Recent results on Timed Systems

Time Petri Nets and Timed Automata

Béatrice Bérard

LAMSADE

Université Paris-Dauphine & CNRS

berard@lamsade.dauphine.fr

Based on joint work with F. Cassez, S. Haddad, D. Lime, O.H. Roux

VECoS'08, July 2nd 2008

General context

System

Communication protocol
Automated System...

Formal specification,
Algorithm, source code...

Properties

Reachability
Response time...

General context

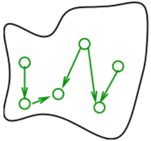
System

Communication protocol
Automated System...

Formal specification,
Algorithm, source code...

Modelling
expressiveness, concision

Models

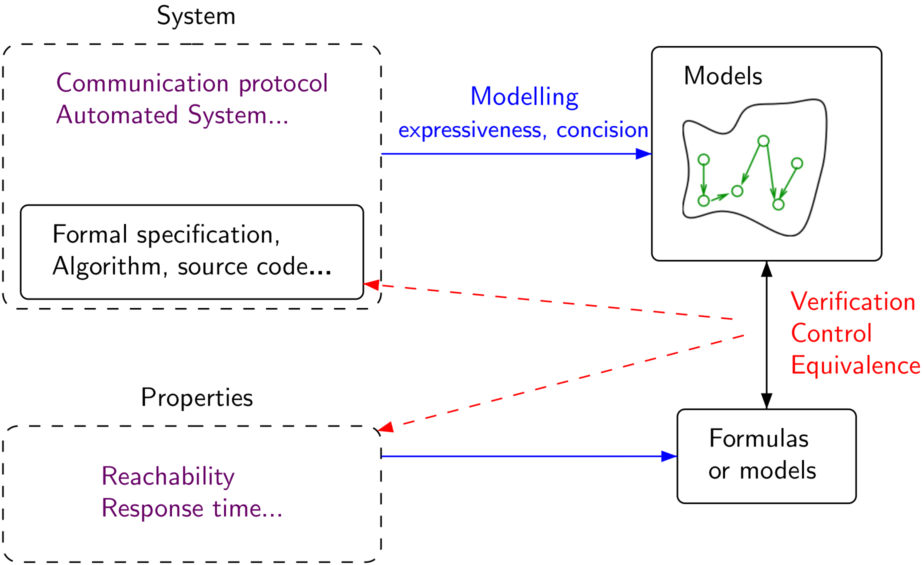


Properties

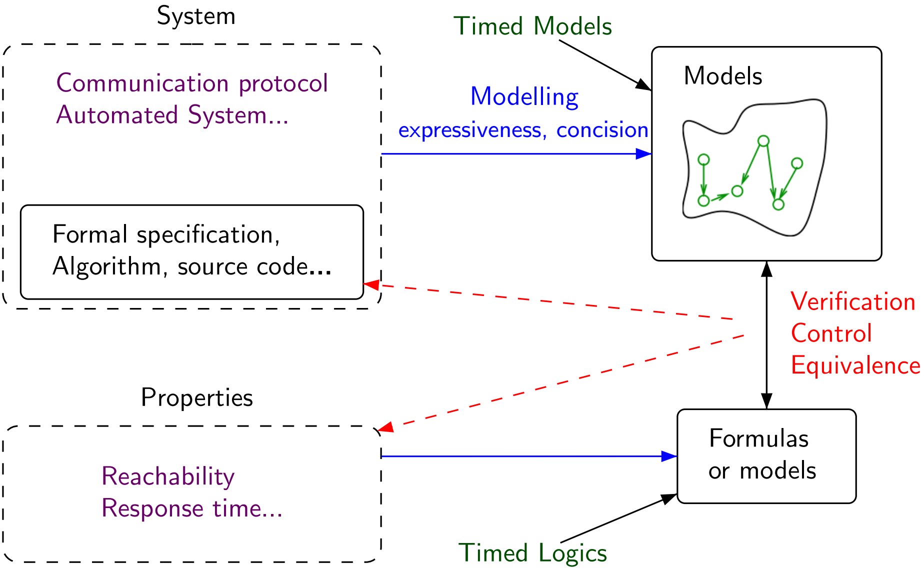
Reachability
Response time...

Formulas
or models

General context



General context

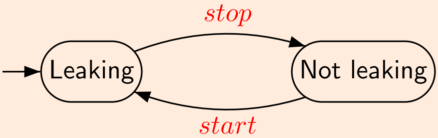


Why add time ?

The gas burner example [ACHH93]

The gas burner may leak and :

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

Timed features are needed in the model and in the properties:

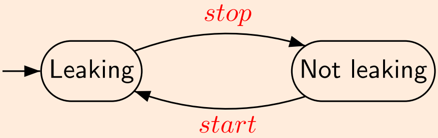
Instead of observing a sequence of events $a_1 \ a_2 \ldots$
observe a sequence of alternating events and delays: $a_1 \ d_1 \ a_2 \ d_2 \ldots$

Why add time ?

The gas burner example [ACHH93]

The gas burner may leak and :

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

Timed features are needed in the model and in the properties:

Instead of observing a sequence of events $a_1\ a_2 \ldots$
observe a sequence of alternating events and delays: $a_1\ d_1\ a_2\ d_2 \ldots$

Outline

Timed Models

Comparing timed automata and time Petri nets

Conclusion

Outline

Timed Models

Comparing timed automata and time Petri nets

Conclusion

Timed models and their semantics

A Timed Model

is obtained from a classical one by introducing delay transitions, with a dense or discrete time:

- either by adding clocks
- or (a particular case) by associating firing intervals with transitions.

Semantics: a Timed Transition System

Act alphabet of actions,

$\mathcal{T} = (S, s_0, E)$ transition system

- S set of configurations, s_0 initial configuration,
- $E \subseteq S \times Act \times S$ contains

  action transitions: $s \xrightarrow{a} s'$, instantaneous execution of a

  delay transitions: $s \xrightarrow{d} s'$, time elapsing for d time units.

Timed models and their semantics

A Timed Model

is obtained from a classical one by introducing delay transitions, with a dense or discrete time:

- either by adding clocks
- or (a particular case) by associating firing intervals with transitions.

Semantics: a Timed Transition System

Act alphabet of actions, $\mathbb{T}$ time domain contained in $\mathbb{R}_{\geq 0}$,

$\mathcal{T} = (S, s_0, E)$ timed transition system

- S set of configurations, s_0 initial configuration,
- $E \subseteq S \times (Act \cup \mathbb{T}) \times S$ contains

  action transitions: $\boldsymbol{s} \xrightarrow{\boldsymbol{a}} \boldsymbol{s'}$, instantaneous execution of $\boldsymbol{a}$

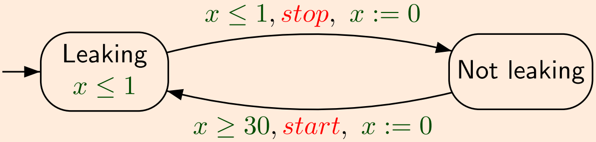
  delay transitions: $\boldsymbol{s} \xrightarrow{\boldsymbol{d}} \boldsymbol{s'}$, time elapsing for $\boldsymbol{d}$ time units.

Adding clocks: timed automata

a variation of [Alur Dill 1990]

The gas burner as a timed automaton

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



x is a real valued clock, **invariant** $x \leq 1$ is associated with state Leaking, $x \geq 30$ and $x \leq 1$ are **guards** and $x := 0$ is a **reset**.

Configuration: (q, v) where $q \in \{\mathsf{L}, \mathsf{NL}\}$ and v a value of clock x.
An execution:
$(\mathbf{L}, [0]) \xrightarrow{0.3} (\mathbf{L}, [0.3]) \xrightarrow{stop} (\mathbf{NL}, [0]) \xrightarrow{35} (\mathbf{NL}, [35]) \xrightarrow{start} (\mathbf{L}, [0]) \cdots$

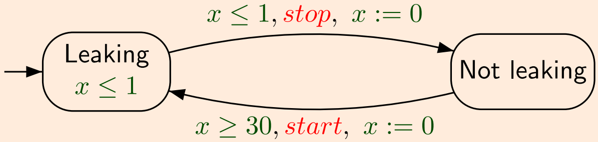
Not expressive enough for the property: Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

Adding clocks: timed automata

a variation of [Alur Dill 1990]

The gas burner as a timed automaton

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



x is a real valued clock, **invariant** $x \leq 1$ is associated with state Leaking, $x \geq 30$ and $x \leq 1$ are **guards** and $x := 0$ is a **reset**.

