Symbolic Approaches for Verification of Hybrid Systems

Ashish Tiwari

Tiwari@csl.sri.com

Computer Science Laboratory

SRI International

Menlo Park CA 94025

http://www.csl.sri.com/~tiwari

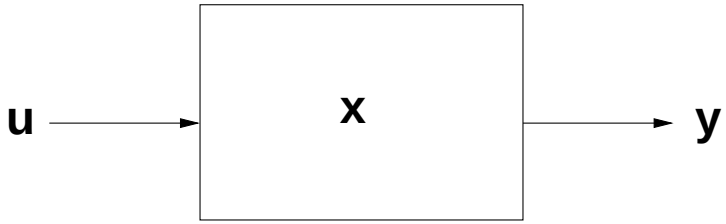
Outline

1. Hybrid Systems: Introduction and motivation
2. High level overview of our approach
3. Details
   - HybridAbstraction
     - intelligent seed forms, composition, locality, language
   - Model Checking
   - Decision Procedures
4. Examples
5. Conclusion

Formal Models: The Past...

- Continuous dynamical systems
  - The state space formulation

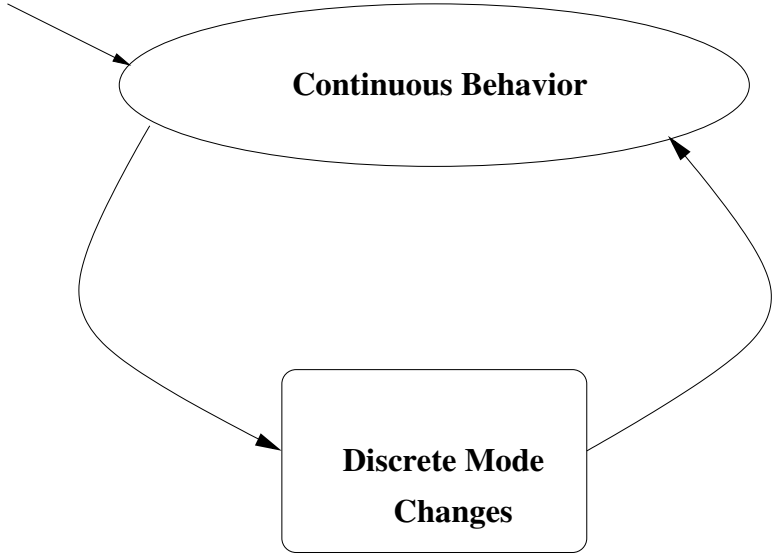
$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), t) \\ y(t) &= h(x(t), t) \end{aligned}$$



- Finite state automata
  - Mealy/Moore machines
  - State transition systems

Hybrid Systems

- Systems are no longer purely continuous, or purely discrete
- But consist of interacting discrete and continuous components

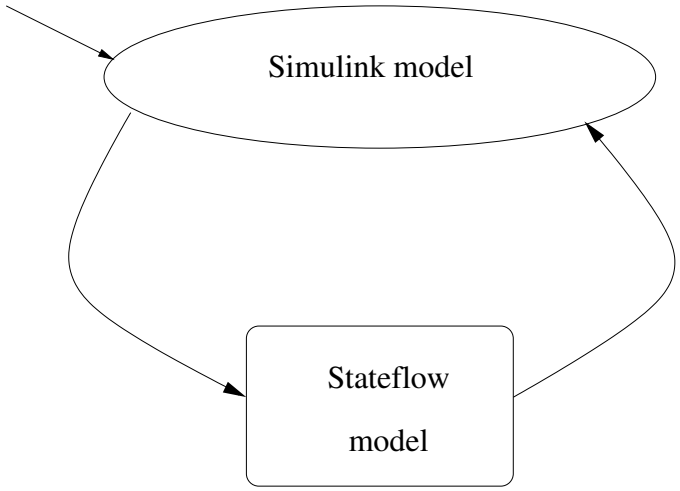


- Embedded systems; controllers for chemical plants, automobiles; genetic networks.

Formal Models: Present...

Putting the two formal models together, Hybrid Automata:

- Embed a continuous dynamical system inside each state
- World now evolves in two different ways
  - Move from one state to another via a discrete transition
  - Remain in the state and let the continuous world evolve
- A modeling tool—



Example: Hybrid System

Consider a simple thermostat controller with:

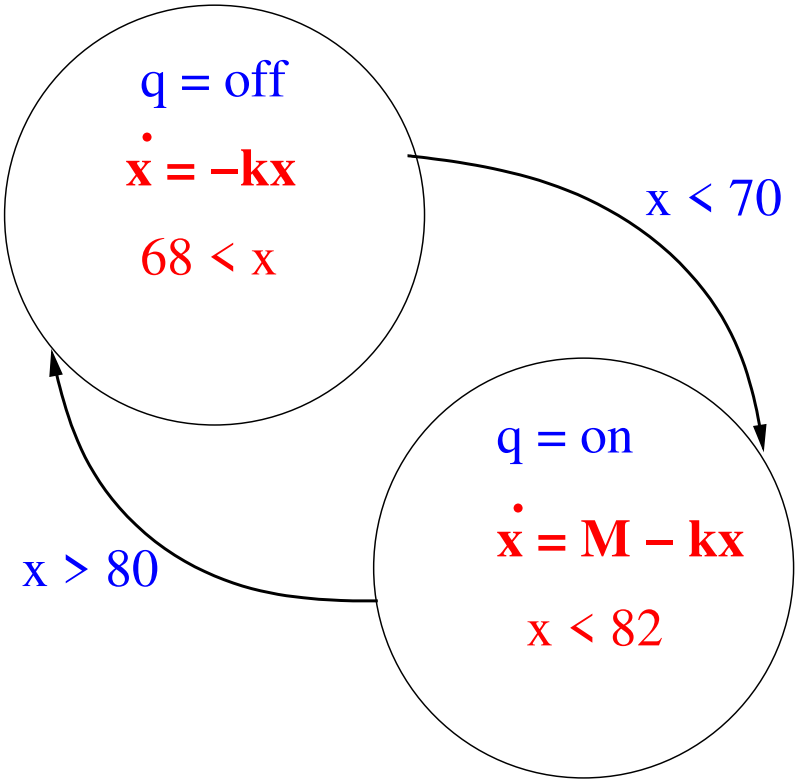
- Discrete modes: Two modes, $q = on$ and $q = off$
- Continuous variable: The temperature x
- Initial State: $q = off$ and $x = 75$
- Discrete Transitions:

$$q = off \ and \ x \leq 70 \quad \longrightarrow \quad q' = on$$
$$q = on \ and \ x \geq 80 \quad \longrightarrow \quad q' = off$$

- Continuous Flow:

$$q = off \ and \ x > 68 \quad \longrightarrow \quad \dot{x} = -Kx$$
$$q = on \ and \ x < 82 \quad \longrightarrow \quad \dot{x} = K(h - x)$$

Hybrid Automata: In picture



Engineers use an "inside-out" representation, as exemplified in Matlab Simulink/Stateflow.

ThermoStat in HybridSAL

LOCAL q : BOOLEAN

LOCAL x : REAL

INITIALIZATION

$$q = TRUE; x = 75$$

TRANSITION

$$q = off \ and \ x \leq 70 \quad \longrightarrow \quad q' = on$$

$$q = on \ and \ x \geq 80 \quad \longrightarrow \quad q' = off$$

$$q = off \ and \ x > 68 \quad \longrightarrow \quad xdot = -Kx$$

$$q = on \ and \ x < 82 \quad \longrightarrow \quad xdot = M - Kx$$

ThermoStat in HybridSAL

Inside-out representation:

LOCAL q : BOOLEAN

LOCAL x, $dummy$: REAL

INITIALIZATION

$x = 75$

TRANSITION

$$\begin{aligned} dummy &= \textit{if } x \leq 70 \textit{ then } M \textit{ elsif } x \geq 80 \textit{ then } 0 \\ &\quad \textit{elsif } q = \textit{on then } M \textit{ else } 0 \textit{ endif } ; \\ q &= \textit{if } x \leq 70 \textit{ then } \textit{ TRUE } \textit{ elsif } x \geq 80 \textit{ then } \textit{ FALSE} \\ &\quad \textit{elsif } q = \textit{on then } \textit{ TRUE } \textit{ else } \textit{ FALSE } \textit{ endif } ; \\ TRUE &\longrightarrow xdot = dummy - Kx \end{aligned}$$

Analysis

What can we say (deduce, compute) about the model?

- Reachability. Is there a way to get from point A to point B
- Safety. Does the system stay out of a bad region
  - Can the car ever collide with the car in front?
  - Can the gear box get stuck in the second gear?
- Liveness. Does something good always happen

Does the model satisfy some property.

Property is described in a logic interpreted over the formal models.

Problem

- Given a hybrid automata
- And a safety property
- Show that the property is true of the model

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

Analysis of Hybrid Systems

- Is not easy
- Complexity is a function of the dimension of the state space
- Current state of the art tools can handle only upto 4 variables!
- Practical models (e.g. gearbox) contain around 100 variables.
- Even for low dimension systems, quite elusive.

Goal: Verify the discrete logic using a very coarse abstraction of the continuous component.

Abstraction: The General Idea

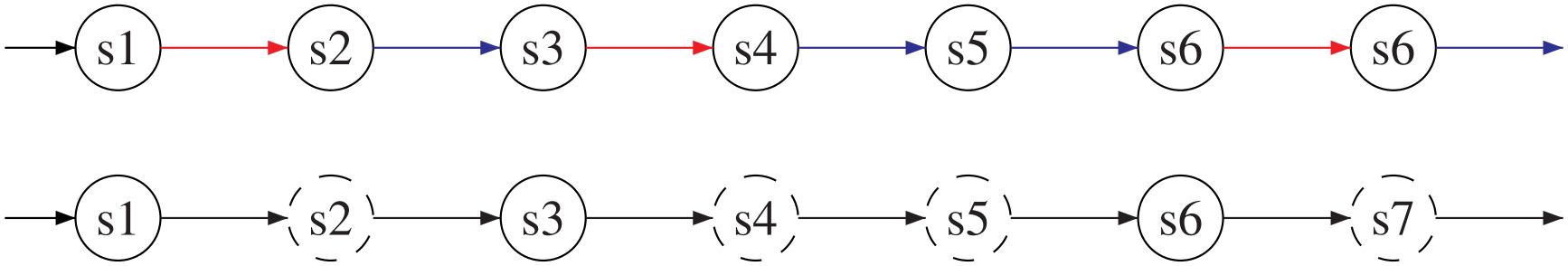
1. Partition the state space: each member of the partition is a state of a new system
   - defines an abstraction map from concrete states onto a smaller set of "abstract states"
   - one abstract state corresponds to a set of concrete states
2. Quotient the original system modulo the partition:
   - 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

