Semantics and Verification 2010

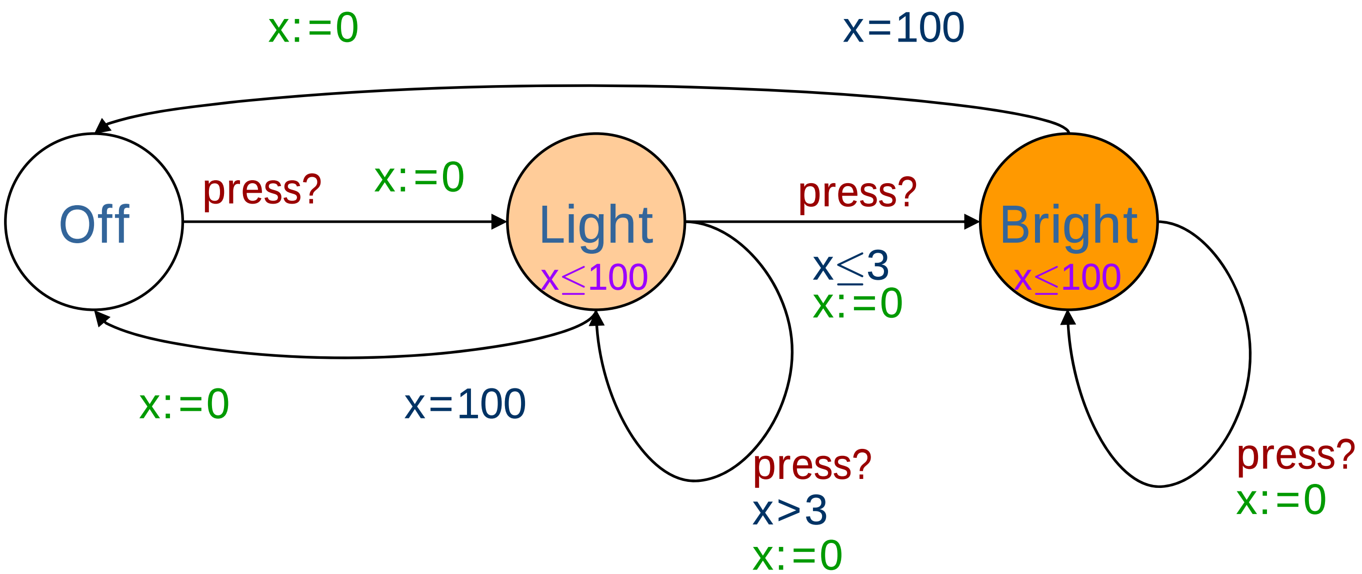
Lecture 10

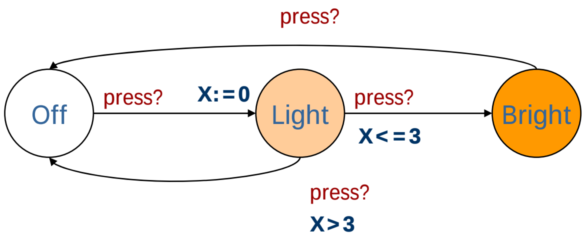
Overview

- Timed Automata – review
- Networks of Timed Automata
- Equivalence Checking Problems
- Region Graph and Reachability

Intelligent Light Control

Using Invariants





Definition of TA: Clock Constraints

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: $x \leq 3 \ \wedge \ y > 0 \ \wedge \ y - x = 2$

Clock Valuation

Clock valuation

Clock valuation v is a function $v : C \longrightarrow \mathbb{R}^{\geq 0}$.

Clock Valuation

Clock valuation

Clock valuation v is a function $v : C \rightarrow \mathbb{R}^{\geq 0}$.

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

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

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

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

Evaluation of Clock Constraints

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

$$
\begin{array}{ll}
v \models x < n & \text{iff } v(x) < n \\
v \models x \leq n & \text{iff } v(x) \leq n \\
v \models x = n & \text{iff } v(x) = n \\
\vdots & \\
v \models x - y < n & \text{iff } v(x) - v(y) < n \\
v \models x - y \leq n & \text{iff } v(x) - v(y) \leq n \\
\vdots & \\
v \models g_1 \wedge g_2 & \text{iff } v \models g_1 \text{ and } v \models g_2
\end{array}
$$

Syntax of Timed Automata

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 $(\ell, g, a, r, \ell') \in E$.

Semantics of Timed Automata

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

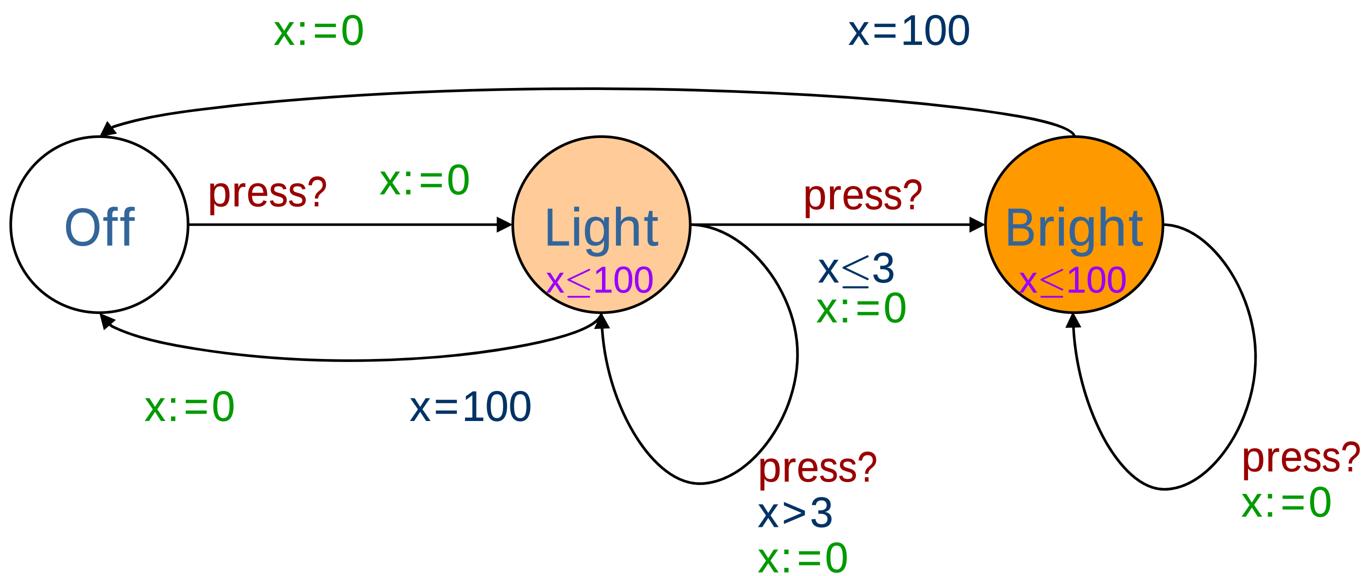
- $Proc = L \times (C \to \mathbb{R}^{\geq 0})$, i.e. states are of the form (ℓ, v) where ℓ is a location and v a valuation
- $Act = N \cup \mathbb{R}^{\geq 0}$
- $\longrightarrow$ is defined as follows:

$(\ell, v) \xrightarrow{a} (\ell', v')$ 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 $v + d \models I(\ell)$

Intelligent Light Control

Using Invariants



Transitions:

	(Off , x=0)
delay 4.32	→ (Off , x=4.32)
press?	→ (Light , x=0)
delay 4.51	→ (Light , x=4.51)
press?	→ (Light , x=0)
delay 100	→ (Light , x=100)
τ	→ (Off , x=0)

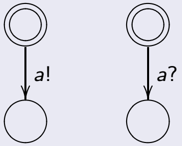
Note: X

(Light , x=0) delay 103 →

Invariants ensures progress

Networks of Timed Automata

Timed Automata in Parallel

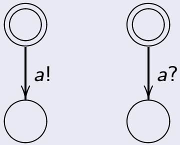


Intuition in CCS

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

Networks of Timed Automata

Timed Automata in Parallel



Intuition in CCS

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

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

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

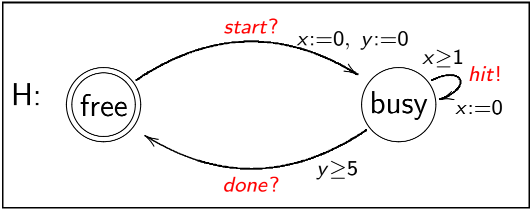
- $N = \{c! \mid c \in Chan\} \cup \{c? \mid c \in Chan\} \cup \{\tau\}$.

