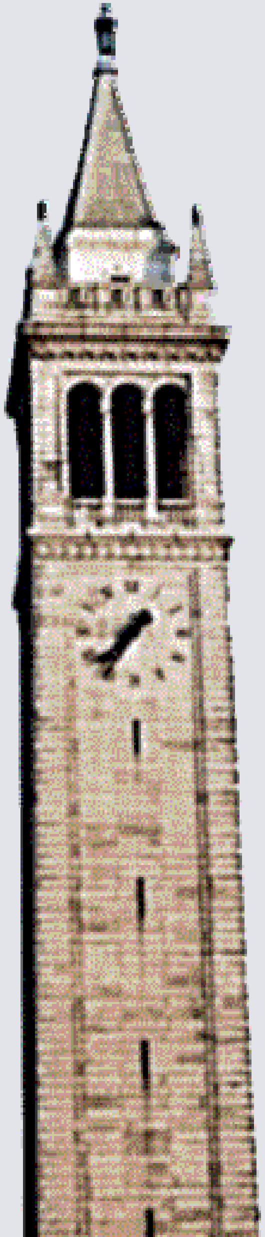
Advances in Hybrid System Theory: Overview

Edited and presented by

Claire J. Tomlin

UC Berkeley

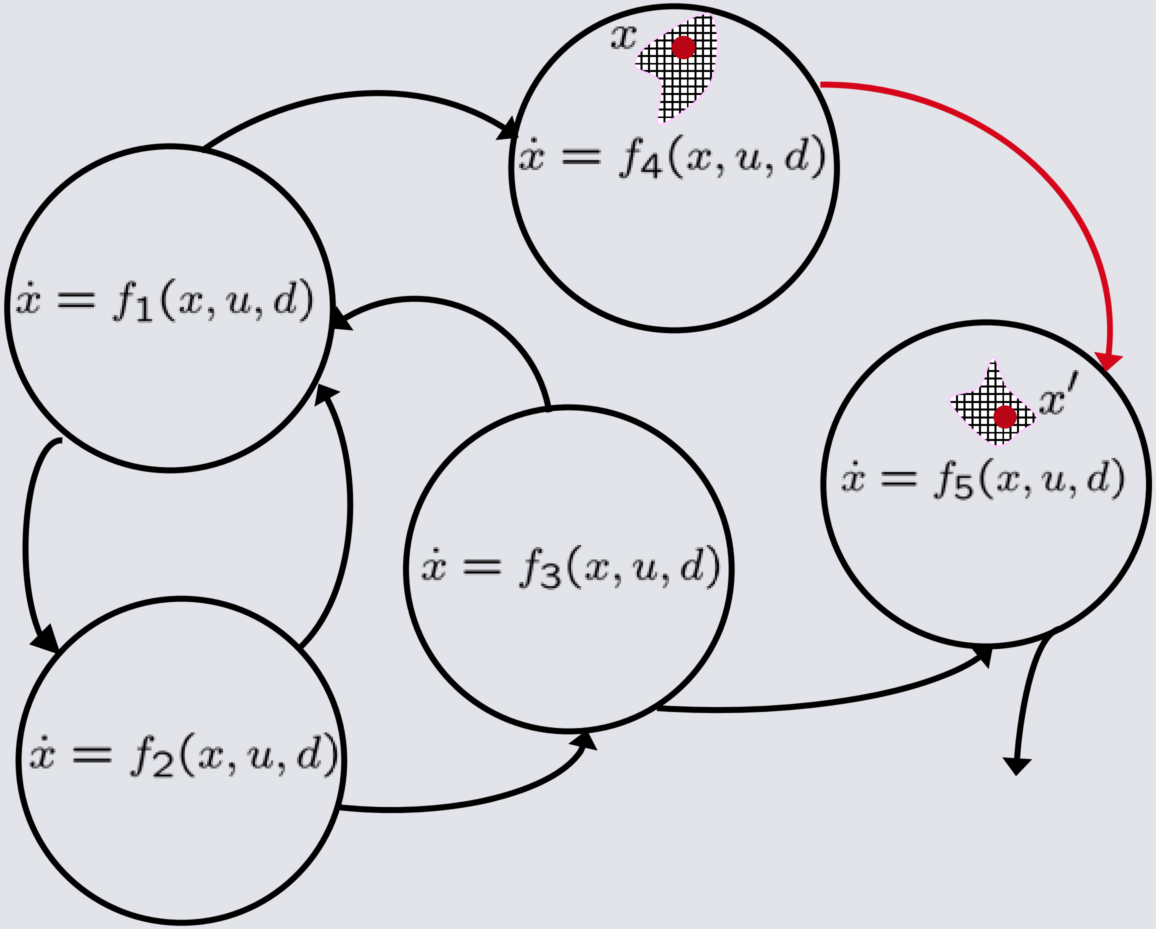
Chess Review
November 21, 2005
Berkeley, CA



Thrust I: Hybrid System Theory

- Models and semantics
  - Abstract semantics for Interchange Format
  - Hybrid Category Theory
- Analysis and verification
  - Detecting Zeno
  - Automated abstraction and refinement
    - Fast numerical algorithm
    - Symbolic algorithm
- Control
  - Stochastic games
  - Optimal control of stochastic hybrid systems

Hybrid System Model: Basics



Interchange format for HS: Abstract Semantics (Model)

Definition: A HS is a tuple $H = (V, E, \mathcal{D}, I, \sigma, \omega, \rho)$

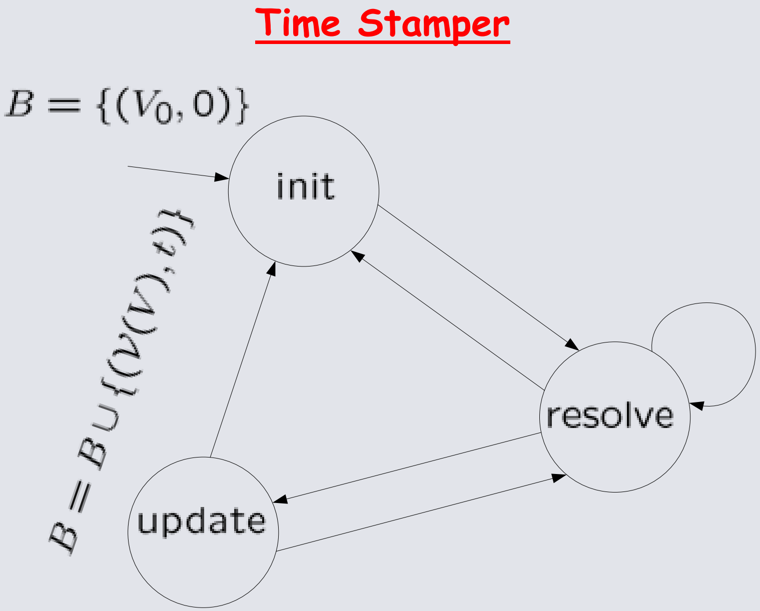
- $V = \{v_1, ..., v_n\}$ is a set of variables
- $E = \{e_1, ..., e_m\}$ is a set of equations
- $\mathcal{D} \subseteq 2^{\mathcal{R}(V)}$ is a set of domains
- $I \subseteq \mathbb{N}$ is a set of indexes
- $\sigma : 2^{\mathcal{R}(V)} \to 2^I$ associates a set of indexes to each domain
- $\omega : I \to 2^E$ associates a set of equations to each index
- $\rho : 2^{\mathcal{R}(V)} \times 2^{\mathcal{R}(V)} \times \mathcal{R}(V) \to 2^{\mathcal{R}(V)}$ is the reset mapping
- Composition defined

[Pinto, Sangiovanni-Vincentelli]

Interchange format for HS: Abstract Semantics (Execution)

The semantics is defined by the set B of pairs (γ, t) of valuations and time stamps.

The set B is determined by the following elements: $(H, T, \mathtt{resolve}, \mathtt{init}, \mathtt{update})$



```
resolve(t)
$\mathcal{D}' \Leftarrow \{D \in \mathcal{D} | val(V_t) \in D\}$ //Active domains
$I \Leftarrow \emptyset$, $E_t \Leftarrow \emptyset$
$I \Leftarrow \cup_{D \in \mathcal{D}'} \sigma(D)$ //Active dynamics
for all $i \in I$ do
    $E_t = E_t \cup \omega(i)$ //Active equations
end for
sort($E_t$, $\pi$) //Order the equations
for all $e_i \in E_t$ do
    solve($e_i$,$t$)
end for
$\mathcal{D}'' \Leftarrow \{D \in \mathcal{D} | val(V_t) \in D\}$ //Active domains*
markchange ( $D'$,$D''$ ) //Domain change
```

[Pinto, Sangiovanni-Vincentelli]

Hybrid Category Theory

- Reformulates hybrid systems categorically so that they can be more easily reasoned about
- Unifies, but clearly separates, the discrete and continuous components of a hybrid system
- Arbitrary non-hybrid objects can be generalized to a hybrid setting
- Novel results can be established

[**Ames**, Sastry]

Hybrid Category Theory: Framework

- One begins with:
  - A collection of “non-hybrid” mathematical objects
  - A notion of how these objects are related to one another (morphisms between the objects)
    - Example: vector spaces, manifolds, dynamical systems
- Therefore, the non-hybrid objects of interest form a category, T
    - Example: T = Vect; T = Man; T = Dyn;
- The objects being considered can be “hybridized” by considering a small category (or “graph”) H together with a functor (or “function”):

$$\boxed{S: H\ !\quad T}$$

- H is the “discrete” component of the hybrid system
- T is the “continuous” component
    - Example: hybrid vector space S:H ! Vect; hybrid manifold S:H ! Man; hybrid system S:H ! Dyn

[Ames, Sastry]

Hybrid Category Theory: Properties

- **Composition:** hybrid category theory can be used to reason about heterogeneous system composition:
  - Prove that composition is the limit of a hybrid object over this category

$$\mathscr{P}_1 \|_{\mathscr{M}} \mathscr{P}_2 = \varprojlim \left(\mathscr{P}_1 \xrightarrow{\alpha_1} \mathscr{M} \xleftarrow{\alpha_2} \mathscr{P}_2 \right)$$

  - Derive necessary and sufficient conditions on when behavior is preserved by composition
- **Reduction:** can be used to decrease the dimensionality of systems; a variety of mathematical objects needed (vector spaces, manifolds, maps), hybrid category theory allows easy "hybridization" of these.