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
- $F : \mathbf{Q} \mapsto (\mathbf{X} \mapsto \mathbb{R}^{|X|})$ specifies the rate of flow, $\dot{x} = F(q)(x)$
- $Inv : \mathbf{Q} \mapsto 2^{\mathbb{R}^{|X|}}$ gives the invariant set
- $R \subseteq \mathbf{Q} \times 2^{\mathbf{X}} \mapsto \mathbf{Q} \times 2^{\mathbf{X}}$ 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

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

Top Level Verification Approach

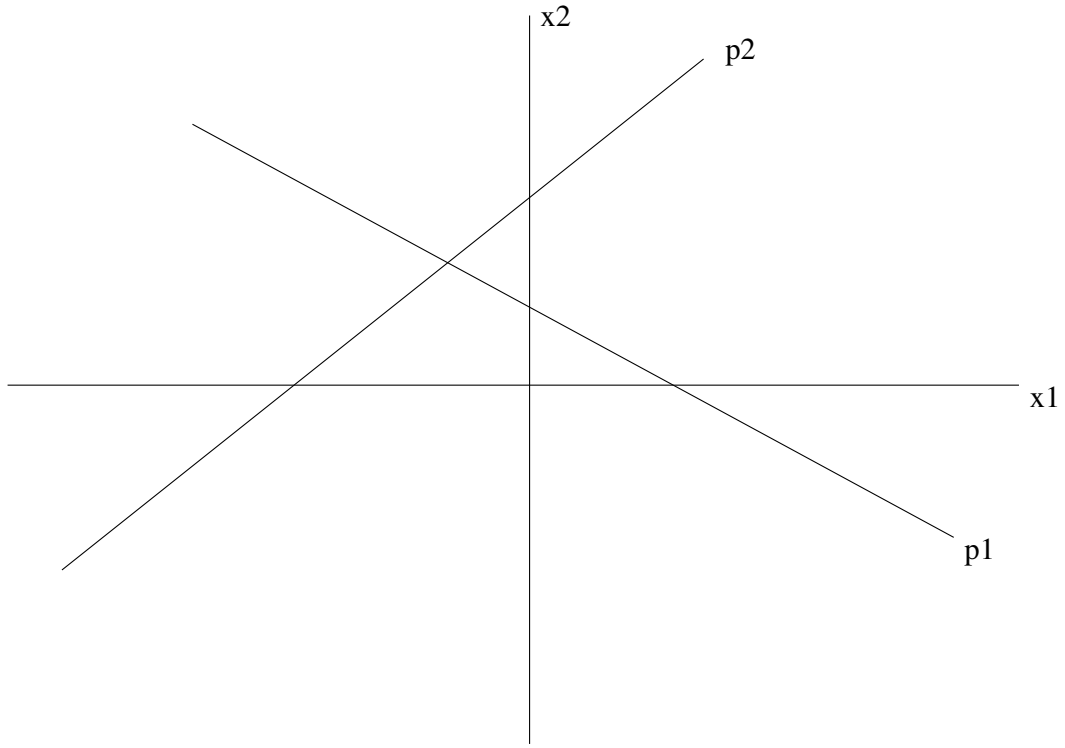
- Hybrid Abstraction For hybrid systems:
  - What abstraction mapping?
  - How to abstract the continuous dynamics?
  - How to abstract the discrete dynamics?
  - What theorem proving obligations, which theory?
- Model Checking

Partitioning the State Space

- The discrete component, $\mathbf{Q}$, of the state space, $\mathbf{Q} \times \mathbf{X}$, is not abstracted
- The continuous component, $\mathbf{X}$, of the state space is partitioned based on the signs of finite number of functions over the continuous state variables
  - Let $f_1, f_2, \ldots, f_m$ be functions from $\mathbf{X}$ to $\mathbb{R}$
  - Let $q_{f1}, q_{f2}, \ldots, q_{fm}$ be m corresponding discrete variables interpreted over $\{pos, neg, zero\}$ s.t.
    - $q_{f1} = pos$ iff $f_1 > 0$
    - $q_{f1} = neg$ iff $f_1 < 0$
    - $q_{f1} = zero$ iff $f_1 = 0$

