Advanced Topic (1): Systems with Discontinuities

Joseph L. Hellerstein, Jie Liu
Microsoft

Feb. 25, 2008

Announcements

- Paper Reviews and Discussions

Paper Reviews			
due date	topics	<2 credits	2 credits
2/29/2008	Performance Control	pick 2	all 3
3/7/2008	Resource provisioning	pick 2	all 3
3/14/2008	Network/distributed sys	pick 2	pick 3
Presentations			
data	topics	2 credits	
3/3/2008	Performance Control	3	
3/10/2008	Resource provisioning	3	
3/17/2008	Network/distributed sys	4	

- If you registered for 2 credits, let us know by the end of today which paper(s) you'd like to present. We may overrule your choices and assign you a paper if you don't pick one.

Paper discussion topics

- 3/3/08 (Performance Control):
  - Sujay Parekh, Kevin Rose, Yixin Diao, Victor Chang, Joseph L. Hellerstein, Sam Lightstone, Matthew Huras, "Throttling Utilities in the IBM DB2 Universal Database Server," *American Control Conference*, 2004.
  - S Parekh, N Gandhi, JL Hellerstein, D Tilbury, TS Jayram, J Bigus, "Using Control Theory to Achieve Service Level Objectives in Performance Management," Real Time Systems Journal, Vol.23, No. 1-2, 2002.
  - Ying Lu, Tarek F. Abdelzaher, Avneesh Saxena. "Design, Implementation, and Evaluation of Differentiated Caching Services." *IEEE Transactions on Parallel and Distributed Systems* Vol. 15, No. 5, pp. 440-452, May 2004..

Paper discussion topics

- 3/10/08 (Resource provisioning)
  - Jin Heo, Dan Henriksson, Xue Liu, Tare Abdelzaher, "Integrating Adaptive Components: An Emerging Challenge in Performance-Adaptive Systems and a Server Farm Case-Study," in Proceedings of the 28th IEEE Real-Time Systems Symposium (RTSS'07), Tucson, Arizona, 2007.
  - Pradeep Padala, Kang G. Shin, Xiaoyun Zhu, Mustafa Uysal, Zhikui Wang, Sharad Singhal, Arif Merchant, Kenneth Salem. "Adaptive Control of Virtualized Resources in Utility Computing," Eurosys, 2007
  - Dara Kusic and Nagarajan Kandasamy, "Risk-Aware Limited Lookahead Control for Dynamic Resource Provisioning in Enterprise Computing Systems" IEEE International Conference on Autonomic Computing (ICAC '06), June 2006, pp 74-83.

Paper discussion topics

- 3/17/2008 (Network and distributed systems)
  - C. V. Hollot, Vishal Misra, Don Towsley, and Weibo Gong. A control theoretic analysis of RED. In Proceedings of the IEEE Conference on Computer Communications (INFOCOM), Anchorage, AK, USA, April 22–26 2001. IEEE
  - Srinivasan Keshav. "A control-theoretic approach to flow control." In Proceedings of the ACM Conference on Communications Architecture & Protocols (SIGCOMM '91), pages 3–15, Zurich, Switzerland, September 1991. ACM, ACM Press.
  - Hieu Le Khac, Dan Henriksson, and Tarek F Abdelzaher, "A Control Theory Approach to Throughput Optimization in Multi-Channel Collection Sensor Networks", IPSN 2007, Cambridge, MA,
  - X. Wang, D. Jia, C. Lu and X. Koutsoukos, "DEUCON: Decentralized End-to-End Utilization Control for Distributed Real-Time Systems," IEEE Transactions on Parallel and Distributed Systems, 18(7):996-1009, July 2007

Today

- Lyapunov stability
- Systems with Discontinuities
  - Model discontinuity using state machines
  - Hybrid systems
    - Timed automata
    - Switched linear systems
    - Markov jump linear systems
- ITA Software example (Carl de Marcken)

Stability

- Why are control people so obsessed with stability?
  - It’s a safety property for feedback systems.
  - Stability is about convergence.
  - Disturbance rejection
  - Reference tracking
  - Robustness
- Classes of stability
  - Bounded-input-bounded-output (BIBO) stability
  - State-based (Lyapunov) stability

Recall: State Trajectories

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k)$$

$$\begin{aligned}
\mathbf{x}(1) &= \mathbf{A}\mathbf{x}(0) + \mathbf{B}\mathbf{u}(0) \\
\mathbf{x}(2) &= \mathbf{A}\mathbf{x}(1) + \mathbf{B}\mathbf{u}(1) \\
&= \mathbf{A}^2\mathbf{x}(0) + \mathbf{A}\mathbf{B}\mathbf{u}(0) + \mathbf{B}\mathbf{u}(1) \\
\mathbf{x}(3) &= A\mathbf{x}(2) + \mathbf{B}\mathbf{u}(2) \\
&= \mathbf{A}^3\mathbf{x}(0) + \mathbf{A}^2\mathbf{B}\mathbf{u}(0) + \mathbf{A}\mathbf{B}\mathbf{u}(1) + \mathbf{B}\mathbf{u}(2)
\end{aligned}$$

$$\begin{aligned}
\mathbf{x}(k) &= \mathbf{A}^k\mathbf{x}(0) + \mathbf{A}^{k-1}\mathbf{B}\mathbf{u}(0) + \mathbf{A}^{k-2}\mathbf{B}\mathbf{u}(1) + \cdots + \mathbf{A}\mathbf{B}\mathbf{u}(k-2) + \mathbf{B}\mathbf{u}(k-1) \\
&= \mathbf{A}^k\mathbf{x}(0) + \sum_{i=0}^{k-1} \mathbf{A}^{k-1-i}\mathbf{B}\mathbf{u}(i)
\end{aligned}$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) = \mathbf{C}\mathbf{A}^k\mathbf{x}(0) + \mathbf{C}\sum_{i=0}^{k-1} \mathbf{A}^{k-1-i}\mathbf{B}\mathbf{u}(i)$$

Mathematical tools for analyzing stability

- Vector Norm $\|.\|$

For vector space $\Re^n$, $\|.\|:\Re^n \to \Re_0^+$ satisfying:

  - For $\alpha \in \Re,\ x \in \Re^n,\ \|\alpha \cdot x\| = |\alpha| \cdot \|x\|$
  - For $x, y \in \Re^n, \|x+y\| \le \|x\| + \|y\|$
  - $\|x\| = 0 \Leftrightarrow x = 0$