[Ames, Sastry]

Hybrid Reduction Theorem

Classical Reduction Theorem

- *Given a symplectic manifold M (the phase space), there exists a symplectic manifold M_μ such that M_μ inherits the symplectic structure from that of M.*
- *Dynamical trajectories of the Hamiltonian H on M determine corresponding trajectories on the reduced space.*

Hybrid Reduction Theorem

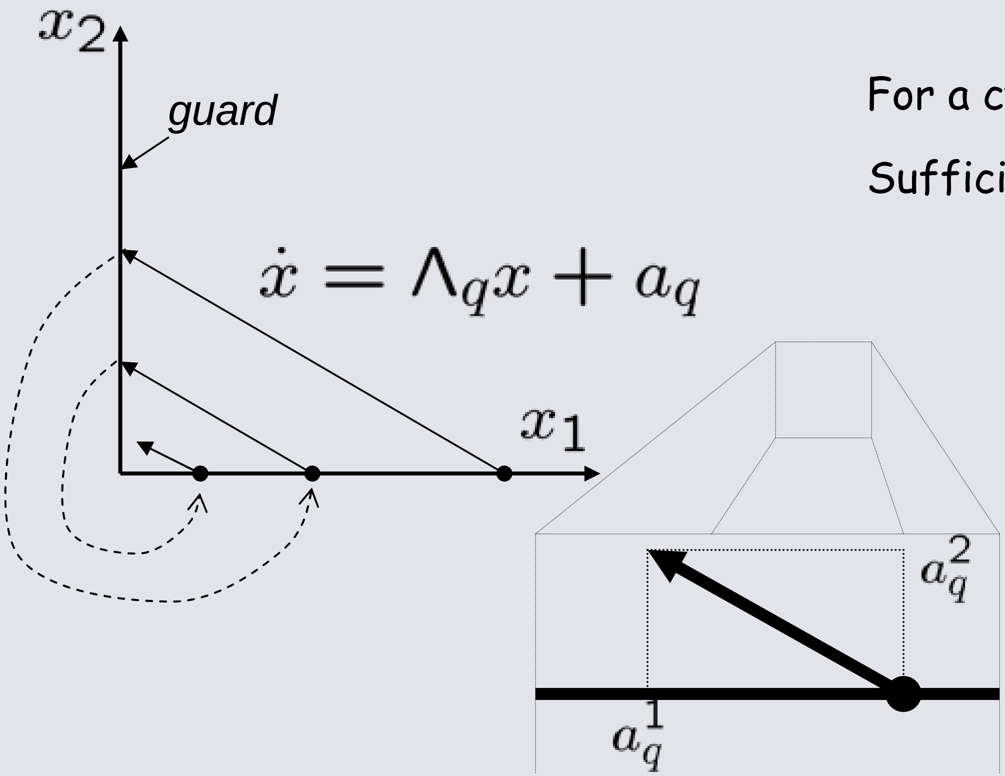
- *Given a hybrid symplectic manifold $\mathbf{M}$ (the hybrid phase space), there exists a hybrid symplectic manifold $\mathbf{M}_\mu$ such that $\mathbf{M}_\mu$ inherits the hybrid symplectic structure from that of $\mathbf{M}$.*
- *Dynamical hybrid trajectories of the hybrid Hamiltonian $\boldsymbol{H}$ on $\mathbf{M}$ determine corresponding hybrid trajectories on the reduced hybrid space.*

[Ames, Sastry]

Other results: detecting zeno

- **Zeno:** hybrid trajectory switches infinitely often in a finite amount of time
- Detection of Zeno is critical in control design
- Progress in identification of *Sufficient Conditions* for detection

Diagonal, "First Quadrant" HS



$$\dot{x} = \Lambda_q x + a_q$$

For a cycle $1 \to 2 \ldots \to K \to 1 \ldots$

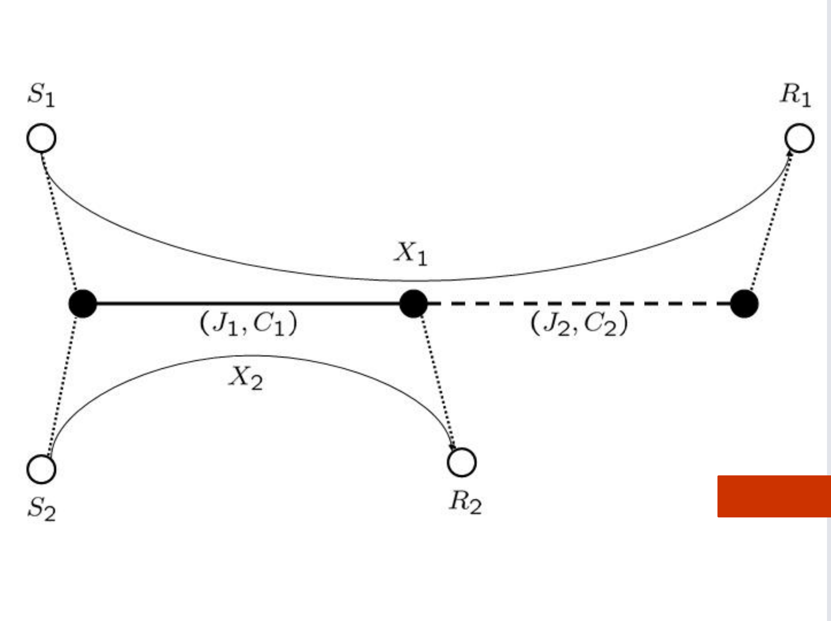
Sufficient Conditions: for all $q \in \{1, 2 \ldots, K\}$

$$\left.\begin{array}{c} \lambda_q^1 \leq 0 \\ \\ a_q^1 < 0 < a_q^2 \\ \\ \left|\prod_{q=0}^{K} \frac{a_q^2}{a_q^1}\right| < 1 \end{array}\right\}$$

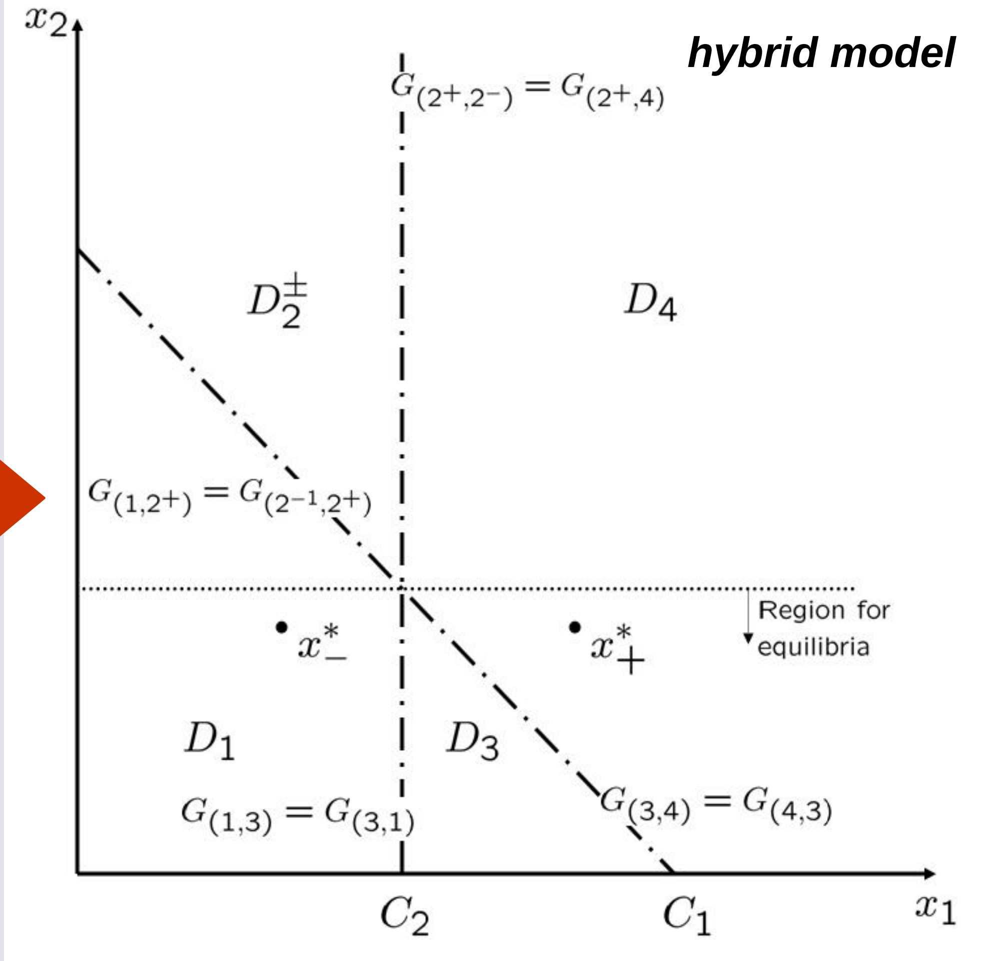
Genuine Zeno Behavior

[Abate, Ames, Sastry]

Zeno: a TCP control example



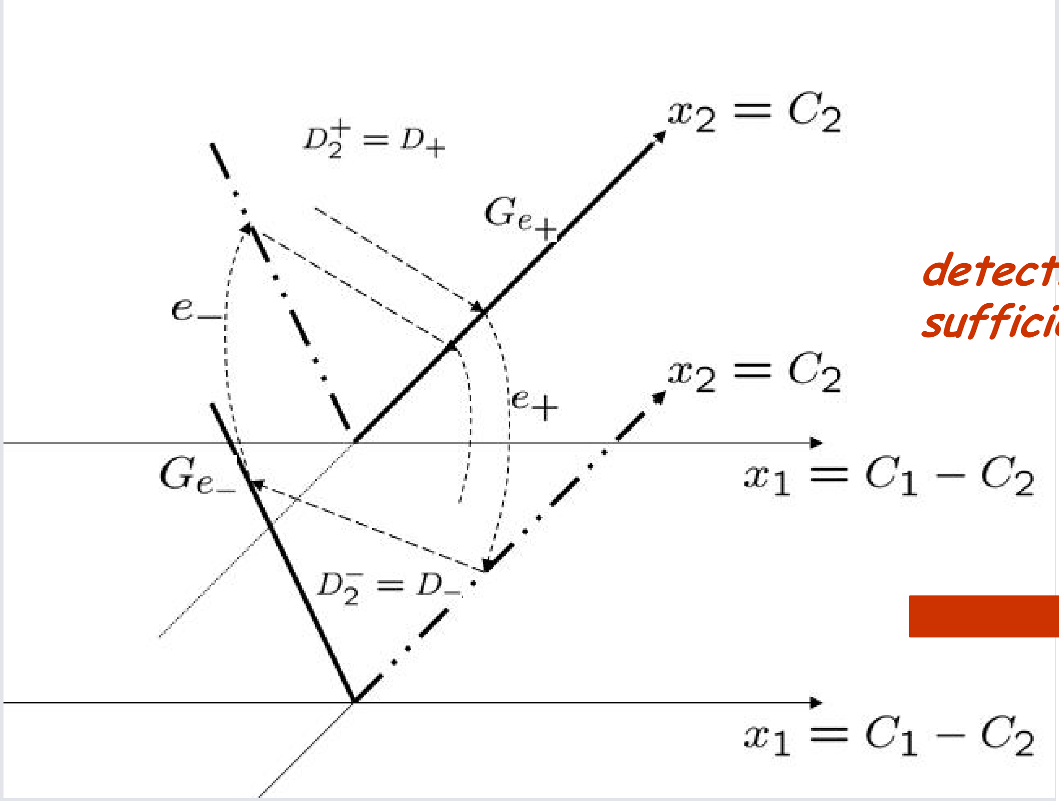
Topology of a 2-user, 2 links (one wireline, one wireless) network



