over X (in $\Re$) that represents the set of reachable states and $P = Poly(\phi)$. Let ψ be the reachable set computed by the algorithm with seed P. If the saturation process terminates, then $\psi = \phi^c$.

Note further that:

- Abstractions can be refined by adding more polynomials,
- Only simple computational steps involved.

Example: Hybrid System

Consider a simple thermostat controller with:

- Discrete modes: Two modes, $q = on$ and $q = \mathit{off}$
- Continuous variable: The temperature x
- Initial State: $q = \mathit{off}$ and $x = 75$
- Discrete Transitions:

$$
\begin{array}{rcl}
q = \mathit{off}\ \ and\ x \leq 70 & \longrightarrow & q' = on \\
q = on\ \ and\ x \geq 80 & \longrightarrow & q' = \mathit{off}
\end{array}
$$

- Continuous Flow:

$$
\begin{array}{rcl}
q = \mathit{off}\ \ and\ x > 68 & \longrightarrow & \dot{x} = -Kx \\
q = on\ \ and\ x < 82 & \longrightarrow & \dot{x} = K(h - x)
\end{array}
$$

Abstraction Algorithm

Consists of three steps:

- Seed: Start with a set P_0 of polynomials of interest

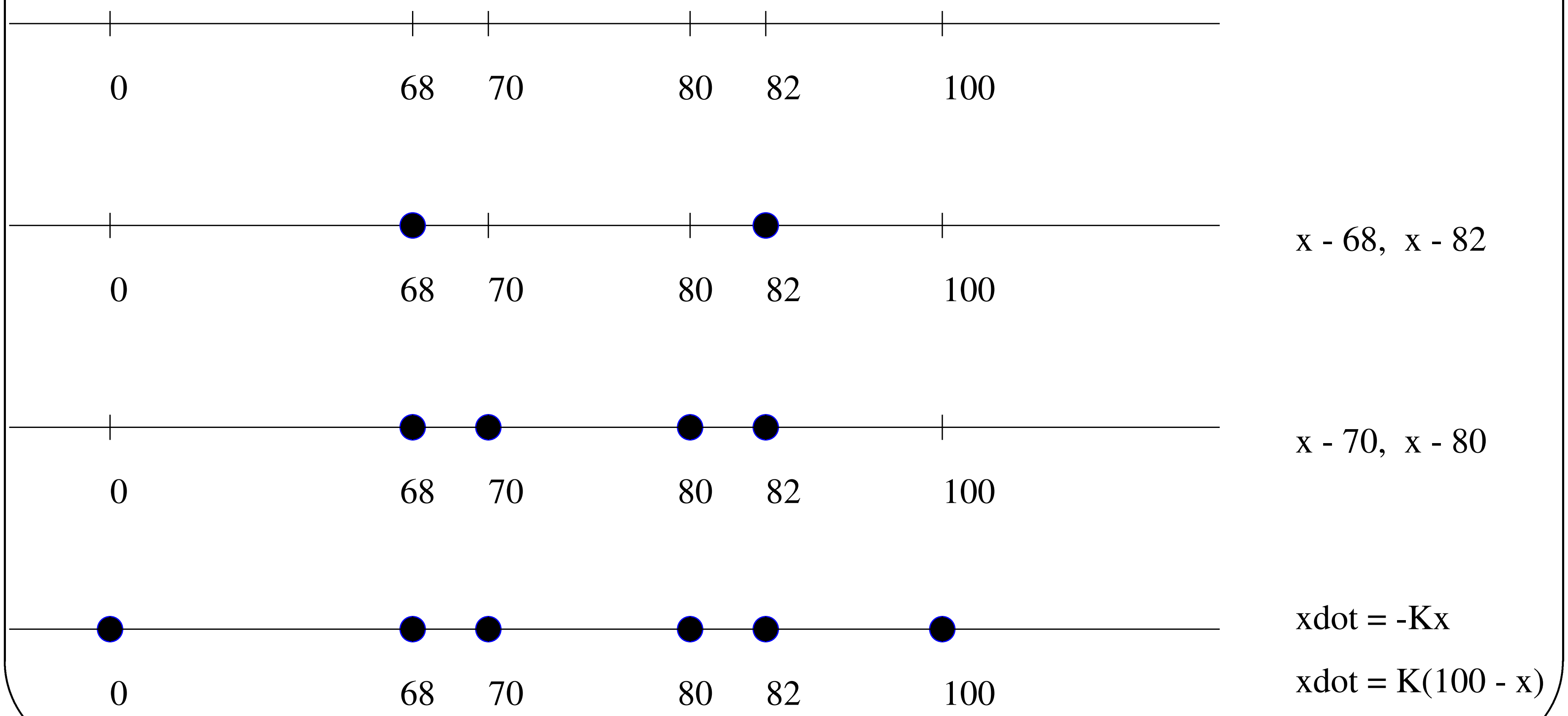
$$P_0 = \{x - 68, x - 82\}$$

- Saturate:

