Modeling and Analysis of Hybrid Systems

Hybrid systems and their modeling

Prof. Dr. Erika Ábrahám

Informatik 2 - Theory of Hybrid Systems
RWTH Aachen University

SS 2013

Contents

Contents

Motivation

- Dynamical system: continuous evolution of the state over time
  Discrete system: instantaneous state changes
  Hybrid system: combination

Motivation

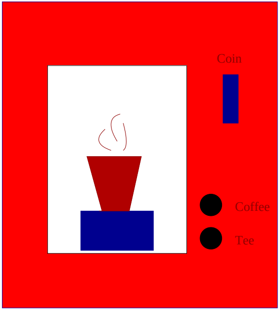
- Dynamical system: continuous evolution of the state over time
  Discrete system: instantaneous state changes
  Hybrid system: combination
- Time model:
  - continuous $\rightsquigarrow$ $t \in \mathbb{R}$
  - discrete $\rightsquigarrow$ $k \in \mathbb{Z}$
  - hybrid $\rightsquigarrow$ continuous time, but there are also discrete "instants" where something "special" happens

Motivation

- Dynamical system: continuous evolution of the state over time
  Discrete system: instantaneous state changes
  Hybrid system: combination
- 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

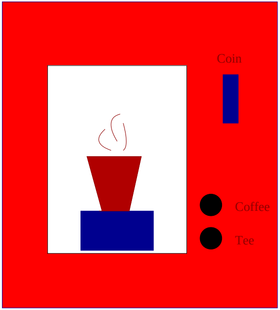
Example: Vending machine

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



Example: Vending machine

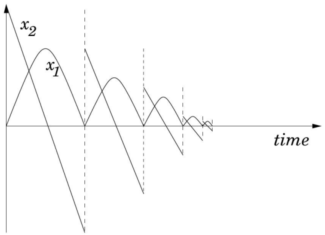
- insert coin
- choose beverage (coffee/tee)
- wait for cup
- take cup



$\rightsquigarrow$ Can be modeled discretely, when abstracting away from time and physical processes

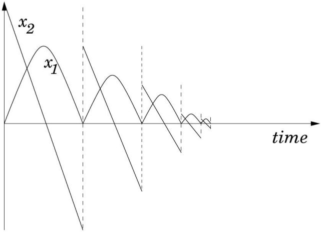
Example: Bouncing ball

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



Example: Bouncing ball

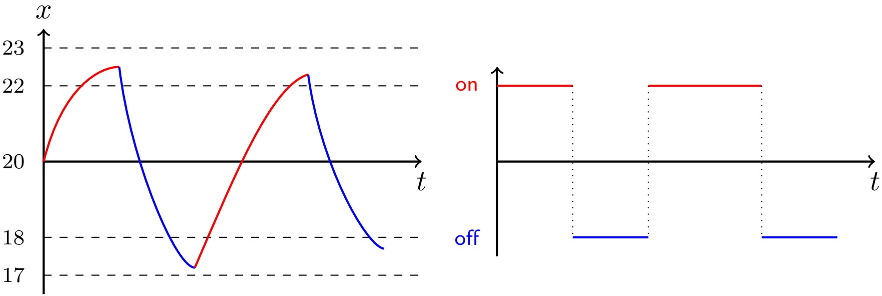
- vertical position of the ball x_1
- velocity x_2
- continuous changes of position between bounces
- discrete changes at bounce time



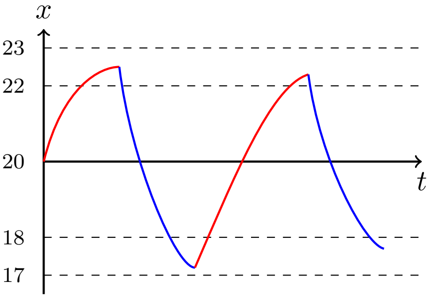
$\rightsquigarrow$ Hybrid

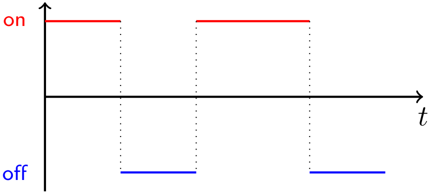
Example: Thermostat

- 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"



- 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"

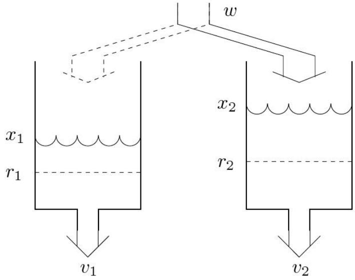




$\rightsquigarrow$ Hybrid

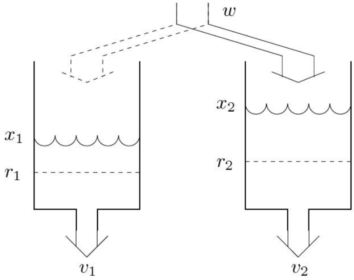
Example: Water tank system

- 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



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

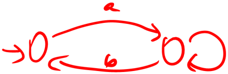


$\rightsquigarrow$ Hybrid

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



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 $Init \subseteq \Sigma$.

