Efficient timed diagnosis
using automata with timed domains

Patricia Bouyer Samy Jaziri
CNRS – LSV (ENS Paris-Saclay, France)

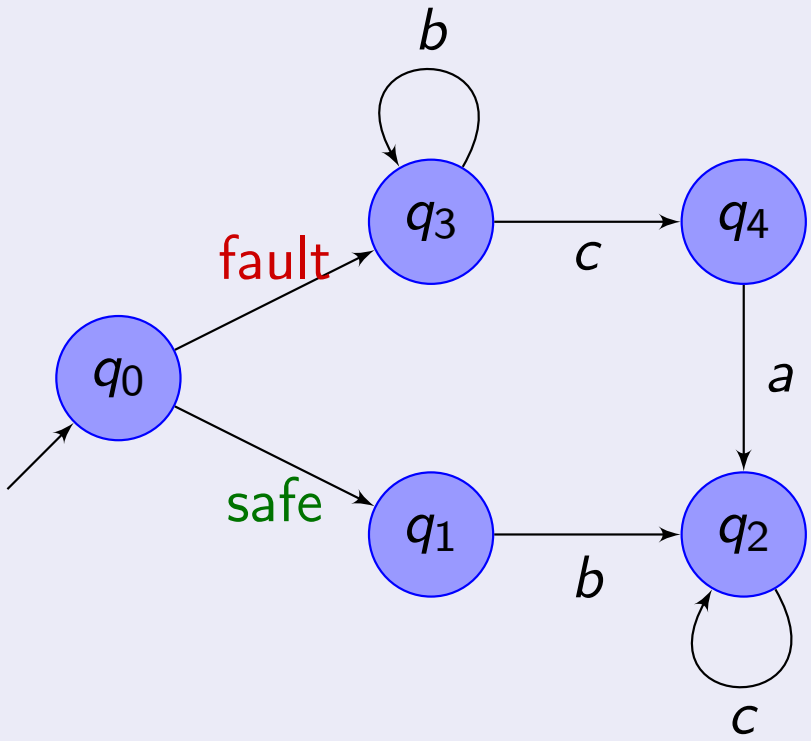
Nicolas Markey
CNRS – IRISA (Univ. Rennes, France)

RV 2018 – November 11, 2018

EQualIS

Fault diagnosis [SSL+95]

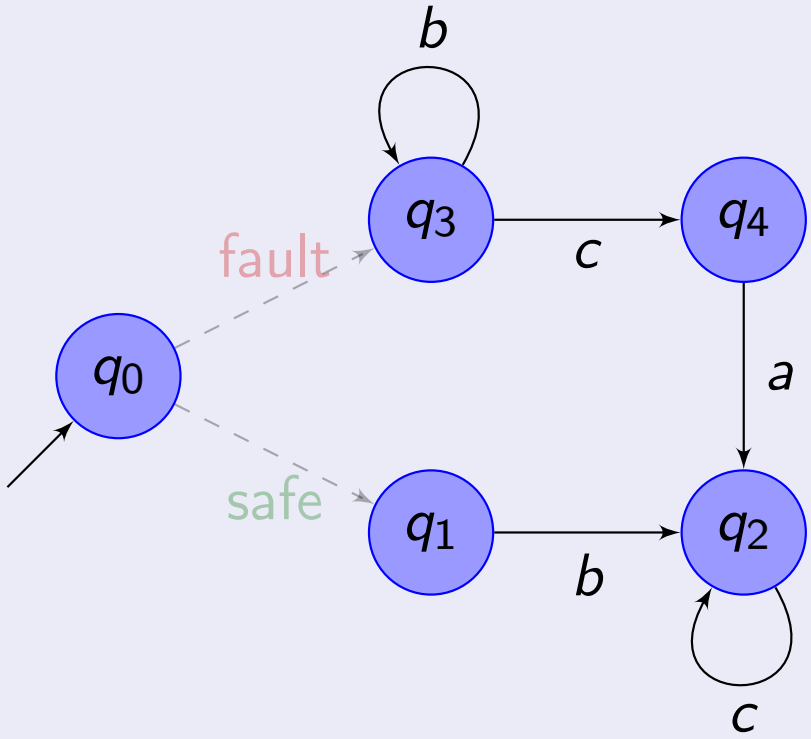
Partially-observable systems with faults



[SSL+95] Sampath *et al.* Diagnosability of discrete-event systems. TAC, 1995.

Fault diagnosis [SSL+95]

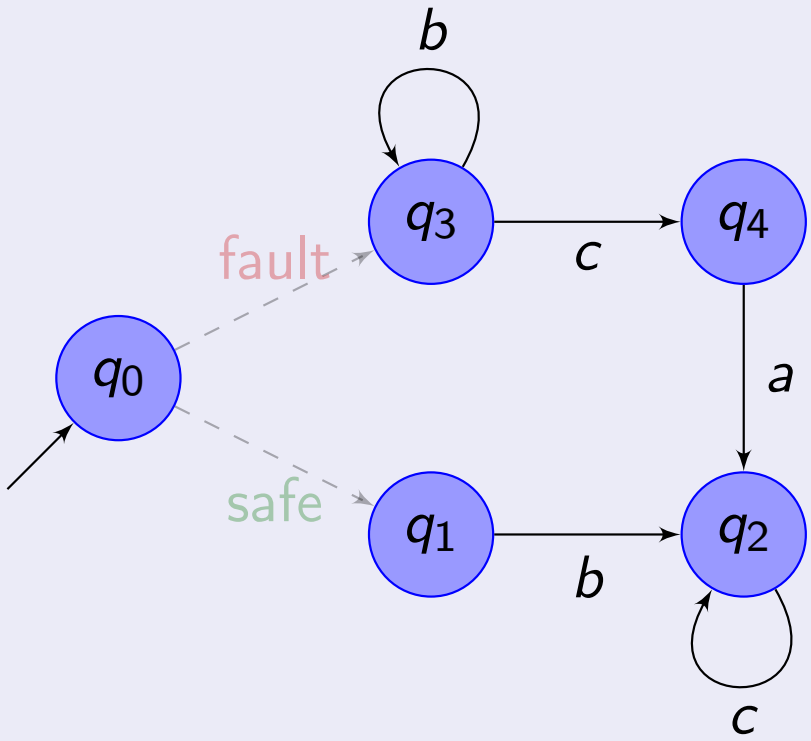
Partially-observable systems with faults



[SSL+95] Sampath *et al.* Diagnosability of discrete-event systems. TAC, 1995.

Fault diagnosis [SSL+95]

Partially-observable systems with faults

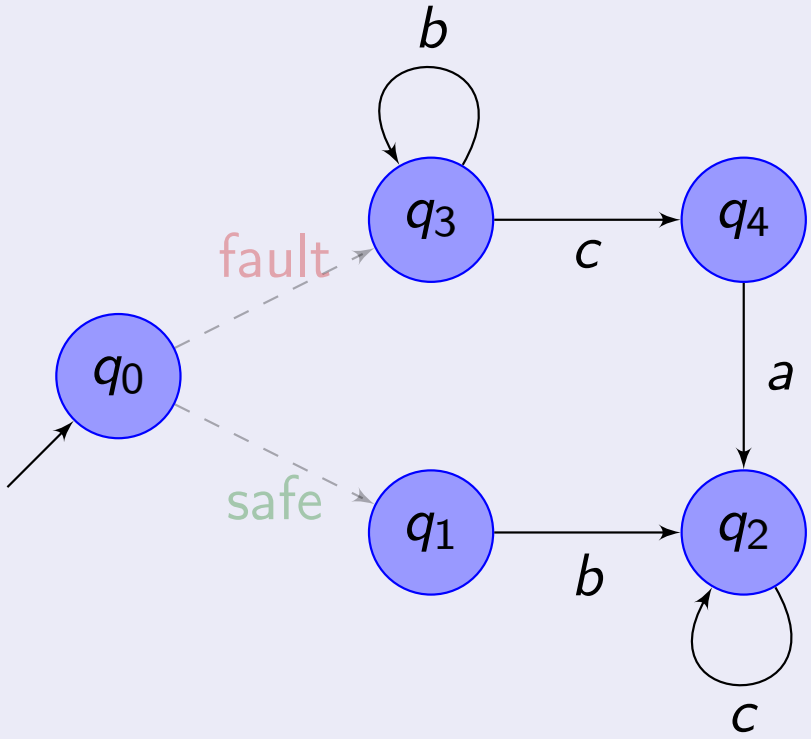


- observation *bca* corresponds to a faulty run;
- observation *bcc* corresponds to a safe run;
- observation *bc* is ambiguous.

[SSL+95] Sampath *et al.* Diagnosability of discrete-event systems. TAC, 1995.

Fault diagnosis [SSL+95]

Partially-observable systems with faults



Diagnozer

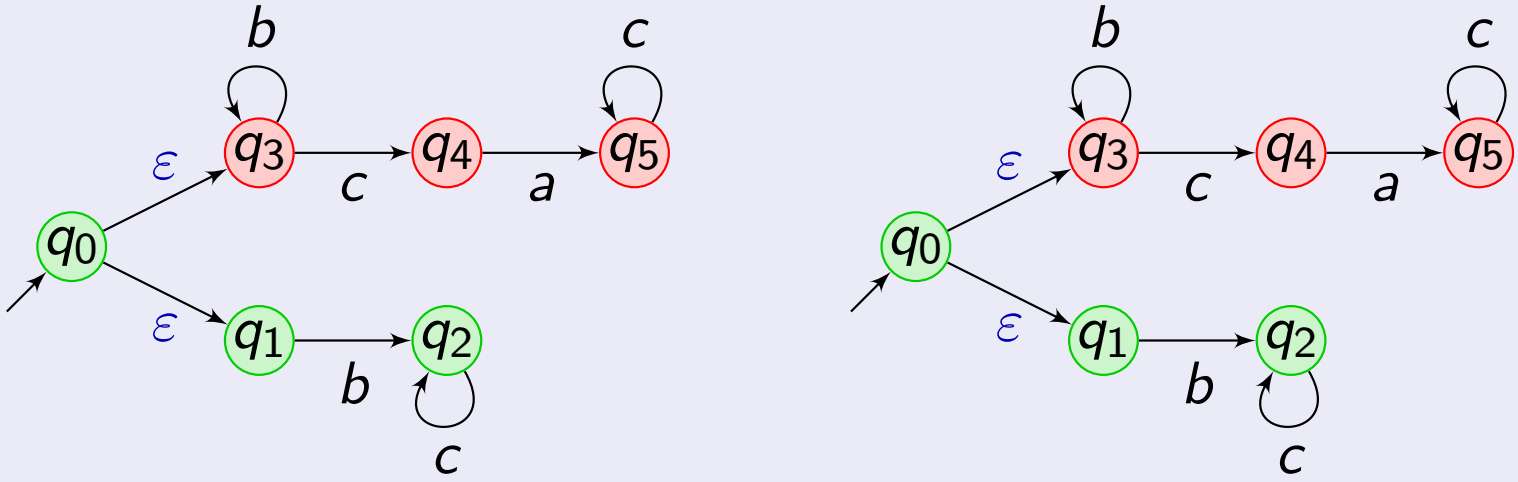
A Δ-diagnozer is a mapping $\Sigma^* \to \{\texttt{safe}, \texttt{fault}\}$ detecting a fault at most Δ steps after one occurred.

[SSL+95] Sampath *et al.* Diagnosability of discrete-event systems. TAC, 1995.

