Symbolic techniques 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

The World: Then...

- 20th century modelling of the physical world by physicists, chemists, biologists, engineers, etc:
  - Continuous state variable
  - Updated according to differential equations

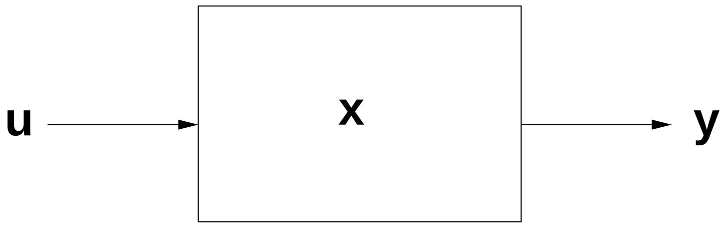
$$m\ddot{x} = F - k\dot{x}$$

- Advent of computational machines
  - Discrete variables
  - Updated according to specific rules

Formal Models: Legacy...

- 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

The World: Now...

- Machines interacting with the physical world
  - embedded systems
  - controllers for chemical plants, airplanes
- Or, maybe a new approach to model the same old continuous world
  - analysis need does not translate to restricting to continuous mathematics
  - some state variables are better viewed discrete

Formal Models: Now...

Putting the two formal models together, Hybrid Automata:

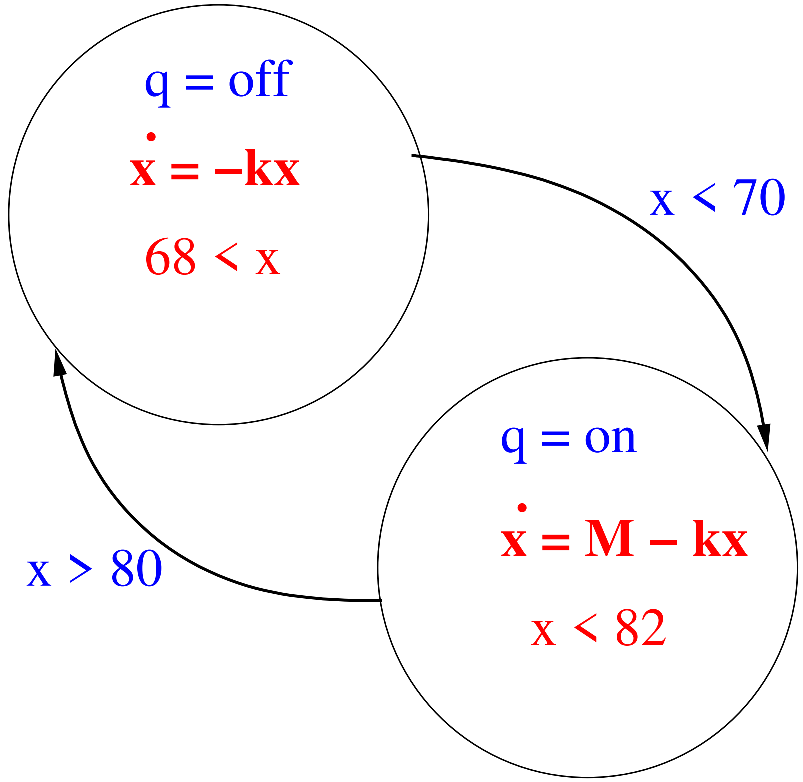
- Embed a continuous dynamical system inside each state
- World now evolves in two different ways
  - Move from one state to another via a discrete transition
  - Remain in the state and let the continuous world evolve
- System has different modes of operation, while some discrete logic performs mode switches

Hybrid Automata

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

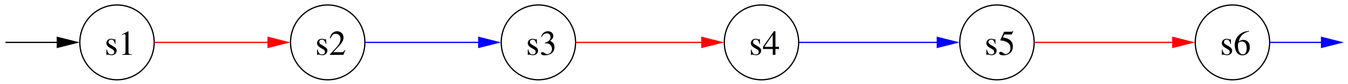
- Q: finite set of discrete variables
- X: finite set of continuous variables
- $\mathbf{X} = \mathbb{R}^{|X|}$, $\mathbf{Q}$ = set of all valuations for Q
- $\mathbf{S} = \mathbf{Q} \times \mathbf{X}$
- $\mathbf{S}_0 \subseteq \mathbf{S}$ is the set of initial states
- $F : \mathbf{Q} \mapsto (\mathbf{X} \mapsto \mathbb{R}^{|X|})$ 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

Hybrid Automata: In picture



Engineers use an "inside-out" representation, as exemplified in Matlab Simulink/Stateflow: *I could scare you, but I won't.*

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

$$\begin{aligned} x_{i+1} &= f(\delta) & x_i &= f(0) \\ \dot{f} &= F(l) & f(t) &\in Inv(l) \textit{ for } 0 \le t \le \delta \end{aligned}$$

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

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 Abstraction

To abstract a hybrid system into a finite-state discrete transition system

- Abstraction mapping, or Partitioning the state-space
- Abstracting the discrete dynamics
- Abstracting the continuous transitions

