Timed Automata Can Always Be Made Implementable

Patricia Bouyer[1], Kim Larsen[2], Nicolas Markey[1], **Ocan Sankur**[1], Claus Thrane[2]

[1] LSV, CNRS & Ecole Normale Supérieure de Cachan

[2] Dept. of Computer Science, Aalborg University

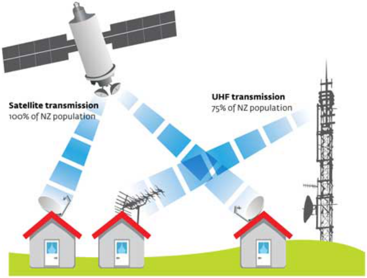
CONCUR 2011

September 6, 2011

Timed Systems: Systems with timing constraints

- Communication protocols,
- Multimedia applications,
- Car/airplane components, ...

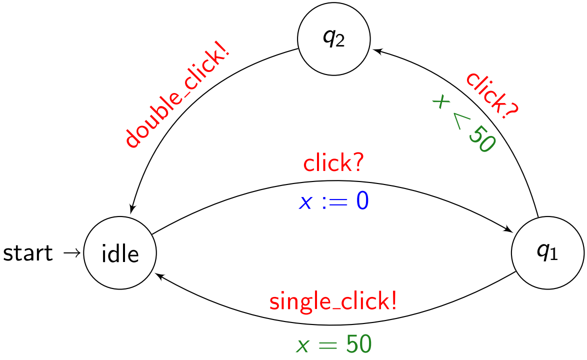
To faithfully model systems, one often needs to talk about **time**.



- We model these by *Timed Automata*.

Abstract Model: Timed Automata (TA)

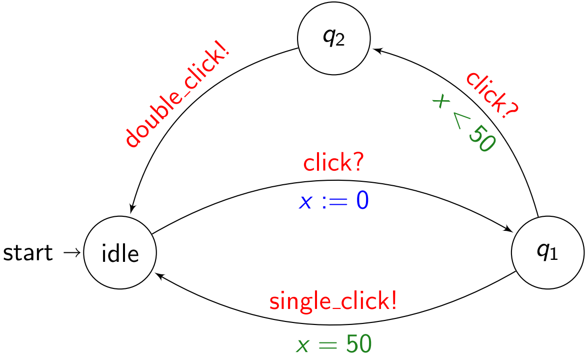
Timed automata = Finite automata + Analog clocks. [Alur and Dill 1994]



- Clocks cannot be stopped, all grow at the same rate.
- An edge is activated when its clock constraint holds.
- A clock can be reset by a transition.

Abstract Model: Timed Automata (TA)

Timed automata = Finite automata + Analog clocks. [Alur and Dill 1994]



Runs of a timed automaton

$(\text{idle}, x = 0) \xrightarrow{23.7} (\text{idle}, x = 23.7) \xrightarrow{click?} (q_1, x = 0) \xrightarrow{10} (q_1, x = 10)$
$\xrightarrow{click?} (q_2, x = 10) \xrightarrow{double\_click} (\text{idle}, x = 10) \ldots$

Robustness Issues in Timed Automata

The semantics of timed automata is idealistic:

- No minimum delay between actions, $\xrightarrow{a}\xrightarrow{0.00001}\xrightarrow{b}$.
- clocks are infinitely precise. "$1 \leq x \leq 3$".

But real world systems have finite frequency, digital clocks...

Robustness Issues in Timed Automata

The semantics of timed automata is idealistic:

- No minimum delay between actions, $\xrightarrow{a}\xrightarrow{0.00001}\xrightarrow{b}$.
- clocks are infinitely precise. "$1 \leq x \leq 3$".

But real world systems have finite frequency, digital clocks...

Two types of implementation behaviour

- **Sampled semantics**
- **Imprecise semantics**

Robustness Issues in Timed Automata

The semantics of timed automata is idealistic:

- No minimum delay between actions, $\xrightarrow{a}\xrightarrow{0.00001}\xrightarrow{b}$.
- clocks are infinitely precise. "$1 \leq x \leq 3$".

But real world systems have finite frequency, digital clocks...