Configuration: (q, v) where $q \in \{\mathsf{L}, \mathsf{NL}\}$ and v a value of clock x.
An execution:
$(\mathbf{L}, [0]) \xrightarrow{0.3} (\mathbf{L}, [0.3]) \xrightarrow{stop} (\mathbf{NL}, [0]) \xrightarrow{35} (\mathbf{NL}, [35]) \xrightarrow{start} (\mathbf{L}, [0]) \cdots$

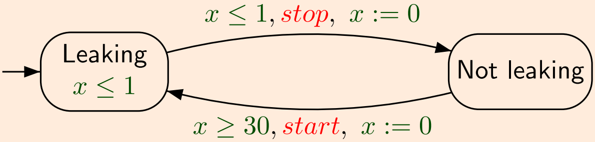
Not expressive enough for the property: Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

Adding clocks: timed automata

a variation of [Alur Dill 1990]

The gas burner as a timed automaton

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



x is a real valued clock, **invariant** $x \leq 1$ is associated with state Leaking, $x \geq 30$ and $x \leq 1$ are **guards** and $x := 0$ is a **reset**.

Configuration: (q, v) where $q \in \{\mathsf{L}, \mathsf{NL}\}$ and v a value of clock x.
An execution:
$(\mathbf{L}, [0]) \xrightarrow{\mathbf{0.3}} (\mathbf{L}, [\mathbf{0.3}]) \xrightarrow{stop} (\mathbf{NL}, [0]) \xrightarrow{35} (\mathbf{NL}, [35]) \xrightarrow{start} (\mathbf{L}, [0]) \cdots$

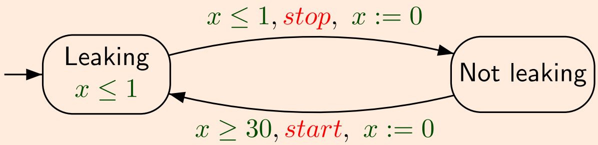
Not expressive enough for the property: Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

Adding clocks: timed automata

a variation of [Alur Dill 1990]

The gas burner as a timed automaton

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



x is a real valued clock, **invariant** $x \le 1$ is associated with state Leaking, $x \ge 30$ and $x \le 1$ are **guards** and $x := 0$ is a **reset**.

Configuration: (q, v) where $q \in \{\mathsf{L}, \mathsf{NL}\}$ and v a value of clock x.
An execution:

$$(\mathbf{L}, [0]) \xrightarrow{0.3} (\mathbf{L}, [0.3]) \xrightarrow{stop} (\mathbf{NL}, [0]) \xrightarrow{35} (\mathbf{NL}, [35]) \xrightarrow{start} (\mathbf{L}, [0]) \cdots$$

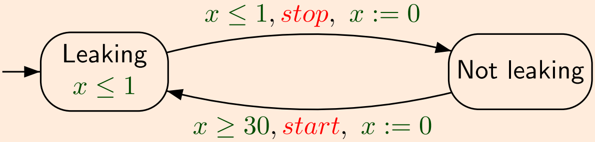
Not expressive enough for the property: Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

Adding clocks: timed automata

a variation of [Alur Dill 1990]

The gas burner as a timed automaton

- each time leaking is detected, it is repaired or stopped in less than 1s
- two leaking periods are separated by at least 30s



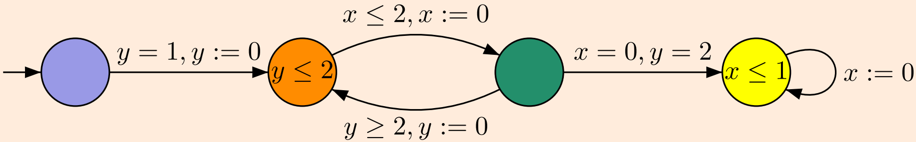
x is a real valued clock, **invariant** $x \leq 1$ is associated with state Leaking, $x \geq 30$ and $x \leq 1$ are **guards** and $x := 0$ is a **reset**.

Configuration: (q, v) where $q \in \{\mathsf{L}, \mathsf{NL}\}$ and v a value of clock x.
An execution:
$(\mathbf{L}, [0]) \xrightarrow{\mathbf{0.3}} (\mathbf{L}, [\mathbf{0.3}]) \xrightarrow{\mathbf{\mathit{stop}}} (\mathbf{NL}, [0]) \xrightarrow{\mathbf{35}} (\mathbf{NL}, [\mathbf{35}]) \xrightarrow{\mathbf{\mathit{start}}} (\mathbf{L}, [0]) \cdots$

Not expressive enough for the property: Is it possible that the gas burner leaks during a time greater than $\frac{1}{20}$ of the global time after the first 60s?

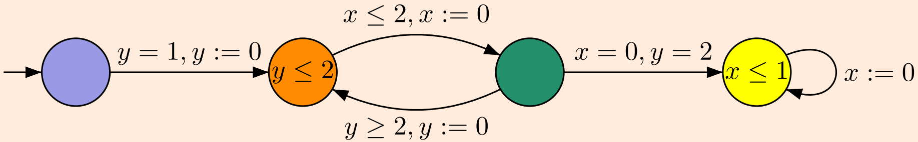
Semantics of timed automata

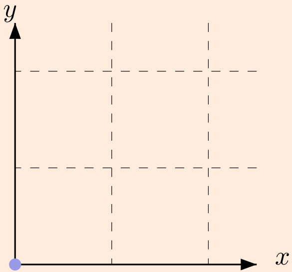
A geometric view with two clocks x et y



Semantics of timed automata

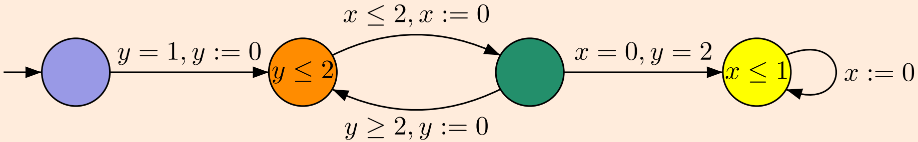
A geometric view with two clocks x et y

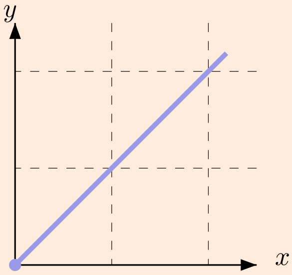




Semantics of timed automata

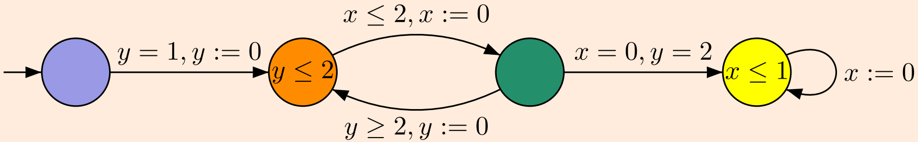
A geometric view with two clocks x et y

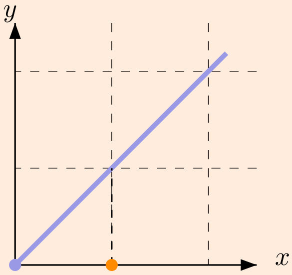




Semantics of timed automata

A geometric view with two clocks x et y

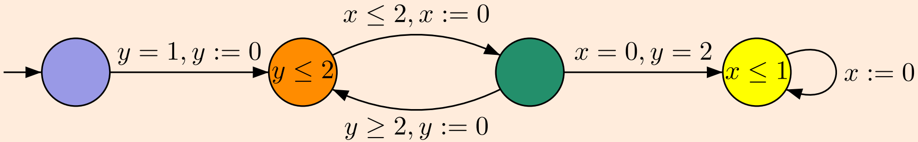


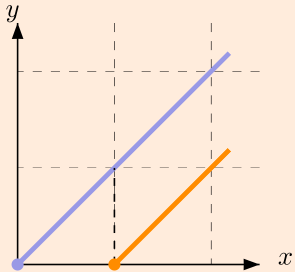


$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix}$$

Semantics of timed automata

A geometric view with two clocks x et y

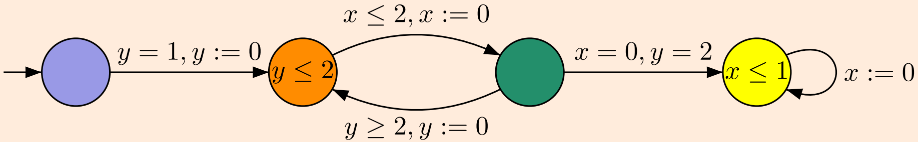


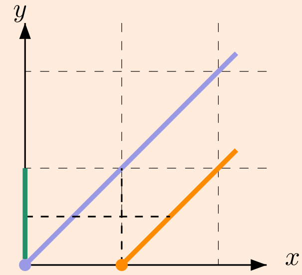


$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix}$$

Semantics of timed automata

A geometric view with two clocks x et y

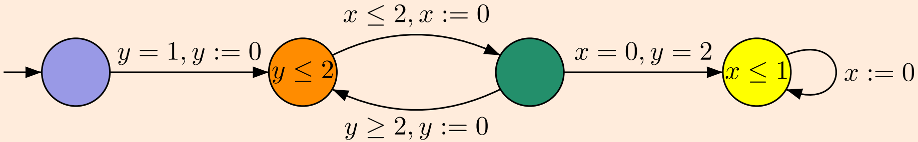


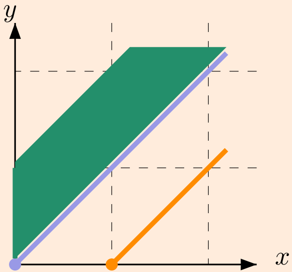


$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix} \xrightarrow{\boldsymbol{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix} \cdots$$

