On Decidability of Prebisimulation for Timed Automata

Shibashis Guha, Chinmay Narayan, S. Arun-Kumar

Department of Computer Science & Engineering
Indian Institute of Technology, Delhi

July 12, 2012

Motivation

- Real time systems require performance and *timing* constraints are satisfied.
- Given two systems with same behavior, determine which performs better in terms of time.

Example

Timed Automata formalism to model systems

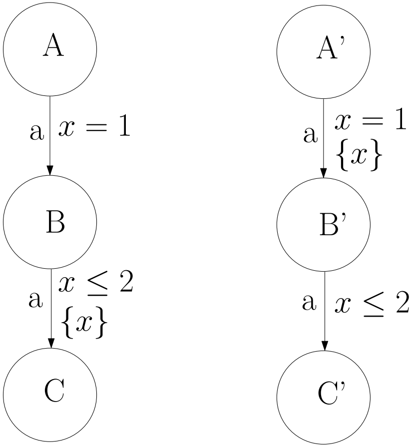


Figure: Example: An *at least as fast as* relation

Contribution

- Defined a relation between two timed (automata) systems to compare their performances.
  Timed Performance Prebisimulation

- Designed an algorithm to decide *timed performance prebisimulation* relation

Related Work

- Timed Actor Interfaces [Geilen, Tripakis, Wiggers 11]
- Performance Preorder [Corradini, Gorrieri, Roccetti 95]
- Efficiency Preorder [S. Arun-Kumar, Hennessy 91]

Timed Automata

Definition

- Set of clocks C, finite set of actions Act.
- The clock constraints $\mathcal{B}(C)$ over a set of clocks C can be specified using the following grammar:

  $$g ::= \ x \smile c \mid g \wedge g$$

  where $c \in \mathbb{N}$ and $x \in C$ and $\smile \in \{<, \leq, =, >, \geq\}$.
- timed automaton over a finite set of clocks C and finite set of actions Act is the quadruple

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

  where
  L is a finite set of locations, ranged over by ℓ,
  $\ell_0 \in L$ is the initial location,
  $E \subseteq L \times \mathcal{B}(C) \times Act \times 2^C \times L$ is a finite set of *edges*, and
  $I : L \to \mathcal{B}(C)$ assigns invariants to locations.

Timed automaton Semantics: Timed Labeled Transition System (TLTS)

- Infinite transition graph structure
- Nodes are timed automaton states or configurations; tuple (ℓ, v)
- Two types of transitions

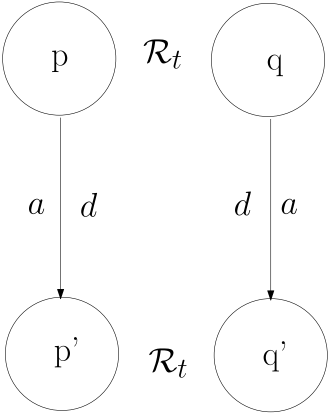
  $a \in Act$: $(\ell, v) \xrightarrow{a} (\ell', v')$ if there is an edge $(\ell \xrightarrow{g,a,r} \ell') \in E$ and $v \models g, v' = v[r]$ and $v' \models I(\ell')$

  $d \in \mathbb{R}_{\geq 0}$: $(\ell, v) \xrightarrow{d} (\ell, v + d)$ such that $v \models I(\ell)$ and $v + d \models I(\ell)$.

Timed Equivalences

Timed Bisimulation

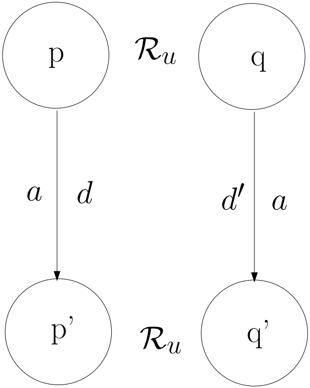
p and q are two timed valuations.



