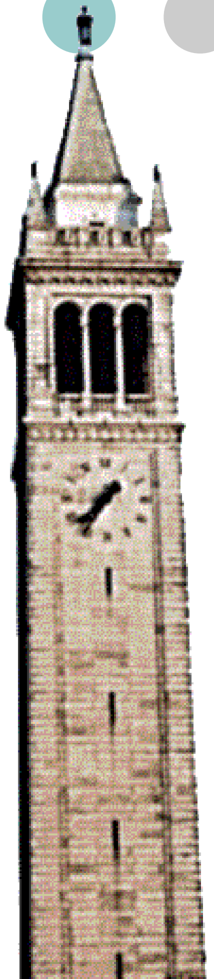
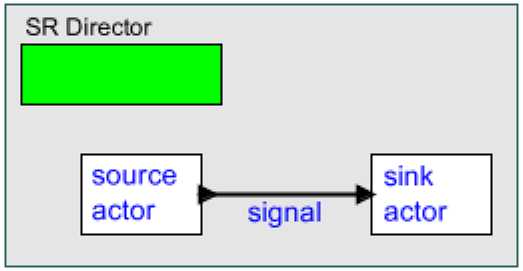


Fundamental Algorithms for System Modeling, Analysis, and Optimization

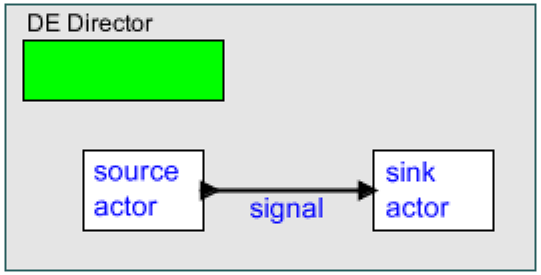
Edward A. Lee, Jaijeet Roychowdhury, Sanjit A. Seshia
UC Berkeley
EECS 144/244
Fall 2011

Lecture N+2: Discrete to Continuous Systems

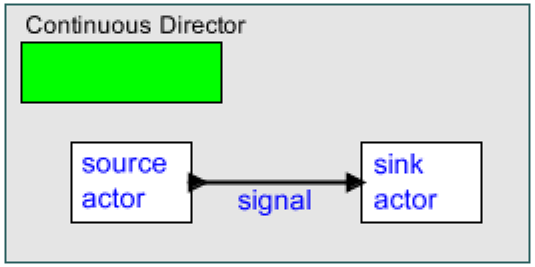
Synchronous, Discrete-Event, and Continuous-Time models in Ptolemy II



At each tick of a "clock," signals acquire values (or remain unknown, for non-constructive circuits), based on a least fixed-point semantics.



A signal is a set of events with time stamps (in model time). Components see input events in time-stamp order.



A signal is defined everywhere (in model time). A "solver" determines where (on the time line) each signal is evaluated. The value of the signal may be "absent," allowing for signals that are discrete or have gaps.

First Attempt at a Model for Time

Let $\mathbb{R}_+$ be the non-negative real numbers. Let V be an arbitrary family of values (a data type, or alphabet). Let

$$V_\varepsilon = V \cup \{\varepsilon\}$$

be the set of values plus "absent." Let s be a signal, given as a partial function:

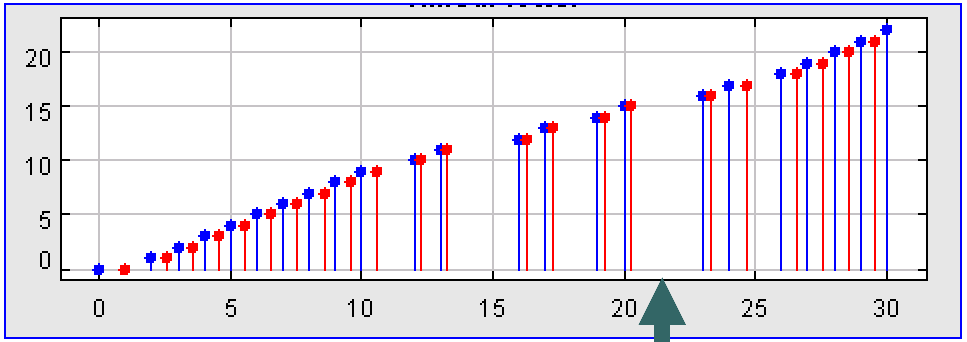
$$s : \mathbb{R}_+ \rightharpoonup V_\varepsilon$$

defined on an initial segment of $\mathbb{R}_+$

First Attempt at a Model for Time

$$s\colon \mathbb{R}_+ \rightharpoonup V_\varepsilon$$

Values V

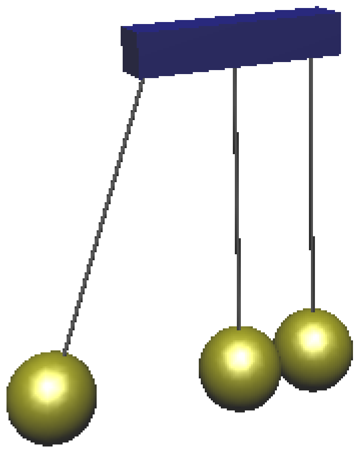


Initial segment $I \subseteq \mathbb{R}_+$ where the signal is defined.

Absent: $s(\tau) = \varepsilon$ for almost all $\tau \in I$.

This model is not rich enough because it does not allow a signal to have multiple events at the same time.

Example Motivating the Need for Simultaneous Events Within a Signal



Newton's Cradle:

Steel balls on strings

Collisions are events

Momentum of the middle ball has three values at the time of collision.

This example has continuous dynamics as well

(I will return to this)

Other examples:

- Batch arrivals at a queue.
- Software sequences abstracted as instantaneous.
- Transient states.

A Better Model for Signals: *Super-Dense Time*

Let $T = \mathbb{R}_+ \times \mathbb{N}$ be a set of “tags” where $\mathbb{N}$ is the natural numbers, and give a signal s as a partial function:

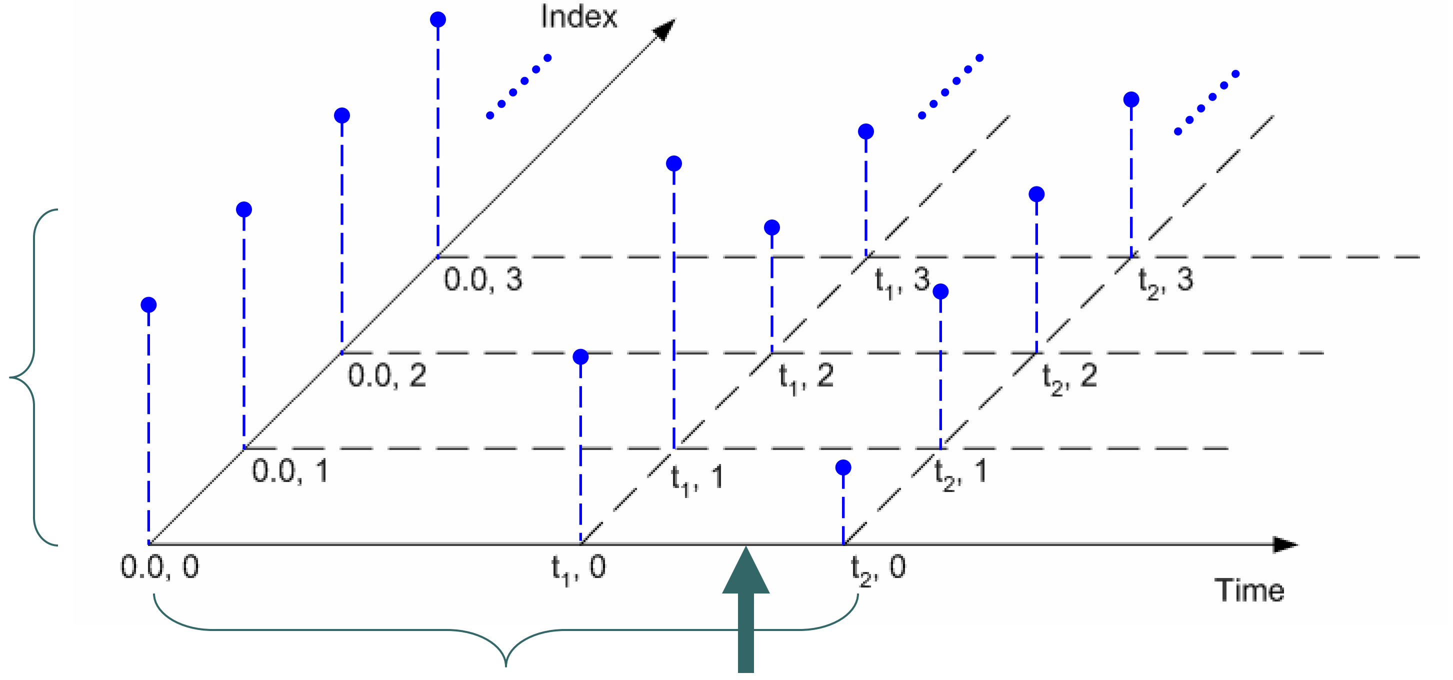
$$s : T \rightharpoonup V_\varepsilon$$

defined on an initial segment of T, assuming a lexical ordering on T:

$$(t_1, n_1) \le (t_2, n_2) \iff t_1 < t_2, \text{ or } t_1 = t_2 \text{ and } n_1 \le n_2 \,.$$

This allows signals to have a sequence of values at any real time t.

Super Dense Time



Values V

Initial segment $I \subseteq \mathbb{R}_+ \times \mathbb{N}$ where the signal is defined

Absent: $s(\tau) = \varepsilon$ for almost all $\tau \in I$.

Events and Firings

$$s : T \rightharpoonup V_\varepsilon$$

- A *tag* is a time-index pair, $\tau = (t, n) \in T = \mathbb{R}_+ \times \mathbb{N}$.
- An *event* is a tag-value pair, $e = (\tau, v) \in T \times V$.
- $s(\tau)$ is an event if $s(\tau) \neq \varepsilon$.

