Specifying Urgency in Timed I/O Automata

Biniam Gebremichael  Frits Vaandrager
B.Gebremichael@cs.ru.nl  F.Vaandrager@cs.ru.nl

Nijmegen Institute for Computing and Information Sciences
Radboud University of Nijmegen
The Netherlands

January 2005, Enschede, The Netherlands
Motivation

How to specify real-time systems?

- define a clock $x, y \in \mathbb{R}^\geq 0$ with $d(x) = 1$ and it can be reseted $(x := 0)$ [Alur and Dill 1994].

- action enabled: use guards $x \leq 4$

- progress/urgency

[Alur and Henzinger 1994] as in UPPAAL.

transc criterion [Alur and Dill 1994]

[Bornot and Sifakis 1998]

[Alur and Dill 1994]

[Alur and Henzinger 1994]

[Bornot and Sifakis 1998]

[Alur and Dill 1994]

[Alur and Dill 1994]

[Kaynar et al. 2003]
Motivation

How to specifying real-time systems?

- Define a clock $x, y \in \mathcal{R}^{\geq 0}$ with $d(x) = 1$ and it can be resetted $(x := 0)$ [Alur and Dill 1994]

- Action enabled: use guards $x \leq 4$

- Progress/urgency
  - Location invariant [Alur and Henzinger 1994] as in UPPAAL.
  - Deadline on transition [Bornot and Sifakis 1998]
  - Büchi acceptance criterion [Alur and Dill 1994].
  - Stoping condition [Kaynar et al. 2003]
Motivation

How to specifying real-time systems?

- define a clock $x, y \in \mathcal{R}^{\geq 0}$ with $d(x) = 1$ and it can be reseted $(x := 0)$ [Alur and Dill 1994].

- action enabled: use guards $x \leq 4$

- progress/urgency
  - location invariant [Alur and Henzinger 1994] as in UPPAAL.
  - deadline on transition [Bornot and Sifakis 1998]
  - Büchi acceptance criterion [Alur and Dill 1994].
  - stoping condition [Kaynar et al. 2003]

which style of specifying urgency to use for Timed I/O Automata?
Content

1. Introduction
   - Timed I/O Automata
   - Deadline perdicates and Timed Automata with Deadline

2. TIOA with Urgency
   - Adding Urgency
   - I/O distiction
   - Composition

3. Expressivity: Comparison and translation between: urgency predicates, deadline predicates, stopping condition and location invariant.

4. Urgency in UPPAAL

5. Proving System Invariants

6. Conclusion and Future Work
Timed I/O Automaton (TIOA)

A mathematical framework to model and analyze timed systems. [Kaynar, Lynch, Segala and Vaandrager 2003]

- special class of HIOA without continuous interaction
- Input/Output distinction (Input always enabled)
- precondition-effect style
- More expressive than Timed Automata [Alur and Dill 94]
  - no restriction on dynamic type of variables
  - parametrized transition
  - less expressive because of I/O distinction
Timed Automaton

Formally, it is a tuple $A = (X, Q, \Theta, E, H, D, T)$

$X$: internal variables

$Q$: states, a set of valuations of $X$

$\Theta$: start states a non-empty subset of $Q$

$E, H$: external and internal actions, $A \triangleq E \cup H$

$D \subseteq Q \times A \times Q$: discrete transitions

$T$: a set of trajectories for $X$ such that $\tau(t) \in Q$

for all $t$ in $\text{domain}(\tau)$. 

Timed Automaton with Deadlines (TAD)

TAD [Bornot and Sifakis 1998] is TA with a deadline predicate $D : \mathcal{E} \rightarrow \Psi$ associated with each edge $e \in \mathcal{E}$ that specifies when $e$ becomes urgent.