Deciding diagnozability

Diagnozability is decidable

Synchronized product of two (modified) copies of the automaton:

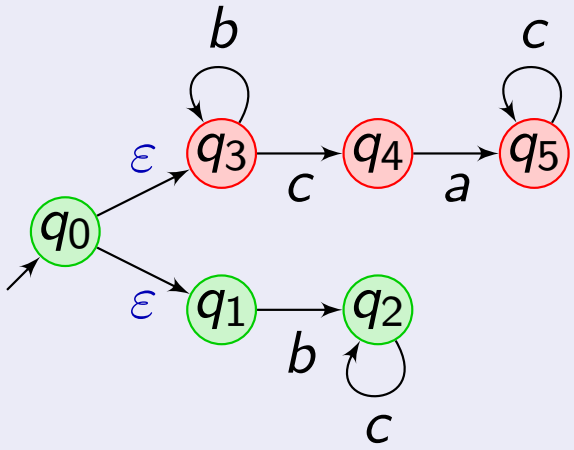


$\mathcal{A}$ is diagnozable iff $\mathcal{A}' \times \mathcal{A}'$ has no infinite ◯ × ◯ run.

Building a diagnozer

Building a diagnozer as a deterministic finite-state automaton

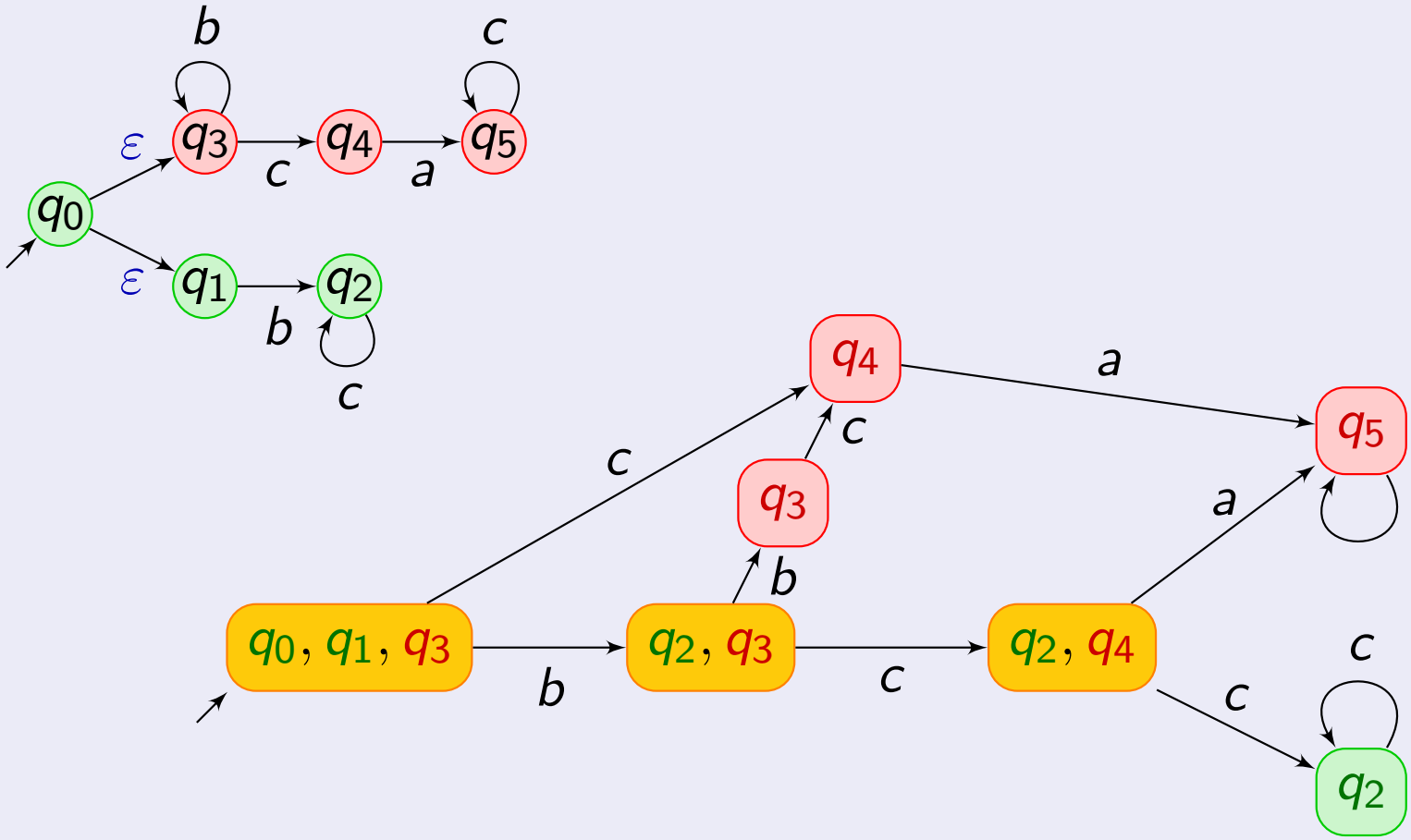
Perform state estimation using a powerset construction:



Building a diagnozer

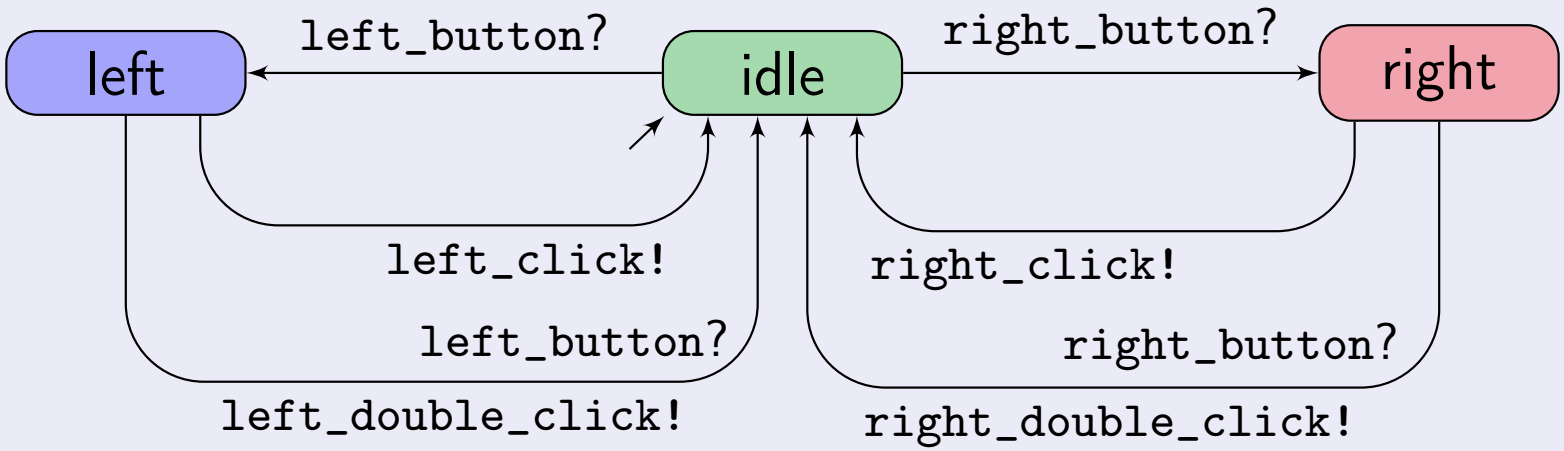
Building a diagnozer as a deterministic finite-state automaton

Perform state estimation using a powerset construction:



Timed automata [AD90]

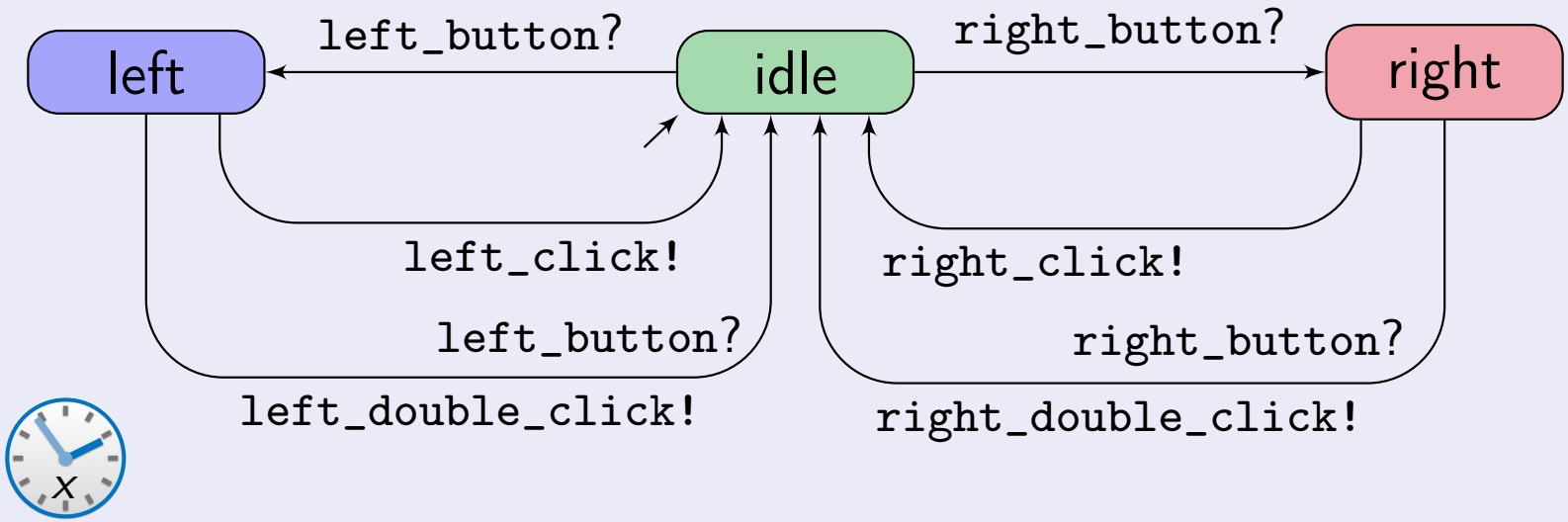
Example of a computer mouse



[AD90] Alur and Dill. Automata For Modeling Real-Time Systems. ICALP 1990.