Operationally, events are processed by presenting all input events at a tag to an actor and then *firing* it.

However, this is not always possible!

Lee & Seshia, 2011

Probing Further: Discrete Signals

Discrete signals consist of a sequence of instantaneous events in time. Here, we make this intuitive concept precise.

Consider a signal of the form $e\colon \mathbb{R} \to \{absent\} \cup X$, where X is any set of values. This signal is a **discrete signal** if, intuitively, it is absent most of the time and we can count, in order, the times at which it is present (not absent). Each time it is present, we have a discrete event.

This ability to count the events in order is important. For example, if e is present at all rational numbers t, then we do not call this signal discrete. The times at which it is present cannot be counted in order. It is not, intuitively, a sequence of instantaneous events in time (it is a *set* of instantaneous events in time, but not a *sequence*).

To define this formally, let $T \subseteq \mathbb{R}$ be the set of times where e is present. Specifically,

$$T = \{t \in \mathbb{R} \,:\, e(t) \neq absent\}.$$

Then e is discrete if there exists a one-to-one function $f\colon T \to \mathbb{N}$ that is **order preserving**. Order preserving simply means that for all $t_1, t_2 \in T$ where $t_1 \leq t_2$, we have that $f(t_1) \leq f(t_2)$. The existence of such a one-to-one function ensures that we can count off the events *in temporal order*. Some properties of discrete signals are studied in Exercise 6.

Extending “Discrete” to Superdense Time

Just replace the reals with the set of superdense times and use the lexical ordering.

Examples

1. Suppose we have a signal s whose tag set is

$$\{(\tau, 0) \mid \tau \in R\}$$

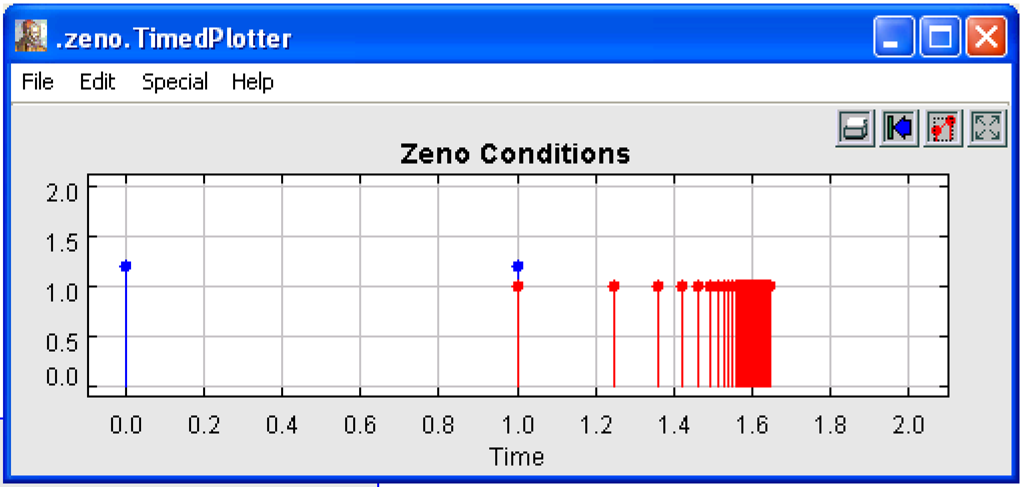
(this is a *continuous-time* signal). This signal is not discrete.

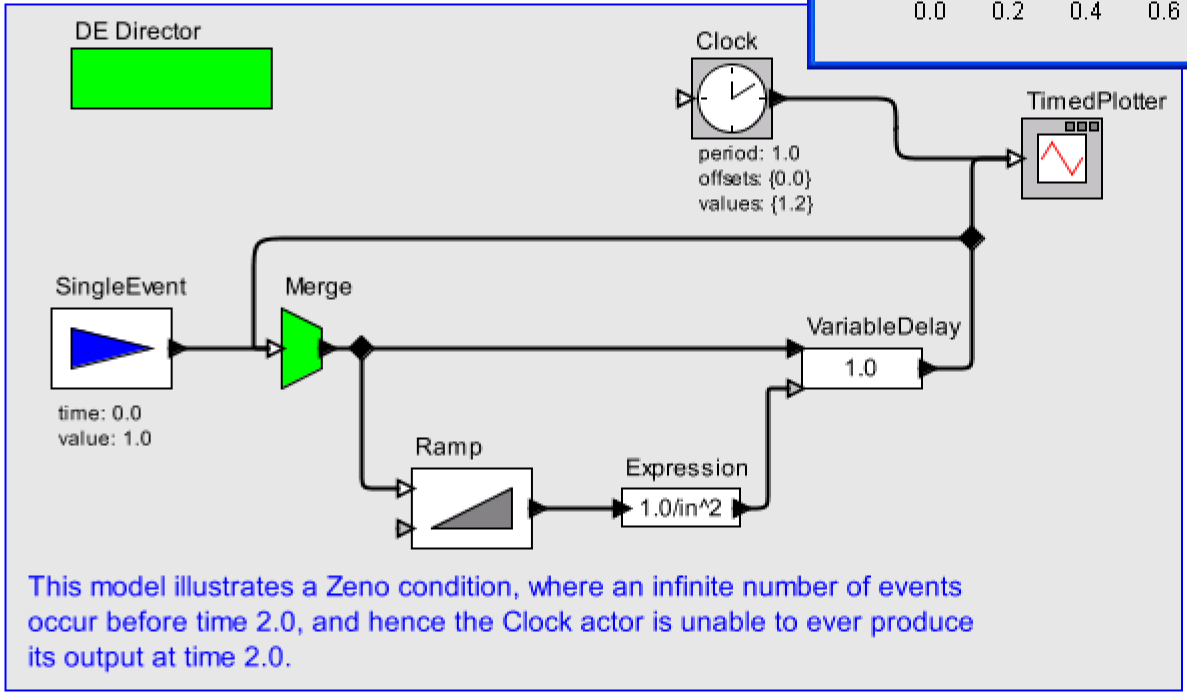
2. Suppose we have a signal s whose tag set is

$$\{(\tau, 0) \mid \tau \in Rationals\}$$

This signal is also not discrete.

A Zeno system is not discrete.



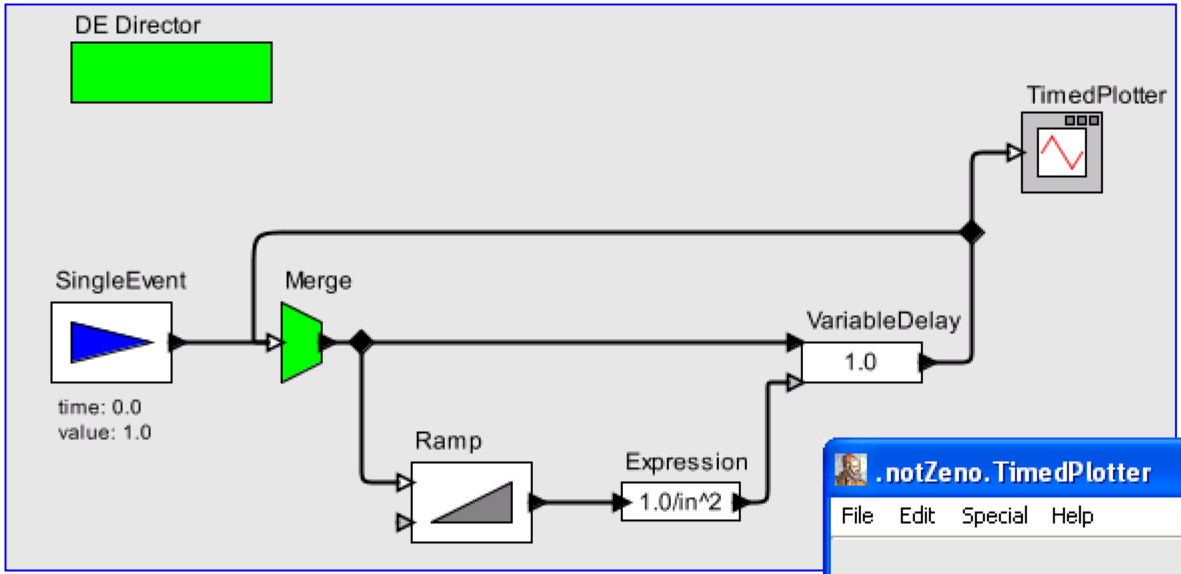


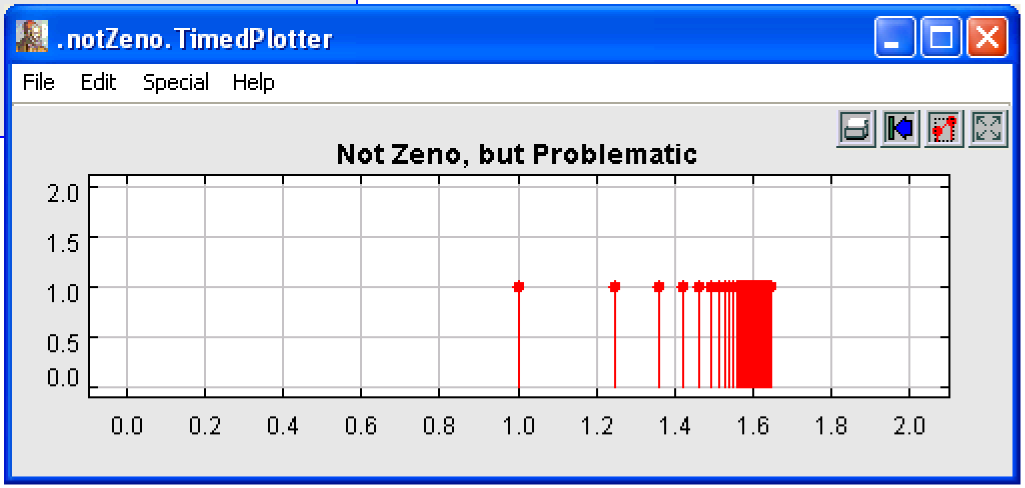
This model illustrates a Zeno condition, where an infinite number of events occur before time 2.0, and hence the Clock actor is unable to ever produce its output at time 2.0.

