Computing maximally-permissive strategies in *timed games*

Emily Clement[1] Thierry Jéron[2] Nicolas Markey[3] David Mentré[4]

[1]INRIA - Mitsubishi Electric, Rennes

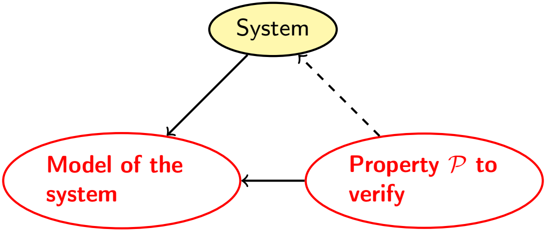
[2]INRIA-IRISA, Rennes

[3]CNRS-IRISA, Rennes

[4]Mitsubishi Electric, Rennes

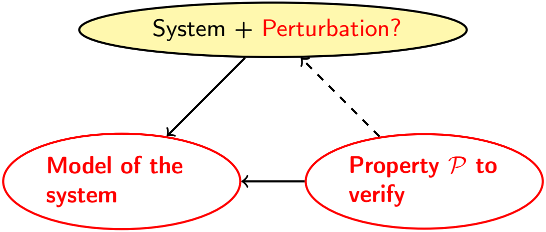
28 May 2019

Motivations: Verify properties despite perturbations



▷ How to model it? Timed automata + Verification of $\mathcal{P}$.

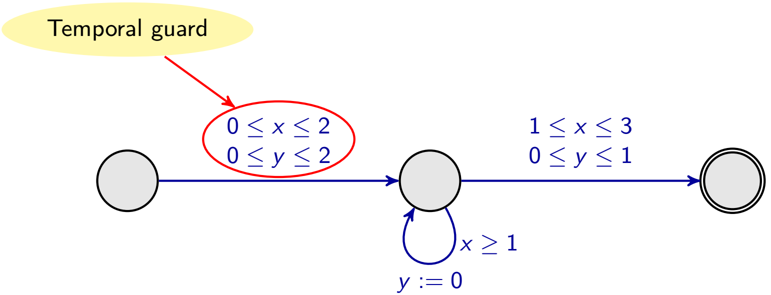
Motivations: Verify properties despite perturbations



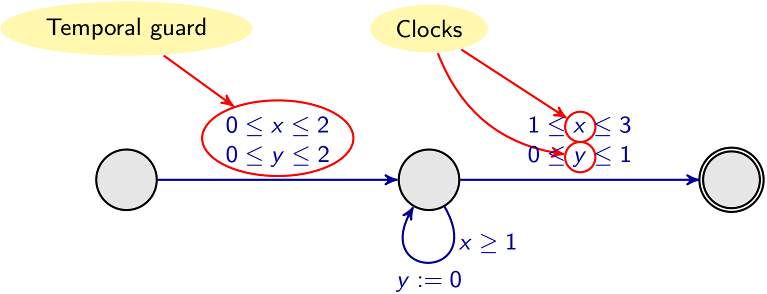
- ▷ How to model it? Timed automata + Verification of $\mathcal{P}$.
- ▷ Our goal? Verify **with robustness**.