Two types of implementation behaviour

- **Sampled semantics**
  Time domain is replaced by $\frac{1}{n}\mathbb{N}$ for some $n \in \mathbb{N}_+$.
  applies to digital circuits, synchronous systems..
- **Imprecise semantics**

Robustness Issues in Timed Automata

The semantics of timed automata is idealistic:

- No minimum delay between actions, $\xrightarrow{a}\xrightarrow{0.00001}\xrightarrow{b}$.
- clocks are infinitely precise. "$1 \leq x \leq 3$".

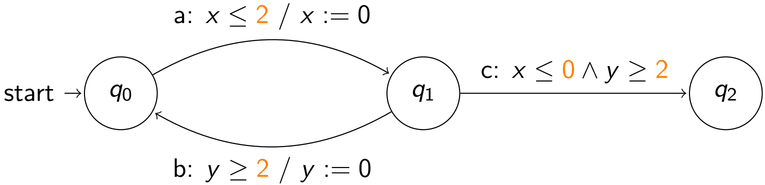
But real world systems have finite frequency, digital clocks...

Two types of implementation behaviour

- **Sampled semantics**
- **Imprecise semantics**
  applies to programs interacting with physical environment: – **next slide.**

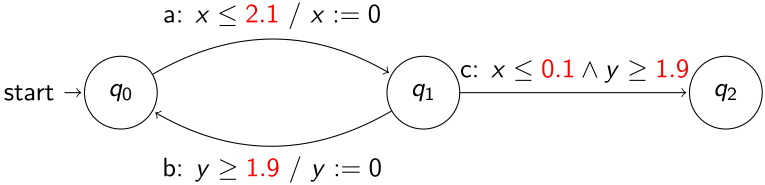
Imprecise Semantics

Clock imprecisions can be modelled by **enlarging** the clock constraints. Consider the timed automaton $\mathcal{A}$:



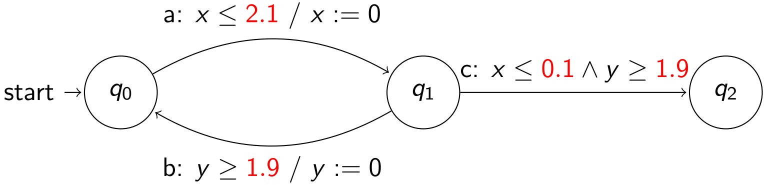
Imprecise Semantics

Clock imprecisions can be modelled by **enlarging** the clock constraints.
For $\Delta = 0.1$, $\mathsf{Imprecise}_\Delta(\mathcal{A})$ is defined by,



Imprecise Semantics

Clock imprecisions can be modelled by **enlarging** the clock constraints.
For $\Delta = 0.1$, $\text{Imprecise}_\Delta(\mathcal{A})$ is defined by,

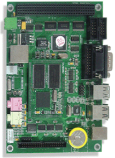


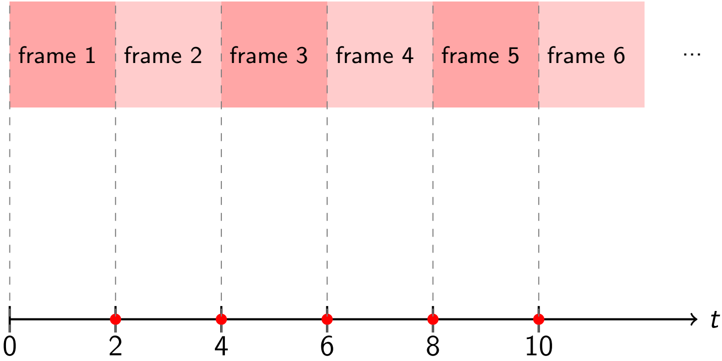
This is an over-approximation of a concrete semantics
when $\mathcal{A}$ is "executed" by a micro-processor.