The tag set here includes $\{ 0, 1, 2, \ldots\}$ and $\{ 1, 1.25, 1.36, 1.42, \ldots\}$.

Exercise: Prove that this system is not discrete.

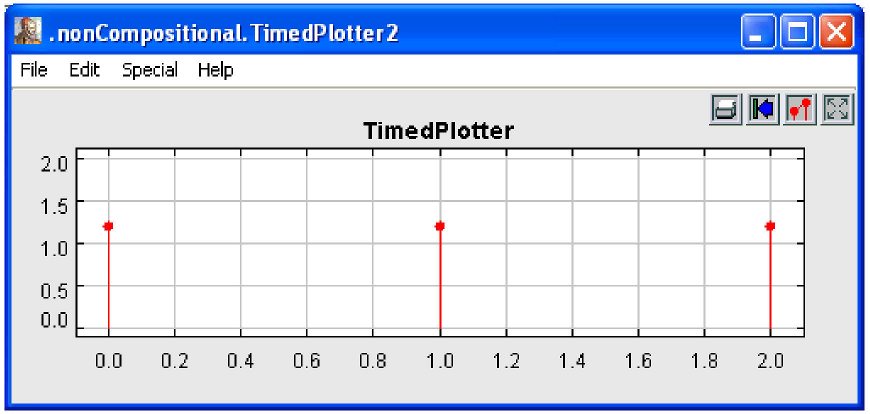
Is the following system discrete?

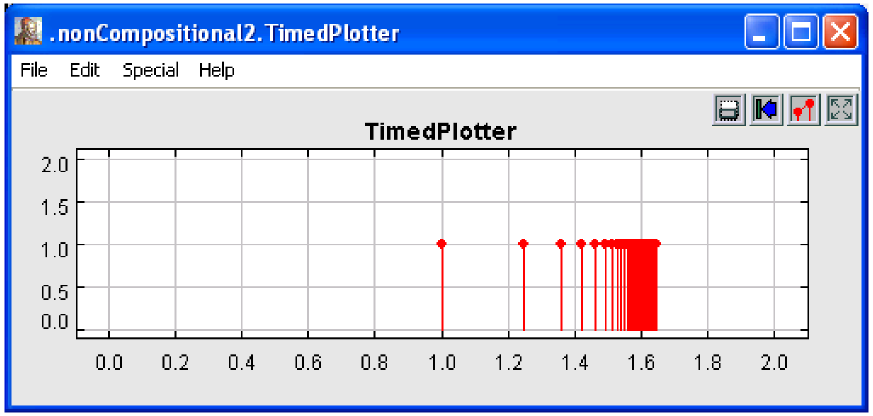




Discreteness is Not a Compositional Property

Given two discrete signals s, s' it is not necessarily true that $S = \{ s, s' \}$ is a discrete system.

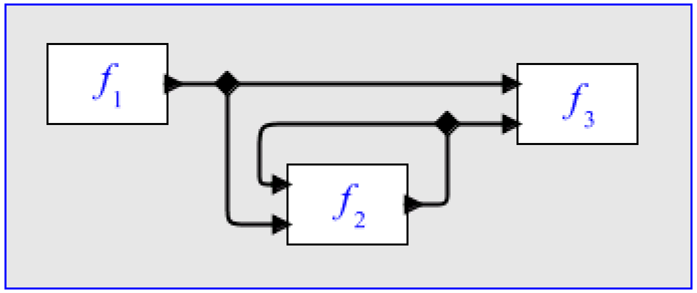




Putting these two signals in the same model creates a Zeno condition.

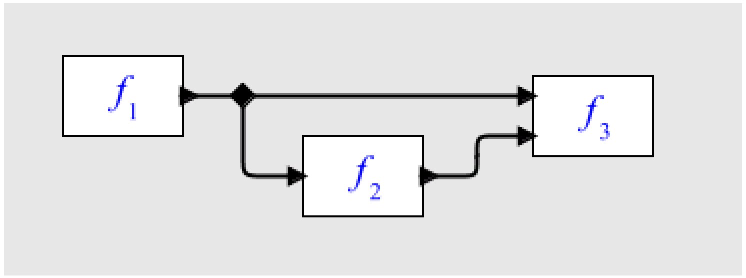
Question 1:

Can we find necessary and/or sufficient conditions to avoid Zeno systems?



Question 2:

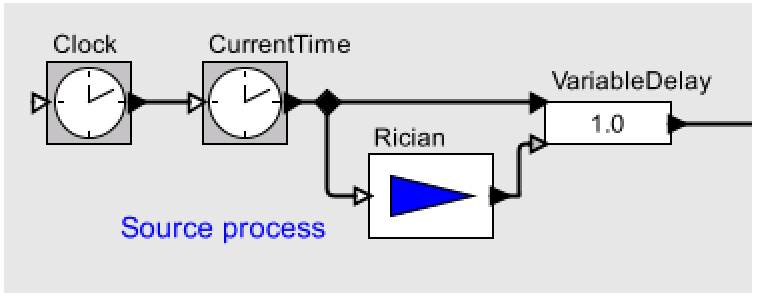
In the following model, if f_2 has no delay, should f_3 see two simultaneous input events with the same tag? Should it react to them at once, or separately?



In Verilog, it is nondeterministic. In VHDL, it sees a sequence of two distinct events separated by “delta time” and reacts twice, once to each input. In the Ptolemy II DE domain, it sees the events together and reacts once.

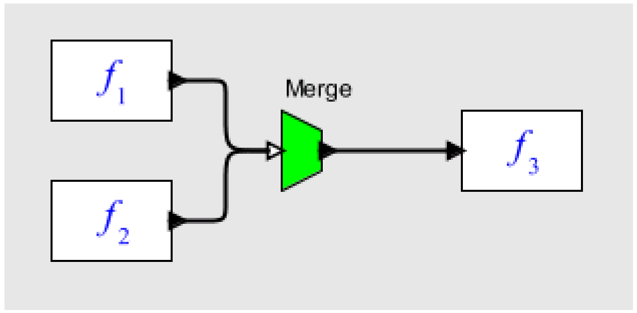
Example

In the following segment of a model, clearly we wish that the VariableDelay see the output of Rician when it processes an input from CurrentTime.



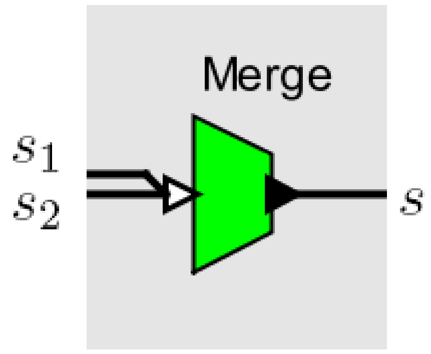
Question 3:

What if the two sources in the following model deliver an event with the same tag? Can the output signal have distinct events with the same tag?



Recall that we require that a signal be a partial function $s : T \to V$, where V is a set of possible event values (a data type), and T is a totally ordered set of *tags*.

One Possible Semantics for DE Merge



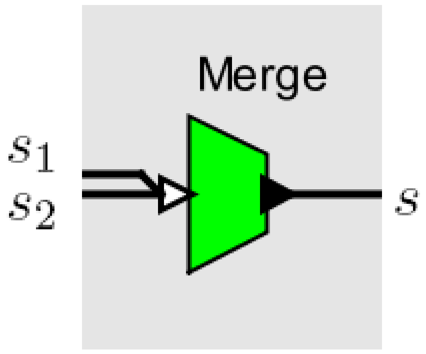
At time t, input sequences are interleaved. That is, if the inputs are s_1 and s_2 and

$$s_1(t,0) = v_1,$$
$$s_2(t,0) = w_1, \quad s_1(t,1) = w_2$$

(otherwise absent) then the output s is

$$s(t,0) = v_1, \quad s(t,1) = w_1, \quad s(t,2) = w_2.$$

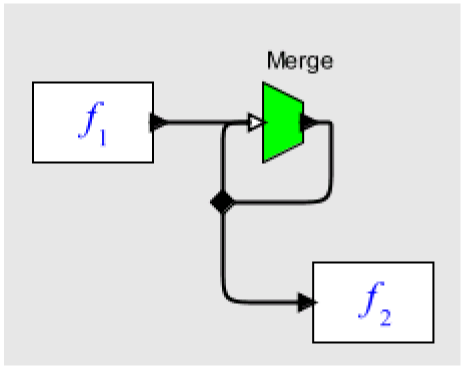
Implementation of DE Merge



```
private List pendingEvents;
fire() {
  foreach input s {
    if (s is present) {
      pendingEvents.append(event from s);
    }
  }
  if (pendingEvents has events) {
    send to output (pendingEvents.first);
    pendingEvents.removeFirst();
  }
  if (pendingEvents has events) {
    post event at the next index on the event queue;
  }
}
```