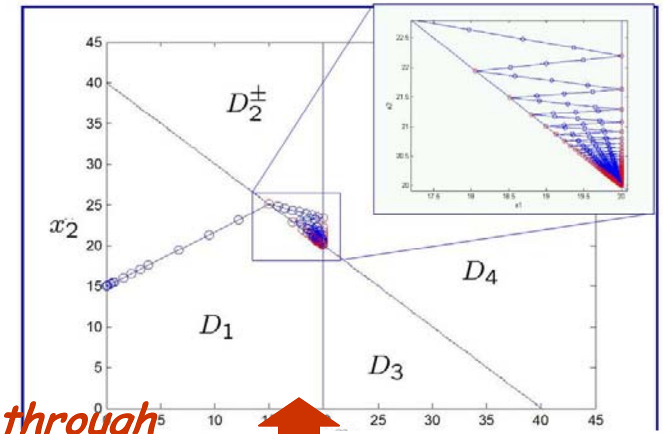
[Abate, Ames, Sastry]

Zeno: a TCP control example

study of a cycle
reduction in first quadrant form

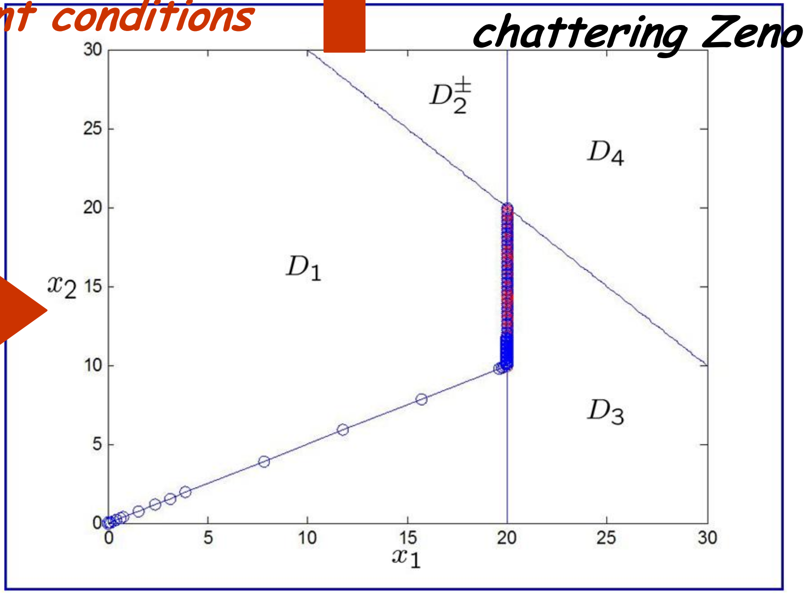


genuine Zeno



detection through sufficient conditions

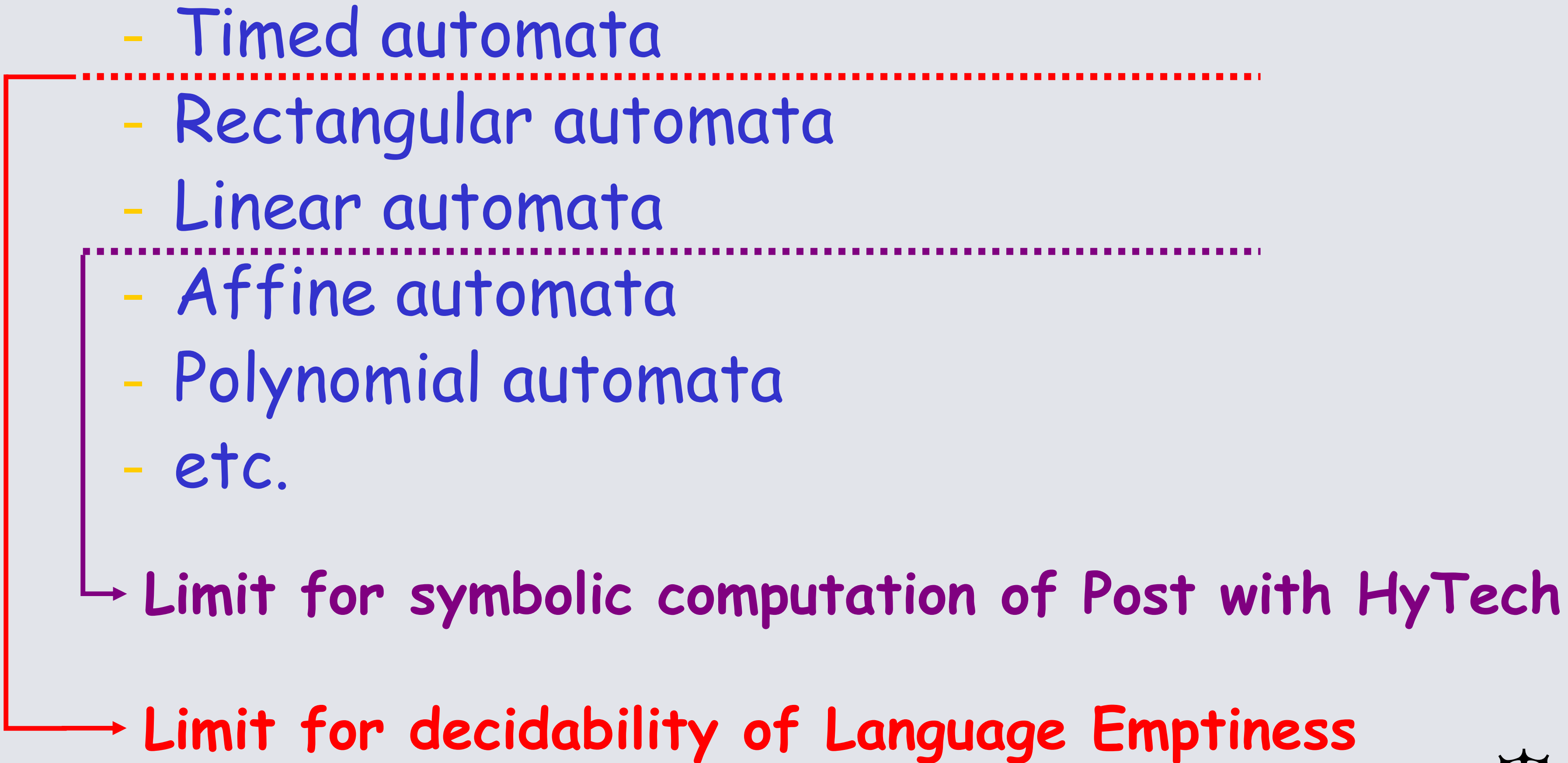
chattering Zeno



[**Abate**, Ames, Sastry]

Reminder

Some classes of hybrid automata:



- Timed automata
- Rectangular automata
- Linear automata
- Affine automata
- Polynomial automata
- etc.

Limit for symbolic computation of Post with HyTech

Limit for decidability of Language Emptiness

[Doyen, Henzinger, Raskin]

Methodology

- Affine automaton A and set of states Bad
- Check that Reach(A) ∩ Bad = ∅
- Affine dynamics is too complex ?

➡ Abstract it automatically !

- Abstraction is too coarse ?

➡ Refine it automatically !

[Doyen, Henzinger, Raskin]

Methodology

1. Abstraction: over-approximation

Affine dynamics → Rectangular dynamics

$$\begin{array}{c} \dot{x} = 2 - x \\ 0 \le x \le 3 \end{array} \quad \rightarrow \quad \begin{array}{c} \dot{x} \in [-1, 2] \\ 0 \le x \le 3 \end{array}$$

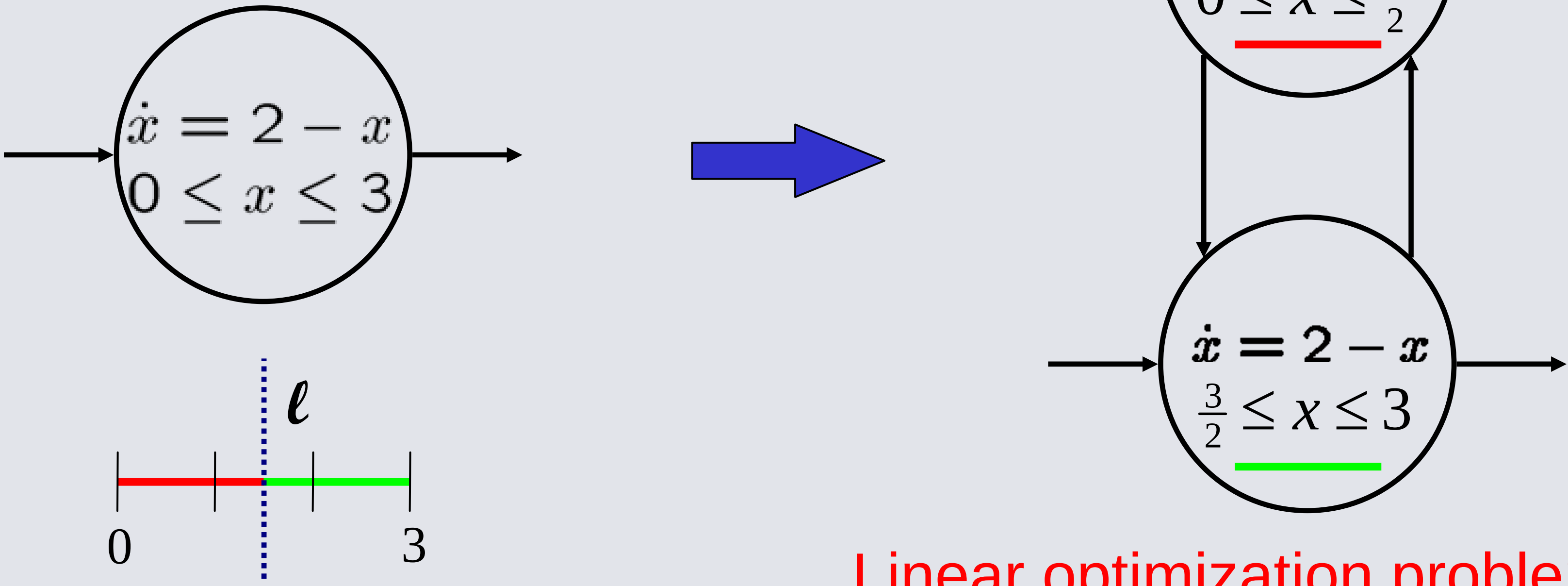
Let $\begin{cases} f(x) = 2\text{-}x \\ \text{Inv} = \{0 \le x \le 3\} \end{cases}$ Then $[-1,2] = [\min_{x\in\text{Inv}} f(x), \max_{x\in\text{Inv}} f(x)]$

