

Real-Time Systems

Lecture 10: Timed Automata

2013-06-04

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

– 10 – 2013-06-04 – main –

Contents & Goals

Last Lecture:

- PLC, PLC automata

This Lecture:

- **Educational Objectives:** Capabilities for following tasks/questions.
 - what's notable about TA syntax? What's simple clock constraint?
 - what's a configuration of a TA? When are two in transition relation?
 - what's the difference between guard and invariant? Why have both?
 - what's a computation path? A run? Zeno behaviour?
- **Content:**
 - Timed automata syntax
 - TA operational semantics

– 10 – 2013-06-04 – Prelim –

Content

Introduction

- **First-order Logic**
- **Duration Calculus (DC)**
- Semantical Correctness Proofs with DC
- DC Decidability
- DC Implementables
- **PLC-Automata**
- **Timed Automata (TA)**, Uppaal
- Networks of Timed Automata
- Region/Zone-Abstraction
- Extended Timed Automata
- Undecidability Results

$obs : \text{Time} \rightarrow \mathcal{D}(obs)$

$\langle obs_0, \nu_0 \rangle, t_0 \xrightarrow{\lambda_0} \langle obs_1, \nu_1 \rangle, t_0 \xrightarrow{\lambda_1} \langle obs_{1/2}, t_{1/2} \rangle$

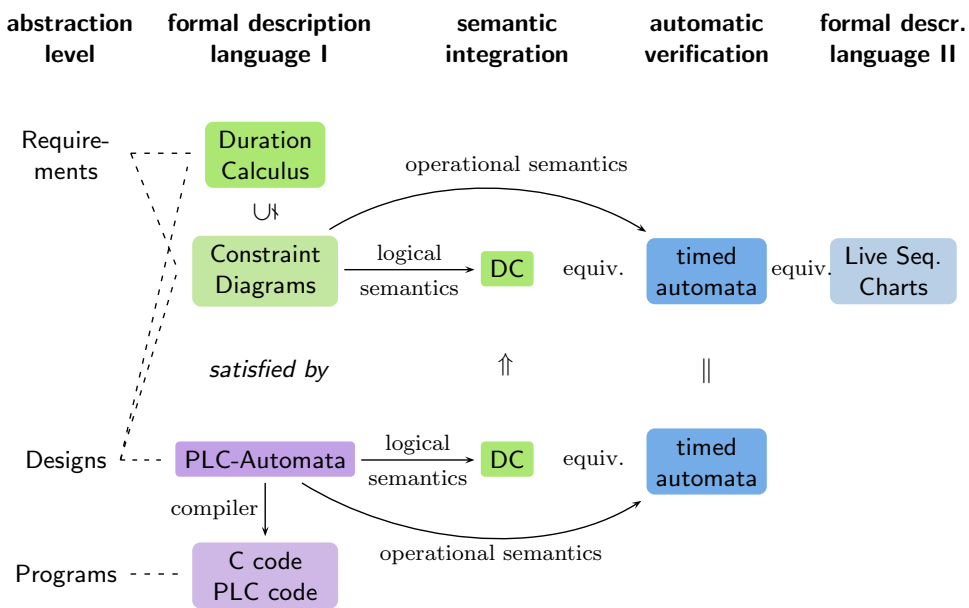
- **Automatic Verification...**
- ...whether TA satisfies DC formula, observer-based