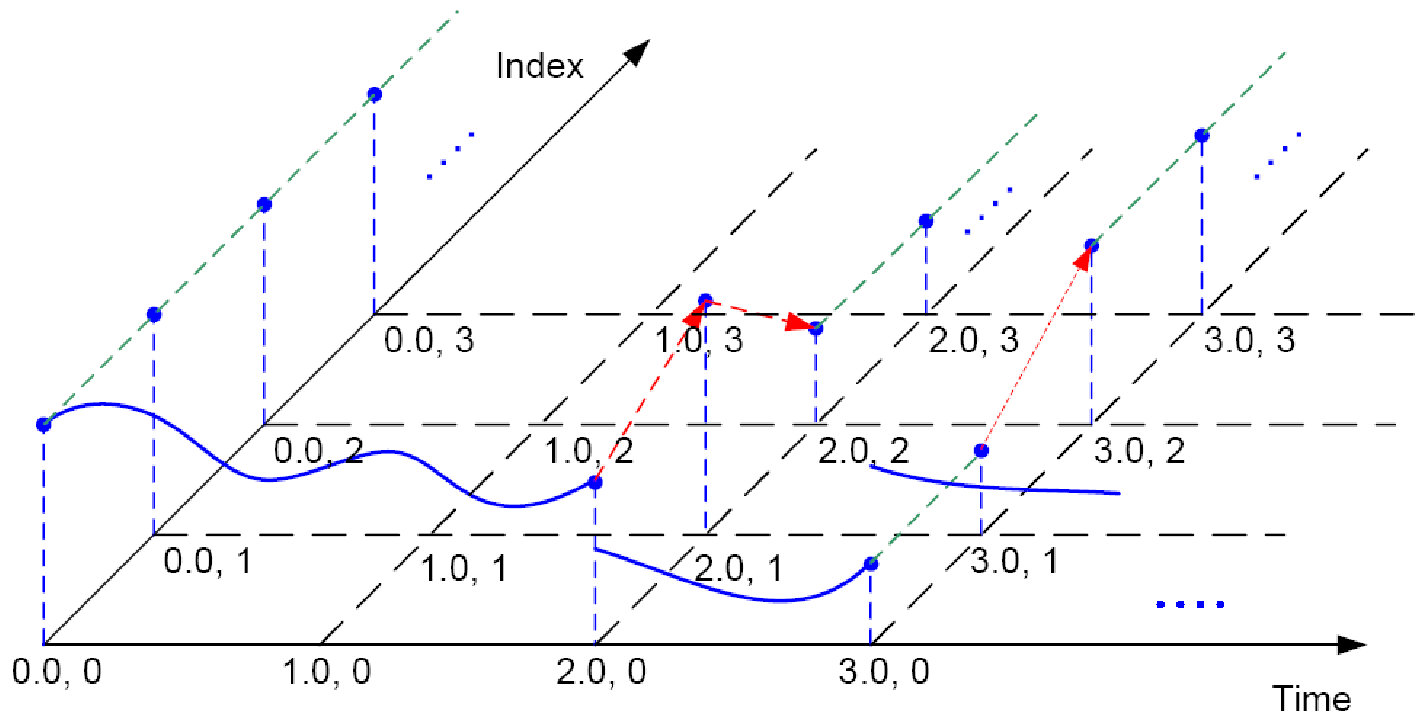
Question 4:

What does this mean?



The Merge presumably does not introduce delay, so what is the meaning of this model?

Superdense Time for Continuous-Time Signals



At each tag, the signal has exactly one value. At each time point, the signal has an infinite number of values. The red arrows indicate value changes between tags, which correspond to discontinuities.

Initial and Final Value Signals

A signal $x : T \times \mathbb{N} \to V$ has no *chattering Zeno* condition if there is an integer $m > 0$ such that

$$\forall n > m, \quad x(t,n) = x(t,m)$$

A non-chattering signal has a corresponding *final value signal*, $x_f : T \to V$ where

$$\forall\, t \in T, \quad x_f(t) = x(t,m)$$

It also has an *initial value signal* $x_i : T \to V$ where

$$\forall\, t \in T, \quad x_i(t) = x(t,0)$$

Piecewise Continuous Signals

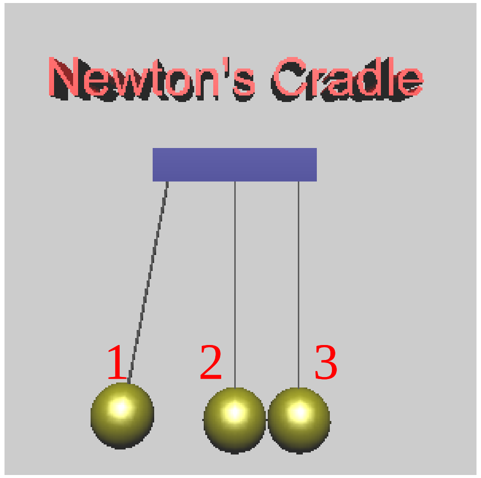
A piecewise continuous signal is a non-chattering signal

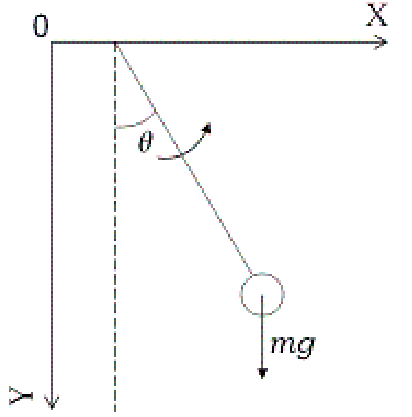
$$x : T \times \mathbb{N} \to V$$

where

- The initial signal x_i is continuous on the left,
- The final signal x_f is continuous on the right, and
- The signal x has only one value at all $t \in T \setminus D$ where $D \subset T$ is a discrete set.

Example: Newton's Cradle



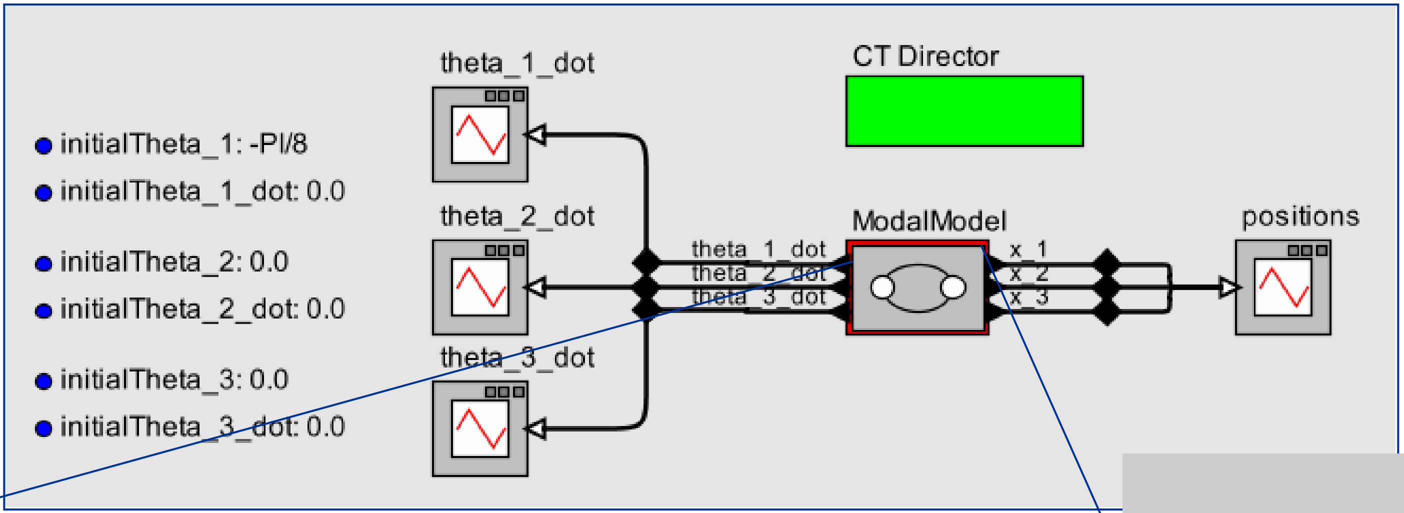


$$ml\ddot{\theta} = -mg\sin(\theta)$$

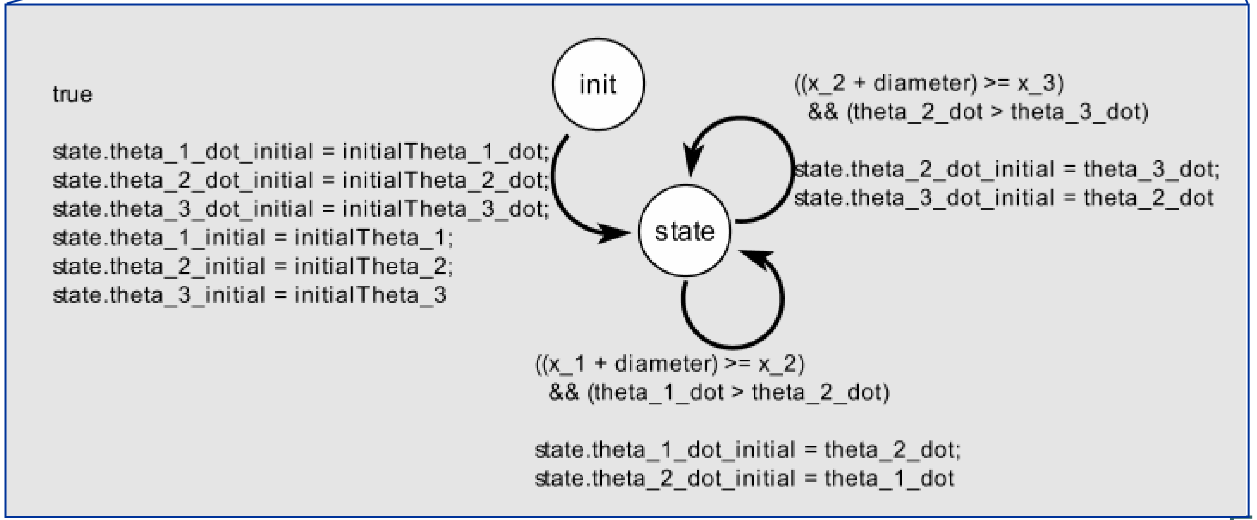
Assumptions

- Ideal pendulum
- Balls have the same mass.
- Collisions happen instantaneously.
- When a collision happens, two and only two balls are involved.