Timed automata [AD90]

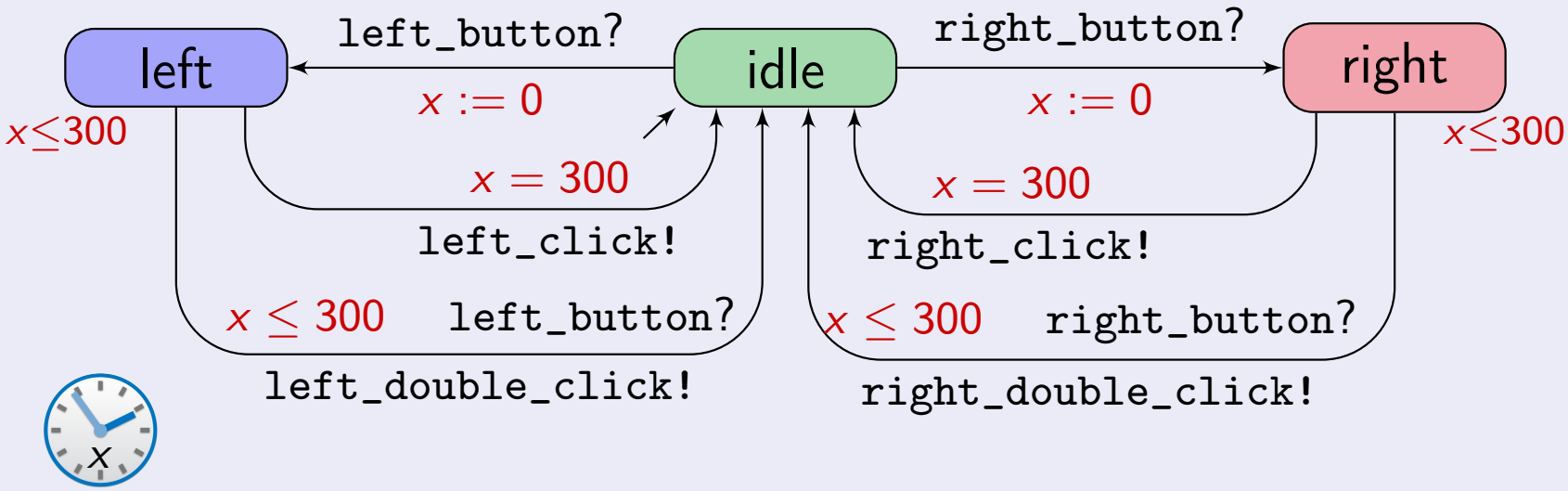
Example of a computer mouse



[AD90] Alur and Dill. Automata For Modeling Real-Time Systems. ICALP 1990.

Timed automata [AD90]

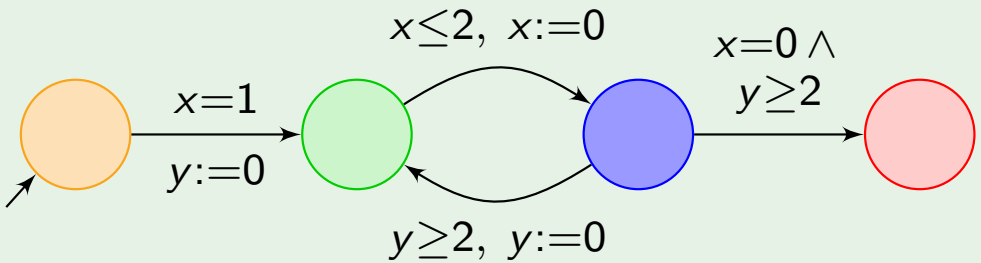
Example of a computer mouse

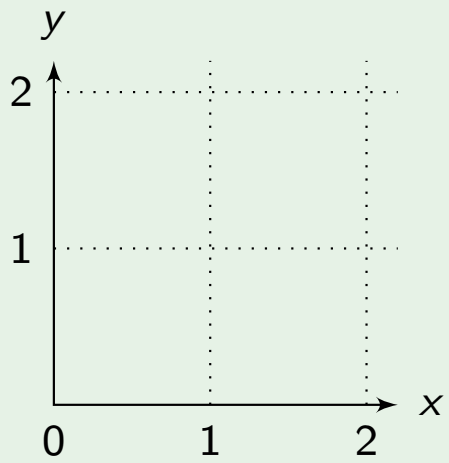


[AD90] Alur and Dill. Automata For Modeling Real-Time Systems. ICALP 1990.

