Hybrid Systems

Hybrid systems and their modeling

Andreas Podelski

Freiburg University

SS 2012

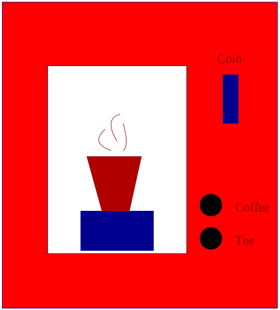
Contents

Motivation

- Dynamical system: continuous evolution of the state over time
- Time model:
  continuous $\rightsquigarrow$ $t \in \mathbb{R}$
  discrete $\rightsquigarrow$ $k \in \mathbb{Z}$
  hybrid $\rightsquigarrow$ continuous time, but there are also discrete "instants" where something "special" happens
- State model:
  continuous $\rightsquigarrow$ evolution described by *ordinary differential equations (ODEs)* $\dot{x} = f(x, u)$
  discrete $\rightsquigarrow$ evolution described by *difference equations* $x_{k+1} = f(x_k, u_k)$
  hybrid $\rightsquigarrow$ continuous space, but there are also discrete "instants" for that something "special" holds

Example: Vending machine

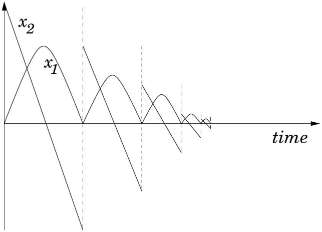
- insert coin
- choose beverage (coffee/tee)
- wait for cup
- take cup



⇝ Discrete

Example: Bouncing ball

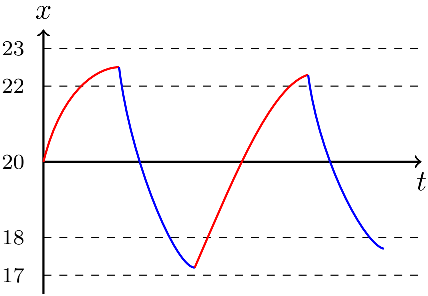
- vertical position of the ball x_1
- velocity x_2
- continuous changes of position between bounces
- discrete changes at bounce time

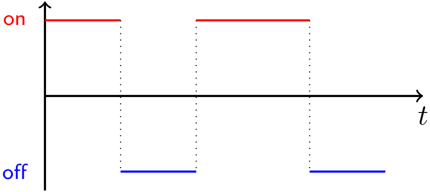


$\rightsquigarrow$ Hybrid

Example: Thermostat

- Temperature x is controlled by switching a heater on and off
- x is regulated by a thermostat:
  - $17° \leq x \leq 18°$ $\rightsquigarrow$ "heater on"
  - $22° \leq x \leq 23°$ $\rightsquigarrow$ "heater off"

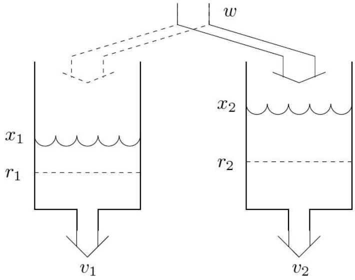




$\rightsquigarrow$ Hybrid

Example: Water tank system

- two constantly leaking tanks v_1 and v_2
- hose w refills exactly one tank at one point in time
- w can switch between tanks instantaneously



$\rightsquigarrow$ Hybrid

There are much more complex examples of hybrid systems...

- automobils, trains, etc.
- automated highway systems
- collision-avoidance and free flight for aircrafts
- biological cell growth and division

Contents

Labeled state transition systems

Definition

A labeled state transition system (LSTS) is a tuple
$\mathcal{LSTS} = (\Sigma, Lab, Edge, Init)$ with

- a (probably infinite) state set Σ,
- a label set Lab,
- a transition relation $Edge \subseteq \Sigma \times Lab \times \Sigma$,
- non-empty set of initial states $Init \subseteq \Sigma$.