Table of contents

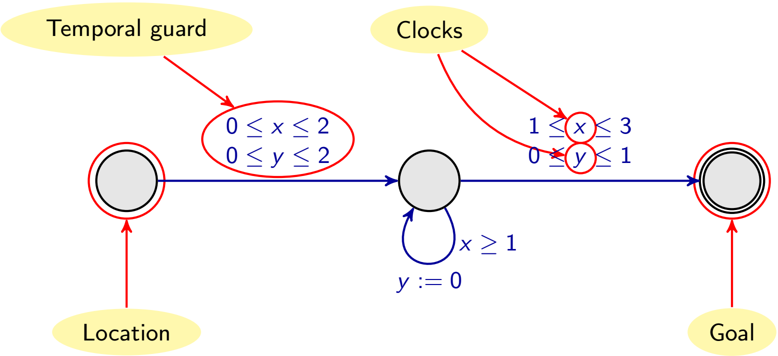
Vocabulary of a timed automaton



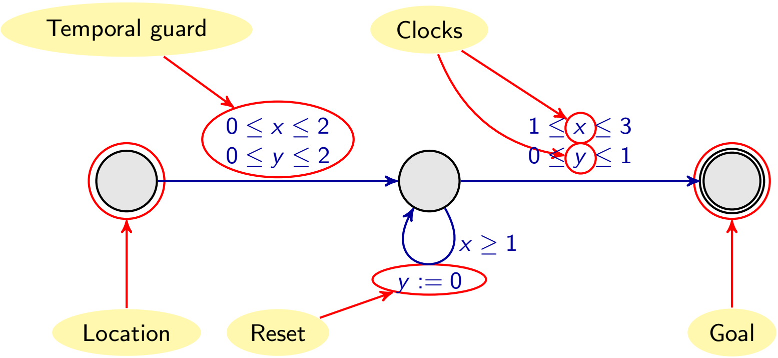
Vocabulary of a timed automaton



Vocabulary of a timed automaton

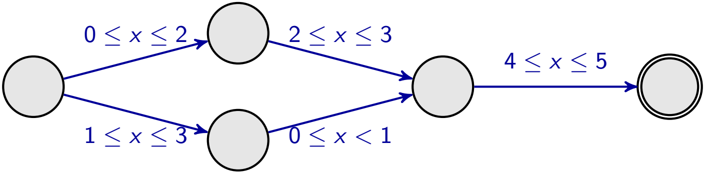


Vocabulary of a timed automaton



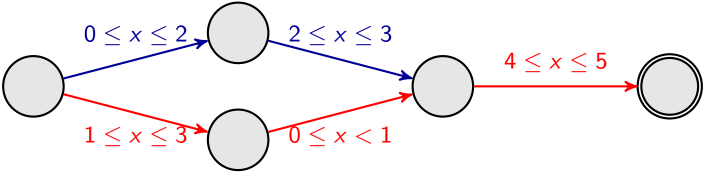
Issues for reachability & robustness

- (Temporal) reachability ?



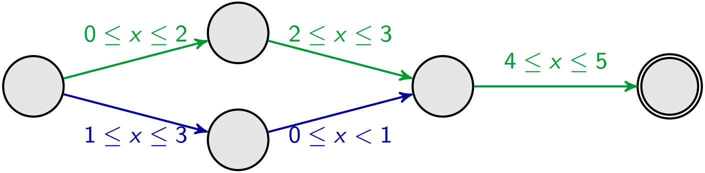
Issues for reachability & robustness

- (Temporal) reachability



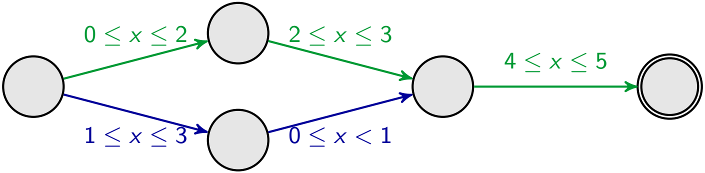
Issues for reachability & robustness

- (Temporal) reachability ✓

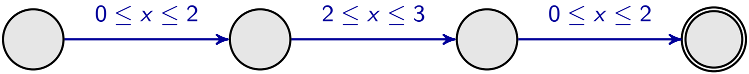


Issues for reachability & robustness

- (Temporal) reachability ✓

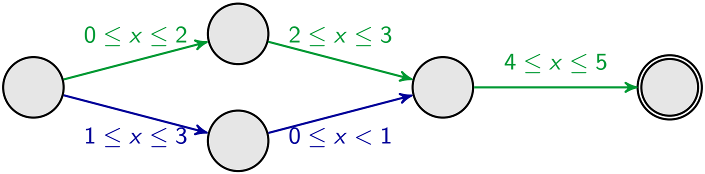


- Robustness ?