Semantics of timed automata

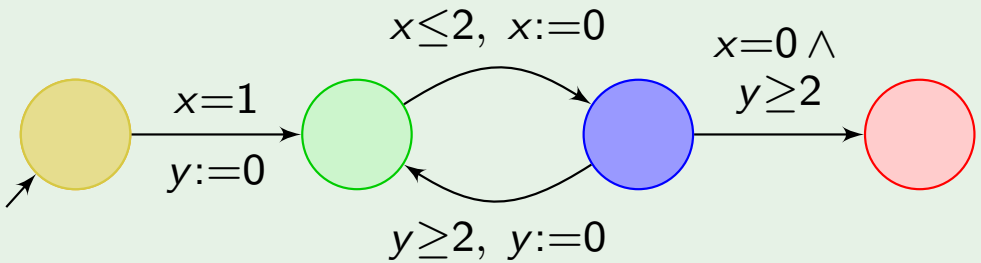
Example

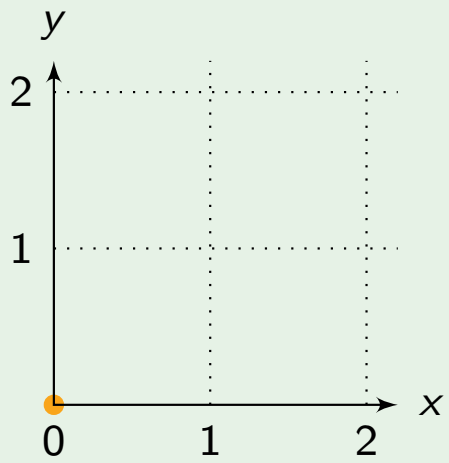




Semantics of timed automata

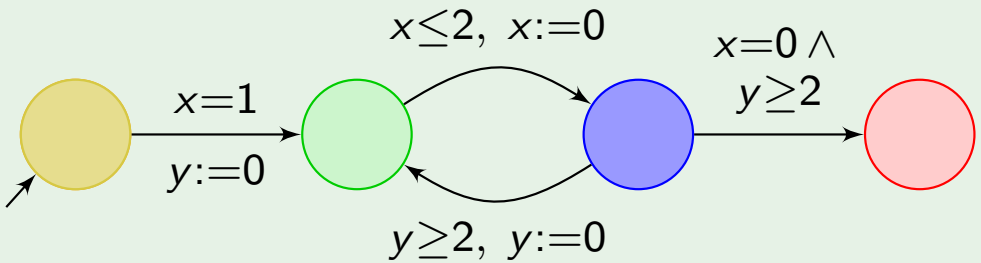
Example

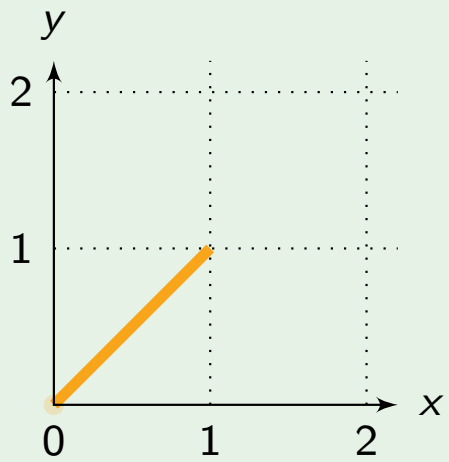




Semantics of timed automata

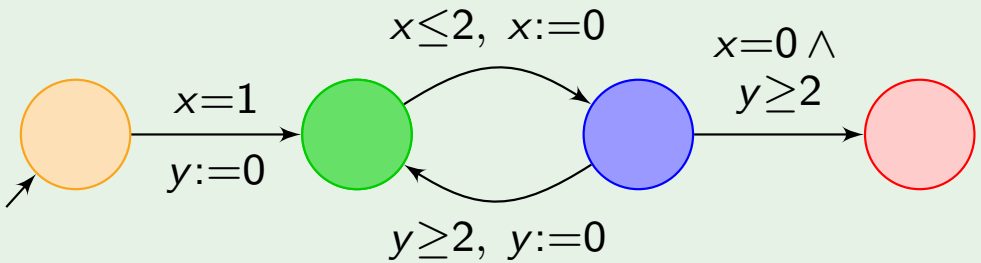
Example

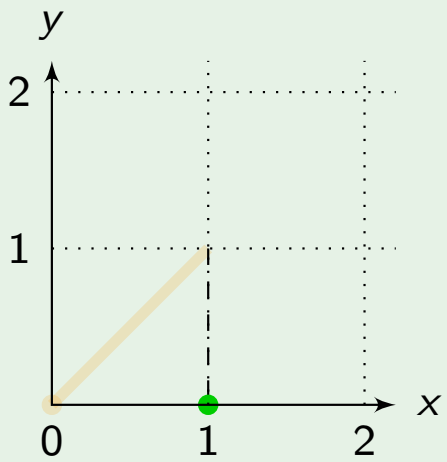




Semantics of timed automata

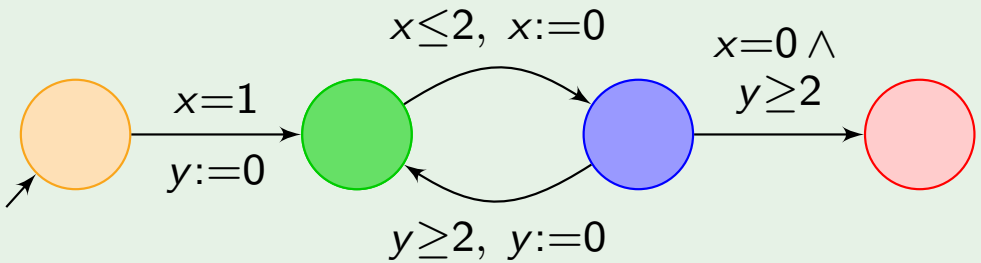
Example

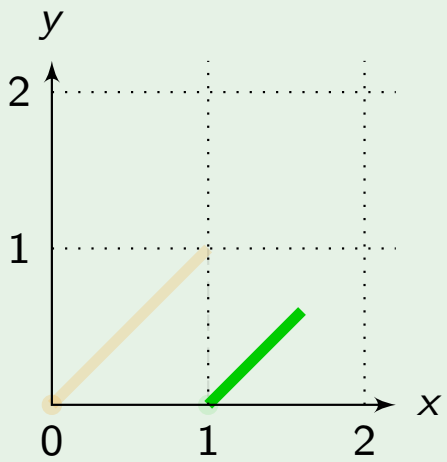




Semantics of timed automata

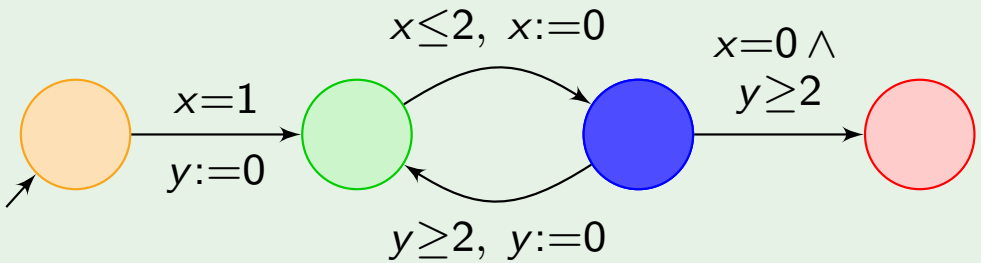
Example

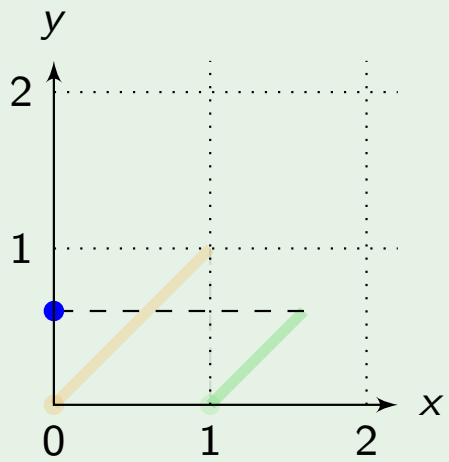




Semantics of timed automata

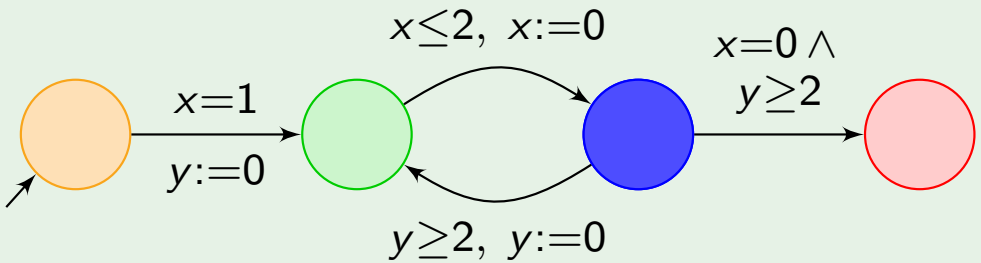
Example

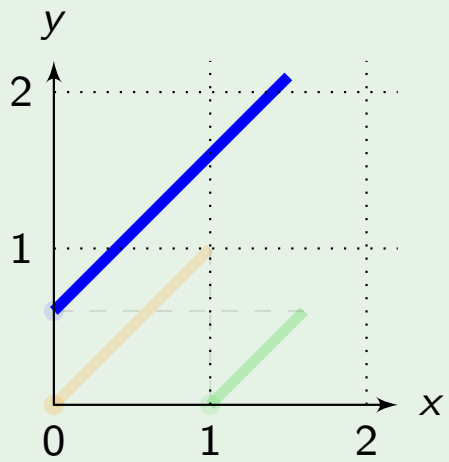




Semantics of timed automata

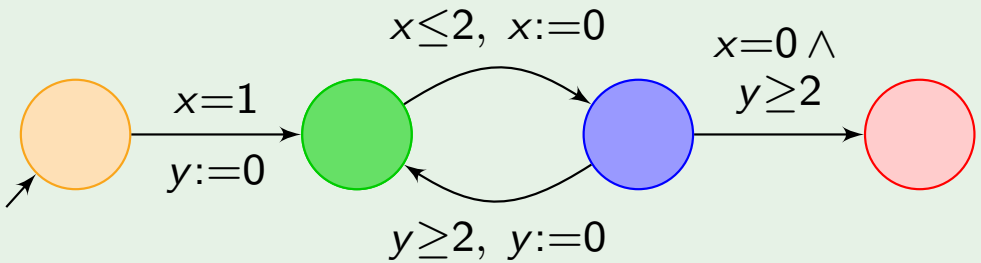
Example

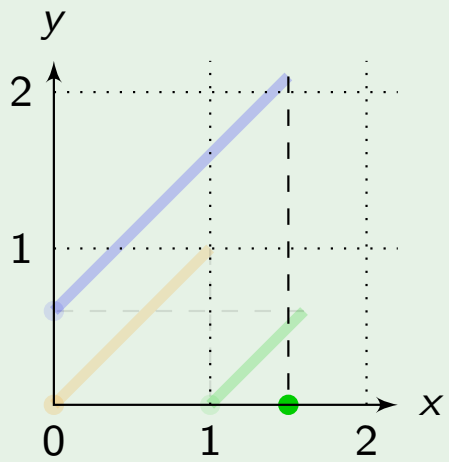




Semantics of timed automata

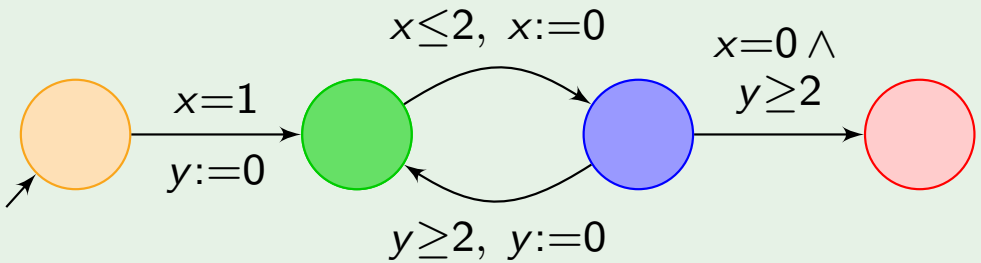
Example

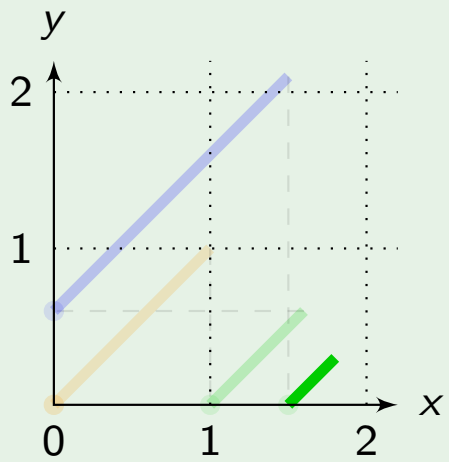




Semantics of timed automata

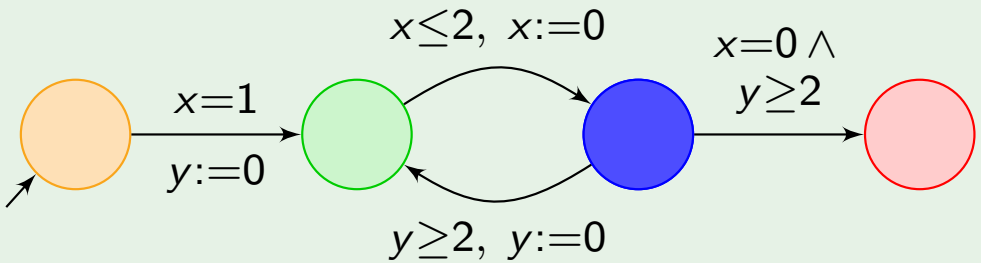
Example

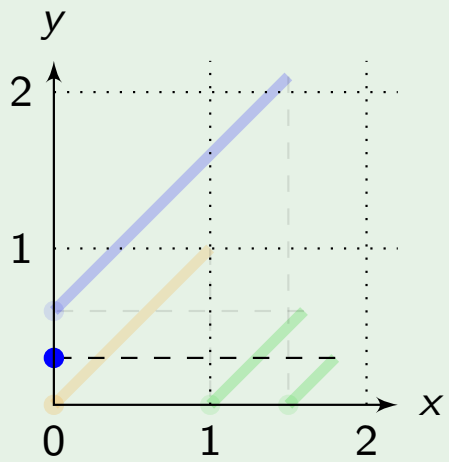




Semantics of timed automata

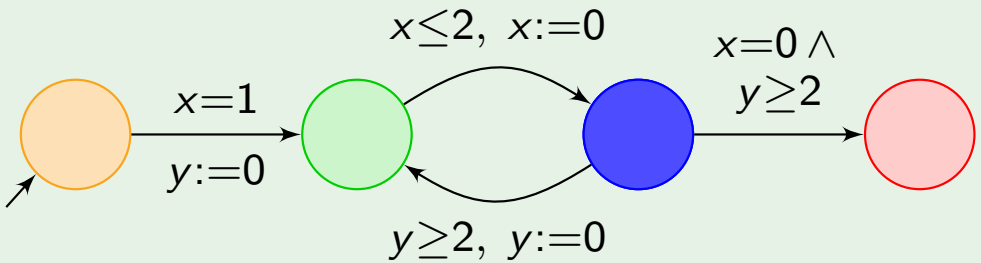
Example

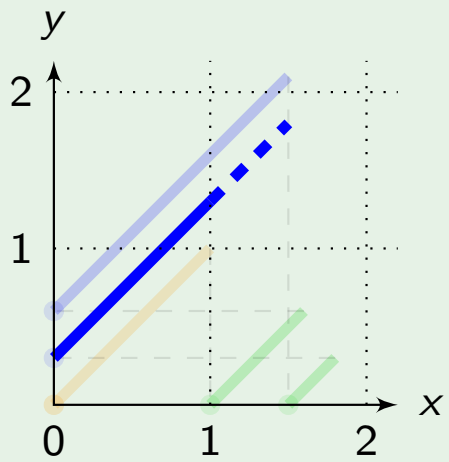




Semantics of timed automata

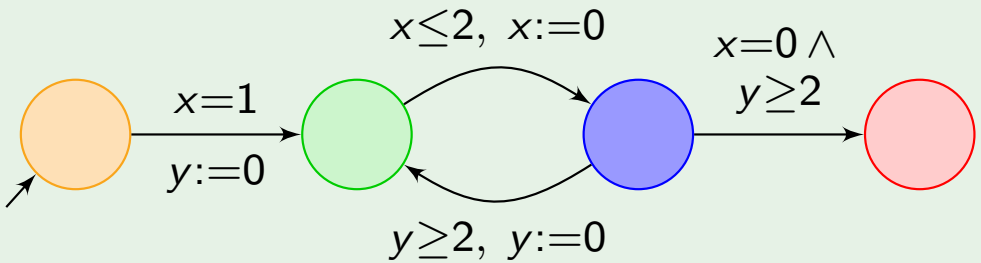
Example

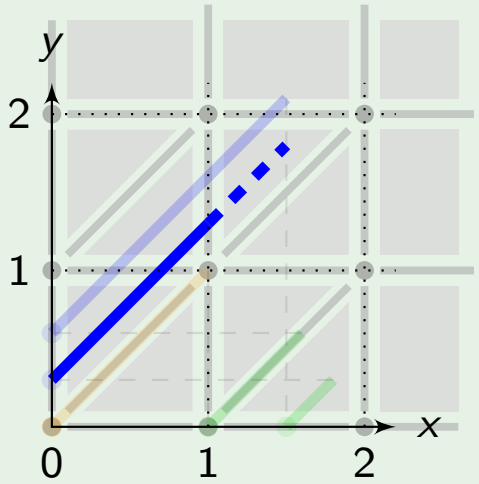




Semantics of timed automata

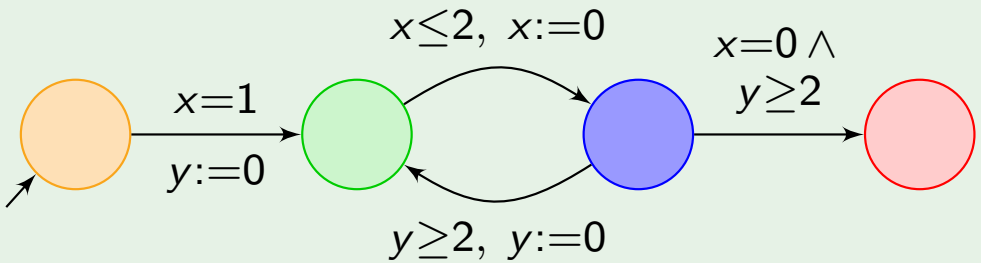
Example

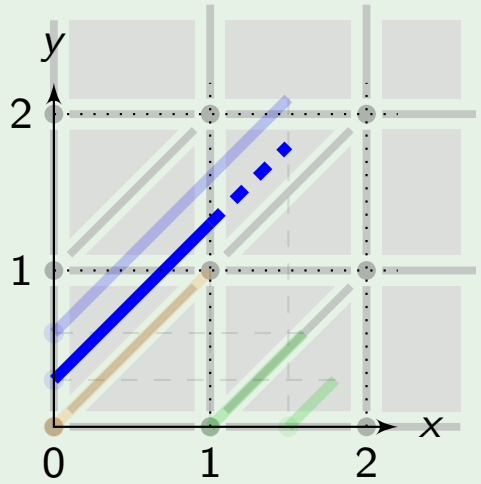




Semantics of timed automata

Example





Theorem ([AD90])

Reachability in timed automata is decidable.