$\{x \geq 0\}$

$Init = \{x \in \mathbb{Z} \mid x \geq 0\}$

l_0: while $(x > 0)$ {

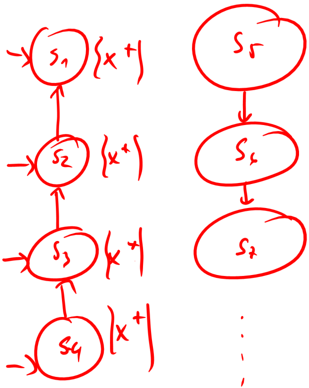
l_1: $x := x - 1$

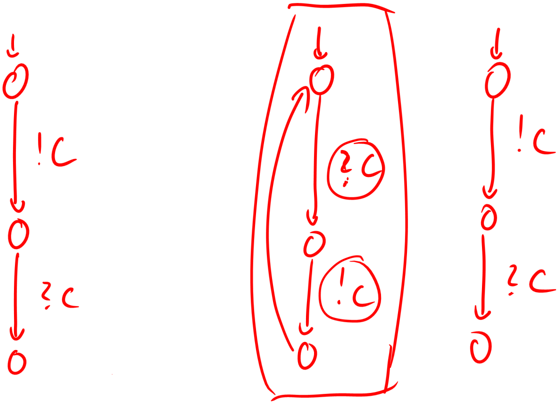
l_2: }

$\Sigma = \mathbb{Z}$

$Edge = \{x \rightarrow x' \mid x' = x - 1 \wedge x > 0\}$

$AG\ x^+$





$\alpha ab = \{ !c, ?c \}$

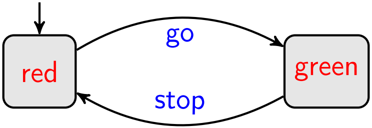
Semantics of LSTS

Operational semantics is trivial:

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

- 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.

Definition

Let

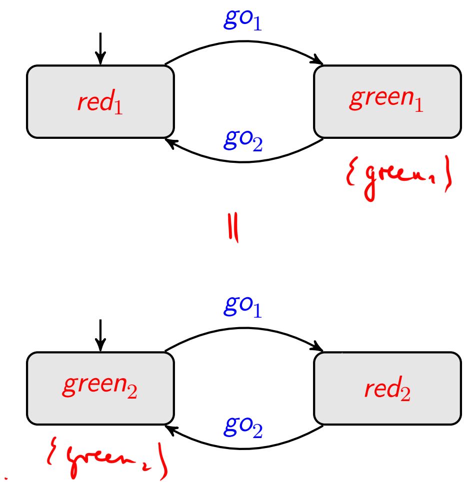
$$\begin{aligned} \mathcal{LSTS}_1 &= (\Sigma_1, Lab_1, Edge_1, Init_1) \textit{ and} \\ \mathcal{LSTS}_2 &= (\Sigma_2, Lab_2, Edge_2, Init_2) \end{aligned}$$

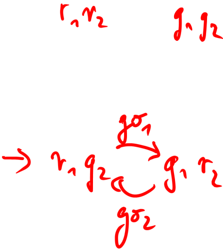
be two LSTSs. The parallel composition $\mathcal{LSTS}_1 || \mathcal{LSTS}_2$ is the LSTS $(\Sigma, Lab, Edge, Init)$ with

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

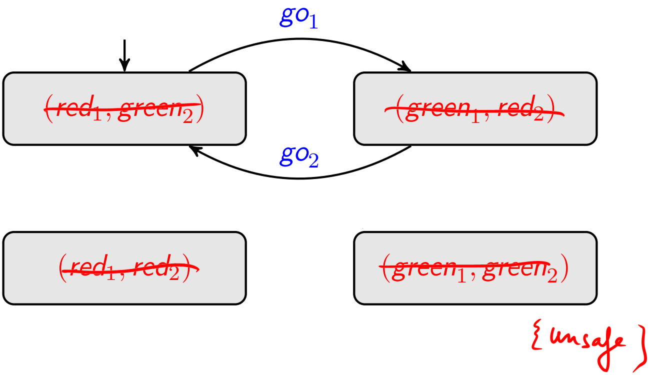
$s_1 \in \Sigma_1$ $s_2 \in \Sigma_2$ $(s_1, s_2) \in \Sigma_1 \times \Sigma_2 = \Sigma$

