Timed Automata

Stefan Ratschan

Katedra číslicového návrhu
Fakulta informačních technologií
České vysoké učení technické v Praze

Evropský sociální fond Praha & EU: Investujeme do vaší budoucnosti

How can we describe and computationally analyze complex systems?

complex system

discrete-time system

describe

discrete-time automaton, transition system

Petri net

timed automaton

computationally analyze

testing, BMC, unbounded MC

constraint solver (SAT, linear programming etc.)

Uppaal Demo

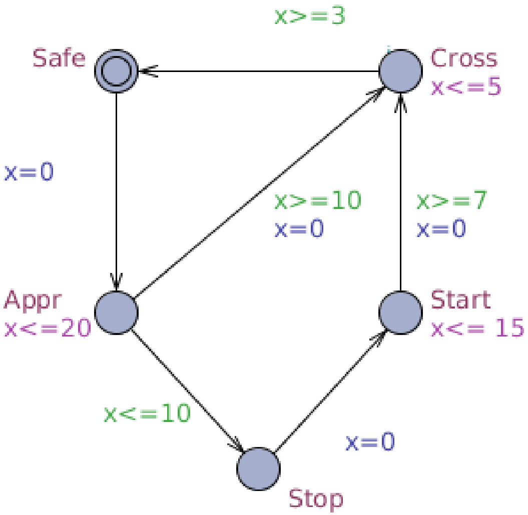
Train-gate example

Basic formal model: timed automaton

Uppaal:

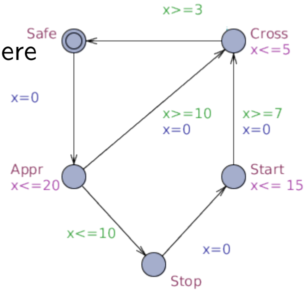
- timed automata +
- composition +
- message passing +
- finite data structures

Train-Gate



Timed Automata

Clock constraint: conjunction of strict or non-strict inequalities between variables and rational numbers, denoted by $\mathcal{C}(X)$



A *timed automaton* is a 5-tuple $(L, L_0, X, \mathcal{I}, T)$ where

- L is a finite set (*locations*)
- $L_0 \subseteq L$ (*initial locations*)
- X is a finite set (*clocks*)
- $\mathcal{I} : L \to \mathcal{C}(X)$ (*location invariants*)
- $\mathcal{T} \subseteq L \times \mathcal{C}(X) \times \mathcal{P}(X) \times L$
  (*transitions* with *guards* and *resets*)

Example: Train: $(\{Safe, Appr, Stop, Start, Cross\}, \{Safe\}, \{x\},$
$\{Appr \mapsto x \leq 20, Start \mapsto x \leq 15, Cross \mapsto x \leq 5, Safe \mapsto \top, Stop \mapsto \top\},$
$\{(Safe, \top, \{x\}, Appr), (Appr, x \leq 10, \{\}, Stop), (Stop, \top, \{x\}, Start), \dots\})$

Attention: overloading of name "invariant"!

No inputs/outputs, can be added

State of Timed Automaton

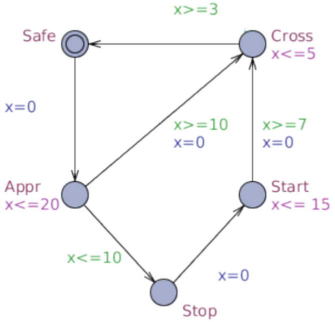
Example:

- currently in location Appr, and
- 10 seconds have elapsed since resetting clock x

State: pair (l, v), where

- $l \in L$ is a location,
- $v : X \to \mathbb{R}^{\geq 0}$ (*clock assignment*)

Example: $(Appr, \{x \mapsto 10\})$



Evolution of State

Two possibilities:

- Elapsing of time while staying at the same location, for example, from $(Appr, \{x \mapsto 10, y \mapsto 7\})$ to $(Appr, \{x \mapsto 15, y \mapsto 12\})$
- Changing locations, e.g., from $(Safe, \{x \mapsto 355\})$ to $(Appr, \{x \mapsto 0\})$

Evolution of time: For *clock assignment* v, $d \in \mathbb{R}^{\geq 0}$,
$v + d$ is a clock assignment such that
for all $x \in X$, $(v + d)(x) := v(x) + d$.