A Model of Newton's Cradle



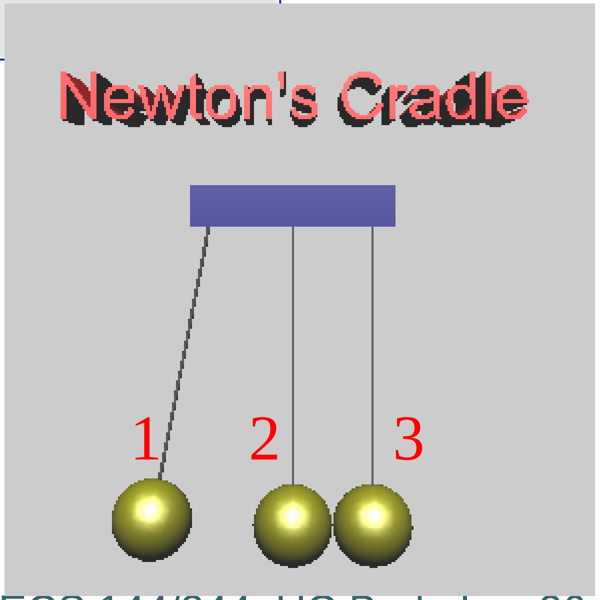
- initialTheta_1: -PI/8
- initialTheta_1_dot: 0.0
- initialTheta_2: 0.0
- initialTheta_2_dot: 0.0
- initialTheta_3: 0.0
- initialTheta_3_dot: 0.0



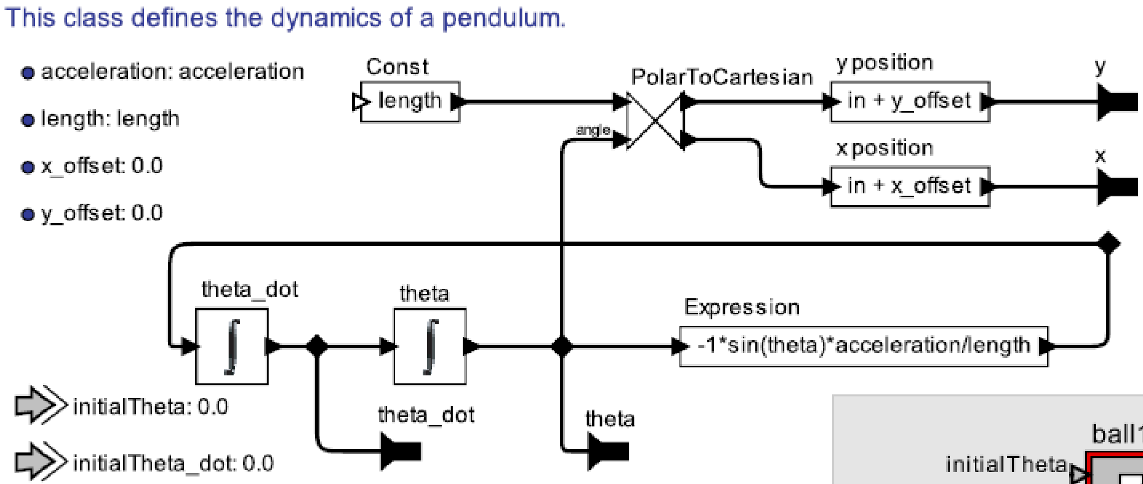
```
true

state.theta_1_dot_initial = initialTheta_1_dot;
state.theta_2_dot_initial = initialTheta_2_dot;
state.theta_3_dot_initial = initialTheta_3_dot;
state.theta_1_initial = initialTheta_1;
state.theta_2_initial = initialTheta_2;
state.theta_3_initial = initialTheta_3
```

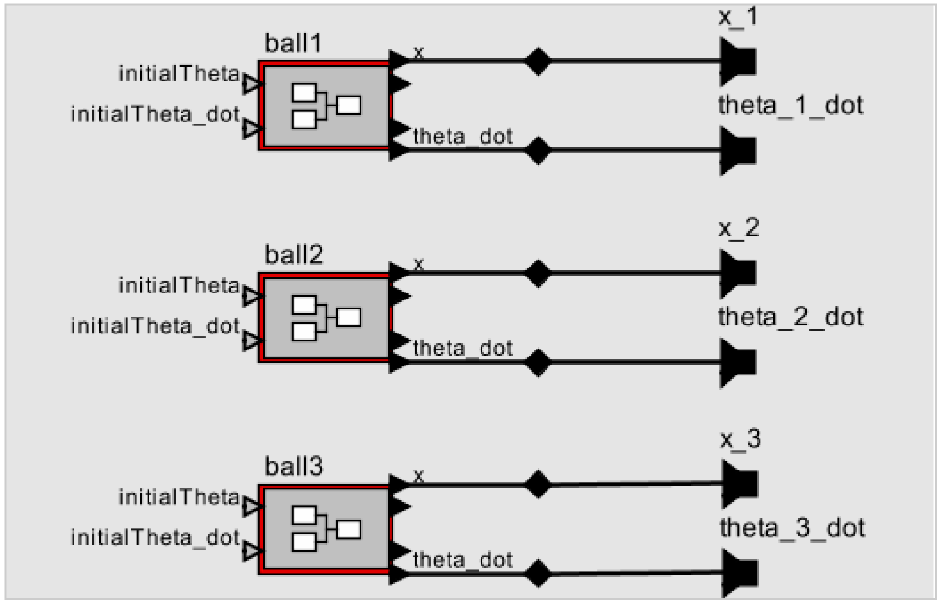
```
((x_2 + diameter) >= x_3)
  && (theta_2_dot > theta_3_dot)

state.theta_2_dot_initial = theta_3_dot;
state.theta_3_dot_initial = theta_2_dot
```

```
((x_1 + diameter) >= x_2)
  && (theta_1_dot > theta_2_dot)

state.theta_1_dot_initial = theta_2_dot;
state.theta_2_dot_initial = theta_1_dot
```



Dynamics of Balls





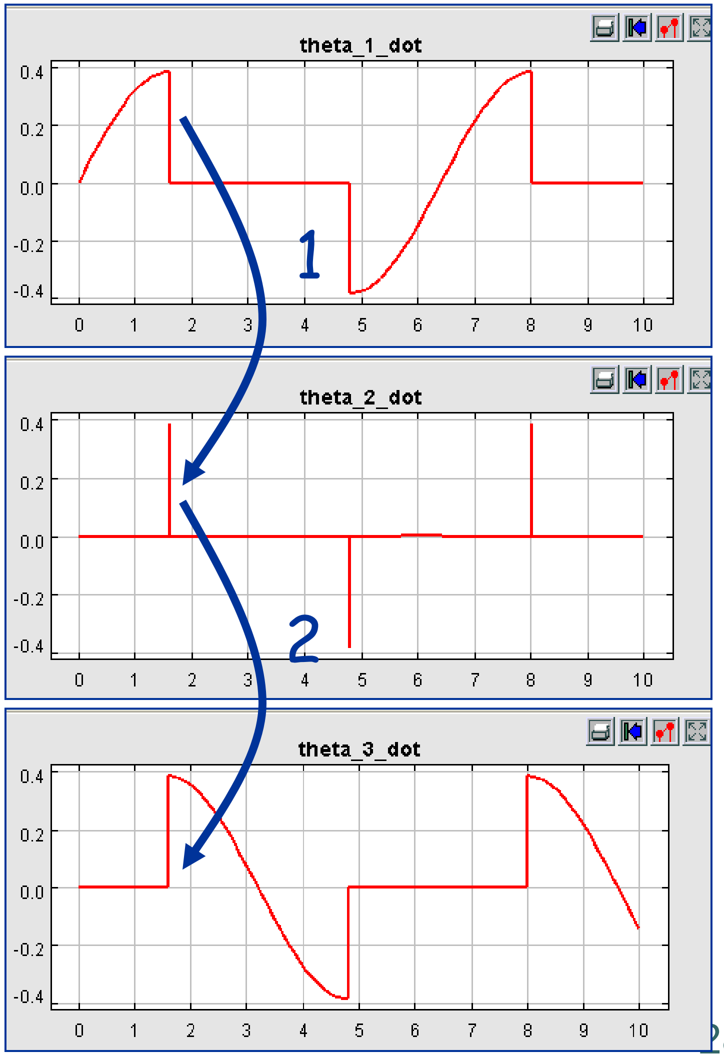
Three second order ODE’s are used to model the dynamics of three pendulums.

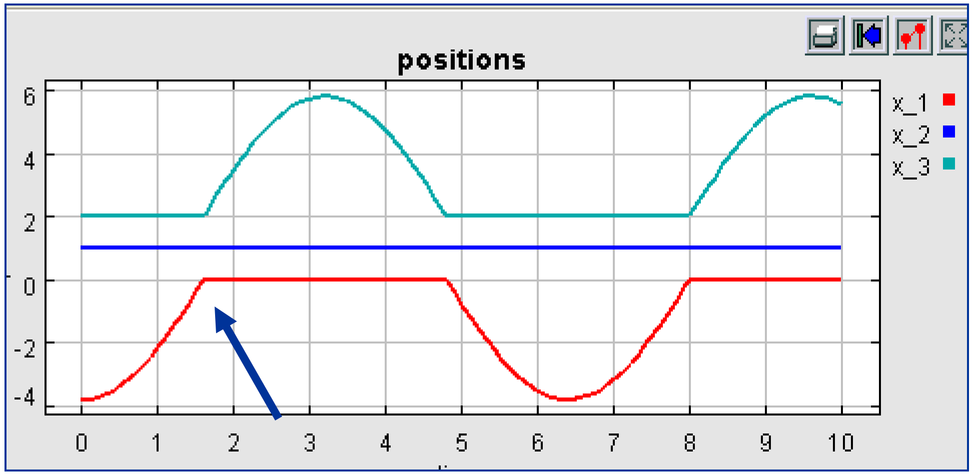
One Behavior

Ball #1 is moved away from its equilibrium position with angle PI/8.

Perfectly elastic collisions.