[Doyen, Henzinger, Raskin]

Methodology

2. Refinement: split locations by a line cut

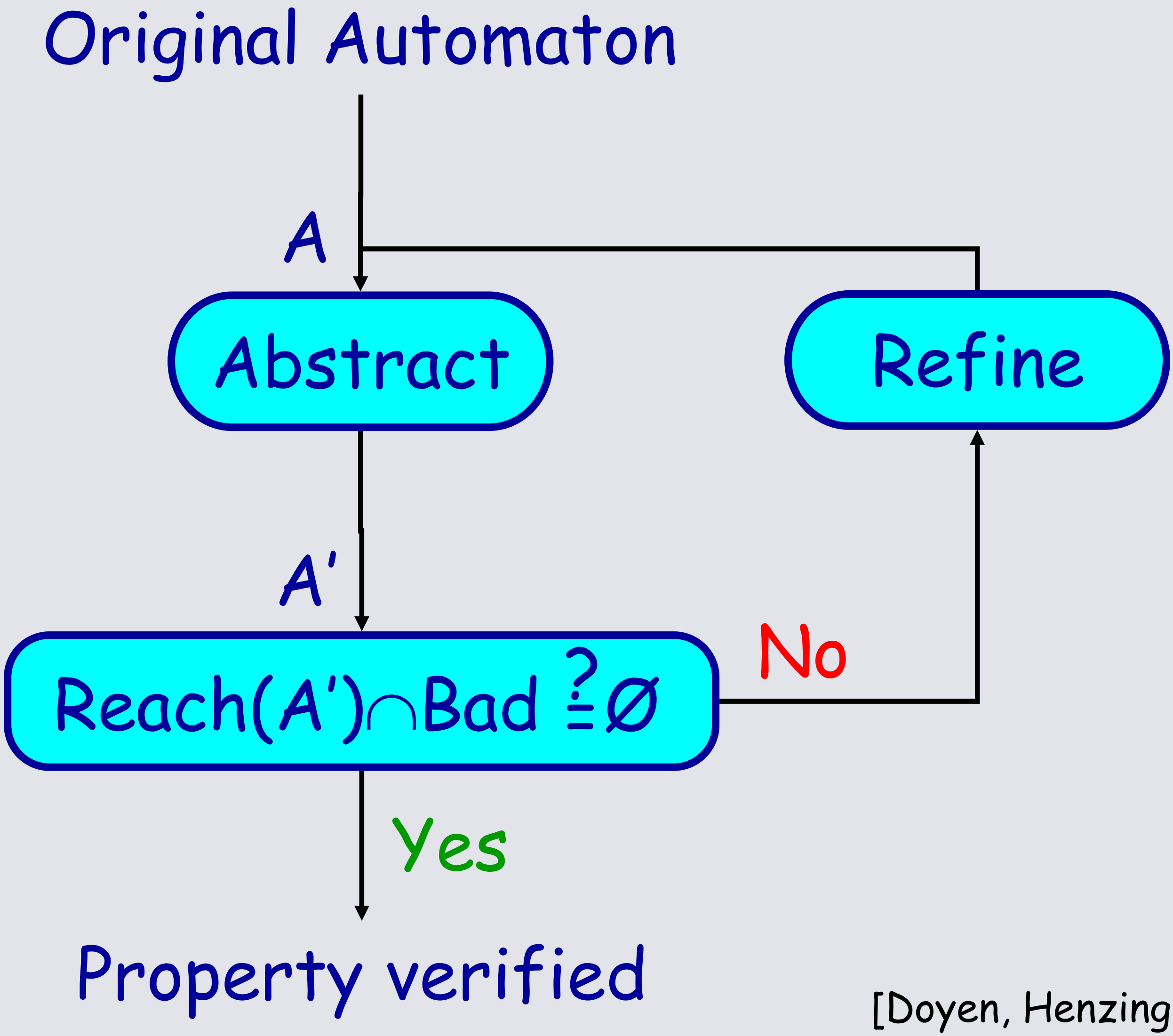
Line l $\equiv x = \frac{3}{2}$



Linear optimization problem !

[Doyen, Henzinger, Raskin]

Methodology



Original Automaton

A

Abstract

Refine

A'

$\mathrm{Reach}(A') \cap \mathrm{Bad} \overset{?}{=} \emptyset$

No

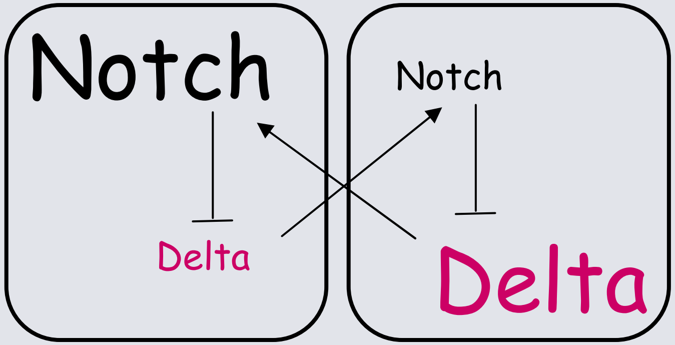
Yes

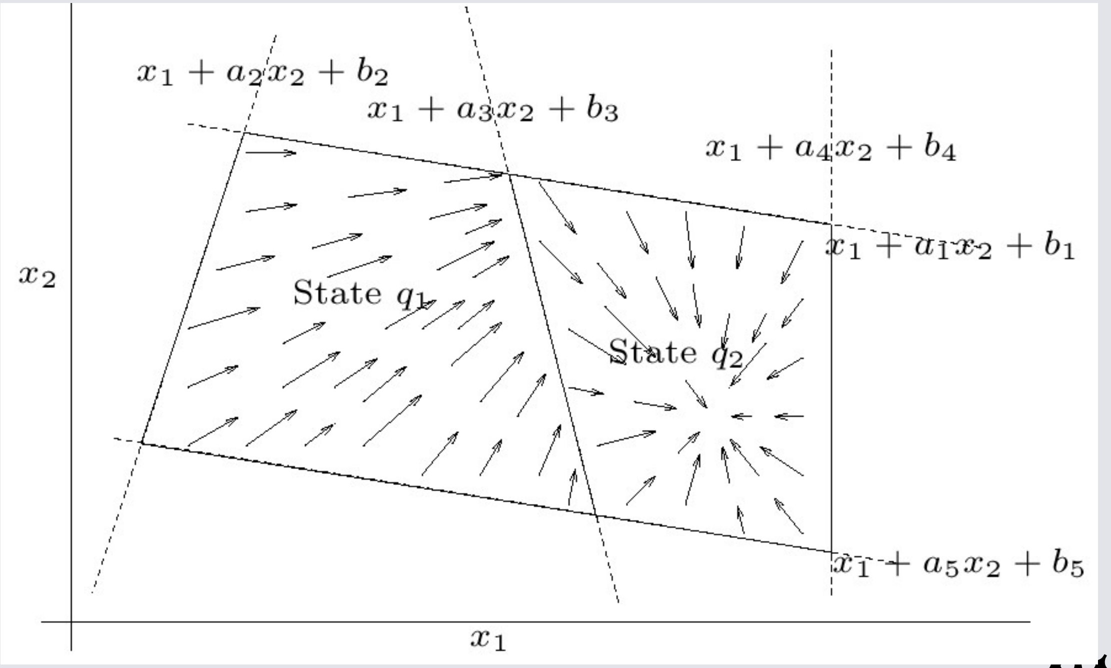
Property verified

[Doyen, Henzinger, Raskin]

Symbolic Reachability Analysis

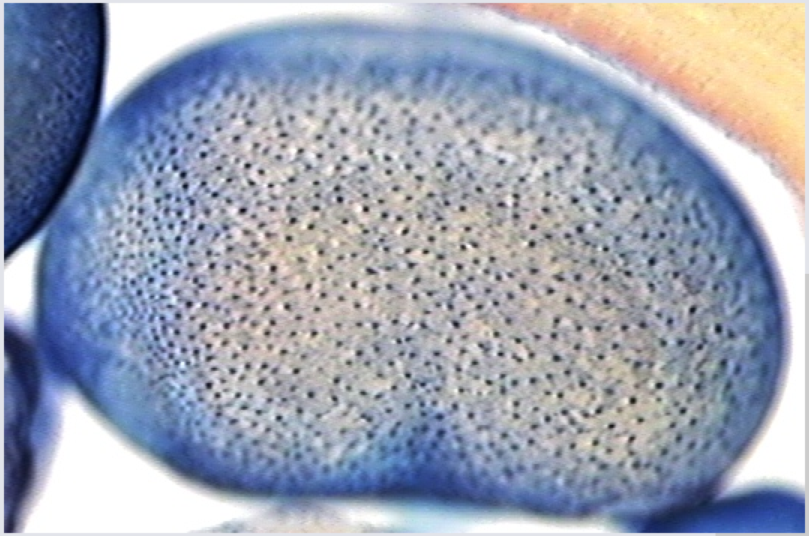
- Want to find initial conditions that converge to a particular steady-state
- Compute reach sets symbolically, in terms of model parameters, from the desired reachable states
- Problem:
  - Large state space
- Solution
  - Abstract!