Semantics of timed automata

A geometric view with two clocks x et y

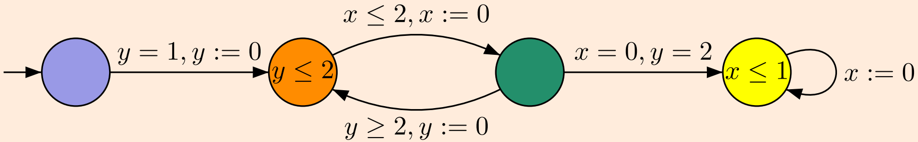


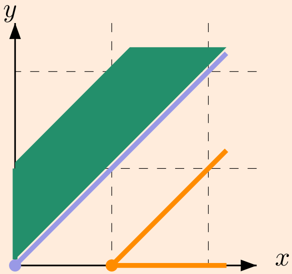


$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix} \xrightarrow{\boldsymbol{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix} \cdots$$

Semantics of timed automata

A geometric view with two clocks x et y

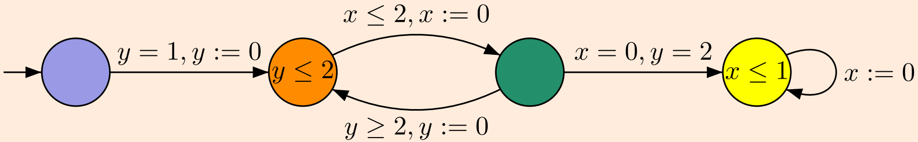


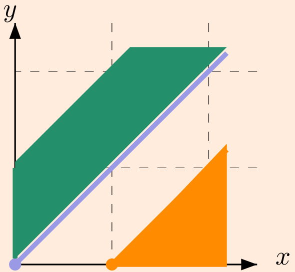


$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix} \xrightarrow{\boldsymbol{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix} \ldots$$

Semantics of timed automata

A geometric view with two clocks x et y

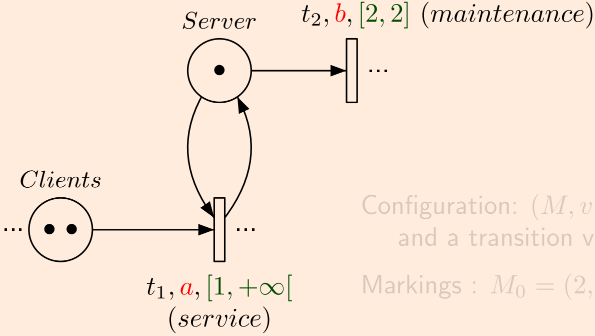




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix} \xrightarrow{\boldsymbol{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix} \cdots$$

Adding time intervals on transitions

Example : Time Petri Nets [Merlin 1974]



Configuration: (M, v) for a marking M
and a transition valuation v

Markings : $M_0 = (2,1)$, $M_1 = (1,1)$, $M_2 = (1,0)$

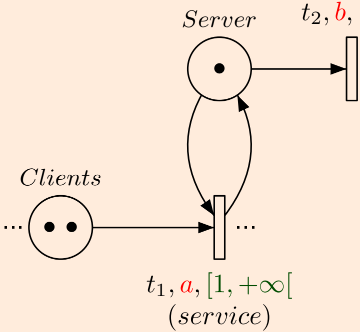
Valuation of transition t: time elapsed since t was last enabled, $\perp$ if t is not enabled.
Classical semantics: when a firing occurs, an enabled transition is **newly enabled** if it was disabled after the token consumption or if it is the transition fired.

An execution:

$$(M_0, [0,0]) \xrightarrow{1.3} (M_0, [1.3, 1.3]) \xrightarrow{a} (M_1, [0,0]) \xrightarrow{2} (M_1, [2,2]) \xrightarrow{b} (M_2, [\perp, \perp])$$

Adding time intervals on transitions

Example : Time Petri Nets [Merlin 1974]



Configuration: (M, v) for a marking M
and a transition valuation v

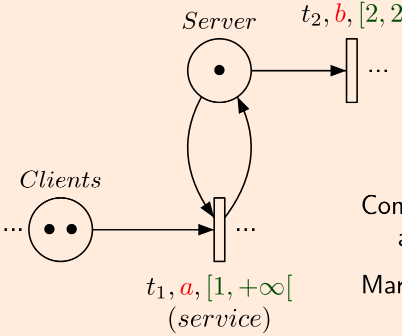
Markings : $M_0 = (2,1)$, $M_1 = (1,1)$, $M_2 = (1,0)$

Valuation of transition t: time elapsed since t was last enabled, $\perp$ if t is not enabled.
Classical semantics: when a firing occurs, an enabled transition is **newly enabled** if it was disabled after the token consumption or if it is the transition fired.

An execution:
$(M_0, [0,0]) \xrightarrow{1.3} (M_0, [1.3, 1.3]) \xrightarrow{a} (M_1, [0,0]) \xrightarrow{2} (M_1, [2,2]) \xrightarrow{b} (M_2, [\perp, \perp])$

Adding time intervals on transitions

Example : Time Petri Nets [Merlin 1974]



Configuration: (M, v) for a marking M
and a transition valuation v

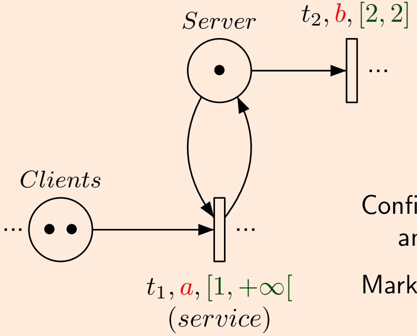
Markings : $M_0 = (2,1)$, $M_1 = (1,1)$, $M_2 = (1,0)$

Valuation of transition t: time elapsed since t was last enabled, $\perp$ if t is not enabled.
Classical semantics: when a firing occurs, an enabled transition is **newly enabled** if it was disabled after the token consumption or if it is the transition fired.

An execution:
$(M_0, [0,0]) \xrightarrow{1.3} (M_0, [1.3, 1.3]) \xrightarrow{a} (M_1, [0,0]) \xrightarrow{2} (M_1, [2,2]) \xrightarrow{b} (M_2, [\perp, \perp])$

Adding time intervals on transitions

Example : Time Petri Nets [Merlin 1974]



Configuration: (M, v) for a marking M
and a transition valuation v

Markings : $M_0 = (2,1)$, $M_1 = (1,1)$, $M_2 = (1,0)$

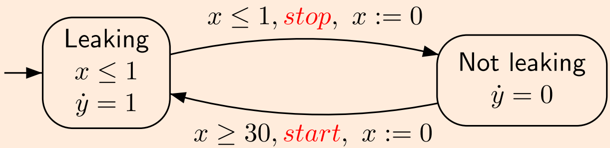
Valuation of transition t: time elapsed since t was last enabled, $\perp$ if t is not enabled.
Classical semantics: when a firing occurs, an enabled transition is **newly enabled** if it was disabled after the token consumption or if it is the transition fired.

An execution:
$$(M_0, [0,0]) \xrightarrow{1.3} (M_0, [1.3, 1.3]) \xrightarrow{a} (M_1, [0,0]) \xrightarrow{2} (M_1, [2,2]) \xrightarrow{b} (M_2, [\perp, \perp])$$

Other timed models and timed logics

The gas burner

as a linear hybrid automaton



Add a stopwatch y and a clock z which are never reset

and use these variables in a CTL-like formula:

$$\mathrm{AG}(z \geq 60 \Rightarrow 20y \leq z)$$

the gas burner always leaks during a time less than or equal to $\frac{1}{20}$ of the global time after the first 60s.