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

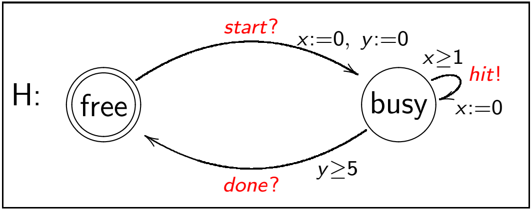
Networks of Timed Automata

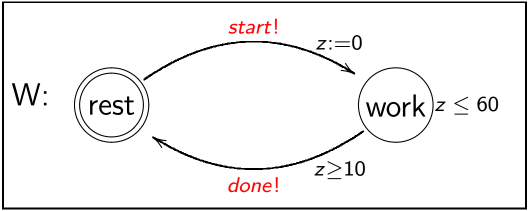
We call $A = A_1|A_2|\cdots|A_n$ a networks of timed automata.

Example: Hammer, Worker, Nail

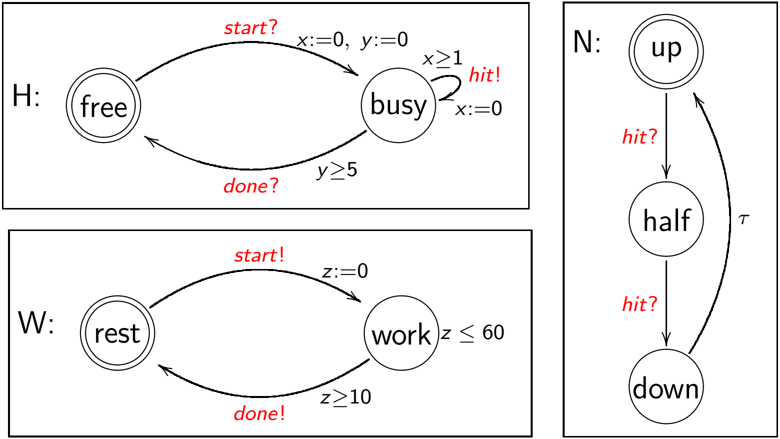


Example: Hammer, Worker, Nail





Example: Hammer, Worker, Nail



Timed Transition System Generated by $A = A_1 | \cdots | A_n$

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

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

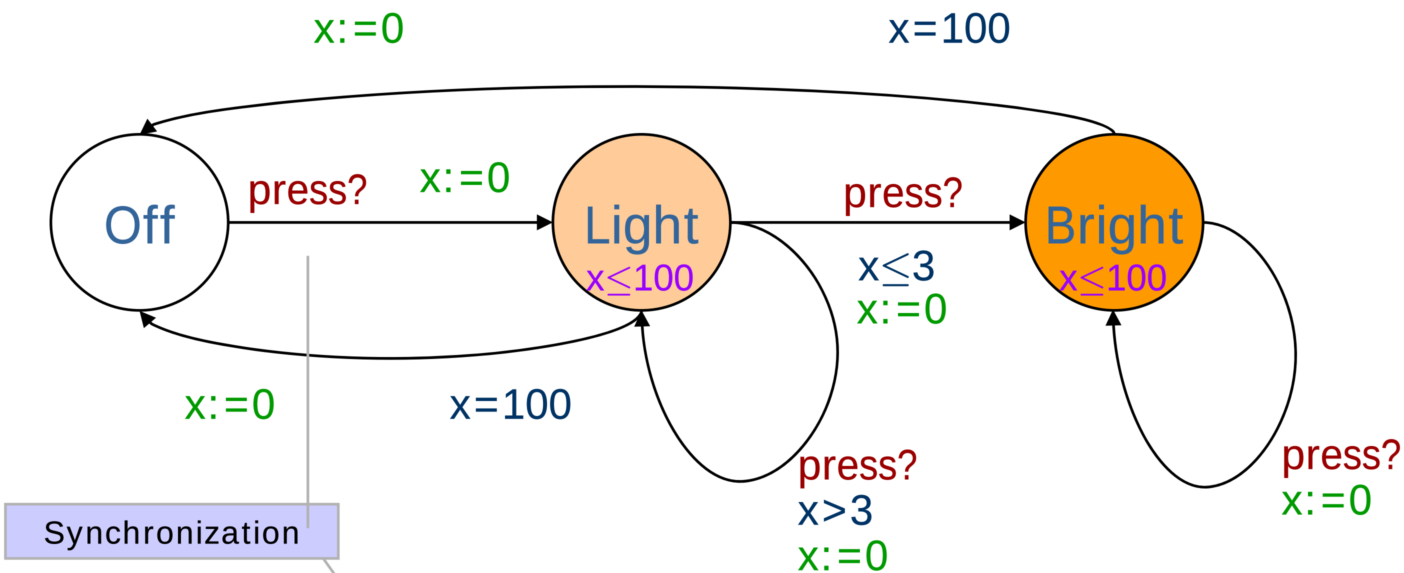
$((\ell_1, \ldots, \ell_n), v) \xrightarrow{d} ((\ell_1, \ldots, \ell_n), v + d)$ for all $d \in \mathbb{R}^{\geq 0}$ s.t. $v \models \bigwedge_k I_k(\ell_k)$ and $v + d \models \bigwedge_k I_k(\ell_k)$

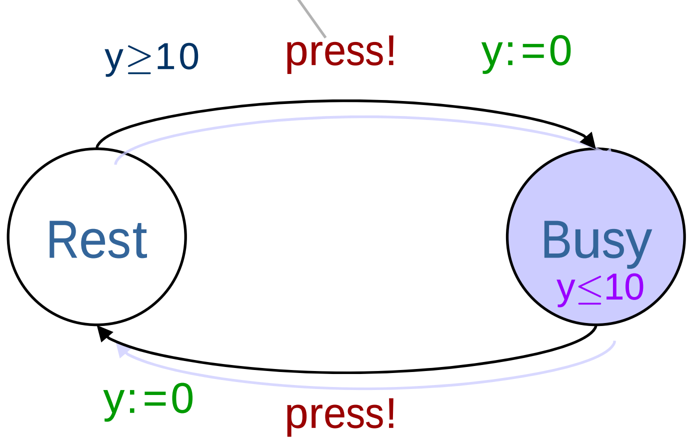
Continuation

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

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 $v' \models I_i(\ell_i') \wedge I_j(\ell_j') \wedge \bigwedge_{k \neq i,j} I_k(\ell_k)$

Networks Light Controller & User





Transitions:

	(Off, Rest, x=0, y=0)
delay 20	→ (Off, Rest, x=20, y=20)
press?!	→ (Light, Busy, x=0, y=0)
delay 2	→ (Light, Busy, x=2, y=2)
press?!	→ (Bright, Rest, x=0, y=0)

Timed Bisimilarity

Let A_1 and A_2 be timed automata.

Timed Bisimilarity

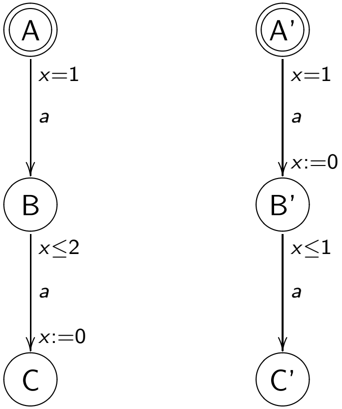
We say that A_1 and A_2 are timed bisimilar iff the transition systems $T(A_1)$ and $T(A_2)$ generated by A_1 and A_2 are strongly bisimilar.

Remark: both

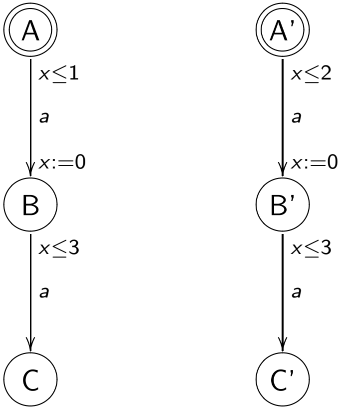
- $\xrightarrow{a}$ for $a \in Act$ and
- $\xrightarrow{d}$ for $d \in \mathbb{R}^{\geq 0}$

are considered as normal (visible) transitions.

Example of Timed Bisimilar Automata



Example of Timed Non-Bisimilar Automata



Untimed Bisimilarity

Let A_1 and A_2 be timed automata. Let ϵ be a new (fresh) action.

Untimed Bisimilarity

We say that A_1 and A_2 are untimed bisimilar iff the transition systems $T(A_1)$ and $T(A_2)$ generated by A_1 and A_2 where every transition of the form $\xrightarrow{d}$ for $d \in \mathbb{R}^{\geq 0}$ is replaced with $\xrightarrow{\epsilon}$ are strongly bisimilar.

Remark:

- $\xrightarrow{a}$ for $a \in N$ is treated as a visible transition, while
- $\xrightarrow{d}$ for $d \in \mathbb{R}^{\geq 0}$ are all labelled by a single visible action $\xrightarrow{\epsilon}$.

Untimed Bisimilarity

Let A_1 and A_2 be timed automata. Let ϵ be a new (fresh) action.