Recap

- 01 - 2013-04-16 - Scontent -

17/37

Recall: Tying It All Together

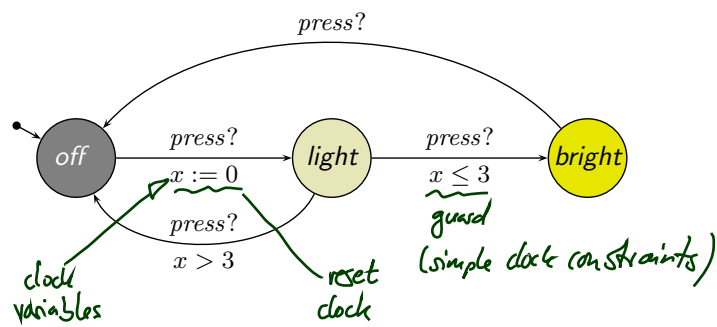


- 10 - 2013-06-04 - Prelim -

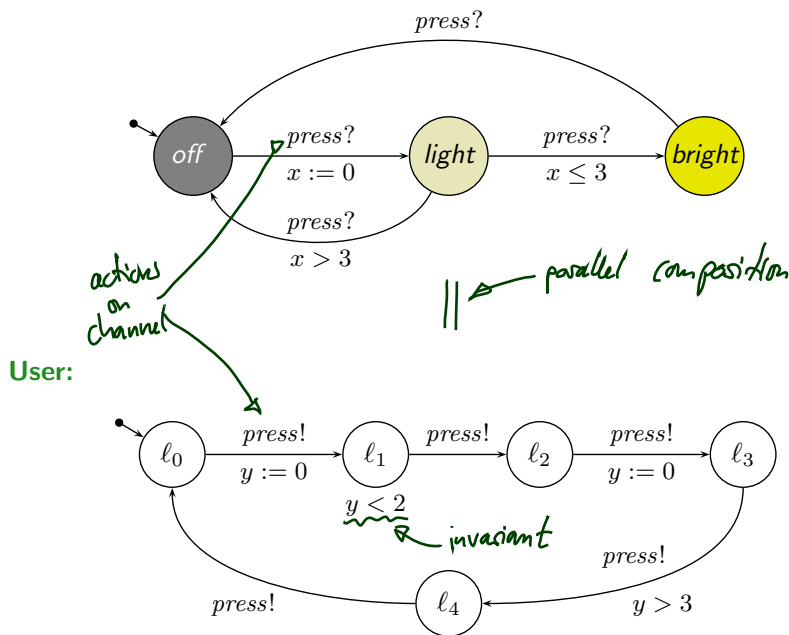
3/31

Example: Off/Light/Bright

Example



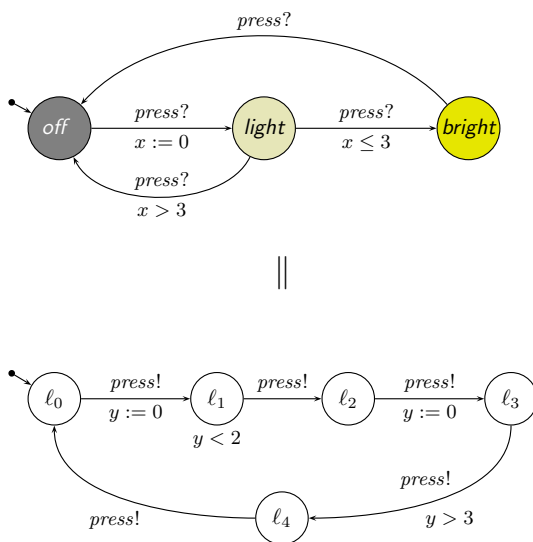
Example



- 10 - 2013-06-04 - Seta -

5/31

Example Cont'd



Problems:

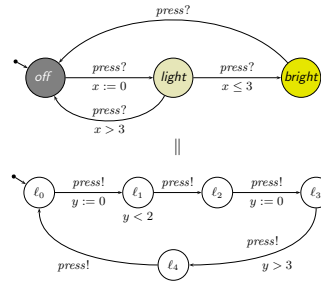
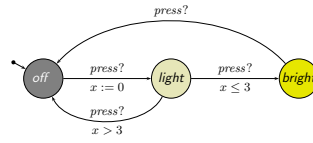
- Deadlock freedom [Behrmann et al., 2004]
- Location Reachability ("Is this user able to reach 'bright'?")
- Constraint Reachability ("Can the controller's clock go past 5?")

- 10 - 2013-06-04 - Seta -

6/31

Plan

- **Pure TA** syntax
 - channels, actions
 - (simple) clock constraints
 - Def. TA
- **Pure TA** operational semantics
 - clock valuation, time shift, modification
 - operational semantics
 - discussion
- Transition sequence, computation path, run
- **Network of TA**
 - parallel composition (syntactical)
 - restriction
 - network of TA semantics
- **Uppaal Demo**
- Region abstraction; zones
- **Extended TA**; Logic of Uppaal



