

# Robustness in real-time systems

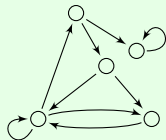
Nicolas Markey

LSV, CNRS & ENS Cachan, France

SIES'11 – June 15, 2011

# Verification of (real-time) computerized systems

system:



model-checking  
algorithm

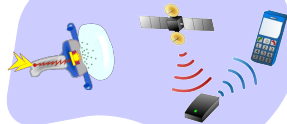


**Always safe**



yes/no

property:

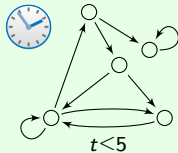
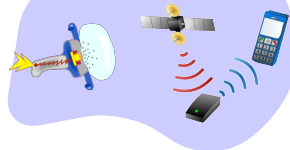


# Verification of (real-time) computerized systems

system:



property:



model-checking  
algorithm

**Always safe**

yes/no

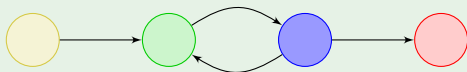
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,

## Example



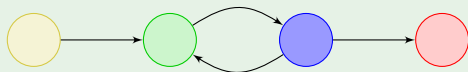
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,

## Example



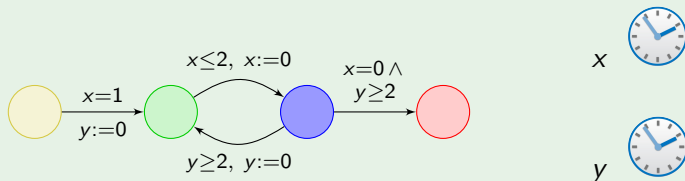
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



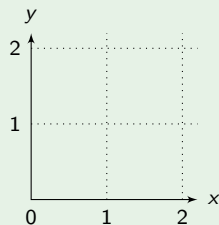
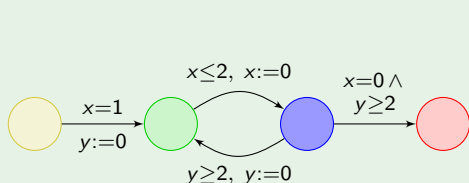
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



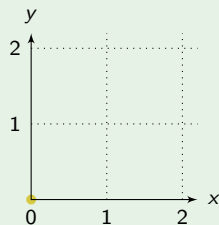
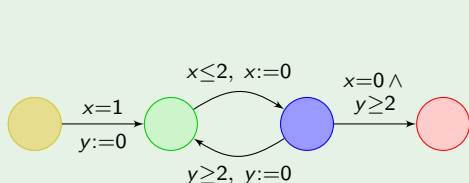
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



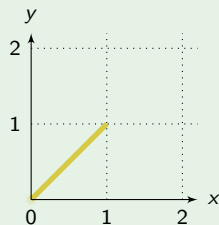
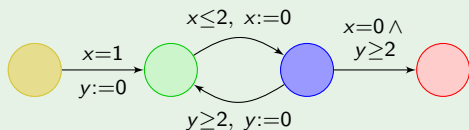
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



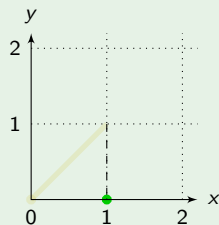
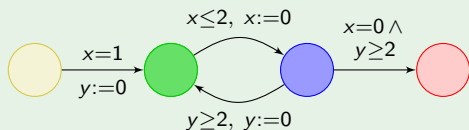
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



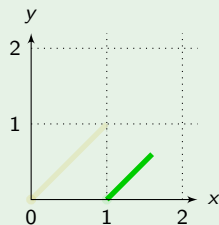
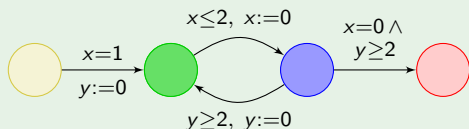
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



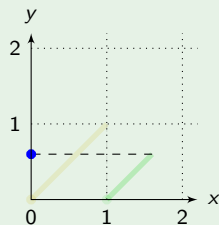
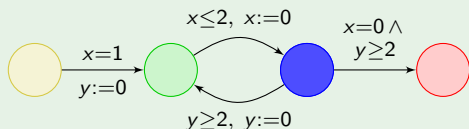
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



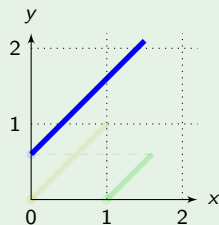
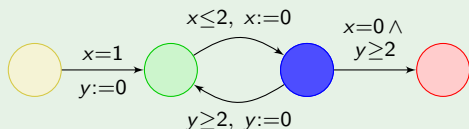
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



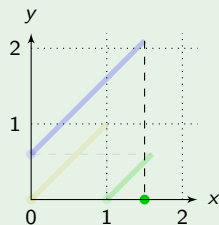
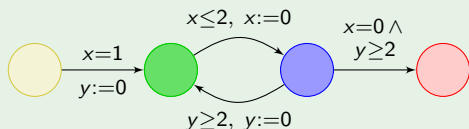
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



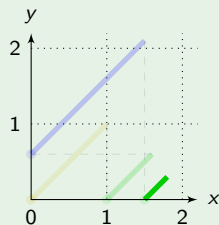
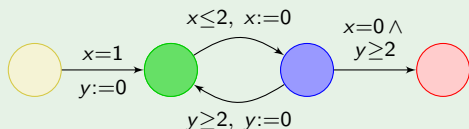
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



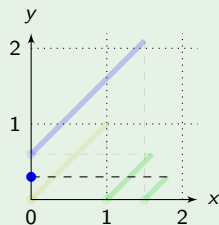
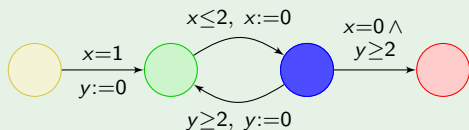
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



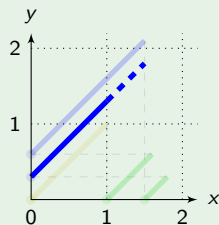
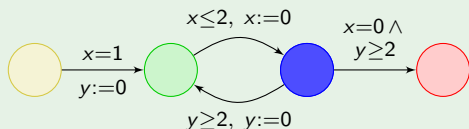
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