Abstraction Algorithm: 1

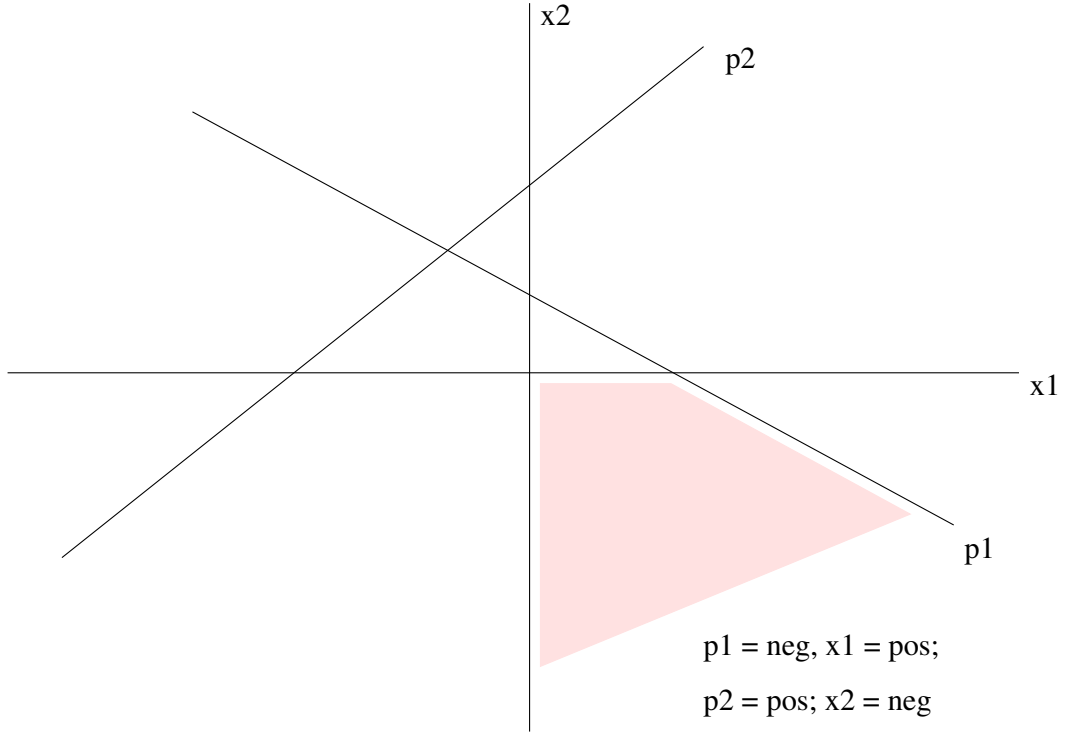
Consider a continuous dynamical system with two state variables. Concrete state space: $\mathbb{R}^2$



Partitioned w.r.t signs of four linear forms x_1, x_2, p_1, and p_2.

Abstraction Algorithm: 2

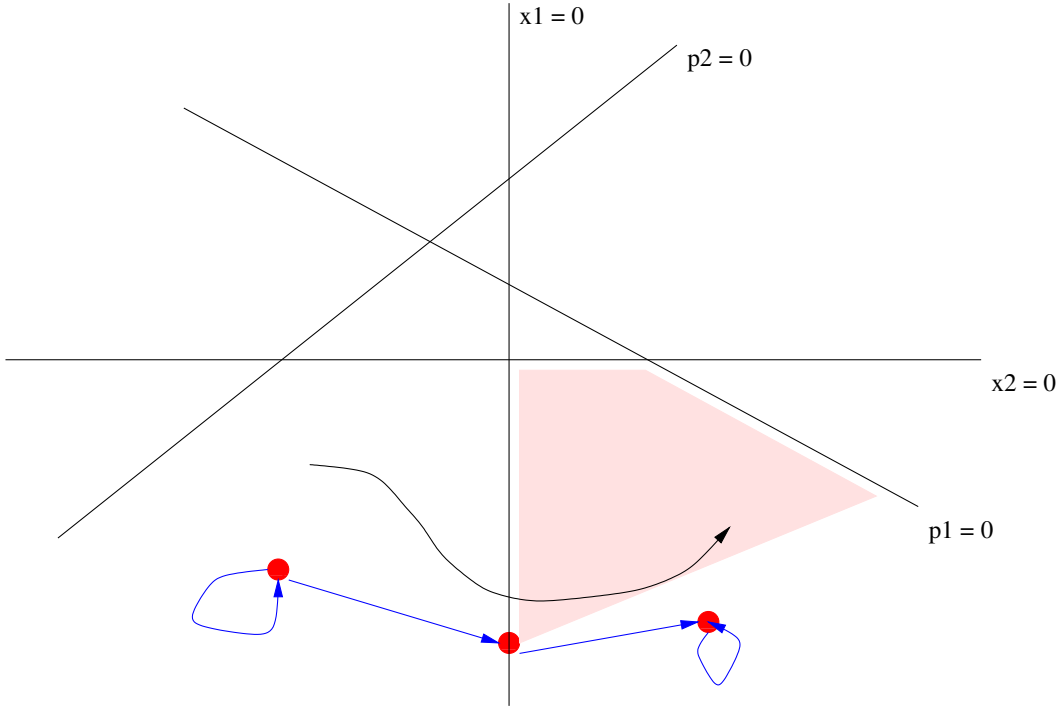
Abstract states correspond to subsets of concrete states.



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

Abstraction Algorithm: 3

Abstract transitions overapproximate concrete transitions.



How to abstract the continuous transitions?.

Abstracting the Continuous Dynamics

Concrete state space : $\mathbf{Q} \times \mathbb{R}^n$

Abstract state space : $\mathbf{Q} \times 3^m$

Our approach is based on qualitative reasoning

- For a given mode (continuous dynamics), for each f_i

$$f_i > 0 \ \wedge \ \dot{f}_i \geq 0 \quad \longrightarrow \quad f_i' > 0$$
$$f_i > 0 \ \wedge \ \dot{f}_i < 0 \quad \longrightarrow \quad f_i' > 0 \ \vee \ f_i' = 0$$

- Do this for all possible modes in the system to get abstractions of the continuous evolutions in all modes

But how do I know the sign of $\dot{f}_1$

Choosing Predicates for Abstraction

- If $\dot{f}_i \in \{f_1, f_2, \ldots, f_m\}$, then we know the sign of $\dot{f}_i$
  - Explicitly add derivatives of f_i's to the set
- If not, we could still deduce the sign of $\dot{f}_i$ from
  - the signs of $\{f_1, f_2, \ldots, f_m\}$
  - state invariants

  More theorem proving proof obligations!
- Add those functions to the set whose sign can be "easily" inferred
  - For linear dynamical systems, forms corresponding to eigenvectors of transpose

Abstraction Algorithm

Consists of three steps:

- Seed: Start with a set P_0 of polynomials of interest. $G(68 < x < 82)$.