$$a \in Act, d \in R_{\geq 0}$$

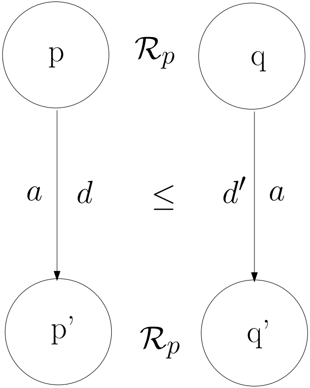
Timed Equivalences

Time Abstracted Bisimulation



$a \in \ Act, \ d, d' \in R_{\geq 0}$

Timed Performance Prebisimulation



$$a \in Act,\ d, d' \in R_{\geq 0}$$

$$\sim_t \subseteq \precsim \subseteq \sim_u$$

captures functional behaviour and performance simultaneously

Decidability

- Timed Bisimualation and Time Abstracted bisimulation have been proved to be decidable for timed automata.

- Is Timed Performance Prebisimulation decidable?

  Yes

Decidability

- Timed Bisimualation and Time Abstracted bisimulation have been proved to be decidable for timed automata.

- Is Timed Performance Prebisimulation decidable?

  Yes

Algorithm

Outline

- Given two timed automata A_1 and A_2 or two reachable configurations p and q, in timed automata, create the **zone valuation graphs** $Z_{(A_1,p)}$ and $Z_{(A_2,q)}$.
- Check for strong bisimilarity between the initial nodes of the zone valuation graphs and simultaneously for every pair (s_1, s_2) of bisimilar nodes in these two zone valuation graphs check if the **span** of s_1 is $\leq$ (or $\geq$) the **span** of s_2.

Zone Graph

A zone is a set of all clock valuations which satisfy a collection of formula of the form $x \smile c$ or $x - y \smile c$.

For a timed automaton $A = (L, l_0, E, I)$, a *zone graph* is a transition system $(S, s_0, Lep, \rightarrow)$, where

- $Lep = Act \cup \{\varepsilon\}$,
- ε is an action corresponding to delay transitions of the processes of the zone,
- $S \subseteq L \times \Phi_{\vee}(C)$ is the set of nodes, $s_0 = (l_0, \phi_0(C))$, $\rightarrow \subseteq S \times Lep \times S$ is connected,
- $\phi_0(C)$ is the formula where all the clocks in C are 0.

Zone Valuation Graph

A *zone graph* $Z = (S, s_0, Lep, \rightarrow)$ with the following properties

1. set S is finite.
2. For every node $s \in S$ the zone corresponding to the constraints ϕ_s is convex.
3. $v_{l_j} \models \phi_{s_r}$. Note that v_{l_j} may or may not satisfy $\phi_0(C)$.
4. For any two processes $p, q \in T(A)$, if their valuation satisfies the formula ϕ_r for the same node $r \in S$ then $p \sim_u q$, i.e. p is time abstracted bisimilar to q.
5. For two timed automata A_1, A_2 and two processes $p \in T(A_1)$ and $q \in T(A_2)$, $Z_{(A_1,p)} \sim Z_{(A_2,q)} \Leftrightarrow p \sim_u q$.
6. It is minimal to the extent of preserving convexity of the zones.

Stages of Creating Zone Valuation Graph

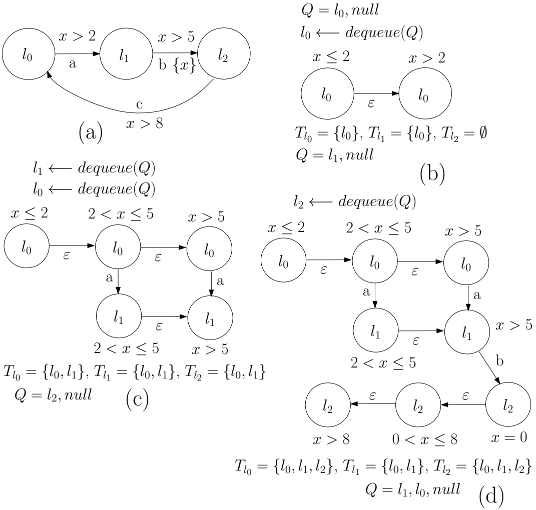
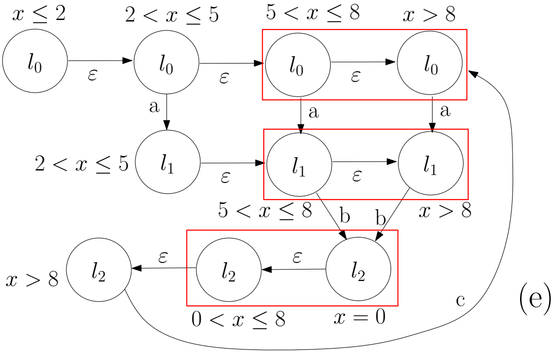