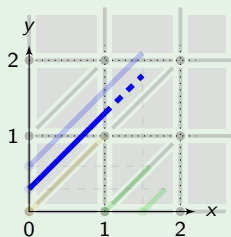
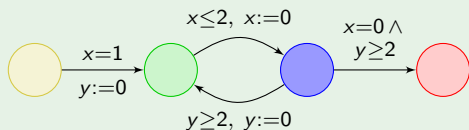
# Timed automata

## Timed automata (AD90)

A **timed automaton** is made of

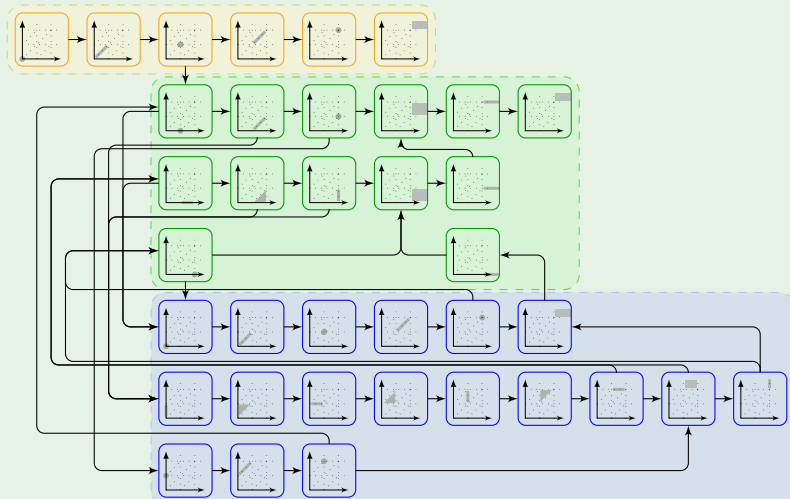
- a transition system,
- a set of clocks,
- a labelling of transitions with timing informations.

## Example



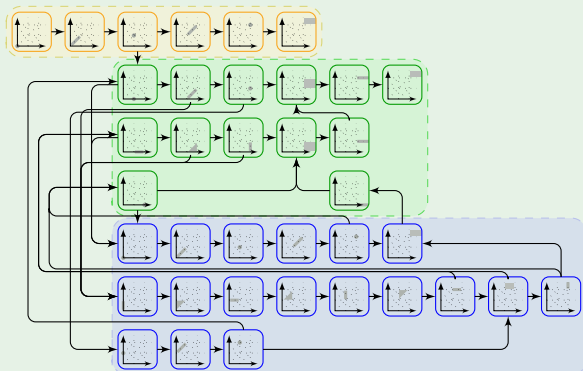
# Region automata

## Example



# Region automata

## Example

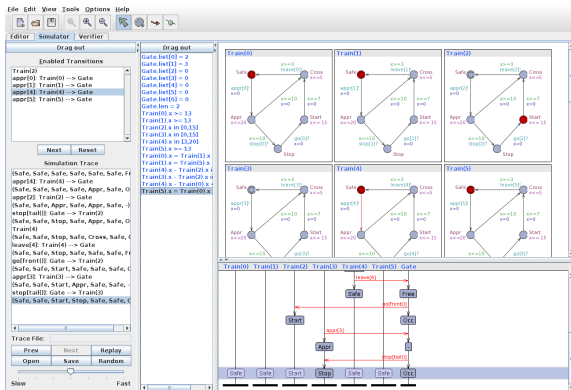


## Theorem (AD90)

Reachability (and  $\omega$ -regular properties) in timed automata can be checked in exponential time (and are PSPACE-complete).

# Analysing timed automata in practice

- symbolic algorithms (using zones)
- efficient implementations (Uppaal, Kronos, ...)



# Outline of the presentation

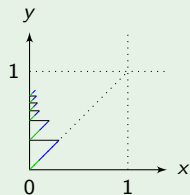
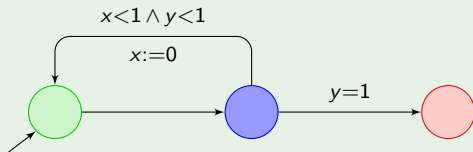
- 1 Introduction – Timed automata
- 2 Robustness issues in timed automata
- 3 Several approaches
  - Tube semantics
  - Probabilistic semantics
  - Sampled semantics
- 4 Enlarged semantics
  - A different approach
  - Checking robustness against enlargement
  - Making timed automata robust
- 5 Conclusions and perspectives

# Outline of the presentation

- 1 Introduction – Timed automata
- 2 Robustness issues in timed automata
- 3 Several approaches
  - Tube semantics
  - Probabilistic semantics
  - Sampled semantics
- 4 Enlarged semantics
  - A different approach
  - Checking robustness against enlargement
  - Making timed automata robust
- 5 Conclusions and perspectives

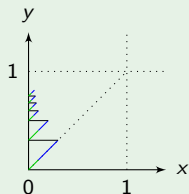
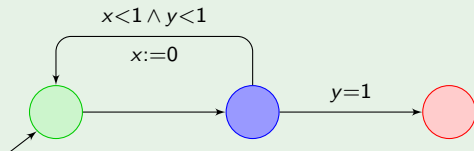
# Robustness issues in timed automata

## Zeno behaviours



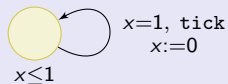
# Robustness issues in timed automata

## Zeno behaviours



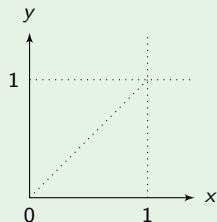
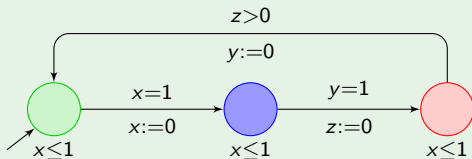
## Theorem (AD90)

Checking  $\omega$ -regular properties under non-Zenoness requirement can be done in exponential time.