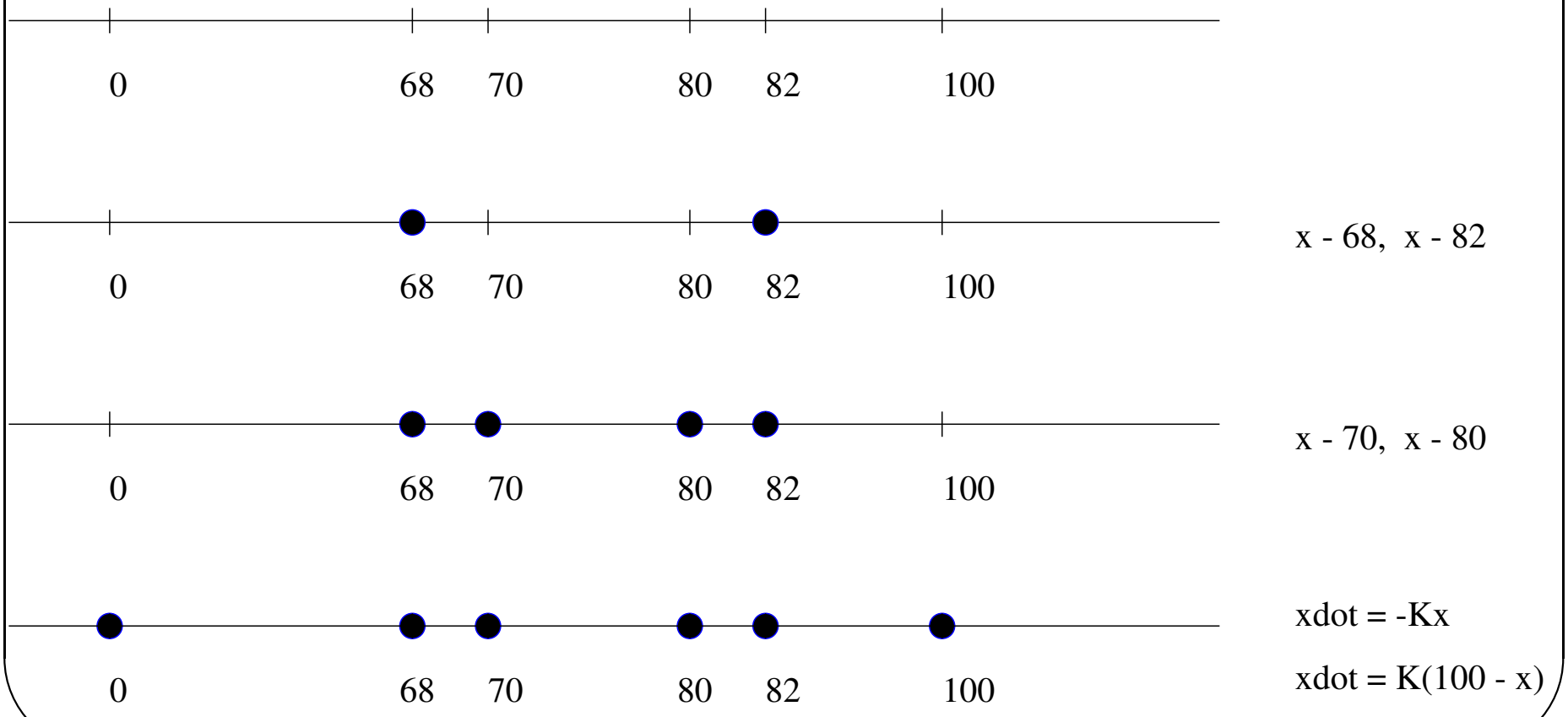
$$P_0 = \{x - 68, x - 82\}$$

- Saturate:

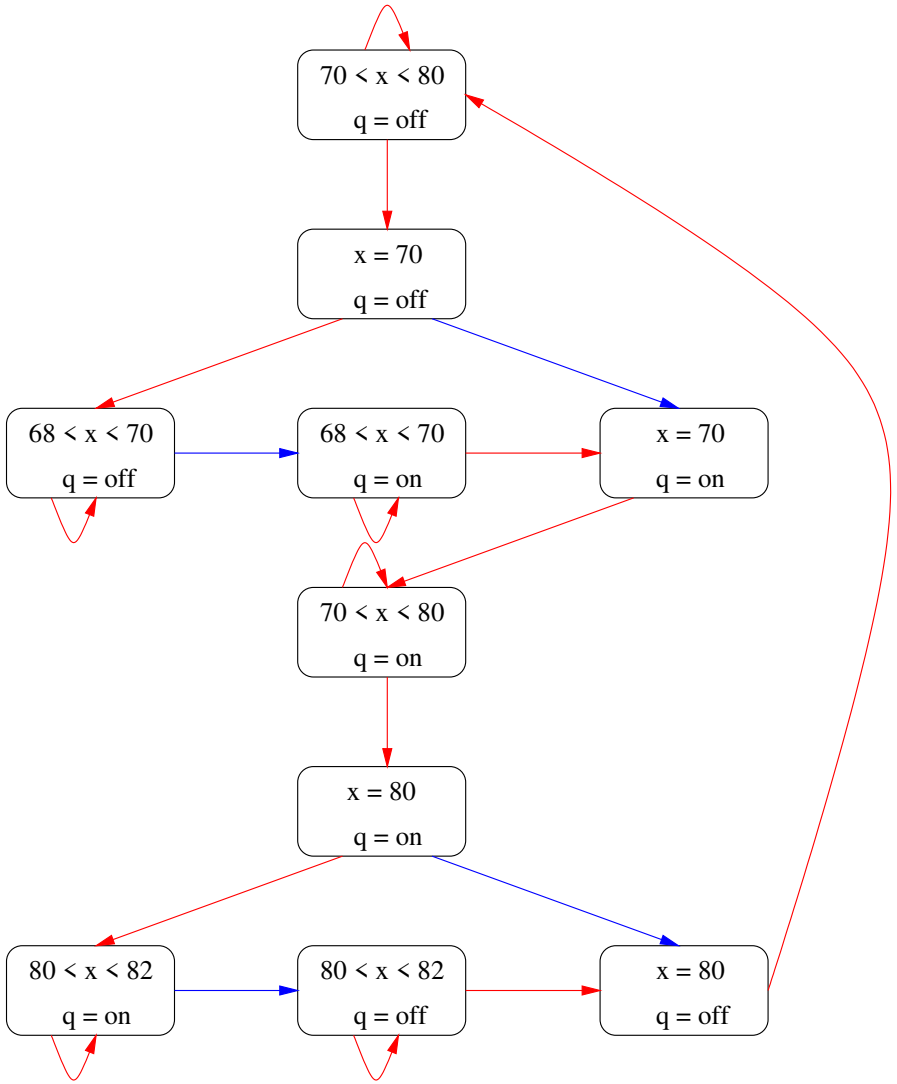
$$\begin{aligned} P_1 &= \{x - 68, x - 82, x - 70, x - 80\} \\ P_2 &= P_1 \cup \{-Kx, M - Kx)\} \\ P_3 &= P_1 \cup \{x, M - Kx\} \end{aligned}$$

- Abstract

Generating New Polynomials



Abstract Thermostat System



Choosing Partition Forms

Linear system $\dot{\vec{x}} = A\vec{x}$

Assume linear form $f = \vec{r}^T\vec{x}$

Question: For what $\vec{r}$ is this "good"?

We want $\dot{f} = kf$

i.e., $\dot{\vec{r}}^T\vec{x} = k\vec{r}^T\vec{x}$

i.e., $\vec{r}^T A\vec{x} = k\vec{r}^T\vec{x}$

i.e., $\vec{r}$ is an eigenvector of A^T corresponding to eigenvalue k

We want k to be real

For such f, qualitative abstraction of cont dynamics is easy

Choosing Partition Forms

Linear system $\dot{\vec{x}} = A\vec{x}$

Assume linear form $f = \vec{r}^T\vec{x}$

Question: For what $\vec{r}$ is this "good"?

We want $\ddot{f} = b\dot{f} + cf$

i.e., $\ddot{\vec{r}}^T\vec{x} = b\dot{\vec{r}}^T\vec{x} + c\vec{r}^T\vec{x}$

i.e., $\vec{r}^T A^2\vec{x} = b\vec{r}^T A\vec{x} + c\vec{r}^T\vec{x}$

i.e., $\vec{r}$ is in the null space of $(A^T)^2 - bA^T - cI$

Generalize to a primary decomposition for A^T!

For such f, sign of $\ddot{f}$ can be inferred from the signs of f and $\dot{f}$, at least in some cases

Choosing Partition Forms: Primary Decomposition

Say $ch(A) = \Pi_i(a_i\lambda^2 + b_i\lambda + c_i)$, $a_i, b_i, c_i \in \mathbb{R}$

Let $\vec{r} \in Null(a_i(A^T)^2 + b_i A^T + c_i I)$

Then, $f = \vec{r}^T\vec{x}$ is a "good" linear form for abstraction

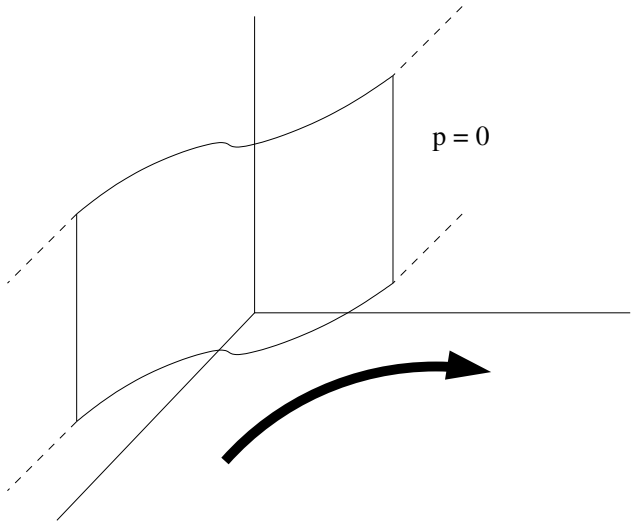
What does "goodness" translate to?