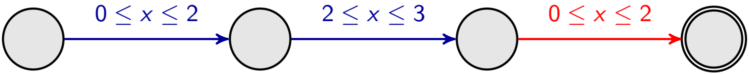


Issues for reachability & robustness

- (Temporal) reachability ✓

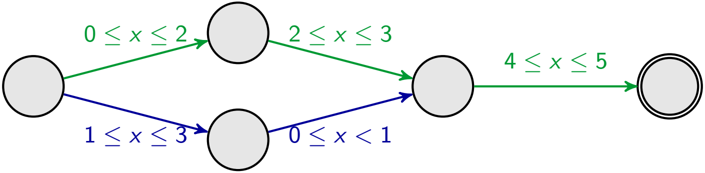


- Robustness

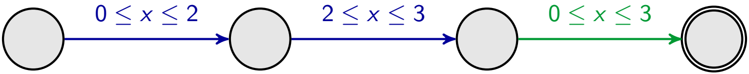


Issues for reachability & robustness

- (Temporal) reachability ✓



- Robustness ✓



Our goal

- Define our semantic of robustness

Our goal

- Define our semantic of robustness
- Construct an algorithm that answers the following question:

For $p \in \mathbb{R}$, a timed automaton $\mathcal{A}$ and a configuration, is it at least $p-$robust?

Our goal

- Define our semantic of robustness
- Construct an algorithm that answers the following question:

For $p \in \mathbb{R}$, a timed automaton $\mathcal{A}$ and a configuration, is it at least $p-$robust?

- Our Method
  - ▷ Construct an algorithm that computes **exactly** the robustness of any automaton/configuration.

Table of contents

State of the art of the robustness

- Topological robustness
  - ▷ Gupta, Hezinger, Jagadeesan "Robust Timed Automata", 1997
  - ▷ Tools: stability theorems.

State of the art of the robustness

- Topological robustness
  - ▷ Gupta, Hezinger, Jagadeesan "Robust Timed Automata", 1997
  - ▷ Tools: stability theorems.
- Guard enlargement
  - ▷ Sankur "Robustness in Timed Automata", PhD Thesis, 2013
  - ▷ Tools: game theory.

State of the art of the robustness

- Topological robustness
  - ▷ Gupta, Hezinger, Jagadeesan "Robust Timed Automata", 1997
  - ▷ Tools: stability theorems.
- Guard enlargement
  - ▷ Sankur "Robustness in Timed Automata", PhD Thesis, 2013
  - ▷ Tools: game theory.
- Delay enlargement
  - ▷ Bouyer, Fang, Markey "Permissive strategies in timed automata and games", AVOCS'15
  - ▷ Tools: game theory

State of the art of the robustness

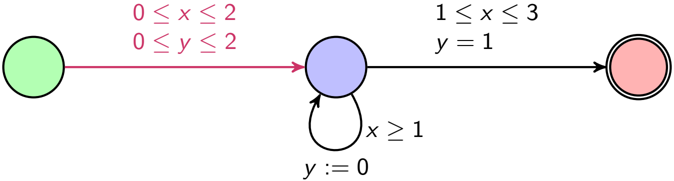
- Topological robustness
  - ▷ Gupta, Hezinger, Jagadeesan "Robust Timed Automata", 1997
  - ▷ Tools: stability theorems.
- Guard enlargement
  - ▷ Sankur "Robustness in Timed Automata", PhD Thesis, 2013
  - ▷ Tools: game theory.
- Delay enlargement
  - ▷ Bouyer, Fang, Markey "Permissive strategies in timed automata and games", AVOCS'15
  - ▷ Tools: game theory
  - ▷ A constructive algorithm: ✓

State of the art of the robustness

- Topological robustness
  - ▷ Gupta, Hezinger, Jagadeesan "Robust Timed Automata", 1997
  - ▷ Tools: stability theorems.
- Guard enlargement
  - ▷ Sankur "Robustness in Timed Automata", PhD Thesis, 2013
  - ▷ Tools: game theory.
- Delay enlargement
  - ▷ Bouyer, Fang, Markey "Permissive strategies in timed automata and games", AVOCS'15
  - ▷ Tools: game theory
  - ▷ A constructive algorithm: ✓
  - ▷ Multiple clocks: ✗.