Formalization, evolution of state:

- *Delay transition*: $(l, v) \xrightarrow{d} (l, v + d)$, where $d \in \mathbb{R}^{\geq 0}$, iff for every $e \in [0, d]$, $v + e \models \mathcal{I}(l)$
- *Action transition*: $(l, v) \hat{\rightarrow} (l', v')$, iff there is a $(l, \phi, \lambda, l') \in \mathcal{T}$ s.t. $v \models \phi$, and for all $x \in X$, $v'(x) = \begin{cases} 0, & \text{if } x \in \lambda, \\ v(x), & \text{otherwise.} \end{cases}$

Evolution of Timed Automaton

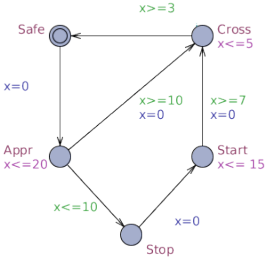
Transition: combination of delay and action transition, that is :
$(l, v) \to (l', v')$ if there is $d \in \mathbb{R}^{\geq 0}$ s.t.

$$(l, v) \xrightarrow{d} (l, v + d), (l, v + d) \hat{\to} (l', v')$$

Example:
$(Cross, \{x \mapsto 0\}) \to (Safe, \{x \mapsto 4\})$ because

$$(Cross, \{x \mapsto 0\}) \xrightarrow{4} (Cross, \{x \mapsto 4\}),$$
$$(Cross, \{x \mapsto 4\}) \hat{\to} (Safe, \{x \mapsto 4\})$$

with choice $d \leftarrow 4$



Non-determinism: from some states several different transitions possible
(choice of d, further location s')