7/31

- 10 - 2013-06-04 - Seta -

Pure TA Syntax

- 10 - 2013-06-04 - main -

8/31

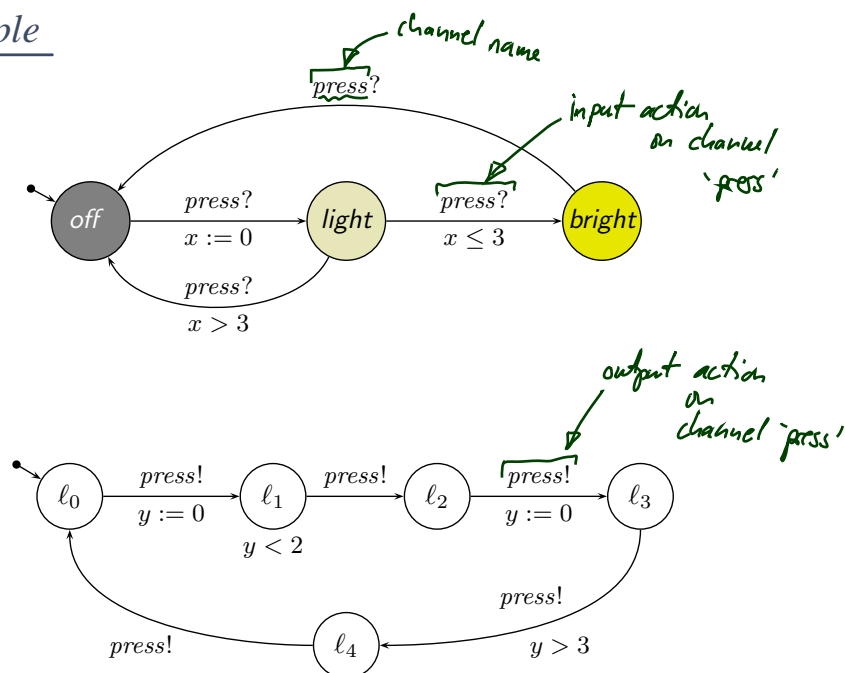
Channel Names and Actions

To define timed automata formally, we need the following sets of symbols:

- A set $(a, b \in) \text{Chan}$ of **channel names** or **channels**.
- For each channel $a \in \text{Chan}$, two **visible actions**:
 $a?$ and $a!$ denote **input** and **output** on the **channel** ($a?, a! \notin \text{Chan}$).
- $\tau \notin \text{Chan}$ represents an **internal action**, not visible from outside.
- $(\alpha, \beta \in) \text{Act} := \{a? \mid a \in \text{Chan}\} \cup \{a! \mid a \in \text{Chan}\} \cup \{\tau\}$
is the set of **actions**.
- An **alphabet** B is a set of **channels**, i.e. $B \subseteq \text{Chan}$.
- For each alphabet B , we define the corresponding **action set**

$$B_{?!} := \{a? \mid a \in B\} \cup \{a! \mid a \in B\} \cup \{\tau\}.$$
- Note: $\text{Chan}_{?!} = \text{Act}$.

Example



Simple Clock Constraints

- Let $(x, y \in X)$ be a set of **clock variables** (or **clocks**).
- The set $(\varphi \in \Phi(X))$ of **(simple) clock constraints** (over X) is defined by the following grammar:

$$\varphi ::= x \sim c \mid x - y \sim c \mid \varphi_1 \wedge \varphi_2 \mid \text{true}$$

where

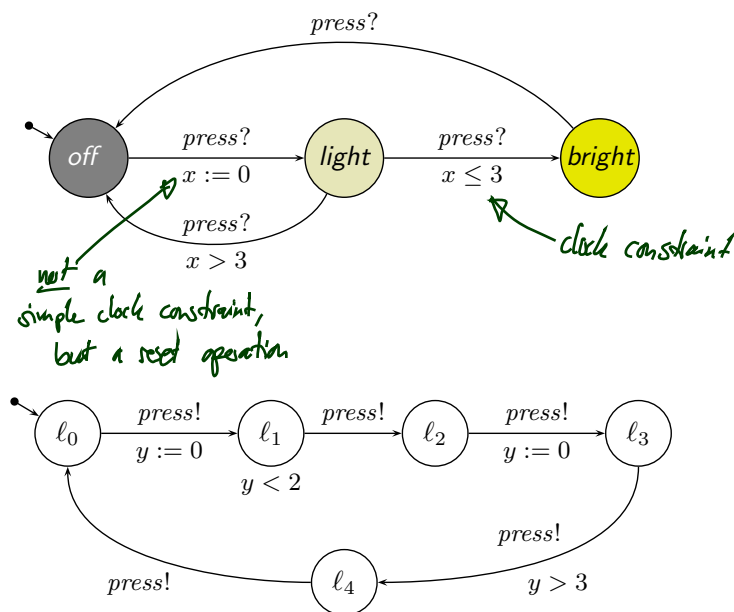
- $x, y \in X$,
- $c \in \mathbb{Q}_0^+$, and
- $\sim \in \{<, >, \leq, \geq\}$.

