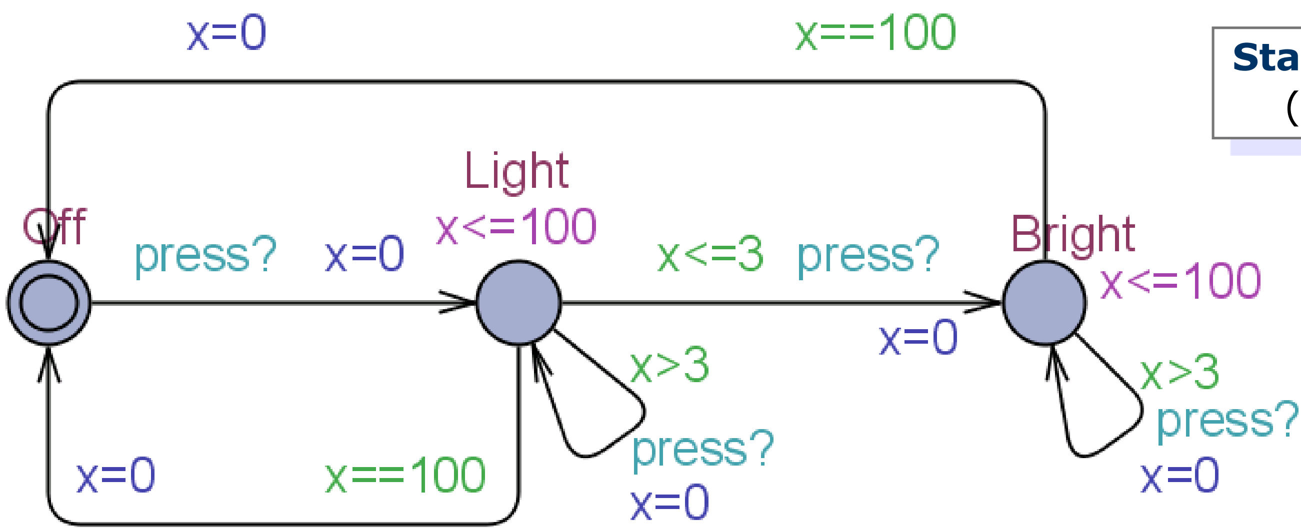
Semantics & Verification 2016

Lecture 10

- Timed Automata ....**Formally**
  - Networks of Timed Automata
- (Un-) Timed Bisimulation
- Automatic Verification of Timed Automata
  - Regions

Timed Automata

Intellingent Light Controller



States:
$(\ \ell oc\ ,x = v)$ where $v \in \mathbb{R}$

Transitions:

$(O, x=0) \xrightarrow{pr?} (L, x=0) \xrightarrow{5.7} (L, x=5.7)$

$\xrightarrow{pr?} (L, x=0)$

$\cancel{\xrightarrow{102}}$?? $(L, x=102)$

Clock Valuations

Let $C = \{x, y, \ldots\}$ be a finite set of clocks.

Set $\mathcal{B}(C)$ of clock constraints over C

$\mathcal{B}(C)$ is defined by the following abstract syntax

$$g, g_1, g_2 \ ::= x \sim n \mid x - y \sim n \mid g_1 \wedge g_2$$

where $x, y \in C$ are clocks, $n \in \mathbb{N}$ and $\sim \in \{\leq, <, =, >, \geq\}$.

Example: $x \leq 4 \ \wedge \ y > 5 \ \wedge \ x - y = 3$

Clock Valuation – Operations

Clock valuation

Clock valuation v is a function $v : C \to \mathbb{R}^{\geq 0}$. $V = [x=3.1, y=0]$

Let v be a clock valuation. Then

- $v + d$ is a clock valuation for any $d \in \mathbb{R}^{\geq 0}$ and it is defined by

$$(v + d)(x) = v(x) + d \text{ for all } x \in C$$

$V + 2.2 = [x=5.3, y=2.2]$

- $v[r]$ is a clock valuation for any $r \subseteq C$ and it is defined by

$$v[r](x) \begin{cases} 0 & \text{if } x \in r \\ v(x) & \text{otherwise.} \end{cases}$$

$V[\{x\}] = [x=0, y=0]$

Clock Valuation – Evaluation

Evaluation of clock constraints ($v \models g$)

$v \models x < n$ iff $v(x) < n$

$v \models x \leq n$ iff $v(x) \leq n$

$v \models x = n$ iff $v(x) = n$

$\vdots$

$v \models x - y < n$ iff $v(x) - v(y) < n$

$v \models x - y \leq n$ iff $v(x) - v(y) \leq n$

$\vdots$

$v \models g_1 \wedge g_2$ iff $v \models g_1$ and $v \models g_2$

$[x = 3.1, y = 0] \models x \leq 4$

$[x = 3.1, y = 0] \not\models y > 3$

Timed Automata – Syntax

Definition

A timed automaton over a set of clocks C and a set of labels N is a tuple

$$(L, \ell_0, E, I)$$

where

- L is a finite set of locations
- $\ell_0 \in L$ is the initial location
- $E \subseteq L \times \mathcal{B}(C) \times N \times 2^C \times L$ is the set of edges
- $I : L \to \mathcal{B}(C)$ assigns invariants to locations.

We usually write $\ell \xrightarrow{g,a,r} \ell'$ whenever $(\ell, g, a, r, \ell') \in E$.

Timed Automata – Semantics

Let $A = (L, \ell_0, E, I)$ be a timed automaton.

Timed transition system generated by A

$T(A) = (Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ where

- $Proc = L \times (C \to \mathbb{R}^{\geq 0})$, i.e. states are of the form (ℓ, v) where ℓ is a location and v a valuation
- $Act = N \cup \mathbb{R}^{\geq 0}$
- $\longrightarrow$ is defined as follows:

$(\ell, v) \xrightarrow{a} (\ell', v')$ $\quad \exists (\ell, g, a, r, \ell').$

$v \models g \land v' = v[r]$

$(\ell, v) \xrightarrow{d} (\ell', v')$

$\ell' = \ell \land v' = v + d \land v \models I(\ell) \land v' \models I(\ell)$

Timed Automata – Semantics

Let $A = (L, \ell_0, E, I)$ be a timed automaton.

Timed transition system generated by A

$T(A) = (Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ where

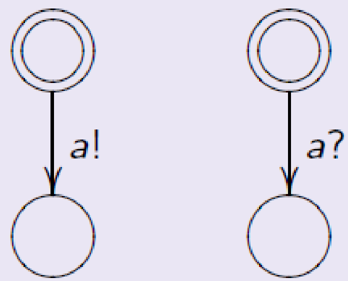
- $Proc = L \times (C \to \mathbb{R}^{\geq 0})$, i.e. states are of the form (ℓ, v) where ℓ is a location and v a valuation
- $Act = N \cup \mathbb{R}^{\geq 0}$
- $\longrightarrow$ is defined as follows:

$(\ell, v) \xrightarrow{a} (\ell', v')$ if there is $(\ell \xrightarrow{g,a,r} \ell') \in E$ s.t. $v \models g$ and $v' = v[r]$

$(\ell, v) \xrightarrow{d} (\ell, v + d)$ for all $d \in \mathbb{R}^{\geq 0}$ s.t. $v \models I(\ell)$ and $v + d \models I(\ell)$

Networks of Timed Automata

Timed Automata in Parallel



Intuition in CCS

$$(a.Nil \mid \overline{a}.Nil) \smallsetminus \{a\}$$

closed system

Let C be a set of clocks and $Chan$ a set of channels.