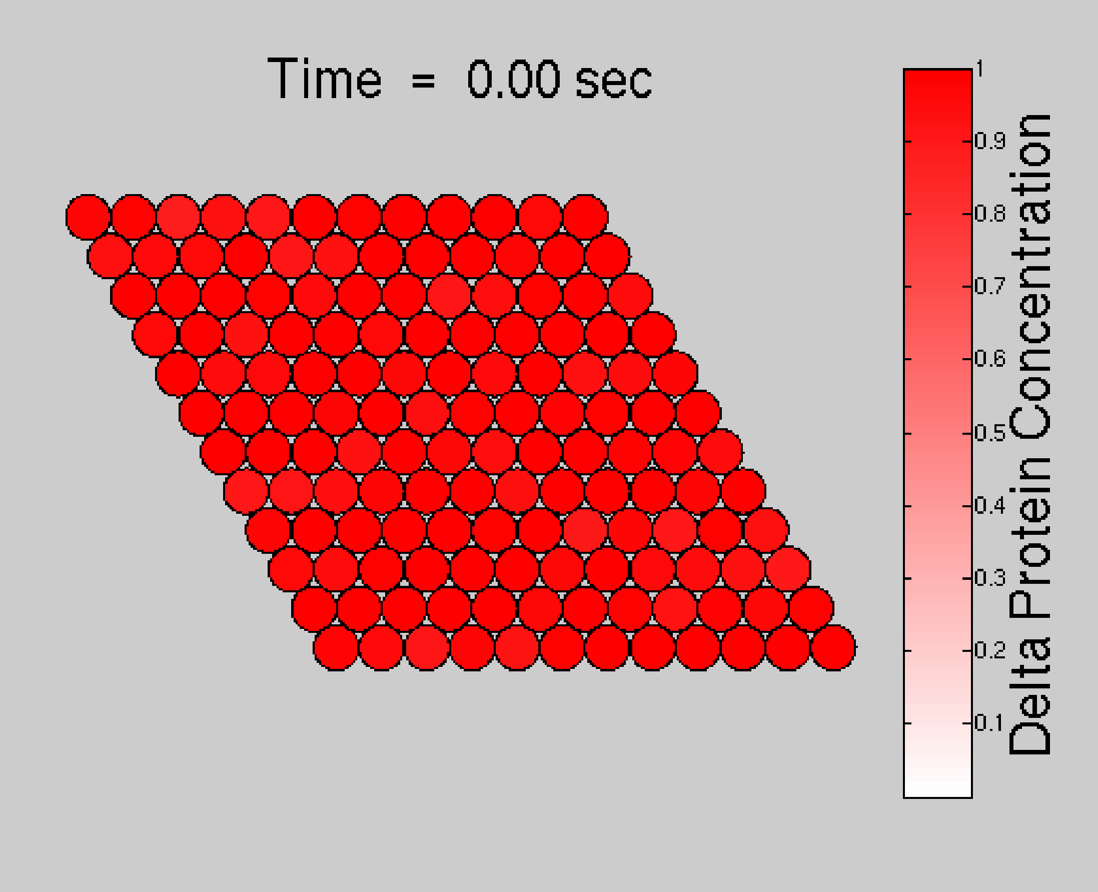




[Ghosh, Tomlin]

Differentiation in *Xenopus*

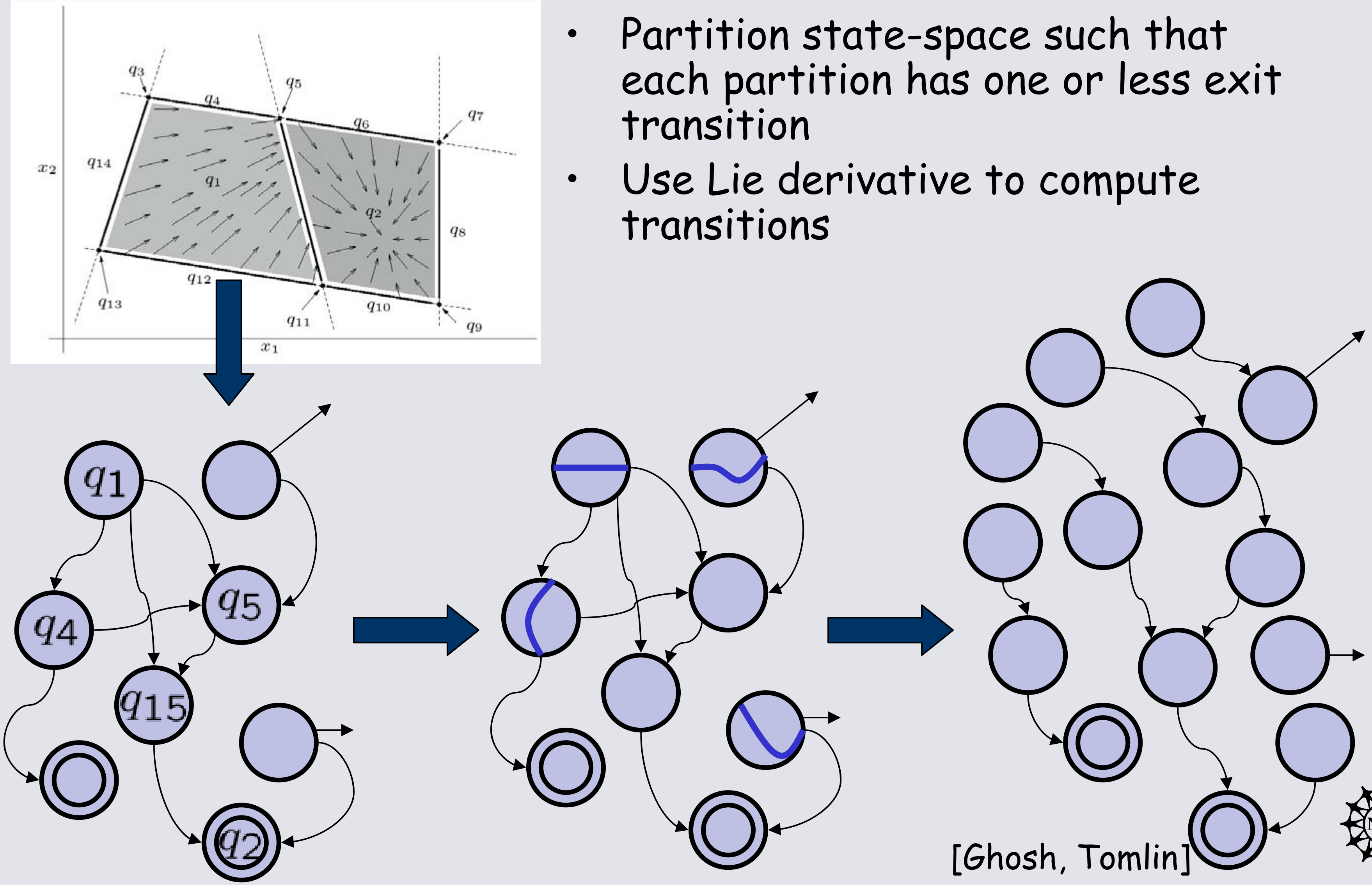




[Ghosh, Tomlin]

Abstraction Algorithm

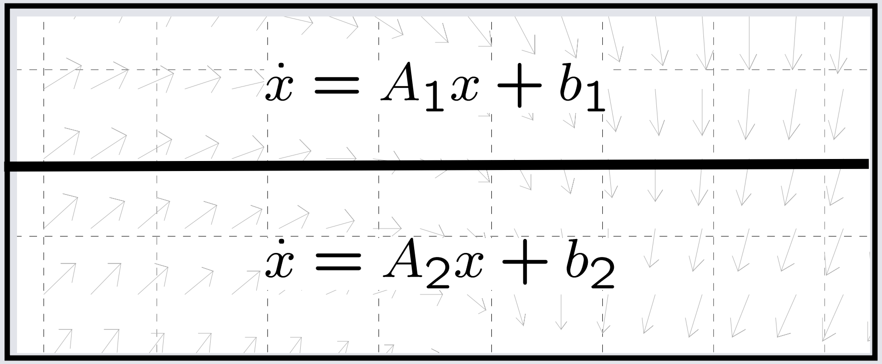
- Partition state-space such that each partition has one or less exit transition
- Use Lie derivative to compute transitions



[Ghosh, Tomlin]

Abstraction Algorithm Step 1

A simple example:

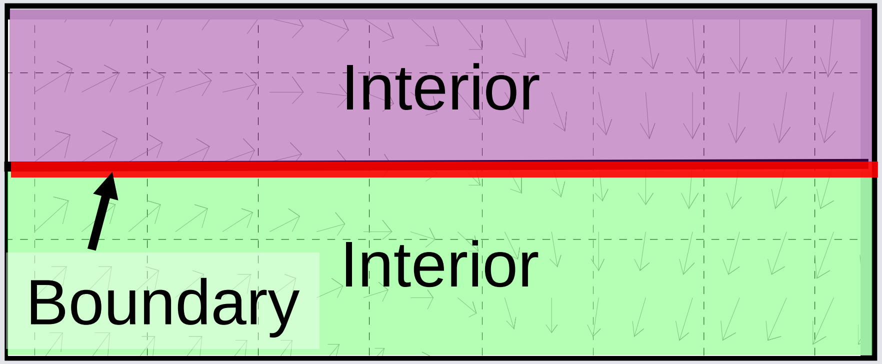


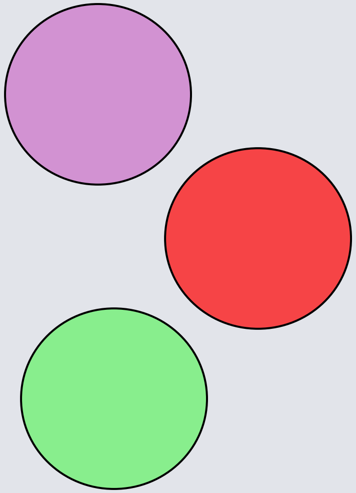
$A_i, b_i, \alpha_i, \beta_i$ are symbolic