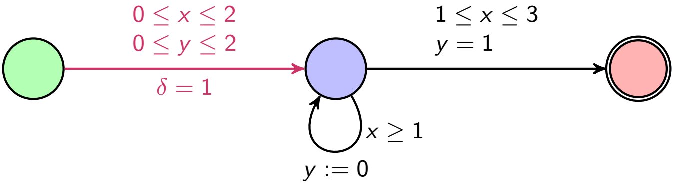
Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement



Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

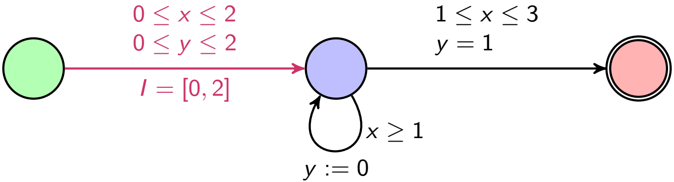


How to quantify robustness?

- **No delay enlargement**: We propose $\delta = 1$.

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

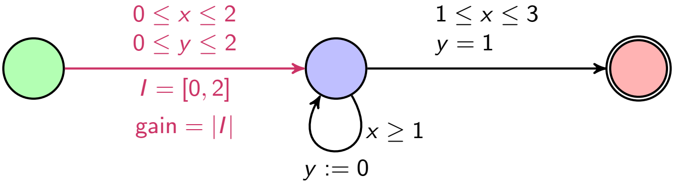


How to quantify robustness?

- **Delay enlargement**: we propose an **interval** $I = [0, 2]$.

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

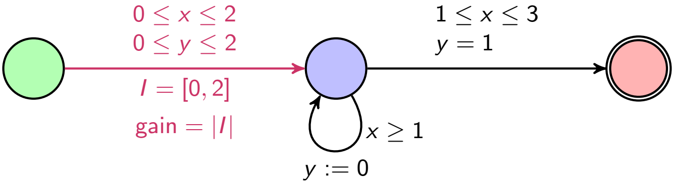


How to quantify robustness?

- **Delay enlargement**: we propose an **interval** $I = [0, 2]$.
  How to quantify the robustness: gain $|I|$

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

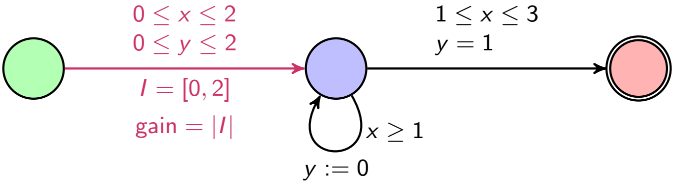


How to quantify robustness?

- **Delay enlargement**: we propose an **interval** $I = [0, 2]$.
  How to quantify the robustness: gain $|I|$
  Conditions: Every delay of $\delta \in I$ verifies the guards

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

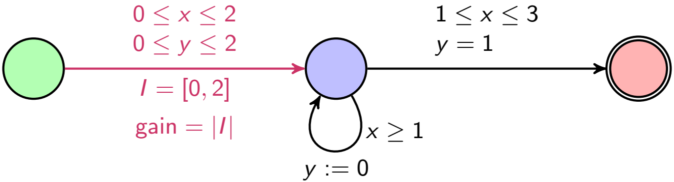


How to quantify robustness?

- **Delay enlargement**: we propose an **interval** $I = [0, 2]$.
  How to quantify the robustness: gain $|I|$
  Conditions: Every delay of $\delta \in I$ verifies the guards
- Which delay is applied ? The worst case.