Δ corresponds to the *clock error* and *hardware frequency*

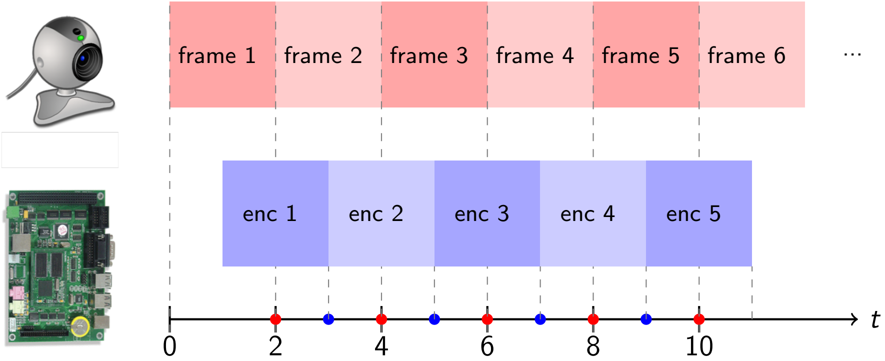
[De Wulf, Doyen, Raskin 2004]

A Non-Robust Timed System in the Imprecise Semantics

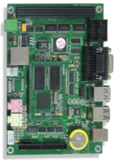


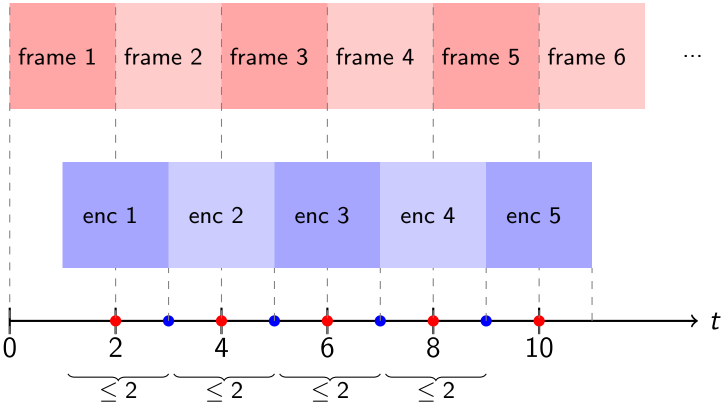


A Non-Robust Timed System in the Imprecise Semantics

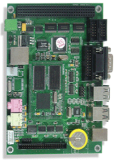


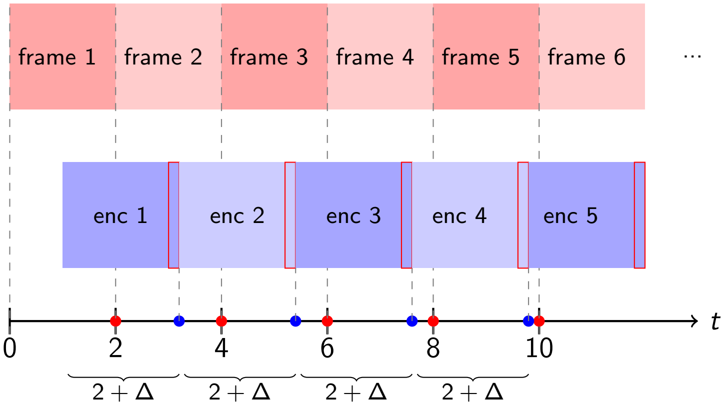
A Non-Robust Timed System in the Imprecise Semantics



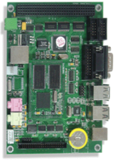


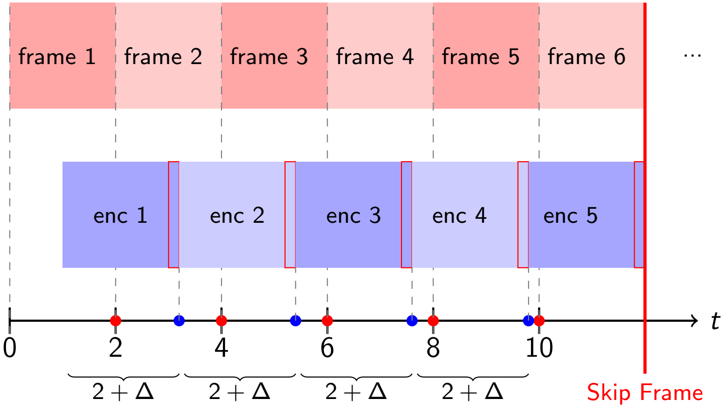
A Non-Robust Timed System in the Imprecise Semantics