*we may use $x=c$ ($x=y$) as an abbreviation
 $x \in \mathbb{C} \wedge x > c$
 $(x-y \geq 0 \wedge x-y \leq 0)$*

if $X \neq \emptyset$, this can be an abbreviation for $x \geq 0, x \in X$.

- Clock constraints of the form $x - y \sim c$ are called **difference constraints**.

Example



Timed Automaton

Definition 4.3. [Timed automaton]

A (pure) **timed automaton** \mathcal{A} is a structure

$$\mathcal{A} = (L, B, X, I, E, \ell_{ini})$$

where

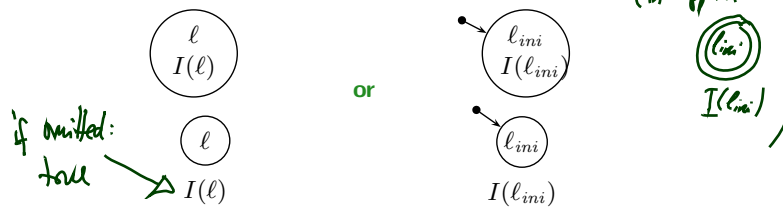
- $(\ell \in) L$ is a finite set of **locations** (or **control states**),
- $B \subseteq \text{Chan}$,
- X is a finite set of clocks,
- $I : L \rightarrow \Phi(X)$ assigns to each location a clock constraint, its **invariant**,
- $E \subseteq L \times B \times \Phi(X) \times 2^X \times L$ a finite set of **directed edges**.
Edges $(\ell, \alpha, \varphi, Y, \ell')$ from location ℓ to ℓ' are labelled with an **action** α , a **guard** φ , and a set Y of clocks that will be **reset**.
- ℓ_{ini} is the **initial location**.

- 10 - 2013-06-04 - Stasyn -

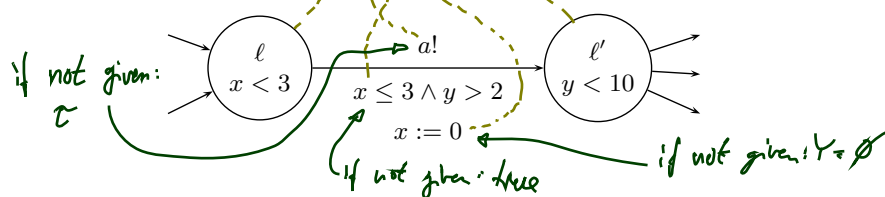
Graphical Representation of Timed Automata

$$\mathcal{A} = (L, B, X, I, E, \ell_{ini})$$

- **Locations (control states)** and their invariants:



- **Edge (control states):** $(\ell, \alpha, \varphi, Y, \ell') \in L \times B \times \Phi(X) \times 2^X \times L$



- 10 - 2013-06-04 - Stasyn -

Pure TA Operational Semantics

Clock Valuations

- Let X be a set of clocks. A **valuation ν of clocks** in X is a mapping

$$\nu : X \rightarrow \text{Time}$$

assigning each clock $x \in X$ the **current time** $\nu(x)$.

- Let φ be a clock constraint.
The **satisfaction** relation between clock valuations ν and clock constraints φ , denoted by $\nu \models \varphi$, is defined inductively:

- $\nu \models x \sim c$ iff $\nu(x) \sim c$ $\nu(x) \hat{=} \hat{c}$
- $\nu \models x - y \sim c$ iff $\nu(x) - \nu(y) \sim c$ $\nu(x) \hat{=} \nu(y) \hat{=} \hat{c}$
- $\nu \models \varphi_1 \wedge \varphi_2$ iff $\nu \models \varphi_1$ and $\nu \models \varphi_2$ $\nu \models \varphi_1$ and $\nu \models \varphi_2$

- Two clock constraints φ_1 and φ_2 are called (**logically**) **equivalent** if and only if for all clock valuations ν , we have

$$\nu \models \varphi_1 \text{ if and only if } \nu \models \varphi_2.$$

In that case we write $\varphi_1 \iff \varphi_2$.

Operations on Clock Valuations

Let ν be a valuation of clocks in X and $t \in \text{Time}$.