Untimed Bisimilarity

We say that A_1 and A_2 are untimed bisimilar iff the transition systems $T(A_1)$ and $T(A_2)$ generated by A_1 and A_2 where every transition of the form $\xrightarrow{d}$ for $d \in \mathbb{R}^{\geq 0}$ is replaced with $\xrightarrow{\epsilon}$ are strongly bisimilar.

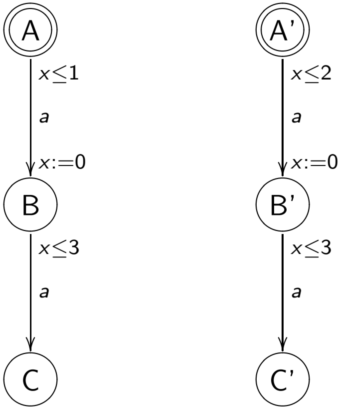
Remark:

- $\xrightarrow{a}$ for $a \in N$ is treated as a visible transition, while
- $\xrightarrow{d}$ for $d \in \mathbb{R}^{\geq 0}$ are all labelled by a single visible action $\xrightarrow{\epsilon}$.

Corollary

Any two timed bisimilar automata are also untimed bisimilar.

Timed Non-Bisimilar but Untimed Bisimilar Automata



Decidability of Timed and Untimed Bisimilarity

Theorem [Cerans'92]

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

Decidability of Timed and Untimed Bisimilarity

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

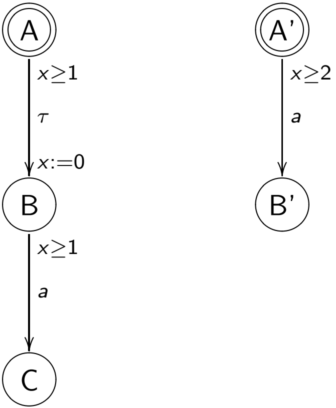
We introduce the following derived transition relations:

- $s \stackrel{a}{\Longrightarrow} s'$ iff $s \stackrel{\tau}{\longrightarrow}^{*} \stackrel{a}{\longrightarrow} \stackrel{\tau}{\longrightarrow}^{*} s'$ when a is a discrete action.
- $s \stackrel{d}{\Longrightarrow} s'$ iff $s \stackrel{\tau}{\longrightarrow}^{*} \stackrel{d_1}{\longrightarrow} \stackrel{\tau}{\longrightarrow}^{*} \cdots \stackrel{\tau}{\longrightarrow}^{*} \stackrel{d_n}{\longrightarrow} \stackrel{\tau}{\longrightarrow}^{*} s'$ 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 $\stackrel{a}{\Longrightarrow}$ and $\stackrel{d}{\Longrightarrow}$ are strongly bisimilar.

Weakly Timed Bisimilar Automata



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

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

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

Theorem [Alur, Dill'94]

Untimed language equivalence for timed automata is decidable.

Automatic Verification of Timed Automata

Fact

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

Automatic Verification of Timed Automata

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?

Automatic Verification of Timed Automata

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 $v, v' : C \to \mathbb{R}^{\geq 0}$ be clock valuations.

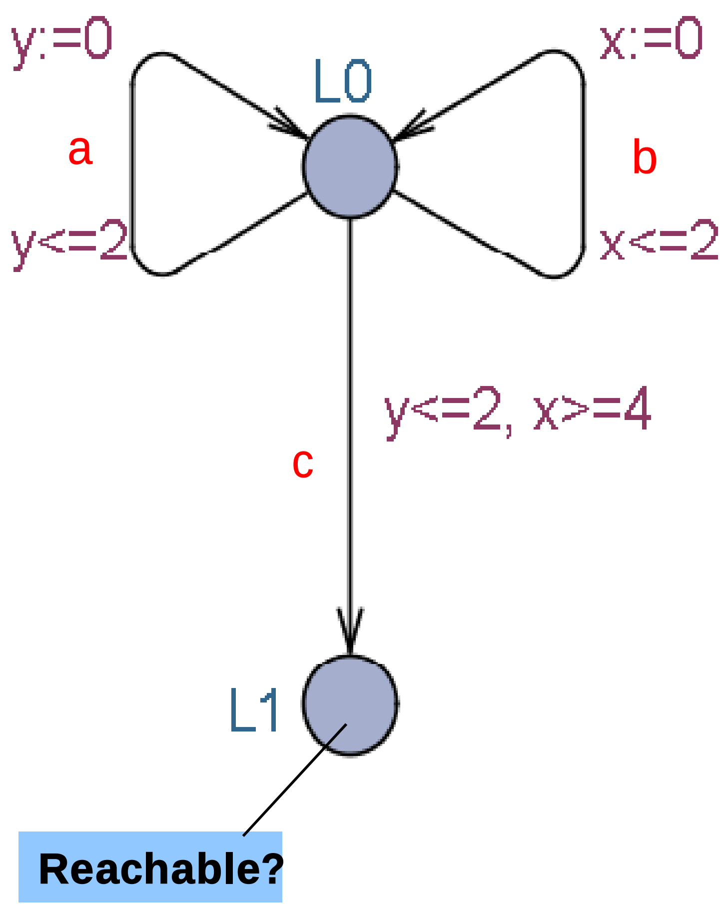
Let $\sim$ denote untimed bisimilarity of timed transition systems.

Our Aim

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

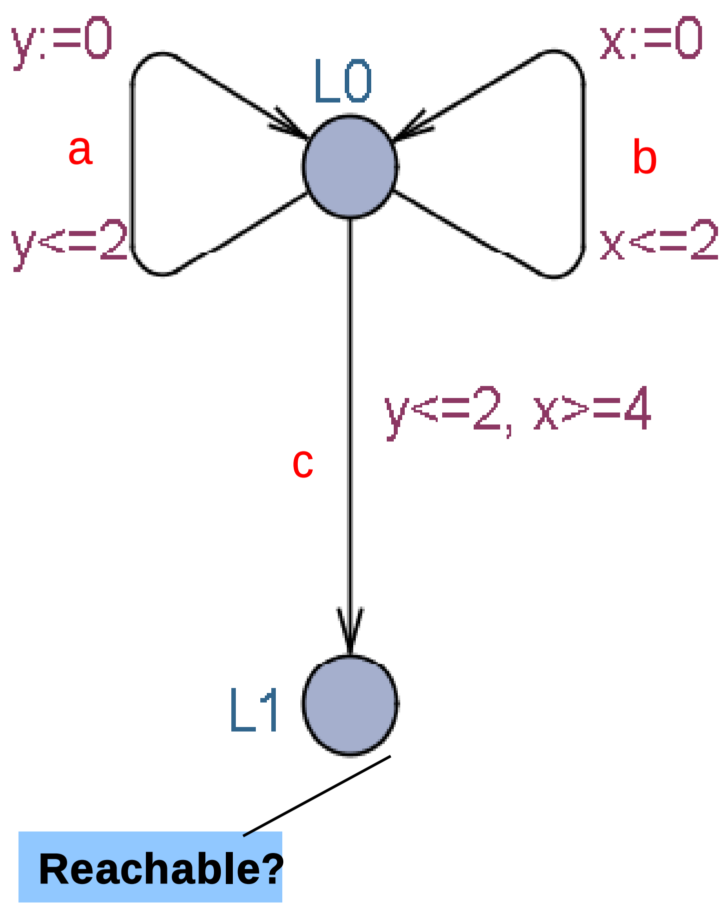
1. $v \equiv v'$ implies $(\ell, v) \sim (\ell, v')$ for any location ℓ
2. $\equiv$ has only finitely many equivalence classes.

Decidability ?