We let $Act = N \cup \mathbb{R}^{\geq 0}$ where

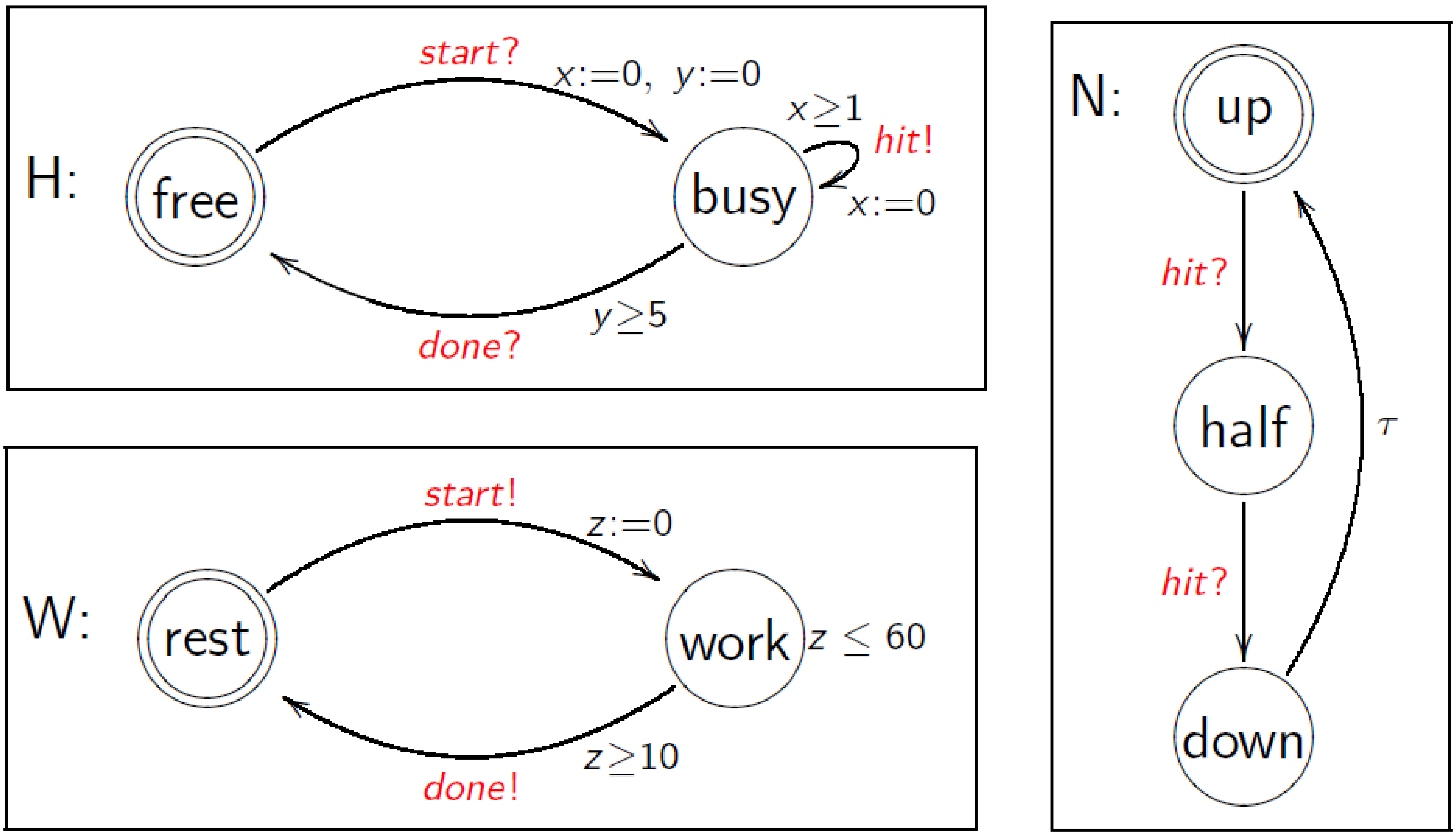
- $N = \{c! \mid c \in Chan\} \cup \{c? \mid c \in Chan\} \cup \{\tau\}$.

Let $A_i = (L_i, \ell_0^i, E_i, I_i)$ be timed automata for $1 \leq i \leq n$.

Networks of Timed Automata

We call $A = A_1 | A_2 | \cdots | A_n$ a networks of timed automata.

Example: Hammer, Worker, Nail



TLTS Generated by $A = A_1 | \dots | A_n$

$T(A) = (Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ where

- $Proc = (L_1 \times L_2 \times \cdots \times L_n) \times (C \to \mathbb{R}^{\geq 0})$, i.e. states are of the form $((\ell_1, \ell_2, \dots, \ell_n), v)$ where ℓ_i is a location in A_i
- $Act = \{\tau\} \cup \mathbb{R}^{\geq 0}$
- $\longrightarrow$ is defined as follows:

a) $((\ell_1, \dots, \ell_i, \dots, \ell_n), v) \xrightarrow{\tau} ((\ell_1, \dots, \ell_i', \dots, \ell_n), v')$ if there is $(\ell_i \xrightarrow{g,\tau,r} \ell_i') \in E_i$ s.t. $v \models g$ and $v' = v[r]$ and $v' \models I_i(\ell_i') \wedge \bigwedge_{k \neq i} I_k(\ell_k)$

b) $((\ell_1, \dots, \ell_n), v) \xrightarrow{d} ((\ell_1, \dots, \ell_n), v + d)$ for all $d \in \mathbb{R}^{\geq 0}$ s.t. $v \models \bigwedge_k I_k(\ell_k)$ and $v + d \models \bigwedge_k I_k(\ell_k)$

TLTS Generated by $A = A_1 | \ldots | A_n$

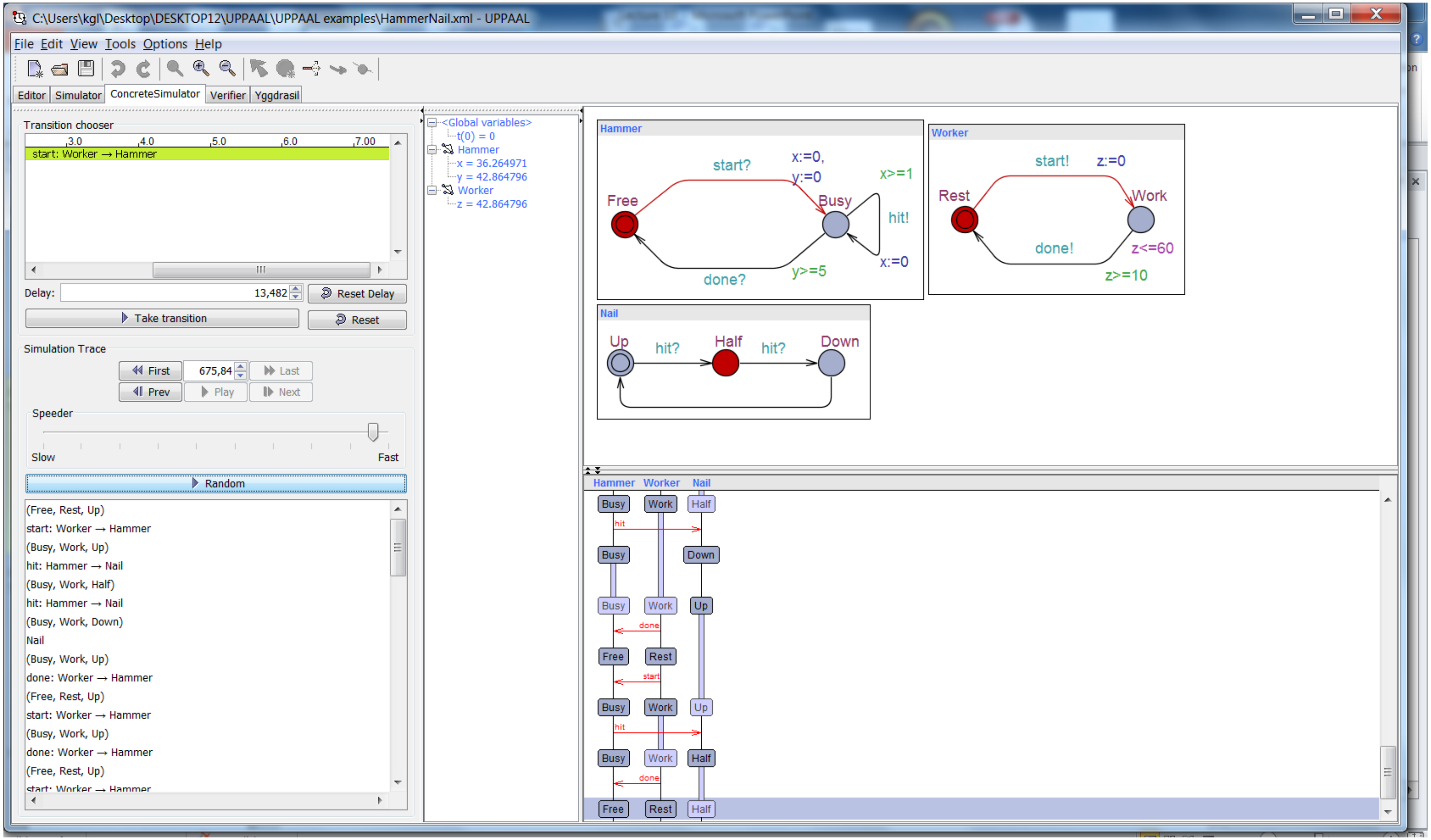
$T(A) = (Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ where