- time can progress at a location as long as all the deadline conditions associated with the outgoing edges are false.
- requires: deadline implies precondition
- several parallel composition operations
  - **Stiff**: for strongly coupled components / synchronous
  - **Flexible**: for loosely coupled components / asynchronous
  - **Priority**: priority based composition
  - preserves time reactivity.
Adding Urgency

- \( \mathcal{A} = (X, Q, \Theta, E, H, D, T) \)
- \( A \triangleq E \cup H \) – external and internal actions
- \( U : Q \times A \rightarrow \{ \text{true, false} \} \) – Urgency predicate
- \( \text{TAU} = (\mathcal{A}, U) \)

\[ a \text{ is enabled in } x \iff \exists x' : (x, a, x') \in D \]
\[ a \text{ is urgent in } x \iff a \text{ enabled in } x \land U(x, a) = \text{true} \]

Unlike TAD, we does not assume \( U(x, a) \Rightarrow \text{pre} \)
Timed Automata Example – Train

**Automaton** Train

**states**
- discrete control ∈ \{start, light, gate\} initially start
- clock t ∈ R initially 0

**signature**
- external coming, approaching, passing

**transitions**
- external coming
  - pre t > 2 ∧ control = start
  - urgent when t ≥ 5
  - eff control := light, t := 0

- external approaching
  - pre t > 5 ∧ control = light
  - urgent when t ≥ 10
  - eff control := gate, t := 0

- external passing
  - pre t = 2 ∧ control = gate
  - urgent when true
  - eff control := start, t := 0
Timed Automata Example – *Train*

- **Start** ($t \leq 5$)
- **Coming** ($t > 2$)
- **Gate** ($t \leq 2$)
- **Light** ($t \leq 10$)
- **Passing** ($t = 2$)
- **Approaching** ($t \geq 5$)
Example - parametrized transitions in TIOA

**Automaton** $\text{TimedChannel}(b, M)$

**states**
- discrete *queue*, a finite sequence of elements from $M \times R$
- initially empty
- analog *now* $\in R$ initially 0

**signature**
- external $\text{send}(m)$, $\text{receive}(m)$

**transitions**
- external $\text{send}(m)$
  - eff $\text{add}(m, now + b)$ to the end of the *queue*
- external $\text{receive}(m)$
  - pre $\exists u. (m, u)$ is first element of *queue*
  - urgent when $(\text{now} \geq u)$
  - eff remove first element of *queue*
Semantics

Axioms on trajectories

**T1**  *Existence of point trajectories*

**T2**  *Prefix closure*

**T3**  *Suffix closure*

**T4**  (*urgency*) For every $\tau \in T$, $t \in \text{dom}(\tau)$ and $a \in A$: if $a$ is urgent in $\tau(t)$ then $t = \tau.ltime$.

**T5**  (*Maximality*): For every $\tau \in T$, if $\tau$ is maximal and finite then $\tau$ is right-closed and some $a \in A$ is urgent in $\tau.lval$
Counter Example for T5

automaton $\mathcal{A}$
states $b : \text{Bool}$ initially false
clock $x$ initially 0
signature external $a$
transitions external $a$
  pre $x > 4 \land b = \text{false}$
  urgent when true
  eff $b := \text{true}$

- time stops at $x = 4$
- no transition is enabled at $x = 4$
- Axiom T5 does not hold
Proving T5

For each transition definition \( tr \), let

\[
\begin{align*}
b(\vec{h}) \\
\text{pre} & \quad \text{pre}(\vec{v}, \vec{h}) \\
\text{urgent when} & \quad \text{urg}(\vec{v}, \vec{h}) \\
\text{eff} & \quad \text{eff}(\vec{v}, \vec{h}, \vec{v}')
\end{align*}
\]

\[
\text{Urg}(tr)(\vec{v}, \vec{h}) \triangleq \exists \vec{h} : \text{pre}(\vec{v}, \vec{h}) \land \text{urg}(\vec{v}, \vec{h})
\]

**Theorem 1** If the predicate \( \bigvee_{tr} \text{Urg}(tr) \) is left-closed then axiom T5 holds.
Counter Example for T5 … continues

automaton $A$

<table>
<thead>
<tr>
<th>states</th>
<th>$b : \text{Bool}$ initially false</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock $x$</td>
<td>initially 0</td>
</tr>
</tbody>
</table>

signature

<table>
<thead>
<tr>
<th>transitions</th>
<th>external $a_1, a_2$</th>
<th>external $a_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>pre</td>
<td>$x &gt; 4 \land b = \text{false}$</td>
<td>$x = 4 \land b = \text{false}$</td>
</tr>
<tr>
<td>urgent when</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>eff</td>
<td>$b := \text{true}$</td>
<td>$b := \text{true}$</td>
</tr>
</tbody>
</table>