[AD90] Alur and Dill. Automata For Modeling Real-Time Systems. ICALP 1990.

Diagnosability for timed automata

Diagnozer for timed automata

A Δ-diagnozer is a mapping $T\Sigma^* \to \{\texttt{safe}, \texttt{fault}\}$ detecting a fault at most Δ time units after one occurred.

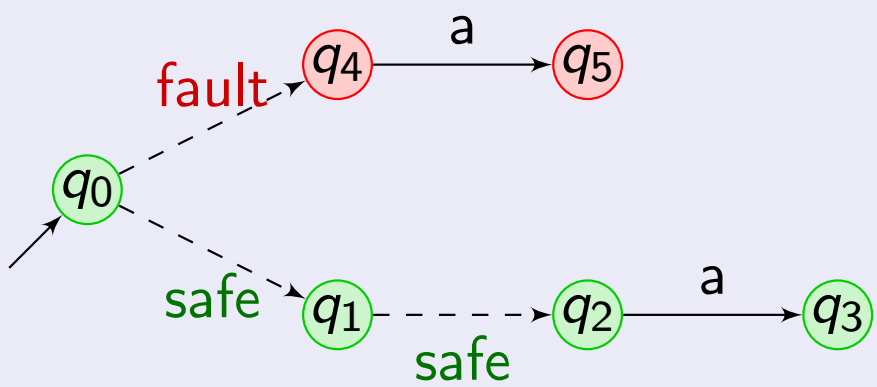
Diagnosability for timed automata

Diagnozer for timed automata

A Δ-diagnozer is a mapping $T\Sigma^* \to \{\texttt{safe}, \texttt{fault}\}$ detecting a fault at most Δ time units after one occurred.

Time can be observed

- even discrete-time makes a difference:



- we want to perform state estimation in real time, even when no events occur.

Diagnosability for timed automata

Diagnozer for timed automata

A Δ-diagnozer is a mapping $T\Sigma^* \to \{\texttt{safe}, \texttt{fault}\}$ detecting a fault at most Δ time units after one occurred.

We can still compute synchronized products and check reachability:

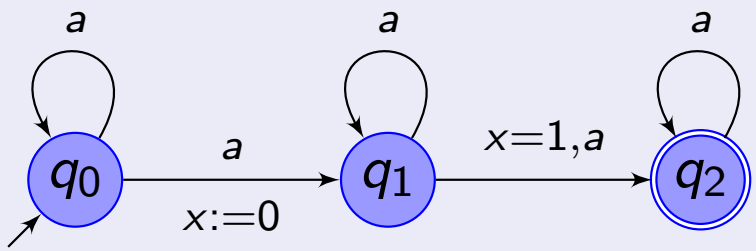
Theorem ([Tri02])

Diagnosability for timed automata is decidable.

[Tri02] Tripakis. Fault diagnosis for timed automata. FTRTFT 2002.

How about building a diagnozer?

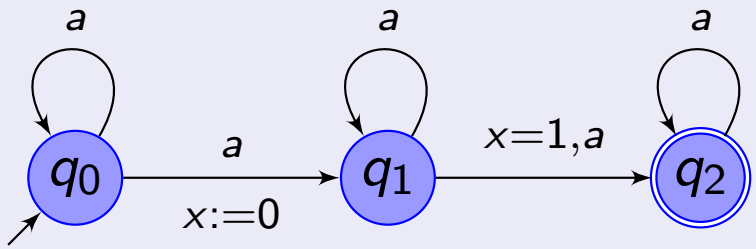
Some timed automata cannot be determinized



No deterministic timed automata accept the same timed language.

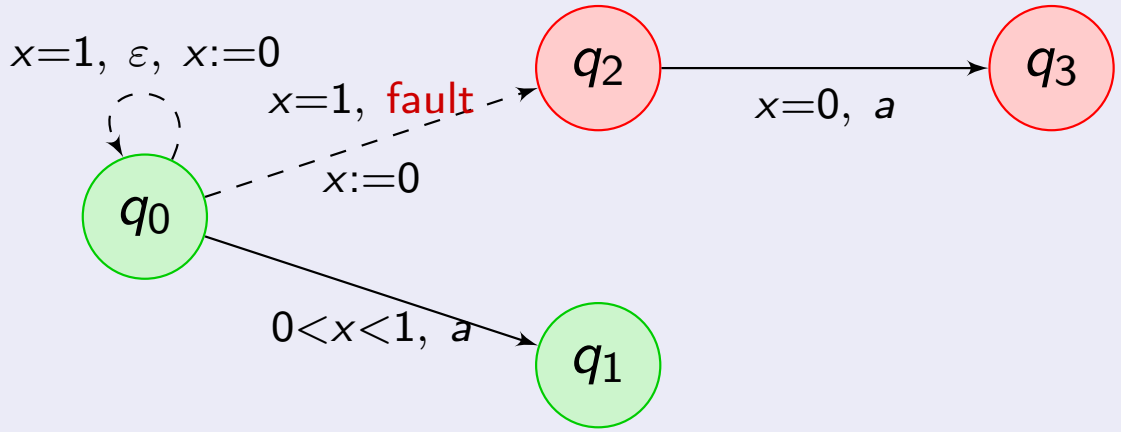
How about building a diagnozer?

Some timed automata cannot be determinized



No deterministic timed automata accept the same timed language.

Some diagnozable timed automata admit no DTA as diagnozers



How about building a diagnozer?

Building a diagnozer as a deterministic timed automaton [BCD05]

Look for DTA-diagnozers with

- limited number of clocks
- limited granularity and range of clocks
- known delay Δ.

Theorem ([BCD05])

*DTA-diagnozability under such restrictions is (effectively) decidable in 2*EXPTIME.

[BCD05] Bouyer, Chevalier, D'Souza. Fault diagnosis using timed automata. FoSSaCS 2005.

How about building a diagnozer?

Building a diagnozer as a deterministic timed automaton [BCD05]

Building an algorithm performing state estimation [Tri02]

$$R_{\text{obs}}(S, a) = \{(q', v') \mid \exists (q, v) \in S.\ (q, v) \xrightarrow{a} (q', v')\}$$

$$R_{\text{unobs}}(S, \delta) = \{(q', v') \mid \exists (q, v) \in S.\ (q, v) \xrightarrow{\delta}{}^{*}\ (q', v')\}$$

- compute state estimation when observing a after delay δ:

$$R_{\text{obs}}(R_{\text{unobs}}(S, \delta), a)$$

- return fault if only faulty states are obtained.

Computing R_{unobs} is expensive!

[BCD05] Bouyer, Chevalier, D'Souza. Fault diagnosis using timed automata. FoSSaCS 2005.
[Tri02] Tripakis. Fault diagnosis for timed automata. FTRTFT 2002.

Our contribution

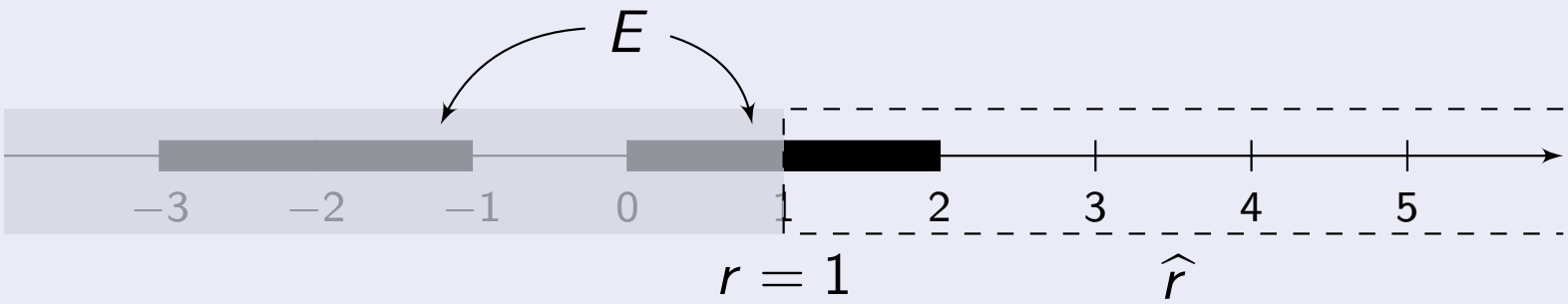
For timed automata with one clock, we

- use a powerset construction for timed automata (borrowed from [BJM17]) in order to represent the diagnozer;
- compute a convenient representation for this diagnozer; in particular, (part of) R_{unobs} is pre-computed, so that state estimation is much more efficient for delay transitions.
- implement our approach and compare to [Tri02].

[BJM17] Bouyer, Jaziri, Markey. On the determinization of timed systems. FORMATS 2017.
[Tri02] Tripakis. Fault diagnosis for timed automata. FTRTFT 2002.

Timed sets

Atomic timed sets

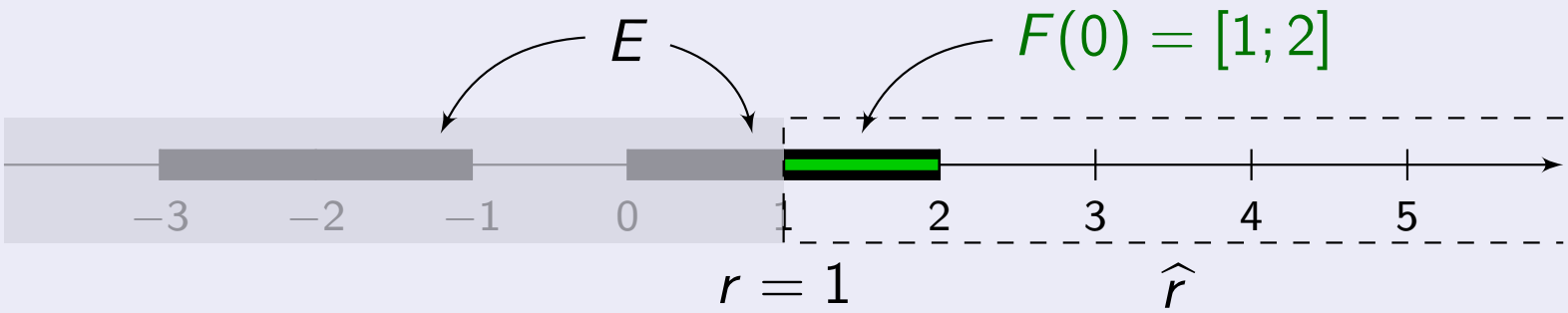


Given a finite union of intervals E and a right-infinite interval $\widehat{r}$, the timed set $F = (E; \widehat{r})$ is defined as