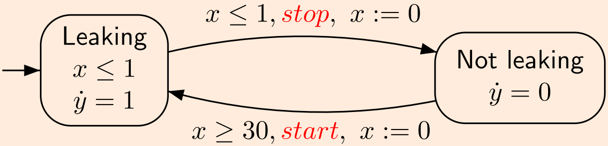
Timed temporal logics

have been defined to extend Linear Temporal Logic LTL and Computational Tree Logic CTL.

Other timed models and timed logics

The gas burner

as a linear hybrid automaton



Add a stopwatch y and a clock z which are never reset

and use these variables in a CTL-like formula:

$$\mathsf{AG}(z \geq 60 \Rightarrow 20y \leq z)$$

the gas burner always leaks during a time less than or equal to $\frac{1}{20}$ of the global time after the first 60s.

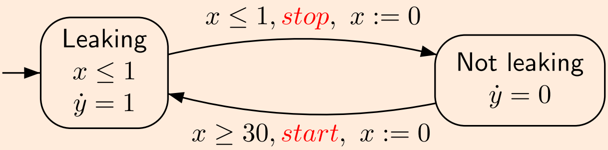
Timed temporal logics

have been defined to extend Linear Temporal Logic LTL and Computational Tree Logic CTL.

Other timed models and timed logics

The gas burner

as a linear hybrid automaton



Add a stopwatch y and a clock z which are never reset

and use these variables in a CTL-like formula:

$$\mathsf{AG}(z \geq 60 \Rightarrow 20y \leq z)$$

the gas burner always leaks during a time less than or equal to $\frac{1}{20}$ of the global time after the first 60s.

Timed temporal logics

have been defined to extend Linear Temporal Logic LTL and Computational Tree Logic CTL.

Verification

is often not possible

Reachability of a control state is undecidable for linear hybrid automata [Alur et al. 1995].

but can sometimes be done

Reachability of a control state for timed automata is PSPACE-complete [Alur, Dill 1990].

Several tools

have been developed and applied to case studies, in spite of the high complexity:

- Kronos and UppAal for timed automata
- HCMC and HyTech for linear hybrid automata (semi-algorithms)
- TSMV for automata with duration (discrete time)
- Romeo and TINA, for time Petri nets
- ...

Verification

is often not possible

Reachability of a control state is undecidable for linear hybrid automata [Alur et al. 1995].

but can sometimes be done

Reachability of a control state for timed automata is PSPACE-complete [Alur, Dill 1990].

Several tools

have been developed and applied to case studies, in spite of the high complexity:

- KRONOS and UPPAAL for timed automata
- HCMC and HYTECH for linear hybrid automata (semi-algorithms)
- TSMV for automata with duration (discrete time)
- Romeo and TINA, for time Petri nets
- ...

Verification

is often not possible

Reachability of a control state is undecidable for linear hybrid automata [Alur et al. 1995].

but can sometimes be done

Reachability of a control state for timed automata is PSPACE-complete [Alur, Dill 1990].

Several tools

have been developed and applied to case studies, in spite of the high complexity:

- KRONOS and UPPAAL for timed automata
- HCMC and HYTECH for linear hybrid automata (semi-algorithms)
- TSMV for automata with duration (discrete time)
- Romeo and TINA, for time Petri nets
- ...

Outline

Which equivalence relation ?

between two timed transition systems $\mathcal{T}_1 = (S_1, s_1^0, E_1)$ and $\mathcal{T}_2 = (S_2, s_2^0, E_2)$

Language equivalence

$\mathcal{T}_1$ and $\mathcal{T}_2$ are language-equivalent if they accept the same sets of timed observation sequences (with respect to accepting conditions).

Weak timed bisimulation

With an alphabet Act containing an internal action ε, the systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are weaky timed bisimilar if there is an equivalence relation on $S_1 \times S_2$ such that s_1^0 and s_2^0 are equivalent and:

Which equivalence relation ?

between two timed transition systems $\mathcal{T}_1 = (S_1, s_1^0, E_1)$ and $\mathcal{T}_2 = (S_2, s_2^0, E_2)$

Language equivalence

$\mathcal{T}_1$ and $\mathcal{T}_2$ are language-equivalent if they accept the same sets of timed observation sequences (with respect to accepting conditions).

Weak timed bisimulation

With an alphabet Act containing an internal action ε, the systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are weaky timed bisimilar if there is an equivalence relation on $S_1 \times S_2$ such that s_1^0 and s_2^0 are equivalent and:

if
$$\begin{tikzcd}
s_1 \arrow[r, "a"] \arrow[d, phantom, "\sim"] & s_1' \\
s_2 &
\end{tikzcd}$$

Which equivalence relation ?

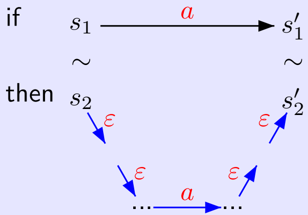
between two timed transition systems $\mathcal{T}_1 = (S_1, s_1^0, E_1)$ and $\mathcal{T}_2 = (S_2, s_2^0, E_2)$

Language equivalence

$\mathcal{T}_1$ and $\mathcal{T}_2$ are language-equivalent if they accept the same sets of timed observation sequences (with respect to accepting conditions).

Weak timed bisimulation

With an alphabet Act containing an internal action ε, the systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are weaky timed bisimilar if there is an equivalence relation on $S_1 \times S_2$ such that s_1^0 and s_2^0 are equivalent and:



Which equivalence relation ?

between two timed transition systems $\mathcal{T}_1 = (S_1, s_1^0, E_1)$ and $\mathcal{T}_2 = (S_2, s_2^0, E_2)$

Language equivalence

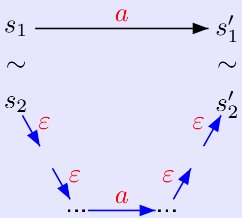
$\mathcal{T}_1$ and $\mathcal{T}_2$ are language-equivalent if they accept the same sets of timed observation sequences (with respect to accepting conditions).

Weak timed bisimulation

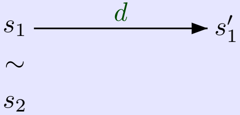
With an alphabet Act containing an internal action ε, the systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are weaky timed bisimilar if there is an equivalence relation on $S_1 \times S_2$ such that s_1^0 and s_2^0 are equivalent and:

if

then



if



Which equivalence relation ?

between two timed transition systems $\mathcal{T}_1 = (S_1, s_1^0, E_1)$ and $\mathcal{T}_2 = (S_2, s_2^0, E_2)$

Language equivalence

$\mathcal{T}_1$ and $\mathcal{T}_2$ are language-equivalent if they accept the same sets of timed observation sequences (with respect to accepting conditions).

Weak timed bisimulation

With an alphabet Act containing an internal action ε, the systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are weaky timed bisimilar if there is an equivalence relation on $S_1 \times S_2$ such that s_1^0 and s_2^0 are equivalent and:

if $s_1 \xrightarrow{a} s_1'$, $s_1 \sim s_2$, $s_1' \sim s_2'$

then $s_2 \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} \cdots \xrightarrow{a} \cdots \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} s_2'$

if $s_1 \xrightarrow{d} s_1'$, $s_1 \sim s_2$, $s_1' \sim s_2'$

then $s_2 \xrightarrow{d_1} \cdots \xrightarrow{\varepsilon} \cdots \xrightarrow{d_2} \cdots \xrightarrow{\varepsilon} \cdots \xrightarrow{d_k} s_2'$

with $\Sigma d_i = d$

Which equivalence relation ?

between two timed transition systems $\mathcal{T}_1 = (S_1, s_1^0, E_1)$ and $\mathcal{T}_2 = (S_2, s_2^0, E_2)$

Language equivalence

$\mathcal{T}_1$ and $\mathcal{T}_2$ are language-equivalent if they accept the same sets of timed observation sequences (with respect to accepting conditions).

Weak timed bisimulation

With an alphabet Act containing an internal action ε, the systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are weaky timed bisimilar if there is an equivalence relation on $S_1 \times S_2$ such that s_1^0 and s_2^0 are equivalent and:

$$\begin{array}{ll} \text{if} & s_1 \xrightarrow{a} s_1' \\ & \sim \qquad\qquad \sim \\ \text{then} & s_2 \xrightarrow{\varepsilon} \xrightarrow{\varepsilon} \cdots \xrightarrow{a} \cdots \xrightarrow{\varepsilon} \xrightarrow{\varepsilon} s_2' \end{array}$$

$$\begin{array}{ll} \text{if} & s_1 \xrightarrow{d} s_1' \\ & \sim \qquad\qquad \sim \\ \text{then} & s_2 \xrightarrow{d_1} \xrightarrow{\varepsilon} \cdots \xrightarrow{d_2} \cdots \xrightarrow{\varepsilon} \xrightarrow{d_k} s_2' \end{array}$$

with $\Sigma d_i = d$

and vice versa.