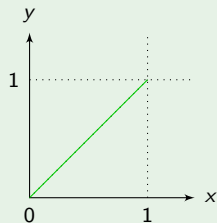
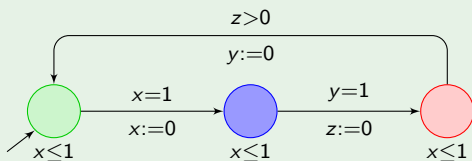
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



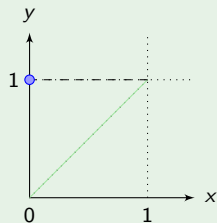
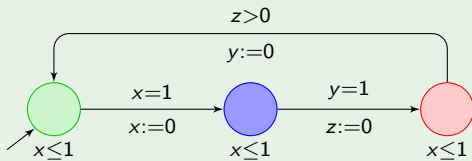
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



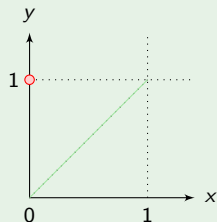
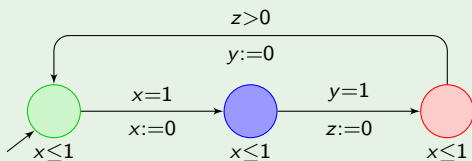
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



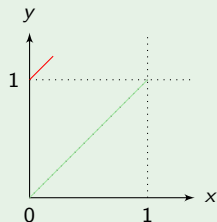
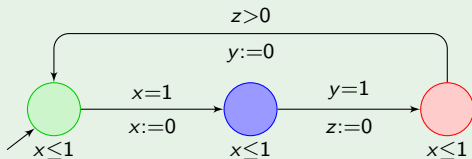
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



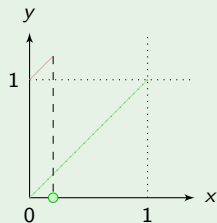
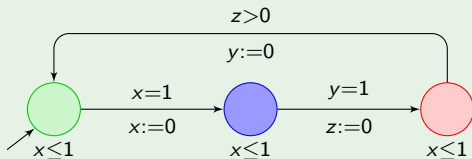
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



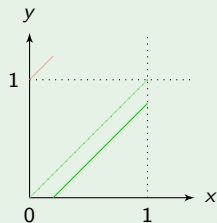
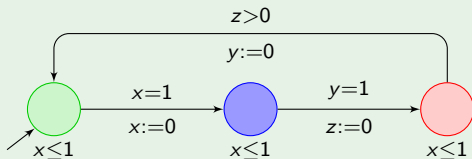
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



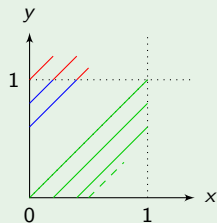
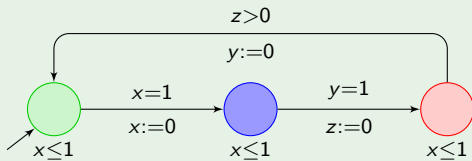
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



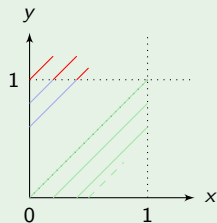
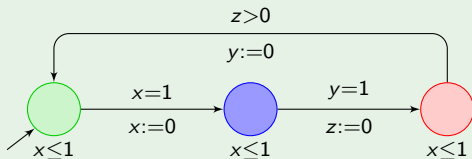
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



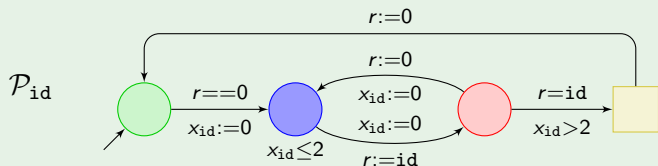
# Robustness issues in timed automata

## Convergence phenomena (CHR02)



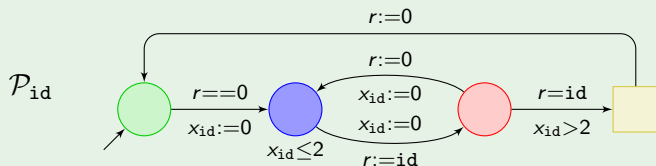
# Robustness issues in timed automata

## Strict timing constraints



# Robustness issues in timed automata

## Strict timing constraints

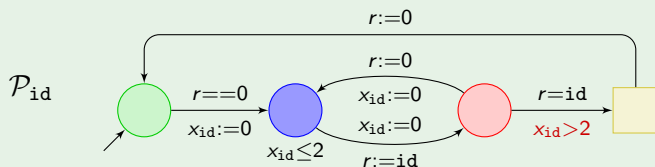


### Theorem (KLL<sup>+</sup>97)

When  $P_1$  and  $P_2$  run in parallel (sharing variable  $r$ ), the state where both of them are in  $\square$  is not reachable.

# Robustness issues in timed automata

## Strict timing constraints



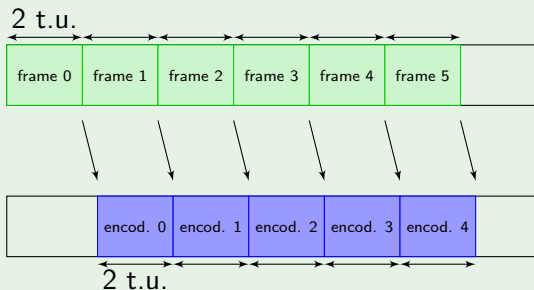
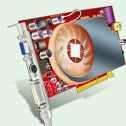
### Theorem (KLL<sup>+</sup>97)

When  $P_1$  and  $P_2$  run in parallel (sharing variable  $r$ ), the state where both of them are in  $\square$  is not reachable.

But this property is lost when  $x_{id} > 2$  is replaced with  $x_{id} \geq 2$ .

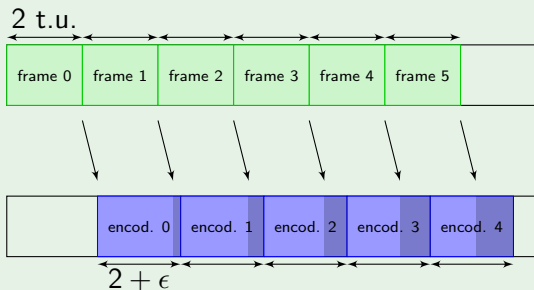
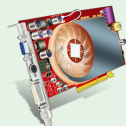
# Robustness issues in timed automata

## Imprecision on clock values (ACS10)



# Robustness issues in timed automata

## Imprecision on clock values (ACS10)



# Outline of the presentation

- 1 Introduction – Timed automata
- 2 Robustness issues in timed automata
- 3 Several approaches
  - Tube semantics
  - Probabilistic semantics
  - Sampled semantics
- 4 Enlarged semantics
  - A different approach
  - Checking robustness against enlargement
  - Making timed automata robust
- 5 Conclusions and perspectives

## Several solutions have been proposed...

### Tube semantics (GHJ97)

- discards behaviours that have too strict constraints;
- only consider traces whose neighbouring traces are accepted;
- safety is decidable.