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement



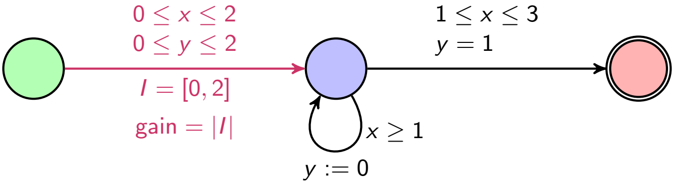
- How to model the "Best case/Worst case"?

Choice of Interval I

Choice of delay δ

Quantifying robustness: The example of delay enlargement

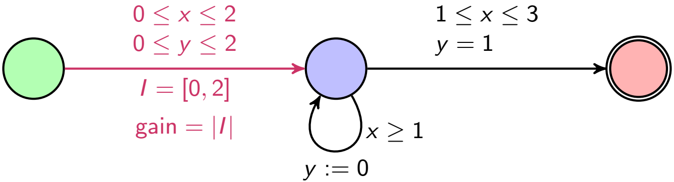
- Delay/ No delay enlargement



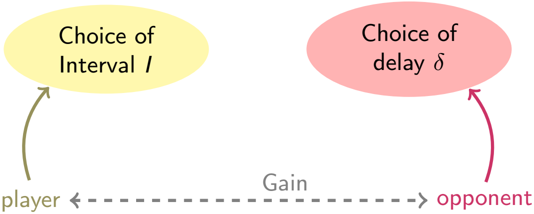
- How to model the "Best case/Worst case"?

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

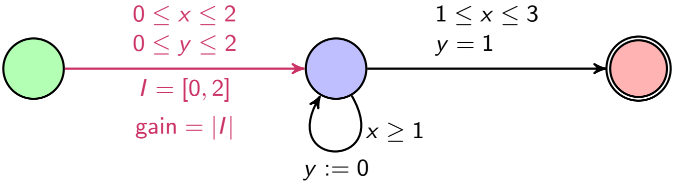


- How to model the "Best case/Worst case"?

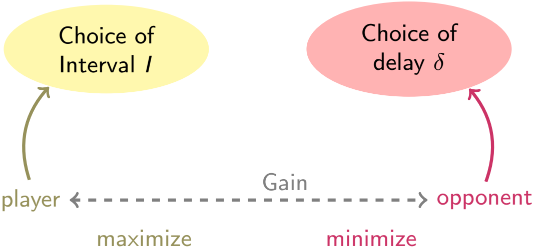


Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

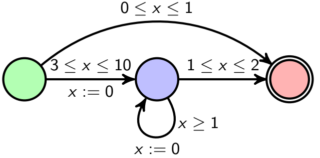


- How to model the "Best case/Worst case"?

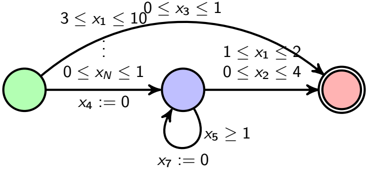


State of the art for delay enlargement: Bouyer et al. vs our model

- Bouyer et al.

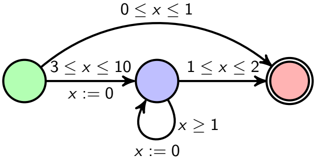


- Our model



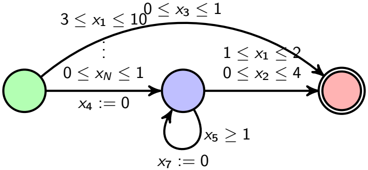
State of the art for delay enlargement: Bouyer et al. vs our model

- Bouyer et al.