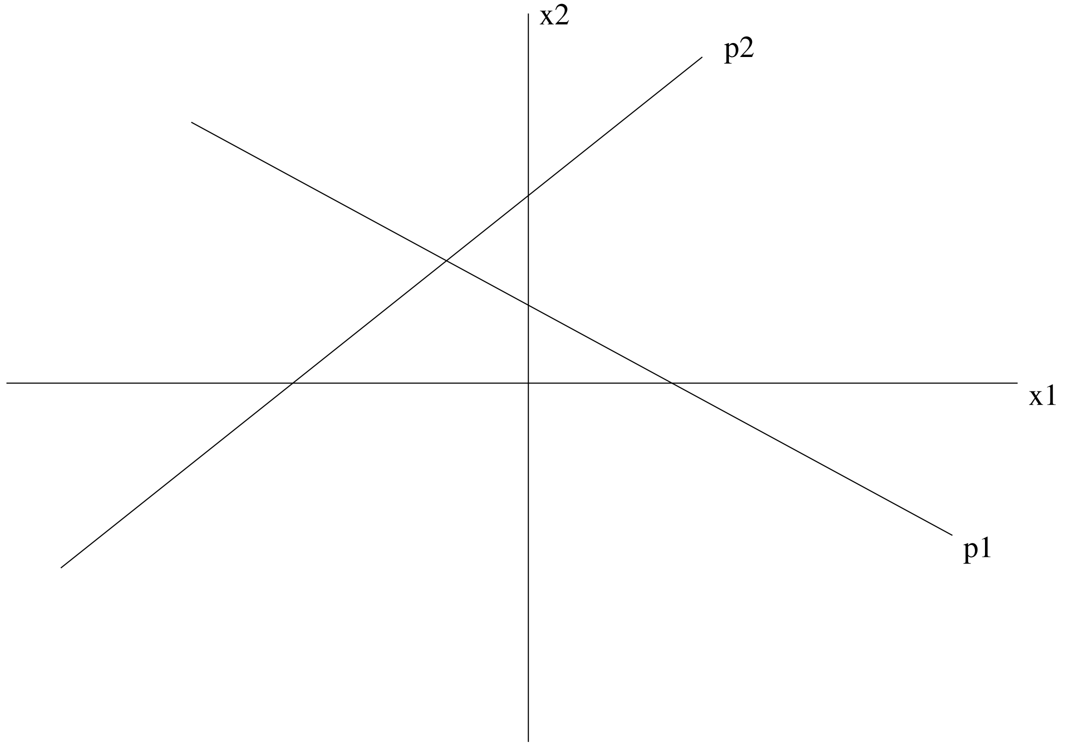
Are all the steps computable/decidable?

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: $\mathbb{R}^2$

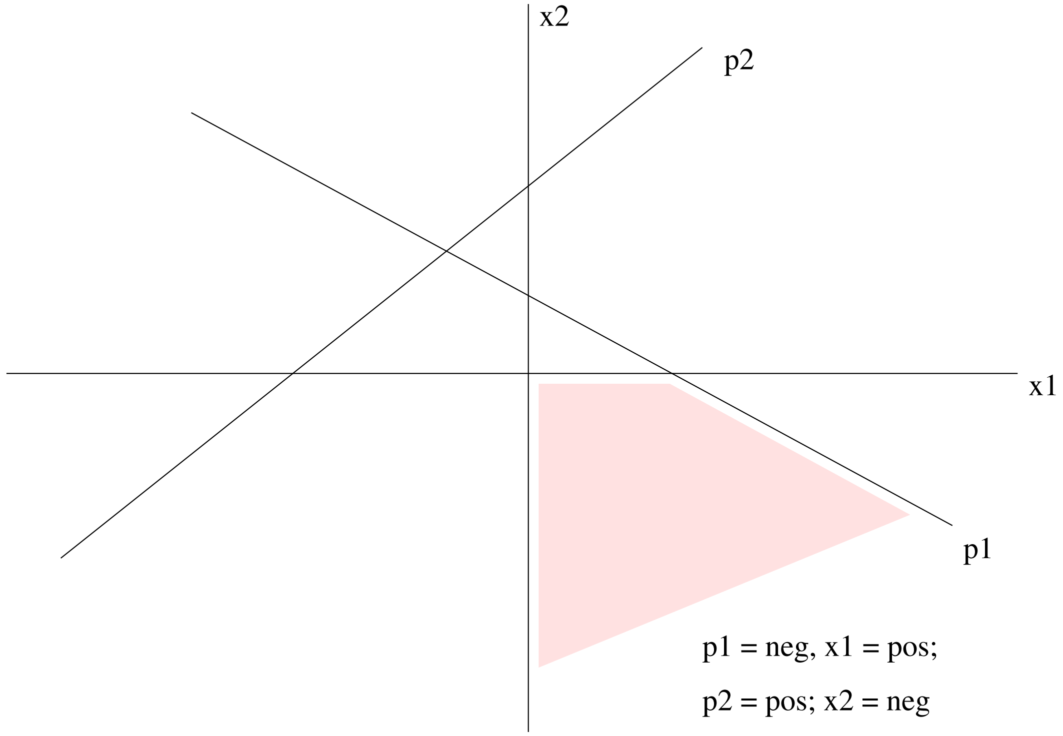


Partitioned w.r.t signs of four linear forms x_1, x_2, p_1, and p_2.

Abstraction Algorithm: 2

Abstract states correspond to subsets of concrete states.