A_i diagonal

$x_1 + \alpha_1 x_2 + \beta_1 = 0$

Step 1: Separate partitions into interiors and boundaries



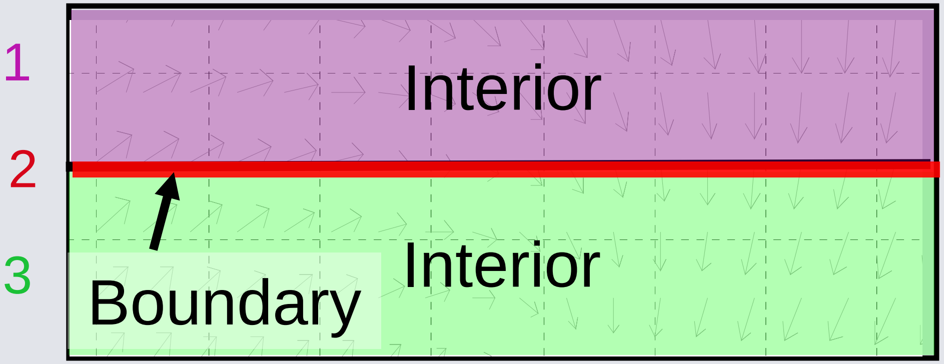


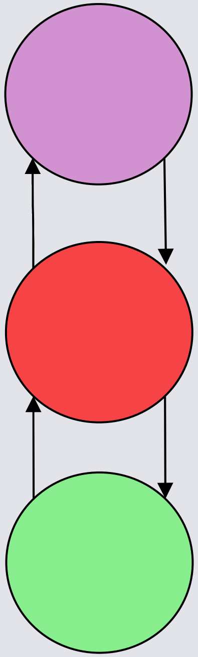
[Ghosh, Tomlin]

Abstraction Algorithm Step 2

Step 2: Compute transitions between modes. In mode 1:

- Determine direction of flow across the boundary
- Compute sign of Lie derivative of function describing boundary, with respect to mode 1 dynamics: $\mathcal{L}_{A_1 x + b_1}(x_1 + \alpha_1 x_2 + \beta_1)$
- If $\mathcal{L} < 0$ then flow is from mode 1 to mode 2
- If $\mathcal{L} = 0$ then flow remains on boundary

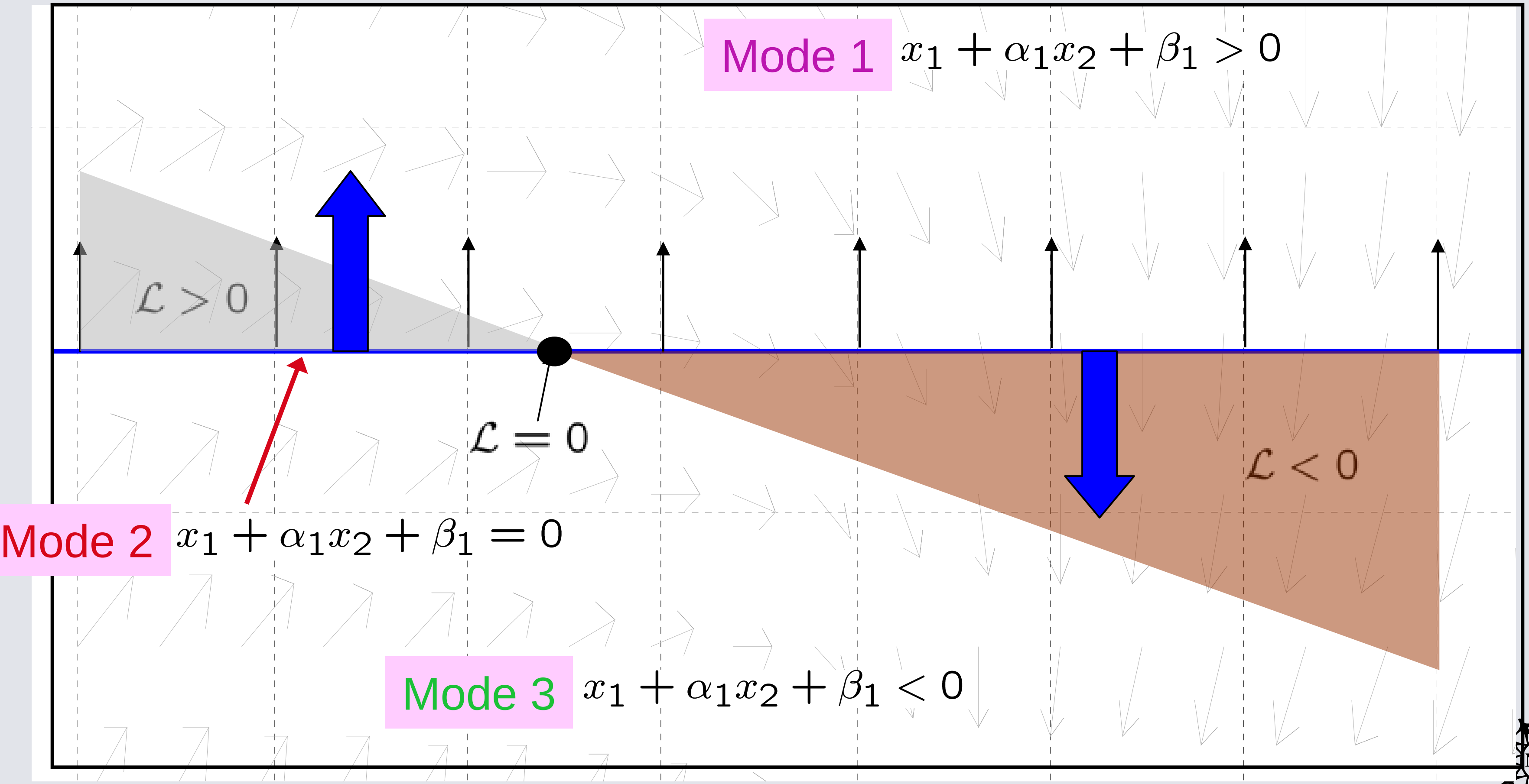




[Ghosh, Tomlin]

Transition Checking: Lie Derivative

$$\mathcal{L}_{A_1x+b_1}(x_1+\alpha_1x_2+\beta_1)=\frac{\delta(x_1+\alpha_1x_2+\beta_1)}{\delta x}(A_1x+b_1)$$

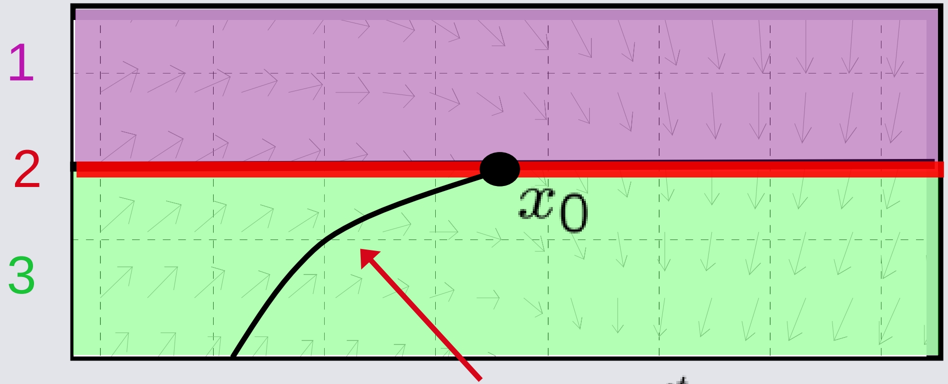

Mode 1 $x_1+\alpha_1x_2+\beta_1>0$
$\mathcal{L}>0$
$\mathcal{L}=0$
$\mathcal{L}<0$
Mode 2 $x_1+\alpha_1x_2+\beta_1=0$
Mode 3 $x_1+\alpha_1x_2+\beta_1<0$

[Ghosh, Tomlin]

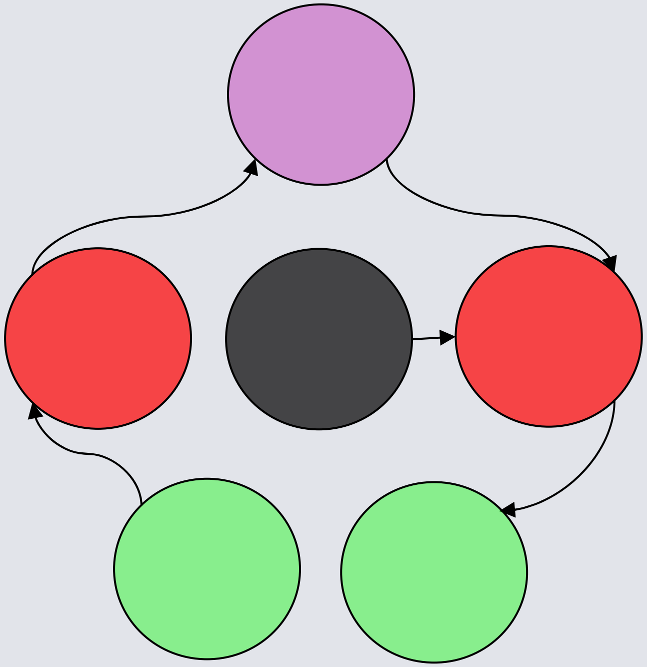
Abstraction Algorithm Step 3

Step 3: Partition modes that have more than one exit transition

- In Mode 2, split the mode at the point of intersection or inflexion, where $\mathcal{L} = 0$
- In Mode 3, partition between those states which remain in 3 and those which enter mode 2. The separation line (or surface) is the analytical solution of the differential equations of the mode passing through the separation point.



