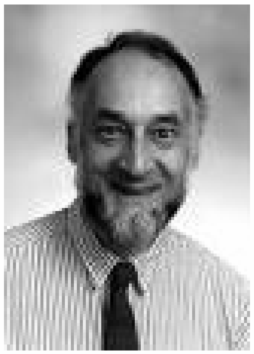
Semantics and Verification 2010

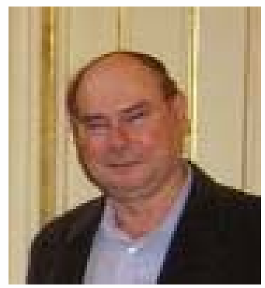
Lecture 9

- Labelled transition systems with time
- Timed CCS; syntax and semantics
- Timed Automata; syntax and semantics

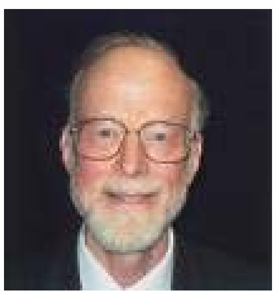
Turing Awards



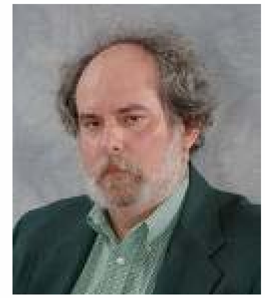
1991, CCS

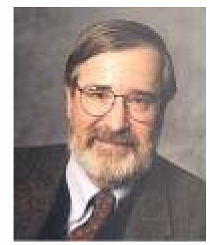


1996, Temporal Logic



1980, CSP





2007, Model Checking

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Need for Introducing Time Features

- Timeout in Alternating Bit protocol:
  - In CCS timeouts were modelled using nondeterminism.
  - Enough to prove that the protocol is safe.
  - Maybe too abstract for certain questions (What is the average time to deliver the messag e?).
- Many real-life systems depend on timing:
  - Real-time controllers (production lines, computers in cars, railway crossings).
  - Embedded systems (mobile phones, remote controllers, digital watch).
  - ...

Why CISS ?

- 80% of all software is embedded
- Demands for
  increased functionality
  with
  minimal resources
- Requires multitude of skills
  - Software construction
  - hardware platforms,
  - communication
  - testing & verification

- **Goal**:
  Give a qualitative lift to current industrial practice
  !!!!!

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

A Light Switch

- Informal Requirement
  - If the switch is off, and is pressed once, then the light will turn on.
  - If the switch is then pressed again soon after the light was turned on, then the light becomes brighter.
  - Otherwise, the light is turned off by the next button press.
  - The light is also turned off by a button press when it is bright.

Labelled Transition Systems with Time

Timed (labelled) transition system (TLTS)

TLTS is a triple $(Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ where

- $Proc$ is a set of states (or processes),
- $Act = N \cup \mathbb{R}^{\geq 0}$ is a set of actions (consisting of labels and time-elapsing steps), and
- for every $a \in Act$, $\xrightarrow{a} \subseteq Proc \times Proc$ is a binary relation on states called the transition relation.

We write

- $s \xrightarrow{a} s'$ if $a \in N$ and $(s, s') \in \xrightarrow{a}$, and
- $s \xrightarrow{d} s'$ if $d \in \mathbb{R}^{\geq 0}$ and $(s, s') \in \xrightarrow{d}$.

Requirements to TLTS

Sanity Requirements

Time additivity: If $s \xrightarrow{d} s'$ and $0 \leq d' \leq d$ then $s \xrightarrow{d'} s'' \xrightarrow{d-d'} s'$ for some state s'';

Zero delay: $s \xrightarrow{0} s$ for all states s;

Time determinism: If $s \xrightarrow{d} s'$ and $s \xrightarrow{d} s''$ then $s' = s''$.

How to Describe Timed Transition Systems?

Syntax		Semantics
unknown entity	$\longrightarrow$	known entity
CCS	$\longrightarrow$	Labelled Transition Systems
???	$\longrightarrow$	Timed Transition Systems

TCCS [Yi'90]:
CCS extended with delays.

Timed Automata [Alur, Dill'90]:
Finite-state automata equipped with clocks.

TCCS = CCS + Delay Prefix

Let $d \in \mathbb{R}^{\geq 0}$ and let P be a process then $\epsilon(d).P$ is the process which after a delay of d time units will behave like P.

Rules for Delay Prefix

We expect the following transitions

- $\epsilon(d).P \xrightarrow{d} P$
- $\epsilon(d).P \xrightarrow{d'} \epsilon(d - d').P$ for $d' \leq d$
- $\epsilon(d).P \xrightarrow{d+d'} P'$ if $P \xrightarrow{d'} P'$.