- Examples:
  - p-norm: $\|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{1/p}$
    - p=1, Manhattan Norm
    - p=2, Euclidean Norm
  - ∞-norm: $\|x\|_\infty = \max(|x_1|, |x_2|, \ldots |x_n|)$

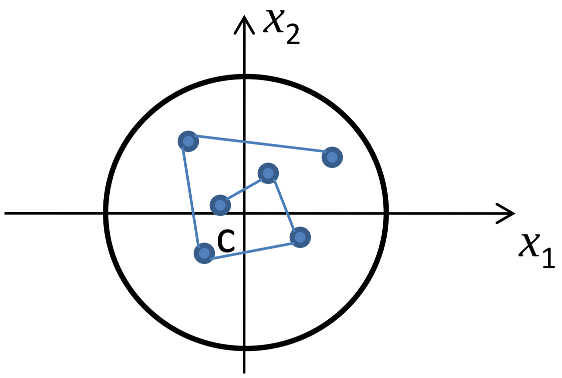
- All norms are equivalent (for finite n):

$$C\|x\|_A \le \|x\|_B \le D\|x\|_A$$

Definitions of Stability

$$x(k+1) = f(x(k))$$

- f is *continuous* at c iff for all $\varepsilon > 0$, there exists a $\delta > 0$, such that for all x, $\|x-c\| < \delta \Rightarrow \|f(x) - f(c)\| < \varepsilon$
  - Locally continuous (wrt c)
  - Globally continuous (for all c)
  - Uniformly continuous (wrt k)
- f is *Lyapunov stable* at c iff for all $\varepsilon > 0$, there exists a $\delta > 0$, such that for all x, $\|x-c\| < \delta \Rightarrow \|f^n(x) - f^n(c)\| < \varepsilon$, for all $n \in \aleph$
  - n = 1, *contraction map*

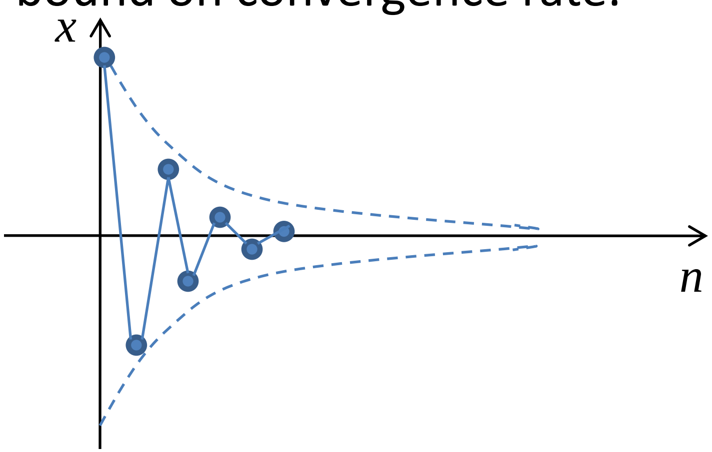


Definitions of Stability

$$x(k+1) = f(x(k))$$

- f is *asymptotically stable* at c iff there exists a $\delta > 0$, such that for all x, $\|x-c\| < \delta \Rightarrow \|f^n(x) - f^n(c)\| \to 0$ as $n \to \infty$
  - Convergence.

- f is *exponentially stable* at c iff there exists a $\delta > 0$ and $a > 0$, such that for all x, $\|x-c\| \le \delta \Rightarrow \|f^n(x) - f^n(c)\| \le a^n$
  - There is a bound on convergence rate.



LTI System Stabilities

$$x(k+1) = Ax(k) + Bu(k)$$

$$\mathbf{x}(k) = \mathbf{A}^k\mathbf{x}(0) + \sum_{i=0}^{k-1} \mathbf{A}^{k-1-i}\mathbf{B}\mathbf{u}(i)$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) = \mathbf{C}\mathbf{A}^k\mathbf{x}(0) + \mathbf{C}\sum_{i=0}^{k-1} \mathbf{A}^{k-1-i}\mathbf{B}\mathbf{u}(i)$$

- For LTI system,
  - local stability = global stability
  - asymptotically stability = exponential stability.
  - State stability $\Rightarrow$ BIBO stability
- To see this: recall $A = P^{-1}\Lambda P$

$$\|x+y\| \le \|x\| + \|y\|$$

$$\|Ax\| \le \lambda_{\max}(A)\|x\|$$

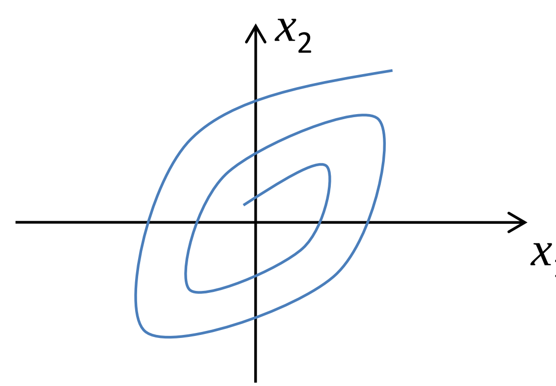
Lyapunov Stability Theorem

- A continuous function f is *positive definite* if $f(0) = 0$ and $f(x) > 0$ for every nonzero x.
  - A matrix A is positive definite if $x^T Ax > 0$ for $x \neq 0$

- Lyapunov direct method: $x(k+1) = f(x(k))$ is stable if there exists a *positive definite* function $V(x)$, such that:

$$\Delta V(k) = V(k+1) - V(k) \leq 0$$

- Remarks:
  - In general, $P \neq I$
  - This is a sufficient condition
  - Works for any system.



Lyapunov Stability Theorem

- Can be shown, for LTI systems can choose:

$$V(x) = x^T P x$$

$$\Delta V(k) = V(k+1) - V(k)$$

$$= x^T(k+1) P x(k+1) - x^T(k) P x(k)$$

$$= x^T(k)(APA^T - P)x(k)$$

- So, look for Q positive semi-definite, such that

$$APA^T - P = -Q$$

Algebraic Lyapunov Equation

- A discrete-time LTI system is asymptotic stability if and only if for all Q positive definite, we can find a unique positive definite P.