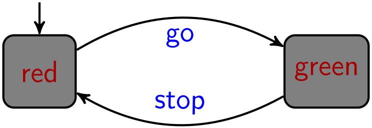
Semantics of LSTS

Operational semantics

$$\frac{(\sigma, a, \sigma') \in Edge}{\sigma \stackrel{a}{\rightarrow} \sigma'}$$

- system run (execution): $\sigma_0 \stackrel{a_0}{\rightarrow} \sigma_1 \stackrel{a_1}{\rightarrow} \sigma_2 \ldots$ with $\sigma_0 \in Init$
- a state is called reachable iff there is a run leading to it

Pedestrian light



Parallel composition

Larger or more complex systems are often modeled compositionally.

- The global system is given by the parallel composition of the components.
- Component-local, non-synchronizing transitions, having labels belonging to one components's label set only, are executed in an interleaved manner.
- Synchronizing transitions of the components, agreeing on the label, are executed synchronously.

Parallel composition of LSTSs

Definition

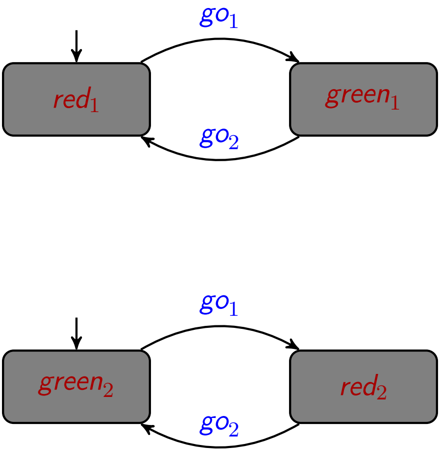
Let

$$\mathcal{LSTS}_1 = (\Sigma_1, Lab_1, Edge_1, Init_1) \text{ and}$$
$$\mathcal{LSTS}_2 = (\Sigma_2, Lab_2, Edge_2, Init_2)$$

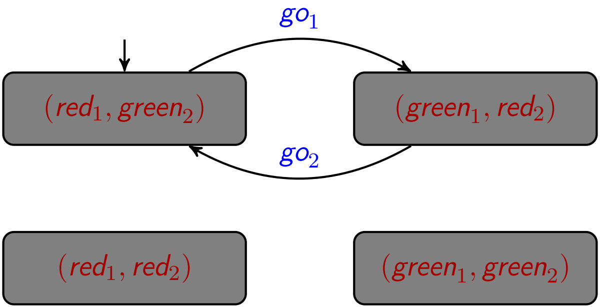
be two LSTSs. The parallel composition $\mathcal{LSTS}_1 || \mathcal{LSTS}_2$ is the LSTS $(\Sigma, Lab, Edge, Init)$ with

- $\Sigma = \Sigma_1 \times \Sigma_2$,
- $Lab = Lab_1 \cup Lab_2$,
- $((s_1, s_2), a, (s_1', s_2')) \in Edge$ iff
  1. $a \in Lab_1 \cap Lab_2$, $(s_1, a, s_1') \in Edge_1$, and $(s_2, a, s_2') \in Edge_2$, or
  2. $a \in Lab_1 \setminus Lab_2$, $(s_1, a, s_1') \in Edge_1$, and $s_2 = s_2'$, or
  3. $a \in Lab_2 \setminus Lab_1$, $(s_2, a, s_2') \in Edge_2$, and $s_1 = s_1'$,
- $Init = (Init_1 \times Init_2)$.

Two traffic lights



Two traffic lights



Labeling

To be able to formalize properties of LSTSs, it is common to define

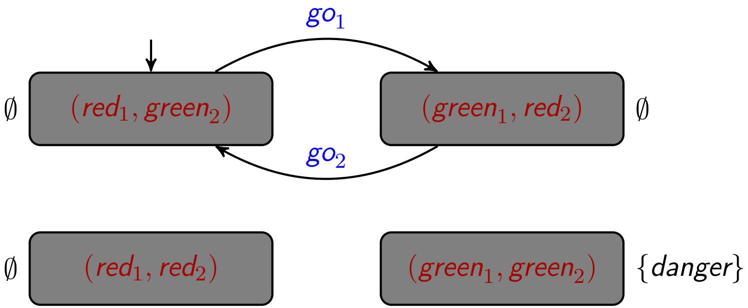
- a set of atomic propositions AP and
- a labeling function $L : \Sigma \to 2^{AP}$ assigning a set of atomic propositions to each state.

The set $L(\sigma)$ consists of all propositions that are defined to hold in σ.