Two traffic lights





Two traffic lights



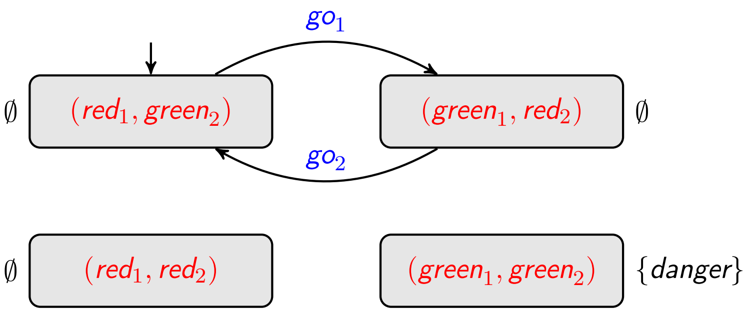
Labeling

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

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

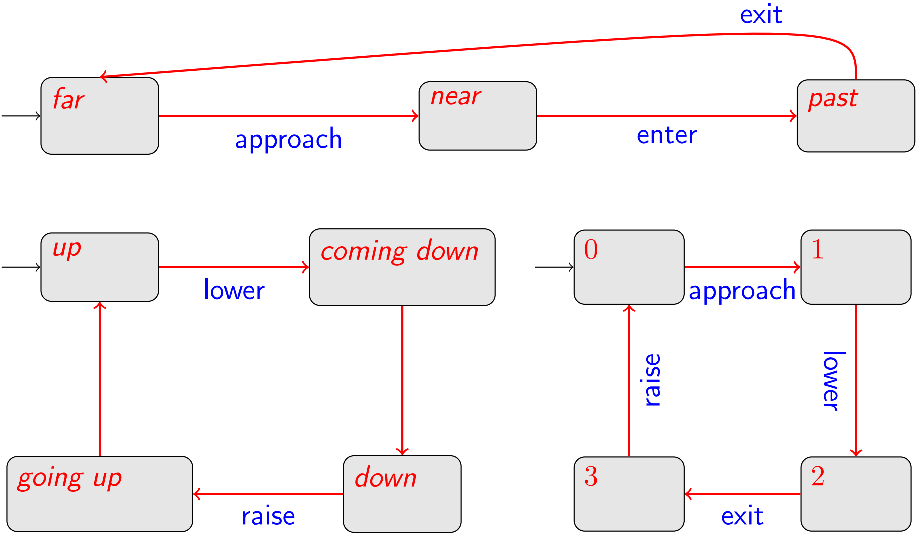
The set $L(\sigma)$ consists of all propositions that are defined to hold in σ.
These propositional labels on states should not be mixed up with the synchronization labels on edges.

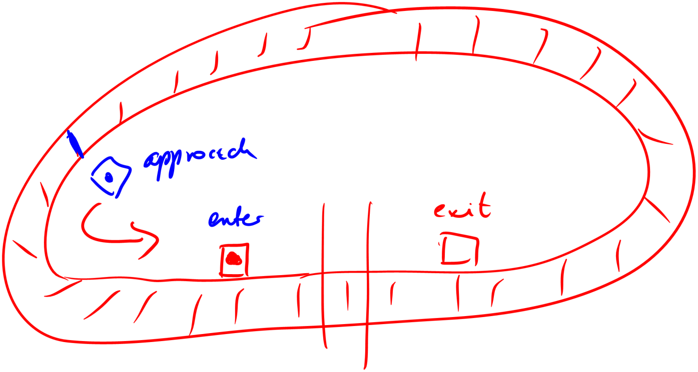
Two traffic lights



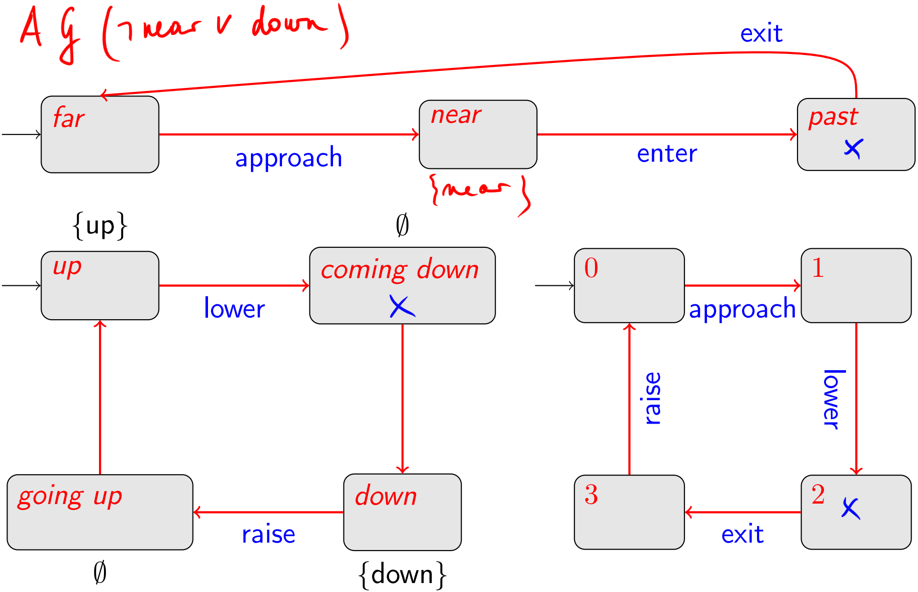
Railroad crossing: Train, controller and gate

Railroad crossing: Train, controller and gate




approach
enter
exit

Railroad crossing: Train, controller and gate



Contents

Labeled transition systems

Labeled transition systems

Definition

A labeled transition system (LTS) is a tuple
$\mathcal{LTS} = (Loc, Var, Lab, Edge, Init)$ with

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

with

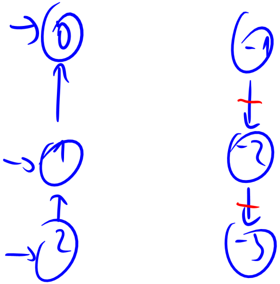
- valuations $\nu : Var \to Domain$, V is the set of valuations
- state $\sigma = (l, \nu) \in Loc \times V$, Σ is the set of states