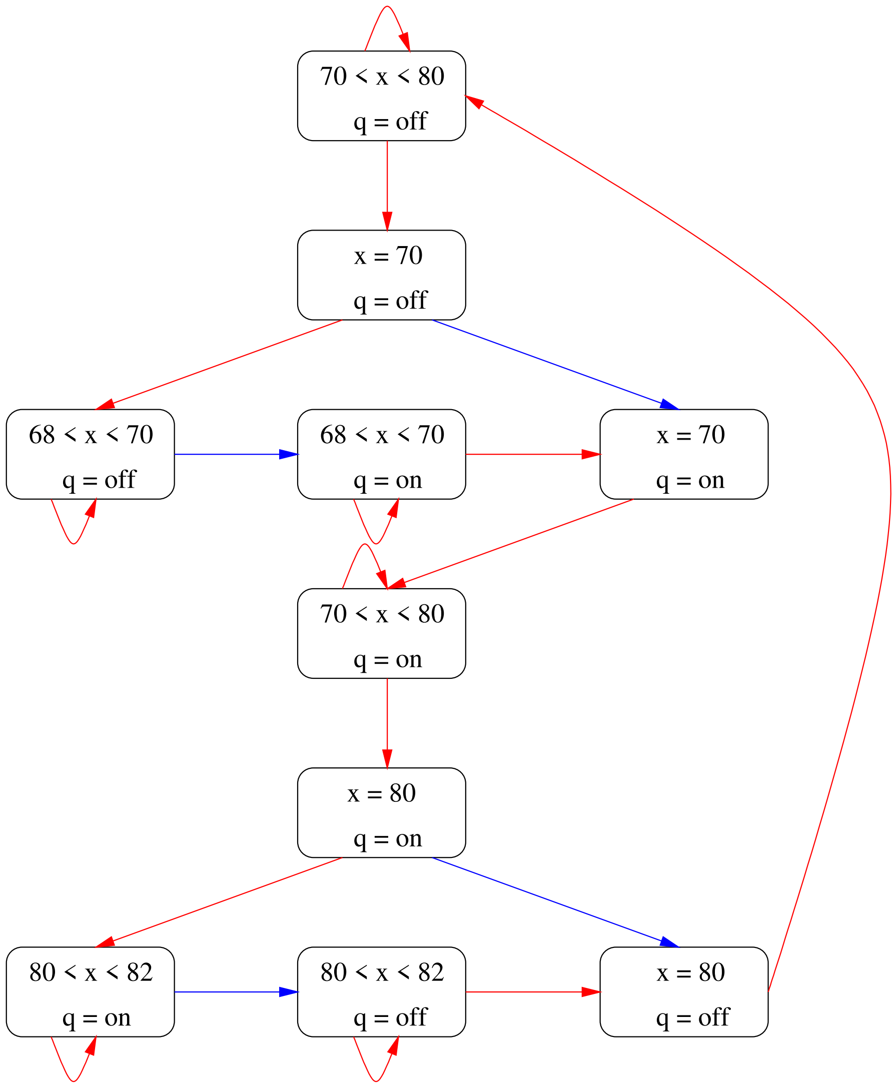
$$\begin{aligned} P_1 &= \{x - 68, x - 82, x - 70, x - 80\} \\ P_2 &= P_1 \cup \{-Kx, K(h - x)\} \\ P_3 &= P_1 \cup \{x, h - x\} \end{aligned}$$

- Abstract

Generating New Polynomials



Abstract Thermostat System



Special Features of the Abstraction Algorithm

- Completely automatic procedure for hybrid automata specified only using polynomials: uses a combination of theorem proving and decision procedure for real-closed fields.
- Abstractions can be refined by either adding new polynomials or higher-order derivatives.
- Discrete transitions are abstracted using well-known data abstraction technique.
- Continuous flows are qualitatively abstracted using arbitrary polynomials over the state variables and their higher-order derivatives.
- The method also applies to more general systems, say with trignometric and exponential functions, but the proof obligations may not be handled automatically.
- Implementation available. Welcome to see the demo!