## Several solutions have been proposed...

### Tube semantics (GHJ97)

- discards behaviours that have too strict constraints;
- only consider traces whose neighbouring traces are accepted;
- safety is decidable.



### Probabilistic semantics (BBBB07)

- defines a measure on traces;
- discards **unlikely behaviours**;
- safety is decidable.



## Several solutions have been proposed...

### Sampled semantics (HMP92,AKY10)

- actions are taken only at integer multiples of  $\tau$ ;
- conceptually simpler to handle, but checking safety still takes exponential time;

## Several solutions have been proposed...

### Sampled semantics (HMP92,AKY10)

- actions are taken only at integer multiples of  $\tau$ ;
- conceptually simpler to handle, but checking safety still takes exponential time;

### Samplability

A timed automaton  $\mathcal{A}$  is **samplable** if there exists  $\tau > 0$  s.t.  $\mathcal{A}$  exhibits similar (untimed) behaviours under the classical semantics as under the  $\tau$ -sampled semantics.

## Several solutions have been proposed...

### Sampled semantics (HMP92,AKY10)

- actions are taken only at integer multiples of  $\tau$ ;
- conceptually simpler to handle, but checking safety still takes exponential time;

### Samplability

A timed automaton  $\mathcal{A}$  is **samplable** if there exists  $\tau > 0$  s.t.  $\mathcal{A}$  exhibits similar (untimed) behaviours under the classical semantics as under the  $\tau$ -sampled semantics.

### Theorem (AKY10)

Samplability is decidable.

# Outline of the presentation

- 1 Introduction – Timed automata
- 2 Robustness issues in timed automata
- 3 Several approaches
  - Tube semantics
  - Probabilistic semantics
  - Sampled semantics
- 4 **Enlarged semantics**
  - A different approach
  - Checking robustness against enlargement
  - Making timed automata robust
- 5 Conclusions and perspectives

## A different solution...

### Enlarged semantics (Pur98)

- clocks evolve at rate in  $[1 - \epsilon, 1 + \epsilon]$  instead of exactly 1;
- clock constraints  $x \in [a, b]$  replaced with  $x \in [a - \delta, b + \delta]$ ;
- contrary to the other approaches, **this semantics adds extra behaviours**, considering that the classical semantics is too precise.

## A different solution...

### Enlarged semantics (Pur98)

- clocks evolve at rate in  $[1 - \epsilon, 1 + \epsilon]$  instead of exactly 1;
- clock constraints  $x \in [a, b]$  replaced with  $x \in [a - \delta, b + \delta]$ ;
- contrary to the other approaches, **this semantics adds extra behaviours**, considering that the classical semantics is too precise.

### Robustness

A timed automaton  $\mathcal{A}$  is **robust** if there exist  $\epsilon > 0$  and/or  $\delta > 0$  s.t.  $\mathcal{A}$  exhibits similar (untimed) behaviours under the classical semantics as under the enlarged semantics.

## A different solution...

### Enlarged semantics (Pur98)

- clocks evolve at rate in  $[1 - \epsilon, 1 + \epsilon]$  instead of exactly 1;
- clock constraints  $x \in [a, b]$  replaced with  $x \in [a - \delta, b + \delta]$ ;
- contrary to the other approaches, **this semantics adds extra behaviours**, considering that the classical semantics is too precise.

### Robustness

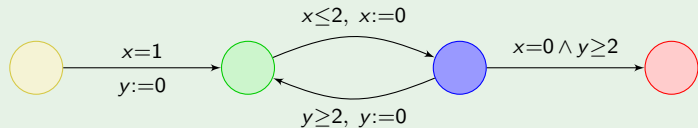
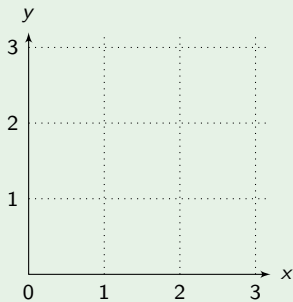
A timed automaton  $\mathcal{A}$  is **robust** if there exist  $\epsilon > 0$  and/or  $\delta > 0$  s.t.  $\mathcal{A}$  exhibits similar (untimed) behaviours under the classical semantics as under the enlarged semantics.

### Theorem (Pur98,DDMR04,BMR06,San11)

Robustness is decidable.

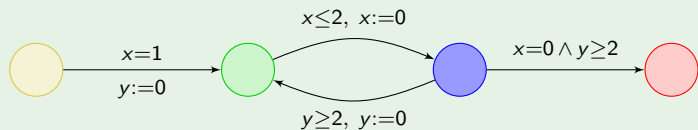
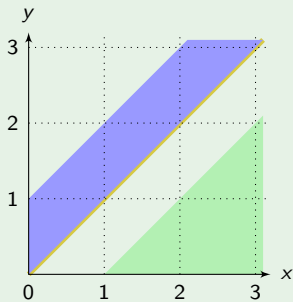
# What happens under the (guard-)enlarged semantics?

## Example



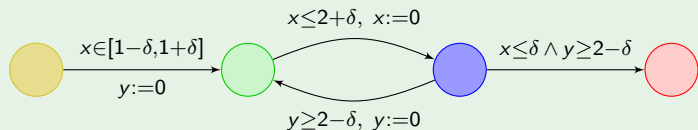
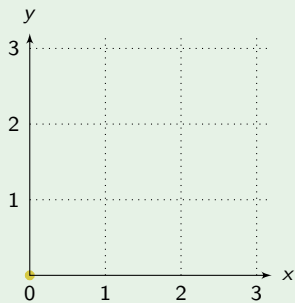
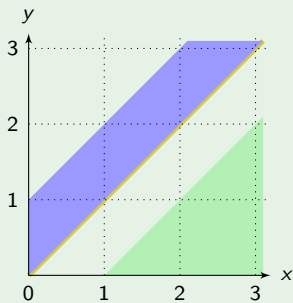
# What happens under the (guard-)enlarged semantics?

## Example



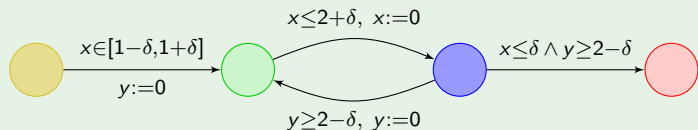
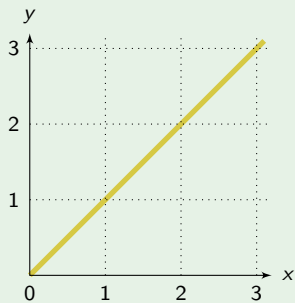
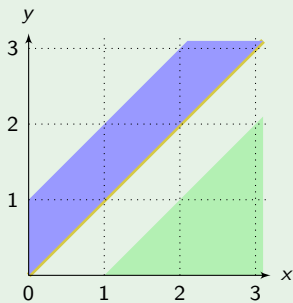
# What happens under the (guard-)enlarged semantics?