- $Proc = (L_1 \times L_2 \times \cdots \times L_n) \times (C \to \mathbb{R}^{\geq 0})$, i.e. states are of the form $((\ell_1, \ell_2, \ldots, \ell_n), v)$ where ℓ_i is a location in A_i
- $Act = \{\tau\} \cup \mathbb{R}^{\geq 0}$
- $\longrightarrow$ is defined as follows:

c) $((\ell_1, \ldots, \ell_i, \ldots, \ell_j, \ldots, \ell_n), v) \xrightarrow{\tau} ((\ell_1, \ldots, \ell_i', \ldots, \ell_j', \ldots, \ell_n), v')$

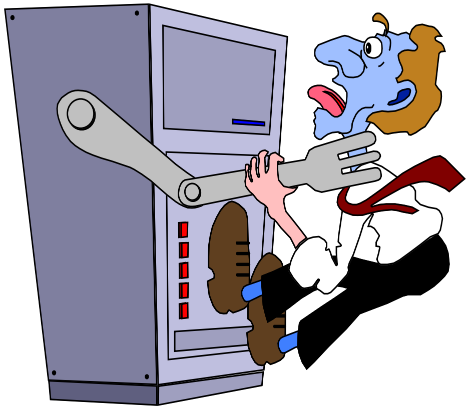
if $i \neq j$ and there are $(\ell_i \xrightarrow{g_i, a!, r_i} \ell_i') \in E_i$ and $(\ell_j \xrightarrow{g_j, a?, r_j} \ell_j') \in E_j$ s.t. $v \models g_i \wedge g_j$ and $v' = v[r_i \cup r_j]$ and $v' \models I_i(\ell_i') \wedge I_j(\ell_j') \wedge \bigwedge_{k \neq i,j} I_k(\ell_k)$

UPPAAL Demo


C:\Users\kgl\Desktop\DESKTOP12\UPPAAL\UPPAAL examples\HammerNail.xml - UPPAAL
File Edit View Tools Options Help
Editor Simulator ConcreteSimulator Verifier Yggdrasil
Transition chooser
start: Worker → Hammer
Delay:
13,482
Reset Delay
Take transition
Reset
Simulation Trace
First
675,84
Last
Prev
Play
Next
Speeder
Slow
Fast
Random
(Free, Rest, Up)
start: Worker → Hammer
(Busy, Work, Up)
hit: Hammer → Nail
(Busy, Work, Half)
hit: Hammer → Nail
(Busy, Work, Down)
Nail
(Busy, Work, Up)
done: Worker → Hammer
(Free, Rest, Up)
start: Worker → Hammer
(Busy, Work, Up)
done: Worker → Hammer
(Free, Rest, Up)
<Global variables>
t(0) = 0
Hammer
x = 36.264971
y = 42.864796
Worker
z = 42.864796
Hammer
start?
x:=0,
y:=0
x>=1
Free
Busy
hit!
x:=0
done?
y>=5
Worker
start!
z:=0
Rest
Work
done!
z<=60
z>=10
Nail
Up
hit?
Half
hit?
Down
Hammer Worker Nail
Busy Work Half
hit
Busy Down
Busy Work Up
done
Free Rest
start
Busy Work Up
hit
Busy Work Half
done
Free Rest Half

(Un)Timed Bisimulation

Equivalences ?



$$M =_{def} \ coin?\,(cof!\,\boldsymbol{M} + \tau.\,\boldsymbol{M})$$
$$\boldsymbol{R} \ =_{def} \ pub!\,coin!\,cof?\,\boldsymbol{R}$$
$$\boldsymbol{Sys} = (\,\boldsymbol{M} \mid \boldsymbol{R}\,)\{pub, coin\}$$

$$\boldsymbol{Spec} =_{def} \ pub!\,\tau.\,(\,\tau.\,\boldsymbol{Spec} + \tau.\,\boldsymbol{O})$$
$$\boldsymbol{Sys} \sim \boldsymbol{Spec}$$

$$\boldsymbol{M} =_{def} \ coin?\,(\,\epsilon(5).\,cof!\,\boldsymbol{M} + \epsilon(30).\,\tau.\,\boldsymbol{M})$$
$$\boldsymbol{R} \ =_{def} \ pub!\,coin!\,\epsilon(7).\,cof?\,\boldsymbol{R}$$
$$\boldsymbol{Sys} = (\,\boldsymbol{M} \mid \boldsymbol{R}\,)\{pub, coin\}$$

$$\boldsymbol{Spec} =_{def} \ pub!\,\tau.\,\epsilon(7).\,\tau.\,\boldsymbol{Spec}$$
$$\boldsymbol{Sys} \sim \boldsymbol{Spec}$$

$$\boldsymbol{WSpec} =_{def} \ pub!\,\epsilon(7).\,\boldsymbol{WSpec}$$
$$\boldsymbol{Sys} \approx \boldsymbol{WSpec}$$

Bisimulation

Definition A binary relation $\mathcal{R}$ over the set of states of an LTS is a *bisimulation* iff whenever $s_1 \mathcal{R} s_2$ and $\alpha \in Act$ then

i. If $s_1 \xrightarrow{\alpha} s_1'$ then $s_2 \xrightarrow{\alpha} s_2'$ with $s_1' \mathcal{R}\, s_2'$

ii. If $s_2 \xrightarrow{\alpha} s_2'$ then $s_1 \xrightarrow{\alpha} s_1'$ with $s_1' \mathcal{R}\, s_2'$

We write $s_1 \sim s_2$ iff there is a bisimulation $\mathcal{R}$ such that $s_1 \mathcal{R} s_2$

Timed Bisimulation

Definition A binary relation $\mathcal{R}$ over the set of states of an TLTS is a *timed bisimulation* iff whenever $s_1 \mathcal{R} s_2$ and $\alpha \in Act$ and $d \in \mathbb{R}$ then

i. If $s_1 \xrightarrow{\alpha} s_1'$ then $s_2 \xrightarrow{\alpha} s_2'$ with $s_1' \mathcal{R}\, s_2'$

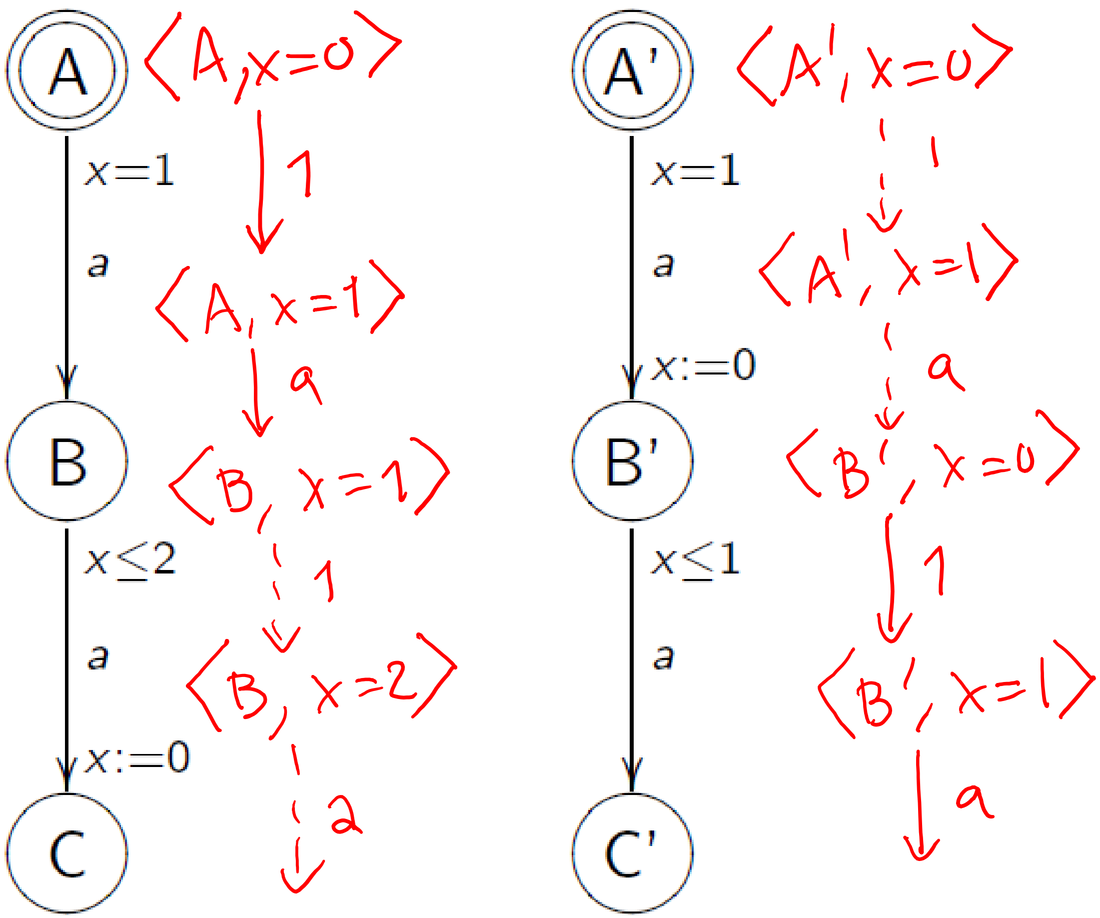
ii. If $s_1 \xrightarrow{d} s_1'$ then $s_2 \xrightarrow{d} s_2'$ with $s_1' \mathcal{R}\, s_2'$

iii. If $s_2 \xrightarrow{\alpha} s_2'$ then $s_1 \xrightarrow{\alpha} s_1'$ with $s_1' \mathcal{R}\, s_2'$

iv. If $s_2 \xrightarrow{d} s_2'$ then $s_1 \xrightarrow{d} s_1'$ with $s_1' \mathcal{R}\, s_2'$