SOS Rules for TCCS

$$\frac{P \xrightarrow{d'} P'}{\epsilon(d).P \xrightarrow{d+d'} P'} \qquad \frac{}{\epsilon(d).P \xrightarrow{d'} \epsilon(d-d').P}\ d' \leq d$$

$$\frac{P \xrightarrow{d} P'}{K \xrightarrow{d} P'}\ K =^{def} P \qquad \frac{}{\alpha.P \xrightarrow{d} \alpha.P}\ \alpha \neq \tau \qquad \frac{P \xrightarrow{d} P' \quad Q \xrightarrow{d} Q'}{P + Q \xrightarrow{d} P' + Q'}$$

$$\frac{P \xrightarrow{d} P'}{P[f] \xrightarrow{d} P'[f]} \qquad \frac{P \xrightarrow{d} P'}{P \setminus L \xrightarrow{d} P' \setminus L}$$

Parallel Composition

Maximal Progress:
If a process can evolve on its own, then it will do so with out any further delay, i.e. if $P \xrightarrow{\tau}$ then $P \not\xrightarrow{d}$ for any $d > 0$.

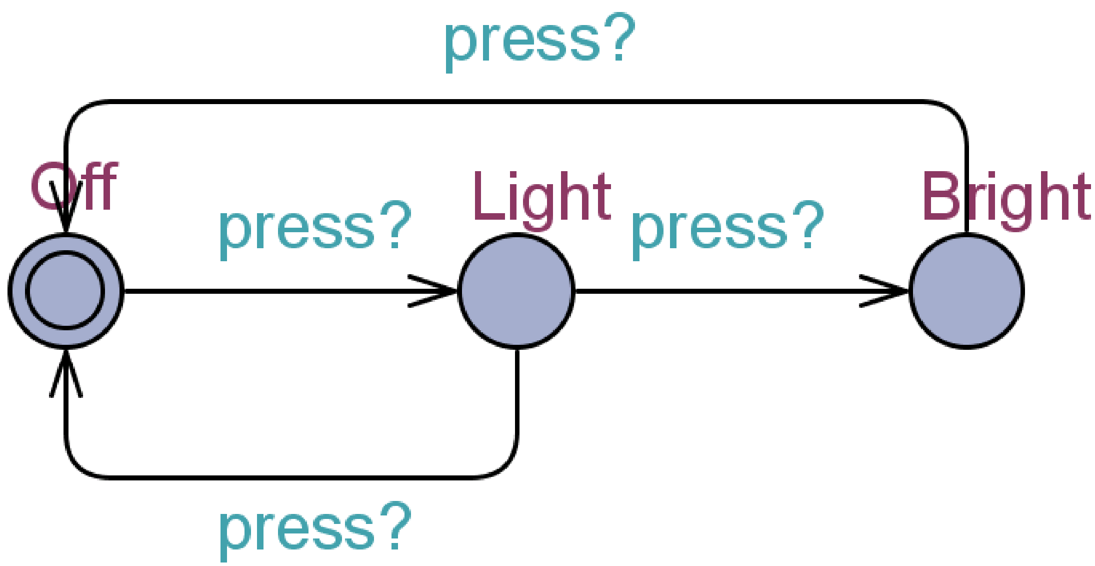
Delay for Parallel Composition

$$\frac{P \xrightarrow{d} P' \quad Q \xrightarrow{d} Q'}{P \mid Q \xrightarrow{d} P' \mid Q'} \quad \mathrm{NoSync}(P, Q, d)$$