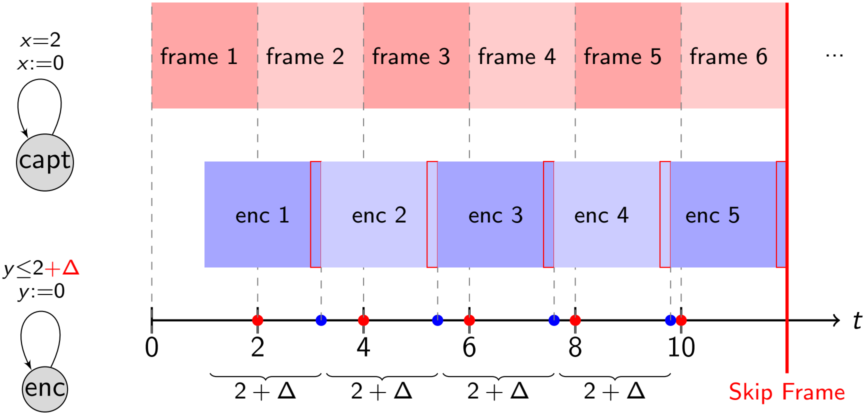


A Non-Robust Timed System in the Imprecise Semantics





A Non-Robust Timed System in the Imprecise Semantics



Background: Imprecise semantics

"Imprecise semantics can **add undesired** behaviour to timed automata".

[Puri 1998, DDMR 2004]

Robustness checking

Given TA $\mathcal{A}$ and property ϕ, decide if $\exists\Delta > 0$, $\text{Imprecise}_\Delta(\mathcal{A}) \models \phi$.

Decidable for:

- Safety, [Puri 1998], [De Wulf, Doyen, Markey, Raskin 2004], [Jaubert, Reynier 2011]
- LTL, a fragment of MTL, [Bouyer, Markey, Reynier 2006 - 2008].
- Untimed language equivalence $L(\mathcal{A}) = L(\text{Imprecise}_\Delta(\mathcal{A}))$ [S. 2011]

Background: Sampled Semantics

"Sampled semantics can **remove desired** behaviour from timed automata". [Cassez, Henzinger, Raskin 02]

Samplability checking

Given TA $\mathcal{A}$ and property ϕ, decide if $\exists n \in \mathbb{N}_+$, $\text{Sampled}_{\frac{1}{n}}(\mathcal{A}) \models \phi$.

Decidable for:

- Reachability, [Krčál, Pelánek 2005]
- Untimed language equivalence, [Abdulla, Krčál, Yi 2010]

Undecidable for:

- Safety, [Cassez, Henzinger, Raskin 2002]

Results

In this work: Instead of robustness/samplability checking transform any timed automaton into an "equivalent" one that is robust/samplable.

Results

Preliminary definition: Two states are ϵ-**bisimilar** if there is a bisimulation in which delays differ by at most ϵ. — denoted by $\sim_\epsilon$

Results

Theorem (Robustness construction)

Given any timed automaton $\mathcal{A}$, any $\epsilon > 0$, there exists $\mathcal{A}'$ such that

- $\mathcal{A} \sim_0 \mathcal{A}'$,
- $\mathcal{A}' \sim_\epsilon \mathsf{Imprecise}_\Delta(\mathcal{A}')$ for all $0 \leq \Delta < O(\epsilon)$,
- $\mathcal{A}' \sim_\epsilon \mathsf{Sampled}_{\frac{1}{n}}(\mathcal{A}')$ for any $0 < \frac{1}{n} < O(\epsilon)$.

▶ We get $\mathcal{A} \sim_\epsilon \mathsf{Imprecise}_\Delta(\mathcal{A}')$ and $\mathcal{A} \sim_\epsilon \mathsf{Sampled}_{\frac{1}{n}}(\mathcal{A}')$.

Practical meaning: Model-check $\mathcal{A}$, then implement $\mathcal{A}'$.

Results

Theorem (Robustness construction)

Given any timed automaton $\mathcal{A}$, any $\epsilon > 0$, there exists $\mathcal{A}'$ such that

- $\mathcal{A} \sim_0 \mathcal{A}'$,
- $\mathcal{A}' \sim_\epsilon \mathrm{Imprecise}_\Delta(\mathcal{A}')$ for all $0 \leq \Delta < O(\epsilon)$,
- $\mathcal{A}' \sim_\epsilon \mathrm{Sampled}_{\frac{1}{n}}(\mathcal{A}')$ for any $0 < \frac{1}{n} < O(\epsilon)$.

