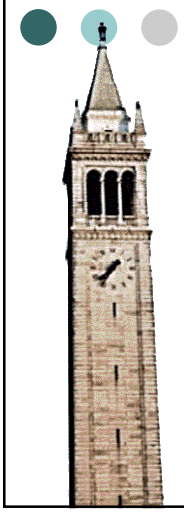


Introduction to Embedded Systems

Edward A. Lee & Sanjit A. Seshia
UC Berkeley
EECS 124
Spring 2008

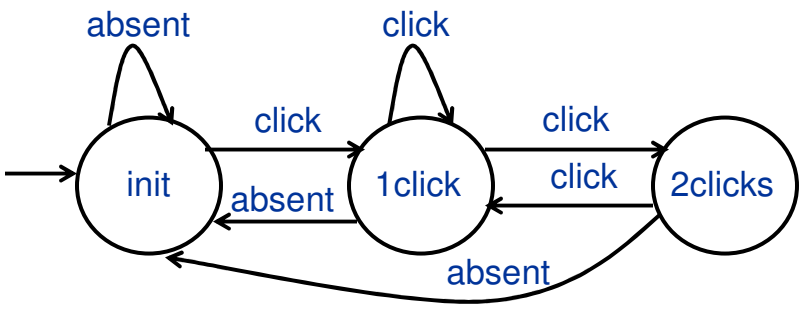
Lecture 6: Hybrid Systems, Part II

Material drawn from notes by R. Alur, P. Bouyer, C. Tomlin

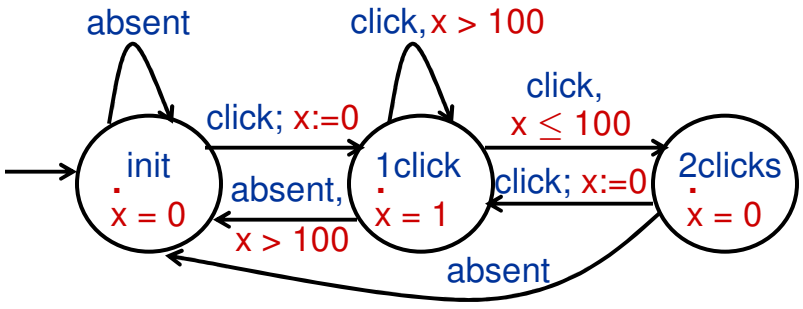
Two Topics in this Lecture

- Timed Automata
  - sub-class of Hybrid Automata useful for modeling real-time systems
- Analysis of Continuous Behavior by Discretization
  - from Timed Automaton, construct a bisimilar FSM

Example: Capturing a “Double-Click” with a FSM



Example: Capturing a “Double-Click” with a Timed Automaton



Recall: Formal Representation of Hybrid Automaton

A hybrid automaton is a tuple: $(Q, X, \Sigma, U, Init, F, J, Inv)$

Q	finite set of modes
X	finite set of continuous state variables $\{x_1, x_2, \ldots, x_n\}$, $x_i \in \mathbb{R}$
Σ	set of discrete input symbols
U	set of continuous input signals, $\{u_1, u_2, \ldots, u_k\}$, $u_i \in \mathbb{R}$
$Init$	initial condition, $Init \subseteq Q \times \mathbb{R}^n$
F	flows, defining differential equations for each variable in each mode
J	jumps, $J : Q \times Guards \rightarrow Q \times Resets$ where an element of $Guards$ is a subset of $\Sigma \times \mathbb{R}^n \times \mathbb{R}^k$, and $Resets$ is a set of assignments of the form $x_i := expr(X, U)$
Inv	mode invariant, mapping a state to the subspace of $\mathbb{R}^n$ in which the X variables can take values

For a timed automaton: x_i's called "clock variables"

All flows are of the form $\dot{x}_i = c$, c a constant

All guards are sets of constraints of the form

$x_i \geq c$ or $x_i \leq c$, $c \in \mathbb{Q}$

All resets are of the form $x_i := 0$

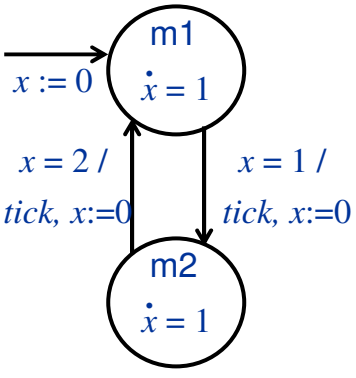
Flavors of Timed Automata

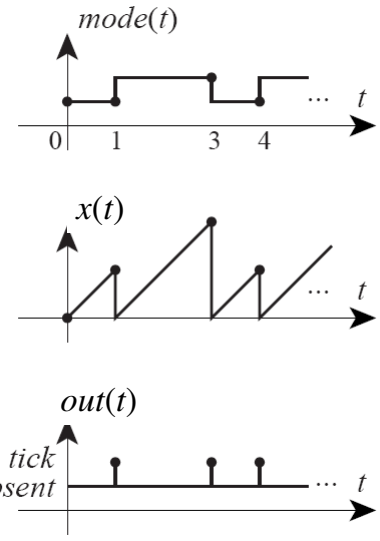
- Classic Timed Automata
  - RHS of all differential equations is 1
  - Single-speed clock that precisely tracks real time
- Multi-rate Automata
  - Can have clocks of different speeds

Applications of Timed Automata

- Real-time controllers
- Self-timed circuits
- Network protocols with timing-dependent behavior
- Scheduling of jobs

Example: A 'Tick' Generator





What does $x(t)$ look like?