## Example



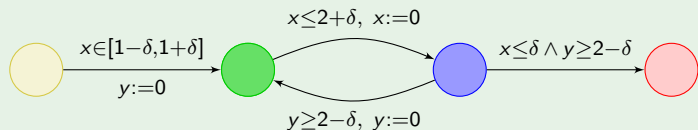
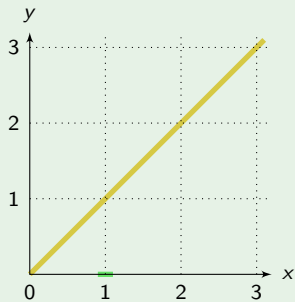
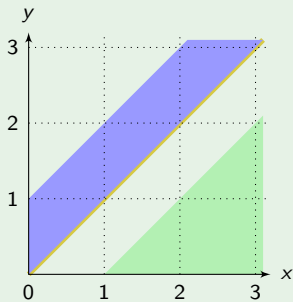
# What happens under the (guard-)enlarged semantics?

## Example



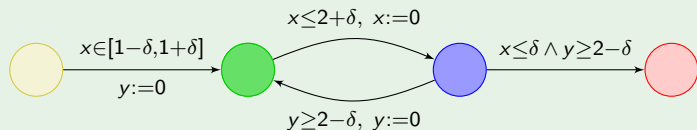
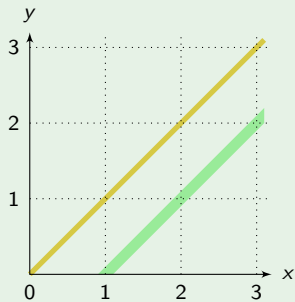
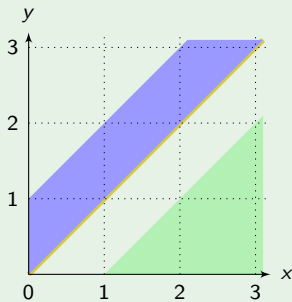
# What happens under the (guard-)enlarged semantics?

## Example



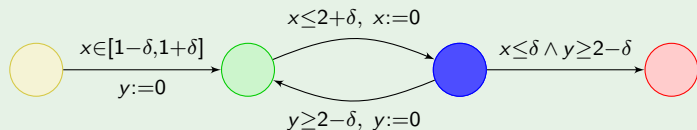
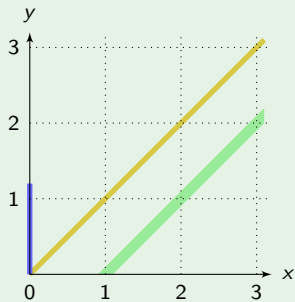
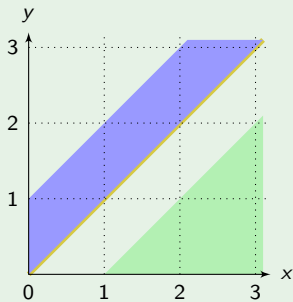
# What happens under the (guard-)enlarged semantics?

## Example



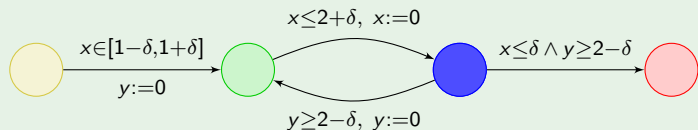
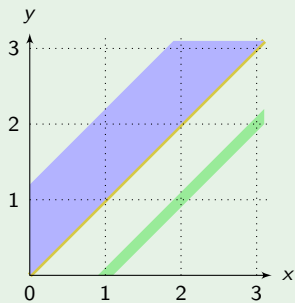
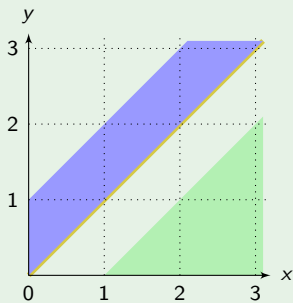
# What happens under the (guard-)enlarged semantics?

## Example



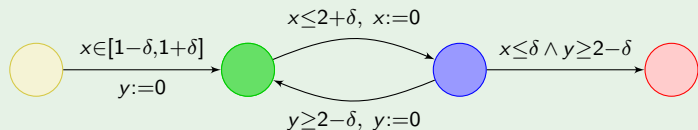
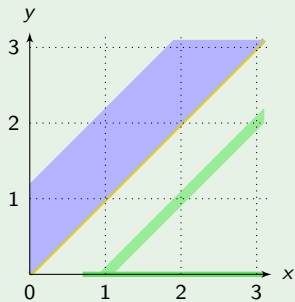
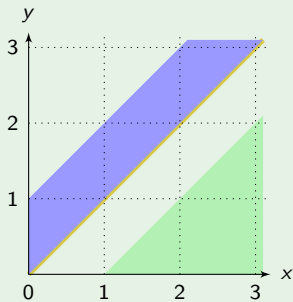
# What happens under the (guard-)enlarged semantics?

## Example



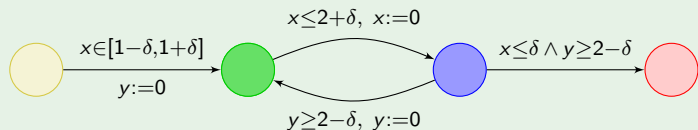
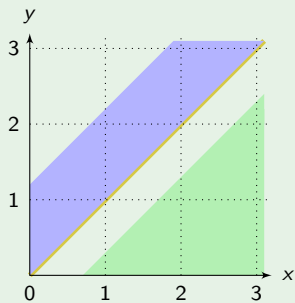
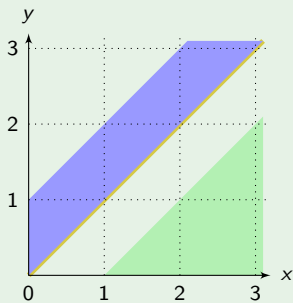
# What happens under the (guard-)enlarged semantics?