Validating Example

Assume: $A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, Q = I$

Solve: $APA^T - P = I$

$$\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} p_1 & p_2 \\ p_3 & p_4 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} - \begin{bmatrix} p_1 & p_2 \\ p_3 & p_4 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$\begin{bmatrix} (\lambda_1^2 - 1) p_1 & (\lambda_1 \lambda_2 - 1) p_2 \\ (\lambda_1 \lambda_2 - 1) p_3 & (\lambda_1^2 - 1) p_4 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$P = \begin{bmatrix} \frac{1}{1-\lambda_1^2} & 0 \\ 0 & \frac{1}{1-\lambda_1^2} \end{bmatrix}$$

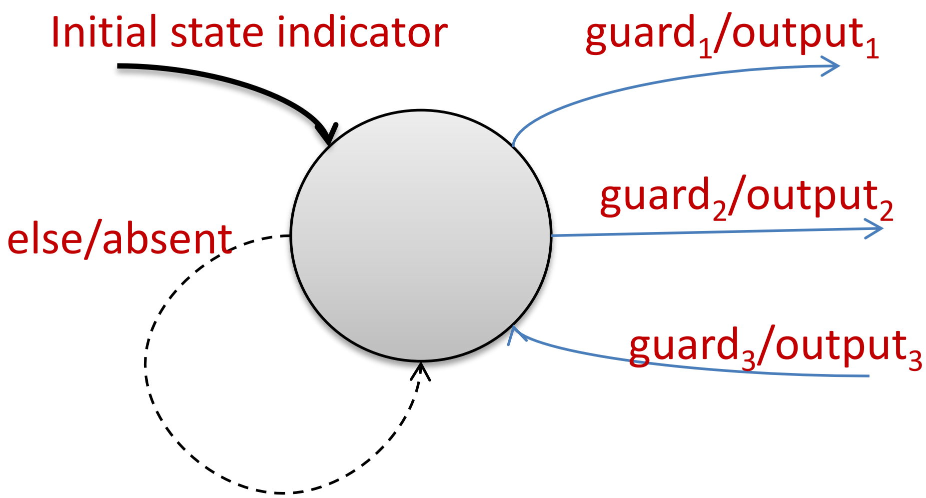
$P > 0 \text{ iff } |\lambda_1| < 1 \text{ and } |\lambda_2| < 1$

Systems with Discontinuities

- State machines
- Hybrid systems
  - Timed automata
  - Switched linear systems
  - Markov jump linear systems

(Mealy) State Machines

- StateMachine = (States, Inputs, Outputs, update, initialState)



update: States X Inputs → States X Outputs

- Stutter:

  absent (⊥) ∈ Inputs, and absent ∈ Outputs

  Update(s, ⊥) = (s, ⊥)

- Finite State Machine: if States set is finite.

Example: Parking Meter

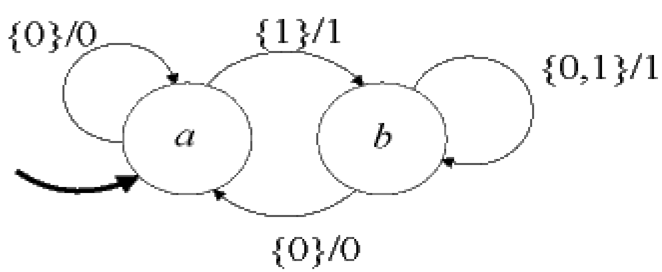
- *States* = {0, 1, 2, ..., 60}
- *Inputs* = {*coin5*, *coin25*, *tick*, *absent*}
- *Outputs* = { *expired*, *safe*, *absent* }
- *initialState* = 0
  - where *safe* simply indicates that there is still money in the meter. The *update* function is given by the following table:

if	**then *update*(*s*, *x*) =**
x = *tick* and (*s* = 0 or *s* = 1)	(0, *expired*)
x = *tick* and *s* > 1	(*s* - 1, *safe*)
x = *coin5*	(min(*s* + 5, 60), *safe*)
x = *coin25*	(min(*s* + 25, 60), *safe*)
x = *absent*	(*s*, *absent*)

- **Example:**
  - *InputSequence* = *coin25*, $tick^{20}$, *coin5*, $tick^{10}$, ...
  - *StateResponse* = 0, 25, 24, ..., 6, 5, 10, 9, 8, ..., 2, 1, 0^5
  - *OutputSequence* = *expired*, *safe*, *safe*, ..., *safe*, *safe*, *safe*, *safe*, *safe*, ..., *safe*, *safe*, $expired^5$

Valid Trajectories

- **Receptiveness**: The machine can always react to an input symbol.
- **Determinism:** For a deterministic machine, the guards on the arcs emerging from any state are mutually exclusive (they have no common elements).
- **Nondeterminism**: When multiple updates (arcs) are enabled by an input symbol, the machine is free to choose any enabled transition.



InputSequence: 0, 1, 0, 1, 0, 1, ...

StateSequence1:*a*, *a*, *b*, *a*, *b*, *a*, *b*, ...
OutputSequence1: 0, 1, 0, 1, 0, 1, ...

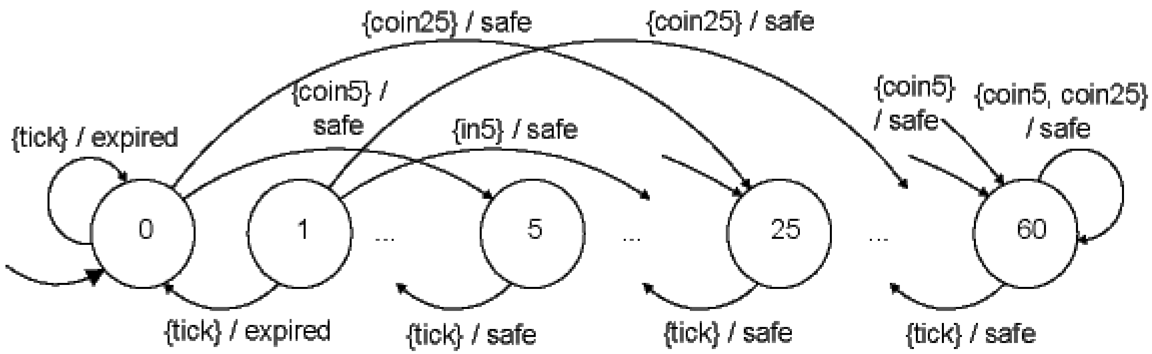
StateSequence2:*a*, *a*, *b*, *b*, *b*, *a*, *b*, ...
OutputSequence2: 0, 1, 1, 1, 0, 1, ...

StateSequence3:*a*, *a*, *b*, *b*, *b*, *b*, *b*, ...
OutputSequence3: 0, 1, 1, 1, 1, 1, ...