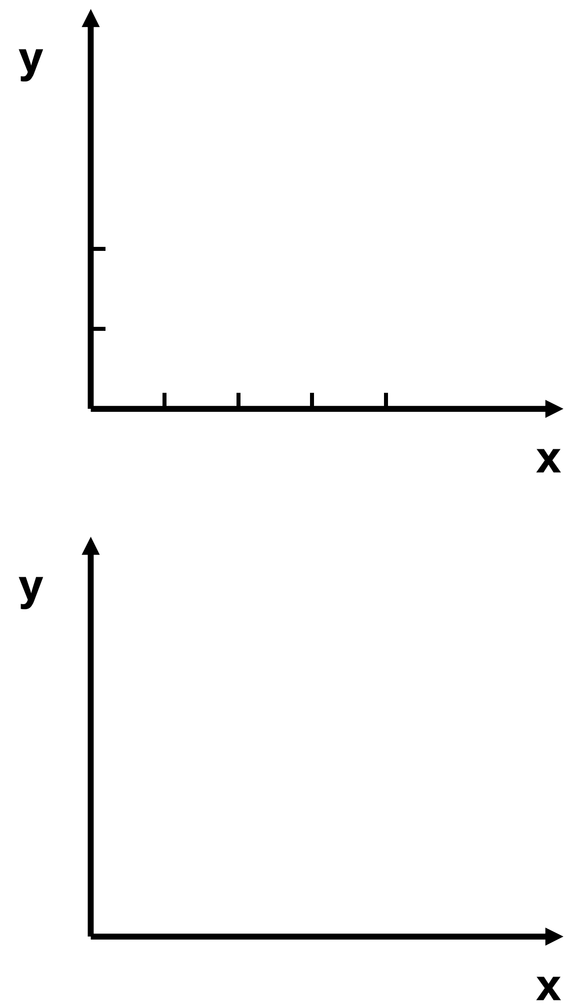


OBSTACLE:
Uncountably infinite state space

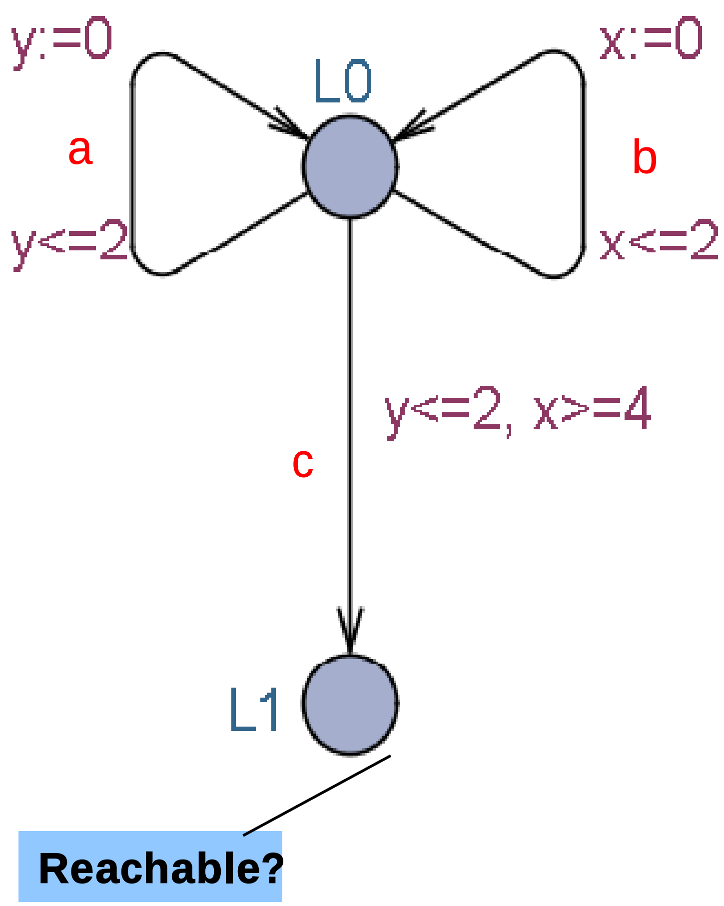
Stable Quotient



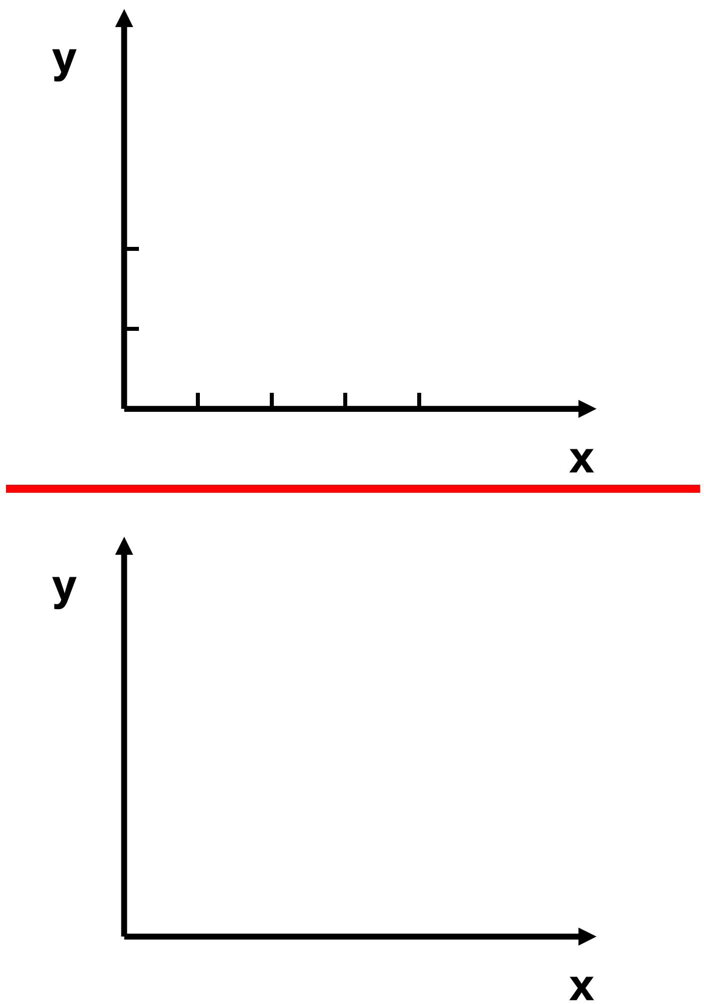
Partitioning



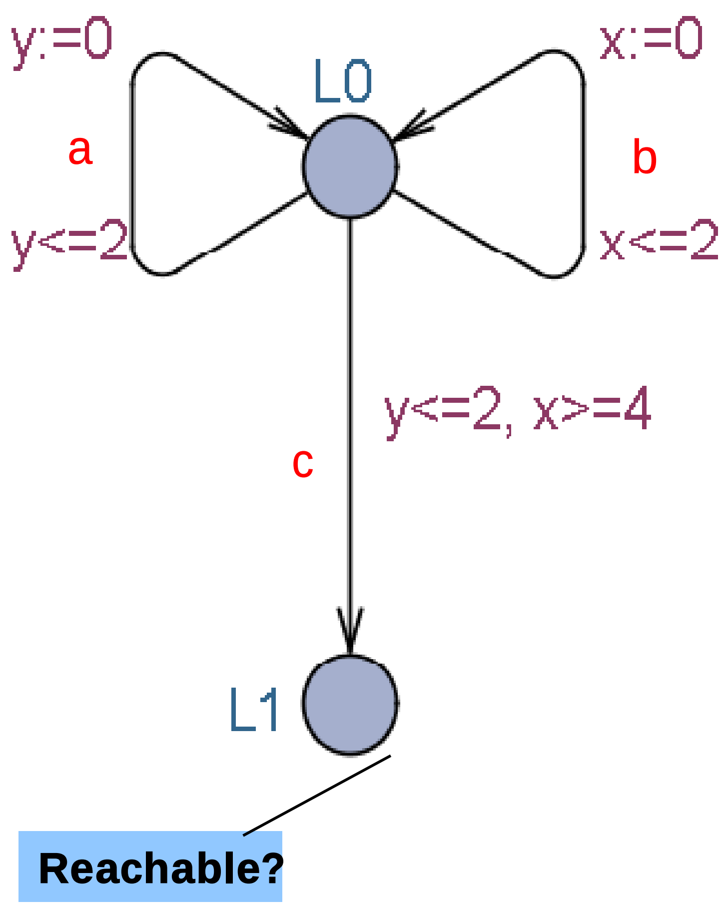
Stable Quotient

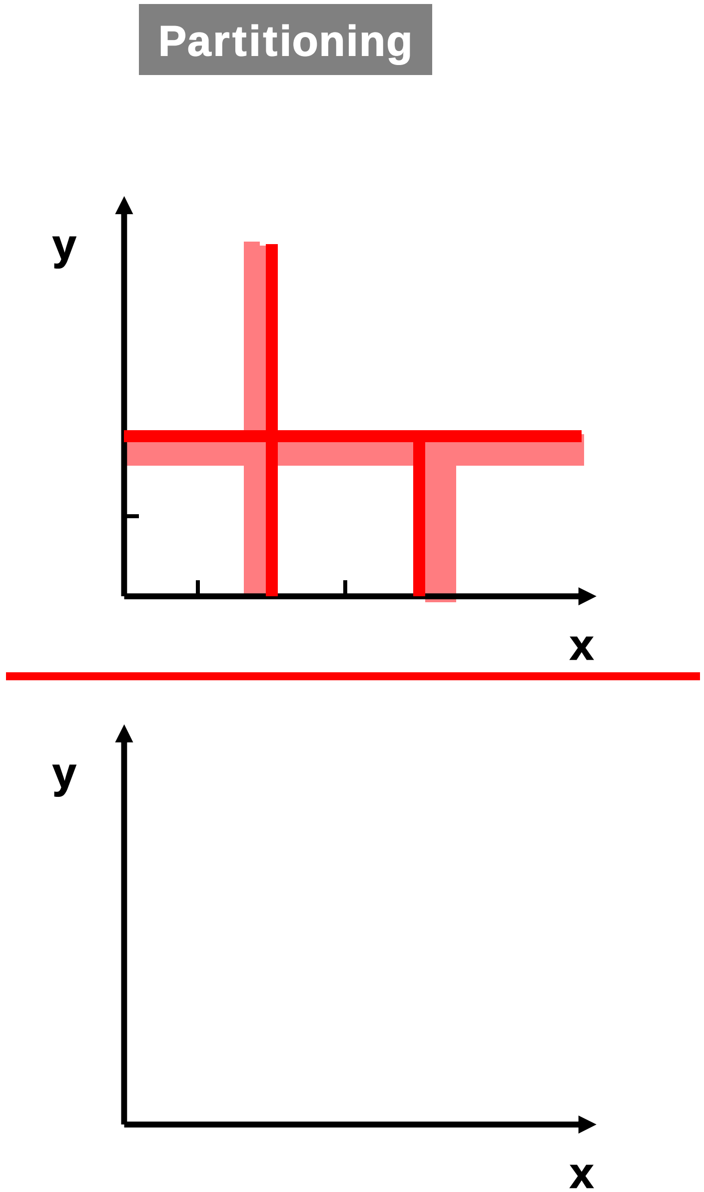


Partitioning

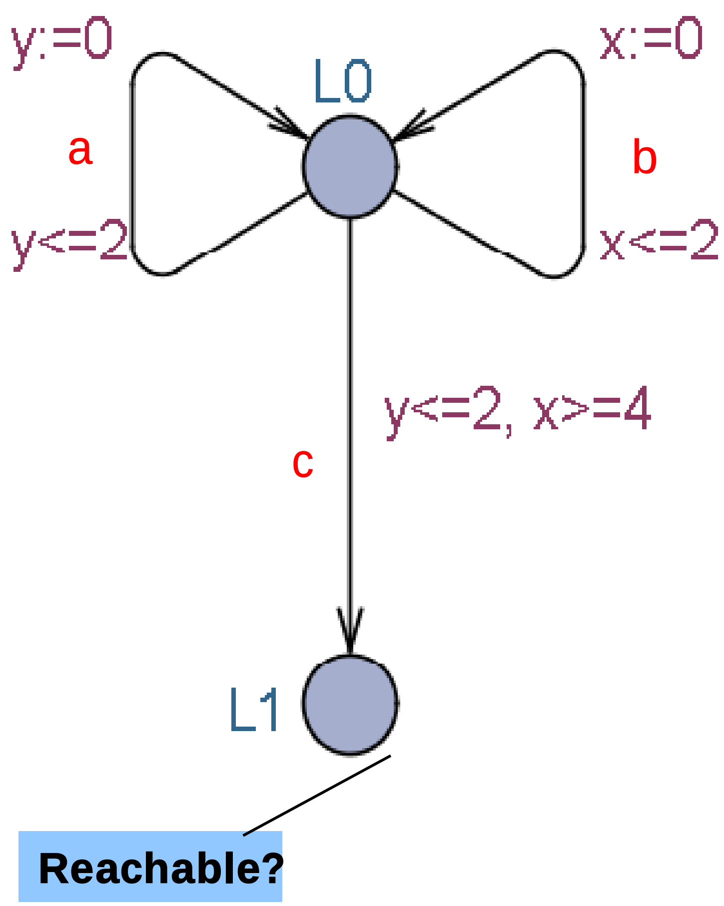


Stable Quotient

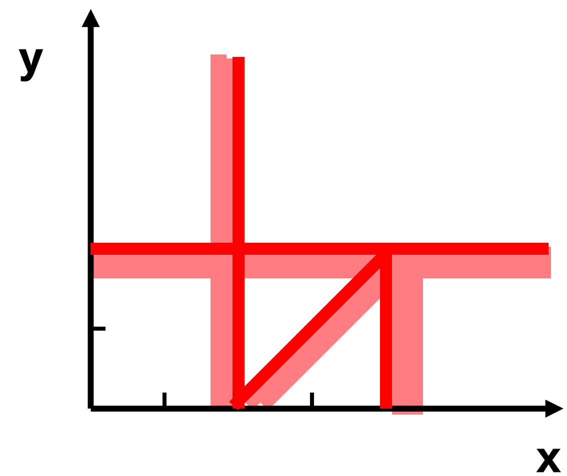


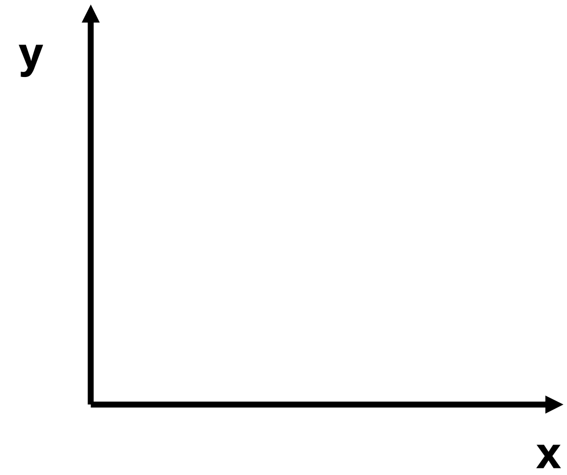


Stable Quotient

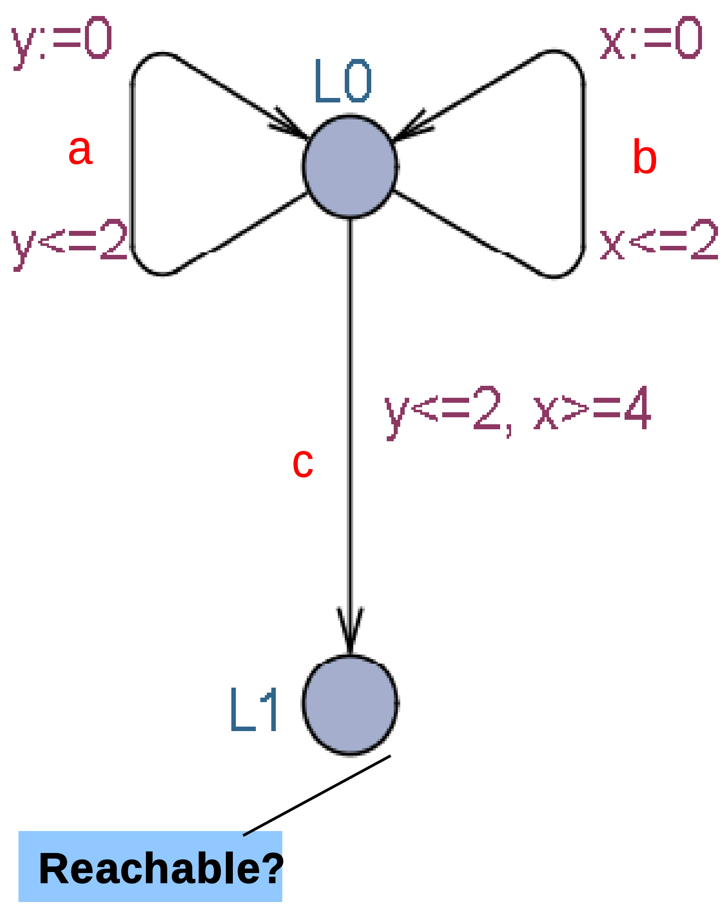


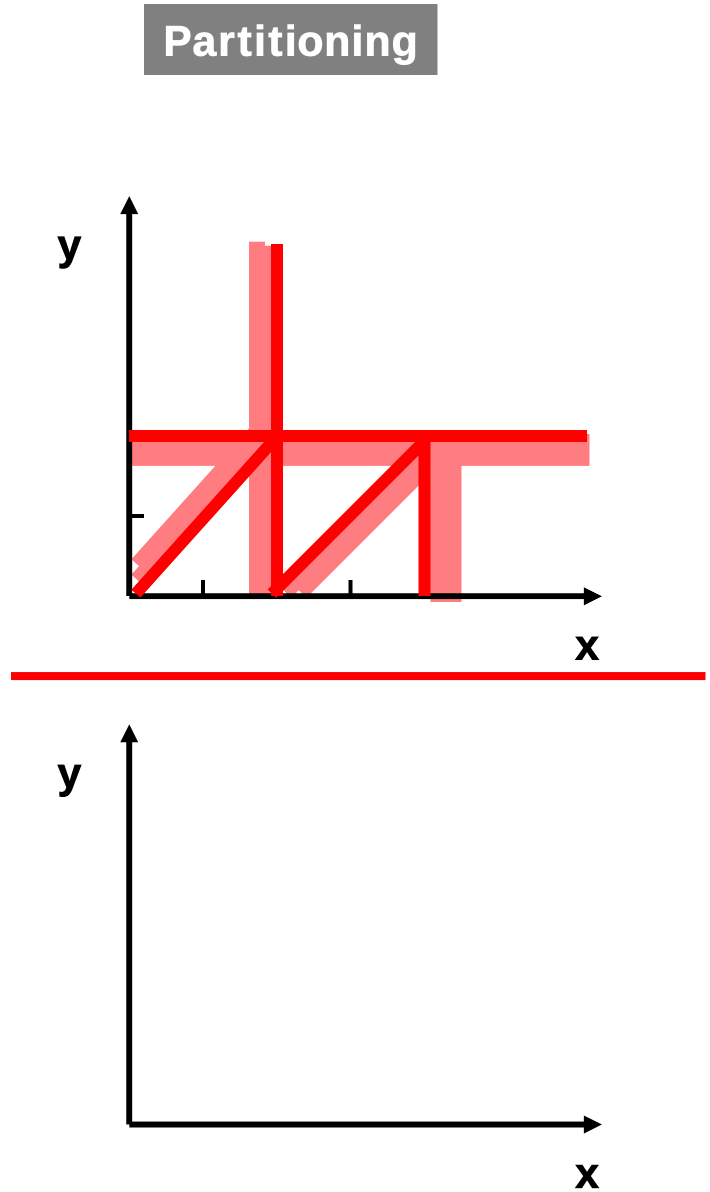
Partitioning





