Verifying Average Dwell Time

by solving optimization problems

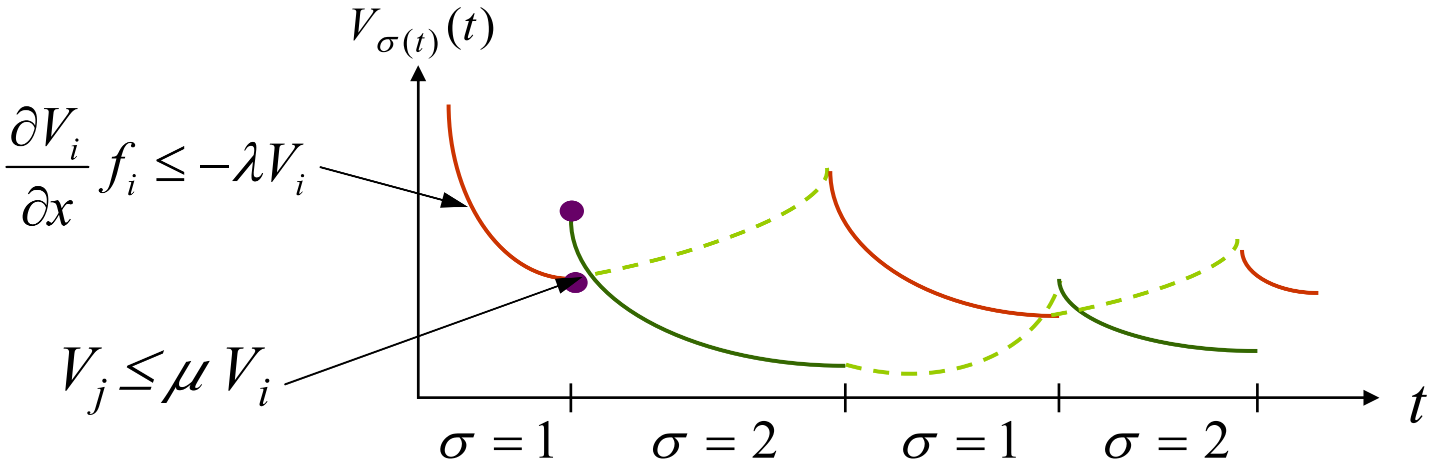
Sayan Mitra, Nancy Lynch and Daniel Liberzon

HSCC 2006, Santa Barbara

Motivation

- Stability properties arise naturally, some examples:
  - Switching supervisory controllers
  - Mobile robots starting from *arbitrary* positions must eventually converge, say on a circle
  - Real-time distributed computing with failures; *once failures stop*, the processes must perform some useful computation.

Stability Under Slow Switchings



- ADT characterizes switching signal σ

- **Definition**: Hybrid automaton ***A*** has average dwell time (ADT) *T* if there exists a constant N_0 such that for every execution α of ***A***,

$$N(\alpha) \leq N_0 + dur(\alpha)/T.$$

N(α): # mode switches in α, dur(α): duration of α

Extra switches: $S_T(\alpha) \equiv N(\alpha) - dur(\alpha)/T$

Problem statement

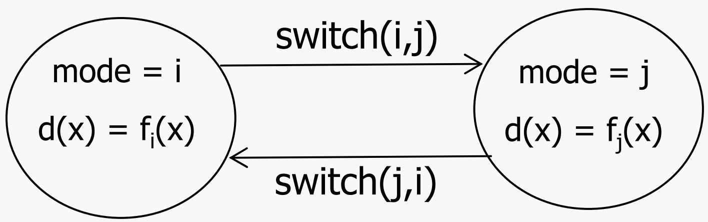
- **Theorem** (Morse & Hespanha): For stability it suffices to show that the modes of ***A*** have a set of Lyapunov functions (λ, μ) and that the ADT of ***A*** $> \log \mu / \lambda$.

- Given hybrid automaton ***A*** and $T>0$, we want to check if T is an ADT for ***A*** ?