We write $s_1 \sim s_2$ iff there is a timed bisimulation $\mathcal{R}$ such that $s_1 \mathcal{R} s_2$

Example 1



$\longrightarrow$ udfordrer

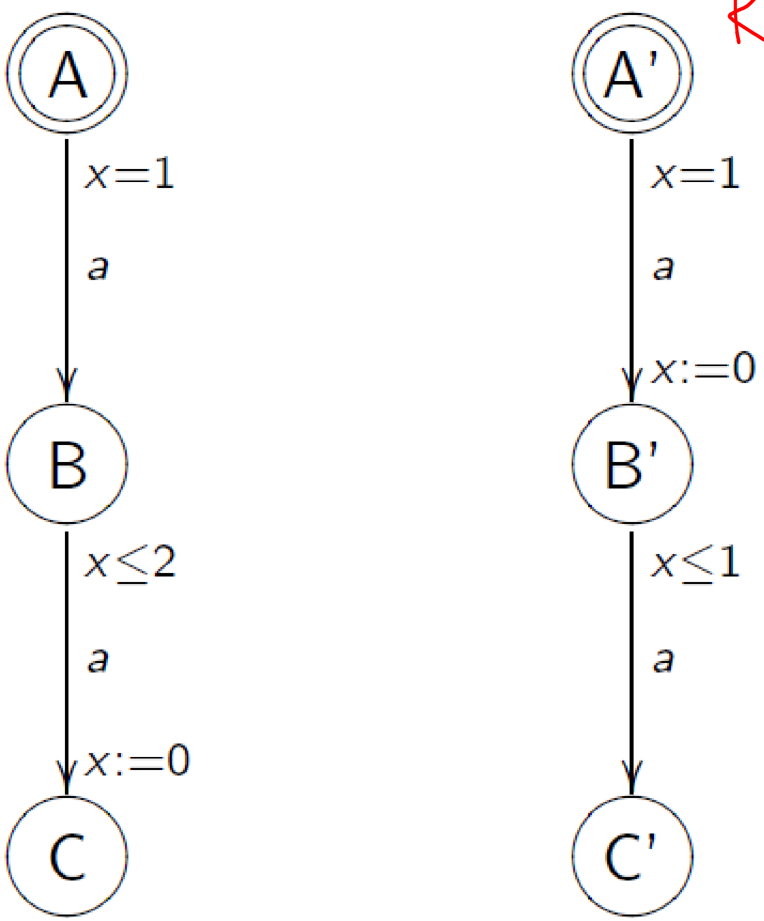
$\dashrightarrow$ forsvarer

$\dashrightarrow$ forsvarer vinder

$\Longrightarrow$

timed bisimular

Example 1

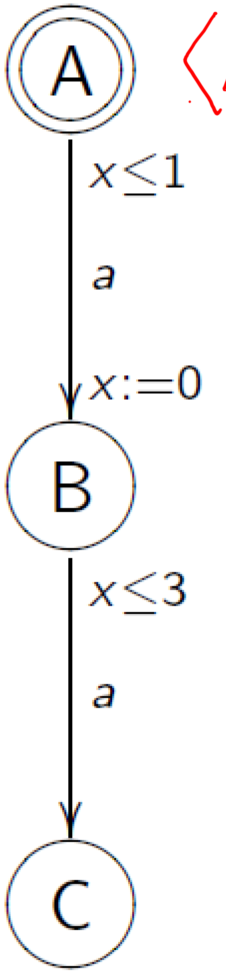


$$R = \{(\langle A, x=d\rangle, \langle A', x=d\rangle) \mid d \geq 0\}$$
$$\cup$$
$$\{(\langle B, x=1+d\rangle, \langle B', x=d\rangle) \mid d \geq 0\}$$
$$\cup$$
$$\{(\langle C, x=d\rangle, \langle C', x=e\rangle) \mid e, d \geq 0\}$$

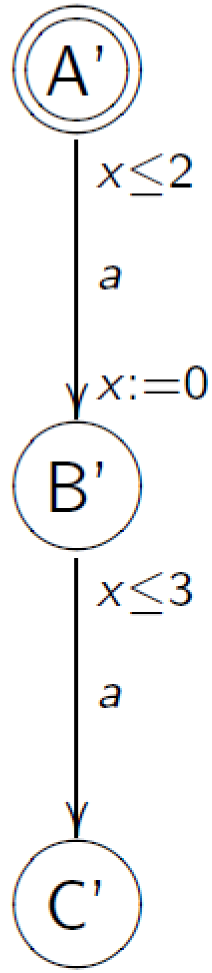
er en "timed bisimulation"

!

Example 2



$\langle A, x=0\rangle \xrightarrow{2} \langle A, x=2\rangle \not\xrightarrow{a}$



$\langle A', x=0\rangle \xrightarrow{2} \langle A', x=2\rangle \xrightarrow{a}$

$\rightarrow$ udfordrer

vinder

$\Longrightarrow$

IKKE timed bisimilar

Untimed Bisimulation

Definition A binary relation $\mathcal{R}$ over the set of states of an TLTS is an *untimed bisimulation* iff whenever $s_1 \mathcal{R} s_2$ and $\alpha \in Act$ and $d \in \mathbb{R}$ then

i. If $s_1 \xrightarrow{\alpha} s_1'$ then $s_2 \xrightarrow{\alpha} s_2'$ with $s_1' \mathcal{R}\, s_2'$

ii. If $s_1 \xrightarrow{d} s_1'$ then $s_2 \xrightarrow{d'} s_2'$ with $s_1' \mathcal{R}\, s_2'$ for some $d' \in \mathbb{R}$

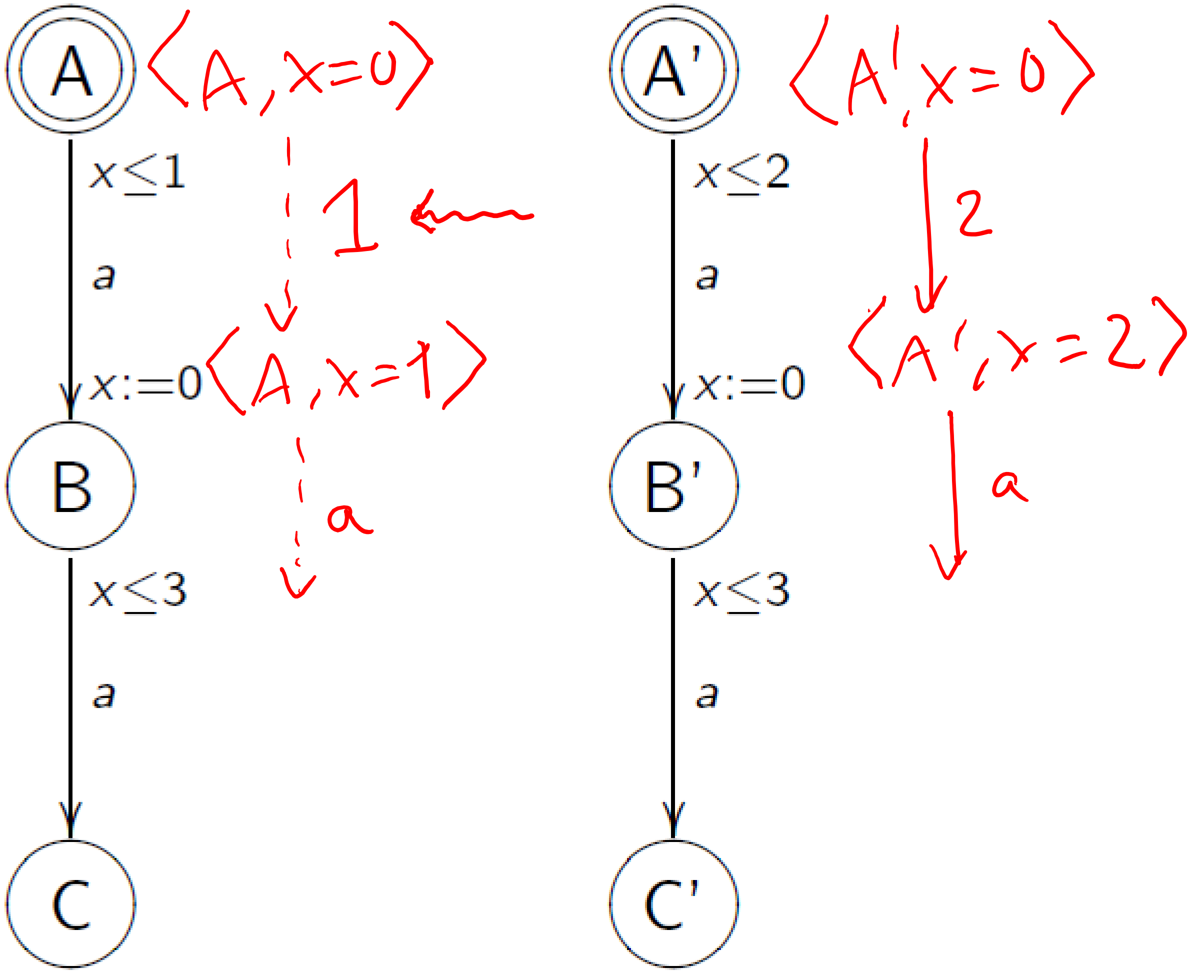
iii. If $s_2 \xrightarrow{\alpha} s_2'$ then $s_1 \xrightarrow{\alpha} s_1'$ with $s_1' \mathcal{R}\, s_2'$