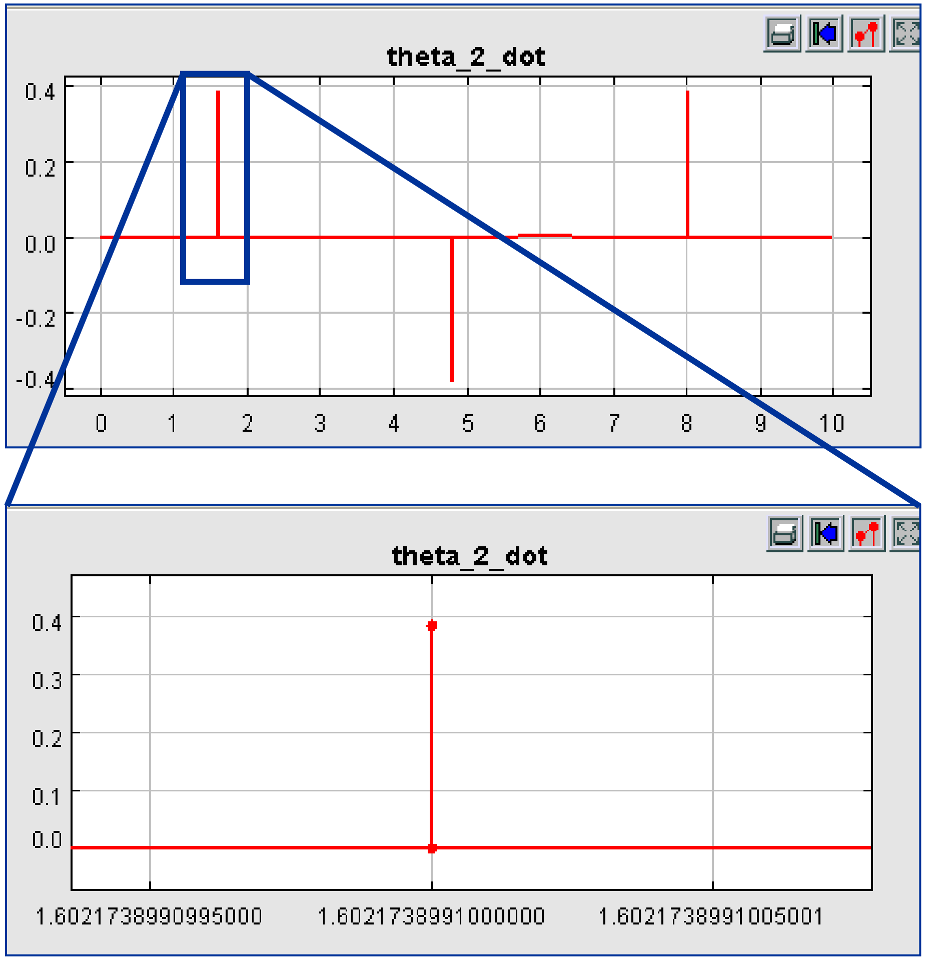
X-axis is time and Y-axis is velocity.





X-axis is time and Y-axis is displacement.

Interactions Between CT and DE Dynamics



Two transitions at the same time, called simultaneous discrete events.

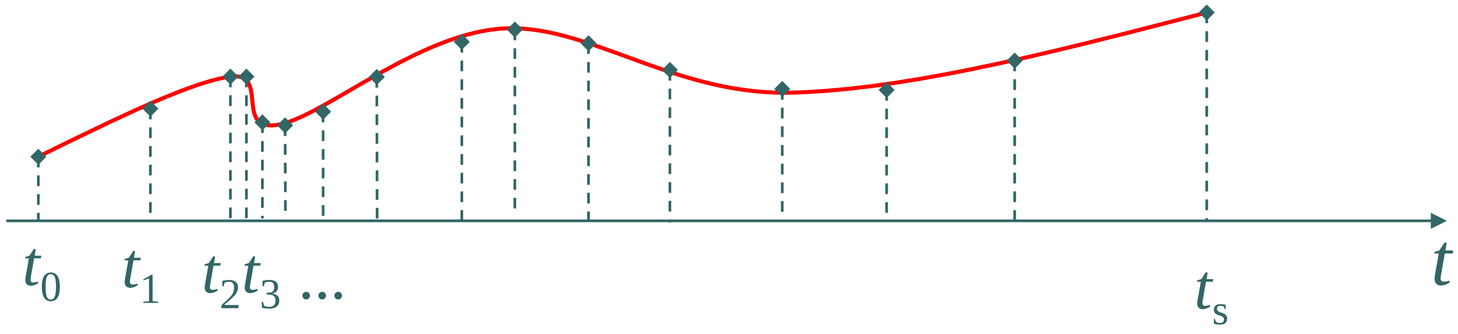
These events cause a discontinuity consisting of three values.

Agreement on the assumption of instantaneous collisions

ODE Solvers

Numerical solution approximates the state trajectory of the ODE by estimating its value at discrete time points:

$$\{t_0, t_1, \ldots\} \subset T$$



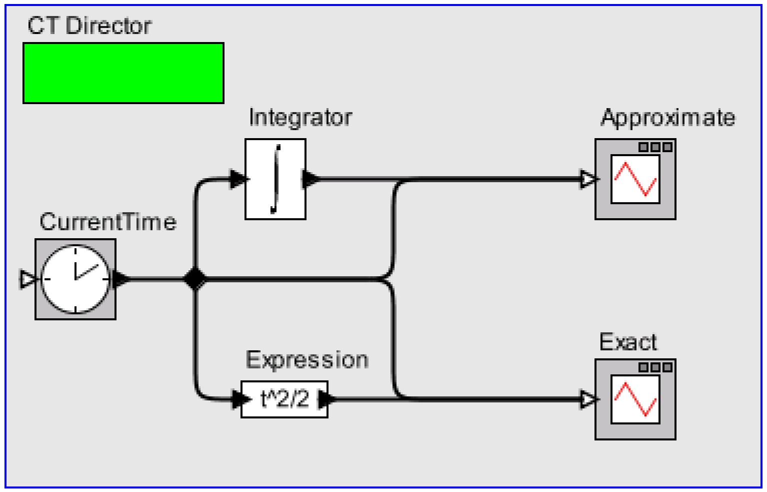
Reasonable choices for these points depend on the function *f*.

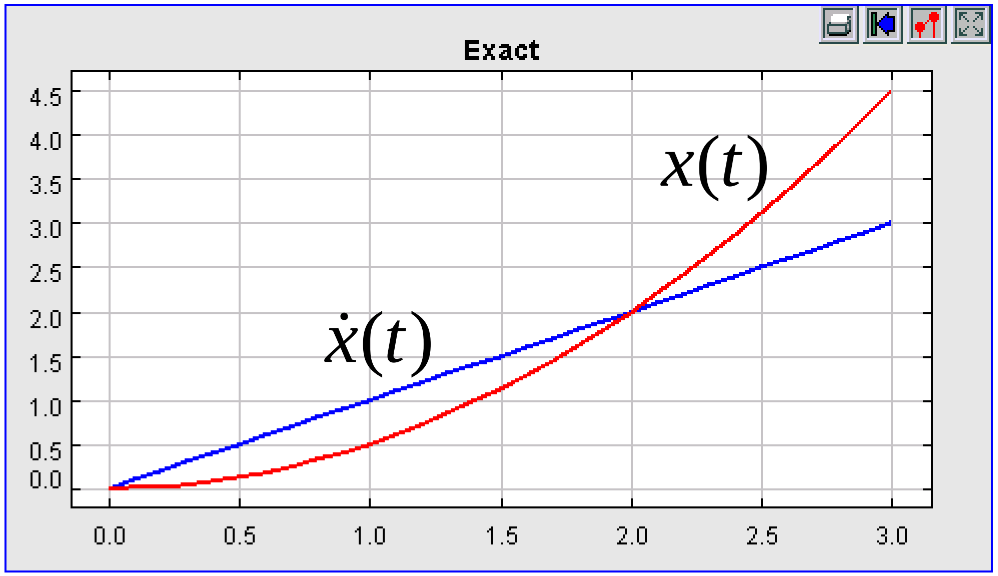
Using such solvers, signals are discrete-event signals.

Simple Example

This simple example integrates a ramp, generated by the CurrentTime actor. In this case, it is easy to find a closed form solution,