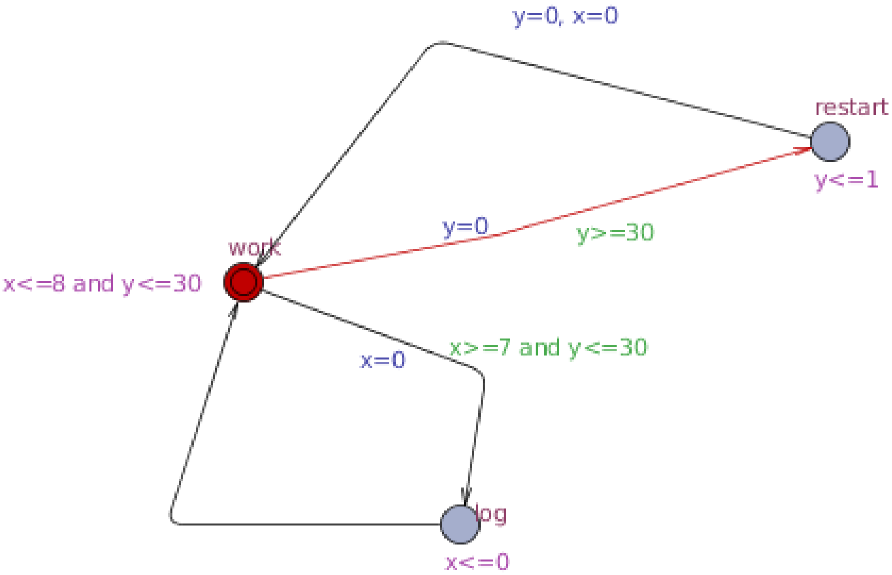
Evolution of Timed Automaton

Set of initial states

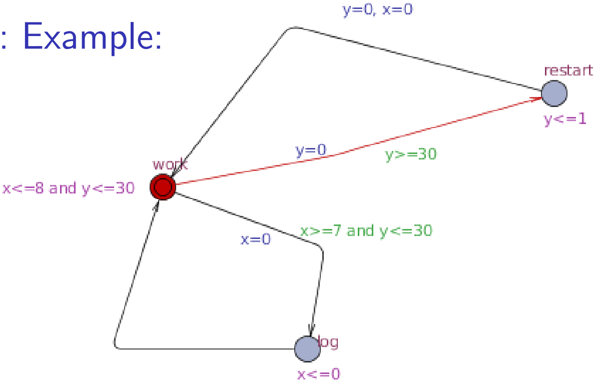
$$\{(l, v) \mid l \in L_0, \text{for all } x \in X, v(x) = 0\}$$

Hence: timed automata can be viewed as infinite transition systems
$(L \times [X \to \mathbb{R}^{\geq 0}], \{(l, v) \mid l \in L_0, \text{for all } x \in X, v(x) = 0\}, \to)$

Path, set of reachable states, safety verification conditions, invariant (of transition system), inductive invariant, ...


y=0, x=0
restart
y<=1
y=0
y>=30
work
x<=8 and y<=30
x=0
x>=7 and y<=30
log
x<=0

Simulation: Example:



$$
\begin{array}{ll}
(\mathit{work}, \{x \mapsto 0, y \mapsto 0\}) & \xrightarrow{7} (\mathit{work}, \{x \mapsto 7, y \mapsto 7\}) \hat{\rightarrow} \\
(\mathit{log}, \{x \mapsto 0, y \mapsto 7\}) & \xrightarrow{0} (\mathit{log}, \{x \mapsto 0, y \mapsto 7\}) \hat{\rightarrow} \\
(\mathit{work}, \{x \mapsto 0, y \mapsto 7\}) & \xrightarrow{8} (\mathit{work}, \{x \mapsto 8, y \mapsto 15\}) \hat{\rightarrow} \\
(\mathit{log}, \{x \mapsto 0, y \mapsto 15\}) & \xrightarrow{0} (\mathit{log}, \{x \mapsto 0, y \mapsto 15\}) \hat{\rightarrow} \\
\dots & \\
(\mathit{restart}, \{x \mapsto 0, y \mapsto 0\}) & \xrightarrow{1} (\mathit{restart}, \{x \mapsto 1, y \mapsto 1\}) \hat{\rightarrow} \\
\dots &
\end{array}
$$

Simulation

Example:

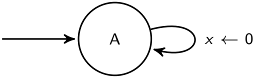
$$
\begin{array}{ll}
(\mathit{work}, \{x \mapsto 0, y \mapsto 0\}) & \xrightarrow{7} (\mathit{work}, \{x \mapsto 7, y \mapsto 7\}) \hat{\rightarrow} \\
(\mathit{log}, \{x \mapsto 0, y \mapsto 7\}) & \xrightarrow{0} (\mathit{log}, \{x \mapsto 0, y \mapsto 7\}) \hat{\rightarrow} \\
(\mathit{work}, \{x \mapsto 0, y \mapsto 7\}) & \xrightarrow{8} (\mathit{work}, \{x \mapsto 8, y \mapsto 15\}) \hat{\rightarrow} \\
(\mathit{log}, \{x \mapsto 0, y \mapsto 15\}) & \xrightarrow{0} (\mathit{log}, \{x \mapsto 0, y \mapsto 15\}) \hat{\rightarrow} \\
\ldots & \\
(\mathit{restart}, \{x \mapsto 0, y \mapsto 0\}) & \xrightarrow{1} (\mathit{restart}, \{x \mapsto 1, y \mapsto 1\}) \hat{\rightarrow} \\
\ldots &
\end{array}
$$

let (l, v) be s.t. $l \in L_0$, $\forall x \in X \,.\, v(x) = 0$
loop
    choose $d \in \mathbb{R}^{\geq 0}$, transition $(l, \phi, \lambda, l') \in \mathcal{T}$ such that
        $\forall e \in [0, d] \,.\, v + e \models \mathcal{I}(l)$ *// invariant not violated*
        $v + d \models \phi$ *// guard for transition to l' holds*
    let (l, v) be result of action transition from $(l, v + d)$ using (l, ϕ, λ, l')