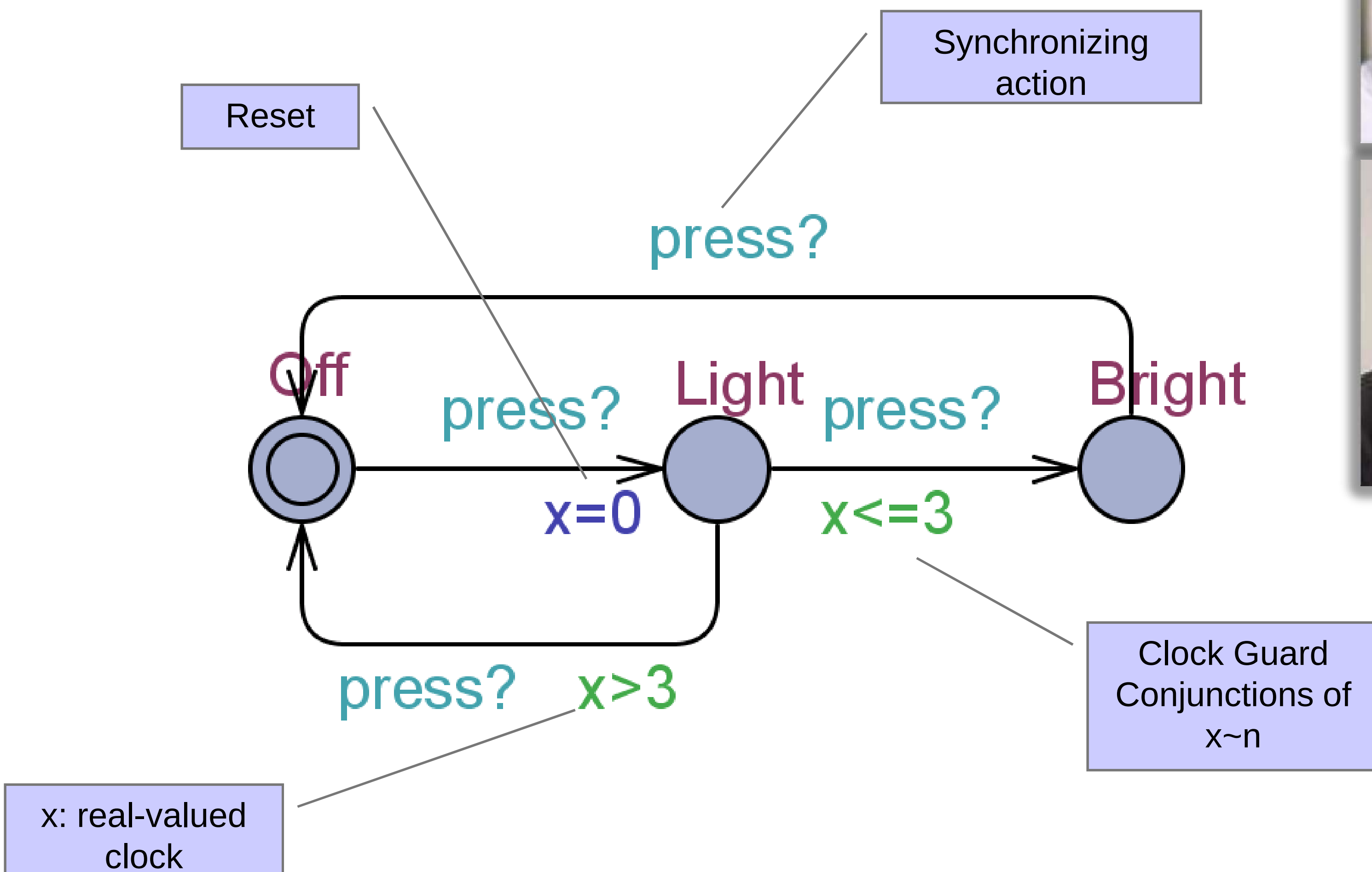
where $\mathrm{NoSync}(P, Q, d)$ holds if for any $d' < d$ whenever $P \xrightarrow{d'} P'$ and $\xrightarrow{d'} Q'$ then $P|Q \not\xrightarrow{\tau}$.

Timed Automata

A Dumb Light Controller

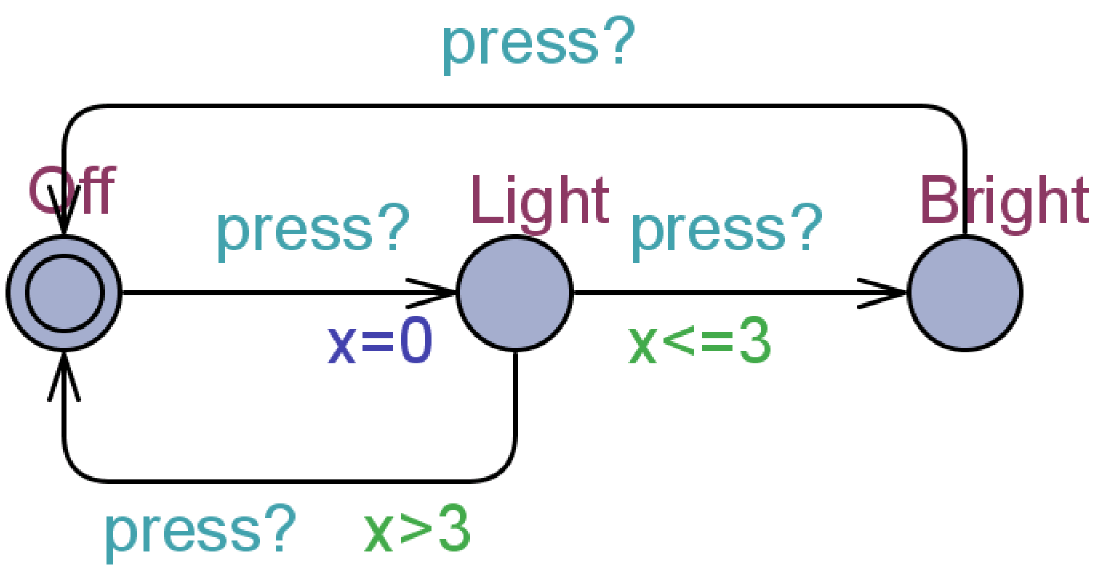


Timed Automata [Alur & Dill'89]