  What is the ADT of ***A*** ?

  - Invariant-based method **[M-Liberzon:CDC04]**
  - Optimization-based method for verifying ADT
  - ADT preserving abstraction: switching simulations

Hybrid I/O Automata
[Lynch,Segala,Vaandrager]

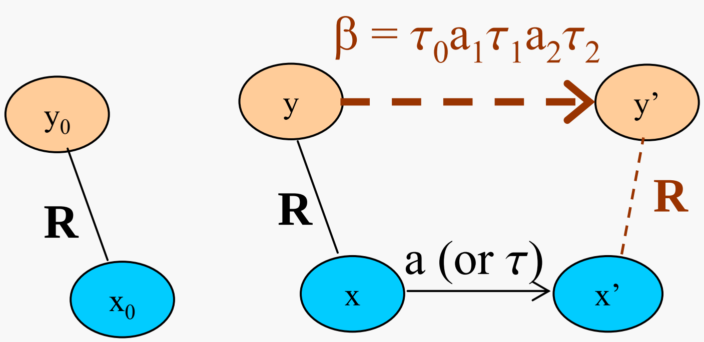


- X: set of state variables, A: set of actions
- Actions bring about state *transitions (mode switches),* $x \rightarrow_a x'$
- A *trajectory* of X is a function τ:[0,t] → *val(X)*
- An *execution* is a sequence $\alpha = \tau_0 a_1 \tau_1 a_2 \tau_2 \ldots$ $dur(\alpha) = \Sigma_i dur(\tau_i)$

- Notion of external behavior, input/output actions and variables
- Abstraction/ implementation relations
- Compositionality

Switching simulation

- Consider two hybrid automata ***A*** and ***B***. A relation $\mathbf{R} \subseteq X_A \times X_B$ is a *switching simulation relation* from ***A*** to ***B*** if :
  - For every start state of ***A*** there is a related start state of ***B***
  - If $x \in X_A$, $y \in X_B$, $x\ \mathbf{R}\ y$ and
    - $x \rightarrow_a x'$, exists an execution fragment β of B,
      s.t. $y \rightarrow_\beta y'$ & $x'\ \mathbf{R}\ y'$ & $N(\beta) \geq 1$, $dur(\beta)=0$
    - $x \rightarrow_\tau x'$, exists an execution fragment β of B,
      s.t. $y \rightarrow_\beta y'$ & $x'\ \mathbf{R}\ y'$ & $dur(\tau) \geq dur(\beta)$