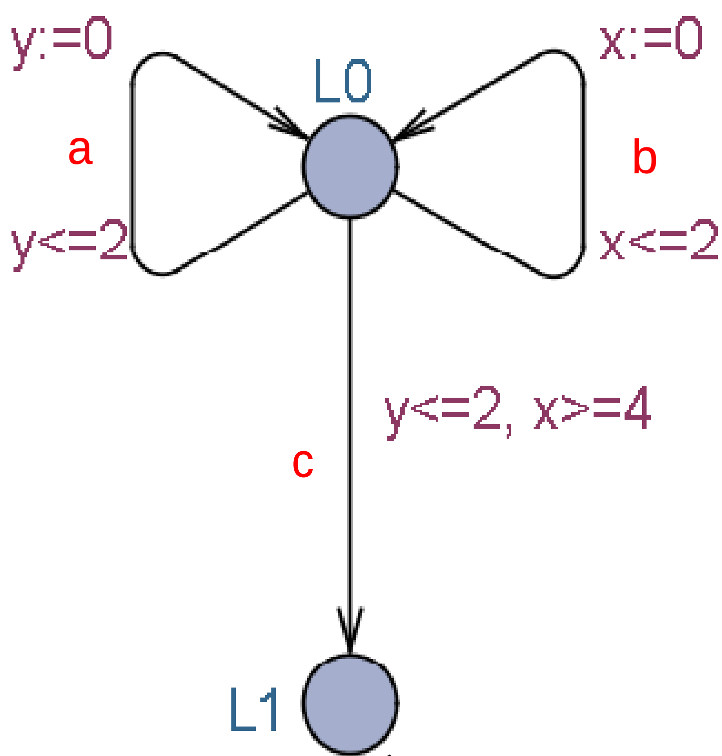
Stable Quotient

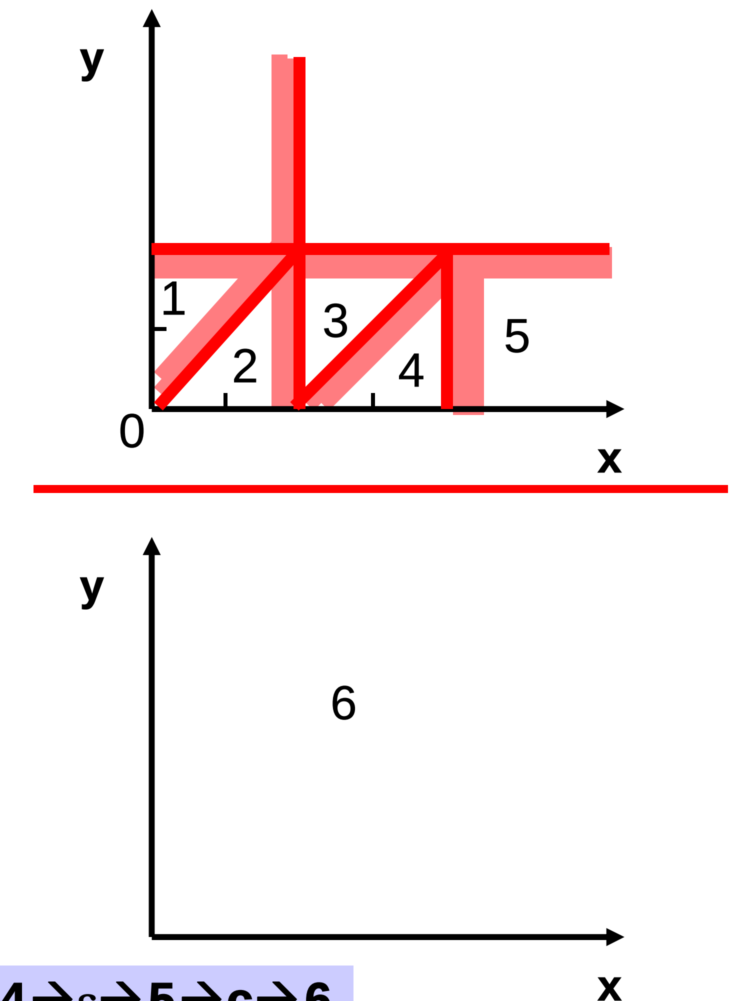




Stable Quotient

Partitioning





Reachable?

0 →ε→1→a→2→ε→3→a→4→ε→5→c→6

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 $frac(2.345) = 0.345$.

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

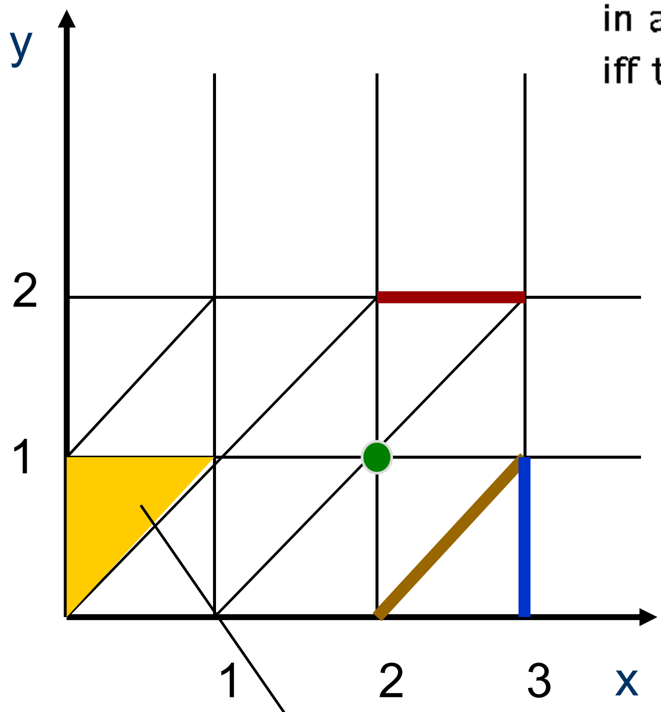
Equivalence Relation on Clock Valuations

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
$$frac(v(x)) \leq frac(v(y)) \quad \text{iff} \quad frac(v'(x)) \leq frac(v'(y))$$

Regions

Finite Partitioning of State Space



For each clock x let c_x be the largest integer with which x is compared in any guard or invariant of A. u and u' are *region equivalent*, $u \cong u'$ iff the following holds:

1. For all $x \in C$, either $\lfloor u(x) \rfloor = \lfloor u'(x) \rfloor$ or $u(x), u'(x) > c_x$;
2. For all $x, y \in C$ with $u(x) \leq c_x$ and $u(y) \leq c_y$, $fr(u(x)) \leq fr(u(y))$ iff $fr(u'(x)) \leq fr(u'(y))$;
3. For all $x \in C$ with $u(x) \leq c_x$, $fr(u(x)) = 0$ iff $fr(u'(x)) = 0$.

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

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.