iv. If $s_2 \xrightarrow{d} s_2'$ then $s_1 \xrightarrow{d'} s_1'$ with $s_1' \mathcal{R}\, s_2'$ for some $d' \in \mathbb{R}$

We write $s_1 \sim_u s_2$ iff there is an untimed bisimulation $\mathcal{R}$ such that $s_1 \mathcal{R} s_2$

Corollary $s_1 \sim s_2$ implies $s_1 \sim_u s_2$

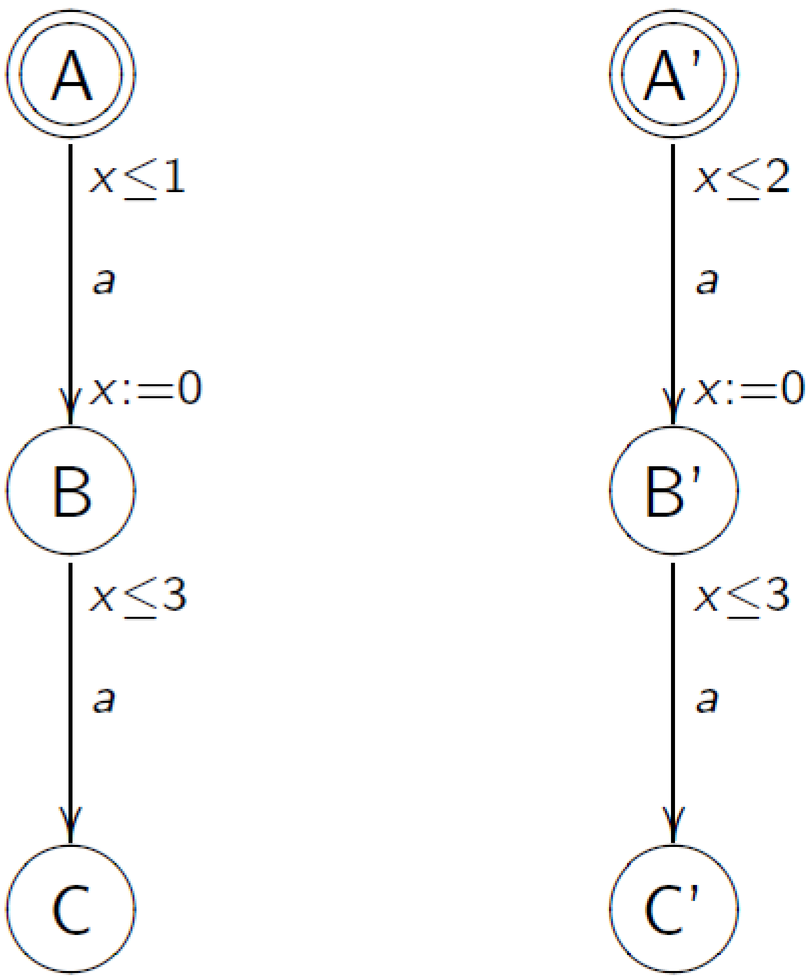
Example 2 – revised



- - -> vinder

$\Longrightarrow$ "untimed bisimilar"

Example 2 – revised



$R = \{ (\;,\;) \mid .. \}$

- - -

opskrives

"untimed bisimulation

OPGAVE for interesserede

Decidability of (un)timed bisimulation

Theorem [Cerans'92]

Timed bisimilarity for timed automata is decidable in EXPTIME (deterministic exponential time).

Theorem [Larsen, Wang'93]

Untimed bisimilarity for timed automata is decidable in EXPTIME (deterministic exponential time).

Weak Timed Bisimulation

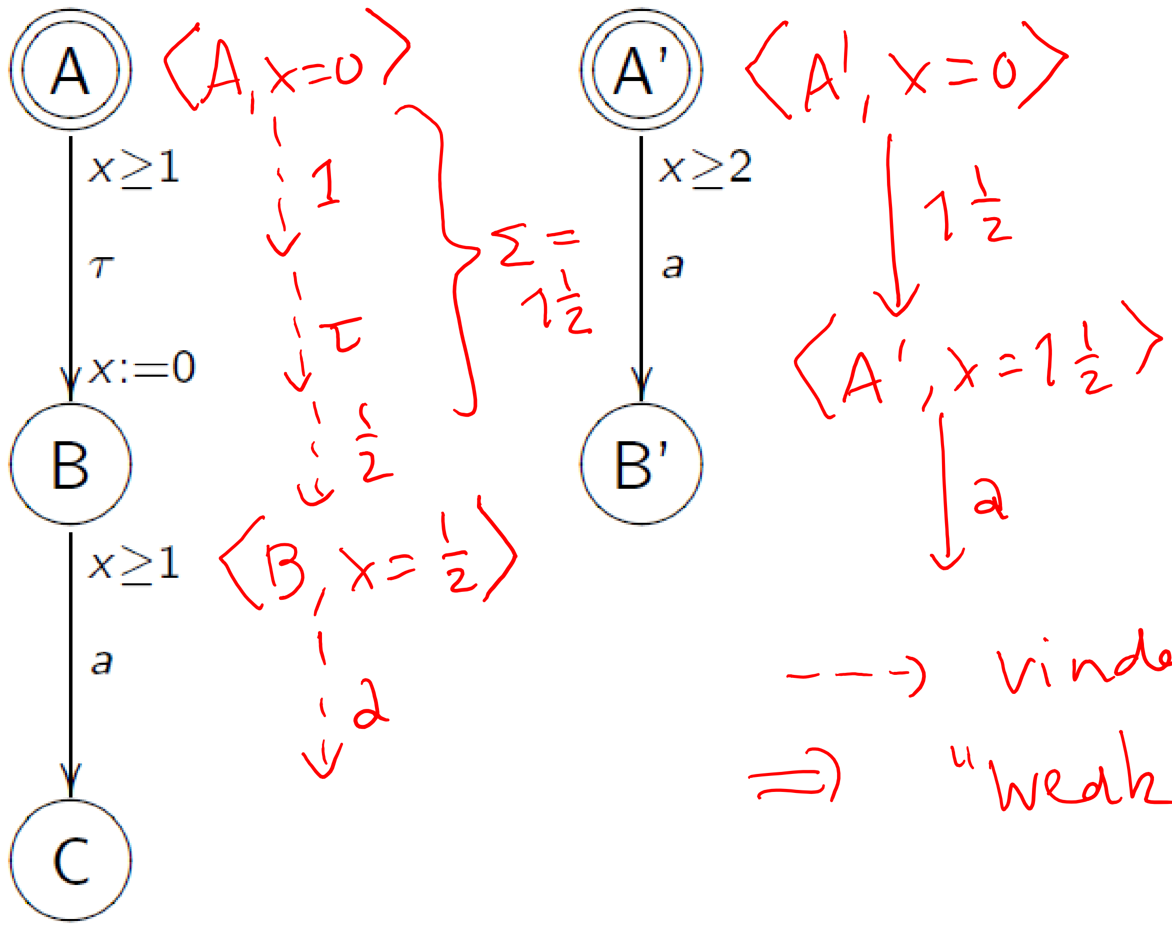
Weak Transition Relation

We introduce the following derived transition relations:

- $s \overset{a}{\Longrightarrow} s'$ iff $s \xrightarrow{\tau}{}^{*} \xrightarrow{a} \xrightarrow{\tau}{}^{*} s'$ when a is a discrete action.
- $s \overset{d}{\Longrightarrow} s'$ iff $s \xrightarrow{\tau}{}^{*} \xrightarrow{d_1} \xrightarrow{\tau}{}^{*} \cdots \xrightarrow{\tau}{}^{*} \xrightarrow{d_n} \xrightarrow{\tau}{}^{*} s'$ with $d = d_1 + d_2 + \cdots + d_n$.