- **Time Shift**

We write $\nu + t$ to denote the clock valuation (for X) with

function name

$$(\nu + t)(x) = \nu(x) + t.$$

for all $x \in X$,

- **Modification**

Let $Y \subseteq X$ be a set of clocks.

We write $\nu[Y := t]$ to denote the clock valuation with

function name

$$(\nu[Y := t])(x) = \begin{cases} t & , \text{ if } x \in Y \\ \nu(x) & , \text{ otherwise} \end{cases}$$

Special case **reset**: $t = 0$.

Operational Semantics of TA

*c.f. (A.)
x > 0*

Definition 4.4. The **operational semantics** of a timed automaton

$$\mathcal{A} = (L, B, X, I, E, \ell_{ini})$$

is defined by the (labelled) transition system

$$\mathcal{T}(\mathcal{A}) = (\text{Conf}(\mathcal{A}), \text{Time} \cup B_{?!}, \{\overset{\lambda}{\rightarrow} \mid \lambda \in \text{Time} \cup B_{?!}\}, C_{ini})$$

where

- $\text{Conf}(\mathcal{A}) = \{\langle \ell, \nu \rangle \mid \ell \in L, \nu : X \rightarrow \text{Time}, \nu \models I(\ell)\}$
- $\text{Time} \cup B_{?!}$ are the transition labels,
- there are **delay transition relations**

$$\langle \ell, \nu \rangle \xrightarrow{\lambda} \langle \ell', \nu' \rangle, \lambda \in \text{Time}$$

and **action transition relations**

$$\langle \ell, \nu \rangle \xrightarrow{\lambda} \langle \ell', \nu' \rangle, \lambda \in B_{?!}$$

(\rightarrow later slides)

- $C_{ini} = \{\langle \ell_{ini}, \nu_0 \rangle\} \cap \text{Conf}(\mathcal{A})$ with $\nu_0(x) = 0$ for all $x \in X$ is the set of **initial configurations**.

*the set of labels
a set (!) of transition relations*

*• can be larger than 1? NO
• always size 1? IN GENERAL NO
• can this be empty? YES, if $\nu_0 \not\models I(\ell_{ini})$*

Operational Semantics of TA Cont'd

$$\mathcal{A} = (L, B, X, I, E, \ell_{ini})$$

$$\mathcal{T}(\mathcal{A}) = (Conf(\mathcal{A}), \text{Time} \cup B?, \{\xrightarrow{\lambda} \mid \lambda \in \text{Time} \cup B?\}, C_{ini})$$

$\subseteq Conf(\mathcal{A}) \times Conf(\mathcal{A})$

- **Time** or **delay transition**:

$$\langle \ell, \nu \rangle \xrightarrow{t} \langle \ell, \nu + t \rangle$$

if and only if $\forall t' \in [0, t] : \nu + t' \models I(\ell)$.

“Some **time** $t \in \text{Time}$ **elapses** respecting invariants, location unchanged.”

- **Action** or **discrete transition**:

$$\langle \ell, \nu \rangle \xrightarrow{\alpha} \langle \ell', \nu' \rangle$$

if and only if there is $(\ell, \alpha, \varphi, Y, \ell') \in E$ such that

$$\nu \models \varphi, \quad \nu' = \nu[Y := 0], \quad \text{and } \nu' \models I(\ell').$$

“An action occurs, location may change, some clocks may be reset, **time does not advance**.”

19/31

- 10 - 2013-06-04 - Statesem -

Transition Sequences, Reachability

- A **transition sequence** of \mathcal{A} is any finite or infinite sequence of the form

$$\langle \ell_0, \nu_0 \rangle \xrightarrow{\lambda_1} \langle \ell_1, \nu_1 \rangle \xrightarrow{\lambda_2} \langle \ell_2, \nu_2 \rangle \xrightarrow{\lambda_3} \dots$$

with

- $\langle \ell_0, \nu_0 \rangle \in C_{ini}$,
- for all $i \in \mathbb{N}$, there is $\xrightarrow{\lambda_{i+1}}$ in $\mathcal{T}(\mathcal{A})$ with $\langle \ell_i, \nu_i \rangle \xrightarrow{\lambda_{i+1}} \langle \ell_{i+1}, \nu_{i+1} \rangle$