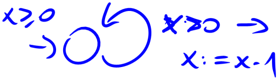
$\{x \geq 0\}$

ℓ_0 while $x > 0$

ℓ_1 $\quad x := x - 1;$

ℓ_2

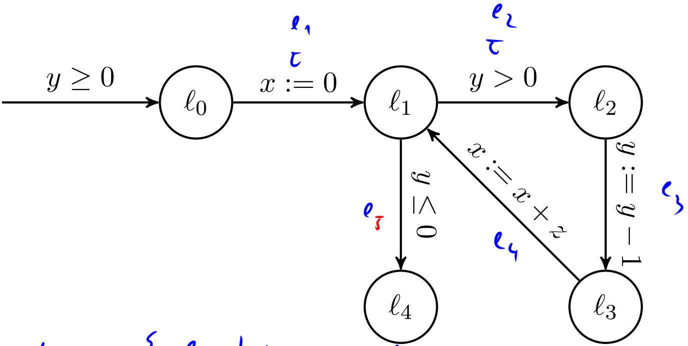




$$\{(\sigma_1, \sigma_2) \in \mathbb{Z} \times \mathbb{Z} \mid \sigma_1 > 0 \wedge \sigma_2 = \sigma_1 - 1\}$$

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  }
```



$Loc = \{ \ell_i \mid i = 0, \ldots, 4 \}$

$Lab = \{ \tau \}$

$Var = \{ x, y, z \}$

$Edge = \{ e_1, e_2, e_3, e_4, e_5 \}$

$e_4 = (\ell_3, \tau, \{ ((x,y,z),(x',y',z')) \in \mathbb{Z}^3 \times \mathbb{Z}^3 \mid y' = y \wedge z' = z \wedge x' = x + z \}, \ell_1)$

$Init = \{ (x,y,z) \in \mathbb{Z}^3 \mid y \geq 0 \}$

Semantics of LTS

Operational semantics has a single rule:

Semantics of LTS

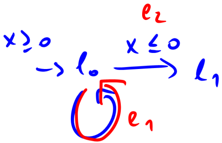
Operational semantics has a single rule:

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

$\{x \geq 0\}$

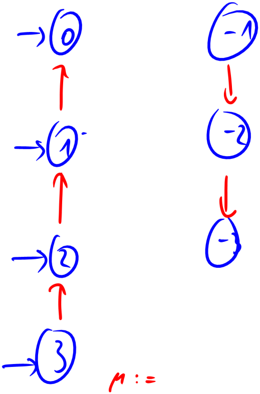
ℓ_0 while$(x > 0)$ $x := x - 1$;

ℓ_1



$x > 0 \to x := x - 1$

$\Leftarrow$



$1 \to 0$

$2 \to 1$

$\mu :=$

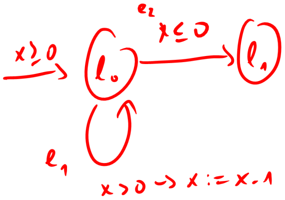
$e_1 = (\ell_0, \tau, \{(\nu, \nu') \in V^2 \mid \nu(x) > 0 \wedge \nu'(x) = \nu(x) - 1\}, \ell_0)$

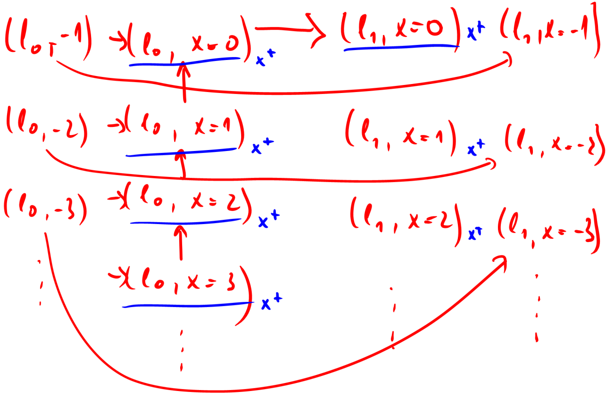
$(\ell_0, x = 2) \to (\ell_0, x = 1) \to (\ell_0, x = 0)$

$\downarrow$

$(\ell_1, x = 0)$