$$x = exp(A_2 t)x_0 + \int_0^t exp(A_2 \tau) b_2 d\tau$$



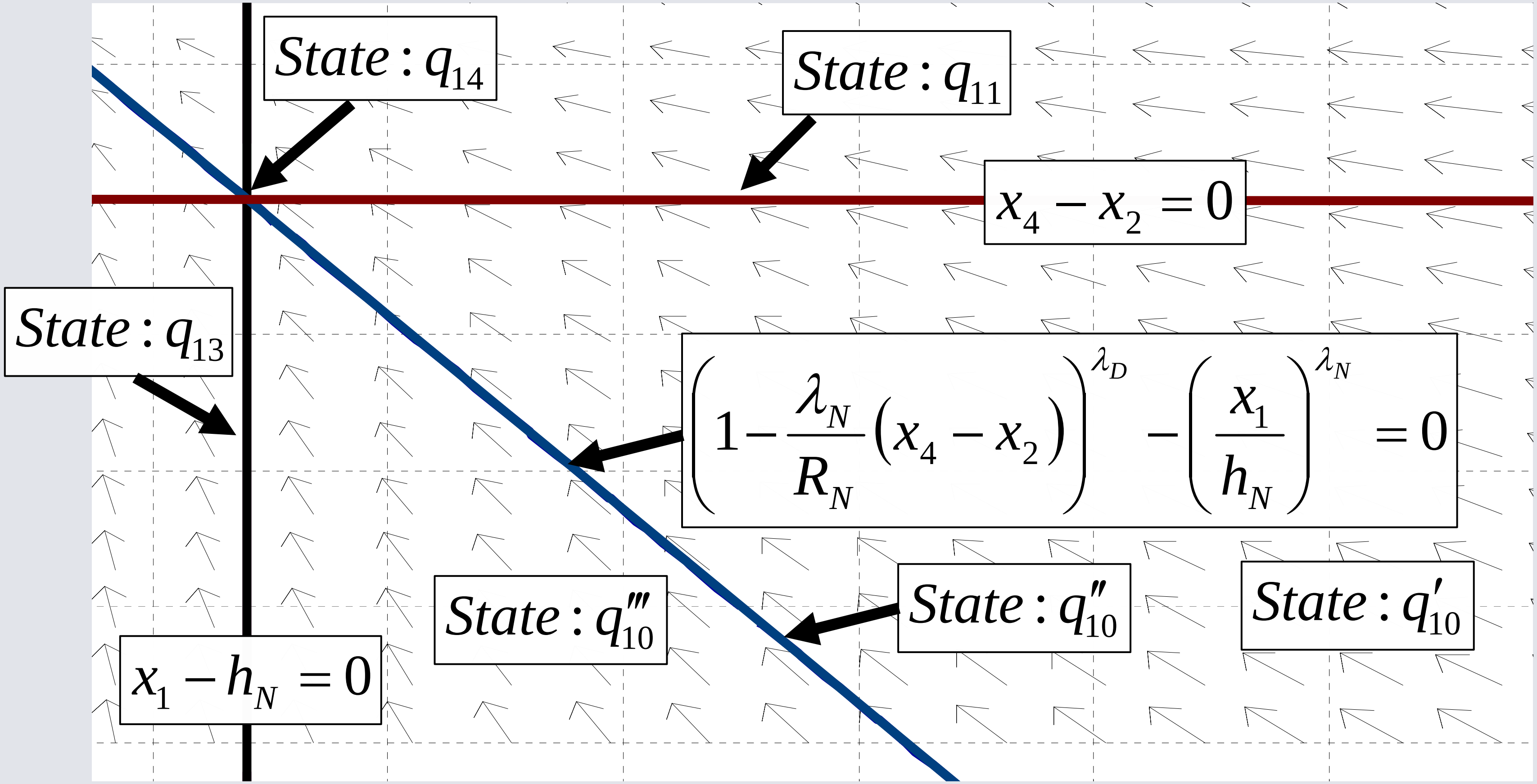
time t is eliminated to form a closed form polynomial expression →

$$x_1^a + x_2^b = c$$
$$x_1 - x_3 = 0$$

[Ghosh, Tomlin]

Illustration: 2 Cell Delta-Notch

- Partitioning step:



$State: q_{14}$

$State: q_{11}$

$x_4 - x_2 = 0$

$State: q_{13}$

$$\left(1-\frac{\lambda_N}{R_N}(x_4-x_2)\right)^{\lambda_D}-\left(\frac{x_1}{h_N}\right)^{\lambda_N}=0$$

$State: q_{10}'''$

$State: q_{10}''$

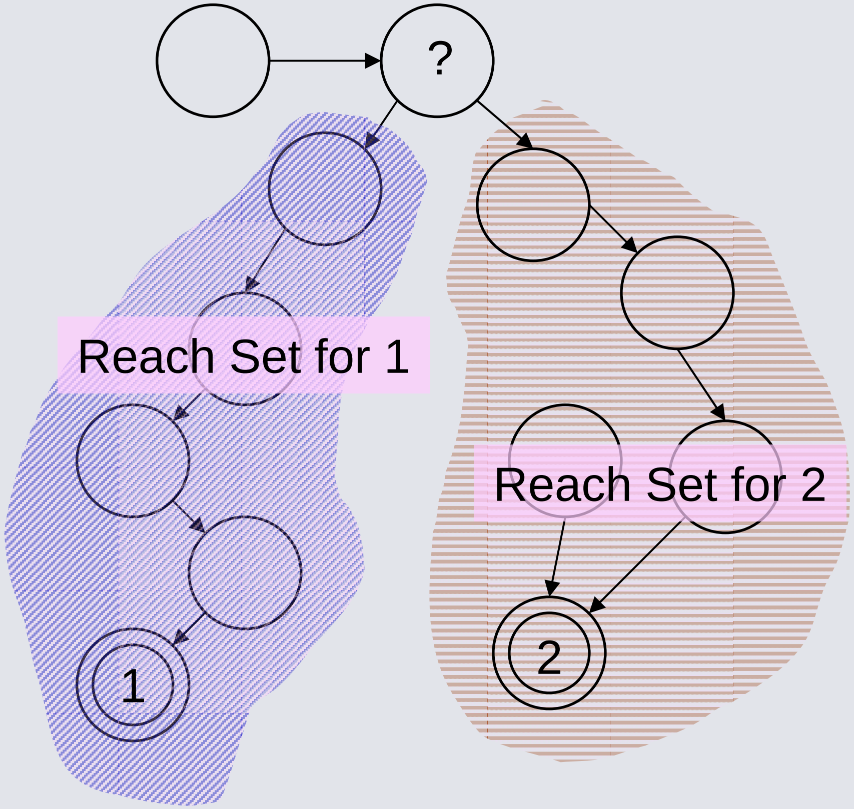
$State: q_{10}'$

$x_1 - h_N = 0$

[Ghosh, Tomlin]

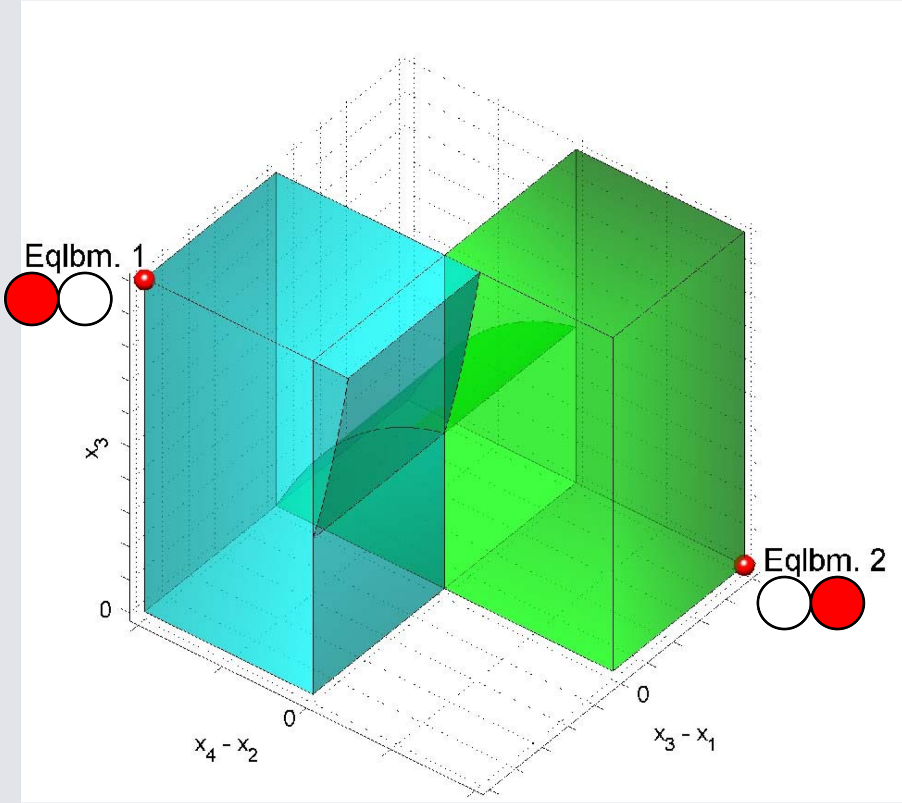
…..And its Results: Reachability

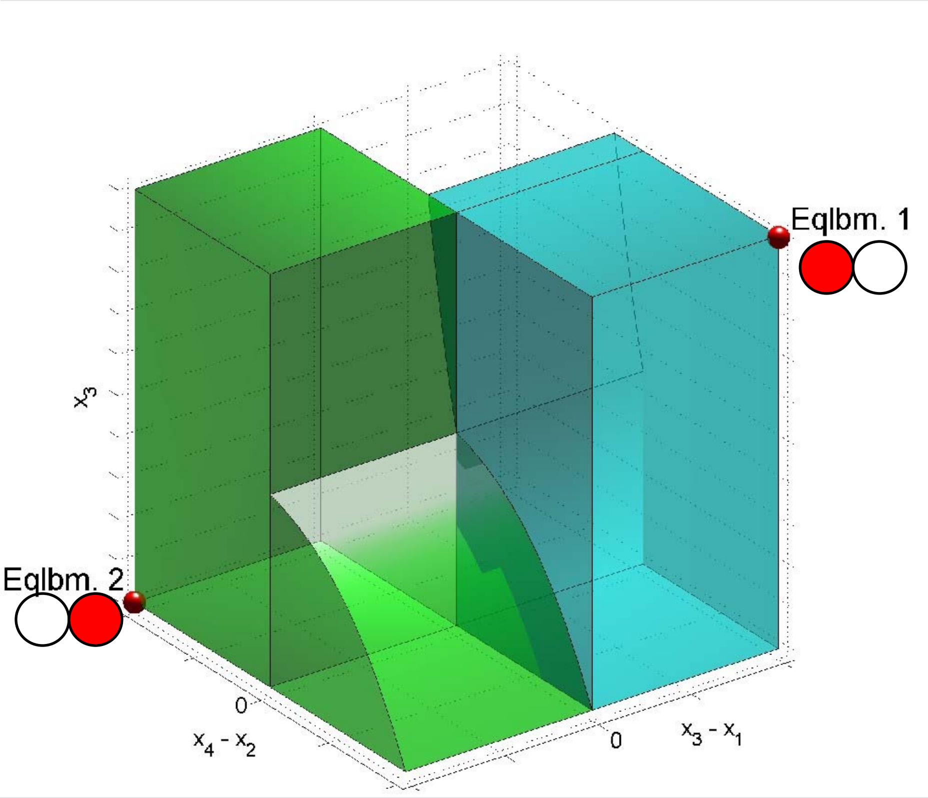
- Compute reachable set from equilibrium states by tracing executions backward through discrete state-space
- Certain regions of continuous state-space may not be resolvable
- Resultant reachable sets are under-approximations



[Ghosh, Tomlin]