$$F : t \in \mathbb{R}_{\geq 0} \mapsto (E + t) \cap \widehat{r}.$$

Timed sets

Atomic timed sets

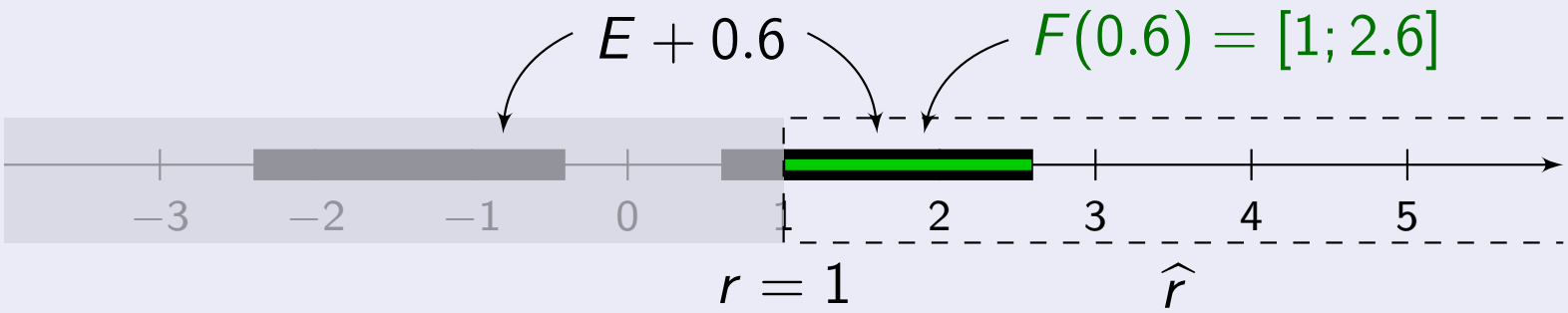


Given a finite union of intervals E and a right-infinite interval $\widehat{r}$, the timed set $F = (E; \widehat{r})$ is defined as

$$F : t \in \mathbb{R}_{\geq 0} \mapsto (E + t) \cap \widehat{r}.$$

Timed sets

Atomic timed sets

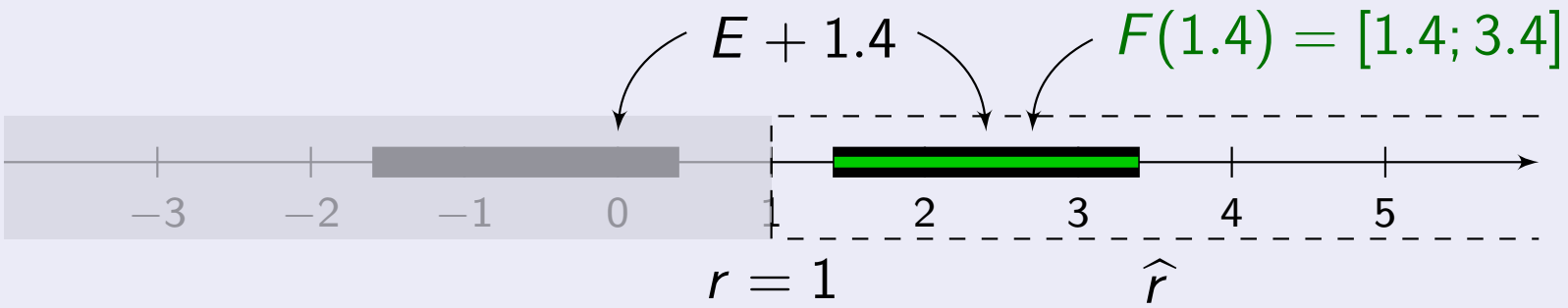


Given a finite union of intervals E and a right-infinite interval $\widehat{r}$, the timed set $F = (E; \widehat{r})$ is defined as

$$F : t \in \mathbb{R}_{\geq 0} \mapsto (E + t) \cap \widehat{r}.$$

Timed sets

Atomic timed sets



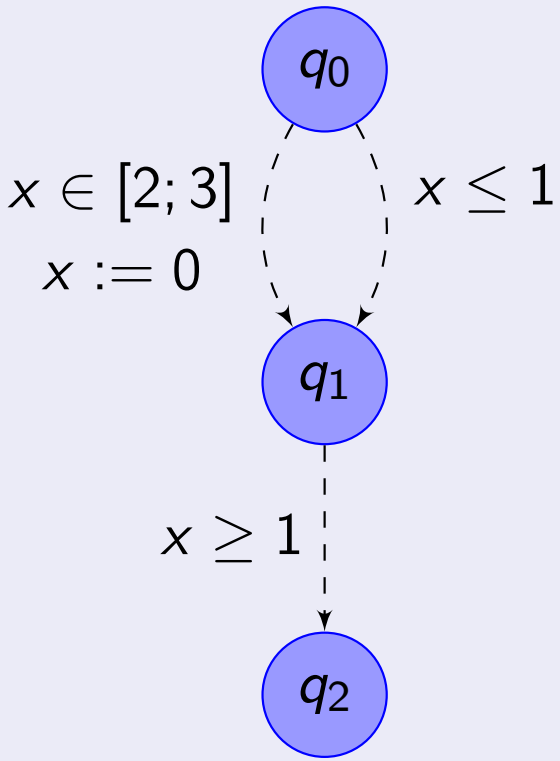
Given a finite union of intervals E and a right-infinite interval $\widehat{r}$, the timed set $F = (E; \widehat{r})$ is defined as

$$F : t \in \mathbb{R}_{\geq 0} \mapsto (E + t) \cap \widehat{r}.$$

Timed sets

Timed sets are used to compute the sets of reachable configurations.

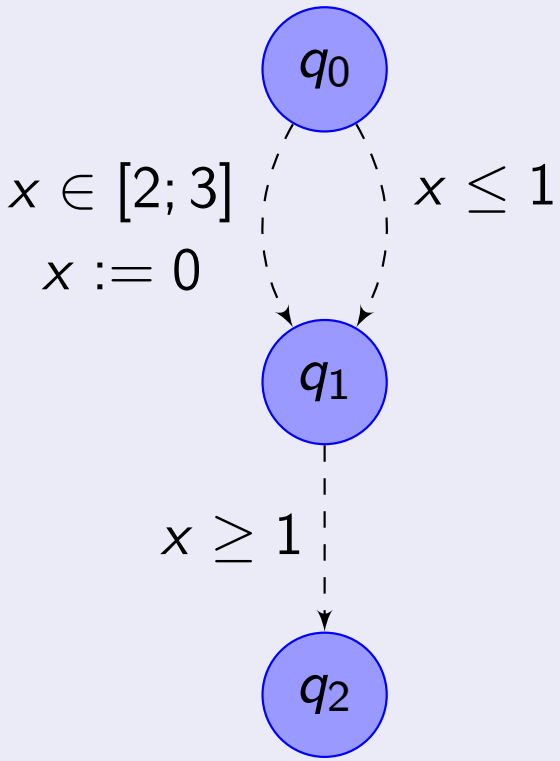
Example

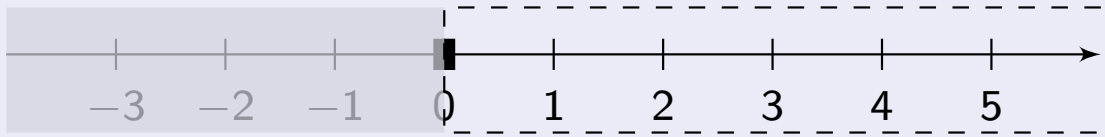


Timed sets

Timed sets are used to compute the sets of reachable configurations.

Example

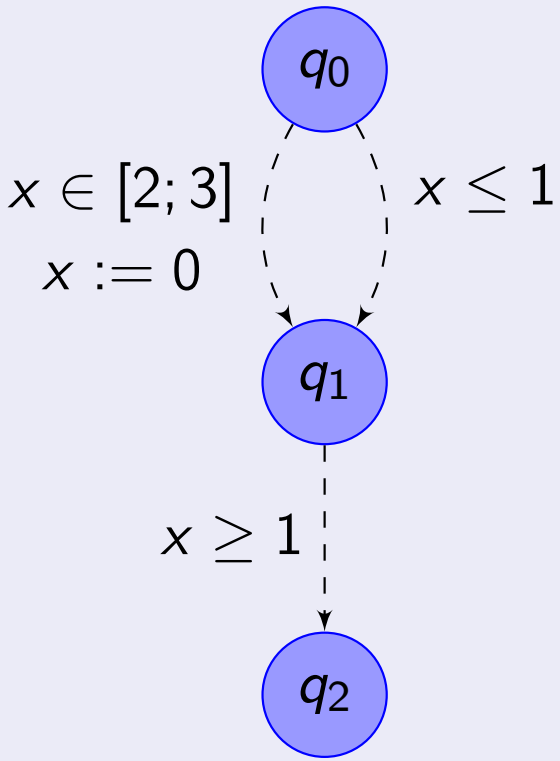


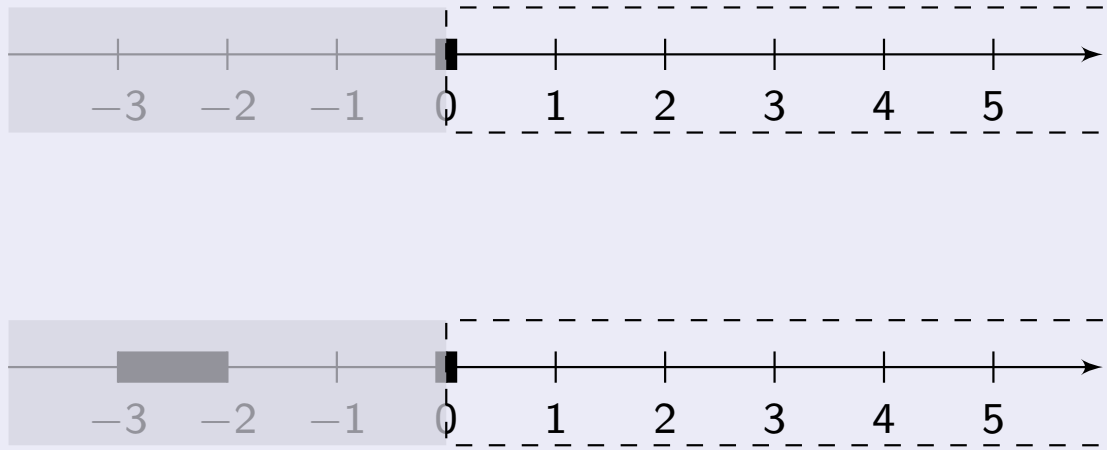


Timed sets

Timed sets are used to compute the sets of reachable configurations.