  ▷ Op: min of sum of the **inverses**: ✓

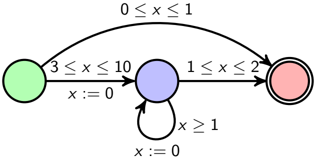
- Our model



  ▷ Op: min: ✓

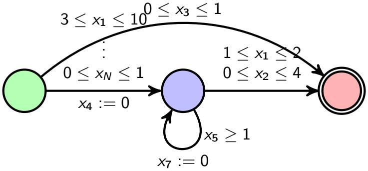
State of the art for delay enlargement: Bouyer et al. vs our model

- Bouyer et al.



  - ▷ Op: min of sum of the **inverses**: ✓
  - ▷ : ✓

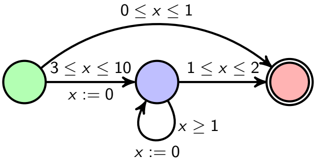
- Our model



  - ▷ Op: min: ✓
  - ▷ : ✓

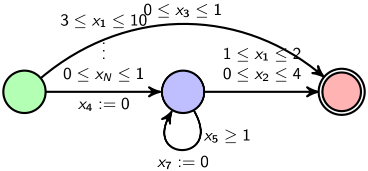
State of the art for delay enlargement: Bouyer et al. vs our model

- Bouyer et al.



  - ▷ Op: min of sum of the **inverses**: ✓
  - ▷ 🕒: ✓
  - ▷ 🕒 . . . 🕒: ✗

- Our model



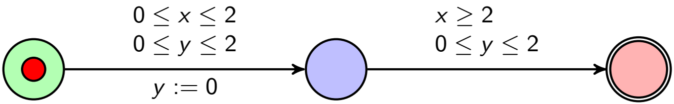
  - ▷ Op: min: ✓
  - ▷ 🕒: ✓
  - ▷ 🕒 . . . 🕒: ✓

Table of contents

An example of gain computation

- Are we 1−robust? Let's lose!

The clock values $(0, 0)$.



Player

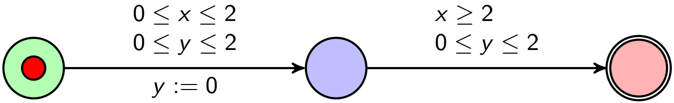
Gain

Opponent

An example of gain computation

- Are we 1−robust? Let's lose!

The clock values $(0, 0)$.



Player

I play $[0, 2]$

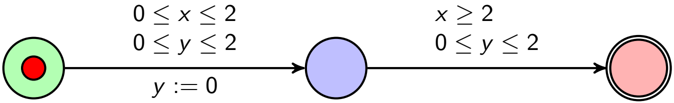
Gain

Opponent

An example of gain computation

- Are we 1−robust? Let's lose!

The clock values $(0, 0)$.



Player

I play $[0, 2]$

Gain

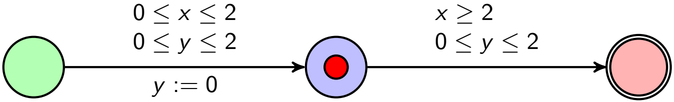
$= 2$

Opponent

An example of gain computation

- Are we 1−robust? Let's lose!

The clock values $(0, 0)$.



Player

I play $[0, 2]$

Gain

$= 2$

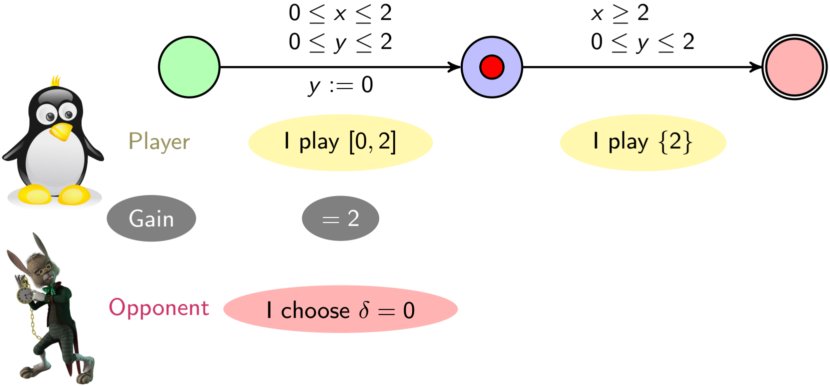
Opponent

I choose $\delta = 0$

An example of gain computation

- Are we 1−robust? Let's lose!