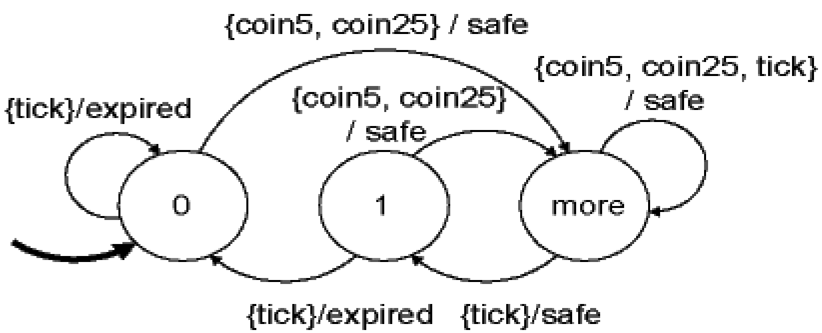
...

Abstraction Using Nondeterminism

Parking meter example



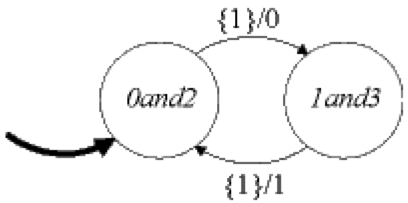
deterministic



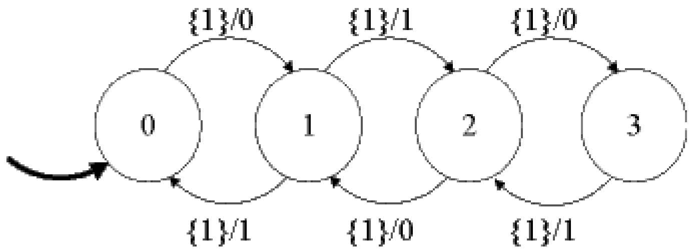
nondeterministic

Simulation of FSM

- Intuitively, it is a game:
  - Consider a game, where each machine starts in its initial state. Then, given an input, *A* reacts, and *B* tries to react in such a way as to produce the same output (given the same input). If *B* can always do this, *B* is said to **simulate** *A*.



Machine A

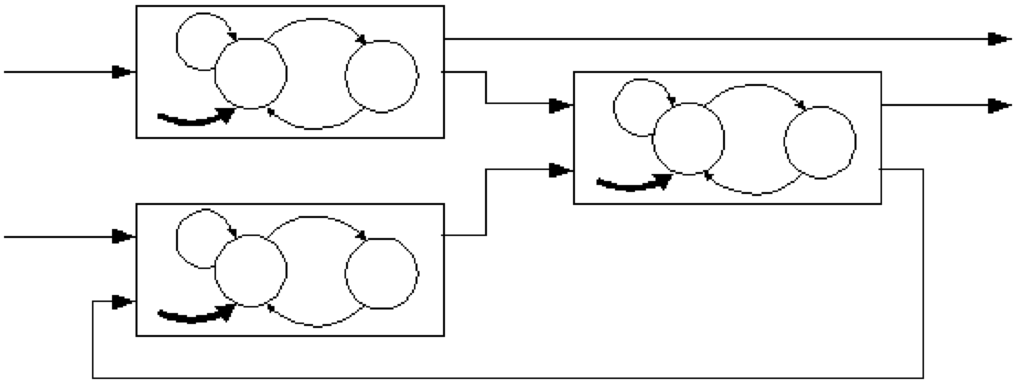


Machine B

- Bisimulation: The game can be played at any state in any order. E.g. A moves for one step, B matches. Then B moves for one step and A matches. And so on.
- A simulates B and B simulates A $\nRightarrow$ A bisimulates B.
  - Simulation is an abstraction relation
  - Bisimulation is an equivalence relation

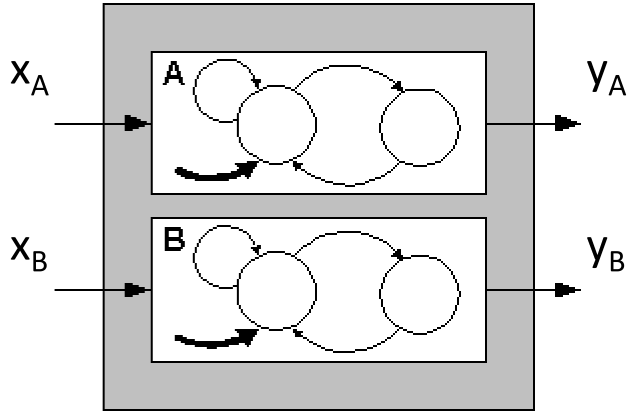
Composition of FSM

- **Synchrony**: Consider a set of interconnected components, where each component is a state machine, as in:



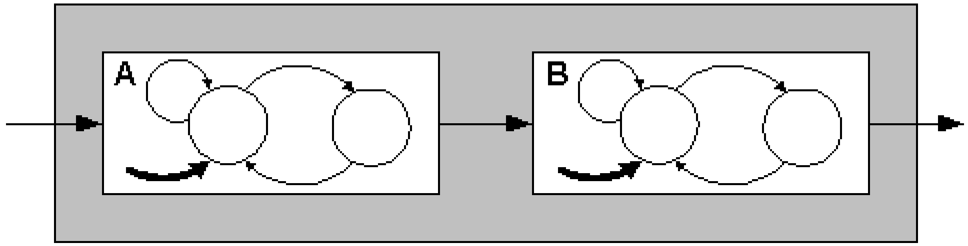
- We construct a state machine model for the composition that is **synchronous** and **reactive:**
  - The reaction of the composite consists of exactly one reaction of each component.
  - The reaction of the composite is triggered by an input to the composite.
  - The component reactions are **simultaneous** and **instantaneous**.
  - The output of each component is simultaneous with its input.
  - The output of the composite is simultaneous with the input to the composite.
  - The output of each component is visible to its destination *in the same reaction.*