Example

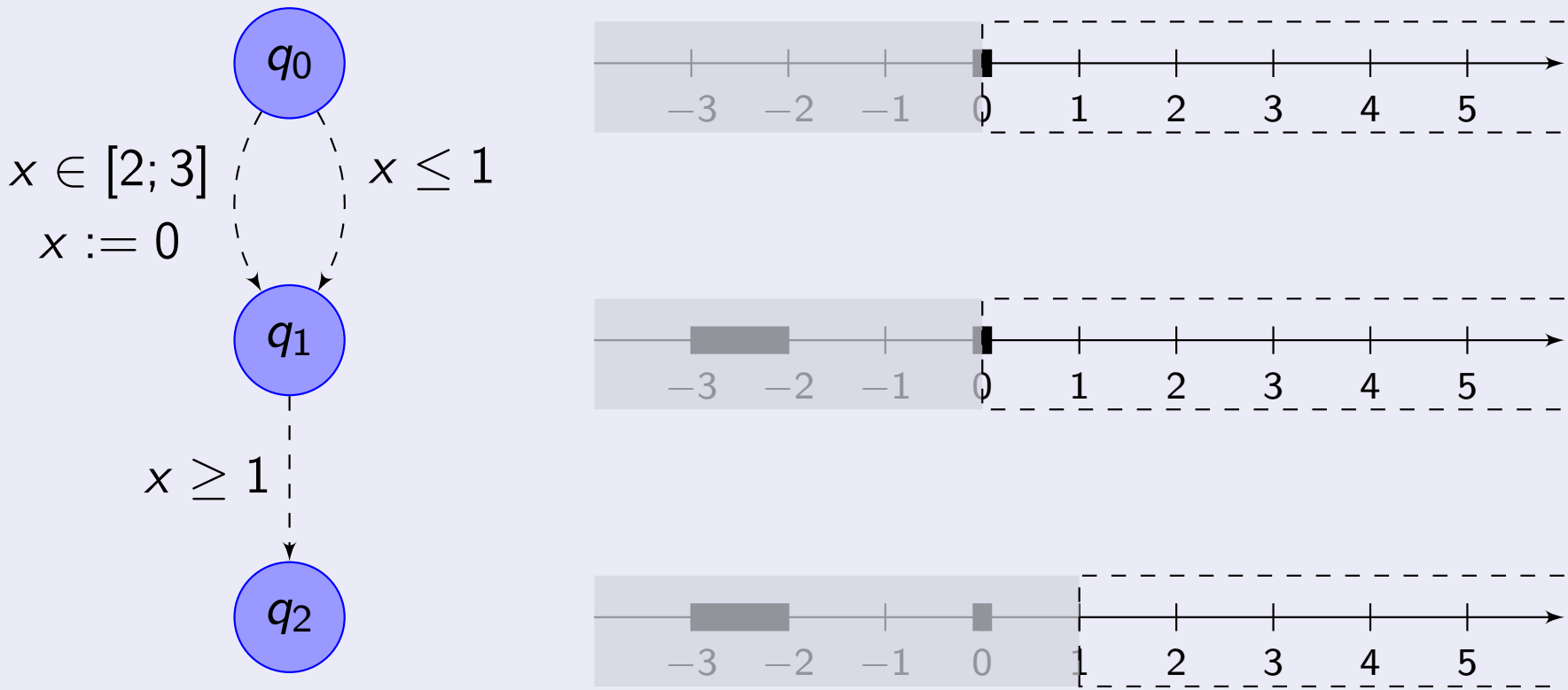




Timed sets

Timed sets are used to compute the sets of reachable configurations.

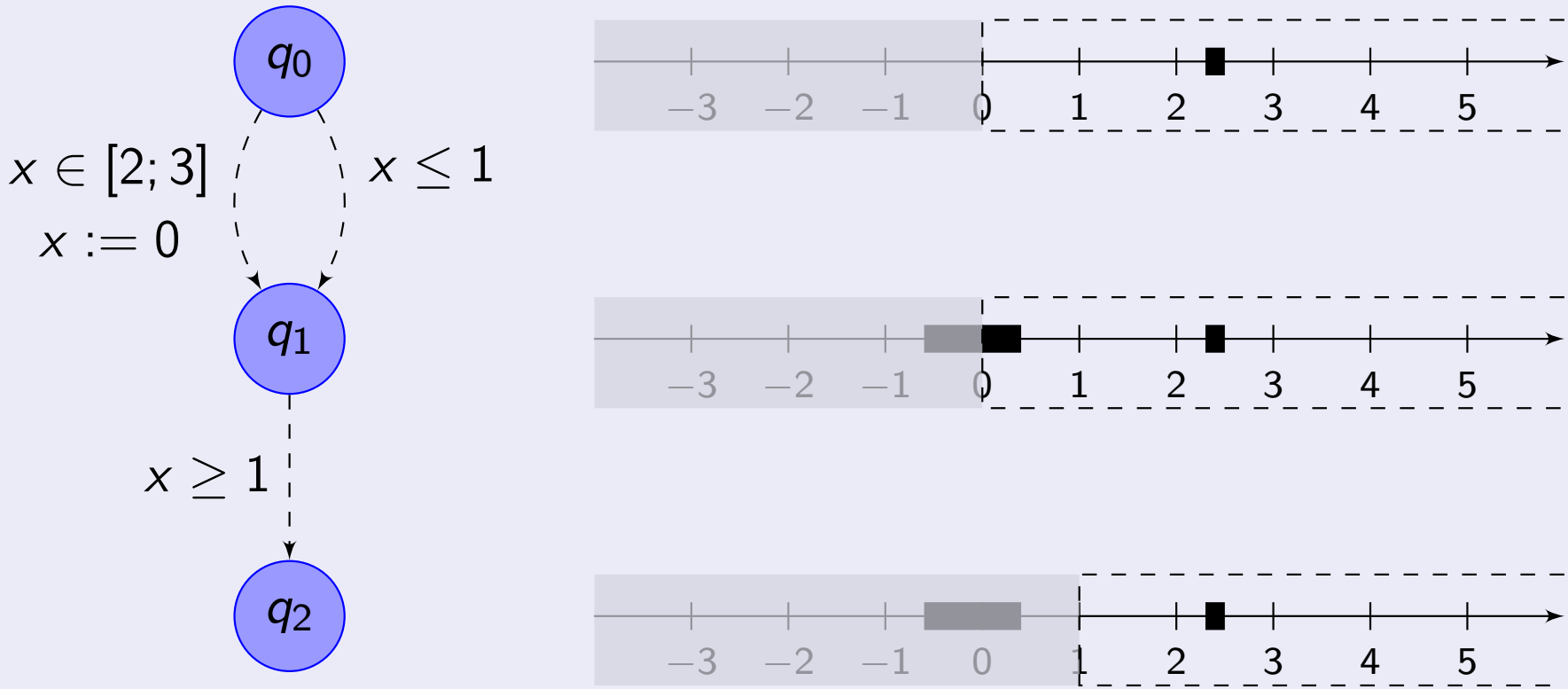
Example



Timed sets

Timed sets are used to compute the sets of reachable configurations.

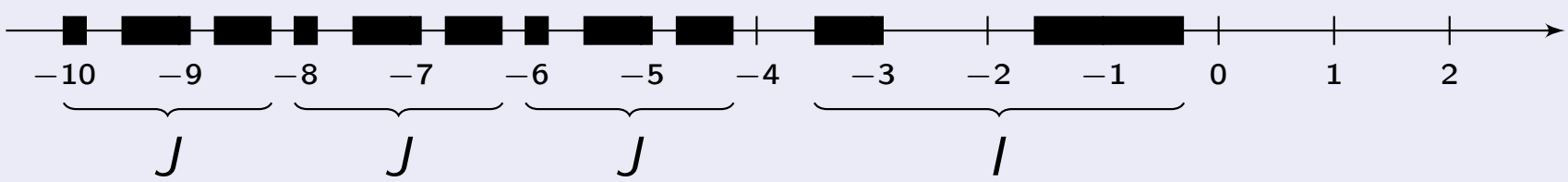
Example



Timed sets

Regular timed sets

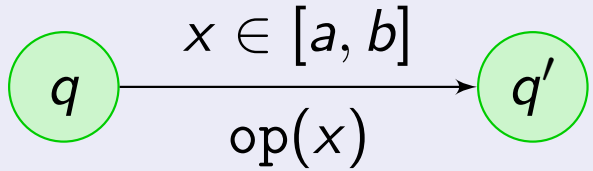
- **regular union of intervals**: sets of the form $I \cup \bigcup_{k=q}^{+\infty} J - k \cdot p$.

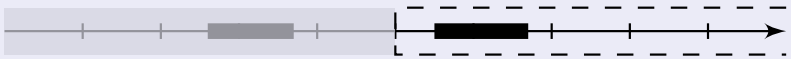


- **regular timed sets**: finite union of timed sets $(E_i, \widehat{r_i})$ for regular unions of intervals E_i.

Image of a timed set by a transition

Computing the effect of a transition

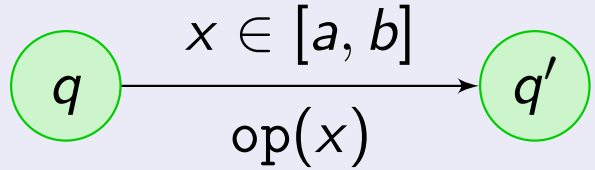




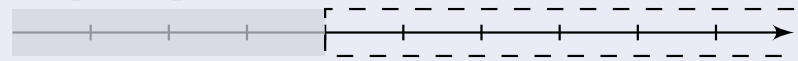
$(E, \widehat{r})$ in q

Image of a timed set by a transition

Computing the effect of a transition



if $[a; b] \cap \widehat{r} = \varnothing$:

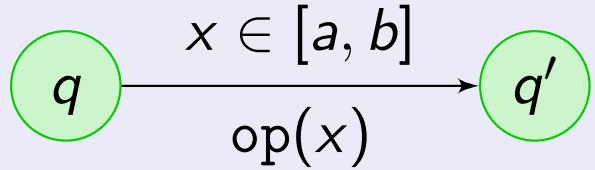


$(\varnothing, [0; +\infty))$

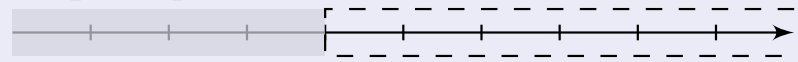
$(E, \widehat{r})$ in q

Image of a timed set by a transition

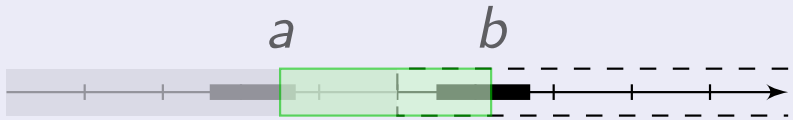
Computing the effect of a transition



if $[a; b] \cap \widehat{r} = \varnothing$:

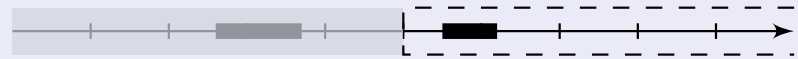


$(\varnothing, [0; +\infty))$



$(E, \widehat{r})$ in q

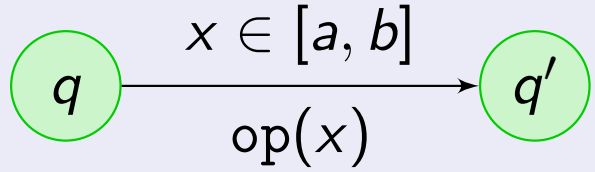
if $[a; b] \cap \widehat{r} \neq \varnothing$ and $\mathrm{op} = \mathtt{id}$:

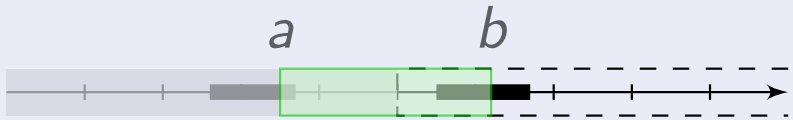


$(E \cap (-\infty; b], [\max\{a, r\}; +\infty))$

Image of a timed set by a transition

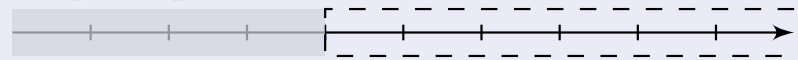
Computing the effect of a transition





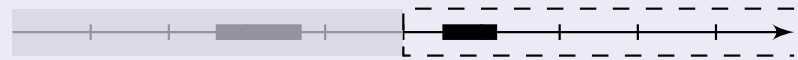
$(E, \widehat{r})$ in q

if $[a; b] \cap \widehat{r} = \varnothing$:



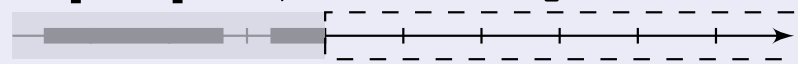
$(\varnothing, [0; +\infty))$

if $[a; b] \cap \widehat{r} \neq \varnothing$ and $\mathtt{op} = \mathtt{id}$:



$(E \cap (-\infty; b], [\max\{a, r\}; +\infty))$

if $[a; b] \cap \widehat{r} \neq \varnothing$ and $\mathtt{op} = \mathtt{reset}$:



$((E - [\max\{a, r\}; b]) \cap \mathbb{R}_{\leq 0}; [0; +\infty))$

ε-closure

Timed markings and their ε-closures

- A timed marking $M\colon s \mapsto \{(E_i, \widehat{r_i})\}$ assigns a timed set with each state.
- For a sequence w of silent transitions:

$$M^w\colon (s', d) \mapsto \left\{ v' \mid \exists s.\, \exists d_0 \leq d.\, \exists v \in M(s)(d_0).\, (s, v) \xrightarrow[w]{d}{}^{*} \ (s', v') \right\}$$

- The ε-closure of M is the union of M^w over all silent paths w.