Specification

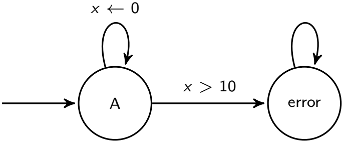
We have a transition system, so we can use LTL:

- $\mathbf{G}ok$, where $\mathcal{I}(ok) = \{(l, v) \mid l = restart \Rightarrow v(y) \leq 20\}$
- $\mathbf{F}goal$ where $\mathcal{I}(goal) = \{(l, v) \mid l = restart\}$



$\mathbf{G}ok$, where $\mathcal{I}(ok) = \{(l, v) \mid v(x) \leq 10\}$?

$$(A, 0) \stackrel{935}{\to} (A, x \mapsto 935) \stackrel{\hat{}}{\to} (A, 0)$$



$\mathbf{G}ok$, where $\mathcal{I}(ok) = \{(l, v) \mid l = A\}$

Testing

Question: Test cases fulfill specification?

How to generate and store test cases?

Simulation:

let (l, v) be s.t. $l \in L_0$, $\forall x \in X \,.\, v(x) = 0$
loop
    choose $d \in \mathbb{R}^{\geq 0}$, transition $(l, \phi, \lambda, l') \in \mathcal{T}$ such that
        $\ldots$

Problem: further occurrence of infinity:

- infinite length of paths
- infinite number of paths
- from each state infinite (even uncountable) number of successors

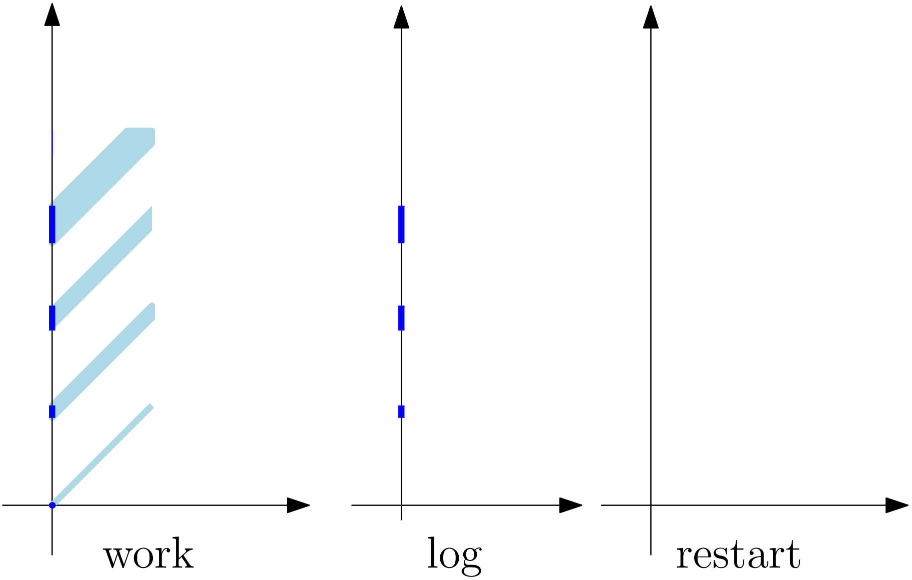
One reason why testing of real-time systems is so difficult.

BMC [Audemard et al., 2002, Cotton and Maler, 2006]:
SAT+constraint solver (difference logic)

Symbolic Simulation

Representation of something infinite? Symbolic representation!