Switching simulations

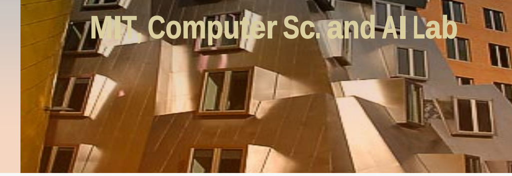
R $\subseteq X_A \times X_B$

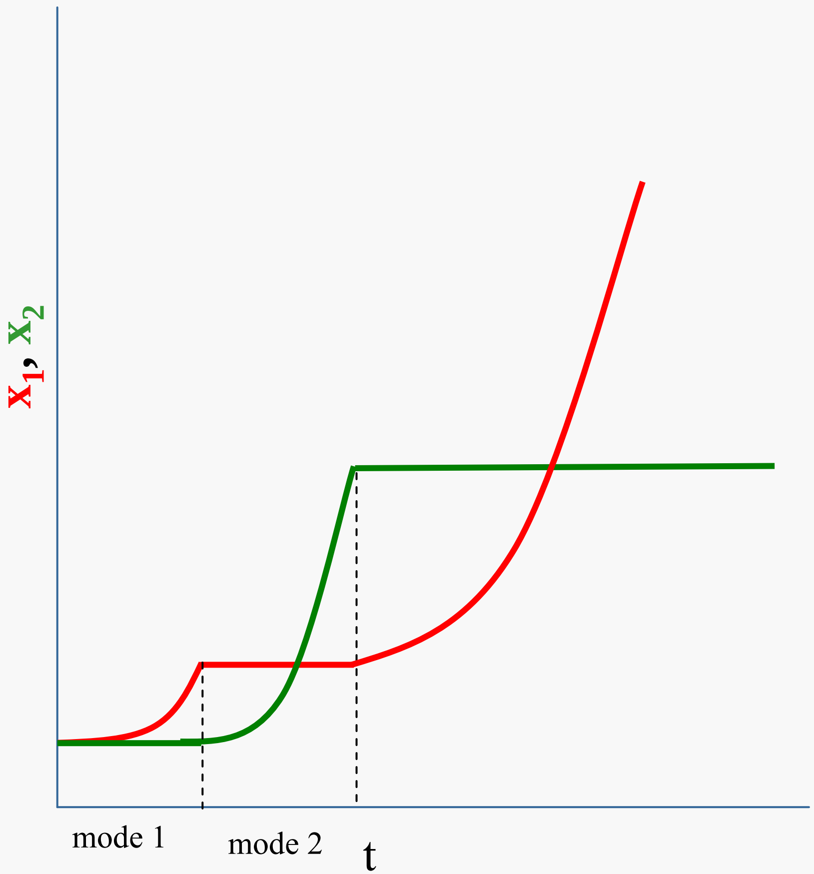
1. For every start state of ***A*** there is a related start state of ***B***
2. If x **R** y and $x \rightarrow_a x'$, $\exists \beta$, s.t. $y \rightarrow_\beta y'$ & x' **R** y' & $N(\beta) \geq 1$, $dur(\beta) = 0$
3. If x **R** y and $x \rightarrow_\tau x'$, $\exists \beta$, s.t. $y \rightarrow_\beta y'$ & x' **R** y' & $dur(\tau) \geq dur(\beta)$

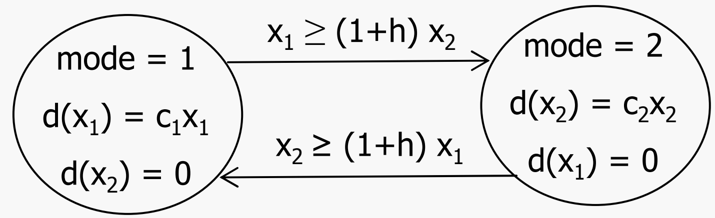
- Suppose **R** is a switching simulation relation from ***A*** to ***B*** and T be ADT of ***A***
- Consider any execution $\alpha = \tau_0 a_1 \tau_1 a_2 \tau_2 \ldots$ of ***A***
- Inductively construct a corresponding execution η of ***B***
- $S_T(\eta) \geq S_T(\alpha)$

- **Theorem:** ***R*** is a switching simulation from ***A*** to ***B*** $\Longrightarrow$ ADT of ***A*** $\geq$ ADT of ***B***.