ε-closure

Timed markings and their ε-closures

- A timed marking $M\colon s \mapsto \{(E_i, \widehat{r_i})\}$ assigns a timed set with each state.
- For a sequence w of silent transitions:

$$M^w\colon (s', d) \mapsto \left\{ v' \mid \exists s.\, \exists d_0 \leq d.\, \exists v \in M(s)(d_0).\, (s, v) \xrightarrow[w]{d}{}^{*} (s', v') \right\}$$

- The ε-closure of M is the union of M^w over all silent paths w.

Theorem

- *The ε-closure of a regular timed marking is a regular timed marking.*
- *ε-closures are finite unions of timed sets of the form $(E - K; \widehat{r'})$ for regular unions of intervals K that can be pre-computed.*

Implementation and experimentations

Implementation of our diagnozer

- prototype in Python (ca. 2000 lines of code)
- precompute sets for ε-closures
- simulate and perform state-estimation for trace entered by user; also state-prediction!
- prototype of the approach of [Tri02] for comparison.

Comparison of the results

Example. over 400 random runs of 10-20 actions on a 7-state TA:

	action	delay	precomputation
[BJM18]	0.17 s	10^{-5} s	11 s / 12 ko
[Tri02]	0.26 s	0.30 s	-

[Tri02] Tripakis. Fault diagnosis for timed automata. FTRTFT 2002.

[BJM18] Bouyer, Jaziri, Markey. Efficient timed diagnosis using automata over timed domains. RV 2018.

Conclusions and future work

Conclusions

- novel approach for building diagnozers for (one-clock) timed automata;
- performs pre-computation to speed-up computation of silent runs;
- can also predict occurrence of states

Future work

- improve implementation
  understand why pre-computation is sometimes very long;
- take invariants into account;
- extension to n clocks: from timed sets to timed zones?
- cost estimation in priced timed automata with unobservable cost.

A powerset construction for timed automata [BJM17]

Timed domains

A timed domain is a pair $(\mathcal{V}, \hookrightarrow)$ where $\mathcal{V}$ is a set and $\hookrightarrow : \mathcal{V} \times \mathbb{R}_{\geq 0} \to \mathcal{V}$ s.t. for all v, d and d',

$$\hookrightarrow(v, d + d') = \hookrightarrow(\hookrightarrow(v, d), d').$$

Example. $\mathcal{V} = [0; M] \cup \{+\infty\}$ and $\hookrightarrow(v, d) = \begin{cases} v + d & \text{if } v + d \leq M \\ +\infty & \text{otherwise.} \end{cases}$

Example. For a timed automaton $\mathcal{A}$ with states Q, the set of *timed markings* of $\mathcal{A}$, equipped with $\hookrightarrow^Q(M, d) : s \mapsto M^\epsilon(s, d)$, is a timed domain.

Update

An update is a mapping $\lambda : \mathcal{V} \to \mathcal{V}$.

[BJM17] Bouyer, Jaziri, Markey. On the determinization of timed systems. FORMATS 2017.

A powerset construction for timed automata [BJM17]

Automata over timed domains

An automaton over timed domain $(\mathcal{V}, \hookrightarrow)$, alphabet Σ and updates Λ is a tuple $(Q, (q_0, v_0), T, F)$ with $T \subseteq Q \times \mathcal{V} \times \Sigma \times \Lambda \times Q$.

Theorem

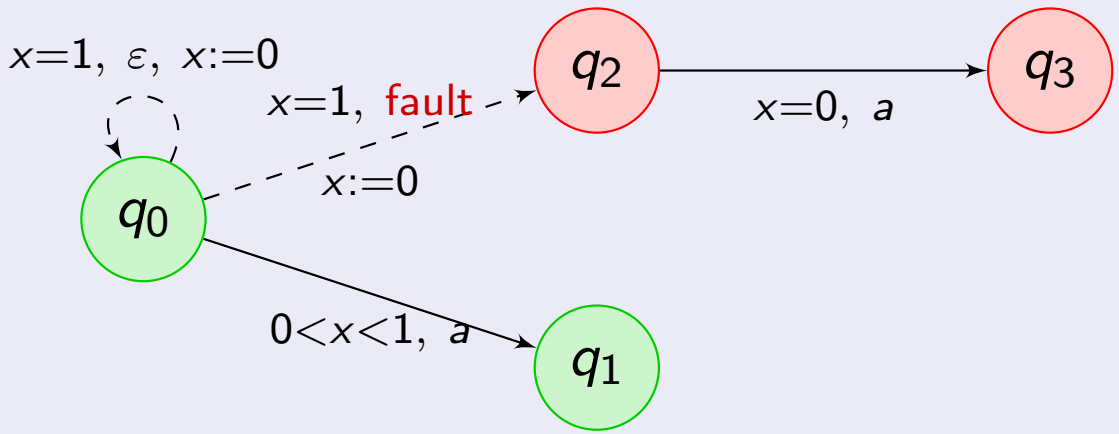
Any automaton over a timed domain $\mathcal{V}$ can be determinized as an automaton over timed domain $\mathcal{P}(\mathcal{V})^{|Q|}$.

Our diagnozer is the automaton we get by this determinization procedure.

[BJM17] Bouyer, Jaziri, Markey. On the determinization of timed systems. FORMATS 2017.

Running the tool

Small example




```
# States.  Size :  4
q0;q1;q2;q3
# Transitions
q2;[0,0];0;q3;a
q0;]0,1[;0;q1;a
q0;[1,1];0;q0;e
q0;[1,1];0;q2;e
```

State evaluation: initially

State : q0:

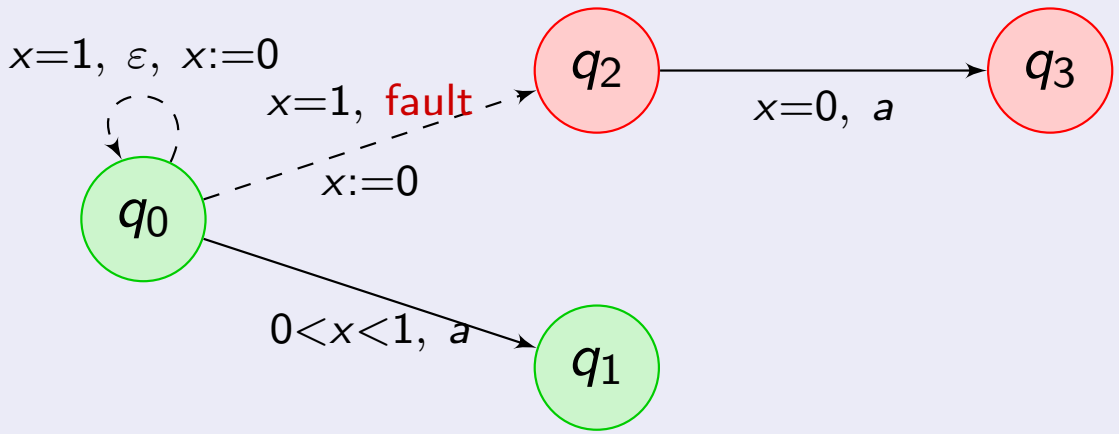
```
(( [0.00,0.00] ) u U_k in N ( [0.00,0.00] - 1.00 -k1.00 ) )
n [0.00,inf[
```

State : q2:

```
(( [-2.00,-2.00] U [-1.00,-1.00] ) u U_k in N ( [0.00,0.00]
- 3.00 -k1.00 ) ) n [0.00,inf[
```

Running the tool

Small example




```
# States.  Size :  4
q0;q1;q2;q3
# Transitions
q2;[0,0];0;q3;a
q0;]0,1[;0;q1;a
q0;[1,1];0;q0;e
q0;[1,1];0;q2;e
```

State evaluation: after delay 0.3

State : q0:

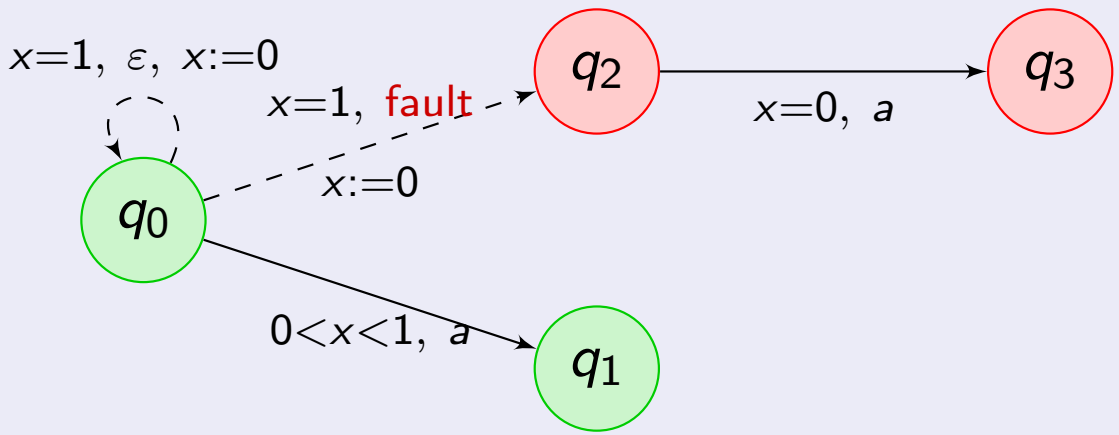
```
(( [0.30,0.30] ) u U_k in N ( [0.00,0.00] - 0.70 -k1.00 ) )
n [0.00,inf[
```

State : q2:

```
(( [-1.70,-1.70] U [-0.70,-0.70] ) u U_k in N ( [0.00,0.00]
- 2.70 -k1.00 ) ) n [0.00,inf[
```

Running the tool

Small example




```
# States.  Size :  4
q0;q1;q2;q3
# Transitions
q2;[0,0];0;q3;a
q0;]0,1[;0;q1;a
q0;[1,1];0;q0;e
q0;[1,1];0;q2;e
```

State evaluation: after delay 0.3 and action a

```
State :  q1:
([0.00,0.00]) n [0.00,inf[
```