How?


work
log
restart

Symbolic Simulation

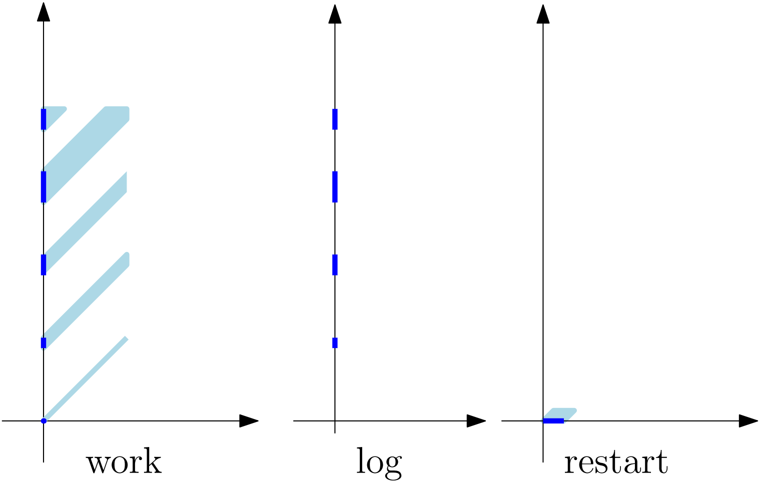
$$
\begin{array}{lcl}
(work, x = 0 \wedge y = 0) & \tilde{\to} & (work, x \leq 8 \wedge x = y) \hat{\to} \\
(log, x = 0 \wedge 7 \leq y \leq 8) & \tilde{\to} & (log, x = 0 \wedge 7 \leq y \leq 8) \hat{\to} \\
(work, x = 0 \wedge 7 \leq y \leq 8) & \tilde{\to} & (work, x \leq 8 \wedge 7 \leq y - x \leq 8) \hat{\to} \\
(log, x = 0 \wedge 14 \leq y \leq 16) & \tilde{\to} & (log, x = 0 \wedge 14 \leq y \leq 16) \hat{\to} \\
(work, x = 0 \wedge 14 \leq y \leq 16) & \tilde{\to} & (work, x \leq 8 \wedge 14 \leq y - x \leq 16) \hat{\to} \\
(log, x = 0 \wedge 21 \leq y \leq 24) & \tilde{\to} & (log, x = 0 \wedge 21 \leq y \leq 24) \\
\ldots & &
\end{array}
$$

$\tilde{\to}$: delay transitions

So: *symbolic state* (l, ν), where
l is a location and ν is a conjunction of inequalities,
representing the set of states $[\![(l, \nu)]\!] := \{(l, v) \mid v \models \nu\}$

Computing Delay Transitions from Symbolic States

$$(work, x = 0 \wedge 21 \leq y \leq 24) \tilde{\to} (work, x \leq 8 \wedge y \leq 30 \wedge 21 \leq y - x \leq 24)$$



Computing Delay Transitions from Symbolic States

$$(work, x = 0 \wedge 21 \leq y \leq 24) \tilde{\rightarrow} (work, x \leq 8 \wedge y \leq 30 \wedge 21 \leq y - x \leq 24)$$

In general (one unique symbolic successor state):

$$(l, \nu) \tilde{\rightarrow} (l', \nu')$$

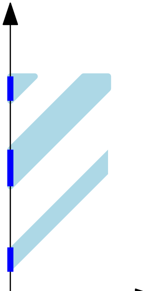
where

$$\begin{aligned} [\![(l', \nu')]\!] = & \quad \{(l', v') \mid \exists (l, v) \in [\![(l, \nu)]\!] \, \exists d \in \mathbb{R}^{\geq 0} \,.\, (l, v) \xrightarrow{d} (l', v')\} \\ = & \quad (l, \{v + d \mid v \models \nu, d \in \mathbb{R}^{\geq 0}, \forall e \in [0, d] \,.\, v + e \models \mathcal{I}(l)\}) \end{aligned}$$

For representing set without invariant
we need $x - y \prec c$, where $\prec$ is $<$ or $\leq$