Timed Traces and Time-Abstract (Untimed) Traces

time

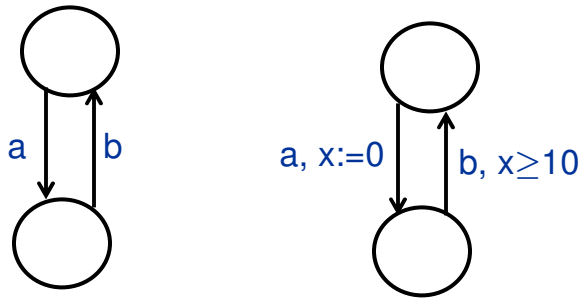
$$\begin{array}{ll} \tau_0 & (q_0, \mathbf{x}_0) \\ \tau_0' & \rightsquigarrow (q_0, \mathbf{x}_0') \\ = & \downarrow \\ \tau_1 & (q_1, \mathbf{x}_1) \\ \tau_1' & \rightsquigarrow (q_1, \mathbf{x}_1') \\ & \downarrow \\ & \vdots \\ & \downarrow \\ \tau_N & (q_N, \mathbf{x}_N) \\ \tau_N' & \rightsquigarrow (q_N, \mathbf{x}_N') \\ & \downarrow \\ & \vdots \end{array}$$

$\mathbf{x}_i' = \mathbf{x}_i + (\tau_i' - \tau_i)$

A time-abstract (untimed) trace of M is a sequence $q_0, q_1, q_2, \ldots$ that can be extended to a timed trace of M

(think of q_i's as also including input and output symbols)

Untimed vs. Timed Automata



Do these automata have the same untimed traces?

Two Problems

Verification

- Does the system do what it's supposed to do?
  - Does the system satisfy its specifications?

Synthesis/Control

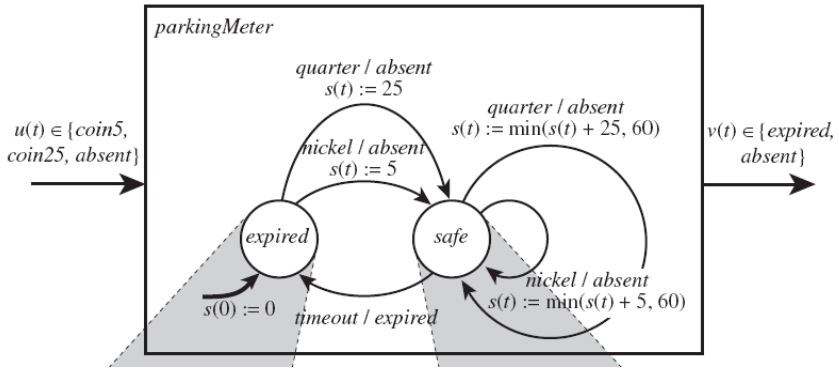
- Construct a system that satisfies its specifications
  - e.g. by synthesizing a controller

In both cases: we need to specify the objective

Untimed Specifications

For many timed systems, we are interested in specifications that do not mention time

e.g., parking meter reaches 'safe' state when coins are added



Timed Automata → FSM

Rather than analyzing the original Timed Automaton, can we construct an FSM such that they both have the same untimed behaviors?

Then untimed verification/control problems can be checked on this FSM representation

- verification/control algorithms for FSMs are more advanced than for hybrid automaton models

We will construct an FSM that is bisimilar to the original TA

Bisimulation (differences with simulation in red)

Let $M_1 = (S_1, I_1, O_1, U_1, s_{10})$ and $M_2 = (S_2, I_2, O_2, U_2, s_{20})$
where $I = I_1 = I_2$ and $O = O_1 = O_2$

We say M_1 bisimulates M_2 iff
there exists a set $R \subseteq S_1 \times S_2$ such that

1. $R(s_{10}, s_{20})$
2. For all $(s_1, s_2) \in R$, the following conditions hold:
   For all $i \in I$, and $(t_2, o_2) = U_2(s_2, i)$,
   there exists a $(t_1, o_1) = U_1(s_1, i)$ s.t.
   $(t_1, o_1) \in R$ and $o_2 = o_1$
   For all $i \in I$, and $(t_1, o_1) = U_1(s_1, i)$,
   there exists a $(t_2, o_2) = U_2(s_2, i)$ s.t.
   $(t_2, o_2) \in R$ and $o_2 = o_1$