ADD a clock x

A Timed Automata (Semantics)

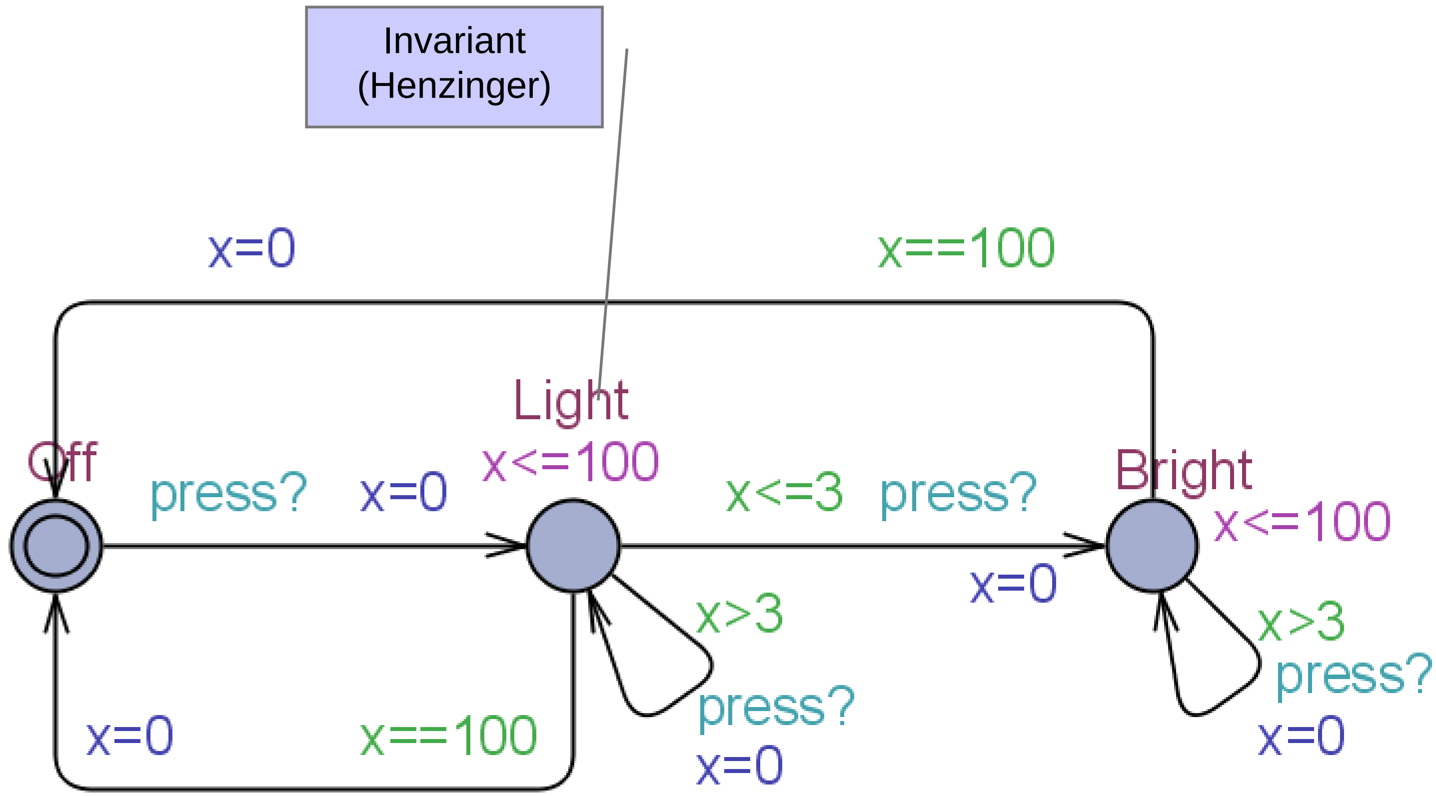


States:
(location , x=v) where v∈**R**

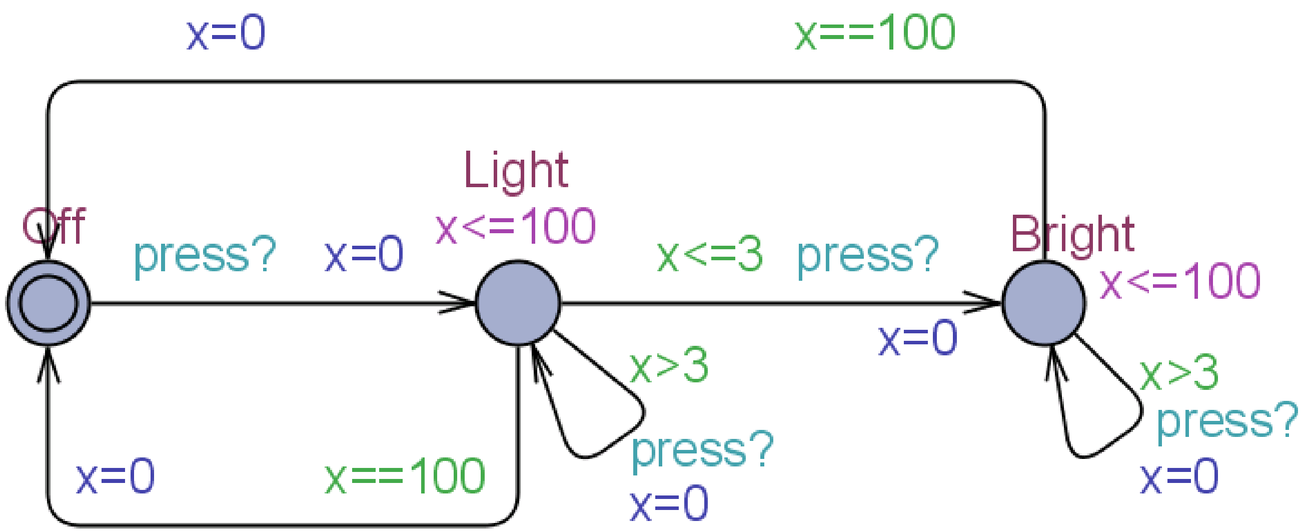
Transitions:

	(Off , x=0)
delay 4.32	→ (Off , x=4.32)
press?	→ (Light , x=0)
delay 2.51	→ (Light , x=2.51)
press?	→ (Bright , x=2.51)

Intelligent Light Controller



Intelligent Light Controller



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:

(Light , x=0) delay 103 → X

Invariants ensures progress