- time stops at $x = 4$ and $a_2$ is enabled.
- Axiom T5 holds
Adding I/O distinction

- Timed I/O Automaton is a Timed Automaton
- the external action $E$ partitioned into input($I$) and output($O$).
- The following axioms are satisfied
  - **E0** Input not urgent
  - **E1** Input action always enabled
  - **E2** Time-passage enabling. For every state $x \in Q$, there exists a trajectory $\tau \in T$ such that $\tau.fstate = x$ and either
    - Time can grow to infinity ($\tau.ltime = \infty$), or
    - A locally controlled action is enabled at the end of the trajectory ($\tau.lstate$)
Time Reactivity

E0 and E1 hold if, at the syntactic level, input actions have

- precondition true, and
- urgency predicate false

**Theorem 2** *(E2 happens for free):*

*Each timed I/O automaton satisfies axiom E2.*

**Proof:** use T5
Composition

**Definition 1** We say that timed automata $A_1$ and $A_2$ are compatible if $H_1 \cap A_2 = H_2 \cap A_1 = \emptyset$ and $X_1 \cap X_2 = \emptyset$.

Composition $A_1 || A_2$

- **Require:** $A_1$ and $A_2$ are compatible

- **enabled:** if $a \in A_i$ then $x[X_i \xrightarrow{a} x'] [X_i$

- **urgent:** $U((x_1, x_2), a) = U_1(x_1, a) \lor U_2(x_2, a)$

**Theorem 3** If $A_1$ and $A_2$ are compatible timed I/O automata then $A_1 || A_2$ is a timed I/O automaton.

In general (for TA with urgency or TAD) this theorem does not hold.
Urgency v. Deadline Predicate [Sifaki et al.]

\[
\begin{array}{ll}
\text{external } a(\bar{h}) & \text{external } a(\bar{h}) \\
\text{pre } pre(\bar{v}, \bar{h}) & \text{pre } pre(\bar{v}, \bar{h}) \\
\text{urgent when } urg(\bar{v}, \bar{h}) & \text{deadline } pre(\bar{v}, \bar{h}) \land urg(\bar{v}, \bar{h}) \\
\text{eff } eff(\bar{v}, \bar{h}, \bar{v}') & \text{eff } eff(\bar{v}, \bar{h}, \bar{v}')
\end{array}
\]

- \( pre(\bar{v}, \bar{h}) \land urg(\bar{v}, \bar{h}) \Rightarrow pre(\bar{v}, \bar{h}) \)

- Urgency predicates are slightly shorter than deadline predicates

- Deadline predicate is urgency predicate

- replace \( urg(\bar{v}, \bar{h}) \) by \( pre(\bar{v}, \bar{h}) \land urg(\bar{v}, \bar{h}) \) then it is a deadline predicate.
Urgency v. Stoping Conditions [Kaynar et al.]

- Urgency predicates are shorter and more natural than deadline predicates.
- Stoping condition does not preserve time reactivity. Can define automaton with time deadlock.
- stop when = disjunction of all $\text{pre}(\bar{v}, \bar{h}) \land \text{urg}(\bar{v}, \bar{h})$ for all transitions.
- urgent when = stop when if E2 holds, and simplify formula
Urgency v. Location Invariant (as in UPPAAL)

\[
\text{external } a(\vec{h}) \\
\text{pre } \pre(\vec{v}, \vec{h}) \\
\text{urgent when } \text{urg}(\vec{v}, \vec{h}) \\
\text{eff } \text{eff}(\vec{v}, \vec{h}, \vec{v}') \\
\text{invariant } \bigvee (\pre(\vec{v}, \vec{h}) \land \text{urg}(\vec{v}, \vec{h}))
\]

- Location invariant does not preserve time reactivity.

\[\text{inv: } x \leq 1\]
Urgency v. Location Invariant . . . continues

- Location invariants are more expressive – can specify strict upper bound.