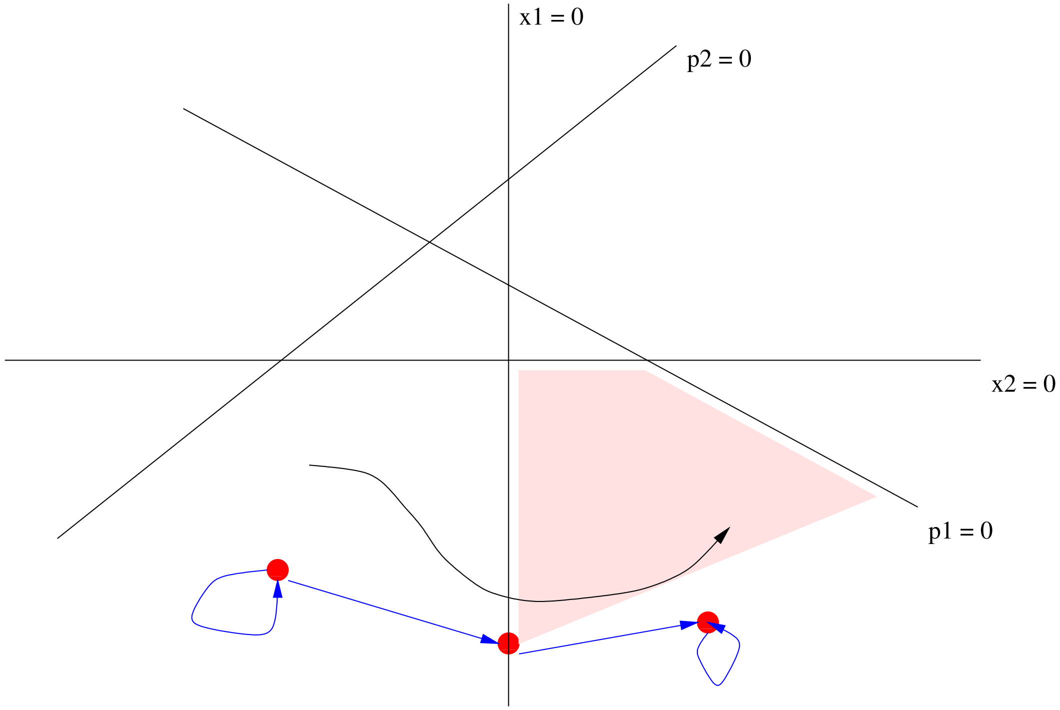
$$x1 > 0, \quad x2 < 0, \quad p1 < 0, \quad p2 > 0$$



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 : $\mathbf{Q} \times 3^m$

Question: Fix a mode. Given

1. an abstract state $f_1 \begin{matrix} > \\ < \\ = \end{matrix} 0,\ f_2 \begin{matrix} > \\ < \\ = \end{matrix} 0,\ \ldots,\ f_m \begin{matrix} > \\ < \\ = \end{matrix} 0$

2. mode dynamics, $\dot{x}_1 = g_1,\ \ldots,\ \dot{x}_n = g_n$

determine all new abstract states $f_1 ?\, 0,\ \ f_2 ?\, 0,\ \ \ldots,\ \ f_m ?\, 0$ reachable from the given abstract state.

Abstracting Cont. Dynamics: the dual question

What is the sign of f_i in the next state?

Our approach is based on qualitative reasoning. If

$$f_1 \begin{smallmatrix} > \\ < \\ = \end{smallmatrix} 0 \;\wedge\; f_2 \begin{smallmatrix} > \\ < \\ = \end{smallmatrix} 0 \;\wedge\; \cdots \;\wedge\; f_m \begin{smallmatrix} > \\ < \\ = \end{smallmatrix} 0 \;\wedge\; \text{state-invariant of mode}$$

$$\Rightarrow \qquad \dot{f}_i > 0$$

then, sign of f_i in the next state is

- $\{pos\}$ if $f_i > 0$ now,
- $\{neg, zero\}$ if $f_i < 0$ now,
- $\{pos\}$ if $f_i = 0$ now.

Decision Procedure

We need a decision procedure to prove

$$f_1 \begin{array}{c} > \\ < \\ = \end{array} 0 \wedge f_2 \begin{array}{c} > \\ < \\ = \end{array} 0 \wedge \cdots \wedge f_m \begin{array}{c} > \\ < \\ = \end{array} 0 \ \wedge \ \text{state-invariant} \ \Rightarrow \ \dot{f}_i > 0$$

If f_i's, $\dot{f}_i$ are polynomials, and state-invariant also only consists of polynomials, then we can use a decision procedure for the QF-theory of reals.

Failure-tolerant Theorem Proving: sound, but incomplete, procedure suffices.

Implementation optimization: (i) Do a clever 3^m enumeration; (ii) Use witness generation capability of decision procedure.

Choosing Predicates for Abstraction

- From initial states, property, and guards
  - If you want to prove $G(x > y)$, add $x - y \begin{array}{c} > \\ < \\ = \end{array} 0$.
- 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
- Add those functions to the set whose sign can be "easily" inferred
  - For linear dynamical systems, forms f corresponding to eigenvectors of transpose satisfy $\dot{f} = \lambda f$.
  - f s.t. $\dot{f} = 0$

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, transition conditions (guards, state-invariants), resets, and property
- Saturate: "Add" derivatives of f_i's to this set
  - For each form f_i, and each location $l \in \mathbf{Q}$, compute $\dot{f}$
  - 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

Abstracting Discrete Transitions

Concrete Discrete Transition:

$$q \quad \xrightarrow[X := New(X)]{guard(X)} \quad q'$$

Abstract Discrete Transition:

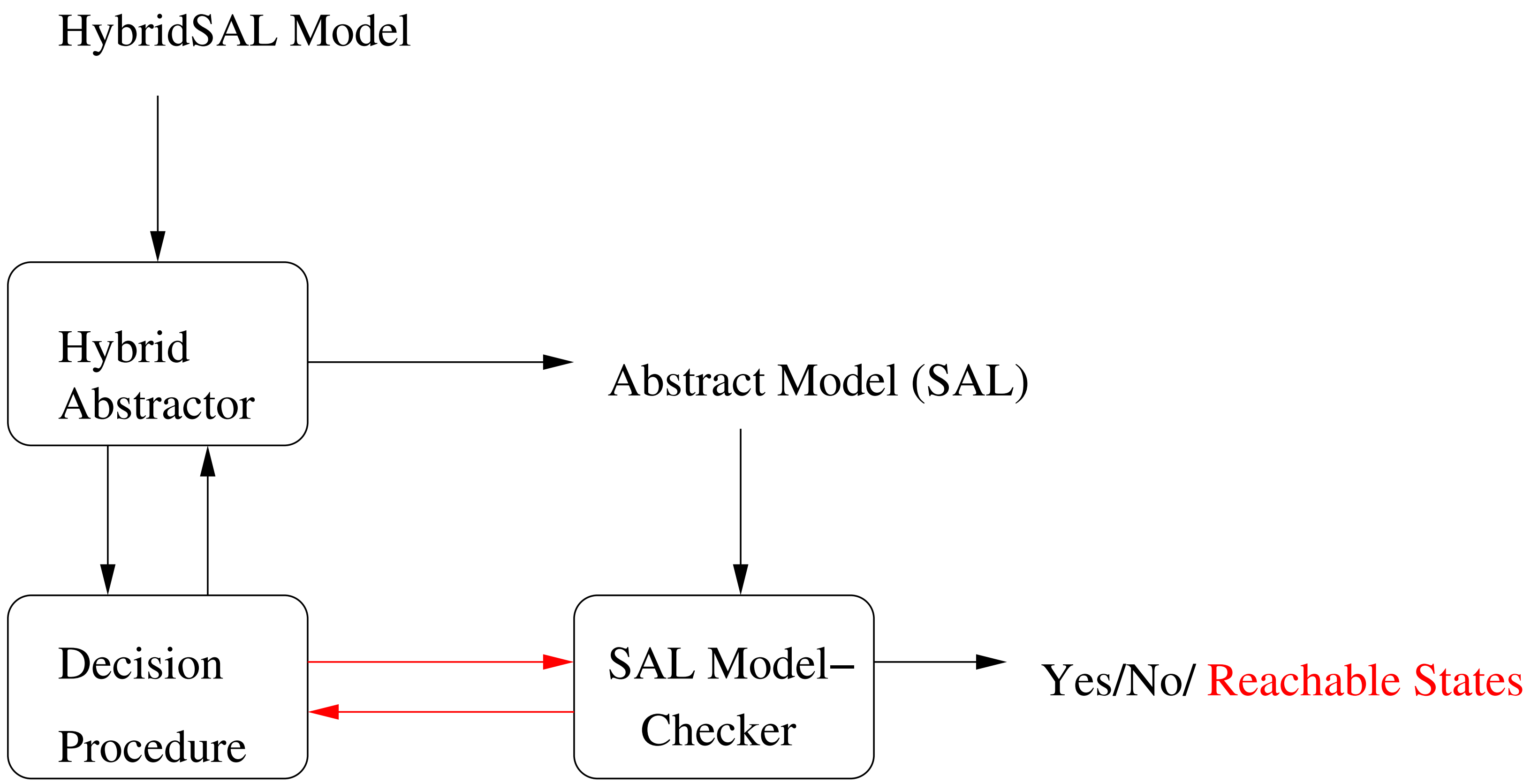
$$(q, f_1 \;\substack{> \\ < \\ =}\; 0, \ldots, f_m \;\substack{> \\ < \\ =}\; 0) \longrightarrow (q', f_1 \;\substack{>' \\ < \\ =}\; 0, \ldots, f_m \;\substack{>' \\ < \\ =}\; 0)$$

If the following formulas are satisfiable

$$guard(X) \wedge f_1 \;\substack{> \\ < \\ =}\; 0 \wedge \cdots \wedge f_m \;\substack{> \\ < \\ =}\; 0$$

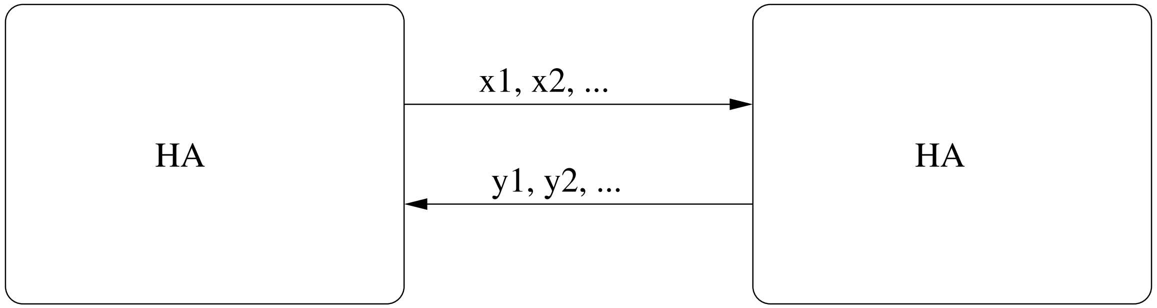
$$\exists X^o : guard(X^o) \wedge X = New(X^o) \wedge f_1 \;\substack{>' \\ < \\ =}\; 0 \wedge \cdots \wedge f_m \;\substack{>' \\ < \\ =}\; 0$$

Tool Architecture



Composition

System consists of two (or more) interacting HA.