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 $v : C \to \mathbb{R}^{\geq 0}$.

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

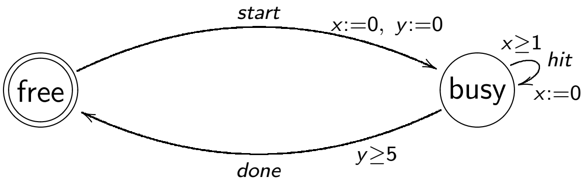
$$(L, \ell_0, E, I)$$

where

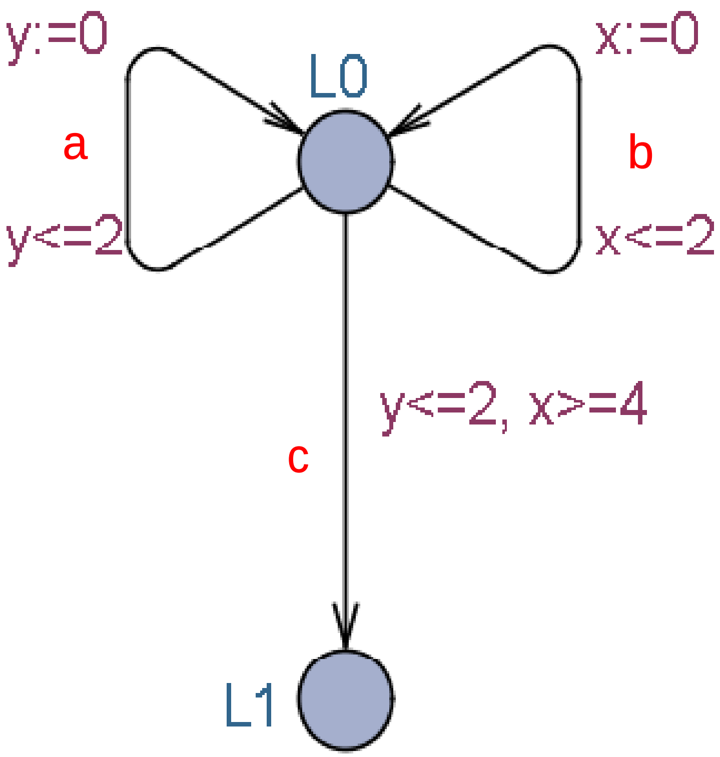
- L is a finite set of locations
- $\ell_0 \in L$ is the initial location
- $E \subseteq L \times \mathcal{B}(C) \times N \times 2^C \times L$ is the set of edges
- $I : L \to \mathcal{B}(C)$ assigns invariants to locations.