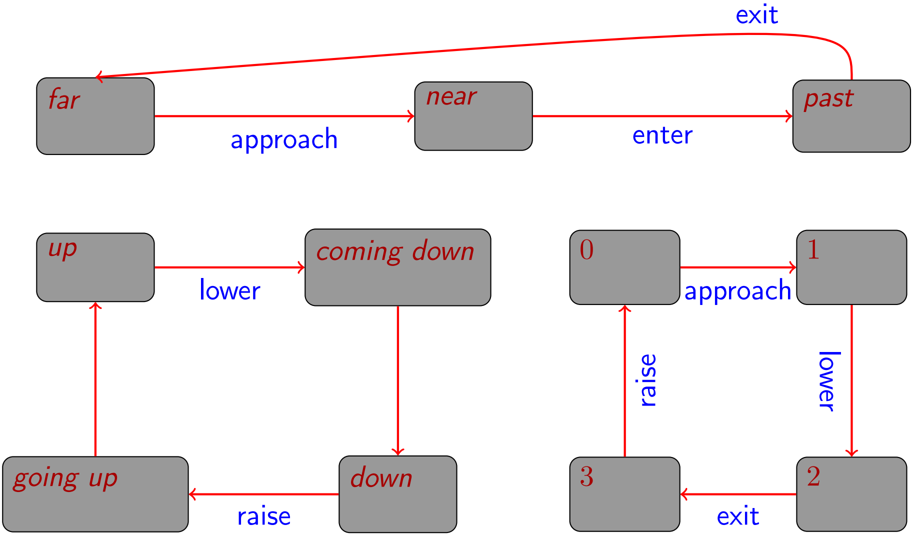
Two kinds of labels:

- propositional labels on states
- synchronization labels on edges

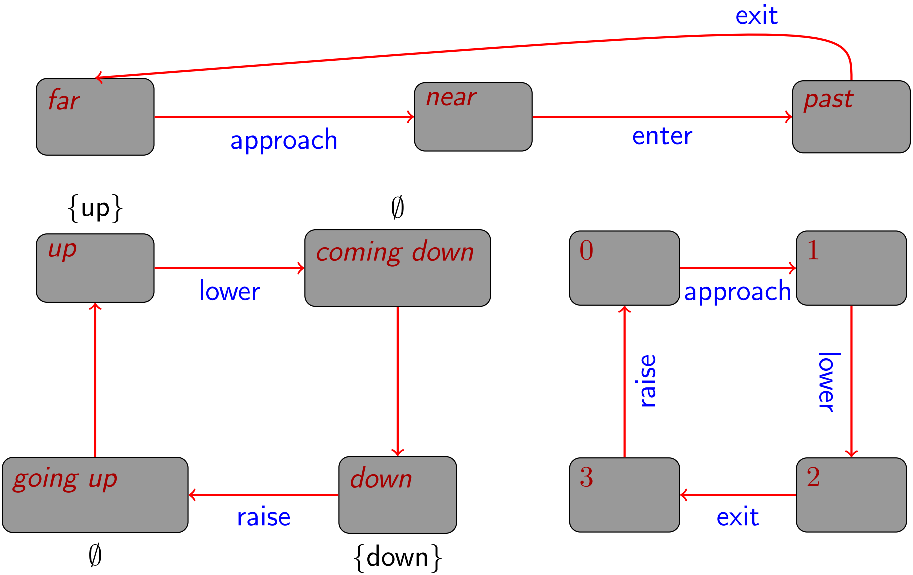
Two traffic lights



Railroad crossing: Train, controller and gate



Railroad crossing: Train, controller and gate



Contents

Labeled transition system

Definition

$$\mathcal{LTS} = (Loc, Var, Lab, Edge, Init)$$

- finite set of locations Loc,
- finite set of (typed) variables Var,
- finite set of synchronization labels Lab
- finite set of edges $Edge \subseteq Loc \times Lab \times 2^{V^2} \times Loc$ (l, τ, Id, l) for each location $l \in Loc$),
- initial states $Init \subseteq \Sigma$.

where

- V is the set of valuations $\nu : Var \rightarrow Domain$
- Σ is the set of state $\sigma = (l, \nu) \in Loc \times V$