$$\frac{(x = 2, x = 1) \in \mu}{(\ell_0, x = 2) \xrightarrow{\tau} (\ell_0, x = 1)}$$





LTS $\xrightarrow{\text{sem}}$ LSTS

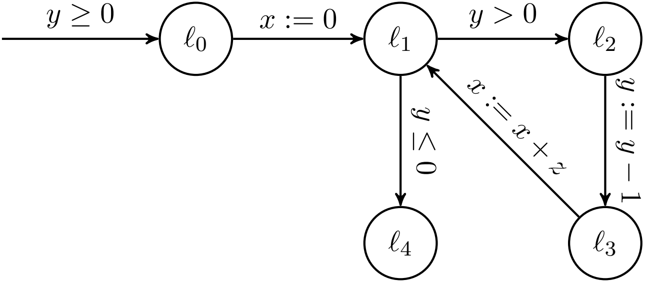
Semantics of LTS

Operational semantics has a single rule:

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

- 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

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

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

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.))

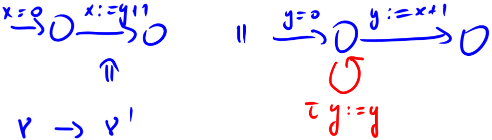
and

- $((l_1, l_2), a, \mu, (l'_1, l'_2)) \in Edge$ 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
  - $\mu = \mu_1 \cap \mu_2$.

$\{x = 0\}$ $\quad$ $\{y = 0\}$

$x := y + 1 \quad \| \quad y := x + 1$

a:

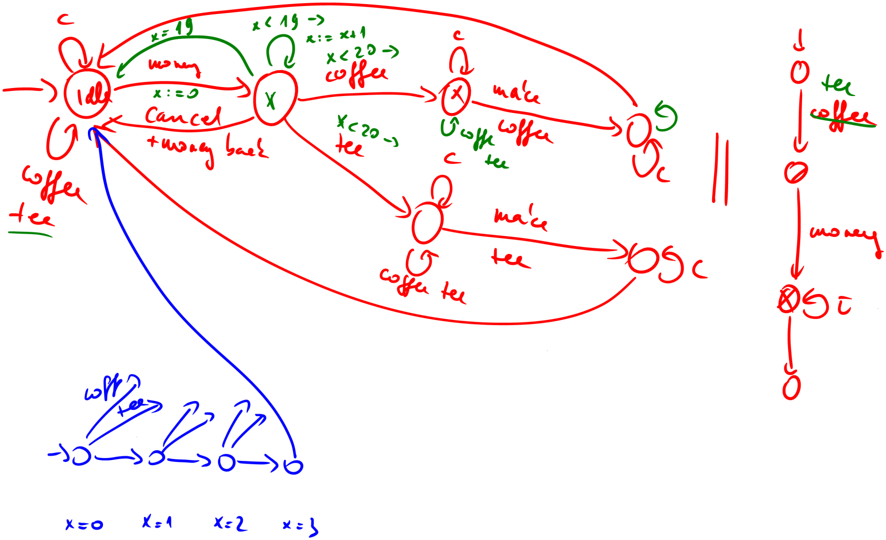


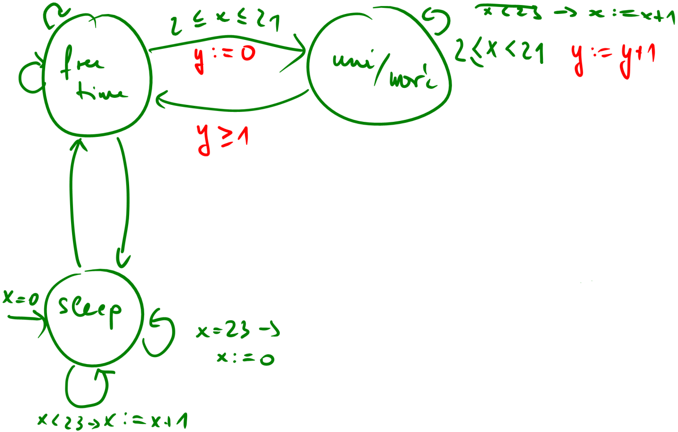
$\nu \rightarrow \nu'$

$\nu'(x) = \nu(y+1)$

~~$\nu'(y) = \nu(y)$~~

Parallel composition of LTSs





free time
uni/work
sleep
2 ≤ x ≤ 21
y := 0
y ≥ 1
x < 23 -> x := x+1
2 ≤ x < 21 y := y+1
x=0
x=23 ->
x := 0
x < 23 -> x := x+1

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, τ, μ_τ, 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 $Init \subseteq \Sigma$.

LTS

$f(x) = x + c$

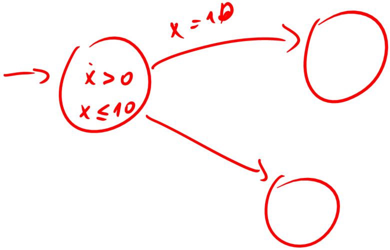
$f(0) = 0$