Figure: Successive stages of creating the zone valuation graph

Stages of Creating Zone Valuation Graph

$l_1 \longleftarrow dequeue(Q)$

$l_0 \longleftarrow dequeue(Q)$



$T_{l_0} = \{l_0, l_1, l_2\}$, $T_{l_1} = \{l_0, l_1, l_2\}$, $T_{l_2} = \{l_0, l_1, l_2\}$ $\quad Q = null$

Figure: Final zone valuation graph

Not the Full Story

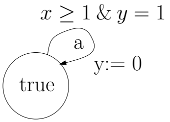


Figure: Timed Automaton with infinite zone graph

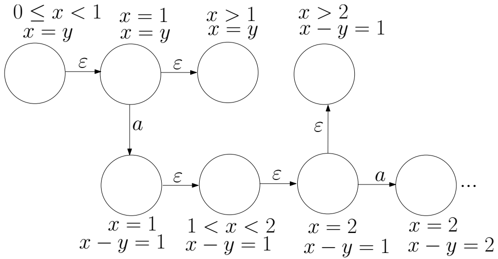


Figure: Infinite zone graph

Abstraction: Location Dependent Maximum Constants

- *Static Guard Analysis in Timed Automata Verification* Behrmann et. al. 03
- For each clock $x \in C$ and each locaion $l \in L$, a maximum constant max_x^l is determined beyond which the actual value of x in l is irrelevant. For a location l and a clock x, $max_x^l \leq c_x$, the global maximum constant with which clock x is compared.
- Thus the number of nodes reduced compared to region graph abstraction.

Zone Graph with Abstraction for Automaton

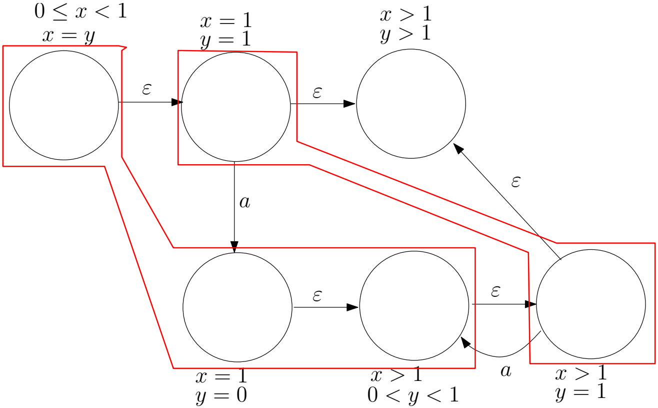


Figure: Abstracted zone graph of Timed Automaton for $max_x^l = 1$ and $max_y^l = 1$

Zone Valuation graph with Abstraction for Automaton

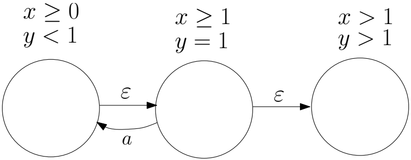


Figure: Canonical abstracted zone graph of Timed Automaton for $max^l_x = 1$ and $max^l_y = 1$

Algorithm

Outline

- Given two timed automata or two reachable configurations in timed automata, create the zone valuation graphs as mentioned above.
- Check for strong bisimilarity between the initial nodes of the zone valuation graphs and simultaneously for every pair (s_1, s_2) of bisimilar nodes in these two zone valuation graphs check if the **span** of s_1 is $\leq$ (or $\geq$) the **span** of s_2.