Labeled transition system with stuttering

Definition

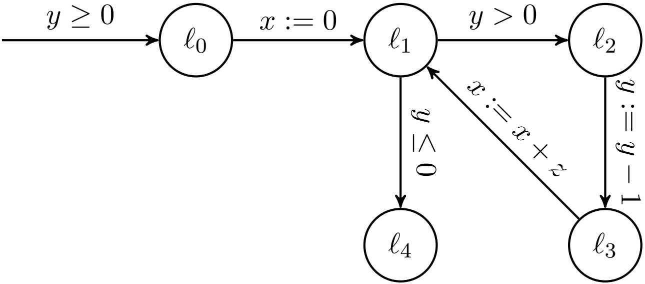
$$\mathcal{LTS} = (Loc, Var, Lab, Edge, Init)$$

- finite set of locations Loc,
- finite set of (typed) variables Var,
- finite set of synchronization labels Lab
  special stutter label $\tau \in Lab$
- finite set of edges $Edge \subseteq Loc \times Lab \times 2^{V^2} \times Loc$
  special stutter transition (l, τ, Id, l) for each location $l \in Loc$
- initial states $Init \subseteq \Sigma$.

Modeling a simple while-program

```
     method mult(int y, int z){
       int x;
ℓ0     x := 0;
ℓ1
       while( y > 0 ) {
ℓ2       y := y-1;
ℓ3       x := x+z;
       }
ℓ4   }
```

Modeling a simple while-program



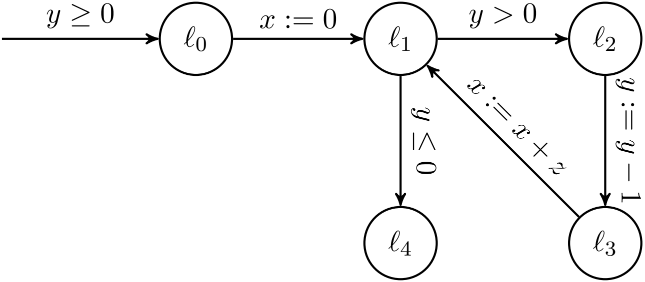
Semantics of LTS

Operational semantics

$$\frac{(l, a, \mu, l') \in Edge \quad (\nu, \nu') \in \mu}{(l, \nu) \xrightarrow{a} (l', \nu')}$$

- system run (execution): $\sigma_0 \xrightarrow{a_0} \sigma_1 \xrightarrow{a_1} \sigma_2 \ldots$ with $\sigma_0 \in Init$
- a state is called reachable iff there is a run leading to it

Semantics of the simple while-program



$$\frac{(l, a, \mu, l') \in Edge \quad (\nu, \nu') \in \mu}{(l, \nu) \xrightarrow{a} (l', \nu')}$$

Parallel composition of LTSs

Definition

Let

$$\mathcal{LTS}_1 = (Loc_1, Var, Lab_1, Edge_1, Init_1) \text{ and}$$
$$\mathcal{LTS}_2 = (Loc_2, Var, Lab_2, Edge_2, Init_2)$$

be two LTSs. The parallel composition or product $\mathcal{LTS}_1 || \mathcal{LTS}_2$ is

$$\mathcal{LTS} = (Loc, Var, Lab, Edge, Init)$$

with

- $Loc = Loc_1 \times Loc_2$,
- $Lab = Lab_1 \cup Lab_2$,
- $Init = \{((l_1, l_2), \nu) \mid (l_1, \nu) \in Init_1 \wedge (l_2, \nu) \in Init_2\}$,

Parallel composition of LTSs

Definition ((Cont.))

and

- $((l_1, l_2), a, \mu, (l_1', l_2')) \in Edge$ iff
  - there exist $(l_1, a_1, \mu_1, l_1') \in Edge_1$ and $(l_2, a_2, \mu_2, l_2') \in Edge_2$ such that
  - either $a_1 = a_2 = a$ or
    $a_1 = a \in Lab_1 \backslash Lab_2$ and $a_2 = \tau$, or
    $a_1 = \tau$ and $a_2 = a \in Lab_2 \backslash Lab_1$, and
  - $\mu = \mu_1 \cap \mu_2$.