Side-by-side Composition



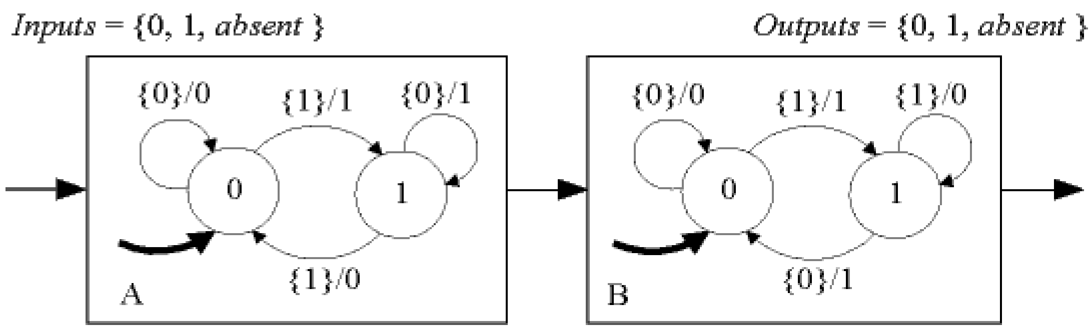
- Let the composition be given by *StateMachine* = (*States, Inputs*, *Outputs*, *update*, *initialState*)
- Definition of the composition:
  - $States = States_A \times States_B$
  - $Inputs = Inputs_A \times Inputs_B$
  - $Outputs = Outputs_A \times Outputs_B$
  - $initialState = (initialState_A, initialState_B)$
  - $update((s_A, s_B), (x_A, x_B)) = ((s'_A, s'_B), (y_A, y_B))$
  - where
    - $(s'_A, y_A) = update_A(s_A, x_A)$
    - $(s'_B, y_B) = update_B(s_B, x_B)$
- Stuttering element: *stutter =* (*absent*, *absent*)

Cascade Composition



- Assumption: $Outputs_A \subseteq Inputs_B$
- Definition of the composition:
  - $States = States_A \times States_B$
  - $Inputs = Inputs_A$
  - $Outputs = Outputs_B$
  - $initialState = (initialState_A, initialState_B)$
  - $update((s_A, s_B), x) = ((s'_A, s'_B), y_B)$
  - where
    - $(s'_A, y_A) = update_A(s_A, x)$
    - $(s'_B, y_B) = update_B(s_B, y_A)$
- Stuttering element: *stutter = absent*

Example: Differential CODEC

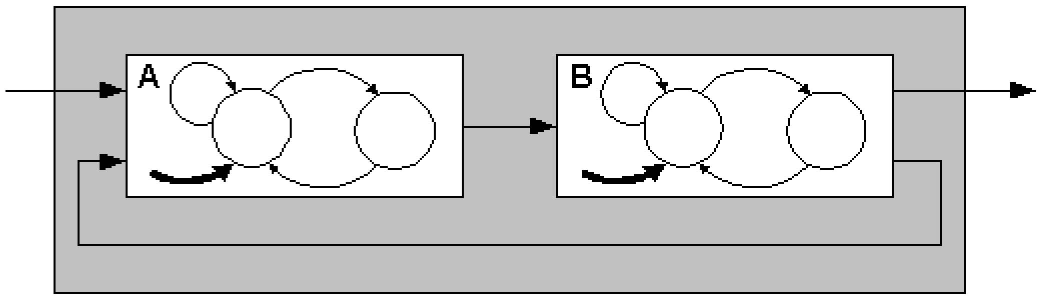


States = {(0, 0), (0, 1), (1, 0), (1, 1)}
Inputs = *Outputs* = {0, 1, *absent* }
initialState = (0, 0)

current state	(next state, output) for input		
	0	1	absent
(0, 0)	((0, 0), 0)	((1, 1), 1)	((0, 0), absent)
(0, 1)	((0, 0), 1)	((1, 1), 0)	((0, 1), absent)
(1, 0)	((1, 1), 1)	((0, 0), 0)	((1, 0), absent)
(1, 1)	((1, 1), 0)	((0, 0), 1)	((1, 1), absent)

- Remarks:
  - The output is always equal to the input. (It works!)
  - States (0, 1) and (1, 0) are not reachable. (This is a form of control!)
  - Can be reduced to a simpler machine that bisimulate the composition.

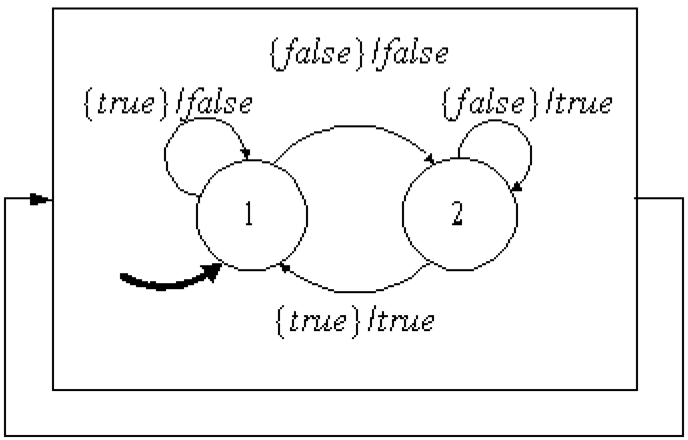
Feedback Composition



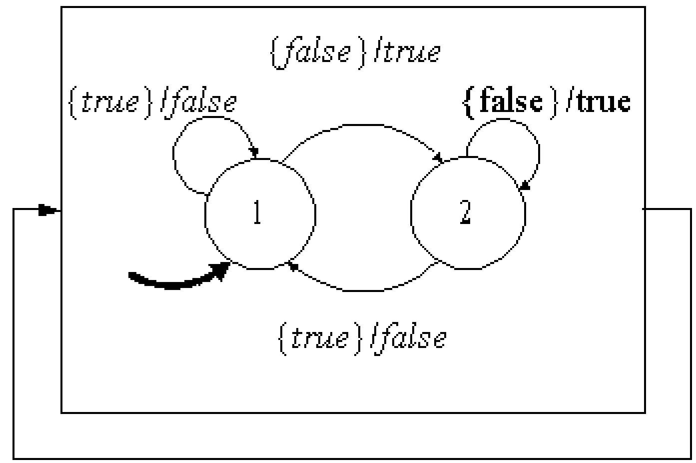
- Assumption:
  - $Outputs_A \subseteq Inputs_B$
  - $Outputs_{B2} \subseteq Inputs_{A2}$
- Definition of the composition:
  - $States = States_A$ ´ $States_B$
  - $Inputs = Inputs_{A1}$
  - $Outputs = Outputs_{B1}$
- *updates* function is found by iteration to a fixed point:
  - Start with *unknown* on the feedback arc
  - Foreach state machine:
    - If output can be determined, produce it
    - If state transition can be determined, take it
  - Repeat until no progress is made.

- Two possible outcomes:
  - All outputs are determined
  - Some signals are still unknown. The composition is ill-formed.