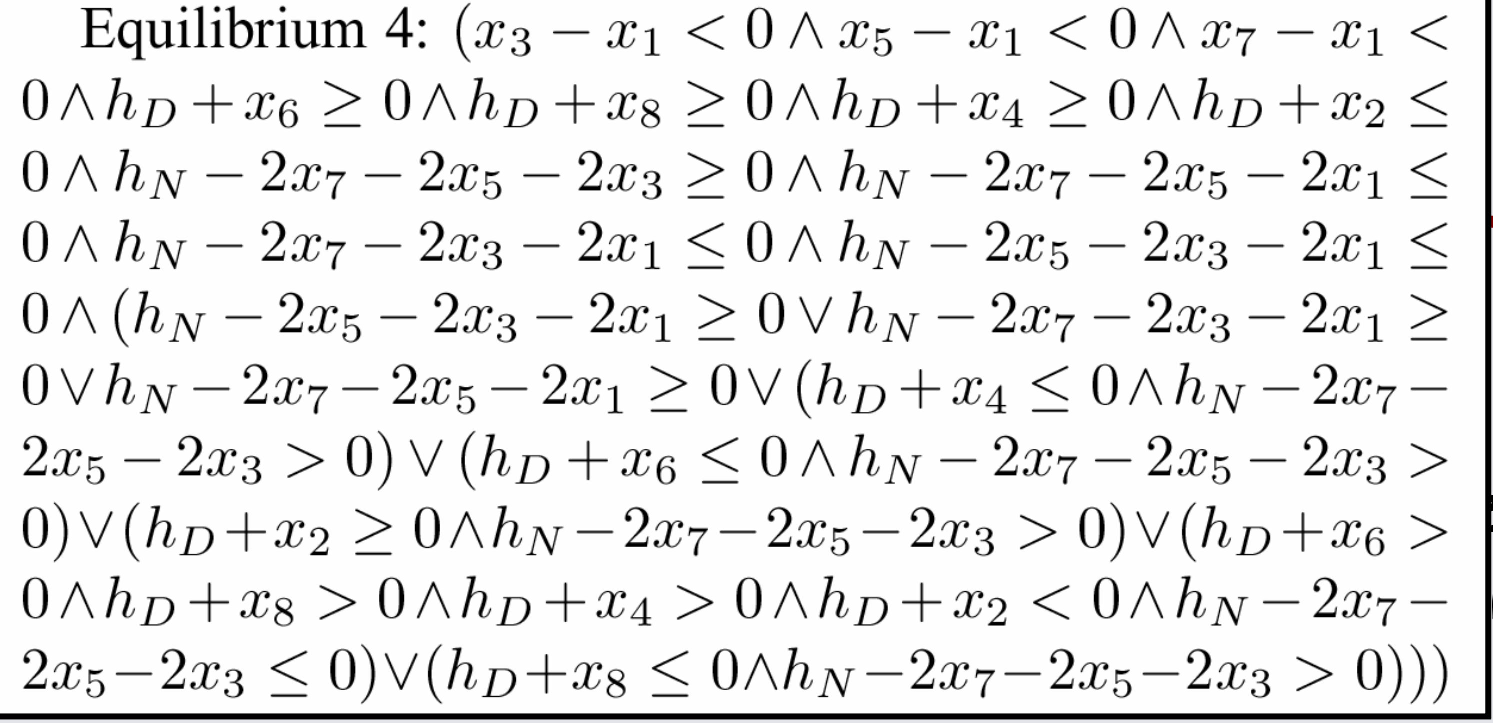
Visualization of Reach Sets





- Projection of symbolic backward reachable sets

[Ghosh, Tomlin]



Equilibrium 4: $(x_3 - x_1 < 0 \wedge x_5 - x_1 < 0 \wedge x_7 - x_1 < 0 \wedge h_D + x_6 \geq 0 \wedge h_D + x_8 \geq 0 \wedge h_D + x_4 \geq 0 \wedge h_D + x_2 \leq 0 \wedge h_N - 2x_7 - 2x_5 - 2x_3 \geq 0 \wedge h_N - 2x_7 - 2x_5 - 2x_1 \leq 0 \wedge h_N - 2x_7 - 2x_3 - 2x_1 \leq 0 \wedge h_N - 2x_5 - 2x_3 - 2x_1 \leq 0 \wedge (h_N - 2x_5 - 2x_3 - 2x_1 \geq 0 \vee h_N - 2x_7 - 2x_3 - 2x_1 \geq 0 \vee h_N - 2x_7 - 2x_5 - 2x_1 \geq 0 \vee (h_D + x_4 \leq 0 \wedge h_N - 2x_7 - 2x_5 - 2x_3 > 0) \vee (h_D + x_6 \leq 0 \wedge h_N - 2x_7 - 2x_5 - 2x_3 > 0) \vee (h_D + x_2 \geq 0 \wedge h_N - 2x_7 - 2x_5 - 2x_3 > 0) \vee (h_D + x_6 > 0 \wedge h_D + x_8 > 0 \wedge h_D + x_4 > 0 \wedge h_D + x_2 < 0 \wedge h_N - 2x_7 - 2x_5 - 2x_3 \leq 0) \vee (h_D + x_8 \leq 0 \wedge h_N - 2x_7 - 2x_5 - 2x_3 > 0)))$

able

e if it

ial

- Computationally tractable: reach set is in disjunctive normal form
- Example query: “What steady state does the system reach if Protein A is initially greater than Protein B?”

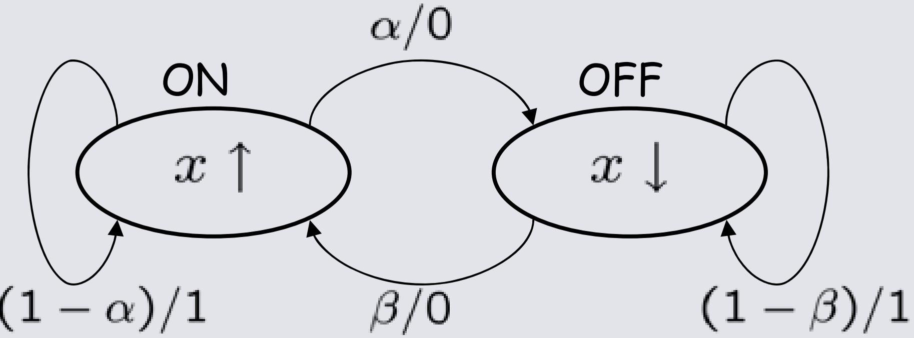
[Ghosh, Tomlin]

Reachability Analysis for Discrete Time Stochastic Hybrid Systems

- Stochastic hybrid systems (SHS) can model uncertain dynamics and stochastic interactions that arise in many systems
- Probabilistic reachability problem:
  - What is the probability that the system can reach a set during some time horizon?
  - (If possible), select a control input to ensure that the system remains outside the set with *sufficiently high* probability

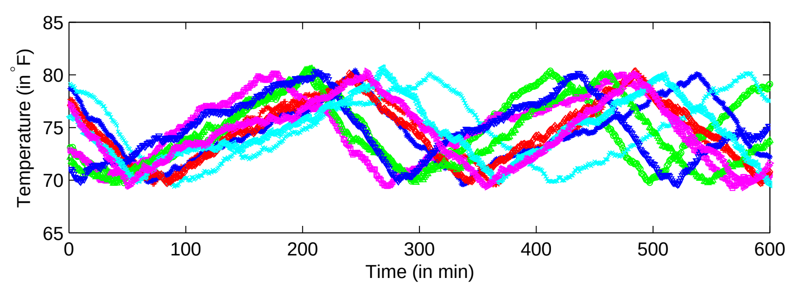
[Amin, Abate, Sastry]

Thermostat



Trivial Switching Control Law

(switch when state hits unsafe set)



Quantitative Verification for Timed Systems

- Defined quantitative notions of similarity between timed systems.
  - Showed quantitative timed similarity and bisimilarity functions can be computed to within any desired degree of accuracy for timed automata.

- Quantitative similarity is robust – close states satisfy similar logic specifications (robustness of TCTL)

- Can view logic formulae as being real valued functions in [0,1] on states.
  - Use *discounting* in the quantification – we would like to satisfy specifications as soon as possible.
  - Defined the logic DCTL – showed model checking decidable for a subset of the logic.

[Prabhu, Majumdar, Henzinger]

Stochastic Games

- Stochastic games: played on game graphs with probabilistic transitions
- Framework for control, controller synthesis, verification
- Classification:
  - How player choose moves
    - Turn-based or Concurrent
  - Information of the players about the game
    - Perfect information or Semi-perfect information or Partial information
- Objectives: ω-regular
  - Captures liveness, safety, fairness
- Results:
  1. Equivalence of semi-perfect turn-based games and perfect concurrent games
  2. Complexity of perfect-information ω-regular turn-based and concurrent games
  3. New notions of equilibria for modular verification
     - Secure equilibria
     - Future directions: application of such equilibria for assume-guarantee style reasoning for modular verification

[Chatterjee, Henzinger]

Optimal control of Stochastic Hybrid Systems

Minimize $E[f(X)]$

Subject to $dX_t = u(X_t, m_t)dt + \sigma(X_t, m_t)dB_t$

$u \in \mathcal{U}$

- $\{B_t \in \mathbb{R}^d : t \geq 0\}$ standard Brownian motion

- $\{X_t \in \mathbb{R}^n : t \geq 0\}$ continuous state. Solves an SDE whose jumps are governed by the discrete state

- $\{m_t \in \{1, \ldots, M\} : t \geq 0\}$ discrete state: continuous time Markov chain.

- $u : \mathbb{R}^n \times \{1, \ldots, M\} \rightarrow \mathbb{R}^n$ control

[**Raffard**, Hu, Tomlin]

Applications:

- Engineering: Maintain dynamical system in safe domain for maximum time.

$$\text{Maximize} \quad E[f(X)] = E[\inf_{t\geq 0}\{t : X(t) \notin U\}]$$

$$\text{Subject to} \quad \frac{dX(t)}{dt} = f(X(t), u(t)) + \sigma(m_t)w(t)$$

- Systems biology: Parameter identification.

$$\text{Minimize} \quad E[f(X)] = ||E[CX_T] - E_{\text{observed}}||$$

$$\text{Subject to} \quad \frac{dX(t)}{dt} = f(X(t), \theta) + \sigma(\theta)w(t)$$

- Finance: Optimal portfolio selection

$$\text{Maximize} \quad E[f(X)] = E[\int_0^{+\infty} e^{-\alpha t} r(X_t)\, dt]$$

$$\text{Subject to} \quad dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dB_t + dJ_t$$

[**Raffard**, Hu, Tomlin]

Major Ongoing Efforts

- Embedded systems modeling and deep compositionality
- Automated abstraction and refinement of hybrid models
- Verification and reachability analysis of approximations
- Algorithms for control and optimization of hybrid systems