Linear Hysteresis switch

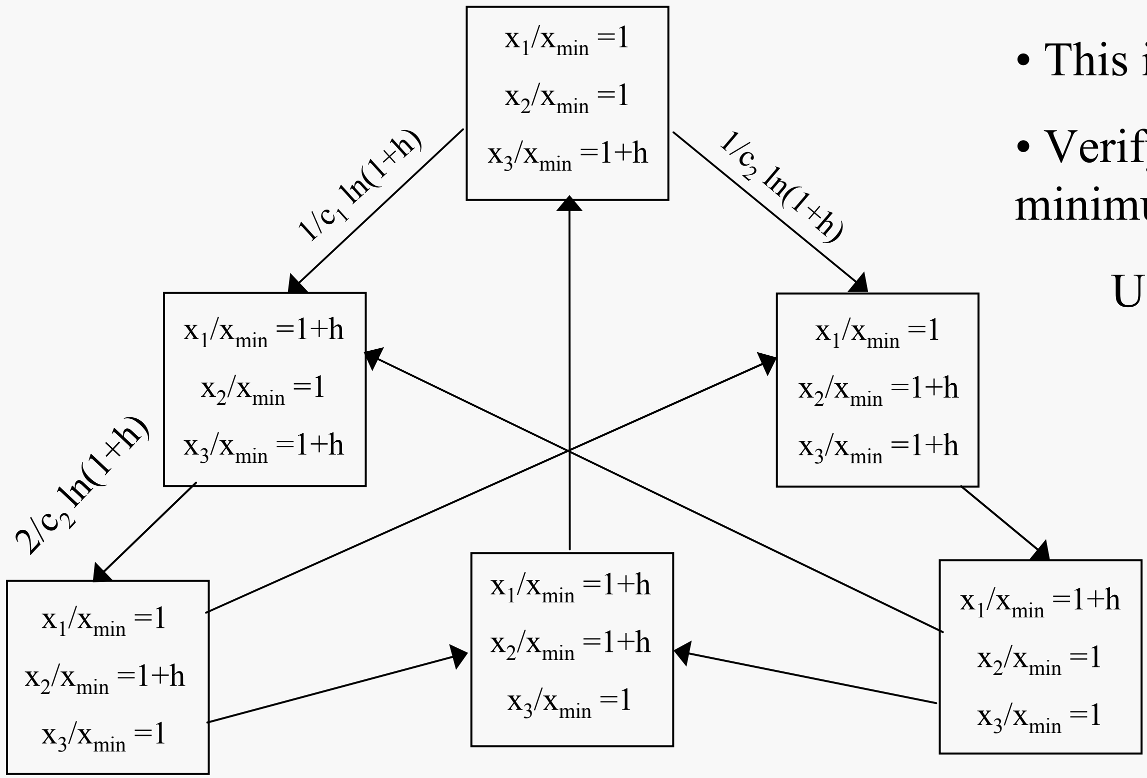






- Not initialized hybrid automaton
- Abstraction using switching simulation

Simple Abstraction



- This is in fact a *one-clock initialized*
- Verifying ADT reduces to finding minimum mean cost cycle.

  Use e.g., Karp's algorithm.

$$
R = \begin{cases} \pi(y.mode) = x.mode \ \wedge \\ \pi(y.mode) = j \Rightarrow \dfrac{x.\mu_j}{x.\mu_{\min}} = e^{c_j y.t} \ \wedge \\ \pi(y.mode) \neq j \Rightarrow \dfrac{x.\mu_j}{x.\mu_{\min}} = y.mode[i][j], i \in \{1,2\} \end{cases}
$$

Optimization problem

- $N(\alpha) \leq N_0 + dur(\alpha)/T$
- $S_T(\alpha) \equiv N(\alpha) - dur(\alpha)/T$