- If $a_i = 0$, then $f = f_0 e^{\frac{-c_i}{b_i}t}$
- If $b_i = 0$, then f oscillates between a min and max value
- If $b_i < 0$, then f oscillates while exponentially decaying, still bounded

Illustration

Let $[r_1, r_2, r_3]$ be an eigenvector corresponding to a negative real eigenvalue of a 3 dimensional linear system.

Let $p = r_1x_1 + r_2x_2 + r_3x_3$.



- If initial states are all on one side, then the systems continues to remain on the same side, i.e., always $p \geq 0$ or always $p \leq 0$
- At, say, $x_2 = 1$ and $x_3 = 2$, the value of x_1 can be bounded

Consequences

- Can do non-trivial reachability computation for linear systems with mixed eigenvalues
  - existing decidability results can not handle this class of systems
  - we extract as much information from the system as available, and bridge the gap between the decidable and undecidable problems
  - incrementally getting to the decidability result: the more eigenvalues are negative reals, the better reachability sets you generate
  - In the limit, do we get back the decidability results?

Observations

Additional interesting consequence:

- Do not need to explicitly compute the real eigenvalues or the eigenvector $\vec{r}$
  - the eigenvalue and eigenvector are easily seen to be algebraic
  - symbolic decision procedures for real closed fields can handle the algebraic expressions representing these eigenvalues
- We assumed linear forms and linear dynamics
- Can we get useful partition forms for nonlinear dynamics using nonlinear forms?

Abstracting the 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

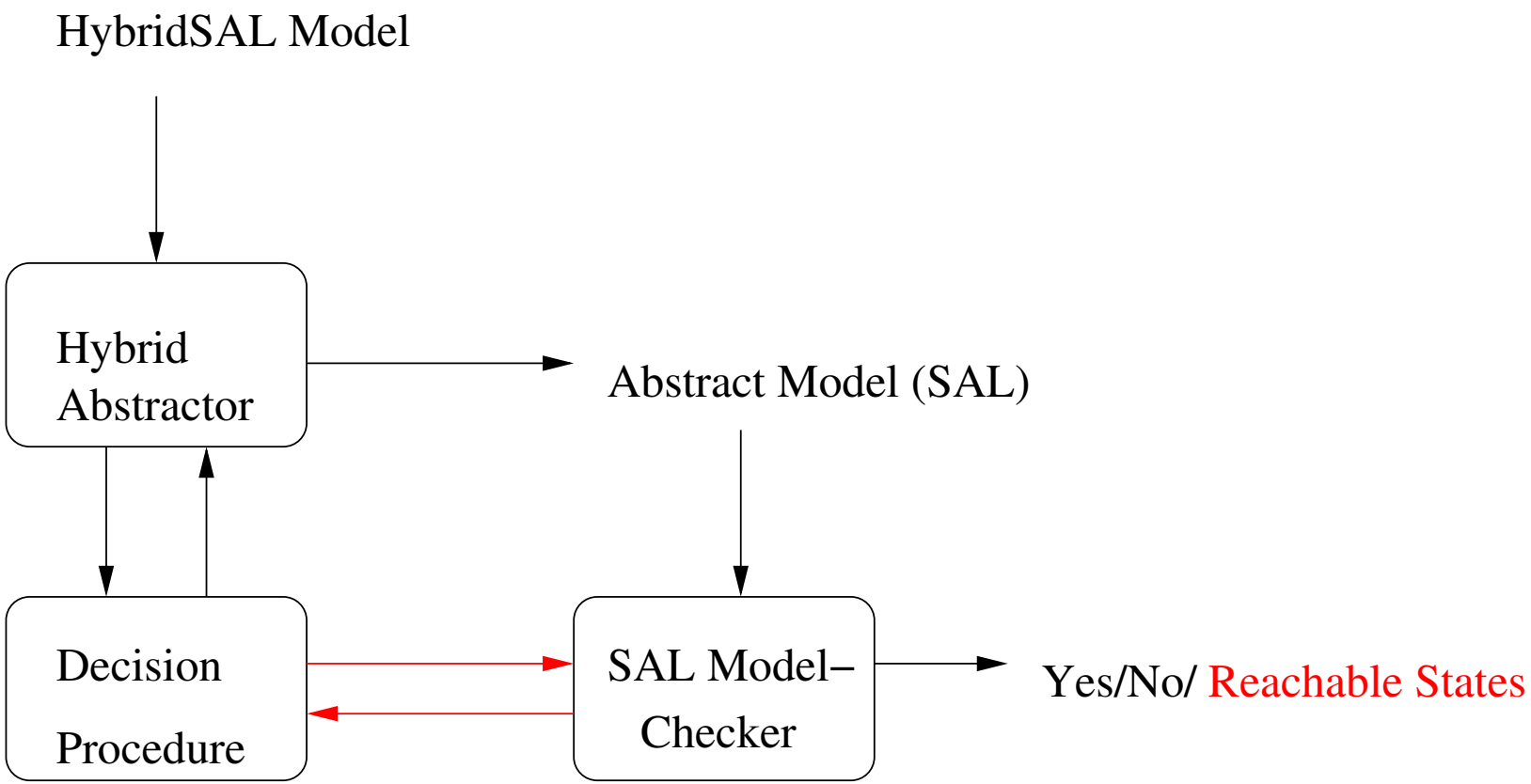
- The formula $\phi_1 \wedge \psi$ is satisfiable and
- The formula $\exists X^o : \psi(X^o) \wedge X = New(X^o) \wedge \phi_2(X)$ is satisfiable.