Weak Timed Bisimilarity

Let A_1 and A_2 be two timed automata. We say that A_1 and A_2 are weakly timed bisimilar iff the transition systems $T(A_1)$ and $T(A_2)$ generated by A_1 and A_2 using weak transitions $\overset{a}{\Longrightarrow}$ and $\overset{d}{\Longrightarrow}$ are strongly bisimilar.



---> vinder

⟹ "weak timed bisimilar"

Timed Traces

Let $A = (L, \ell_0, E, I)$ be a timed automaton over a set of clocks C and a set of labels N.

Timed Traces

A sequence $(t_1, a_1)(t_2, a_2)(t_3, a_3)\ldots$ where $t_i \in \mathbb{R}^{\geq 0}$ and $a_i \in N$ is called a timed trace of A iff there is a transition sequence

$$(\ell_0, v_0) \xrightarrow{d_1} . \xrightarrow{a_1} . \xrightarrow{d_2} . \xrightarrow{a_2} . \xrightarrow{d_3} . \xrightarrow{a_3} \ldots$$

in A such that $v_0(x) = 0$ for all $x \in C$ and

$$t_i = t_{i-1} + d_i \qquad \text{where } t_0 = 0.$$

Intuition: t_i is the absolute time (time-stamp) when a_i happened since the start of the automaton A.

Timed & Untimed Trace Equivalence

The set of all timed traces of an automaton A is denoted by $L(A)$ and called the timed language of A.

Theorem [Alur, Courcoubetis, Dill, Henzinger'94]

Timed language equivalence (the problem whether $L(A_1) = L(A_2)$ for given timed automata A_1 and A_2) is undecidable.

We say that $a_1 a_2 a_3 \ldots$ is an untimed trace of A iff there exist $t_1, t_2, t_3, \ldots \in \mathbb{R}^{\geq 0}$ such that $(t_1, a_1)(t_2, a_2)(t_3, a_3) \ldots$ is a timed trace of A.

Theorem [Alur, Dill'94]

Untimed language equivalence for timed automata is decidable.

Automatic Verification of Timed Automata

Region Graphs

Automatic Verification of TA

Fact

Even very simple timed automata generate timed transition systems with infinitely (even uncountably) many reachable states.

Question

Is any automatic verification approach (like bisimilarity checking, model checking or reachability analysis) possible at all?

Answer

Yes, using region graph techniques.

Key idea: infinitely many clock valuations can be categorized into finitely many equivalence classes.

Intuition

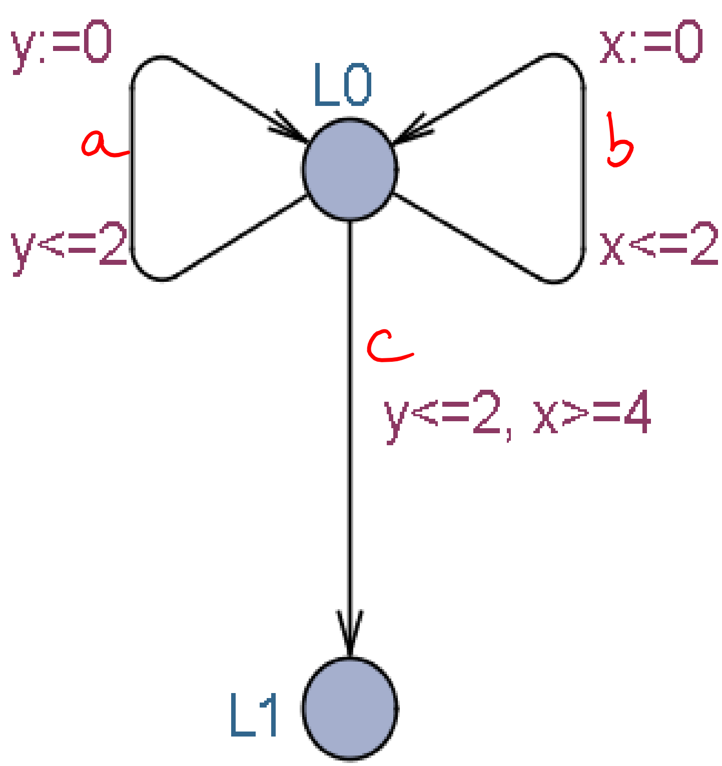
Let $v, v' : C \longrightarrow \mathbb{R}^{\geq 0}$ be clock valuations.
Let $\sim$ denote untimed bisimilarity of timed transition systems.