▶ We get $\mathcal{A} \sim_\epsilon \mathrm{Imprecise}_\Delta(\mathcal{A}')$ and $\mathcal{A} \sim_\epsilon \mathrm{Sampled}_{\frac{1}{n}}(\mathcal{A}')$.

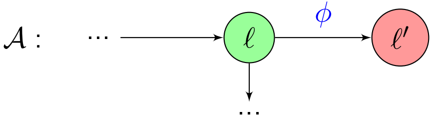
Practical meaning: Model-check $\mathcal{A}$, then implement $\mathcal{A}'$.

Next: Simple Case: Robustness construction for safety

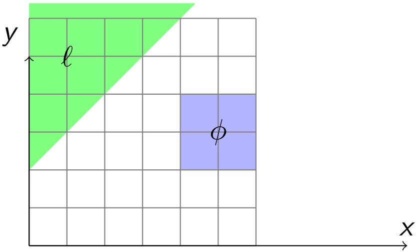
- $\mathcal{A} \sim_0 \mathcal{A}'$,
- $\mathcal{A}$ does not reach a location $\Rightarrow$ neither does $\mathrm{Imprecise}_\Delta(\mathcal{A}')$.

Idea of the construction

Consider a timed automaton $\mathcal{A}$ with clocks x, y,
such that location ℓ' is not reachable:

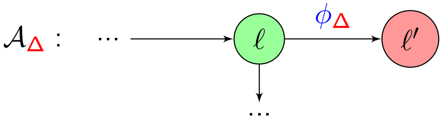


Consider the **reachable states** in ℓ:

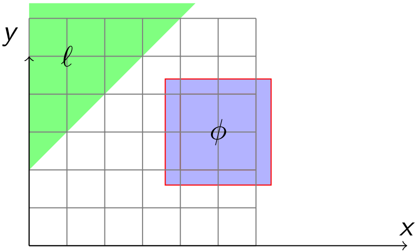


Idea of the construction

Consider a timed automaton $\mathcal{A}$ with clocks x, y,
such that location ℓ' is not reachable:

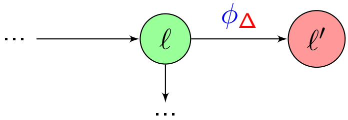


Consider the **reachable states** in ℓ:

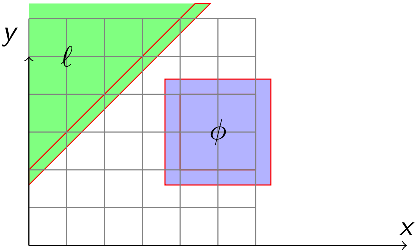


Idea of the construction

Consider a timed automaton $\mathcal{A}$ with clocks x, y,
such that location ℓ' is not reachable:

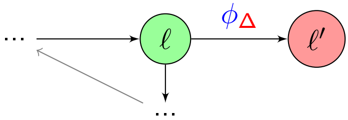


Consider the **reachable states** in ℓ:

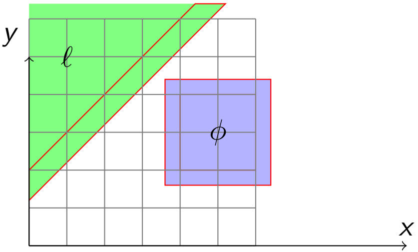


Idea of the construction

Consider a timed automaton $\mathcal{A}$ with clocks x, y,
such that location ℓ' is not reachable:

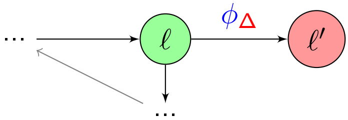


Consider the **reachable states** in ℓ:

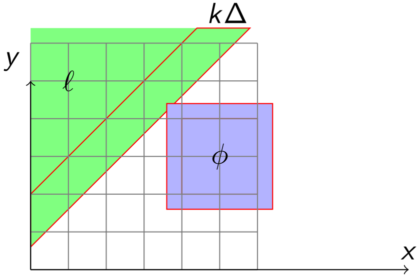


Idea of the construction

Consider a timed automaton $\mathcal{A}$ with clocks x, y, such that location ℓ' is not reachable:

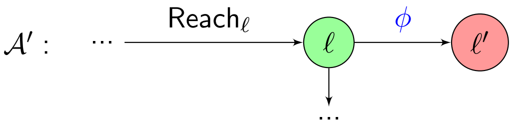


Consider the **reachable states** in ℓ: ℓ' reachable

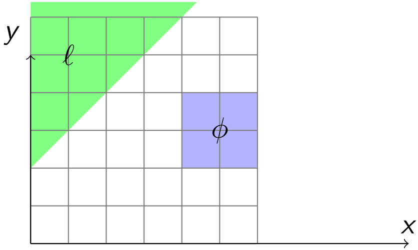


Idea of the construction

Define $\mathcal{A}'$ as follows:

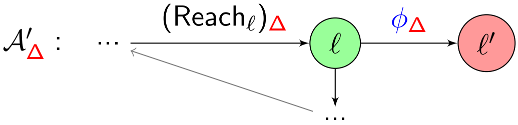


Reachable states in ℓ:

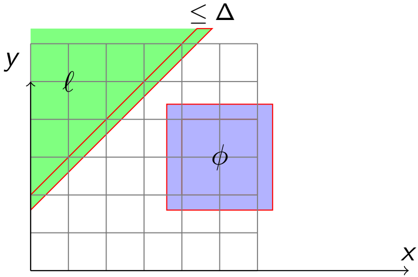


Idea of the construction

Define $\mathcal{A}'$ as follows:



Reachable states in ℓ: ℓ' **not** reachable in $\mathcal{A}'_\Delta$.



Construction for Safety-Robustness

For a timed automaton $\mathcal{A}$,

- Compute the set of reachable states Reach_ℓ at each location ℓ.
- Replace each edge

$$\ell \xrightarrow{\phi} \ell' \qquad \text{by} \qquad \ell \xrightarrow{\phi \wedge \text{Reach}_{\ell'}} \ell'$$

The resulting automaton $\mathcal{A}'$ satisfies

- $\mathcal{A} \sim_0 \mathcal{A}'$,
- $\mathcal{A}$ does not reach ℓ $\quad\Rightarrow\quad$ neither does $\mathcal{A}'_\Delta$ $\forall 0 < \Delta < \frac{1}{2c}$

▶ For the **bisimulation** construction, one needs to split each location to regions.

Property Preservation

Back to bisimulation...

What does $\mathcal{A} \sim_\epsilon \mathcal{A}'_\Delta$ and $\mathcal{A} \sim_\epsilon \mathsf{Sampled}_{\frac{1}{n}}(\mathcal{A}')$ imply?

Preservation of **untimed** properties, but also more...

Proposition (Property preservation)

We consider a quantitative extension of CTL [Fahrenberg, Larsen, Thrane 2010].

$$e.g. \qquad \mathbf{EX}_\sigma^{[2,5]}\top$$

- ϵ-bisimulation preserves satisfaction values of the formulas, up to ϵ.

Property Preservation

Back to bisimulation...

What does $\mathcal{A} \sim_\epsilon \mathcal{A}'_\Delta$ and $\mathcal{A} \sim_\epsilon \text{Sampled}_{\frac{1}{n}}(\mathcal{A}')$ imply?

Preservation of **untimed** properties, but also more...

Proposition (Property preservation)

We consider a quantitative extension of CTL [Fahrenberg, Larsen, Thrane 2010].

$$\phi\{\wedge,\vee\}\phi' \mid \mathbf{EX}_\sigma^{[a,b]}\phi \mid \mathbf{AX}_\sigma^{[a,b]}\phi \mid \mathbf{E}\phi\mathbf{U}_\sigma^{[a,b]}\phi' \mid \mathbf{A}\phi\mathbf{U}_\sigma^{[a,b]}\phi'$$

- ϵ-bisimulation preserves satisfaction values of the formulas, up to ϵ.

Conclusion

Conclusion

- We obtain arbitrarily close approximations of any timed automaton in implementation.
- Two constructions: **safety** (simpler), **bisimulation**.
- **Design advice for robust safety**:

  "Write explicitly all implied invariants in clock constraints."

Next

- Alternative approach: **shrink** the clock constraints
  [S., Bouyer, Markey 2011]
- Robust controller synthesis
- Probabilistic models for imprecisions