Example

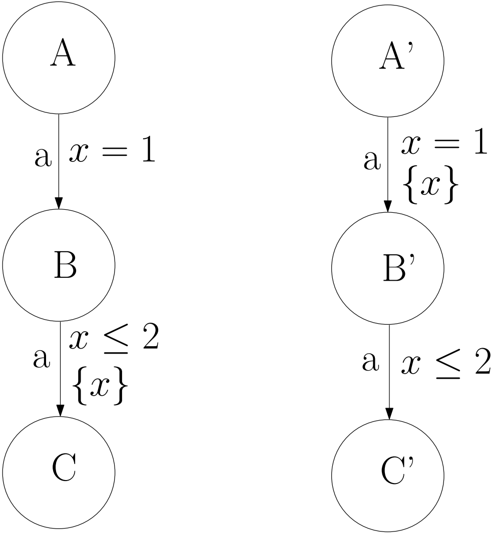


Figure: Example: An *at least as fast as* relation

Zone Valuation Graph: Check Span of Strongly Bisimilar Nodes

Span: Minimum of ranges of clock valuations: $\mathcal{M}(s)$ for node *s*.
critical clock of a node: range equals span

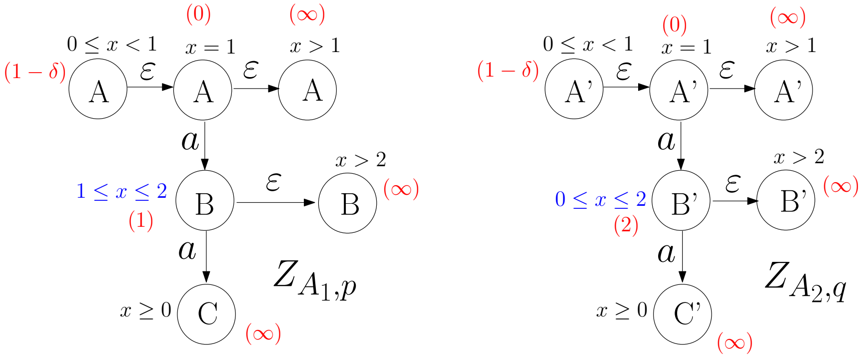


Figure: Zone Valuation Graphs of prebisimilar Timed Automata

Correctness of algorithm

Flip in Delay (FID)

Two zone valuation graphs: $Z_{A_1,p}$ and $Z_{A_2,q}$.

For *any* strong bisimulation relation $\mathcal{B}$, between $Z_{A_1,p}$ and $Z_{A_2,q}$ consider two pairs of bisimilar nodes (s_{p_1}, s_{q_1}) and (s_{p_2}, s_{q_2})

$s_{p_1}, s_{p_2} \in Z_{A_1,p}$ and $s_{q_1}, s_{q_2} \in Z_{A_2,q}$.

FID exists if $\mathcal{M}(s_{p_1}) < \mathcal{M}(s_{q_1})$ and $\mathcal{M}(s_{p_2}) > \mathcal{M}(s_{q_2})$.

Proof of Correctness

Lemma: For $p \in T(A_1)$ and $q \in T(A_2)$,
$FID(Z_{(A_1,p)}, Z_{(A_2,q)}) \Rightarrow (p \not\precsim q \wedge q \not\precsim p)$

Proof Outline: Assume $p \sim_u q$

- $\mathcal{M}(s_{p1}) > \mathcal{M}(s_{q1})$ and $\mathcal{M}(s_{p2}) < \mathcal{M}(s_{q2})$
- $s_{p_1} \sim s_{q_1}$ and $s_{p_2} \sim s_{q_2}$