\[
\begin{align*}
\text{states} & \quad b : \text{Bool} \quad \text{initially} \quad \text{false} \\
\text{t} : \text{R} & \quad \text{initially} \quad 0 \leq t < 1
\end{align*}
\]

\[
\begin{align*}
\text{transitions} & \quad \text{external} \quad a \\
\text{pre} & \quad b = \text{false} \\
\text{urgent when} & \quad x = t \\
\text{eff} & \quad : = \text{true}
\end{align*}
\]

\[
\begin{array}{c}
\text{false} \\
\text{inv: } x < 1
\end{array}
\quad a
\quad
\begin{array}{c}
\text{true}
\end{array}
\]

But strict upper bound are not used in applications. At least we

\[
\begin{align*}
\text{false} \\
\text{inv: } x < 1
\end{align*}
\]
Urgency v. Location Invariant ... continues

Location invariants are more expressive – can specify strict upper bound.

- Location invariants are not used in applications. At least we are not aware of any such application.

\[
\begin{align*}
\text{false} & \quad \text{inv: } x < 1 \\
\text{true} & \quad \text{pre } b = \text{false} \\
\end{align*}
\]

\[a \rightarrow \text{transitions}
\]

\[
\begin{align*}
\text{true} & \quad \text{urgent when } x = t \\
\text{eff } b := \text{true} & \\
\end{align*}
\]

\[
\begin{align*}
\text{states} & \quad b : \text{Bool initially false} \\
\text{initially } 0 \leq t < 1 & \\
\end{align*}
\]

\[
\begin{align*}
\text{external } a & \\
\end{align*}
\]
Urgency v. Location Invariant . . . continues

- Location Invariants are more expressive – can specify strict upper bound.

```
false
inv: x < 1
```

```
true
```

- But strict upper bound are not used in applications. At least we are not aware of any such application.

```
states
b : Bool initially false
t : R initially 0 ≤ t < 1

transitions
external a
pre b = false
urgent when x = t
eff b := true
```
Translating urgency predicates to invariant

**Definition 2** (Stable predicates:) Predicate involves only lowerbound on clocks.

\[
\varphi(\vec{v}) \implies \forall d > 0 : \varphi(\vec{v} \oplus d)
\]

If a TIOA predicate \(\text{Urg}(tr)\) is **stable** and **left closed**

\[
\text{inv} = \neg \left( \bigvee_{tr} \text{Urg}(tr) \right) \lor LH \left( \bigvee_{tr} \text{Urg}(tr) \right),
\]

where \(LH(\varphi) \overset{\Delta}{=} \{ x | \varphi(x) \land \exists \epsilon > 0 \forall 0 < \epsilon' \leq \epsilon : \neg \varphi(x \ominus \epsilon') \}\) is the **lower hull** of the state predicate \(\varphi\).

Provided that.

- \(\text{inv}\) holds initially and
- after a discrete transition \(\text{pre}(\vec{v}, \vec{h}) \land \text{eff}(\vec{v}, \vec{h}, \vec{v}') \implies \text{inv}(\vec{v}')\).
Urgency leads to non-convex zones?

- **urgent when** $x \geq 10 \lor y \geq 5$ is non convex
- the IF tool uses zone splitting
- why it is not happening in UPPAAL
  - rather strong restriction in the syntax of invarinats: only conjunction of upper bound on clocks

\[
\text{inv} = \bigwedge_{i} x_i \leq l_i
\]

- adopt similar restrictions on urgency

\[
\text{urg} = \bigvee_{i} x_i \geq l_i \land \text{(apply I/O distiction)}
\]
only output actions can be urgent.

\{S1, S2, S3\}

\begin{itemize}
  \item If \( a \) is urgent in UPPAAL then location \([S3, T3]\) have a non-convex zone \((x \geq 10^y = 5) \land (y \geq 5^x = 10)\).
\end{itemize}

\[ \text{Urgency and UPPAAL} \]

\[ \begin{align*}
  \text{states} & \quad control \in \{T1, T2, T3\} \\
  \text{clock} & \quad y \\
  \text{output} & \quad a \\
  \text{pre} & \quad control = T2 \land y \geq 5 \\
  \text{urgent when} & \quad true \\
  \text{eff} & \quad control := T3
\end{align*} \]
Urgency and UPPAAL

- If \( a \) is urgent channel in UPPAAL then location \([S3, T3]\) have a non-convex zone \((x \geq 10 \land y = 5) \lor (y \geq 5 \land x = 10)\).