## Example



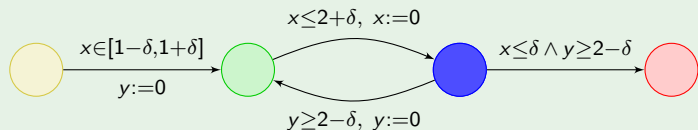
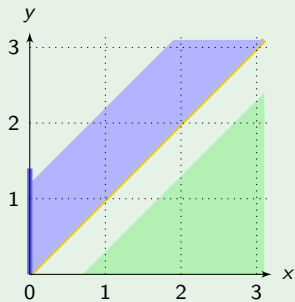
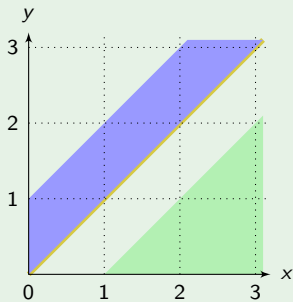
# What happens under the (guard-)enlarged semantics?

## Example



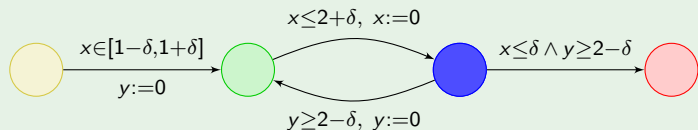
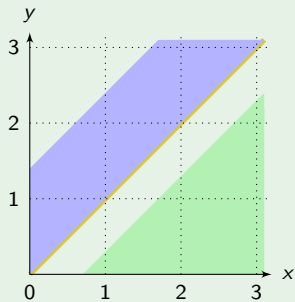
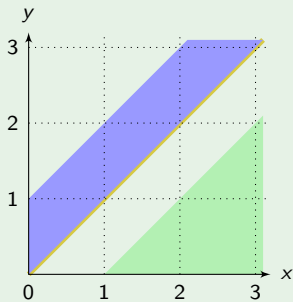
# What happens under the (guard-)enlarged semantics?

## Example



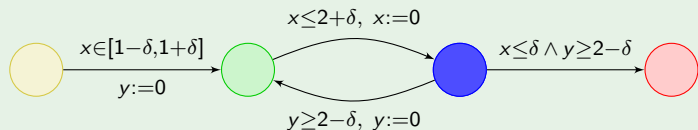
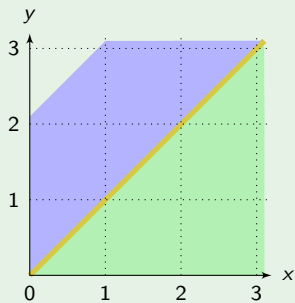
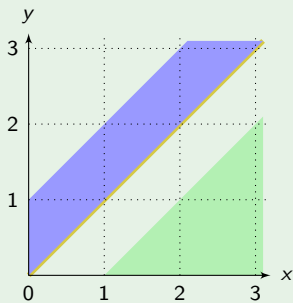
# What happens under the (guard-)enlarged semantics?

## Example



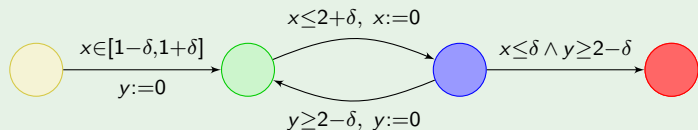
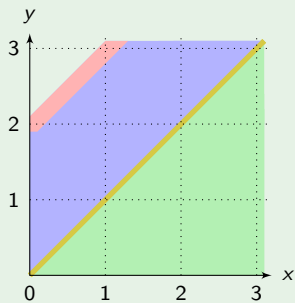
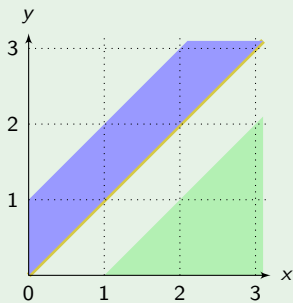
# What happens under the (guard-)enlarged semantics?

## Example



# What happens under the (guard-)enlarged semantics?

## Example



# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r')$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .

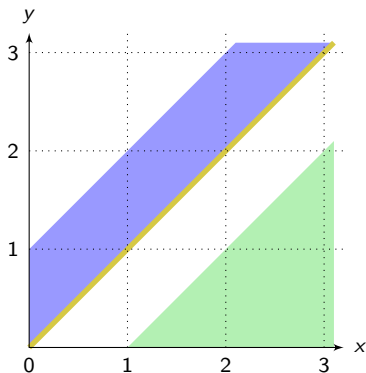
# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r)$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .



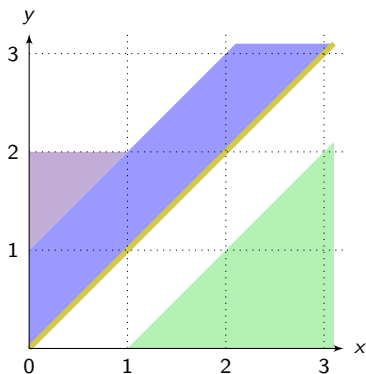
# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r')$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .



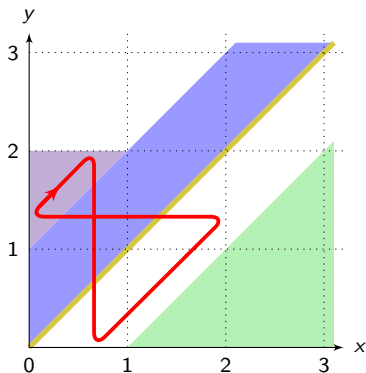
# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r)$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .



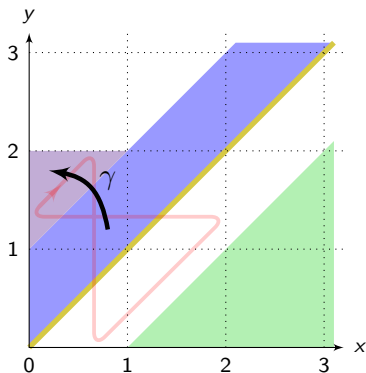
# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r)$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .



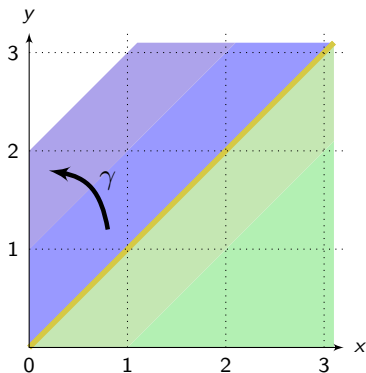
# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r)$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .



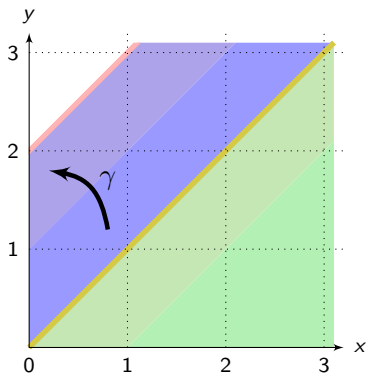
# Safety checking under the enlarged semantics

## Extended region automaton

For any location  $\ell$  and any two regions  $r$  and  $r'$ , if

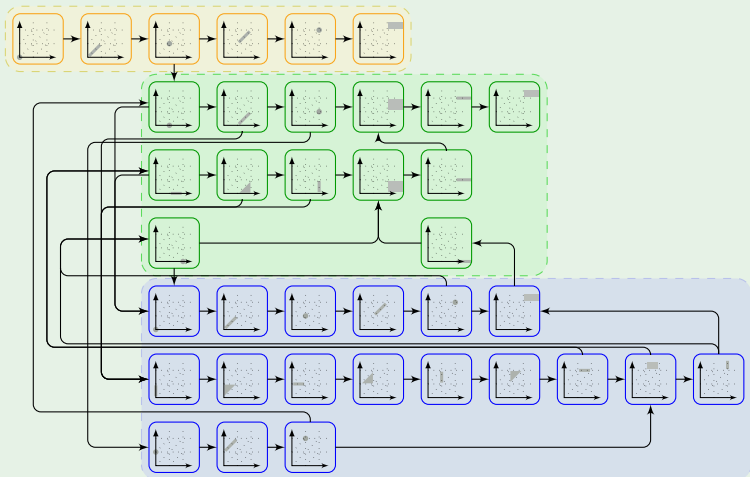
- $\bar{r} \cap \bar{r}' \neq \emptyset$  and
- $(\ell, r)$  belongs to an SCC of  $\mathcal{R}(\mathcal{A})$ ,

then we add a transition  $(\ell, r) \xrightarrow{\gamma} (\ell, r')$ .



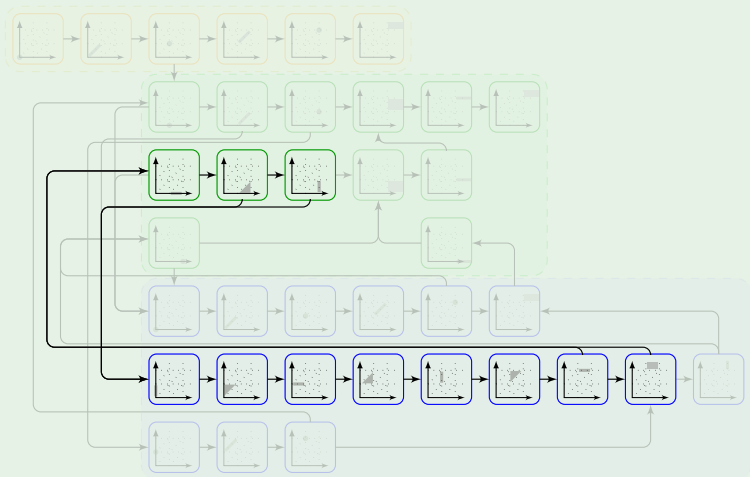
# Safety checking under the enlarged semantics

## Example



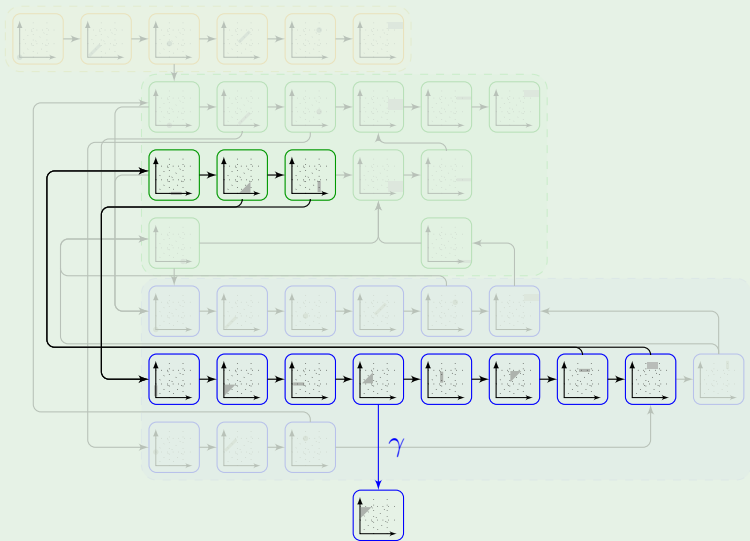
# Safety checking under the enlarged semantics

## Example



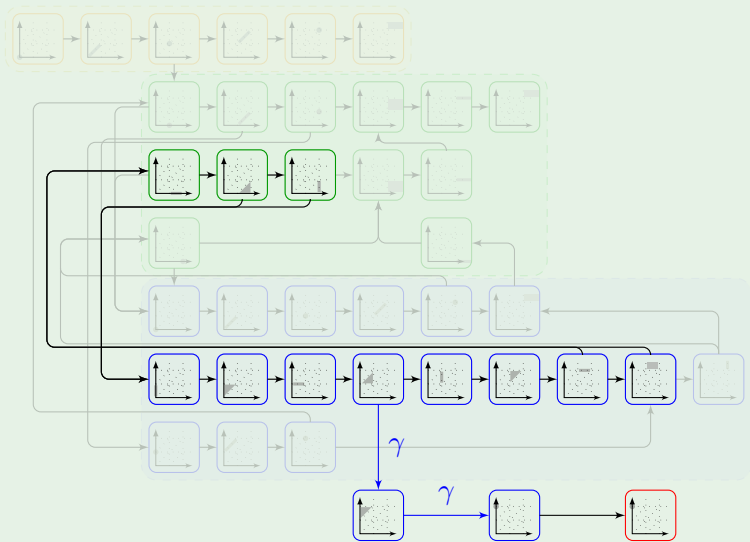
# Safety checking under the enlarged semantics

## Example



# Safety checking under the enlarged semantics

## Example



# Safety checking under the enlarged semantics

## Lemma

The set of reachable regions in the **extended region automaton** is exactly

$$\bigcap_{\delta > 0} \text{Reach}(\mathcal{A}_\delta).$$

(under some technical restrictions)