Contents

Hybrid automata

Hybrid automata

Definition

A hybrid automaton is a tuple $\mathcal{H} = (Loc, Var, Lab, Edge, Act, Inv, Init)$ with

- a finite set of locations Loc,
- a finite set of real-valued variables Var,
- a finite set of synchronization labels Lab, $\tau \in Lab$ (stutter label)
- a finite set of edges $Edge \subseteq Loc \times Lab \times 2^{V^2} \times Loc$ (including stutter transitions $(l, \tau, \mathtt{Id}, l)$ for each location $l \in Loc$),
- Act is a function assigning a set of activities $f : \mathbb{R}^+ \to V$ to each location; the activity sets are time-invariant, i.e., $f \in Act(l)$ implies $(f + t) \in Act(l)$, where $(f + t)(t') = f(t + t')$ f.a. $t' \in \mathbb{R}^+$,
- a function Inv assigning an invariant $Inv(l) \subseteq V$ to each location $l \in Loc$,
- initial states $Init \subseteq \Sigma$.

with

- valuations $\nu : Var \to \mathbb{R}$, V is the set of valuations
- state $(l, \nu) \in Loc \times V$, Σ is the set of states
- transitions: discrete and time

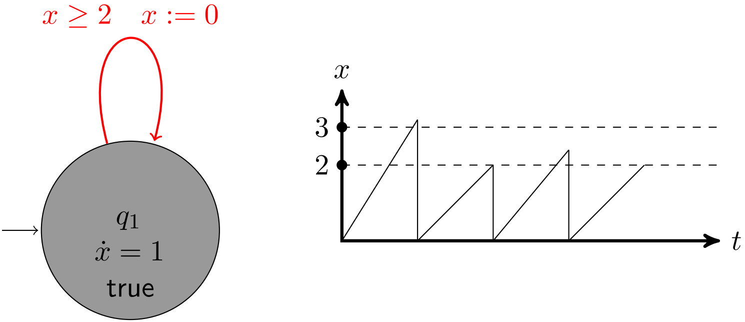
Semantics of hybrid automata

$$\frac{(l,a,\mu,l') \in Edge \quad (\nu,\nu') \in \mu \quad \nu' \in Inv(l')}{(l,\nu) \xrightarrow{a} (l',\nu')} \ \texttt{Rule}_{\texttt{Discrete}}$$

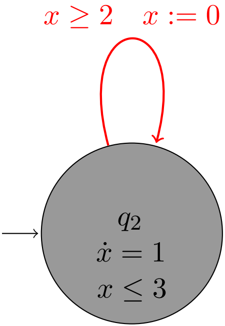
$$\frac{\begin{array}{c} f \in Act(l) \quad f(0) = \nu \quad f(t) = \nu' \\ t \geq 0 \quad \forall 0 \leq t' \leq t . f(t') \in Inv(l) \end{array}}{(l,\nu) \xrightarrow{t} (l,\nu')} \ \texttt{Rule}_{\texttt{Time}}$$

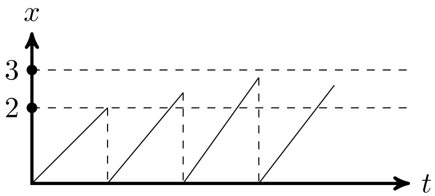
- execution step: $\rightarrow \; = \; \xrightarrow{a} \cup \xrightarrow{t}$
- run: $\sigma_0 \rightarrow \sigma_1 \rightarrow \sigma_2 \ldots$ with $\sigma_0 = (l_0, \nu_0) \in Init$ and $\nu_0 \in Inv(l_0)$
- reachability of a state: exists run leading to the state
- activities are represented in form of differential equations