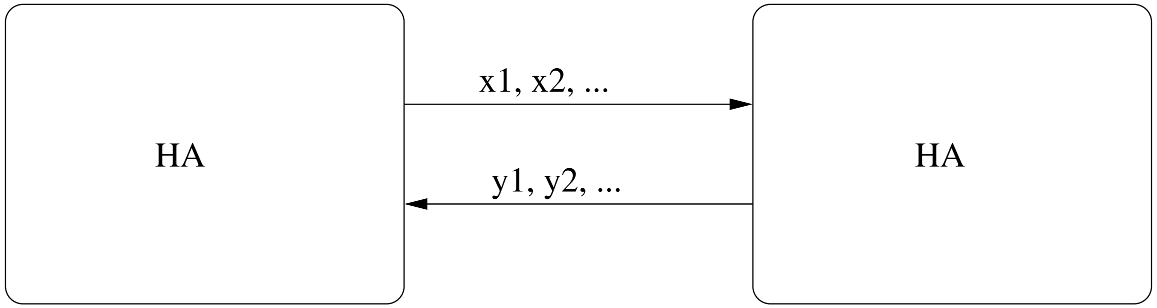


Semantics:

- Time evolution transitions occur simultaneously (synchronized).
- Discrete steps occur asynchronously.

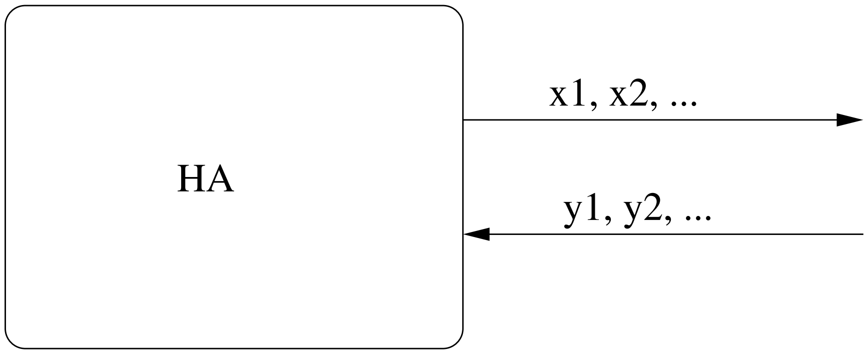
Compositional Abstraction

Depends on the amount of interaction.



- Interaction: Expressions on "input" variables that HA-2 uses:
  - $p(x_1, x_2, \ldots), \ldots,$
  - $q(x_1, x_2, \ldots, y_1, y_2, \ldots), \ldots$
- If second kind do not exist, then we can compositionally abstract

Compositional Abstraction



Polynomials that are used as “seeds” for abstraction of HA-1:

- Other modules need: $p(x_1, x_2, \ldots), \ldots$, and
- Anything this module needs: $r(x_1, x_2, \ldots), \ldots$,

Since HA-1 knows about $\dot{x}_i$, HA-1 can be abstracted.
We also assume that expressions on $y_1, y_2, \ldots$ are used only in the guards in HA-1.

Nonlinear Dynamical Systems

Let $\dot{x}_i = F_i(x_1, \ldots, x_n)$, for $i = 1, \ldots, n$.

Let $(p_1, \ldots, p_k)$ be a syzygy of $q_1, \ldots, q_k$.

Let G be s.t. $\partial G/\partial x_i = p_i$ for all $i = 1, 2, \ldots, n$.

Then G is an invariant of the (nonlinear) system.

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
  - time-automata

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:

$$
\begin{aligned}
q = \mathit{off}\ \ and\ x \le 70 &\quad\longrightarrow\quad q' = on \\
q = on\ \ and\ x \ge 80 &\quad\longrightarrow\quad q' = \mathit{off}
\end{aligned}
$$

- Continuous Flow:

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

Abstraction Algorithm

Consists of three steps:

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

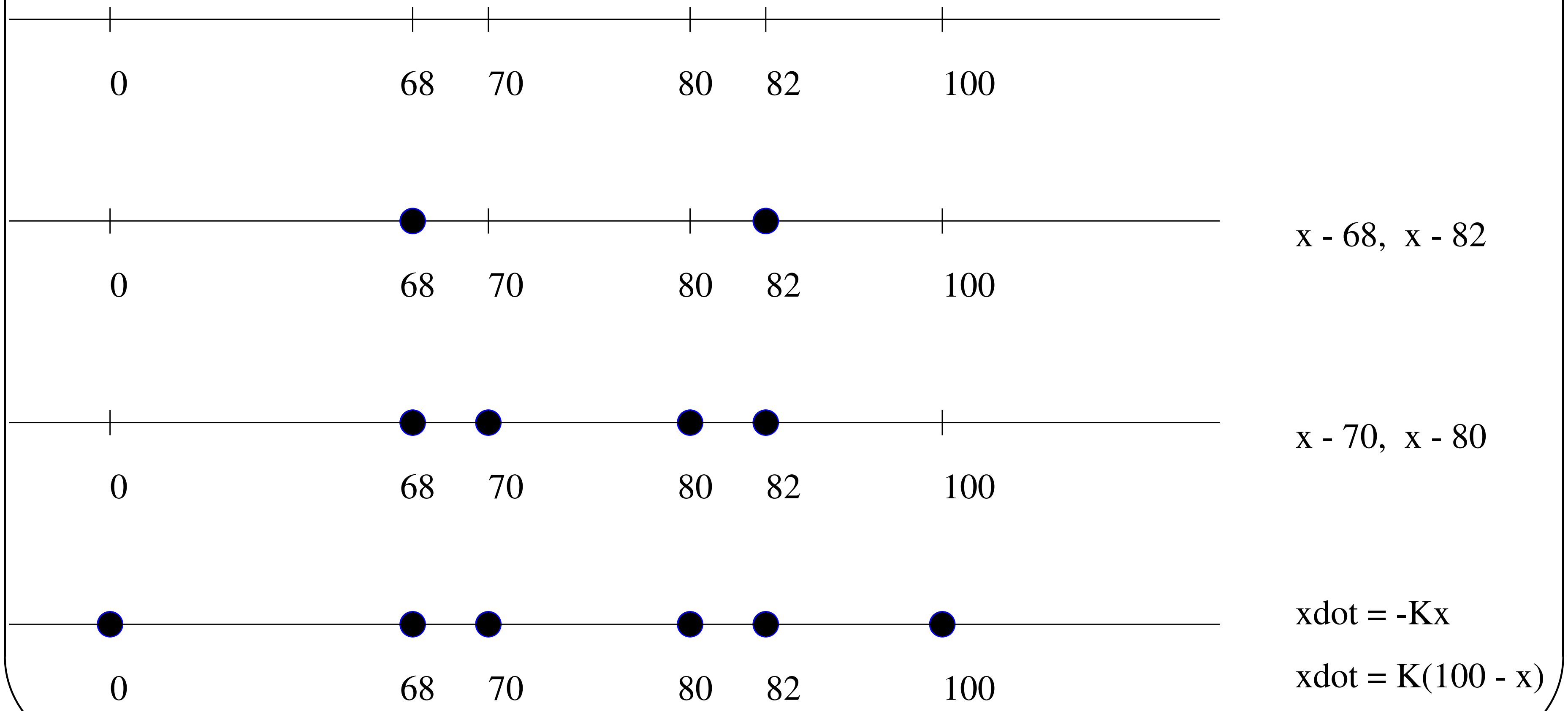
$$P_0 = \{x - 68, x - 82\}$$

- Saturate:

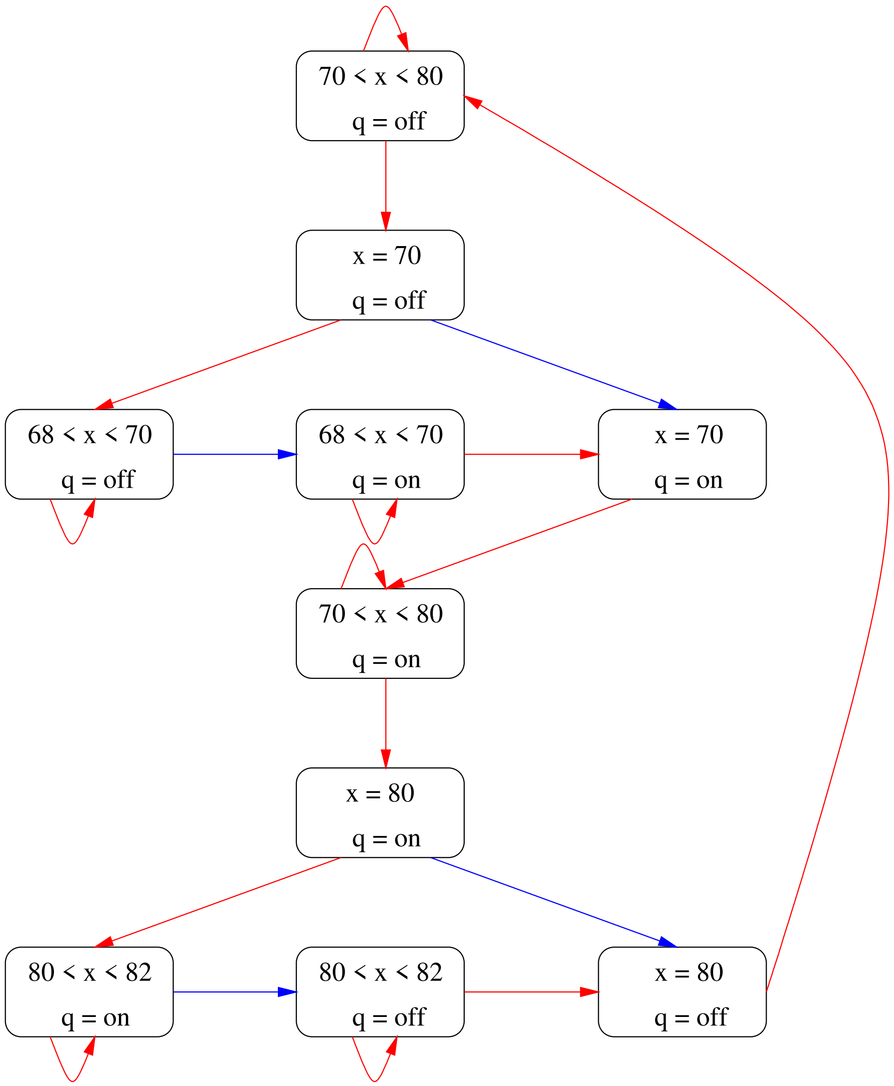
$$\begin{aligned} P_1 &= \{x - 68, x - 82, x - 70, x - 80\} \\ P_2 &= P_1 \cup \{-Kx, K(h - x)\} \\ P_3 &= P_1 \cup \{x, h - x\} \end{aligned}$$

- Abstract

Generating New Polynomials



Abstract Thermostat System



Other Examples

- Cruise control examples [PuriVaraiya94]: 4 variables
- Collision avoidance: 4 variables
- Transmission control: 6 variables
- Delta-Notch Lateral Inhibition: 8 variables
- B.Subtilis Sporulation Initiation: 15 variables

Future examples:

- Multi-modal cruise control
- Full transmission model
- Full stress response of B.Subtilis

Systems Biology

- Tremendous amount of experimental data accumulated over the years,
- Which have been used by biologists to build mental model of biological processes
- But such models have not been formally specified or computationally analyzed
- Goal: Develop models of biological processes and tools to play with the models
- So that wet lab experiments can be replaced by faster and less risky computational ones

Analysis is not the only challenge

Genetic Regulatory 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 of genes
- Which are activated or inhibited by protein concentrations

Traditionally, biologists model the whole network as a continuous dynamical system using the Law of Mass Action

Hybrid Models of Regulatory Pathways

Law of mass action: Rate of a reaction is proportional to the concentrations of its reactants

It breaks down for molecular species that occur in low numbers (master equation)

Genetic regulatory network: There are species present in high numbers and species present in low numbers

Thus, we get a hybrid model

Note: Boolean models are weak, the cont. dyn. models are inaccessible

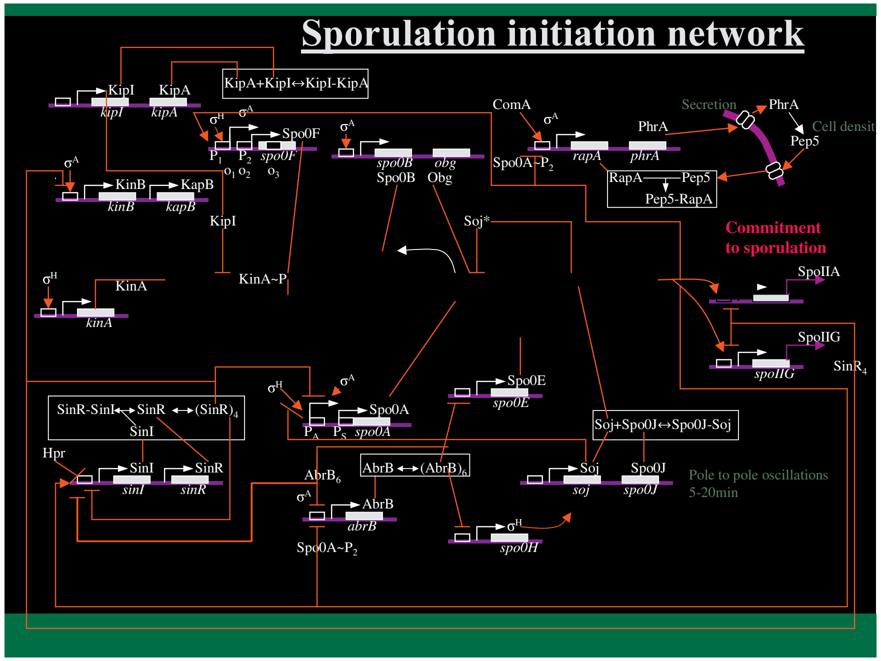
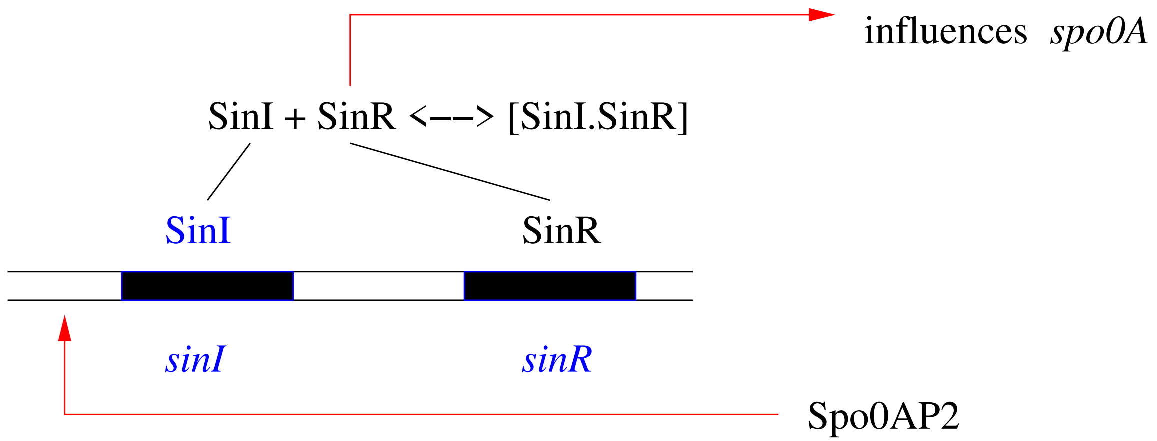


Figure 1: The sporulation inititation network in *B. subtilis*.

The SinIR component

Example of a "component": SinI-SinR global bistable switch



- Tetrameric SinR represses *spo0A*
- SinI inactivates SinR repression by binding to it
- *sinI* expression is controlled by Spo0A-P, SinR, Hpr, and AbrB

Hybrid Models of Metabolic Pathways

- The dynamics of protein concentrations are modeled using differential equations

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

- Genes are present in small numbers: 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