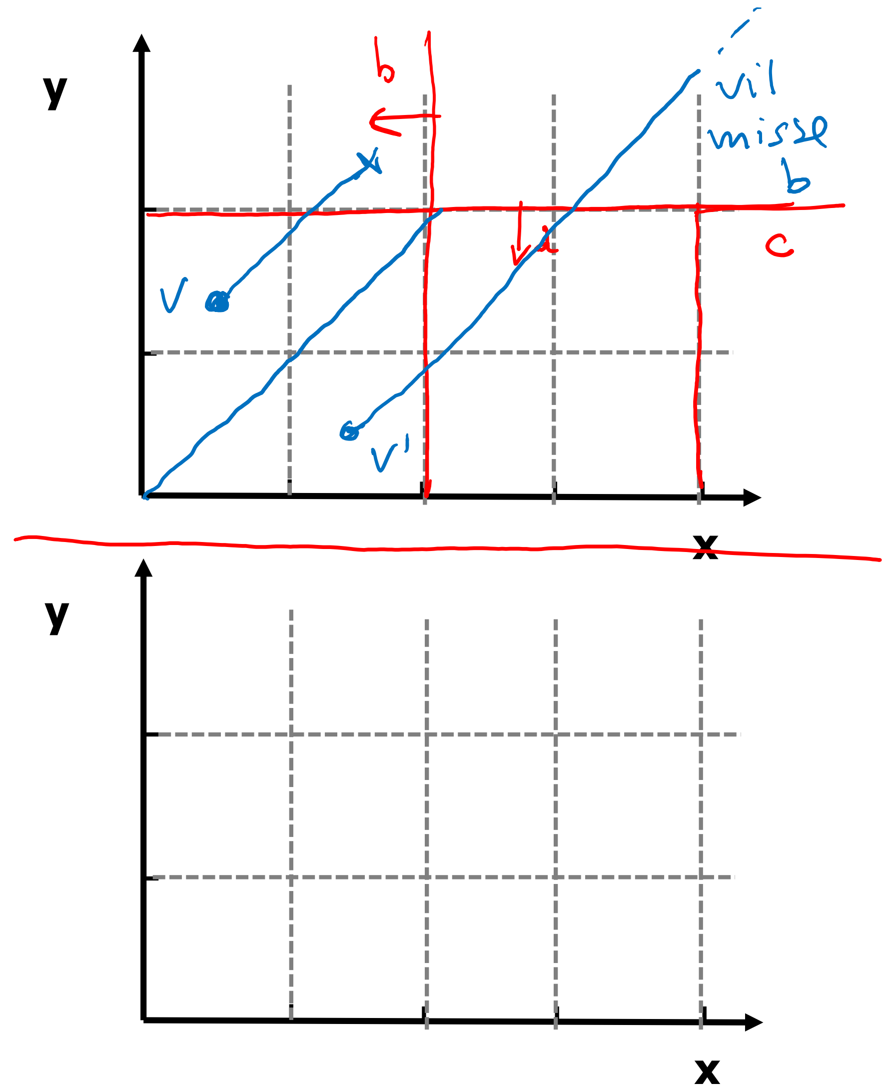
Our Aim

Define an equivalence relation $\equiv$ over clock valuations such that

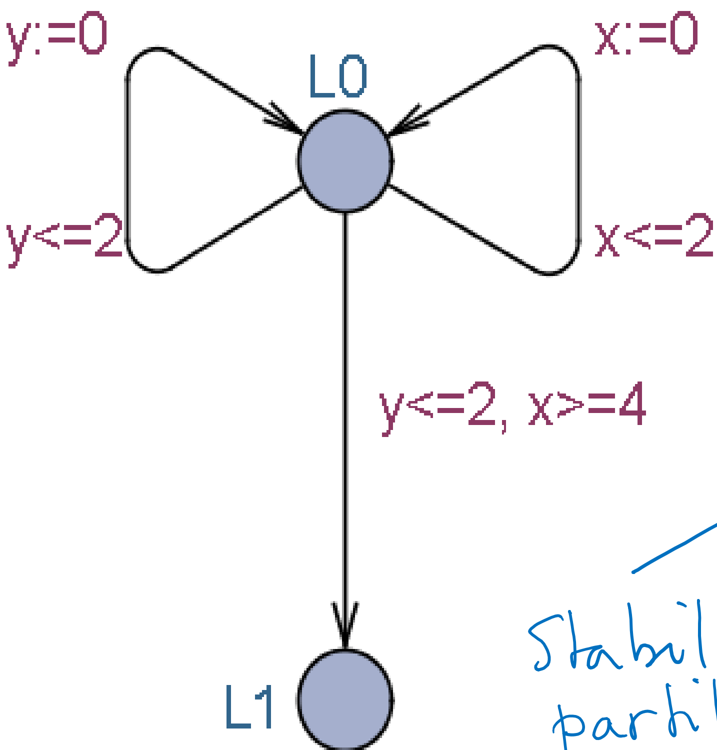
1. $v \equiv v'$ implies $(\ell, v) \sim (\ell, v')$ for any location ℓ
2. $\equiv$ has only finitely many equivalence classes.

Untimed Bisimulation Partitioning

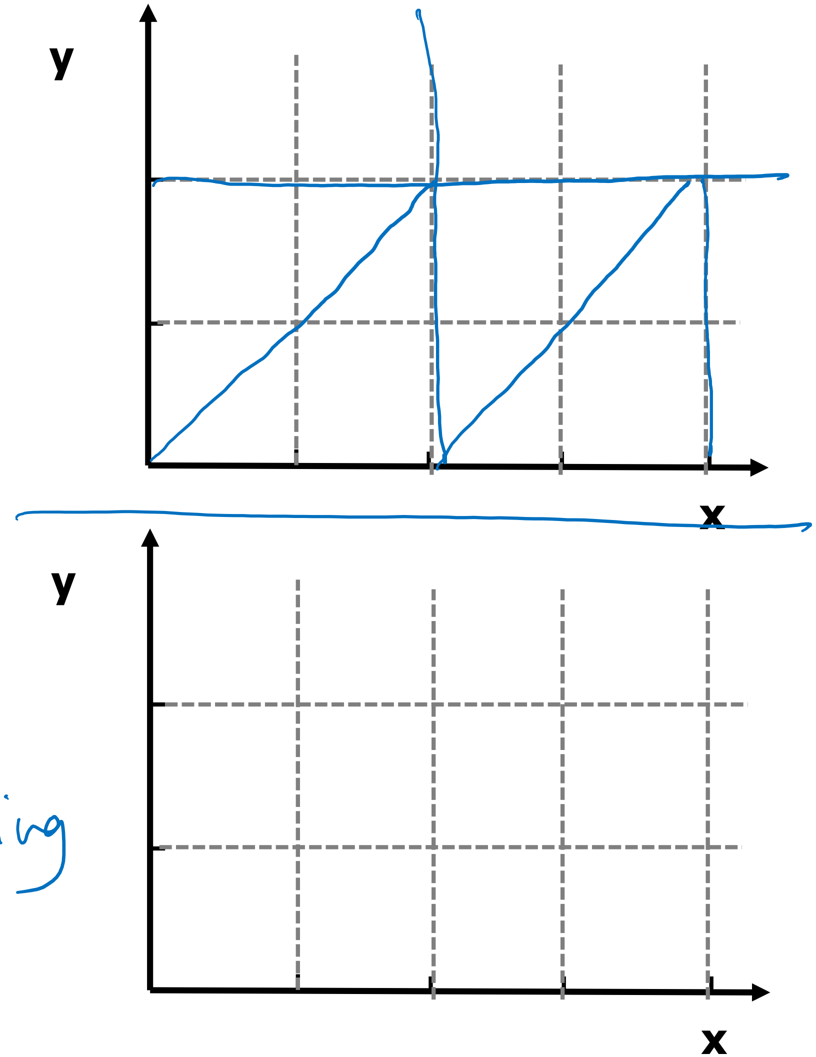




Untimed Bisimulation Partitioning



Stabil
partitionering



Preliminaries

Let $d \in \mathbb{R}^{\geq 0}$. Then

- let $\lfloor d \rfloor$ be the integer part of d, and
- let $frac(d)$ be the fractional part of d.

Any $d \in \mathbb{R}^{\geq 0}$ can be now written as $d = \lfloor d \rfloor + frac(d)$.

Example: $\lfloor 2.345 \rfloor = 2$ and $frac(2.345) = 0.345$.

Let A be a timed automaton and $x \in C$ be a clock. We define

$$c_x \in \mathbb{N}$$

as the largest constant with which the clock x is ever compared either in the guards or in the invariants present in A.

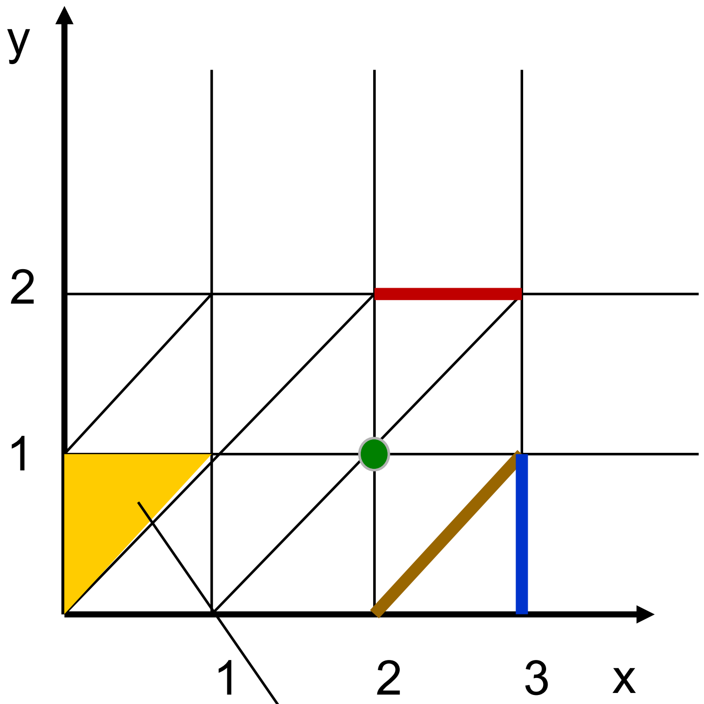
Clock (Region) Equivalence

Equivalence Relation on Clock Valuations

Clock valuations v and v' are equivalent ($v \equiv v'$) iff

1. for all $x \in C$ such that $v(x) \leq c_x$ or $v'(x) \leq c_x$ we have
$$\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$$
2. for all $x \in C$ such that $v(x) \leq c_x$ we have
$$frac(v(x)) = 0 \quad \text{iff} \quad frac(v'(x)) = 0$$
3. for all $x, y \in C$ such that $v(x) \leq c_x$ and $v(y) \leq c_y$ we have
$$frac(v(x)) \leq frac(v(y)) \quad \text{iff} \quad frac(v'(x)) \leq frac(v'(y))$$

Clock (Region) Equivalence



An equivalence class (i.e. a *region*)
in fact there is only a *finite* number of regions!!

Equivalence Relation on Clock Valuations

Clock valuations v and v' are equivalent ($v \equiv v'$) iff

1. for all $x \in C$ such that $v(x) \leq c_x$ or $v'(x) \leq c_x$ we have
$$\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$$
2. for all $x \in C$ such that $v(x) \leq c_x$ we have
$$frac(v(x)) = 0 \quad \text{iff} \quad frac(v'(x)) = 0$$
3. for all $x, y \in C$ such that $v(x) \leq c_x$ and $v(y) \leq c_y$ we have
$$frac(v(x)) \leq frac(v(y)) \quad \text{iff} \quad frac(v'(x)) \leq frac(v'(y))$$

Regions

Let v be a clock valuation. The $\equiv$-equivalence class represented by v is denoted by $[v]$ and defined by $[v] = \{v' \mid v' \equiv v\}$.

