Lecture 8

- labelled transition systems with time
- timed automata
- timed and untimed bisimilarity
- timed and untimed language equivalence
- networks of timed automata

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.
  - But too abstract for certain questions (like "What is the average time to deliver the message?").
- 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).
  - ...

Labelled Transition Systems with Time

Timed (labelled) transition system (TLTS)

TLTS is a tuple $(Proc, Act, (\stackrel{a}{\longrightarrow})_{a \in Act})$ 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 $a \in Act$, $\stackrel{a}{\longrightarrow} \subseteq Proc \times Proc$ is a binary relation on states called the transition relation.

We write

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

Natural requirements for TLTS

- 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 $s' = s''$.

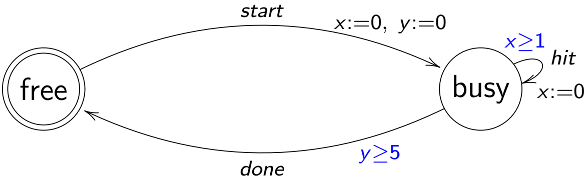
How to Describe Timed Transition Systems?

Syntax		Semantics
CCS	$\longrightarrow$	Labelled Transition Systems
???	$\longrightarrow$	Timed Transition Systems

A possible answer

Timed Automata [Alur, Dill'90]
i.e., finite-state automata equipped with real-valued clocks.

Example: Hammer



Some important questions:

- How to model a real-time system?
- What guards should we allow?
- ....

Definition of TA: Clock Constraints

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

$\mathcal{B}(C)$ ... the set of clock constraints over C

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

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

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

Example: $x \leq 3 \ \wedge \ y > 0$

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 ($v \models g$)

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

Syntax of Timed Automata

Definition

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

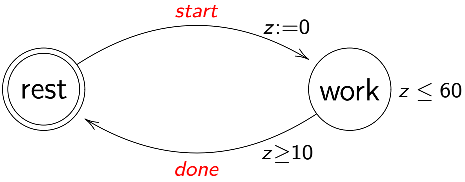
$$(L, \ell_0, E, I)$$

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 $\ell \xrightarrow{g,a,r} \ell'$ instead of (ℓ, g, a, r, ℓ') for the edges in E.

Example: A Worker



What does this mean?

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

Timed transition system generated by A

$T(A) = (Proc, Act, (\xrightarrow{a})_{a \in Act})$ 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 such that $v \models I(\ell)$
- $Act = N \cup \mathbb{R}^{\geq 0}$
- $\xrightarrow{a}$ and $\xrightarrow{d}$ are defined as follows:

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

$(\ell, v) \xrightarrow{d} (\ell, v + d)$ if $v + d' \models I(\ell)$ for each $d' \in [0, d]$.

Timed Bisimilarity

Let A_1 and A_2 be timed automata.

Timed Bisimilarity

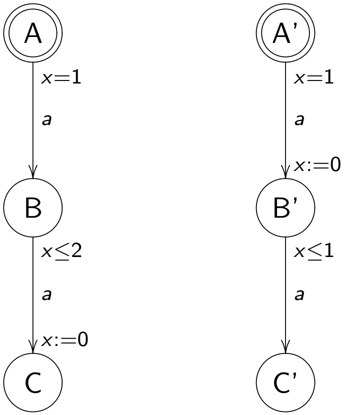
We say that A_1 and A_2 are timed bisimilar iff the transition systems $T(A_1)$ and $T(A_2)$ generated by A_1 and A_2 are strongly bisimilar.

Remark: both

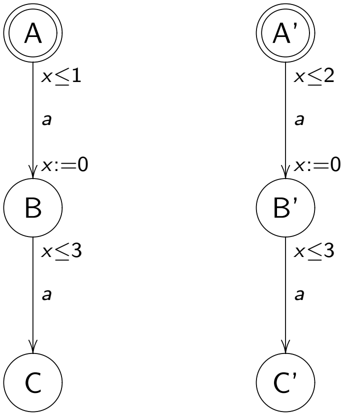
- $\xrightarrow{a}$ for $a \in N$ and
- $\xrightarrow{d}$ for $d \in \mathbb{R}^{\geq 0}$

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.

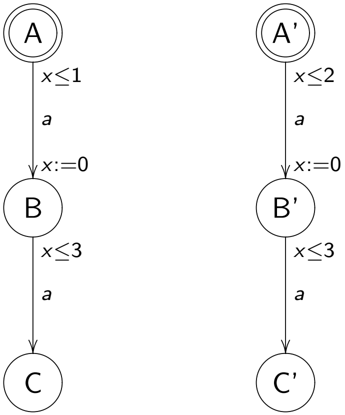
Remark:

- $\xrightarrow{a}$ 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 $\xrightarrow{\epsilon}$.

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 in EXPTIME (decidable in deterministic exponential time).

Theorem [Larsen, Wang'93]

Untimed bisimilarity for timed automata is in EXPTIME.

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 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.

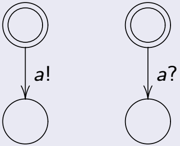
We say that $a_1 a_2 a_3 \ldots$ is an untimed trace of A iff there exist $t_1, t_2, t_3, \ldots \in \mathbb{R}^{\geq 0}$ such that $(t_1, a_1)(t_2, a_2)(t_3, a_3)\ldots$ is a timed trace of A.

Theorem [Alur, Dill'94]

Untimed language equivalence for timed automata is decidable.

Networks of Timed Automata

Timed Automata in Parallel



Intuition in CCS

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

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

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

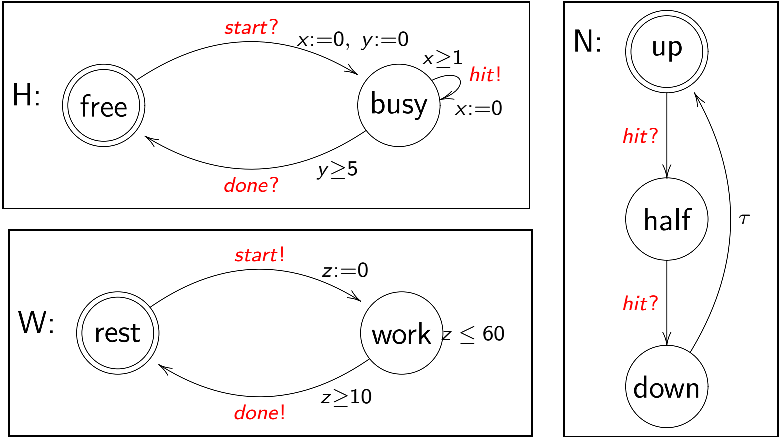
- $N = \{c! \mid c \in Chan\} \cup \{c? \mid c \in Chan\} \cup \{\tau\}$.

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 network of timed automata.

Example: Hammer, Worker, Nail



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

$T(A) = (Proc, Act, (\xrightarrow{a})_{a \in Act})$ 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, $v \models I_i(\ell_i)$
- $Act = \{\tau\} \cup \mathbb{R}^{\geq 0}$
- $\xrightarrow{a}$ are 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 $v + d \models \bigwedge_k I_k(\ell_k)$

$((\ell_1,\ldots,\ell_i,\ldots,\ell_j,\ldots,\ell_n),v) \xrightarrow{\tau} ((\ell_1,\ldots,\ell'_i,\ldots,\ell'_j,\ldots,\ell_n),v')$ 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 $v' \models I_i(\ell'_i) \wedge I_j(\ell'_j) \wedge \bigwedge_{k\neq i,j} I_k(\ell_k)$