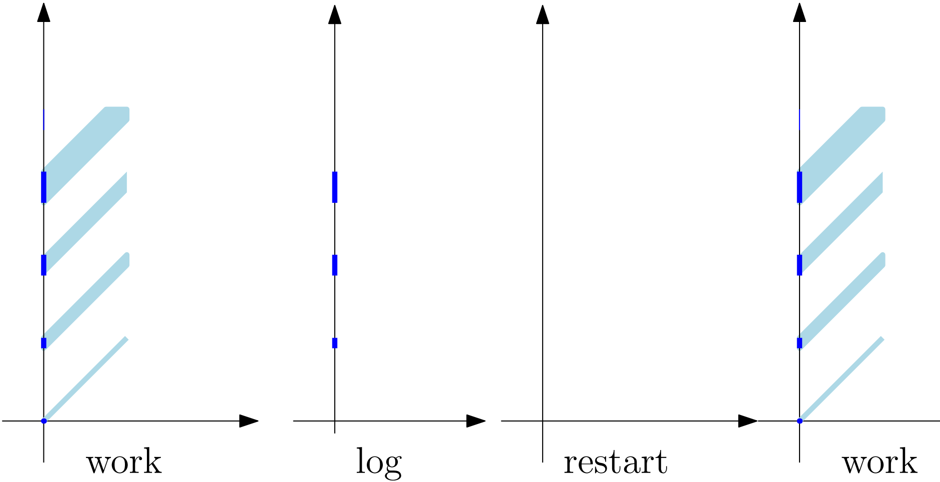
For representing invariants we need $x \prec c$, $c \prec x$

So: conjunction of such inequalities (*clock zone*)



Computing Action Transitions from Symbolic States

$(work, x \leq 8 \wedge y \leq 30 \wedge 21 \leq y - x \leq 24) \hat{\rightarrow}$



$(log, x = 0 \wedge 28 \leq y - x \leq 30), (restart, y = 0 \wedge 6 \leq x \leq 8)$

Computing Action Transitions from Symbolic States

$$(work, x \leq 8 \wedge y \leq 30 \wedge 21 \leq y - x \leq 24) \hat{\rightarrow}$$

Two possible transitions (symbolic simulation chooses one):

$$(log, x = 0 \wedge 28 \leq y - x \leq 30), (restart, y = 0 \wedge 6 \leq x \leq 8)$$

In general, for every symbolic state (l, ν) and transition from l to l' in $\mathcal{T}$ we have a corresponding symbolic action transition

$$(l, \nu) \hat{\rightarrow} (l', \nu')$$

with

$$\begin{aligned} [\![(l', \nu')]\!] &= \{(l', v') \mid \exists (l, v) \in [\![(l, \nu)]\!] \,.\, (l, v) \hat{\rightarrow} (l', v')\} \\ &= \{(l', v[\lambda \leftarrow 0]) \mid v \models \nu, (l, \phi, \lambda, l') \in \mathcal{T}, v \models \phi\} \end{aligned}$$

Computing Action Transitions from Symbolic States: Example:

$(l,\nu)\hat{\to}(l',\nu')$ with

$$[\![(l',\nu')]\!] = \quad \{(l', v[\lambda \leftarrow 0]) \mid v \models \nu, (l,\phi,\lambda,l') \in \mathcal{T}, v \models \phi\}$$

Current symb. state: $(work, x \leq 8 \wedge y \leq 30 \wedge 21 \leq y - x \leq 24)$