Definition of a Region

An $\equiv$-equivalence class $[v]$ represented by some clock valuation v is called a region.

Theorem

For every location ℓ and any two valuations v and v' from the same region ($v \equiv v'$) it holds that

$$(\ell, v) \sim (\ell, v')$$

where $\sim$ stands for untimed bisimilarity.

Symbolic States & Regions Graph

state (ℓ, v) $\rightsquigarrow$ symbolic state $(\ell, [v])$

Note: $v \equiv v'$ implies that $(\ell, [v]) = (\ell, [v'])$.

Region Graph

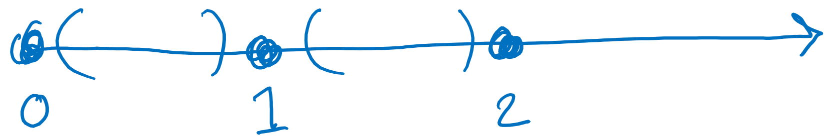
Region graph of a timed automaton A is an unlabelled (and untimed) transition system where

- states are symbolic states
- $\Longrightarrow$ between symbolic states is defined as follows:
  $(\ell, [v]) \Longrightarrow (\ell', [v'])$ iff $(\ell, v) \xrightarrow{a} (\ell', v')$ for some label a
  $(\ell, [v]) \Longrightarrow (\ell, [v'])$ iff $(\ell, v) \xrightarrow{d} (\ell, v')$ for some $d \in \mathbb{R}^{\geq 0}$

Fact

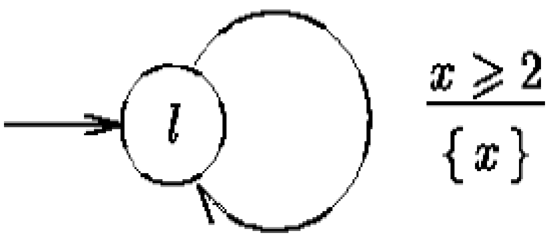
A region graph of any timed automaton is finite.

An Example Region Graph

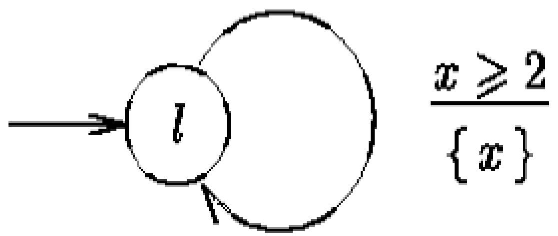


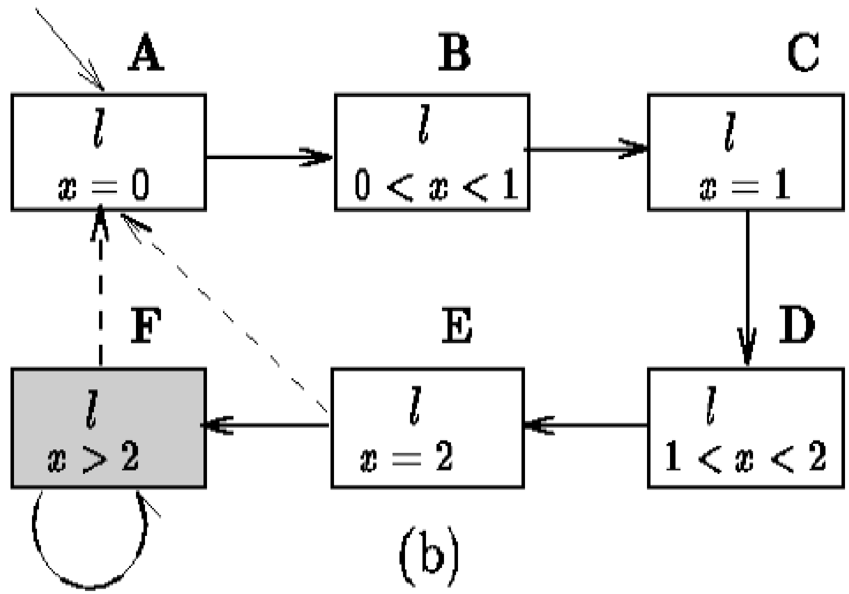
A B C D E F

regioner



An Example Region Graph





Application of Region Graph

Proc

Region graphs provide a natural abstraction which enables to prove decidability of e.g.

- reachability
- timed and untimed bisimilarity
- untimed language equivalence and language emptiness.

Application of Region Graph

Proc

Region graphs provide a natural abstraction which enables to prove decidability of e.g.

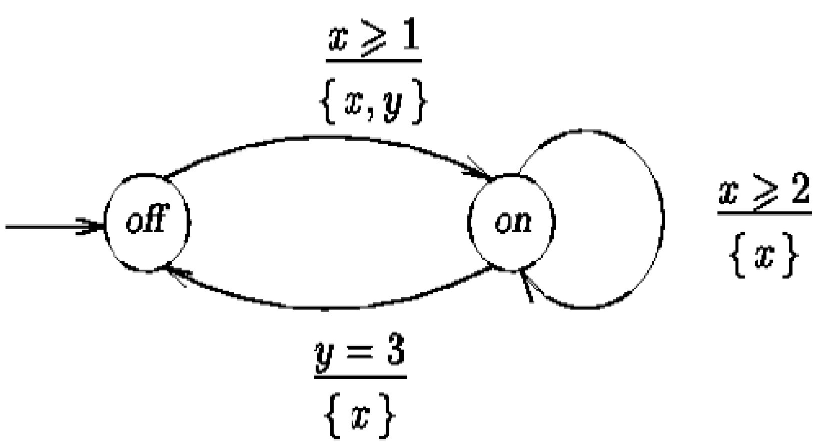
- reachability
- timed and untimed bisimilarity
- untimed language equivalence and language emptiness.

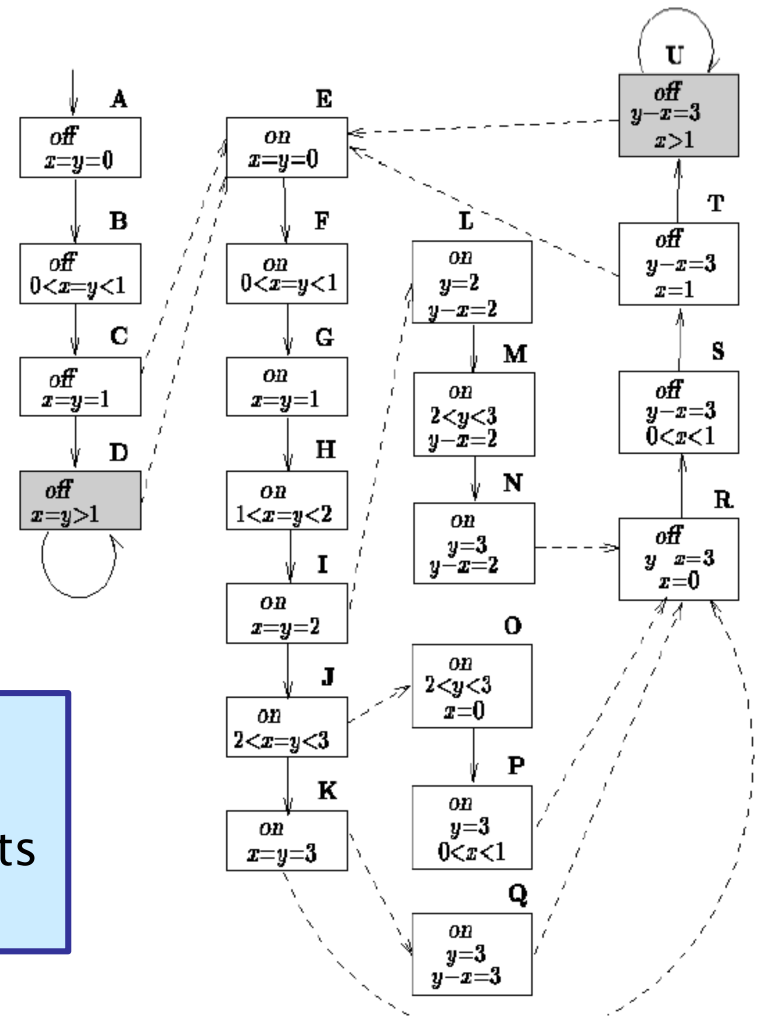
Cons

Region graphs have too large state spaces. State explosion is exponential in

- the number of clocks
- the maximal constants appearing in the guards.

Modified Switch





Property

Always *on* implies

Eventually *off* within 9 time units

????