- 10 - 2013-06-04 - Statesem -

20/31

Transition Sequences, Reachability

- A **transition sequence** of \mathcal{A} is any finite or infinite sequence of the form

$$\langle l_0, \nu_0 \rangle \xrightarrow{\lambda_1} \langle l_1, \nu_1 \rangle \xrightarrow{\lambda_2} \langle l_2, \nu_2 \rangle \xrightarrow{\lambda_3} \dots$$

with

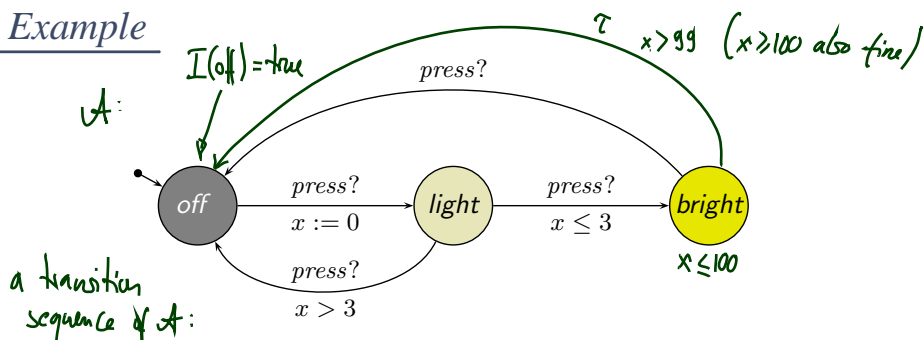
- $\langle l_0, \nu_0 \rangle \in C_{ini}$,
- for all $i \in \mathbb{N}$, there is $\xrightarrow{\lambda_{i+1}}$ in $\mathcal{T}(\mathcal{A})$ with $\langle l_i, \nu_i \rangle \xrightarrow{\lambda_{i+1}} \langle l_{i+1}, \nu_{i+1} \rangle$

- A **configuration** $\langle l, \nu \rangle$ is called **reachable** (in \mathcal{A}) if and only if there is a transition sequence of the form

$$\langle l_0, \nu_0 \rangle \xrightarrow{\lambda_1} \langle l_1, \nu_1 \rangle \xrightarrow{\lambda_2} \langle l_2, \nu_2 \rangle \xrightarrow{\lambda_3} \dots \xrightarrow{\lambda_n} \langle l_n, \nu_n \rangle = \langle l, \nu \rangle$$

- A **location** l is called **reachable** if and only if **any** configuration $\langle l, \nu \rangle$ is reachable, i.e. there exists a valuation ν such that $\langle l, \nu \rangle$ is reachable.

Example

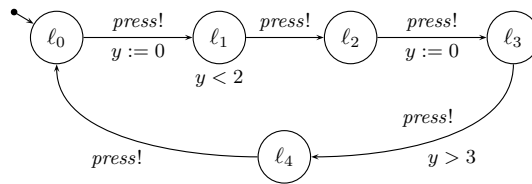


$$\begin{aligned} &\langle \text{off}, x = 0 \rangle \xrightarrow{2.5} \langle \text{off}, x = 2.5 \rangle \xrightarrow{1.7} \langle \text{off}, x = 4.2 \rangle \xrightarrow{10} \langle \text{off}, x = 14.2 \rangle \xrightarrow{0} \langle \text{off}, x = 14.2 \rangle \\ &\quad \underbrace{\text{press?}}_{\text{line } \nu_0} \xrightarrow{\text{press?}} \langle \text{light}, x = 0 \rangle \xrightarrow{2.1} \langle \text{light}, x = 2.1 \rangle \\ &\quad \xrightarrow{\text{press?}} \langle \text{bright}, x = 2.1 \rangle \xrightarrow{10} \langle \text{bright}, x = 12.1 \rangle \\ &\quad \xrightarrow{\text{press?}} \langle \text{off}, x = 12.1 \rangle \\ &\quad \xrightarrow{\text{press?}} \langle \text{light}, x = 0 \rangle \xrightarrow{0} \langle \text{light}, x = 0 \rangle \\ &\quad \xrightarrow{\text{press?}} \langle \text{bright}, x = 0 \rangle \xrightarrow{12.1} \langle \text{bright}, x = 12.1 \rangle \\ &\quad \xrightarrow{10} \langle \text{b.}, x = 100 \rangle \xrightarrow{\tau} \langle \text{off}, x = 100 \rangle \end{aligned}$$

NOT A CONFIGURATION

Discussion: Set of Configurations

Recall the user model for our light controller:



- (“Good”) configurations:

$\langle l_1, y = 0 \rangle, \langle l_1, y = 1.9 \rangle, \langle l_2, y = 1000 \rangle,$

$\langle l_2, y = 0.5 \rangle, \langle l_3, y = 27 \rangle$

- “Bad” configurations: (actually not configs.)