A global view

of timed transition systems

Discrete part
Control states/transitions

$\otimes$

Timed part
Valuations

- Discrete part: TPNs are more expressive than TA
  Unbounded TPNs can represent an infinite number of discrete states.
- Timed part: TA are more expressive than TPNs
  - In TPNs, transitions are controlled by a single clock,
  - clock reset is associated with newly enabled transitions,
  - lazy behaviour in not possible.

For weak timed bisimilarity

Bounded-TPN $\preceq_{\mathcal{W}}$ TA [Cassez, Roux 2004] (and Bounded-TPN $\subset$ TPN).

A global view

of timed transition systems

Discrete part
Control states/transitions

$\otimes$

Timed part
Valuations

- Discrete part: TPNs are more expressive than TA
  Unbounded TPNs can represent an infinite number of discrete states.
- Timed part: TA are more expressive than TPNs
  - In TPNs, transitions are controlled by a single clock,
  - clock reset is associated with newly enabled transitions,
  - lazy behaviour in not possible.

For weak timed bisimilarity

Bounded-TPN $\preceq_{\mathcal{W}}$ TA [Cassez, Roux 2004] (and Bounded-TPN $\subset$ TPN).

A global view

of timed transition systems

Discrete part
Control states/transitions

$\otimes$

Timed part
Valuations

- Discrete part: TPNs are more expressive than TA
  Unbounded TPNs can represent an infinite number of discrete states.
- Timed part: TA are more expressive than TPNs
  - In TPNs, transitions are controlled by a single clock,
  - clock reset is associated with newly enabled transitions,
  - lazy behaviour in not possible.

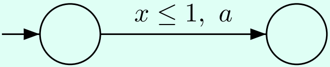
For weak timed bisimilarity

Bounded-TPN $\preceq_{\mathcal{W}}$ TA [Cassez, Roux 2004] (and Bounded-TPN $\subset$ TPN).

Back to semantics

A timing property of TPNs

- Time elapsing does not disable transition firing.
- For the following TA, there is no (weakly timed) bisimilar TPN [BCHLR 2005].



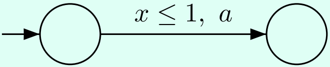
Three questions:

- Investigate the power of reset (memory policy) in TPNs : when should a transition be newly enabled ?
- What about comparing the models with language equivalence ?
- What is the maximal subclass of TA for which there exists a bisimilar TPN ?

Back to semantics

A timing property of TPNs

- Time elapsing does not disable transition firing.
- For the following TA, there is no (weakly timed) bisimilar TPN [BCHLR 2005].



Three questions:

- Investigate the power of reset (memory policy) in TPNs : when should a transition be newly enabled ?
- What about comparing the models with language equivalence ?
- What is the maximal subclass of TA for which there exists a bisimilar TPN ?

Reset in TPNs (1)

We consider three semantics

For a transition enabled after a firing:

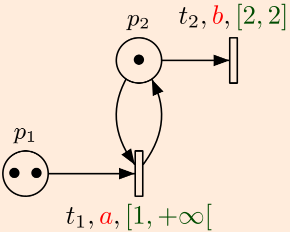
- Intermediate (classical) semantics (I): the transition is newly enabled if it was disabled after the consuming step or if it is the fired transition.
- Atomic semantics (A): the transition is newly enabled if it was disabled before the firing or if it is the fired transition.
- Persistent atomic semantics (PA): the transition is newly enabled if it was disabled before the firing.

Reset in TPNs (1)

We consider three semantics

For a transition enabled after a firing:

- Intermediate (classical) semantics (I): the transition is newly enabled if it was disabled after the consuming step or if it is the fired transition.
- Atomic semantics (A): the transition is newly enabled if it was disabled before the firing or if it is the fired transition.
- Persistent atomic semantics (PA): the transition is newly enabled if it was disabled before the firing.

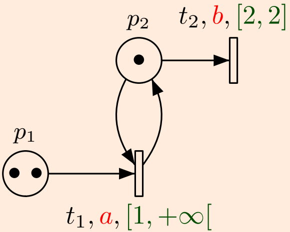


Reset in TPNs (1)

We consider three semantics

For a transition enabled after a firing:

- Intermediate (classical) semantics (I): the transition is newly enabled if it was disabled after the consuming step or if it is the fired transition.
- Atomic semantics (A): the transition is newly enabled if it was disabled before the firing or if it is the fired transition.
- Persistent atomic semantics (PA): the transition is newly enabled if it was disabled before the firing.



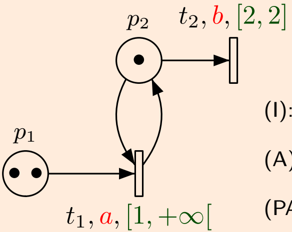
$$(\boldsymbol{2p_1 + p_2, [0,0]}) \xrightarrow{\mathbf{1.3}} (\boldsymbol{2p_1 + p_2, [1.3, 1.3]}) \xrightarrow{\boldsymbol{a}} \cdots$$

Reset in TPNs (1)

We consider three semantics

For a transition enabled after a firing:

- Intermediate (classical) semantics (I): the transition is newly enabled if it was disabled after the consuming step or if it is the fired transition.
- Atomic semantics (A): the transition is newly enabled if it was disabled before the firing or if it is the fired transition.
- Persistent atomic semantics (PA): the transition is newly enabled if it was disabled before the firing.



$$(\boldsymbol{2p_1 + p_2}, [\mathbf{0}, \mathbf{0}]) \xrightarrow{\mathbf{1.3}} (\boldsymbol{2p_1 + p_2}, [\mathbf{1.3}, \mathbf{1.3}]) \xrightarrow{\boldsymbol{a}} \cdots$$

(I): $(\boldsymbol{p_1 + p_2}, [\mathbf{0}, \mathbf{0}]) \xrightarrow{\mathbf{2}} (\boldsymbol{p_1 + p_2}, [\mathbf{2}, \mathbf{2}])$

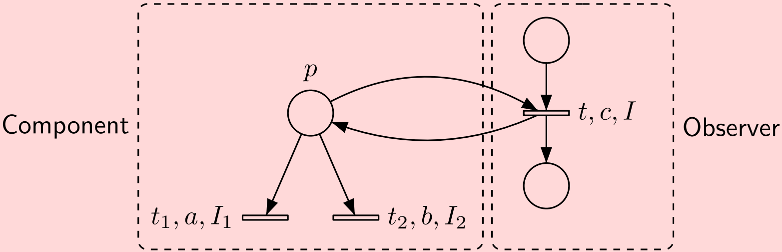
(A): $(\boldsymbol{p_1 + p_2}, [\mathbf{0}, \mathbf{1.3}]) \xrightarrow{\mathbf{0.7}} (\boldsymbol{p_1 + p_2}, [\mathbf{0.7}, \mathbf{2}]) \xrightarrow{\boldsymbol{b}} (\boldsymbol{p_1}, [\perp, \perp])$

(PA): $(\boldsymbol{p_1 + p_2}, [\mathbf{1.3}, \mathbf{1.3}]) \xrightarrow{\boldsymbol{a}} (\boldsymbol{p_2}, [\perp, \mathbf{1.3}])$

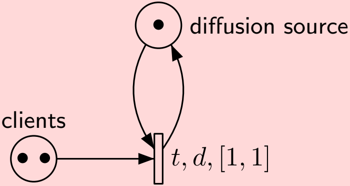
Reset in TPNs (2)

Why alternative semantics ?

- (PA) is closer to the semantics of TA
- (A) or (PA) are sometimes more convenient than (I):



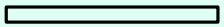
- For e.g. instantaneous multicast, (PA) is more convenient than (A) or (I):



Reset in TPNs: results

(PA) semantics is the most expressive [BCHLR ATVA 2005]

- $\mathsf{TPN}_{(I)} \preceq_{\mathcal{W}} \mathsf{TPN}_{(A)} \preceq_{\mathcal{W}} \mathsf{TPN}_{(PA)}$
- (PA) is strictly more expressive than (A): $\mathsf{TPN}_{(A)} <_{\mathcal{W}} \mathsf{TPN}_{(PA)}$.
  For the following TPN with (PA) semantics, there is no bisimlar TPN with (A) semantics.



$t,\ \varepsilon,\ [0,1[$

- For Bounded-TPNs with upper-closed intervals, the three semantics are equivalent: for any such net in $\mathsf{TPN}_{(PA)}$, there exists a net in $\mathsf{TPN}_{(I)}$ which is bisimilar.