# Safety checking under the enlarged semantics

## Lemma

The set of reachable regions in the **extended region automaton** is exactly

$$\bigcap_{\delta > 0} \text{Reach}(\mathcal{A}_\delta).$$

(under some technical restrictions)

## Lemma

For any timed automata  $\mathcal{A}$  and for any region  $B$ ,

$$\bigcap_{\delta > 0} \text{Reach}_\delta(\mathcal{A}) \cap B = \emptyset \quad \text{iff} \quad \exists \delta > 0. \text{Reach}_\delta(\mathcal{A}) \cap B = \emptyset.$$

# Safety checking under the enlarged semantics

## Lemma

The set of reachable regions in the **extended region automaton** is exactly

$$\bigcap_{\delta > 0} \text{Reach}(\mathcal{A}_\delta).$$

(under some technical restrictions)

## Lemma

For any timed automata  $\mathcal{A}$  and for any region  $B$ ,

$$\bigcap_{\delta > 0} \text{Reach}_\delta(\mathcal{A}) \cap B = \emptyset \quad \text{iff} \quad \exists \delta > 0. \text{Reach}_\delta(\mathcal{A}) \cap B = \emptyset.$$

## Theorem

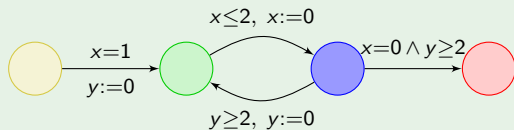
Robust safety in timed automata is decidable in exponential time (and is PSPACE-complete).

# Making timed automata robust

# Making timed automata robust

## Example

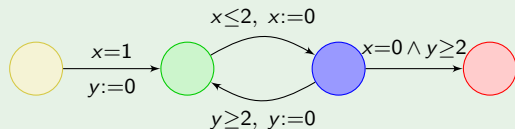
This automaton is not robust:



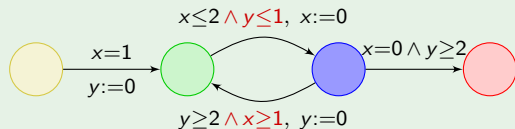
# Making timed automata robust

## Example

This automaton is not robust:



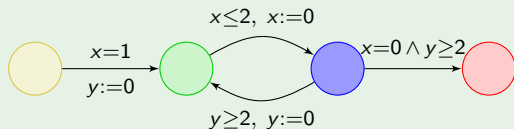
But this one is:



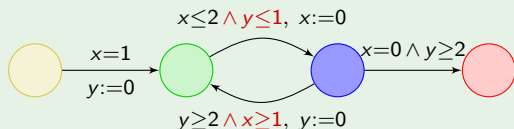
# Making timed automata robust

## Example

This automaton is not robust:



But this one is:



Robustness is a **syntactic** criterion.

# Making timed automata robust

## $\epsilon$ -bisimilarity

$\sim \subseteq S \times S$  is an  $\epsilon$ -bisimulation if

$$s \xrightarrow{a} t$$

$\wr$

$s'$

action transitions

# Making timed automata robust

## $\epsilon$ -bisimilarity

$\sim \subseteq S \times S$  is an  $\epsilon$ -bisimulation if

$$s \xrightarrow{a} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{a} t'$$

action transitions

# Making timed automata robust

## $\epsilon$ -bisimilarity

$\sim \subseteq S \times S$  is an  $\epsilon$ -bisimulation if

$$s \xrightarrow{a} t$$

$\wr$   $\wr$

$$s' \xrightarrow{a} t'$$

action transitions

$$s \xrightarrow{d} t$$

$\wr$

$s'$

delay transitions

# Making timed automata robust

## $\epsilon$ -bisimilarity

$\sim \subseteq S \times S$  is an  $\epsilon$ -bisimulation if

$$s \xrightarrow{a} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{a} t'$$

action transitions

$$s \xrightarrow{d} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{d'} t'$$

delay transitions

$$|d' - d| \leq \epsilon$$

# Making timed automata robust

## $\epsilon$ -bisimilarity

$\sim \subseteq S \times S$  is an  $\epsilon$ -bisimulation if

$$s \xrightarrow{a} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{a} t'$$

action transitions

$$s \xrightarrow{d} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{d'} t'$$

delay transitions

$$|d' - d| \leq \epsilon$$

## Quantitative notion of robustness

A timed automaton  $\mathcal{A}$  is  $\epsilon$ -robust if there exists  $\delta > 0$  s.t.  $\mathcal{A}$  and its  $\delta$ -enlarged semantics  $\mathcal{A}_\delta$  are  $\epsilon$ -bisimilar.

# Making timed automata robust

## $\epsilon$ -bisimilarity

$\sim \subseteq S \times S$  is an  $\epsilon$ -bisimulation if

$$s \xrightarrow{a} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{a} t'$$

action transitions

$$s \xrightarrow{d} t$$

$\wr \qquad \qquad \wr$

$$s' \xrightarrow{d'} t'$$

delay transitions

$$|d' - d| \leq \epsilon$$

## Theorem (BFL<sup>+</sup>11)

Given a timed automaton  $\mathcal{A}$  and  $\epsilon > 0$ , we can build a timed automaton  $\mathcal{A}'$  s.t.

- $\mathcal{A}$  and  $\mathcal{A}'$  are 0-bisimilar;
- $\mathcal{A}'$  is  $\epsilon$ -robust.

# Outline of the presentation

- 1 Introduction – Timed automata
- 2 Robustness issues in timed automata
- 3 Several approaches
  - Tube semantics
  - Probabilistic semantics
  - Sampled semantics
- 4 Enlarged semantics
  - A different approach
  - Checking robustness against enlargement
  - Making timed automata robust
- 5 Conclusions and perspectives

# Conclusions and perspectives

## Robustness is an important issue in timed systems

- timed automata are governed by a mathematical semantics;
- this raises important **robustness issues**:
  - time-convergent behaviours;
  - strict timing constraints...
- several approaches:
  - ignoring **isolated** traces;
  - considering **surrounding** runs.

# Conclusions and perspectives

## Robustness is an important issue in timed systems

- timed automata are governed by a mathematical semantics;
- this raises important **robustness issues**:
  - time-convergent behaviours;
  - strict timing constraints...
- several approaches:
  - ignoring **isolated** traces;
  - considering **surrounding** runs.

## Perspectives

- develop the **quantitative approach** to robustness;
- **probabilistic** (as opposed to worst-case) enlargement;
- **shrinking** timed automata (to counteract enlargement);
- robust controller synthesis;
- robustness in **priced timed automata** (with energy constraints).