Example: Timed automata

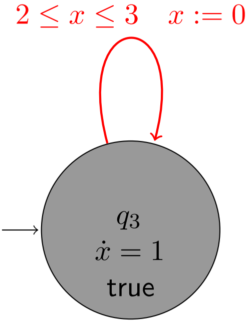


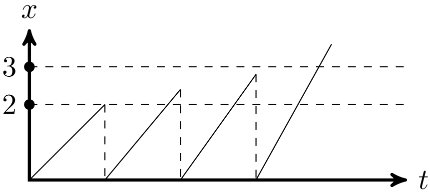
Example: Timed automata





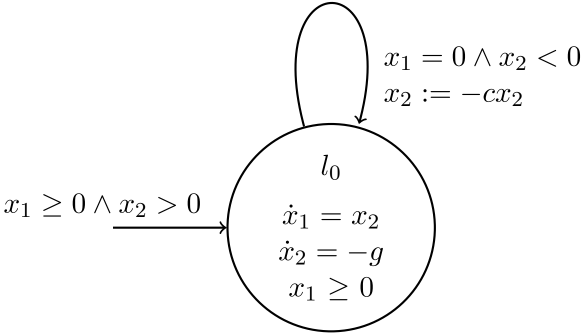
Example: Timed automata





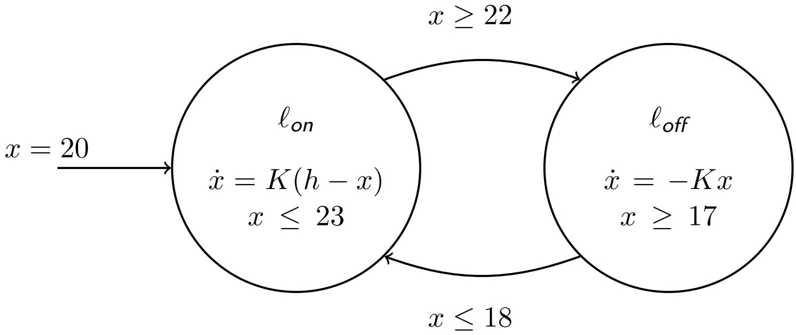
Example revisited: Bouncing ball

- vertical position of the ball x_1
- velocity x_2
- continuous changes of position between bounces
- discrete changes at bounce time



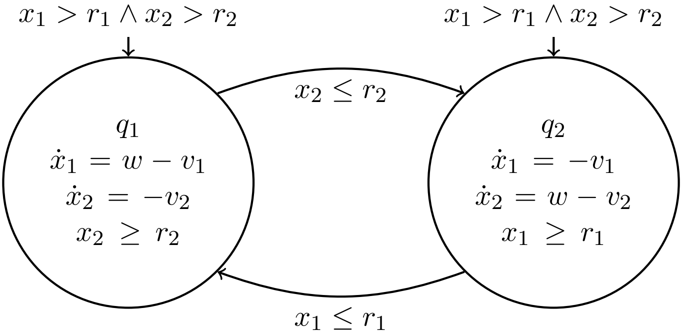
Example revisited: Thermostat

- $17° \leq x \leq 18° \rightsquigarrow$ "heater on"
- $22° \leq x \leq 23° \rightsquigarrow$ "heater off"



Example revisited: Water tank system

- two constantly leaking tanks v_1 and v_2
- hose w refills exactly one tank at one point in time
- w can switch between tanks instantaneously