$$\dot{x}(t) = t \;\Rightarrow\; x(t) = t^2/2$$

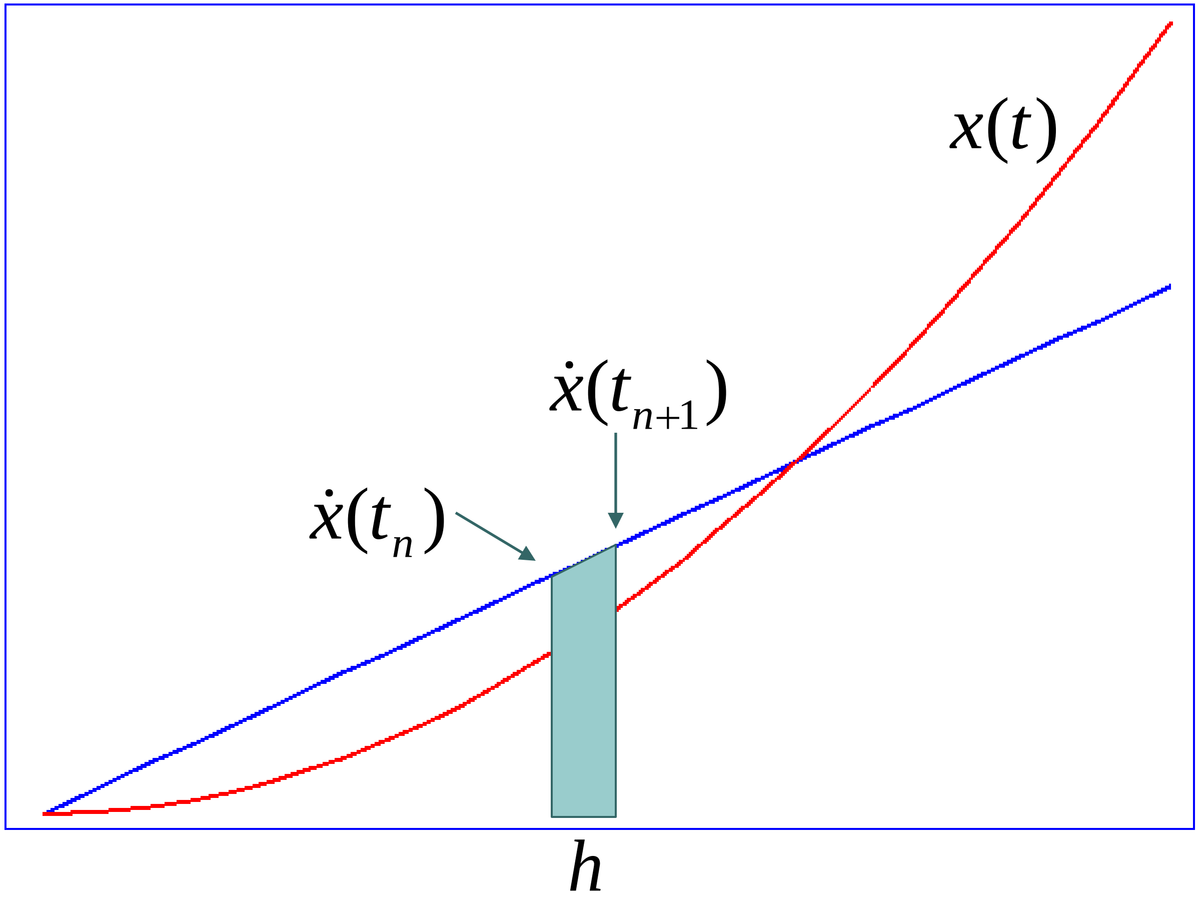




Trapezoidal Method

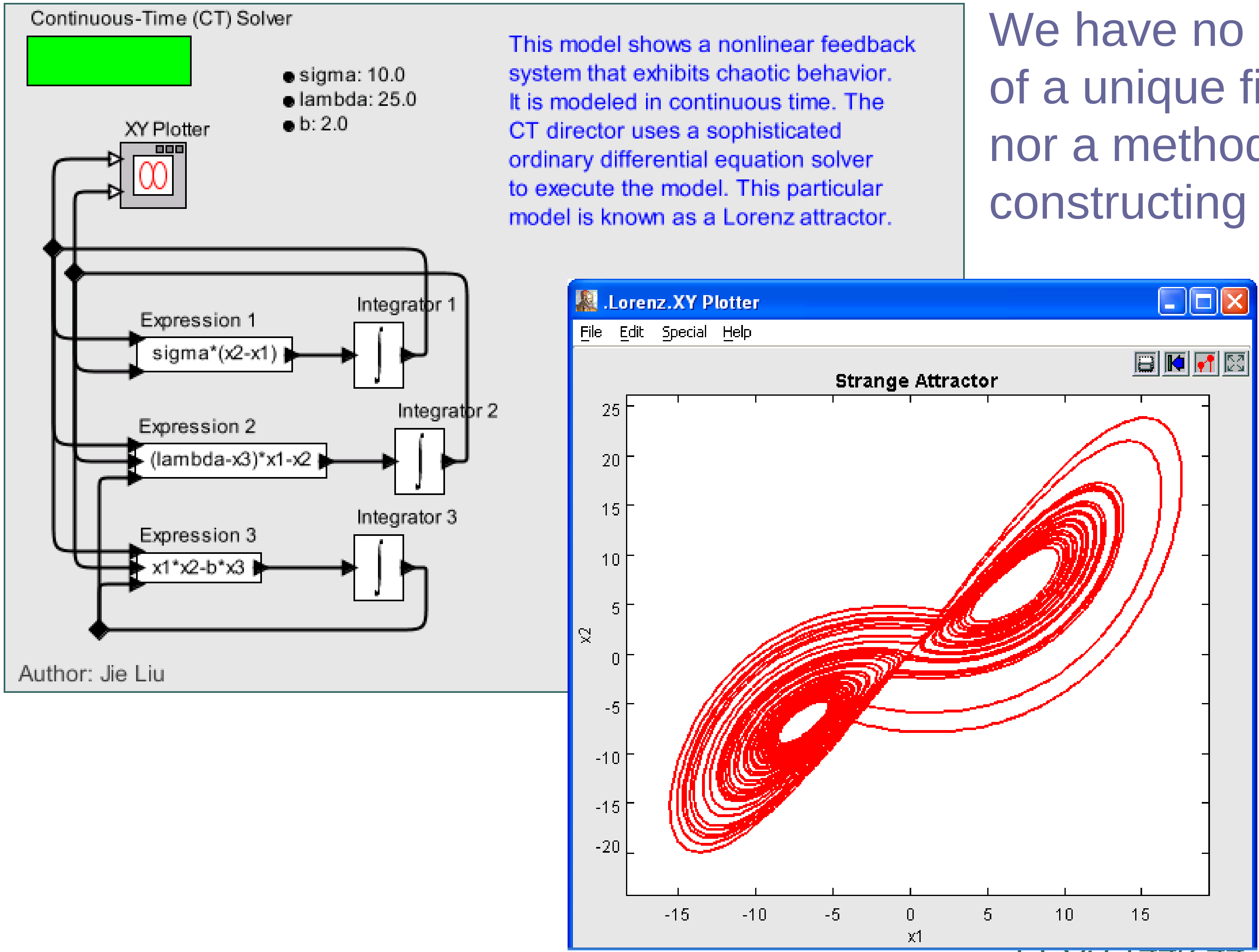
Classical method estimates the area under the curve by calculating the area of trapezoids.

However, with this method, an integrator is only causal, not strictly causal or delta causal.



$$x(t_{n+1}) = x(t_n) + h(\dot{x}(t_n) + \dot{x}(t_{n+1}))/2$$

Trapezoidal Method is Problematic with Feedback



This model shows a nonlinear feedback system that exhibits chaotic behavior. It is modeled in continuous time. The CT director uses a sophisticated ordinary differential equation solver to execute the model. This particular model is known as a Lorenz attractor.

Author: Jie Liu

We have no assurance of a unique fixed point, nor a method for constructing it.

Forward Euler Solver

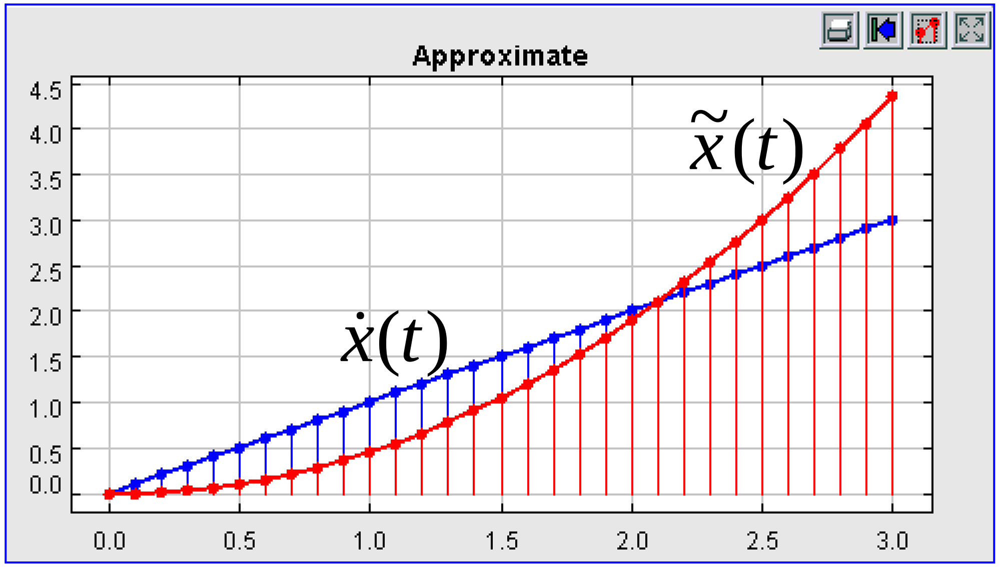
Given $x(t_n)$ and a time increment h, calculate:

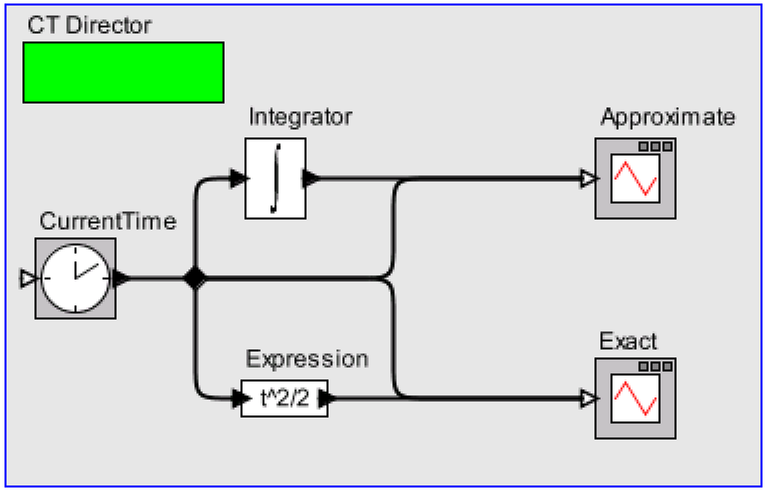
$$t_{n+1} = t_n + h$$
$$x(t_{n+1}) = x(t_n) + h\, f(x(t_n), t_n)$$

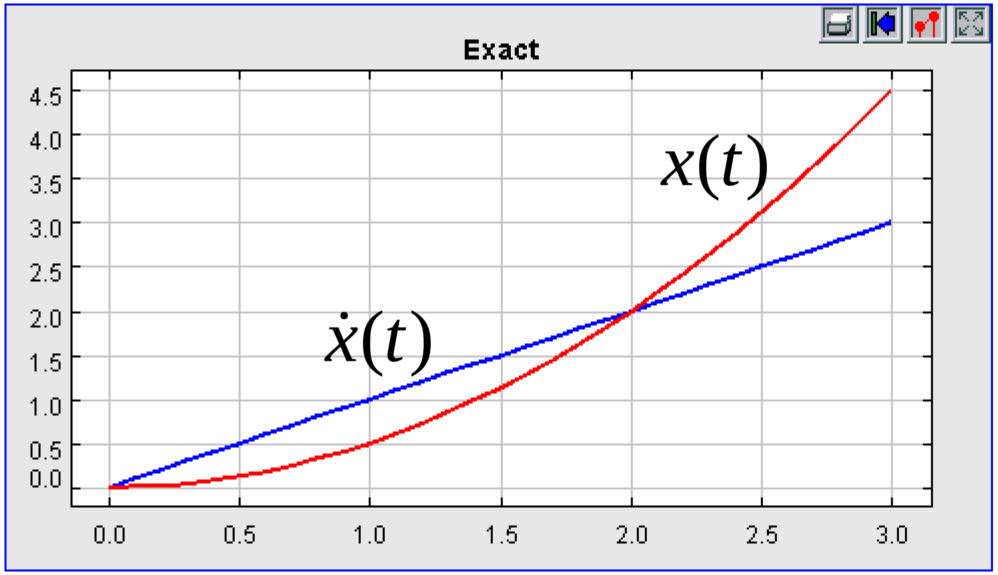
This method is strictly causal, or, with a lower bound on the step size h, delta causal. It can be used in feedback systems. The solution is unique and non-Zeno.