$\langle l_1, y = 2.0 \rangle, \langle l_1, y = 2.5 \rangle$

Two Approaches to Exclude “Bad” Configurations

- The approach taken for TA:

- Rule out **bad** configurations in the step from \mathcal{A} to $\mathcal{T}(\mathcal{A})$.

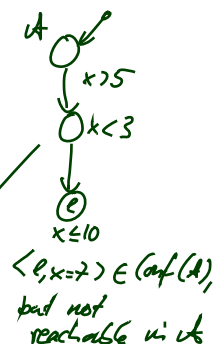
“Bad” configurations are not even configurations!

- Recall Definition 4.4:

- $Conf(\mathcal{A}) = \{ \langle \ell, \nu \rangle \mid \ell \in L, \nu : X \rightarrow \text{Time}, \nu \models I(\ell) \}$

- $C_{ini} = \{ \langle \ell_{ini}, \nu_0 \rangle \} \cap Conf(\mathcal{A})$

- **Note:** Being in $Conf(\mathcal{A})$ doesn't mean to be **reachable**.



- The approach not taken for TA:

- consider every $\langle \ell, \nu \rangle$ to be a configuration, i.e. have

$$Conf(\mathcal{A}) = \{ \langle \ell, \nu \rangle \mid \ell \in L, \nu : X \rightarrow \text{Time} \}$$

- “bad” configurations not in transition relation with others, i.e. have, e.g.,

$$\langle \ell, \nu \rangle \xrightarrow{t} \langle \ell, \nu + t \rangle$$

if and only if $\forall t' \in [0, t] : \nu + t' \models I(\ell)$ **and** $\nu + t' \models I(\ell')$.

Computation Path, Run

Computation Paths

- $\langle \ell, \nu \rangle, t$ is called **time-stamped configuration**, $t \in \text{Time}$
- **time-stamped delay transition**: $\langle \ell, \nu \rangle, t \xrightarrow{t'} \langle \ell, \nu + t' \rangle, t + t'$
iff $t' \in \text{Time}$ and $\langle \ell, \nu \rangle \xrightarrow{t'} \langle \ell, \nu + t' \rangle$.
- **time-stamped action transition**: $\langle \ell, \nu \rangle, t \xrightarrow{\alpha} \langle \ell', \nu' \rangle, t$
iff $\alpha \in B_{?!}$ and $\langle \ell, \nu \rangle \xrightarrow{\alpha} \langle \ell', \nu' \rangle$.

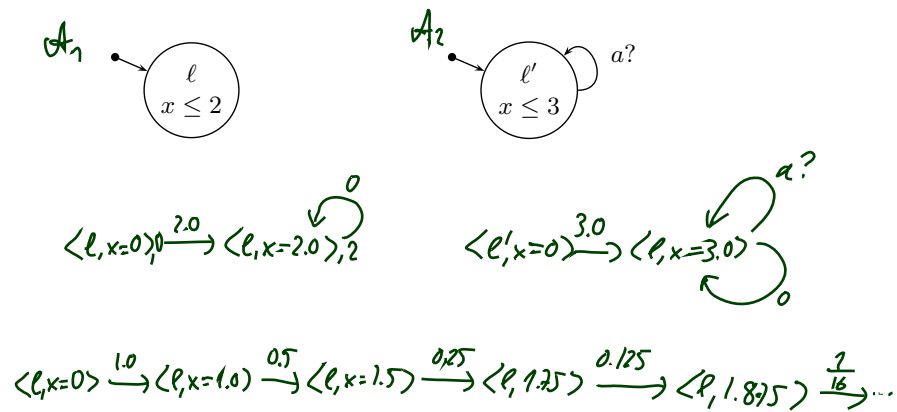
- A sequence of time-stamped configurations

$$\xi = \langle \ell_0, \nu_0 \rangle, t_0 \xrightarrow{\lambda_1} \langle \ell_1, \nu_1 \rangle, t_1 \xrightarrow{\lambda_2} \langle \ell_2, \nu_2 \rangle, t_2 \xrightarrow{\lambda_3} \dots$$

is called **computation path** (or path) of \mathcal{A} starting in $\langle \ell_0, \nu_0 \rangle, t_0$
if and only if it is either infinite or maximally finite.