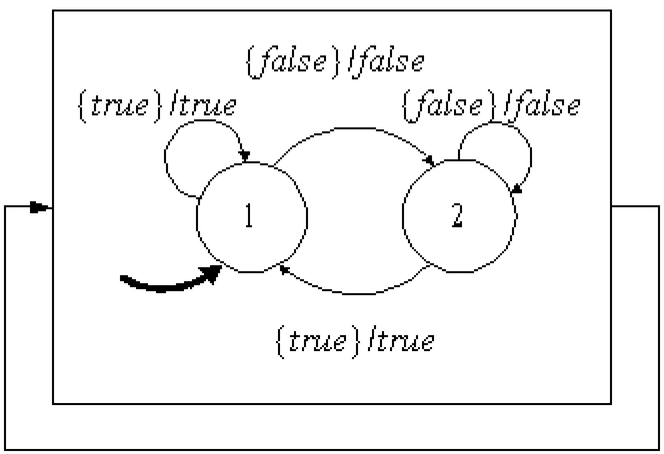
Examples



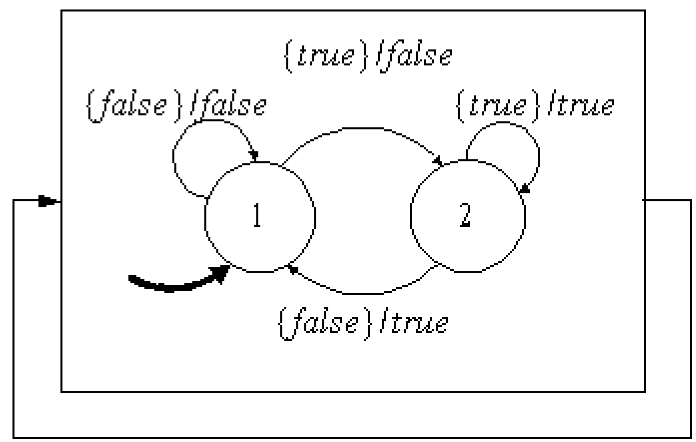
(A)



(B)

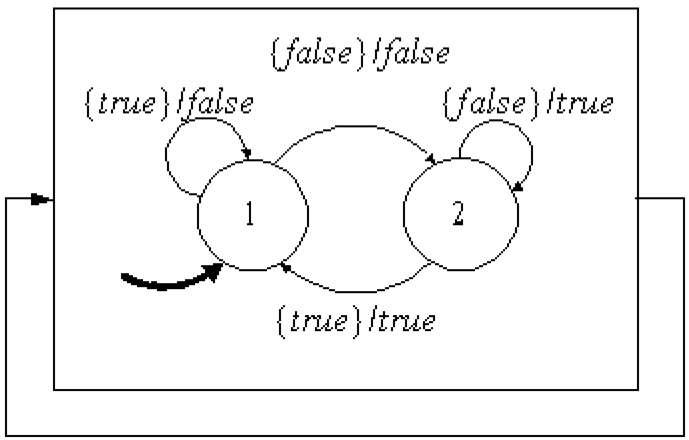


(C)

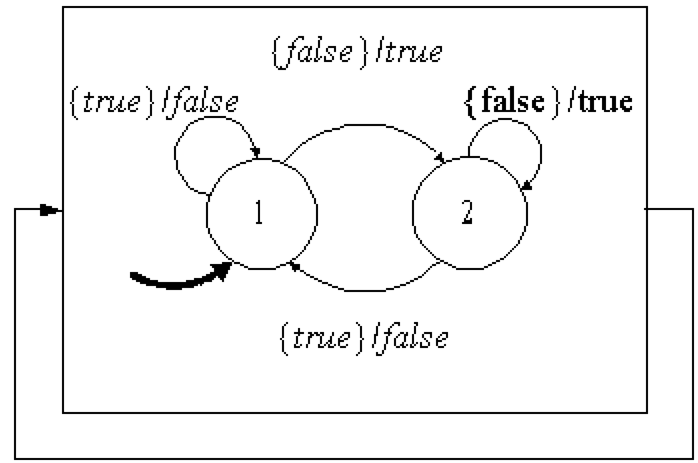


(D)

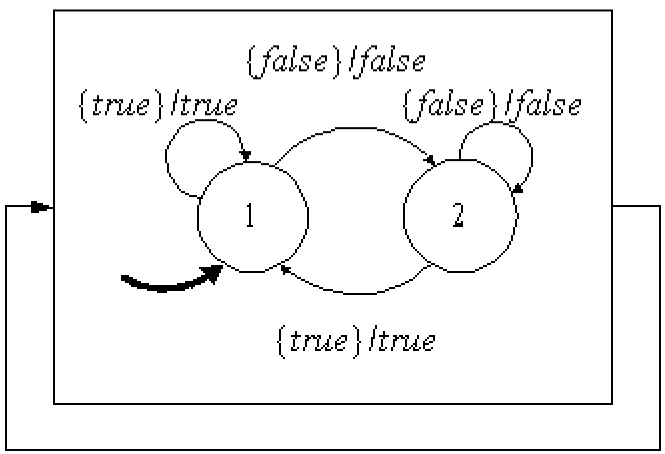
Examples



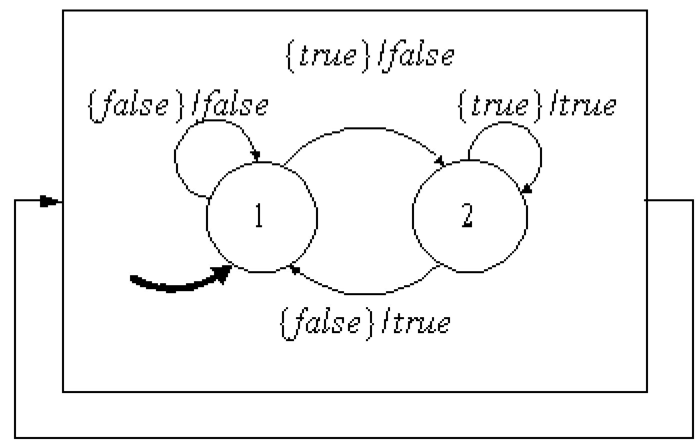
(A): *false, true, false, true, false, ...*



(B): ill-formed (no enabled transitions)



(C): ill-formed (multiple enabled transitions)