Transition: $(work, y \geq 30, \{y\}, restart)$

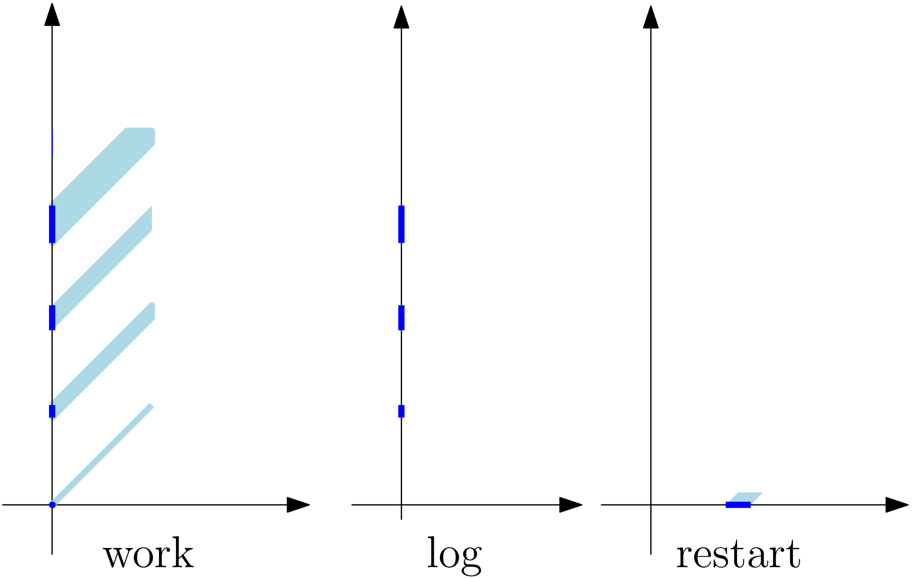
Current clock zone: $x \leq 8 \wedge y \leq 30 \wedge 21 \leq y - x \leq 24$

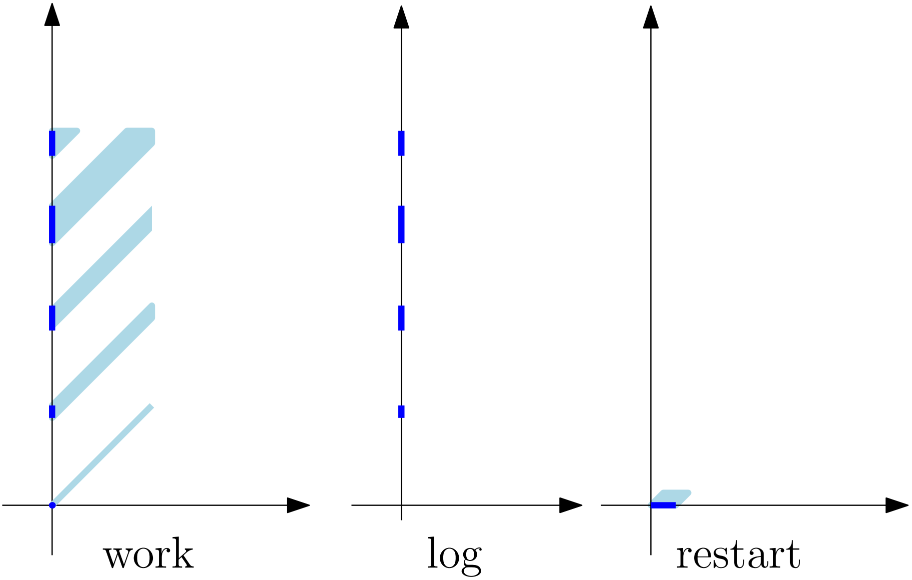
- Intersection with guard: $x \leq 8 \wedge 21 \leq y - x \leq 24 \wedge y \leq 30 \wedge y \geq 30$
- Result of simplification: $6 \leq x \leq 8 \wedge y = 30$
- Reset: $6 \leq x \leq 8 \wedge y = 0$

The result is again a clock zone

Symbolic successor state:

$$(restart, 6 \leq x \leq 8 \wedge y = 0)$$


work
log
restart


work
log
restart

Computation of Successor States

Symbolic transition:
combination of symbolic delay and action transition.