Comparing with language equivalence

TPNs and TA are equally expressive [BCHLR FORMATS 2005]

Bounded-TPN $=_{\mathcal{L}}$ TA

Proof

It consists in the construction of a TPN accepting the same language as a given timed automaton $\mathcal{A}$, and involves constructions of subnets encoding atomic constraints and clock reset.

A transition $e: \ q_1 \xrightarrow{g,a,r} q_2$, with $g = g_1 \wedge g_2$, is simulated by a subnet of the form:

Comparing with language equivalence

TPNs and TA are equally expressive [BCHLR FORMATS 2005]

Bounded-TPN $=_{\mathcal{L}}$ TA

Proof

It consists in the construction of a TPN accepting the same language as a given timed automaton $\mathcal{A}$, and involves constructions of subnets encoding atomic constraints and clock reset.

A transition $e:\ q_1 \xrightarrow{g,a,r} q_2$, with $g = g_1 \wedge g_2$, is simulated by a subnet of the form:

Comparing with language equivalence

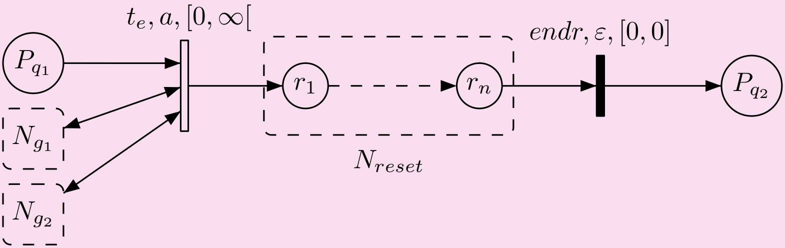
TPNs and TA are equally expressive [BCHLR FORMATS 2005]

Bounded-TPN $=_{\mathcal{L}}$ TA

Proof

It consists in the construction of a TPN accepting the same language as a given timed automaton $\mathcal{A}$, and involves constructions of subnets encoding atomic constraints and clock reset.

A transition $e: \ q_1 \xrightarrow{g,a,r} q_2$, with $g = g_1 \wedge g_2$, is simulated by a subnet of the form:



Characterisation of TA bisimilar to TPNs

Find TA_{wb} the maximal subclass of TA for which there is a bisimilar TPN

The characterisation is expressed with topological properties of the region automaton, used for analysis of a timed automaton.

Characterisation of TA bisimilar to TPNs

Find TA_{wb} the maximal subclass of TA for which there is a bisimilar TPN

The characterisation is expressed with topological properties of the region automaton, used for analysis of a timed automaton.

Characterisation of TA bisimilar to TPNs

Find TA_{wb} the maximal subclass of TA for which there is a bisimilar TPN

The characterisation is expressed with topological properties of the region automaton, used for analysis of a timed automaton.

Timed automaton

Discrete part
Control states/transitions

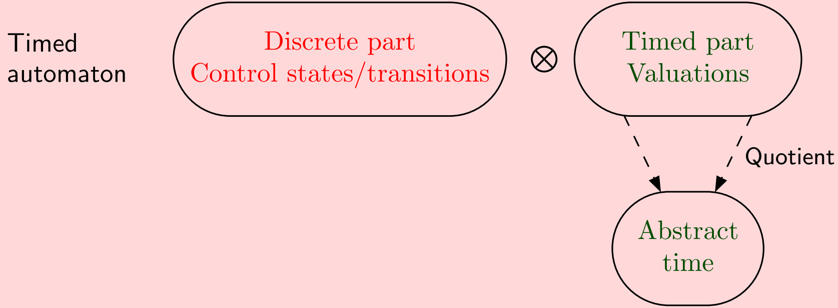
$\otimes$

Timed part
Valuations

Characterisation of TA bisimilar to TPNs

Find TA$_{wb}$ the maximal subclass of TA for which there is a bisimilar TPN

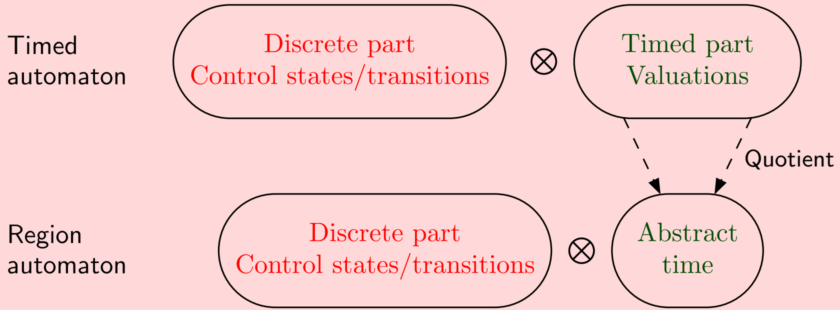
The characterisation is expressed with topological properties of the region automaton, used for analysis of a timed automaton.



Characterisation of TA bisimilar to TPNs

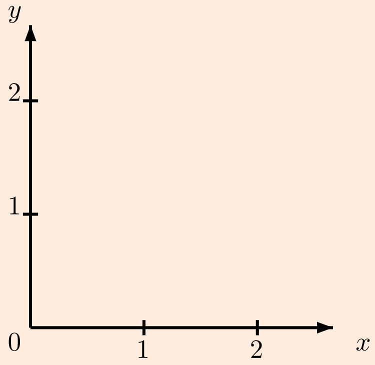
Find TA_{wb} the maximal subclass of TA for which there is a bisimilar TPN

The characterisation is expressed with topological properties of the region automaton, used for analysis of a timed automaton.



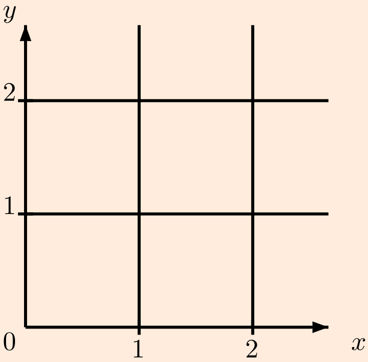
Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



Quotient: a geometric view

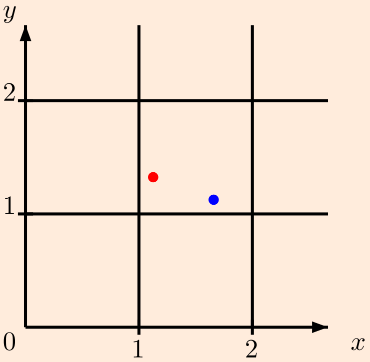
with two clocks x and y and maximal constant $m = 2$



- Equivalent valuations satisfy the same constraints $x \bowtie k$

Quotient: a geometric view

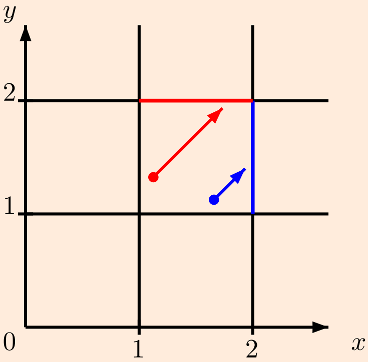
with two clocks x and y and maximal constant $m = 2$



- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

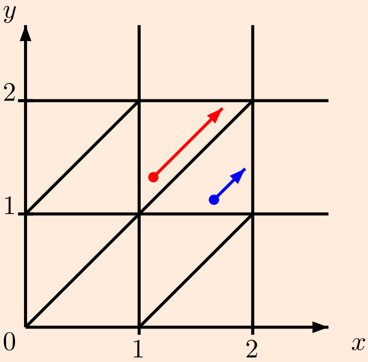
with two clocks x and y and maximal constant $m = 2$



- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

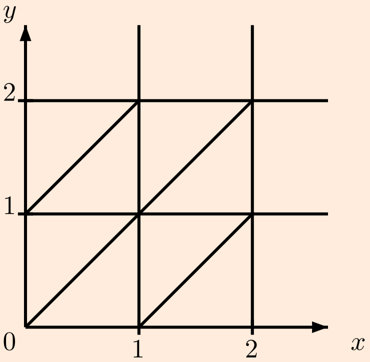
with two clocks x and y and maximal constant $m = 2$



- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

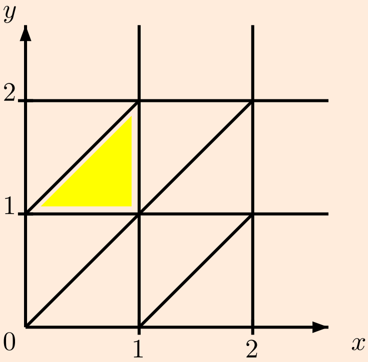
with two clocks x and y and maximal constant $m = 2$



- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



region R defined by

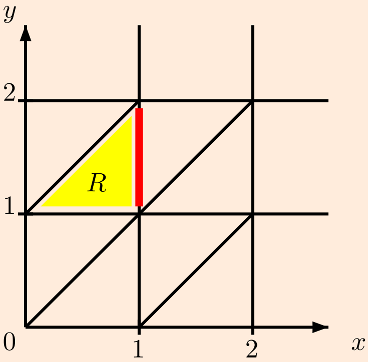
$I_x =]0;1[$, $I_y =]1;2[$

$frac(x) > frac(y)$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



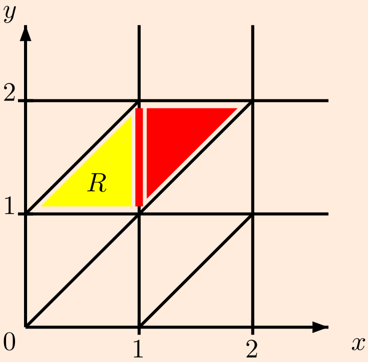
region R defined by
$I_x =]0;1[$, $I_y =]1;2[$
$frac(x) > frac(y)$

Time successor of R
$I_x = [1;1]$, $I_y =]1;2[$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



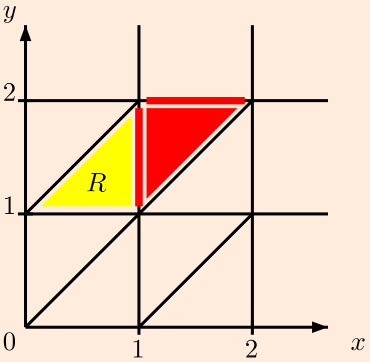
region R defined by
$I_x =]0;1[$, $I_y =]1;2[$
$frac(x) > frac(y)$

Time successor of R
$I_x = [1;1]$, $I_y =]1;2[$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



region R defined by

$I_x =]0;1[$, $I_y =]1;2[$

$frac(x) > frac(y)$

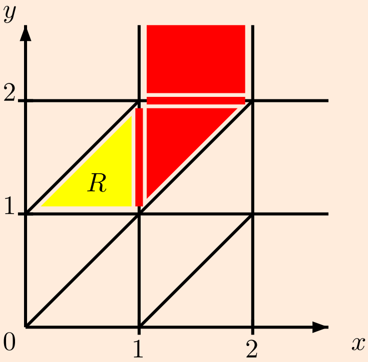
Time successor of R

$I_x = [1;1]$, $I_y =]1;2[$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



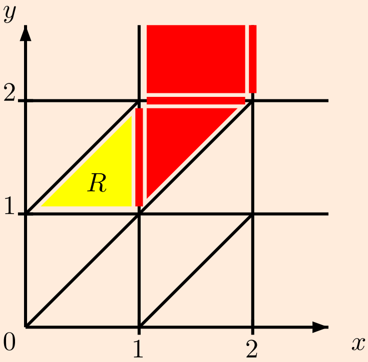
region R defined by
$I_x =]0;1[$, $I_y =]1;2[$
$frac(x) > frac(y)$

Time successor of R
$I_x = [1;1]$, $I_y =]1;2[$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



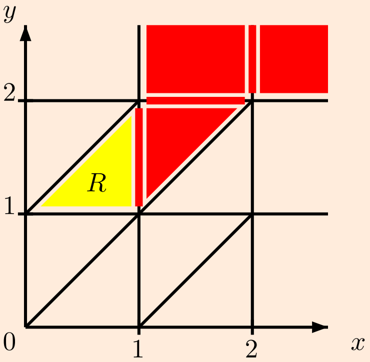
region R defined by
$I_x =]0;1[$, $I_y =]1;2[$
$frac(x) > frac(y)$

Time successor of R
$I_x = [1;1]$, $I_y =]1;2[$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



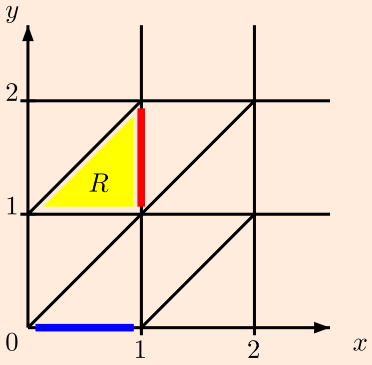
region R defined by
$I_x =]0;1[$, $I_y =]1;2[$
$frac(x) > frac(y)$

Time successor of R
$I_x = [1;1]$, $I_y =]1;2[$

- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Quotient: a geometric view

with two clocks x and y and maximal constant $m = 2$



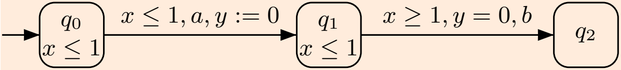
region R defined by
$I_x =]0;1[$, $I_y =]1;2[$
$frac(x) > frac(y)$

Time successor of R
$I_x = [1;1]$, $I_y =]1;2[$

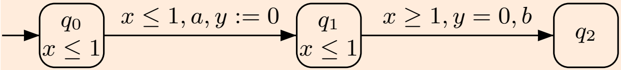
Action successor of R
with $y := 0$
$I_x =]0;1[$, $I_y = [0;0]$

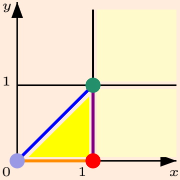
- Equivalent valuations satisfy the same constraints $x \bowtie k$
- Equivalent valuations respect time elapsing