Hybrid Abstraction Summary

Consists of three steps:

- Seed: Start with a set $\{f_1, \ldots, f_m\}$ of forms of interest
  - Expressions specifying the initial region $\mathbf{S_0}$, transition conditions (specification of $\mathbf{R}$), resets (specification of $\mathbf{R}$)
- Saturate: Close the above set under derivative computation
  - For each form f_i, and each location $l \in \mathbf{Q}$, compute $\dot{f}$, given $\dot{X} = F(l)(X)$
  - Add $\dot{f}_i$ to the seed set and recurse
  - Add "interesting" forms computed by other methods
- Add Transitions:
  discrete transitions abstracted using "usual" predicate abstraction
  continuous transitions abstracted qualitatively

Tool Architecture



Discharging Proof Obligations

- If specification uses polynomials and all forms f are polynomials, then we can use a decision procedure for the theory of real closed fields
- QEPCAD is one such decision procedure implementation
- But, abstractor generates thousands of proof obligations
- QEPCAD not robust, fast, and logical enough
- We use a home made restricted chaining calculi based decision procedure, which is sound but incomplete
- Because HybridAbstraction makes a failure-tolerant use of the decision procedure

This gives an automated abstraction algorithm for polynomial hybrid automata

Which can be extended by extending the decision procedure

Satisfiability in the Theory of Reals

Equations—

$$\frac{\begin{array}{c} c_0 s_0 = p \\ d_0 t_0 = q \end{array}}{d_0 p \frac{lcm(s_0,t_0)}{s_0} = c_0 q \frac{lcm(s_0,t_0)}{t_0}} \qquad \frac{\begin{array}{c} x^2 = 1 \\ xy = 2 \end{array}}{y = 2x}$$

- Gröbner basis computation on equations
- Restricted further, in implementation, by $s_0 | t_0$ or $t_0 | s_0$
- Replacing deduction by simplification

The inequalities: Chaining

$$\frac{s_0 + p > 0 \quad -t_0 + q > 0 \quad s_0 t_0 + r > 0}{q s_0 - p t_0 + pq + r > 0} \qquad \frac{x - 1 > 0 \quad -y > 0 \quad xy > 0}{y > 0}$$

- Similar rules for the interaction of $=$, $>$, and $\geq$
- Simplification rules
- Sound and Incomplete
- Terminating? No! Can emulate iterative root-finding numerical algorithms!!! So, force termination.

Welcome to the real world

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 $\psi = \phi^c$.

  Implications to linear systems:

  - nilpotent
  - all real eigenvalues

Example: Cruise Control

Example. Consider a cruise control:

$$\begin{aligned} \dot{v} &= a \\ \dot{a} &= -4v + 3vf - 3a + gap \\ g\dot{a}p &= -v + vf \end{aligned}$$

where v, a is the velocity and acceleration of this car, vf is the velocity of car in front, and gap is the distance between the two cars.

Cruise Control: Example Contd

The transpose matrix A^T has one negative real eigenvalue λ.

If $\vec{r} = [r1, r2, r3, r4]^T$ is the eigenvector corresponding to λ, then consider the polynomial

$$p = r1 * v + r2 * vf + r3 * a + r4 * gap$$

Why is this special?

$$\begin{aligned} \frac{dp}{dt} &= \frac{d}{dt}([v, vf, a, gap]\vec{r}) \\ &= (A[v, vf, a, gap])^T \vec{r} \\ &= [v, vf, a, gap] A^T \vec{r} \\ &= [v, vf, a, gap]\lambda\vec{r} = \lambda p \end{aligned}$$

For negative λ, the polynomial p is monotonically exponentially decaying function over all trajectories of the dynamical system specified by A.

Genetic Networks

A bacteria exhibits variety of complex behaviors: It is reactive with a fine tuned "control"

- Concentration of various proteins are responsible
- Which are produced by transcription and translation
- Which are controlled by protein concentrations

Traditionally, biologists model inhibition and activation of genes by proteins by continuous maths too!

Biological systems are best modeled as reactive hybrid systems.
–David Harel (HSCC'03)

Hybrid Models of Metabolic Pathways

- The dynamics of protein concentrations are modeled using differential equations

$$\frac{d[SinI]}{dt} = DSinI - K*[SinI] - K1*[SinI]*[SinR] + K2*[SinI - SinR]$$

- Influence of protein concentration on transcription is captured using discrete switches

$$DSinI = IF\ ([Spo0AP] = HIGH)\ THEN\ 1\ ELSE\ 0\ ENDIF$$

- This simplifies the differential equations tremendously by removing large chunks of expressions

$$DSinI = f([Spo0AP], k1, k2, \ldots)$$

- Containing parameters with unknown values

$$k1 = ?, k2 = ?$$

Future Work

- Abstraction based theorem proving: *Theorem provers should abstract*
- Sound, but incomplete hierarchies of decision procedures: *Projection*
- Approximating reach sets using structural analysis of the differential equations: *reinventing linear algebra*
- Composition—
  - continuous state space partitioned differently in different modes
  - each mode informs the other what it expects to know
- Model checker: interfaces with the decision procedure for model checking lazy abstractions
- Bounded model checking for such "imprecise" abstractions
- Other interesting "seed" forms— symbolically integrate forms, nonlinear systems

Summary

- 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, if the decision procedure can handle them