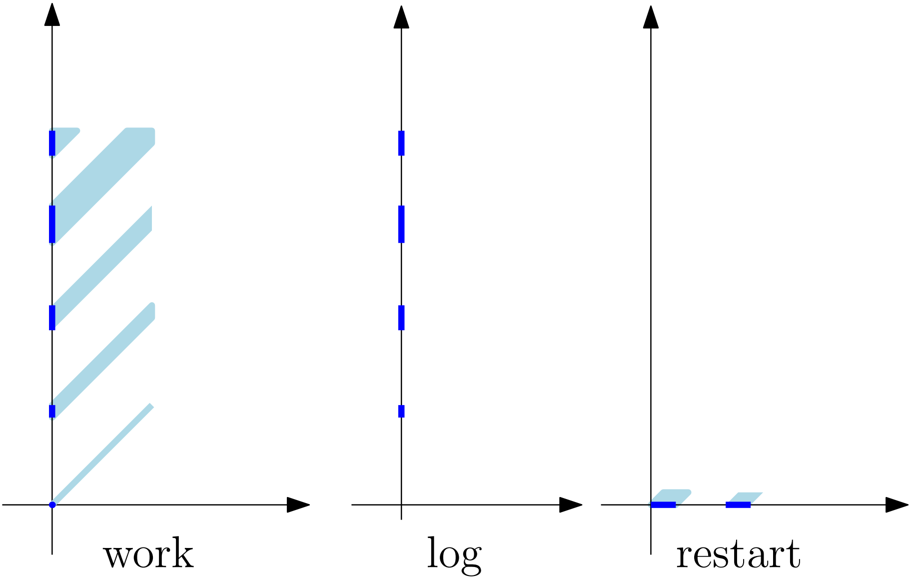
Observation:

- symbolic delay transitions: unique successor
- symbolic action transitions: not more successors than transitions in $\mathcal{T}$

Hence: symbolic simulation: finitely many successor states

So: We can compute all symbolic successor states
and use them to represent
all successor states of the transition system of the timed automaton

So: We can compute all states resulting from one transition
(of the corresponding transition system)


work
log
restart

Unbounded Model Checking

$V \leftarrow S_0$
while there is a transition (x, x') such that $x \in V$, $x' \notin V$ **do**
    $V \leftarrow V \cup \{x' \mid (x, x') \in R, x \in V\}$ // $V \cup Post_R(V)$
return V

Here: represent the set of states V by a set of symbolic states.

Symbolic states may represent overlapping sets, e.g.,

$$\{(work, 0 \leq x \leq 10)), (work, 5 \leq x \leq 15)\}$$

subsets may be removed

Decidability of Unbounded Model Checking

Termination? ✓
(Observation: only integer constants, uniform behavior outside of biggest constant)

Hence: Although timed automata have an infinite state space, checking of $\mathbf{G}ok$ (and all of LTL) decidable.

In Practice

UPPAAL demo

optimized representation of clock zones
(*difference bounded matrices*)

various further optimizations

Industrial usage: uppaal.pdf

Conclusions

- Timed automata provide possibility of modeling behavior in time
- Infinite state space, still decidable
- Key: symbolic representation of infinite sets of states

Symbolic simulation:
For certain choice of finite part of state space (i.e., locations),
all corresponding choices for infinite part (clock assignments)

More and more popular also in the case of software (viz. MI-FME)

Checking LTL decidable, possible in practice

Timed automata very useful for modeling real time systems, but
not enough for modeling of physical phenomena and laws.

Literature

R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

G. Audemard, A. Cimatti, A. Kornilowicz, and R. Sebastiani. Bounded model checking for timed systems. In Doron A. Peled and Moshe Y. Vardi, editors, *Formal Techniques for Networked and Distributed Sytems—FORTE 2002*, volume 2529 of *Lecture Notes in Computer Science*, pages 243–259. Springer Berlin Heidelberg, 2002.

Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In *Lectures on Concurrency and Petri Nets*, volume 3098 of *LNCS*, pages 87–124. Springer, 2004.

Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 1999.

Scott Cotton and Oded Maler. Fast and flexible difference constraint propagation for DPLL(T). In *International Conference on Theory and Applications of Satisfiability Testing*, pages 170–183. Springer, 2006.