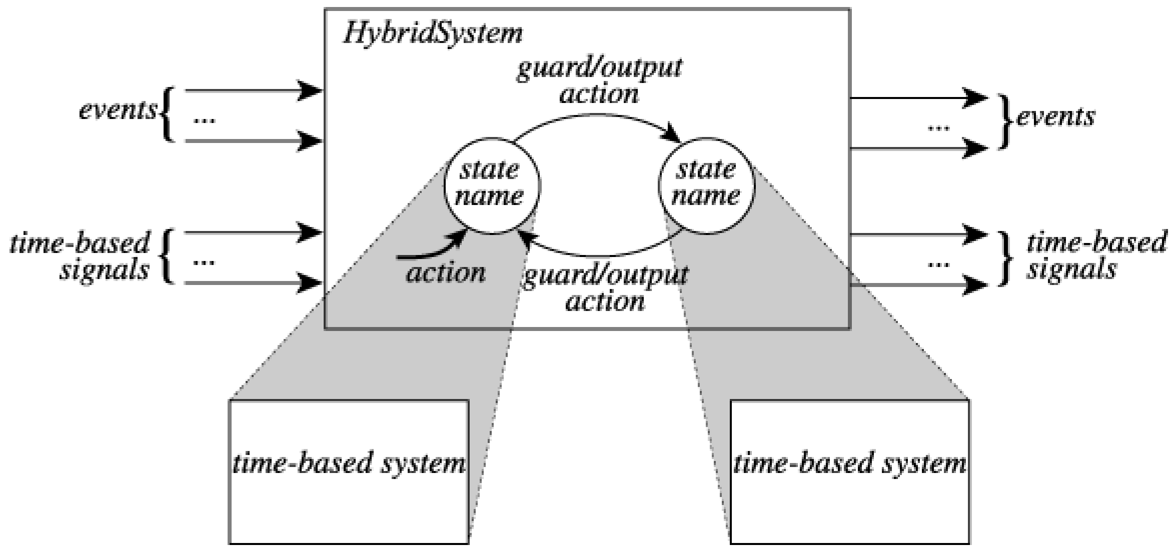
(D): *false, false, false, false, false, ...*

Systems with Discontinuities

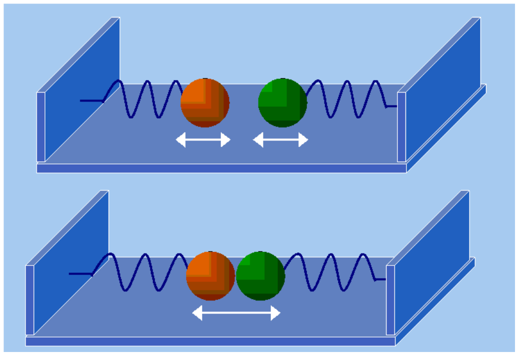
- State machines
- Hybrid systems
  - Timed automata
  - Switched linear systems
  - Markov jump linear systems

Hybrid System Review

- Automata refined into differential/difference equations.

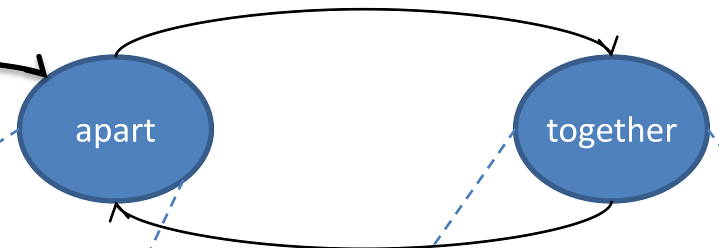


Hybrid System Review



$$y_1(t) = y_2(t) \;/\; y(t) = y_1(t);$$
$$\dot{y}(t) = (\dot{y}_1(t)/m_1 + \dot{y}_2(t)/m_2)/(m_1+m_2)$$

$$y_1 = d_1; y_2 = d_2$$
$$\dot{y}_1 = 0; \dot{y}_2 = 0$$



$$(k_1 - k_2)y(t) + (k_2 p_2 - k_1 p_1) > stickiness \;/$$
$$y_1 = y; y_2 = y; \dot{y}_1 = \dot{y}; \dot{y}_2 = \dot{y}$$

$$\ddot{y}_1(t) = k_1(p_1 - y_1(t))/m_1$$
$$\ddot{y}_2(t) = k_2(p_2 - y_2(t))/m_2$$

$$\ddot{y}(t) = \frac{k_1 p_1 + k_2 p_2 - (k_1+k_2)y(t)}{m_1+m_2}$$

- In general,
  - Refinements can have different state variables;
  - Guard can be defined on state variables;
  - State variables can be assigned to new values on the arcs.
- Lyapunov method is probably your best hope.

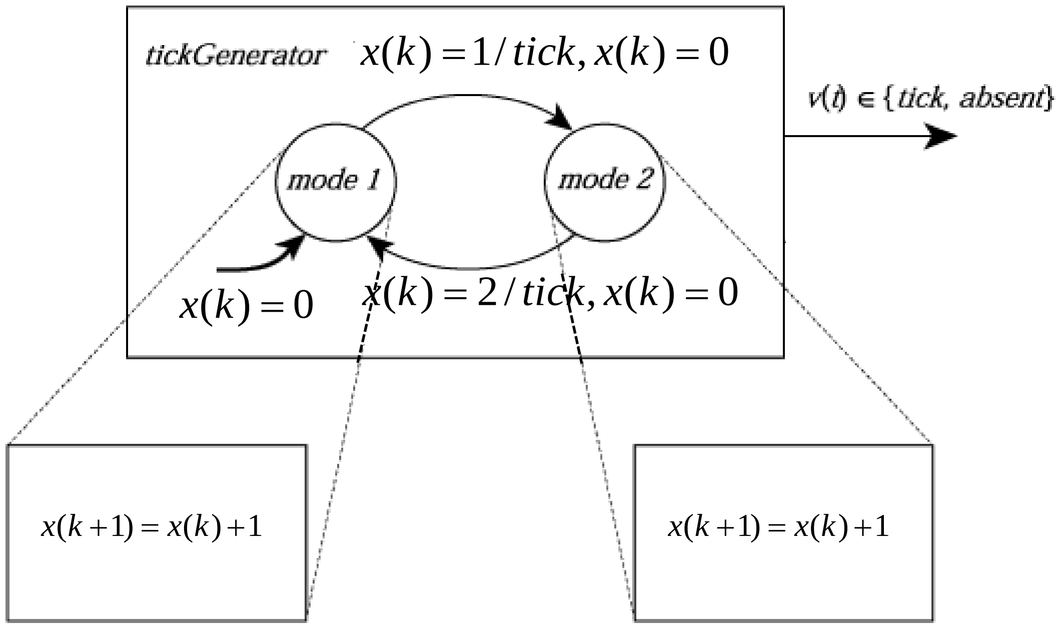
Systems with Discontinuities

- State machines
- Hybrid systems
  - Timed automata
  - Switched linear systems
  - Jump linear systems

Timed Automata

- Introduce clock variables to FSM.