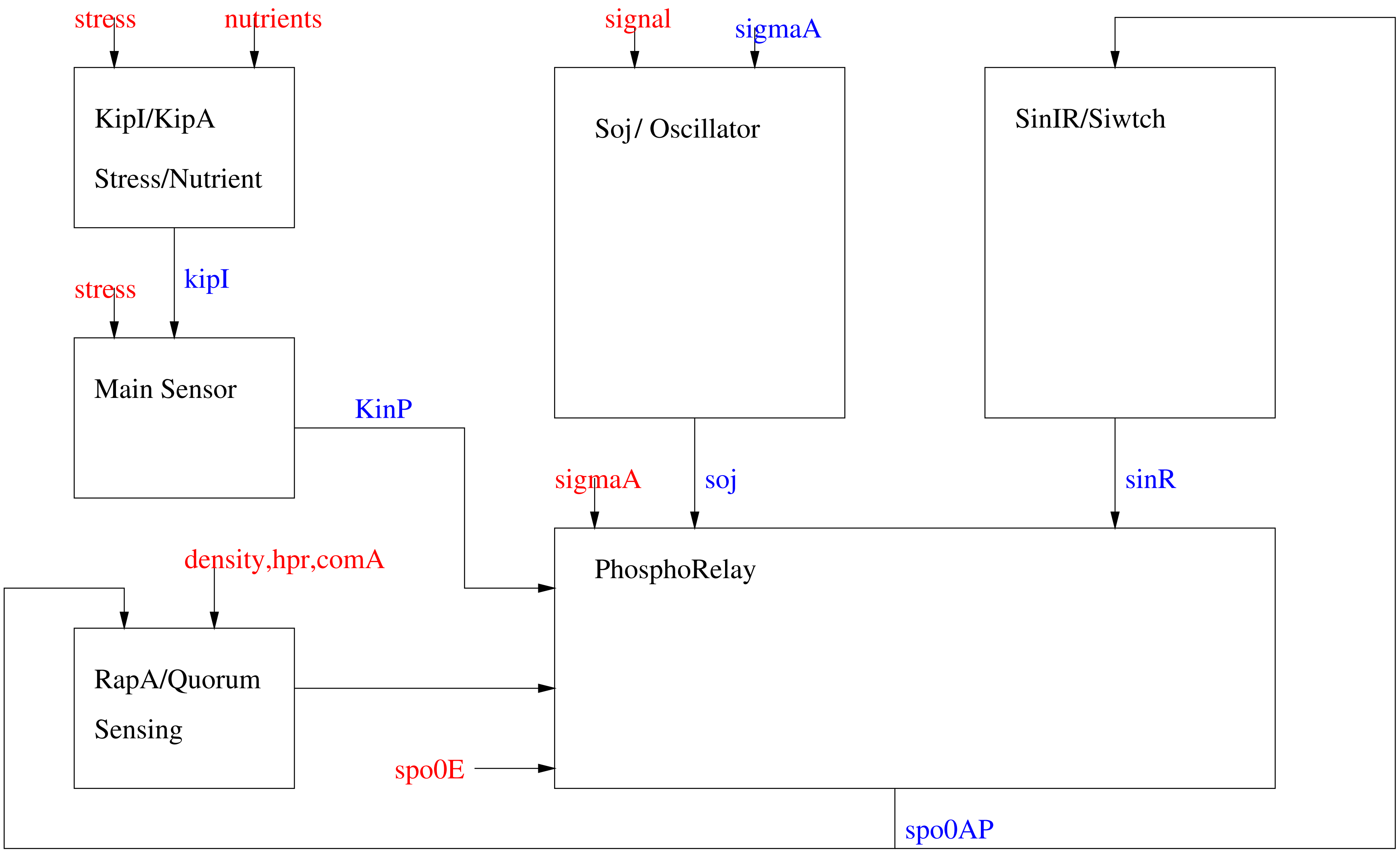
$$k1 = ?, k2 = ?$$

Two Component Systems

Most bacteria/living cells exhibit a sensor/regulator two-component system:

- Transmembrance sensor kinase protein: an environmental change causes a conformational change in it.
- Cytoplasmic regulator protein: The kinase activity phosphorylates the regulator, which promotes transcription of new products from a specific site on the genome.
- Basic control mechanisms: provide phosphate, or strip phosphate
- Examples: osmoregulation, chemotaxis, sporulation.
- Sporulation is more complex than most others.

In The New World...



Challenges

- Most rate constants and parameters are unknown
- But experimental observations related to stability available
- Challenge: Generate constraints on the unknown parameters s.t. resulting model exhibits observed behaviors
- Challenge: Model simplification:

  Boolean Abstraction   Original Hybrid Model

  Replace species that do not change at relevant time-scales by their boolean abstraction

Generating Constraints

Stable region is typically given by specifying which proteins are "high" and which "low":

$$lb_1 \leq x_1 \leq ub_1, \; lb_2 \leq x_2 \leq ub_2, \; \ldots, \; lb_n \leq x_n \leq ub_n$$

A sufficient condition to guarantee this rectangular polytope is stable is that the flow vector points inwards at each face

These constraints lead to validity of certain formulas

Use quantifier elimination to generate constraints

Other Related Work...

- Decision procedure for reals
  - "Logical" decision procedure
  - Abstraction based theorem proving: *Theorem provers should abstract*
  - Sound, but incomplete hierarchies of decision procedures based on *abstraction, refinement*
  - Maybe complete too (positivstellensatz)
- Systems Theory for effective abstractions.
  - Approximating reach sets using structural analysis of the differential equations:
  - linear algebra, algebraic and differential geometry

Other Related Work: Contd...

- Switched Systems
  - exponentially succinct “inside-out” representation
  - can be exploited for efficiently constructing abstractions
- Composition
  - fine grained exploitation at the level of modes
  - continuous state space partitioned differently in different modes
  - resets important now (timed, rectangular automata)
- Model reduction
  - Simplify model by possibly introducing “new” modes
- Model refinement
  - Generate constraints on parameters...

Summary

- Completely automatic procedure for hybrid automata specified only using polynomials: uses a 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