Region automaton: example

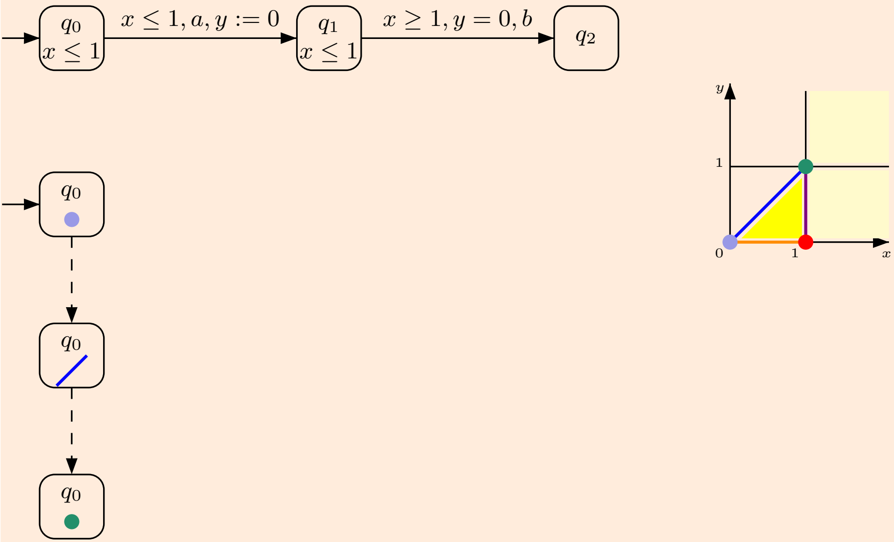


Region automaton: example

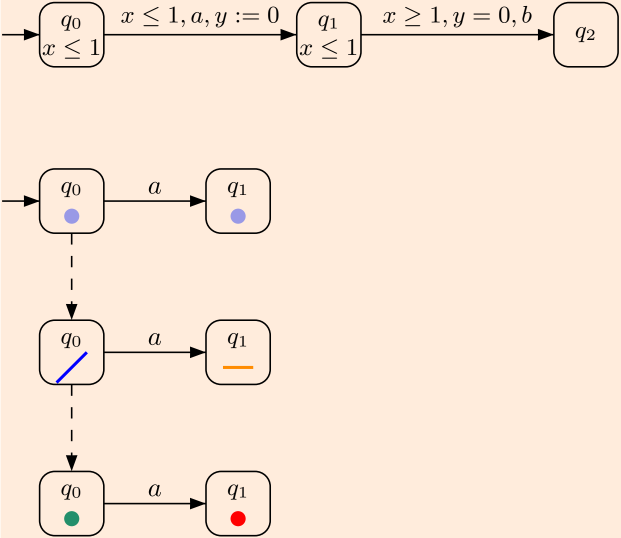


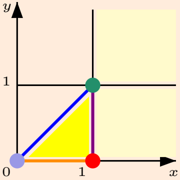


Region automaton: example

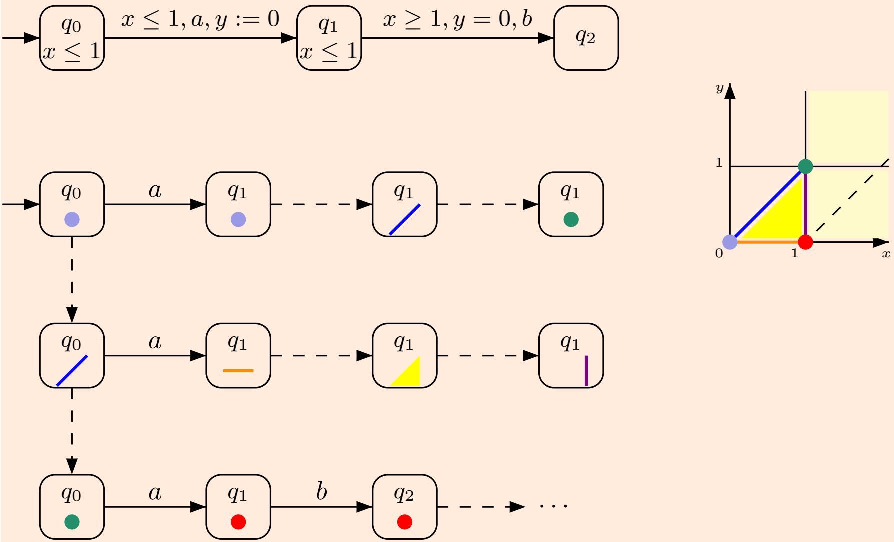


Region automaton: example





Region automaton: example



The maximal subclass TA_{wb}

contains all timed automata

where transitions have a unique label and:

(a) if R is reachable then

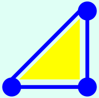
The maximal subclass TA_{wb}

contains all timed automata

where transitions have a unique label and:

(a) 

if R is reachable then
for all R' s.t. $R' \cap closure(R) \neq \emptyset$
R' is reachable



The maximal subclass TA$_{wb}$

contains all timed automata

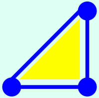
where transitions have a unique label and:

(a)

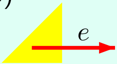
if R is reachable then

for all R' s.t. $R' \cap closure(R) \neq \emptyset$

R' is reachable



(b)



if R is reachable and $R \xrightarrow{e}$

The maximal subclass TA_{wb}

contains all timed automata

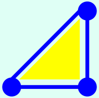
where transitions have a unique label and:

(a)

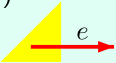
if R is reachable then

for all R' s.t. $R' \cap closure(R) \neq \emptyset$

R' is reachable

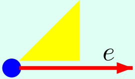


(b)



if R is reachable and $R \xrightarrow{e}$

then $min(R) \xrightarrow{e}$

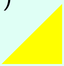


The maximal subclass TA_{wb}

contains all timed automata

where transitions have a unique label and:

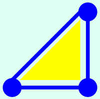
(a)



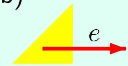
if R is reachable then

for all R' s.t. $R' \cap closure(R) \neq \emptyset$

R' is reachable

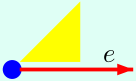


(b)

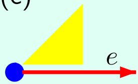


if R is reachable and $R \xrightarrow{e}$

then $min(R) \xrightarrow{e}$



(c)



if R is reachable and $min(R) \xrightarrow{e}$ then

The maximal subclass TA_{wb}

contains all timed automata

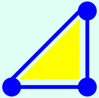
where transitions have a unique label and:

(a)

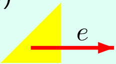
if R is reachable then

for all R' s.t. $R' \cap closure(R) \neq \emptyset$

R' is reachable

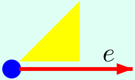


(b)

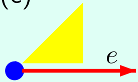


if R is reachable and $R \xrightarrow{e}$

then $min(R) \xrightarrow{e}$



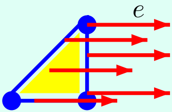
(c)



if R is reachable and $min(R) \xrightarrow{e}$ then

for all R' s.t. $R' \cap closure(R) \neq \emptyset$

$R' \xrightarrow{e}$



The maximal subclass TA$_{wb}$ (cont.)

Sketch of the proof

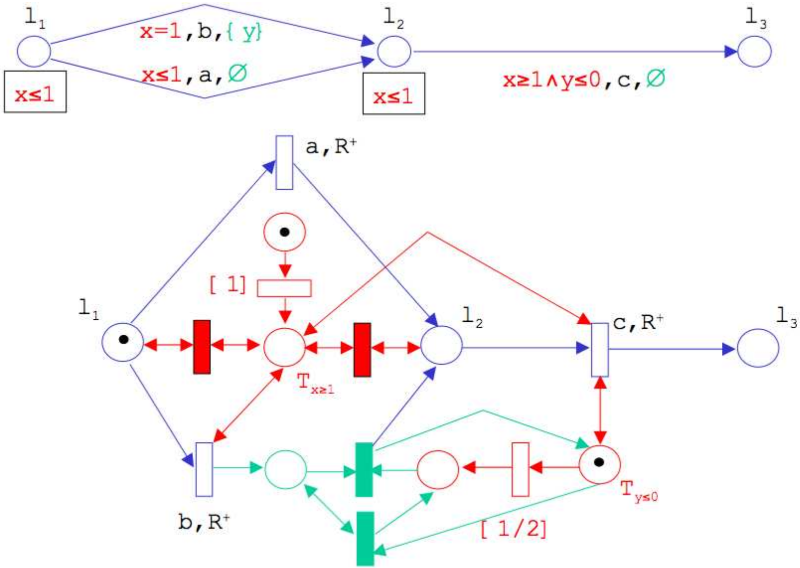
- For a timed automaton $\mathcal{A}$ in TA$_{wb}$, conditions (a), (b) and (c) hold in (a variant of) the region automaton.
  - For a TPN $\mathcal{N}$ with rational constants i/g, weakly timed bisimilar to $\mathcal{A}$, we consider a region automaton $\mathcal{R}(g, \infty)$, based on an infinite grid with granularity g.
  - We prove an extended property called *uniform bisimulation*, which implies conditions (a), (b), (c) for $\mathcal{R}(g, \infty)$.
  - The conditions are then lifted from $\mathcal{R}(g, \infty)$ to $\mathcal{R}(1, \infty)$ and then to some $\mathcal{R}(1, K)$.
- Conversely, if conditions (a), (b) and (c) hold for a timed automaton $\mathcal{A}$ then we can build:
  - a TPN $\mathcal{N}$ with integer constants and size exponential w.r.t. the size of $\mathcal{A}$,
  - a TPN $\mathcal{N}$ with rational constants and size linear w.r.t. the size of $\mathcal{A}$.

The maximal subclass TA_{wb} (cont.)

Sketch of the proof

- For a timed automaton $\mathcal{A}$ in TA_{wb}, conditions (a), (b) and (c) hold in (a variant of) the region automaton.
  - For a TPN $\mathcal{N}$ with rational constants i/g, weakly timed bisimilar to $\mathcal{A}$, we consider a region automaton $\mathcal{R}(g, \infty)$, based on an infinite grid with granularity g.
  - We prove an extended property called *uniform bisimulation*, which implies conditions (a), (b), (c) for $\mathcal{R}(g, \infty)$.
  - The conditions are then lifted from $\mathcal{R}(g, \infty)$ to $\mathcal{R}(1, \infty)$ and then to some $\mathcal{R}(1, K)$.
- Conversely, if conditions (a), (b) and (c) hold for a timed automaton $\mathcal{A}$ then we can build:
  - a TPN $\mathcal{N}$ with integer constants and size exponential w.r.t. the size of $\mathcal{A}$,
  - a TPN $\mathcal{N}$ with rational constants and size linear w.r.t. the size of $\mathcal{A}$.

Illustration of second construction



Outline

Conclusion

Comparing Time Petri nets and Timed Automata

- can be useful for practical purposes: the tools developed for both models are available via translation,
- provides a better view of the behaviour of timed models,
- creates a fruitful relation between the two communities.

Perspectives

- compare unfolding techniques for nets of timed automata and time Petri nets (work in progress in the DOTS project),
- study control problems and game theory for timed models,
- specify non-interference and covert channel detection for timed systems,
- consider other quantitative extensions with costs or probabilities.

Conclusion

Comparing Time Petri nets and Timed Automata

- can be useful for practical purposes: the tools developed for both models are available via translation,
- provides a better view of the behaviour of timed models,
- creates a fruitful relation between the two communities.

Perspectives

- compare unfolding techniques for nets of timed automata and time Petri nets (work in progress in the DOTS project),
- study control problems and game theory for timed models,
- specify non-interference and covert channel detection for timed systems,
- consider other quantitative extensions with costs or probabilities.

Thank you