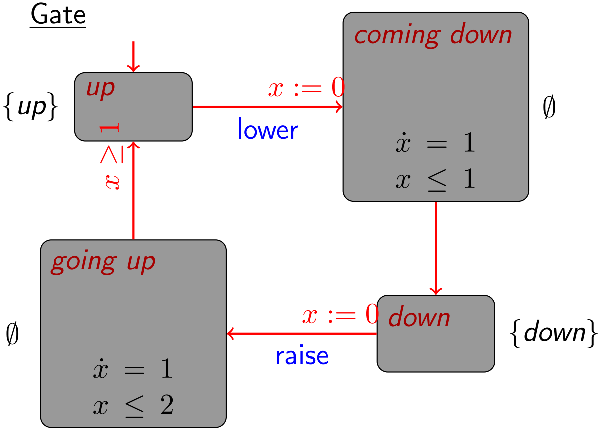
Parallel composition

Definition

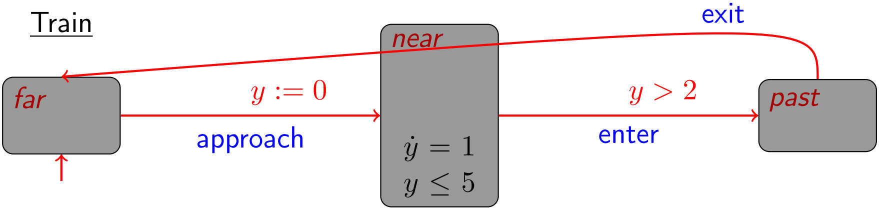
Let $\mathcal{H}_1 = (Loc_1, Var, Lab_1, Edge_1, Act_1, Inv_1, Init_1)$ and
$\mathcal{H}_2 = (Loc_2, Var, Lab_2, Edge_2, Act_2, Inv_2, Init_2)$
be two hybrid automata. The product
$\mathcal{H}_1 || \mathcal{H}_2 = (Loc_1 \times Loc_2, Var, Lab_1 \cup Lab_2, Edge, Act, Inv, Init)$ is the hybrid automaton with

- $Act(l_1, l_2) = Act_1(l_1) \cap Act_2(l_2)$ for all $(l_1, l_2) \in Loc$,
- $Inv(l_1, l_2) = Inv_1(l_1) \cap Inv_2(l_2)$ for all $(l_1, l_2) \in Loc$,
- $Init = \{((l_1, l_2), \nu) | (l_1, \nu) \in Init_1,\ (l_2, \nu) \in Init_2\}$, and
- $((l_1, l_2), a, \mu, (l'_1, l'_2)) \in Edge$ iff
  - $(l_1, a_1, \mu_1, l'_1) \in Edge_1$ and $(l_2, a_2, \mu_2, l'_2) \in Edge_2$, and
  - either $a_1 = a_2 = a$, or $a_1 = a \notin Lab_2$ and $a_2 = \tau$, or $a_1 = \tau$ and $a_2 = a \notin Lab_1$, and
  - $\mu = \mu_1 \cap \mu_2$.

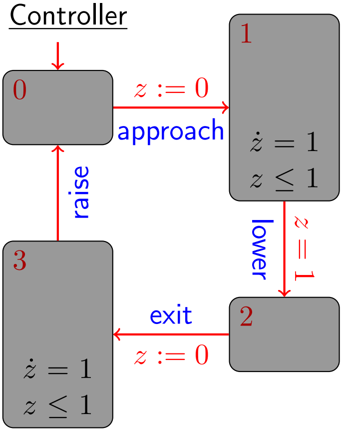
Simplified railroad crossing with time component

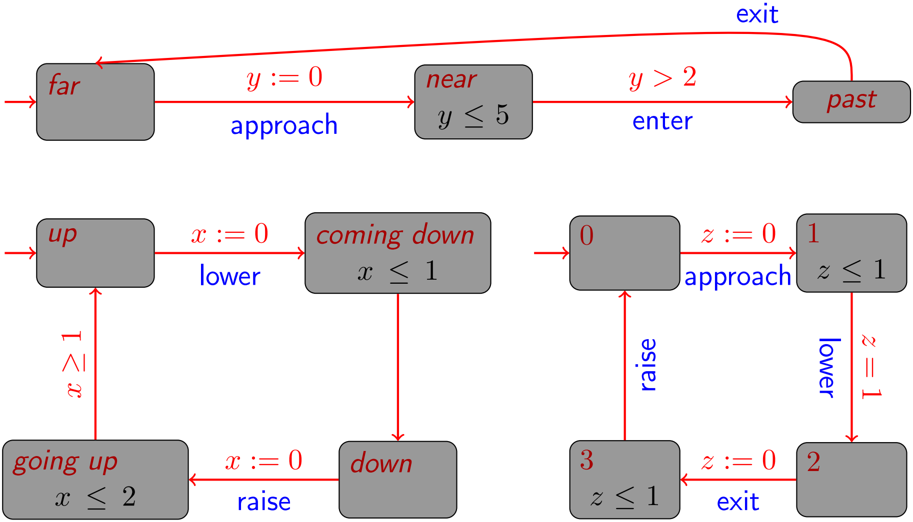


Simplified railroad crossing with time component



Simplified railroad crossing with time component





exit
far
$y := 0$
approach
near
$y \leq 5$
$y > 2$
enter
past
up
$x := 0$
lower
coming down
$x \leq 1$
$x \geq 1$
going up
$x \leq 2$
$x := 0$
raise
down
0
$z := 0$
approach
1
$z \leq 1$
raise
lower
$z = 1$
3
$z \leq 1$
$z := 0$
exit
2