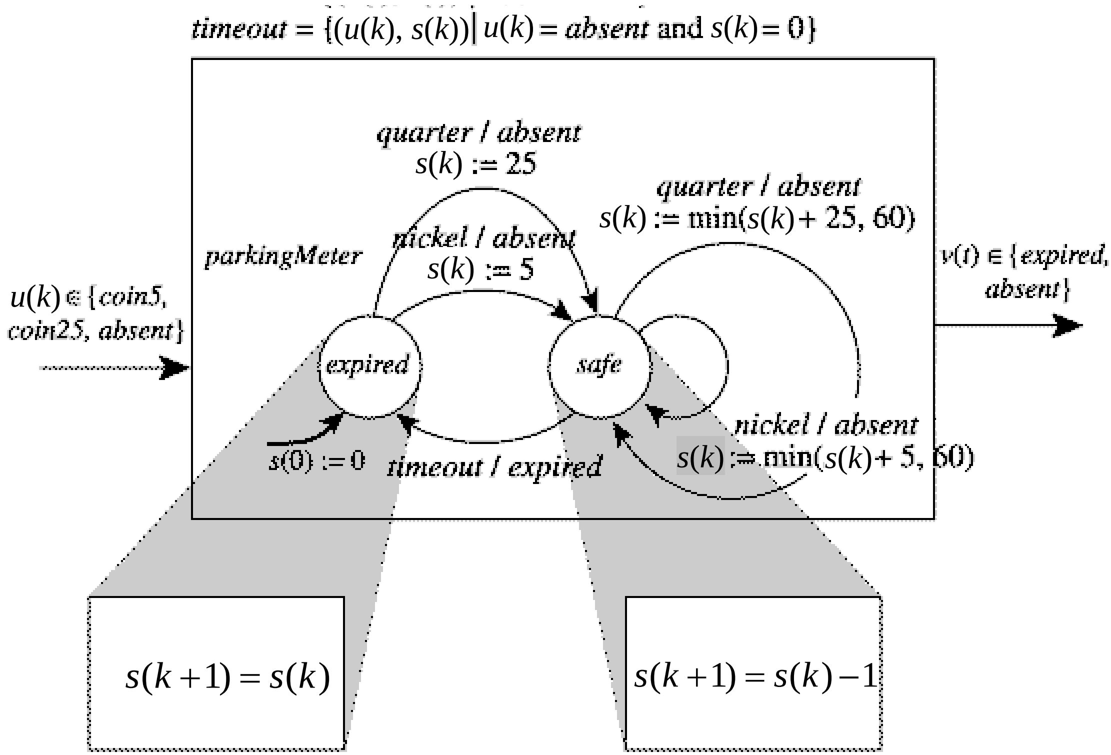
$$x(k+1)=x(k)+c$$



Generate a sequence of ticks at k={1, 3, 4, 6, 7, …}

To avoid confusion, we shall call discrete states *modes* or *locations* from now on.

Example: Parking Meter



Timed automata are particularly useful for modeling timeouts, like in communication protocols, real-time systems, and digital circuits.

Remarks

- Timed automata do not need (explicit) inputs to run. Time is an (implicit) input.
- Transitions do not take time.
  - The trajectory of a timed automaton is an alternation between continuous time elapses and discrete transitions.
- Multi-rate timed automata may have multiple clock variables and they evolve at different rate.
- Composition: similar to FSM, also takes the synchrony assumptions.

Reachability Analysis Basics

- Given a timed automaton A, and a set $L^F \in L$ of target locations, the *reachability* problem is to determine whether some target location is reachable.
- Assume guards involves clock variables have the form:

$$\varphi = x \leq c \,|\, c \leq x \,|\, x < c \,|\, c < x \,|\, \varphi_1 \wedge \varphi_2$$

- Stability in the BIBO sense (on clock variables) can be defined as a reachability problem by introducing "unstable" states with $x > Bound$.
- Basic idea for reachability analysis:
  - A TA typically has infinite states
  - Classify states into equivalent classes (called *stable quotients*).
  - Ensure the equivalent relationship does not pollute non-target locations with target locations.
  - Only need to track a finite set of equivalent classes.
- Time complexity: $n \cdot 2^{O(k \log(kc))}$
  - n locations, k clocks, every clock constraints of A is bounded by c.
- Tools: Timed COSPAN, KRONOS, UPPAAL.

Systems with Discontinuities

- State machines
- Hybrid systems
  - Timed automata
  - Switched linear systems
  - Markov jump linear systems

Switched Linear Systems

$$x(k+1) = A_q x(k) + B_q u(k),\ q \in Q = \{1,\ldots,N\}$$
$$q(k+1) = \delta(q(k), x(k))$$

- Remarks:
  - Same set of variables, continuous states：
    Given $x(k)$, $x(k+1)$ is computed with $q(k)$
  - Piecewise linear
  - Switching decision δ is discontinuous wrt x.
- Lyapunov Stability:
  - Consider Lyapunov functions $V_q(x) = \|W_q x\|_\infty$
  - The switching system is stable if $V_{q_i}(x(k_i+1)) \le V_{q_i}(x(k_i)),\ i = 1,2,\ldots$
  - Can find switching sequence automatically.

X. Koutsoukos, P. Antsaklis, "Design of Stabilizing Switching Control Laws for Discrete and Continuous-Time Linear Systems Using Piecewise-Linear Lyapunov Functions", *International Journal Control*, 75(12), 932-945, 2002

Systems with Discontinuities

- State machines
- Hybrid systems
  - Timed automata
  - Switched linear systems
  - Markov jump linear systems

Markov Jump Linear Systems

$$x(k+1) = A_q x(k) + B_q u(k)$$
$$\Pr(q(k+1) = j \mid q(k) = i\} = p_{ij}(k), \quad 1 \le i, j \le N$$

- Stochastic jumps
- Use second momentum as Lyapunov function:
$$M = \mathbf{E}(x^{\mathrm{T}} x)$$
- Can show that the system is asymptotically stable if there exists real, positive definite matrices $Q_1, \ldots Q_{\mathrm{N}}$, s.t.

$$\sum_{j=1}^{N} p_{ji} A_j Q_j A_j^T < Q_i, \text{ for all } i$$

Summary

classes	Hybrid systems	Timed automata	Switched linear sys	Jump linear sys
Same flow variables in every location	No	No	Yes	Yes
Guards on transitions	$f(x)$	$x < c$	$f(\mathrm{x})$	p_{ij}
Reset flow variables on transitions	Yes	Yes	No	No