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

$\dot{x} = 1$

$f(x) = x + c$

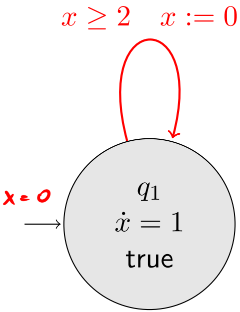

x := 10
ẋ > 0
x ≤ 10

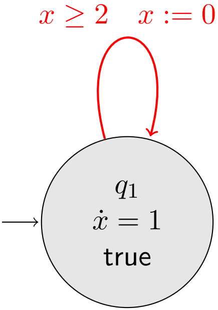
$$\frac{(l, a, \mu, l') \in Edge \quad (\nu, \nu') \in \mu \quad \nu' \in Inv(l')}{(l, \nu) \xrightarrow{a} (l', \nu')} \quad \texttt{Rule}_{\texttt{Discrete}}$$

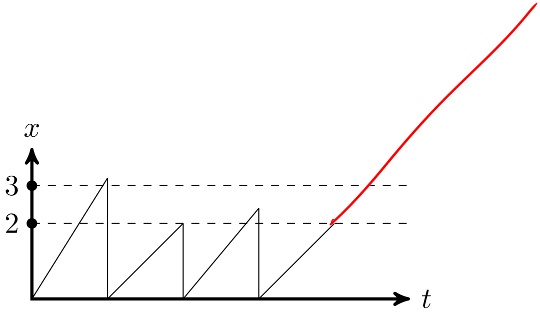
$$\frac{\begin{array}{c} f \in Act(l) \quad f(0) = \nu \quad f(t) = \nu' \\ t \geq 0 \quad \forall 0 \leq t' \leq t. f(t') \in Inv(l) \end{array}}{(l, \nu) \xrightarrow{t} (l, \nu')} \quad \texttt{Rule}_{\texttt{Time}}$$

- 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

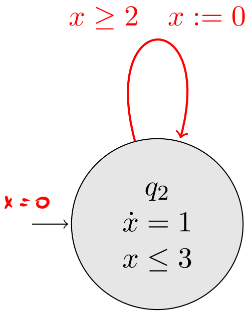
Example: Timed automata



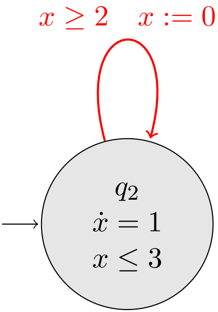

$x \geq 2$ $x := 0$
q_1
$\dot{x} = 1$
true

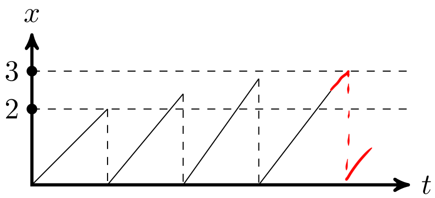

x
3
2
t

Example: Timed automata

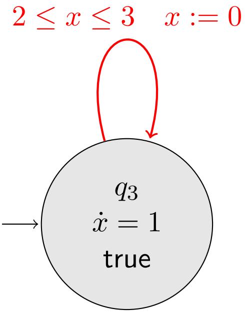


Example: Timed automata

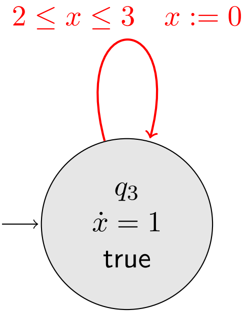


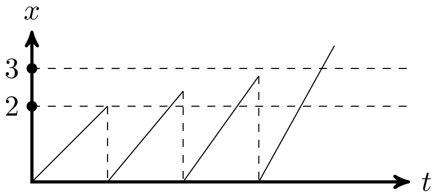


Example: Timed automata



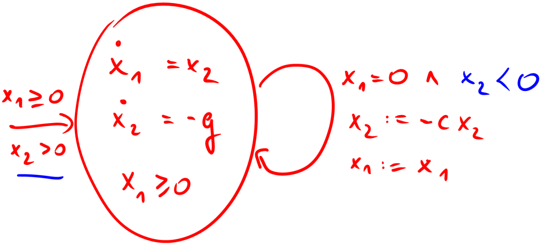
Example: Timed automata





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



$x_1 = 0$, $x_2 < 0$ $\Rightarrow$ $x_1' = 0$, $x_2' = -c \cdot x_2 \geq 0$

$\Downarrow$

$x_1'' = 0$