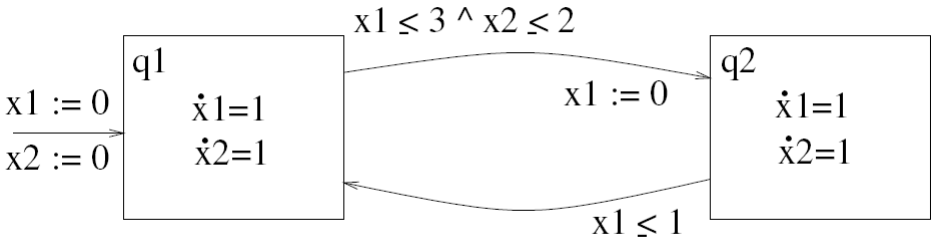
TA → FSM: Strategy

- Collect states of the TA into a finite set of regions
  - States of the FSM are these regions
- All states in a region must have the same mode
  - Hereafter focus on the continuous part of the state – "points in $\mathbb{R}^n$"
- All points in a region must *behave alike* w.r.t. flows and jumps

Conditions on Grouping Points into a Region

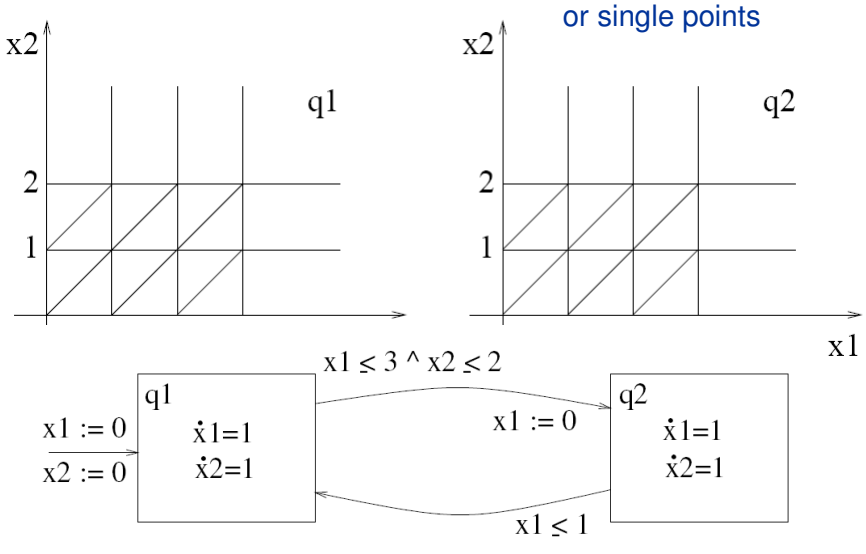
1. Enabling a Jump: A region should either be entirely contained in a guard/invariant or not at all
2. Resets: When a clock reset is performed from all points Region R, the resulting projected region R' must lie entirely within some region T
3. Flows: If some point in region R1 can reach a point in R2 by letting time elapse, then all points in R1 must be able to reach some point in R2

An Example



Regions for the Example

Open line segments, triangles, rectangles, or single points



A Region is an Equivalence Class of Points

1. For all $x_i \in X$, define c_i to be the largest constant that x_i is compared with in a guard or invariant
2. For all valuations v1, v2: $v1(x) \sim v2(x)$ iff
   1. For all $x_i \in X$,
      either: $v1(x_i) \geq c_i$ and $v2(x_i) \geq c_i$
      or: $[v1(x_i)] = [v2(x_i)]$
   2. For all $x_i, x_j \in X$ where $v1(x_i) \leq c_i$ and $v2(x_i) \leq c_i$, $fr(v1(x_i)) \leq fr(v1(x_j))$ iff $fr(v2(x_i)) \leq fr(v2(x_j))$
   3. For all $x_i \in X$ such that $v1(x_i) \leq c_i$, $fr(v1(x_i)) = 0$ iff $fr(v2(x_i)) = 0$

- $\sim$ is an equivalence relation (check this later)

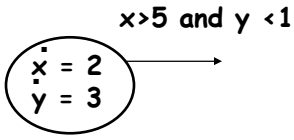
Summary: Timed Automaton → FSM

1. Collect constraints in guards and invariants, identify regions
2. Each region becomes a state in the FSM
3. Initial state of FSM is (initial mode, 0)
4. Let s1 = (q1, R1), s2 =(q2, R2).
   There is a transition from s1 to s2 on input symbol a if there is a jump in the TA from q1 to q2
   with a point in R1 going to a point in R2

Extension: Multi-rate Timed Automata

$\dot{x} = c$, for constant c

Does the regions graph construction work here?



Ans: Yes, just re-scale the variables to make the RHS of all differential equations = 1