- Using urgency: only output actions can be urgent.

\begin{align*}
\text{states} & \quad \text{control} \in \{S1, S2, S3\} \\
\text{clock} & \quad x \\
\text{input} & \quad a \\
\text{pre} & \quad \text{true} \\
\text{urgent when} & \quad \text{false} \\
\text{eff if}(\text{control} = S2 \land x \geq 10) & \quad \text{control} := S3
\end{align*}

\begin{align*}
\text{states} & \quad \text{control} \in \{T1, T2, T3\} \\
\text{clock} & \quad y \\
\text{output} & \quad a \\
\text{pre} & \quad \text{control} = T2 \land y \geq 5 \\
\text{urgent when} & \quad \text{true} \\
\text{eff} & \quad \text{control} := T3
\end{align*}

composition?
Proving System Invariants

\( \varphi \) is an invariant if it holds for all reachable states.

- Proving steps
  - Holds initially
  - preserved by all discrete transitions
  - preserved by time progress function \((time(d))\)

\[
time(d) \\
\text{pre} \bigwedge_{tr} tp(s, tr, d) \\
\text{eff } s := s \oplus d
\]

- \( tp \) (time progress function) of a transition \( tr \) allows time to progress as long as both precondition and urgency does not hold

\[
\forall 0 \leq e < d : \forall h : \neg (\text{pre}(s \oplus e, h) \land \text{urg}(s \oplus e, h))
\]
Quantifier Elimination in Proving Invariant

- Stable under time progress $\varphi(s) \Rightarrow \forall t > 0 : \varphi(s \oplus t)$
  - urgency and precondition involving only lower bound
- Make lower bounds strict

$$\hat{\varphi}(s) \equiv \exists s' \exists t > 0 : \varphi(s') \land (s = s' \oplus t)$$

**Lemma 1** State predicate $\varphi = \text{stable under time progress}$. Then

$$(\forall e : 0 \leq e < d : \neg \varphi(s \oplus e)) \iff \neg \hat{\varphi}(s \oplus d)$$

**Example 1** $\forall 0 \leq e < d :$

- $\neg(t + e > 2 \land control = start \land t + e \geq 5)$
- $\implies \neg(control = start \land t + d > 5)$
Proving Invariant - Example

Example Prove that \( control = start \Rightarrow t < 5 \) is an invariant.

Proof: (1) holds initially \( (t = 0) \)
(2) for action \( \text{coming} \)
\( \forall 0 \leq e < d : \neg(t + e > 2 \land control = start \land t + e \geq 5) \)
equivalent to \( control = start \Rightarrow t + d < 5 \)
(3) similarly \( control = light \Rightarrow t + d < 10 \quad \) (approaching)
\( control = gate \Rightarrow t + d \neq 2 \quad \) (passing)
(4) for \( \text{time}(d) \)

- precondition \( control = start \Rightarrow t + d < 5 \land \ldots \)
- effect \( control' = control \land t' := t \oplus d \)
- \( control' = start \Rightarrow control = start \)
- \( t + d < 5 \Rightarrow t' < 5 \)
Related Work

- TIOA and deadlines
  - I/O automata with upper and lower bounds associated with tasks [Merritt et al. CONCUR’91].

- TAD used in
  - Modeling timeouts without timelocks [Bowman ARTS’99].
  - Real-time Profile for UML [S. Graf and I. Ober 2003].

- tool: IF - TAD based validation tool for timed systems
Conclusion and Future work

- Improved TIOA
  - shorter and natural spec
  - ensures time reactivity by construction
  - easier to prove invariant properties

- A proposal to add urgency to UPPAAL

- Future work
  - Urgency predicates for HIOA
  - proof rules for simulation and liveness proofs with urgency predicate
  - comparing Urgency predicates and location invariant in implementation and expressively