$x_2'' = c^2 x_2 < 0$

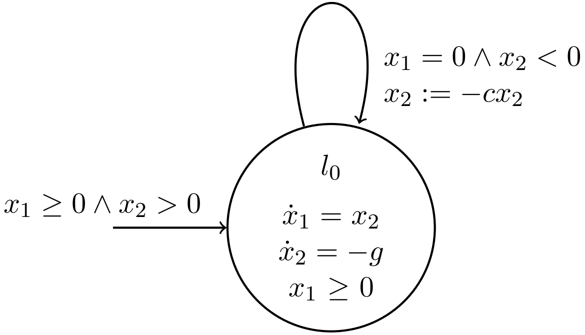
$\Downarrow$

$x_1''' = 0$

$x_2''' = -c^3 x_2 \geq 0$

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

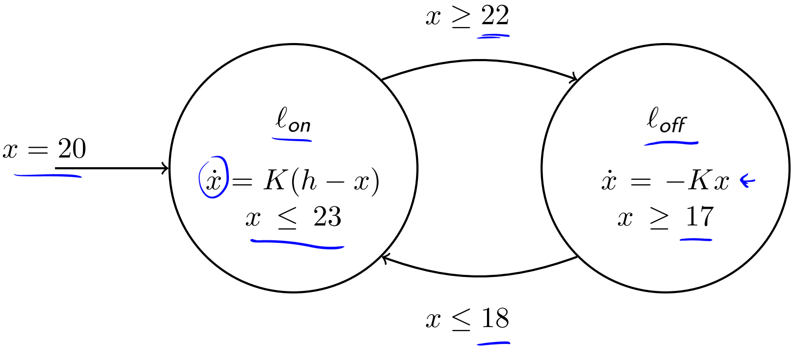


Example revisited: Thermostat

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

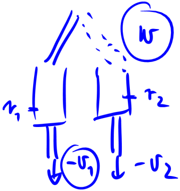
Example revisited: Thermostat

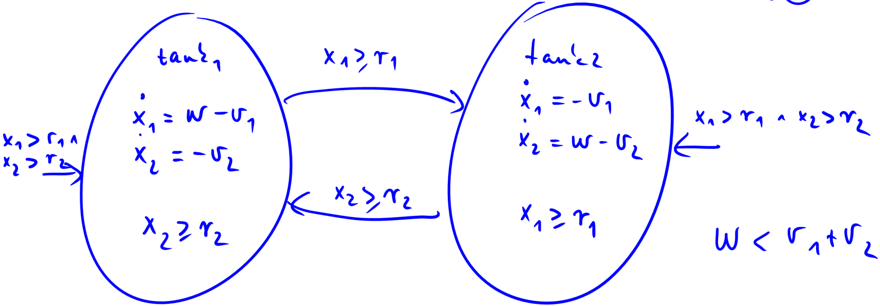
- $17° \leq x \leq 18°$ $\rightsquigarrow$ "heater on"
- $22° \leq x \leq 23°$ $\rightsquigarrow$ "heater off"



Example revisited: Water tank system

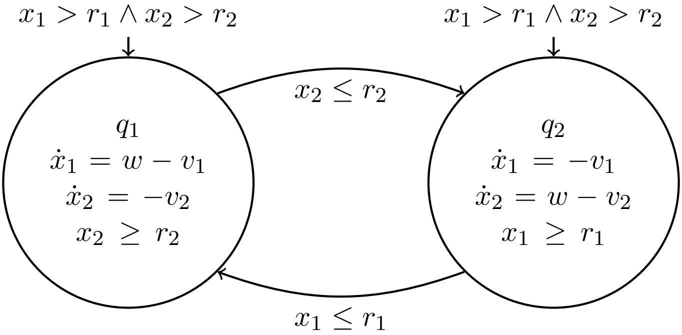
- two constantly leaking tanks v_1 and v_2
- hose w refills exactly one tank at one point in time
- w can switch between tanks instantaneously

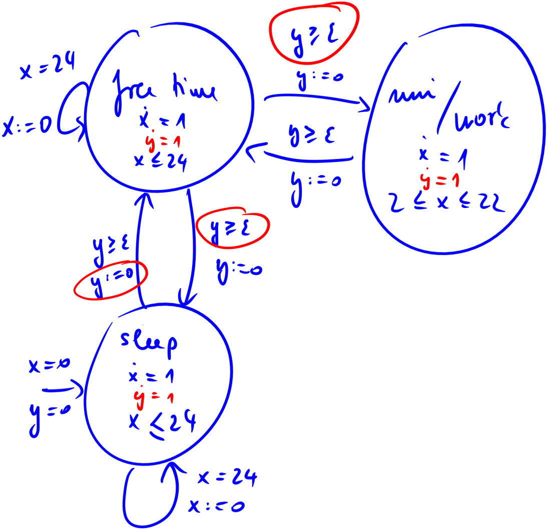




Example revisited: Water tank system

- 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





work:

$2 \leq x \leq 22$

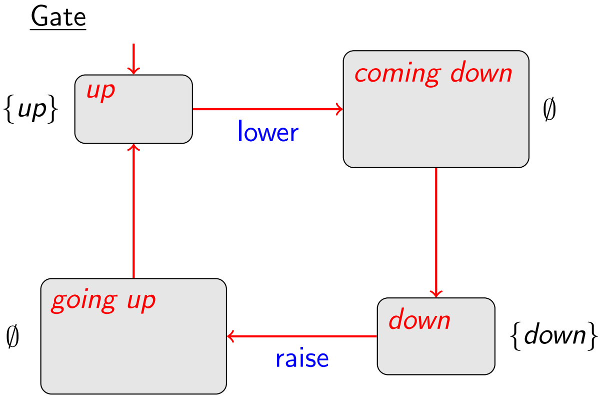
Parallel composition

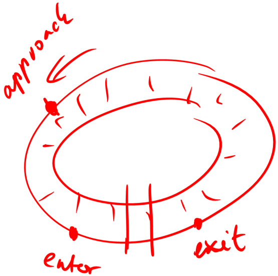
Definition

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

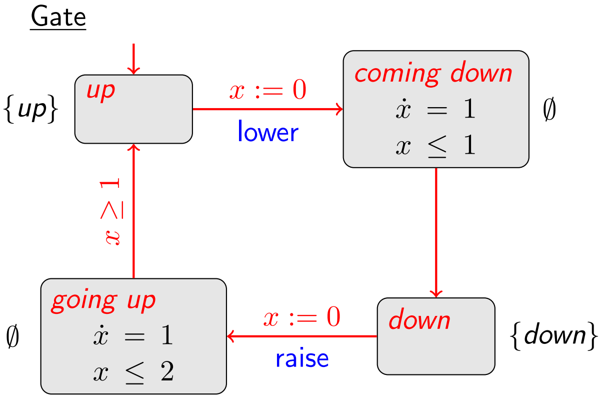
- $Act(l_1, l_2) = Act_1(l_1) \cap Act_2(l_2)$ for all $(l_1, l_2) \in Loc$,
- $Inv(l_1, l_2) = Inv_1(l_1) \cap Inv_2(l_2)$ for all $(l_1, l_2) \in Loc$,