Forward Euler on Simple Example

In this case, we have used a fixed step size $h = 0.1$. The result is close, but diverges over time.







"Stiff" systems require small step sizes

Force due to spring extension:

$$F_1(t) = k(p - x(t))$$

Force due to viscous damping:

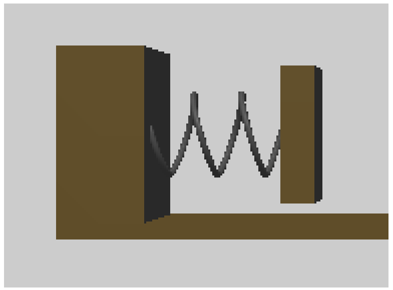
$$F_2(t) = -c\dot{x}(t)$$

Newton's second law:

$$F_1(t) + F_2(t) = M\ddot{x}(t)$$

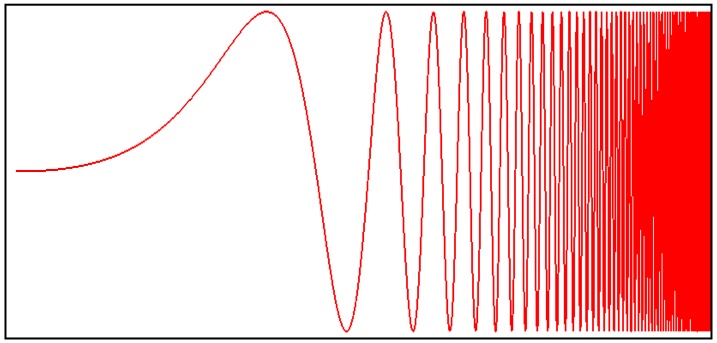
or

$$M\ddot{x}(t) + c\dot{x}(t) + kx(t) = kp.$$



For spring-mass damper, large stiffness constant *k* makes the system "stiff."

Variable step-size methods will dynamically modify the step size *h* in response to estimates of the integration error. Even these, however, run into trouble when stiffness varies over time. Extreme case of increasing stiffness results in Zeno behavior:



Runge-Kutta 2-3 Solver (RK2-3)

Given $x(t_n)$ and a time increment h, calculate

$$K_0 = f(x(t_n), t_n) \longleftarrow \dot{x}(t_n)$$

$$K_1 = f(x(t_n) + 0.5hK_0, t_n + 0.5h) \longleftarrow \text{estimate of } \dot{x}(t_n + 0.5h)$$

$$K_2 = f(x(t_n) + 0.75hK_1, t_n + 0.75h) \longleftarrow \text{estimate of } \dot{x}(t_n + 0.75h)$$

then let

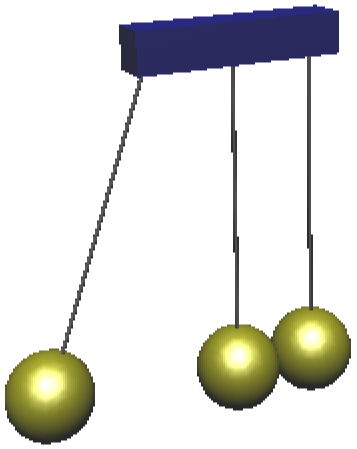
$$t_{n+1} = t_n + h$$

$$x(t_{n+1}) = x(t_n) + (2/9)hK_0 + (3/9)hK_1 + (4/9)hK_2$$

Note that this is strictly (delta) causal, but requires three evaluations of f at three different times with three different inputs.

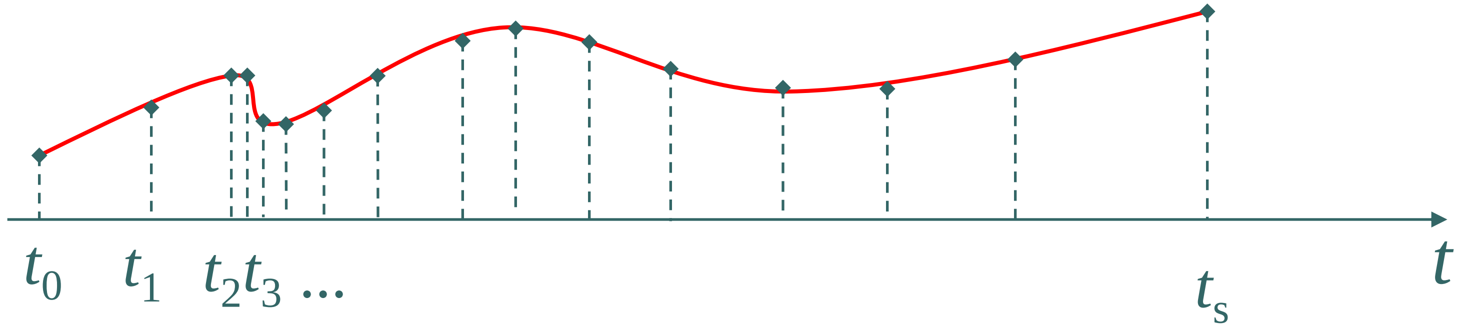
Ideal Solver Semantics for Continuous-Time Systems

[Liu and Lee, HSCC 2003]



In the *ideal solver semantics,* an ODE governing the hybrid system has a unique solution for intervals $[t_i\,,\,t_{i+1})$, the interval between discrete time points. A discrete trace loses nothing by not representing values within these intervals.

This elaborates our DE models only by requiring that an ODE solver be consulted when advancing time.



Ideal Solver Semantics

[Liu and Lee, HSCC 2003]

Given an interval $I = [t_i, t_{i+1}]$ and an initial value $x(t_i)$ and a function $f : R^m \times T \to R^m$ that is Lipschitz in x on the interval (meaning that there exists an $L \geq 0$ such that

$$\forall t \in I, \quad \left\| f(x(t),t) - f(x'(t),t) \right\| \leq L \left\| x(t) - x'(t) \right\|$$

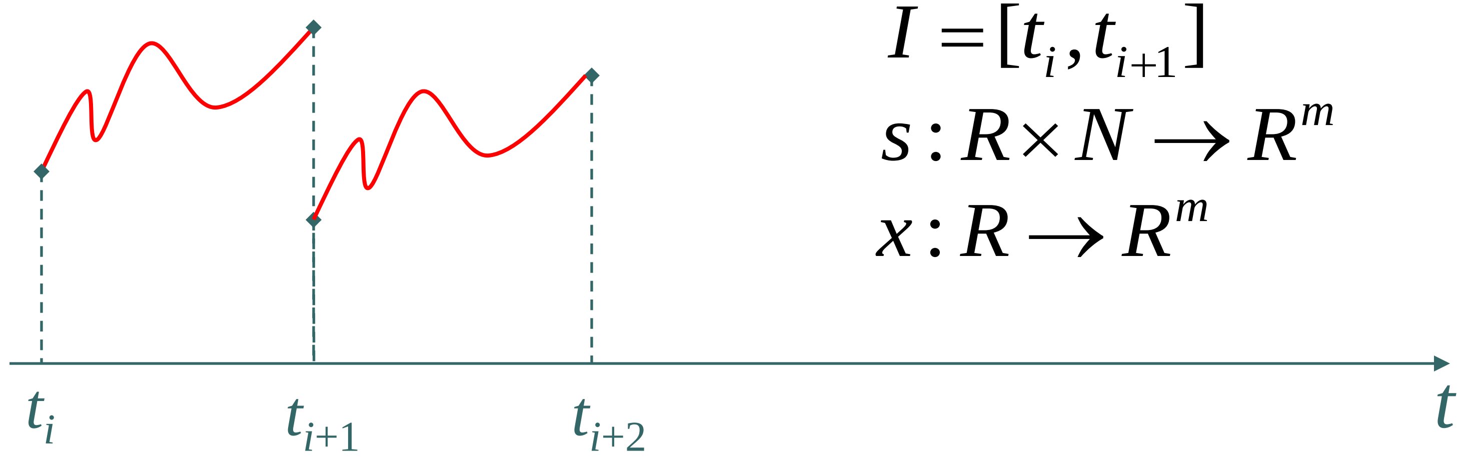
then the following equation has a unique solution x satisfying the initial condition where

$$\forall t \in I, \quad \dot{x}(t) = f(x(t),t)$$

The ideal solver yields the exact value of $x(t_{i+1})$.

Piecewise Lipschitz Systems

In our CT semantics, signals have multiple values at the times of discontinuities. Between discontinuities, a necessary condition that we can impose is that the function f be Lipschitz, where we choose the points at the discontinuities to ensure this:



$$I = [t_i, t_{i+1}]$$
$$s : R \times N \to R^m$$
$$x : R \to R^m$$

Conclusions

- *Discrete-event* models compose components that communicate timed *events*. They are widely used for simulation (of hardware, networks, and complex systems).
- *Superdense time* uses tags that have a real-valued *time-stamp* and a natural number *index*, thus supporting sequences of causally-related simultaneous events.
- A *discrete system* is one where the there is an order embedding from the set of tags in the system to the integers.
- Continuous-time and hybrid systems can be built using superdense time, SR-style fixed-point semantics, and an ODE solver.