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 and Region 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 $d \in \mathbb{R}^{\geq 0}$

Symbolic States and Region 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

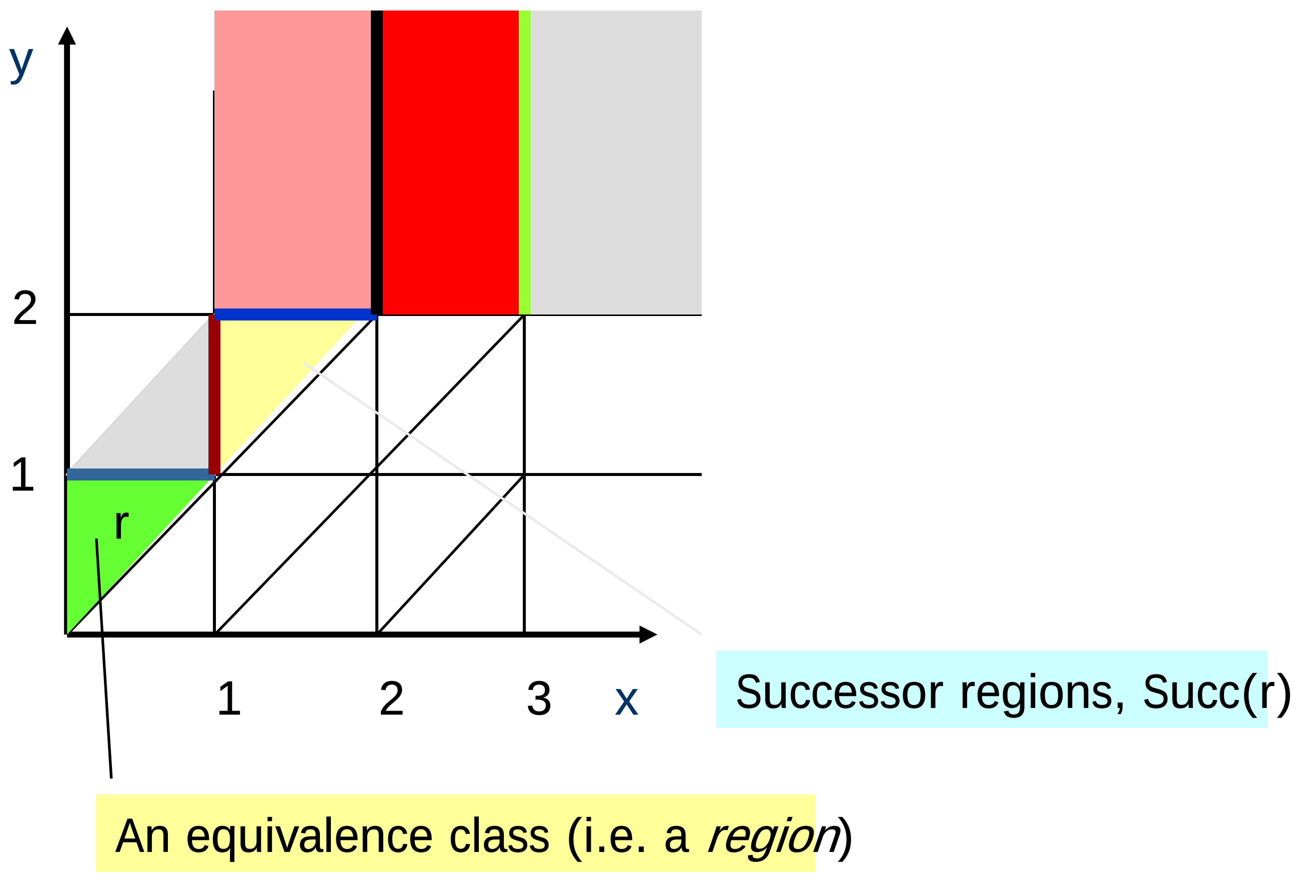
- states are symbolic states
- $\Longrightarrow$ between symbolic states is defined as follows:
  $(\ell, [v]) \Longrightarrow (\ell', [v'])$ iff $(\ell, v) \xrightarrow{a} (\ell', v')$ for some label a
  $(\ell, [v]) \Longrightarrow (\ell, [v'])$ iff $(\ell, v) \xrightarrow{d} (\ell, v')$ for some $d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

Fact

A region graph of any timed automaton is finite.

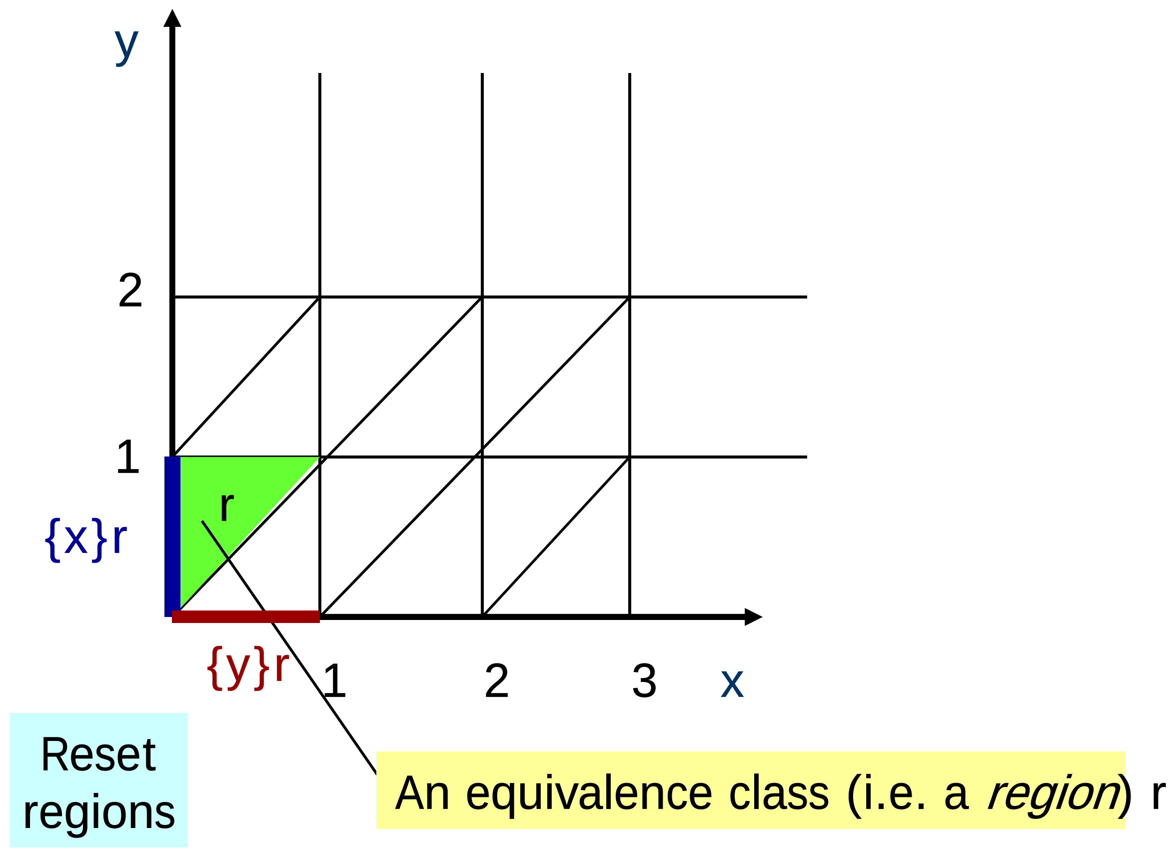
Regions

Successor Operation (wrt delay)