- $Init = \{((l_1, l_2), \nu) | (l_1, \nu) \in Init_1, \ (l_2, \nu) \in Init_2\}$, and
- $((l_1, l_2), a, \mu, (l_1', l_2')) \in Edge$ iff
  - $(l_1, a_1, \mu_1, l_1') \in Edge_1$ 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

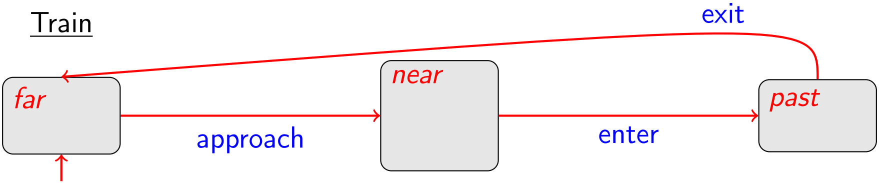




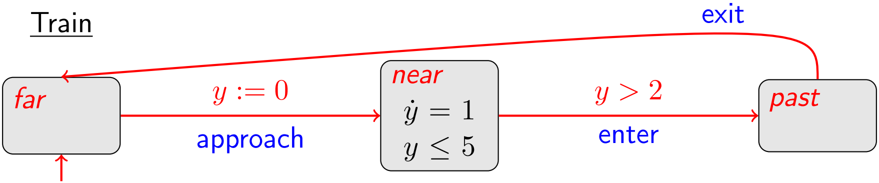
Simplified railroad crossing with time component



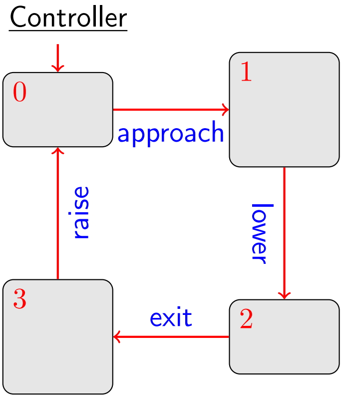
Simplified railroad crossing with time component



Simplified railroad crossing with time component

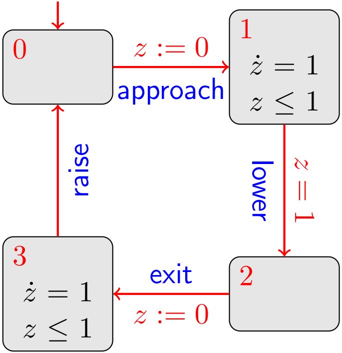


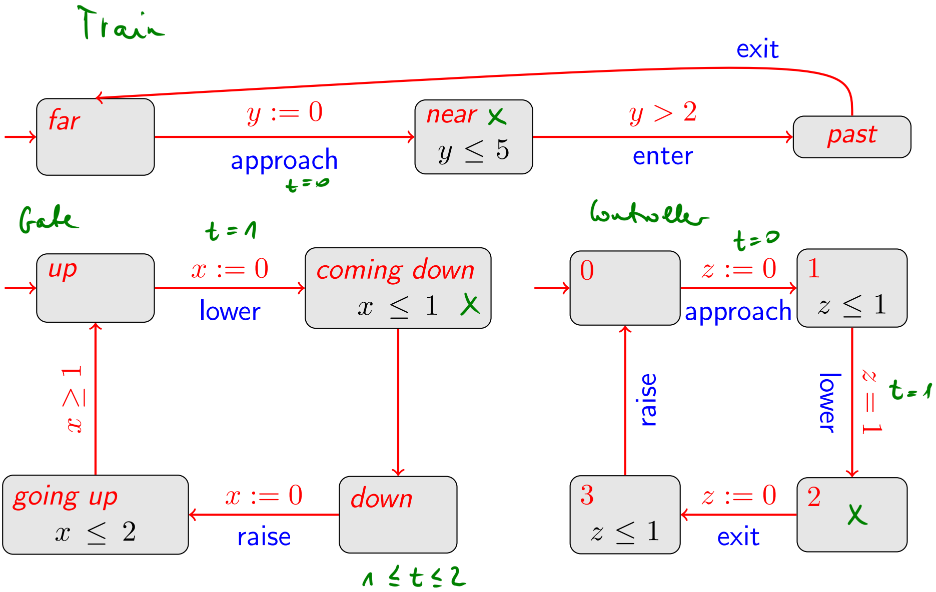
Simplified railroad crossing with time component

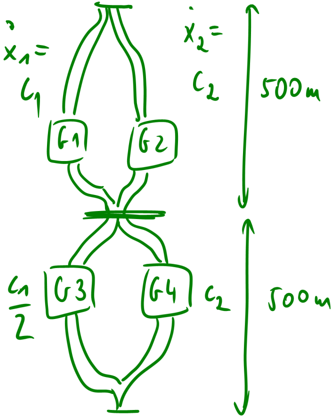


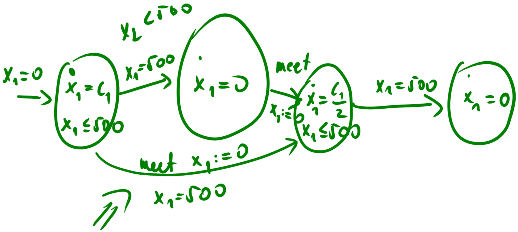
Simplified railroad crossing with time component

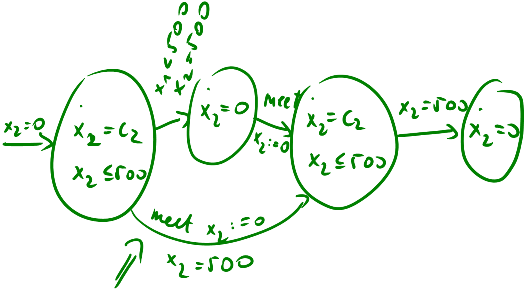
Controller




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


$\dot{x}_1 = c_1$
$\dot{x}_2 = c_2$
G1
G2
G3
G4
$\frac{c_1}{2}$
c_2
500m
500m


$x_1 = 0$
$\dot{x}_1 = c_1$
$x_1 \leq 500$
$x_1 = 500$
$x_2 < 500$
$\dot{x}_1 = 0$
meet
$x_1 := 0$
$\dot{x}_1 = \frac{c_1}{2}$
$x_1 \leq 500$
$x_1 = 500$
$\dot{x}_1 = 0$
meet $x_1 := 0$
$x_1 = 500$


$x_2 := 0$
$\dot{x}_2 = c_2$
$x_2 \leq 500$
$\dot{x}_2 = 0$
meet
$x_2 := 0$
$\dot{x}_2 = c_2$
$x_2 \leq 500$
$x_2 = 500$
$\dot{x}_2 = 0$
meet $x_2 := 0$
$x_2 = 500$