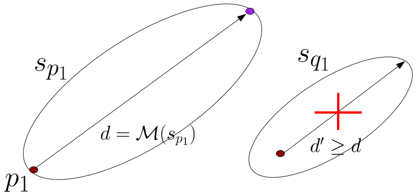


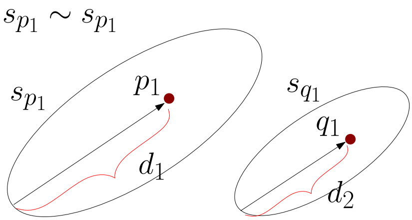
Figure: $\mathcal{M}(s_{p1}) > \mathcal{M}(s_{q1}) \Rightarrow p \not\precsim q$

Similarly, $\mathcal{M}(s_{p2}) < \mathcal{M}(s_{q2}) \Rightarrow q \not\precsim p$

Proof of Correctness

Lemma: For $p \in T(A_1)$ and $q \in T(A_2)$,
$p \sim_u q \wedge \neg FID(Z_{(A_1,p)}, Z_{(A_2,q)}) \Rightarrow p \precsim q \vee q \precsim p$.

Proof Outline: $p \sim_u q \Rightarrow Z_{(A_1,p)} \sim Z_{(A_2,q)}$



$$d_1 = v_{p_1}(x) - min_x(s_{p_1})$$
$$d_2 = d_1 \times (\mathcal{M}(s_{q_1})/\mathcal{M}(s_{p_1}))$$
$$v_{q_1}(y) = min_y(s_{q_1}) + d_2$$

Proof of Correctness

- **Lemma**: For $p \in T(A_1)$ and $q \in T(A_2)$, $FID(Z_{(A_1,p)}, Z_{(A_2,q)}) \Rightarrow (p \not\precsim q \wedge q \not\precsim p)$
- **Lemma**: For $p \in T(A_1)$ and $q \in T(A_2)$, $p \sim_u q \wedge \neg FID(Z_{(A_1,p)}, Z_{(A_2,q)}) \Rightarrow p \precsim q \vee q \precsim p$.
- **Corollary**: For $p \in T(A_1)$ and $q \in T(A_2)$, $q \precsim p \vee p \precsim q \Rightarrow p \sim_u q$ and $\neg FID(Z_{(A_1,p)}, Z_{(A_2,q)})$
- **Theorem**: For $p \in T(A_1)$ and $q \in T(A_2)$, $q \precsim p \vee p \precsim q \Leftrightarrow p \sim_u q$ and $\neg FID(Z_{(A_1,p)}, Z_{(A_2,q)})$

Complexity

Creating Zone Valuation Graph

- Preprocessing: Finding max_x^l for each clock x and each location l: $\mathcal{O}(t^3)$, where $t = |C| \times n$.

- Phase 1: $\mathcal{O}(|S| \times |C| \times n^2 + n^4 \times log\ n)$, where $|S|$ is the number of nodes in zone valuation graph after abstraction.

- Phase 2: Combining nodes that are strongly bisimilar: $\mathcal{O}(|R| \times log\ |S|)$, where $|R|$ is the number of related pairs. [Paige, Tarjan 87]

Checking prebisimulation

- $O(n_1^2 n_2^2 . m_1 m_2 |C| log(n_1 n_2))$, where n_1 and n_2 are the number of nodes in the zone valuation graphs and m_1 and m_2 are the number of edges respectively.

Conclusion and Future Work

- We propose here a zone based algorithm to decide timed performance prebisimulation.
- We have shown how the relation can be established between two protocols for reliable data transfer, *Stop-and-Wait ARQ* and *Alternating bit protocol* and shown that the latter is a better implementation.
- Zone valuation graph can also be used to decide timed bisimulation as well.
- Game characterizations similar to Striling's bisimulation games for timed automata processes.

Future Work

- An implementation to decide timed performance prebisimulation and other similar relations using our approach.

- Define a *weaker* prebisimulation in which one state can be defined to be at least as fast as the other state if the **total** time elapsed is compared over sequence of actions instead of comparing delays at every stage as in timed performance prebisimulation.

- congruence properties, e.g. under parallel composition