- OPT(T): $\alpha^* \in \arg\max_{\alpha \in \text{execs}} S_T(\alpha)$

  If $S_T(\alpha^*)$ is bounded then T is ADT for **A**,

  Otherwise, α^* an execution violating ADT T

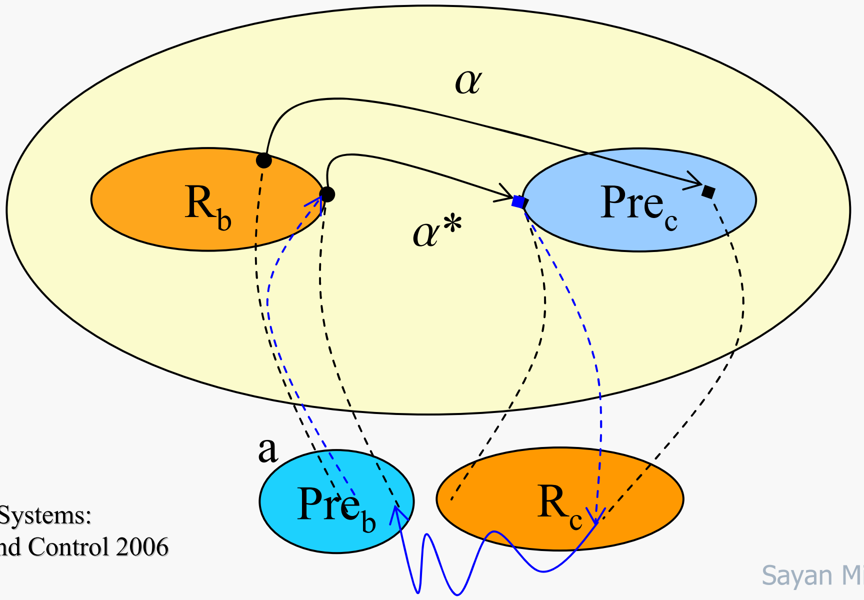
Optimization based approach

Theorem (part 1): If $max_{\alpha \in cycles}$ $S_T(\alpha) > 0$ then OPT(T) is unbounded.

- If ∃ a cycle with $S_T(\alpha) > 0$ then $\alpha . \alpha . \alpha \ldots$ is an execution with unbounded extra switches.

Theorem (part 2): For initialized and rectangular ***A***, OPT(T) is unbounded only if $max_{\alpha \in \text{cycles}}\ S_T(\alpha) > 0$.

- If OPT(T) is unbounded, ∃ execution α, $S_T(\alpha) > m^3$.
- $N(\alpha) > m^3 + dur(\alpha)/T$.
- ∃ sequence of 3 modes that repeat in α, say a-b-c.
- Since ***A*** is rectangular and initialized, ∃ cyclic α^* such that $S_T(\alpha^*) \geq S_T(\alpha)$.



MILP formulation

Mixed Integer Linear Program to find cycles with extra switches for initialized rectangular automata.

Objective function: $S_{\tau_a} : \frac{K}{2} - \frac{1}{\tau_a} \sum_{i=0,2,\ldots}^{K} t_i$

Mode: $\forall\, i \in \{0, 2, \ldots, K\},\ \sum_{j=1}^{N} m_{ij} = 1 \ and\ \forall\, i \in \{1, 3, \ldots, K-1\},\ \sum_{j=1}^{N}\sum_{k=1}^{N} p_{ijk} = 1$ (1)

Cycle: $\mathbf{x}_0 = \mathbf{x}_K$ and $\forall\, j \in \{1, \ldots, N\}, m_{0j} = m_{Kj}$ (2)

Preconds: $\forall\, i \in \{1, 3, \ldots, K-1\},\ \sum_{j=1}^{N}\sum_{k=1}^{N} G[j,k].p_{ijk}.\mathbf{x}_i \leq \sum_{j=1}^{N}\sum_{k=1}^{N} p_{ijk}.g[j,k]$ (3)

Initialize: $\forall\, i \in \{1, 3, \ldots, K-1\},\ \sum_{j=1}^{N}\sum_{k=1}^{N} R[j,k].p_{ijk}.\mathbf{x}_{i+1} \leq \sum_{j=1}^{N}\sum_{k=1}^{N} p_{ijk}.r[j,k]$ (4)

Invariants: $\forall\, i \in \{0, 2, \ldots, K\},\ \sum_{j=1}^{N} A[j].m_{ij}.\mathbf{x}_i \leq \sum_{j=1}^{N} m_{ij}.a[j]$ (5)

Evolve: $\forall\, i \in \{0, 2 \ldots, K\},\ \mathbf{x}_{i+1} = \mathbf{x}_i + \sum_{j=1}^{N} c[j].m_{ij}.t_i$ (6)

Conclusions

- Using powerful existing tools (MILP) for verifying ADT, i.e., proving stability.
- Switching simulations for abstractions

Future work

  - Probabilistic hybrid systems and stability in the stochastic setting, using Lyapunov function like techniques
  - Explore other properties that are quantified over executions; liveness properties
  - Finding switching simulations automatically ?