- A **computation path** (or path) is a computation path starting at $\langle \ell_0, \nu_0 \rangle, 0$
where $\langle \ell_0, \nu_0 \rangle \in C_{ini}$.

Timelocks and Zeno Behaviour



Timelocks and Zeno Behaviour



- **Timelock:**

$$\langle l, x = 0 \rangle, 0 \xrightarrow{2} \langle l, x = 2 \rangle, 2$$

$$\langle l', x = 0 \rangle, 0 \xrightarrow{3} \langle l', x = 3 \rangle, 3 \xrightarrow{a?} \langle l', x = 3 \rangle, 3 \xrightarrow{a?} \dots$$

- **Zeno** behaviour:

$$\langle l, x = 0 \rangle, 0 \xrightarrow{1/2} \langle l, x = 1/2 \rangle, \frac{1}{2} \xrightarrow{1/4} \langle l, x = 3/4 \rangle, \frac{3}{4} \dots$$

$$\xrightarrow{1/2^n} \langle l, x = (2^n - 1)/2^n \rangle, \frac{2^n - 1}{2^n} \dots$$

Real-Time Sequence

Definition 4.9. An infinite sequence

$$t_0, t_1, t_2, \dots$$

of values $t_i \in \text{Time}$ for $i \in \mathbb{N}_0$ is called **real-time sequence** if and only if it has the following properties:

- **Monotonicity:**

$$\forall i \in \mathbb{N}_0 : t_i \leq t_{i+1}$$

- **Non-Zeno behaviour (or unboundedness or progress):**

$$\forall t \in \text{Time} \exists i \in \mathbb{N}_0 : t < t_i$$

Run

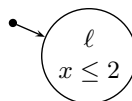
Definition 4.10. A **run** of \mathcal{A} **starting** in the time-stamped configuration $\langle \ell_0, \nu_0 \rangle, t_0$ is an infinite computation path of \mathcal{A}

$$\xi = \langle \ell_0, \nu_0 \rangle, t_0 \xrightarrow{\lambda_1} \langle \ell_1, \nu_1 \rangle, t_1 \xrightarrow{\lambda_2} \langle \ell_2, \nu_2 \rangle, t_2 \xrightarrow{\lambda_3} \dots$$

where $(t_i)_{i \in \mathbb{N}_0}$ is a real-time sequence.

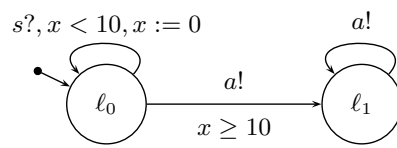
If $\langle \ell_0, \nu_0 \rangle \in C_{ini}$ and $t_0 = 0$, then we call ξ a **run** of \mathcal{A} .

Example:



does not have a path

Example



- $\langle l_0, 0 \rangle, 0 \xrightarrow{s?} \langle l_0, 0 \rangle, 0 \xrightarrow{!0} \langle l_0, 1 \rangle, 1 \xrightarrow{s?} \langle l_0, 0 \rangle, 1 \xrightarrow{!0} \langle l_0, 1 \rangle, 2 \xrightarrow{s?} \langle l_0, 0 \rangle, 2 \dots$ RUN
- $\langle l_0, 0 \rangle, 0 \xrightarrow{!0} \langle l_0, 10 \rangle, 10 \xrightarrow{a!} \langle l_1, 10 \rangle, 10 \xrightarrow{a!} \langle l_1, 10 \rangle, 10 \xrightarrow{a!} \dots$ NOT A RUN
- $\langle l_0, 0 \rangle, 0 \xrightarrow{!0} \langle l_0, 10 \rangle, 10 \xrightarrow{!0} \langle l_1, 11 \rangle, 11 \xrightarrow{!0} \langle l_1, 12 \rangle, 12 \xrightarrow{!0} \dots$ RUN

References

References

- [Behrmann et al., 2004] Behrmann, G., David, A., and Larsen, K. G. (2004). A tutorial on uppaal 2004-11-17. Technical report, Aalborg University, Denmark.
- [Olderog and Dierks, 2008] Olderog, E.-R. and Dierks, H. (2008). Real-Time Systems - Formal Specification and Automatic Verification. Cambridge University Press.