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

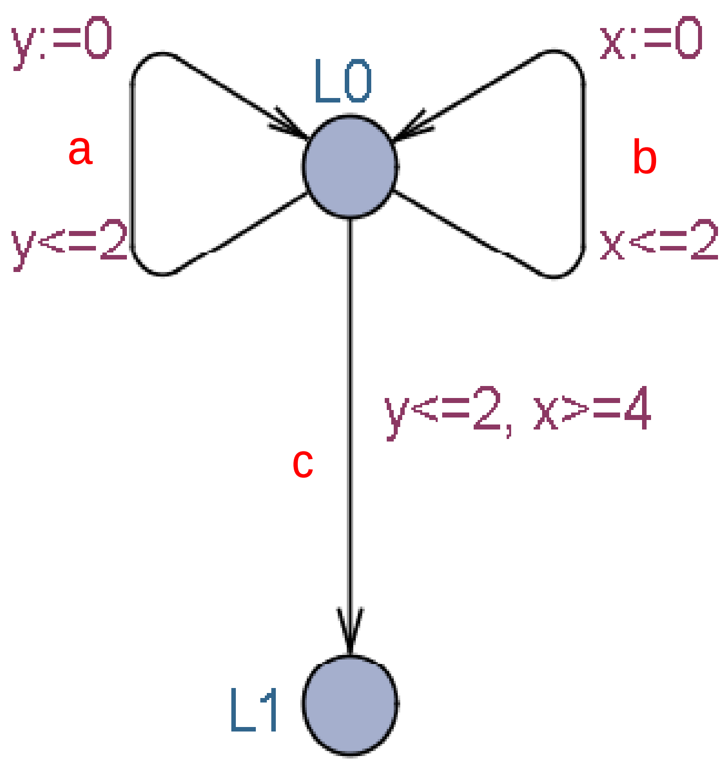
Example: Hammer

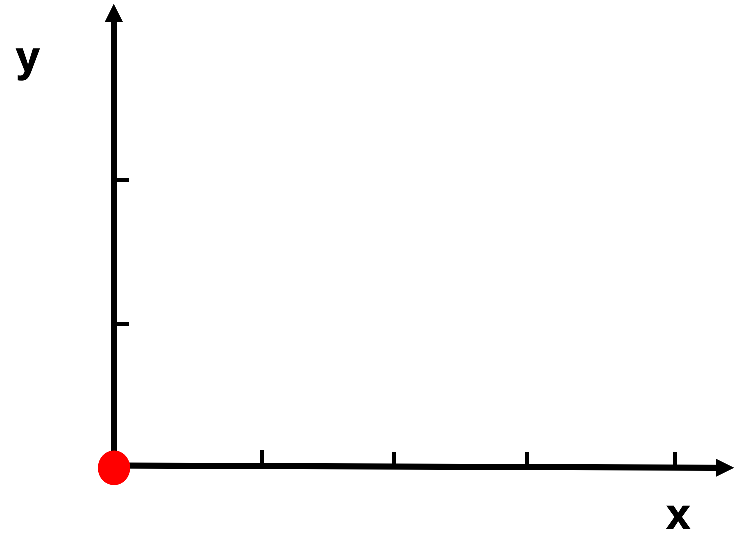


Example



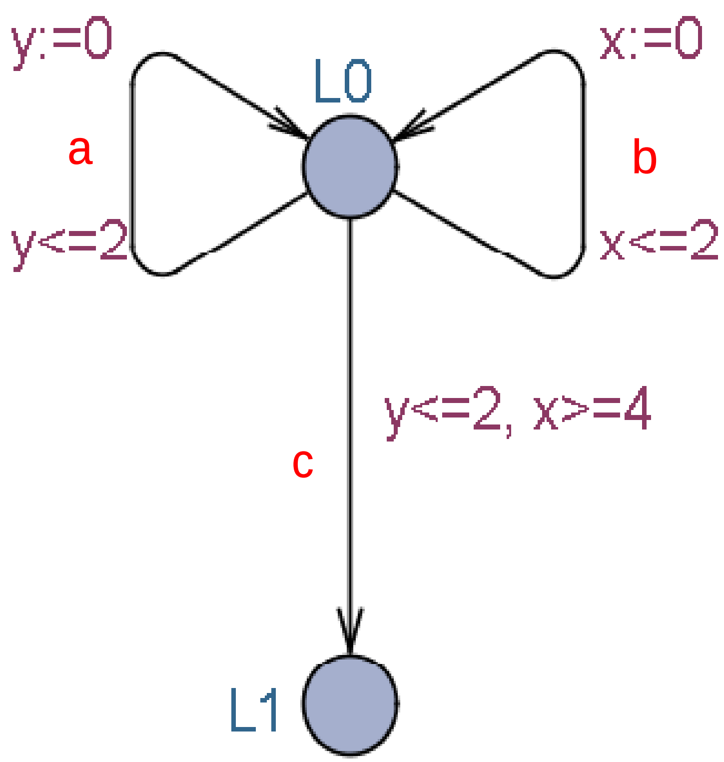
Example

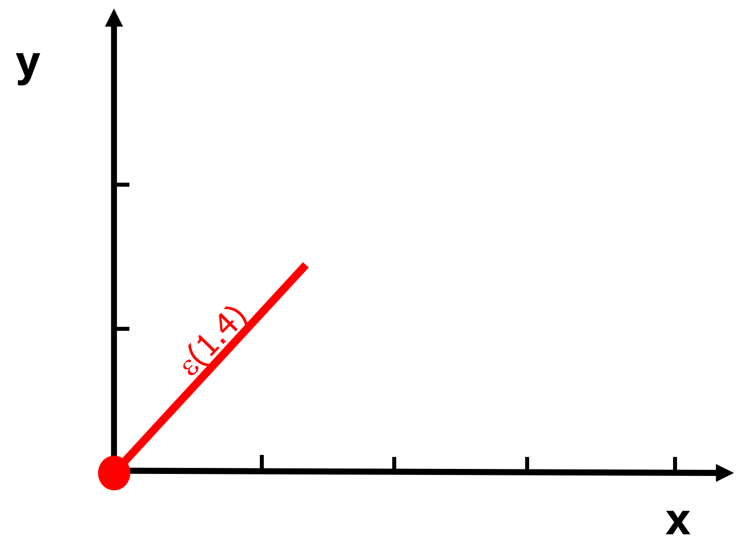




$$(\ell_0, x = 0, y = 0)$$

Example

