# Semantics and Verification 2005

Lecture 10

- region graph and the reachability problem
- networks of timed automata
- model checking of timed automata



## 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.



## 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 $A$.



## 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 $\ell$
2. $\equiv$ has only finitely many equivalence classes.



## 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
$$frac(v(x)) = 0 \quad \textbf{iff} \quad frac(v'(x)) = 0$$
3. for all $x, y \in C$ such that $v(x) \leq c_x$ and $v(y) \leq c_y$ we have
$$frac(v(x)) \leq frac(v(y)) \quad \textbf{iff} \quad frac(v'(x)) \leq frac(v'(y))$$



## 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 $\ell$ 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 $(\ell, v)$ $\quad\rightsquigarrow\quad$ **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**
- $\Longrightarrow$ on 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**.



## 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 $(\ell, v)$.

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

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

### Reduction of Timed Automata Reachability to Region Graphs

Reachability for timed automata is decidable because

$(\ell_0, v_0) \longrightarrow^* (\ell, v)$ in a timed automaton if and only if
$(\ell_0, [v_0]) \Longrightarrow^* (\ell, [v])$ in its (finite) region graph.



## Applicability of Region Graphs

### Pros

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.



## Zones and Zone Graphs

Zones provide a more efficient representation of symbolic state spaces. A number of regions can be described by one zone.

### Zone

A zone is described by a **clock constraint** $g \in \mathcal{B}(C)$.

$$[g] = \{v \mid v \models g\}$$

### Region Graphs

symbolic state: $(\ell, [v])$
where $v$ is a clock valuation

### Zone Graphs

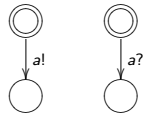
symbolic state: $(\ell, [g])$
where $g$ is a clock constraint

A zone is usually represented (and stored in the memory) as
**DBM (Difference Bound Matrix)**.



## Networks of Timed Automata

### Timed Automata in Parallel




### Intuition in CCS

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

Let $C$ be a set of clocks and $Chan$ a set of channels.
We let $Act = N \cup \mathbb{R}^{\geq 0}$ where

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

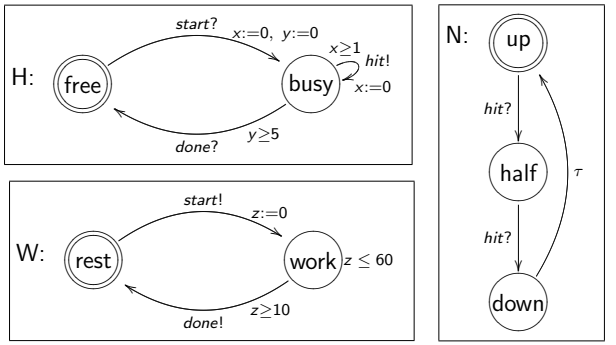
Let $A_i = (L_i, \ell_0^i, E_i, I_i)$ be timed automata for $1 \leq i \leq n$.

### Networks of Timed Automata

We call $A = A_1 | A_2 | \cdots | A_n$ a **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 $\ell_i$ 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 $v' \models I_i(\ell'_i) \wedge I_j(\ell'_j) \wedge \bigwedge_{k \neq i,j} I_k(\ell_k)$



## Logic for Timed Automata in UPPAAL

Let $\phi$ and $\psi$ be **local properties** (check-able locally in a given state).

Example: (H.busy $\wedge$ W.rest $\wedge$ $20 \leq z \leq 30$)

UPPAAL can check the following formulae (subset of TCTL)

- **A**[]$\phi$ — invariantly $\phi$
- **E**$\langle\rangle\phi$ — possibly $\phi$
- **A**$\langle\rangle\phi$ — always eventually $\phi$
- **E**[]$\phi$ — potentially always $\phi$
- $\phi$ –> $\psi$ — $\phi$ always leads to $\psi$ (same as A[]($\phi \implies$ A$\langle\rangle\psi$))

Legend:

- A and E are so called path quantifiers, and
- [] and $\langle\rangle$ quantify over states of a selected path.