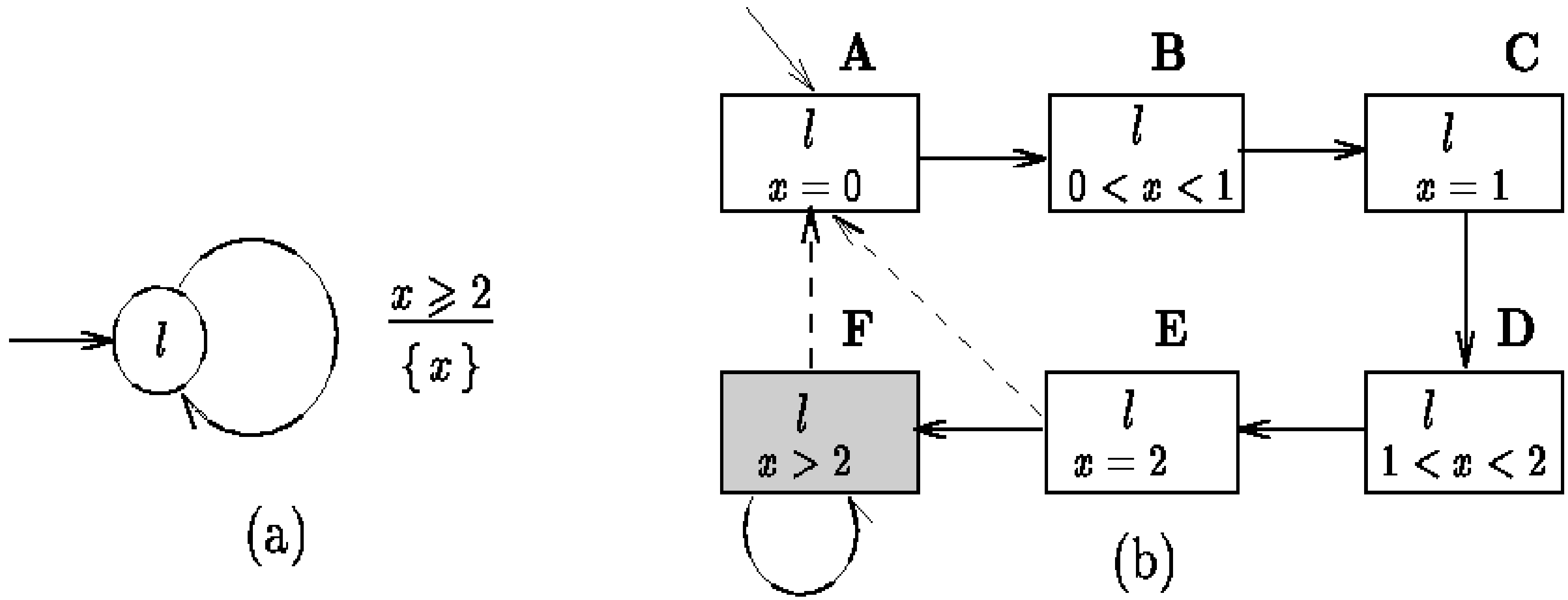


Regions

Reset Operation



An Example Region Graph



Application of Region Graphs to Reachability

We write $(\ell, v) \longrightarrow (\ell', v')$ whenever

- $(\ell, v) \xrightarrow{a} (\ell', v')$ for some label a, or
- $(\ell, v) \xrightarrow{d} (\ell, v')$ for some $d \in \mathbb{R}^{\geq 0}$.

Reachability Problem for Timed Automata

Instance (input): Automaton $A = (L, \ell_0, E, I)$ and a state (ℓ, v).

Question: Is it true that $(\ell_0, v_0) \longrightarrow^* (\ell, v)$?

(where $v_0(x) = 0$ for all $x \in C$)

Application of Region Graphs to Reachability

We write $(\ell, v) \longrightarrow (\ell', v')$ whenever

- $(\ell, v) \xrightarrow{a} (\ell', v')$ for some label a, or
- $(\ell, v) \xrightarrow{d} (\ell, v')$ for some $d \in \mathbb{R}^{\geq 0}$.

Reachability Problem for Timed Automata

Instance (input): Automaton $A = (L, \ell_0, E, I)$ and a state (ℓ, v).

Question: Is it true that $(\ell_0, v_0) \longrightarrow^* (\ell, v)$?

(where $v_0(x) = 0$ for all $x \in C$)

Reduction of Reachability from Timed Automata to Region Graphs

Reachability for timed automata is decidable because

$$(l_0, v_0) \longrightarrow^* (l, v) \text{ in the timed automaton if and only if}$$
$$(l_0, [v_0]) \Longrightarrow^* (l, [v]) \text{ in its (finite) region graph.}$$

Applicability of Region Graphs

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.

Applicability of Region Graphs

Proc

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

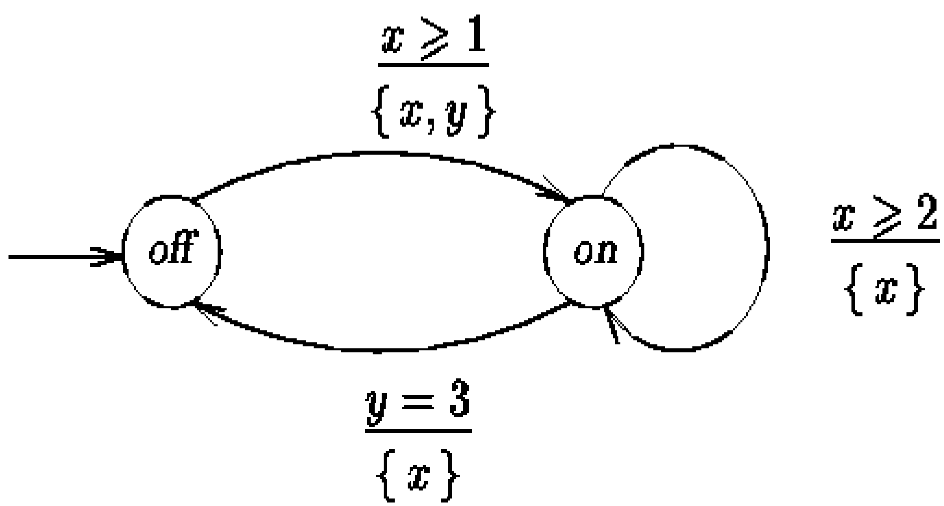
- reachability
- timed and untimed bisimilarity
- untimed language equivalence and language emptiness.

Cons

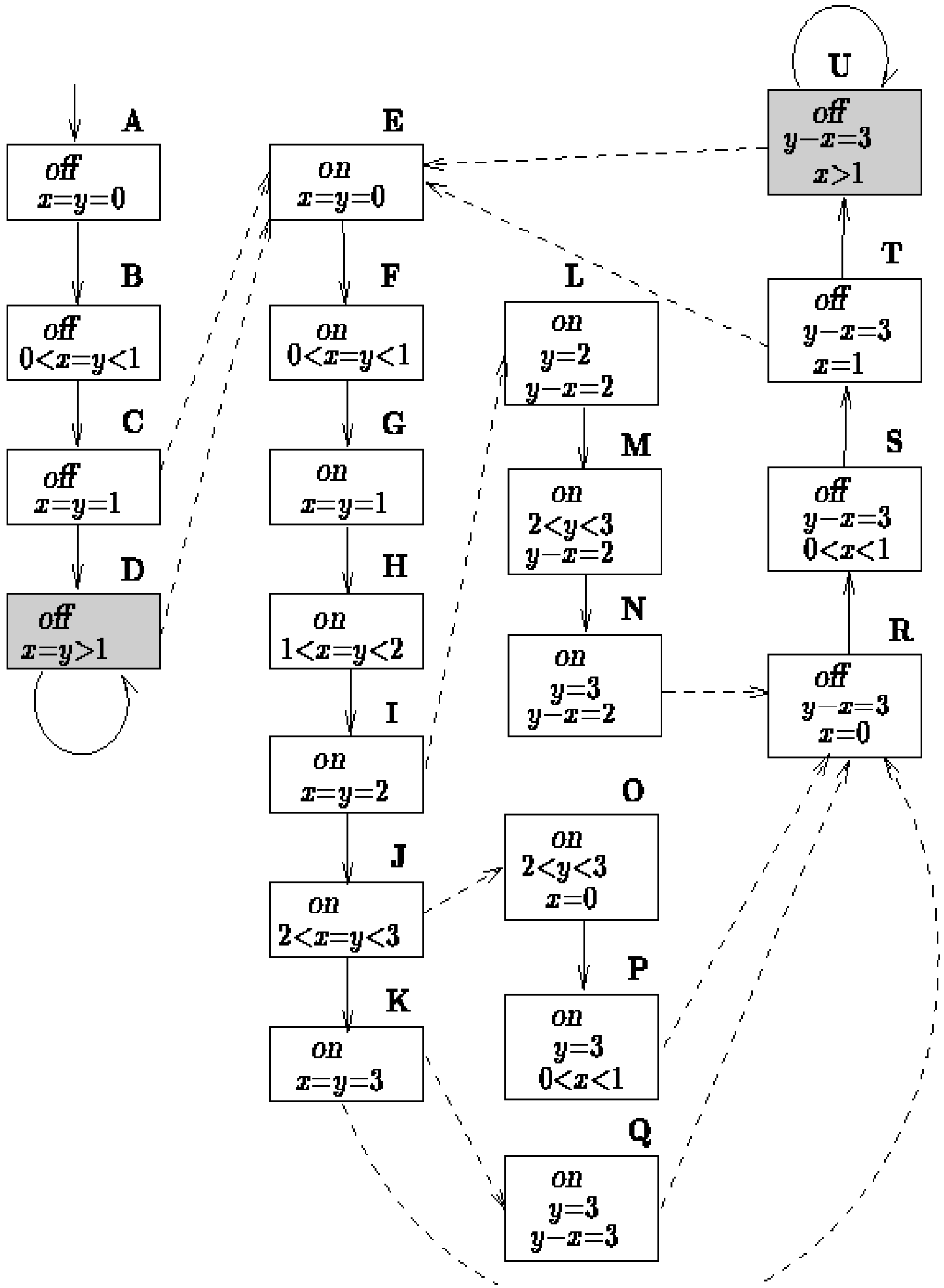
Region graphs have too large state spaces. State explosion is exponential in

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

Modified light switch



$inv(off) = \mathrm{true}$
$inv(on) = y \leqslant 3$



Reachable part of region graph

Properties

$\mathrm{AG}(x \leq y)$

$\mathrm{AG}(on \Rightarrow \mathrm{AF}off)$

$\mathrm{AG}(on \Rightarrow \mathrm{AF}_{\leq 9}off)$