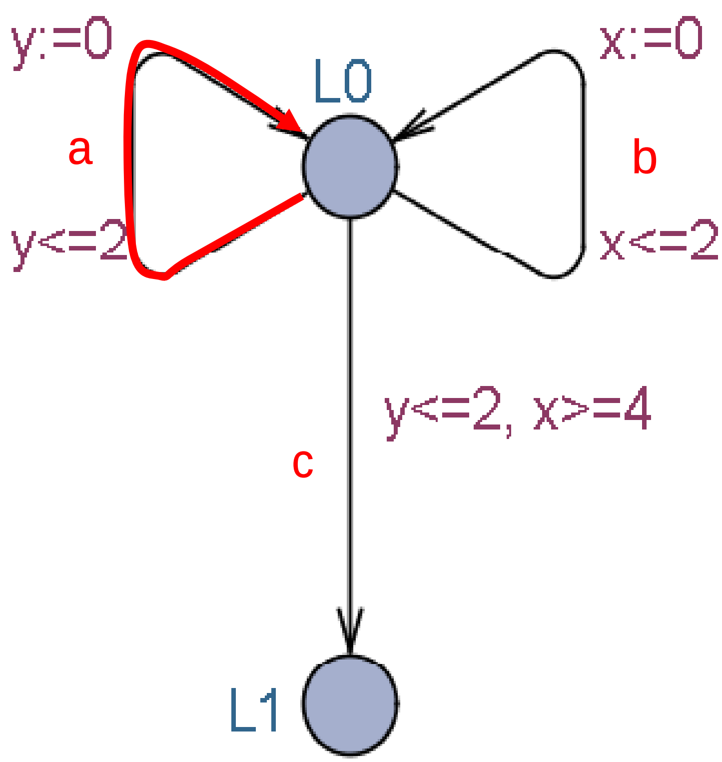


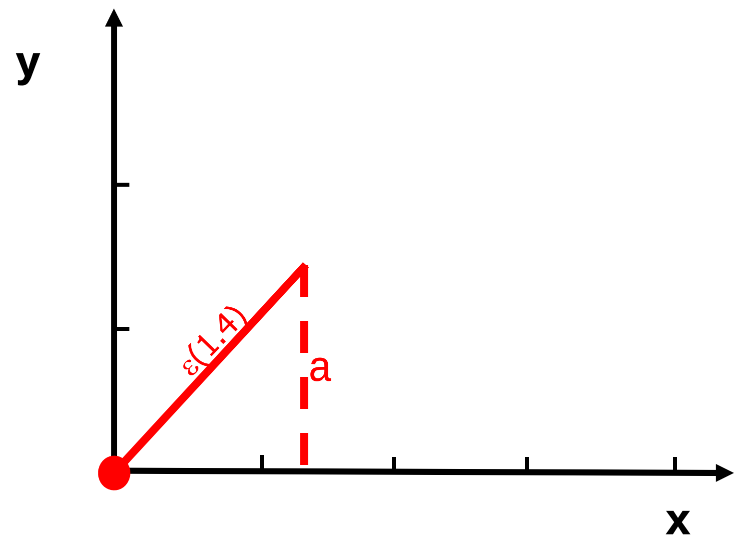


$$(\ell_0, x = 0, y = 0)$$

$$\xrightarrow{1.4} (\ell_0, x = 1.4, y = 1.4)$$

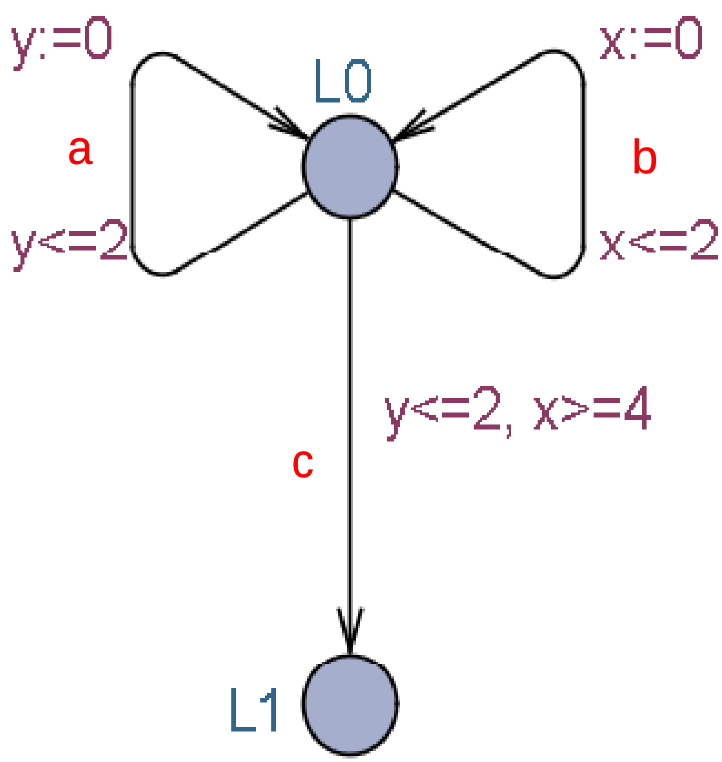
Example

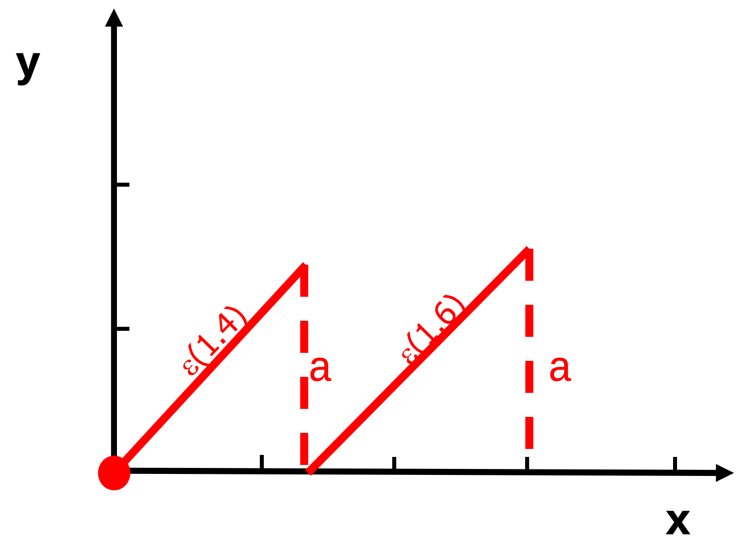




$$(\ell_0, x = 0, y = 0)$$
$$\xrightarrow{1.4} (\ell_0, x = 1.4, y = 1.4)$$
$$\xrightarrow{a} (\ell_0, x = 1.4, y = 0)$$

Example





$$(\ell_0, x = 0, y = 0)$$

$$\xrightarrow{1.4} (\ell_0, x = 1.4, y = 1.4)$$

$$\xrightarrow{a} (\ell_0, x = 1.4, y = 0)$$

$$\xrightarrow{1.6} (\ell_0, x = 3.0, y = 1.6)$$

$$\xrightarrow{a} (\ell_0, x = 3.0, y = 0)$$