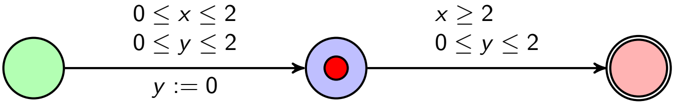
The clock values $(0, 0)$.



An example of gain computation

- Are we 1−robust? Let's lose!

The clock values $(0, 0)$**.**



Player — I play $[0, 2]$ — I play $\{2\}$

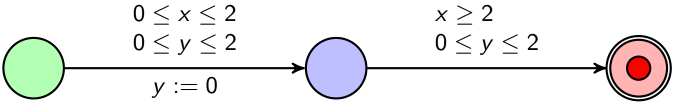
Gain — $= 2$ — $= 0$

Opponent — I choose $\delta = 0$

An example of gain computation

- Are we 1−robust? Let's lose!

The clock values $(2,2)$.



Player — I play $[0,2]$ — I play $\{2\}$

Gain — $= 2$ — $= 0$

Opponent — I choose $\delta = 0$ — I choose $\delta = 2$

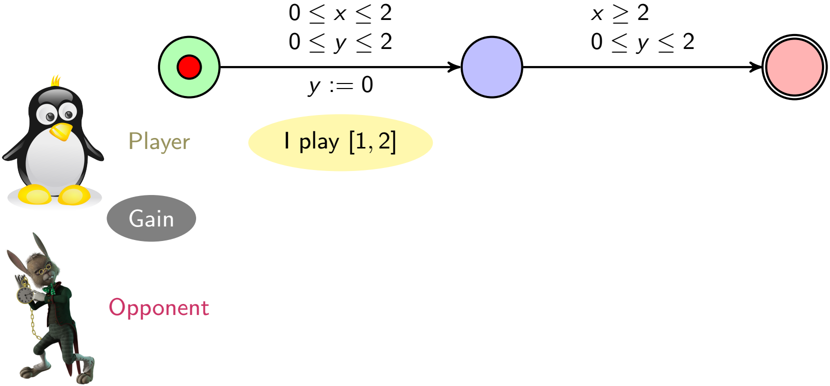
- Final gain

$\min(2,0) = 0$: ☹

An example of gain computation

- Are we 1−robust? Let's win!

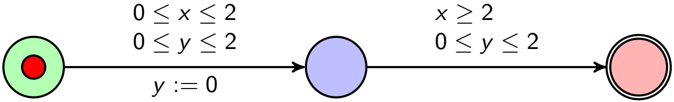
The clock values $(0, 0)$.



An example of gain computation

- Are we 1−robust? Let's win!

The clock values $(0, 0)$.



Player

I play $[1, 2]$

Gain

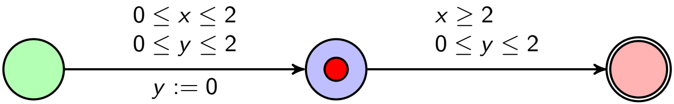
$= 1$

Opponent

An example of gain computation

- Are we 1−robust? Let's win!

The clock values (1, 0).



Player — I play $[1, 2]$

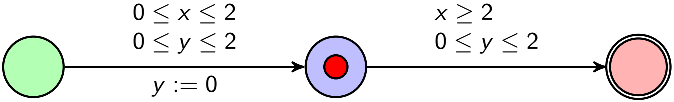
Gain $= 1$

Opponent — I choose $\delta = 1$

An example of gain computation

- Are we 1−robust? Let's win!

The clock values (1, 0).



Player I play $[1, 2]$ I play $[1, 2]$

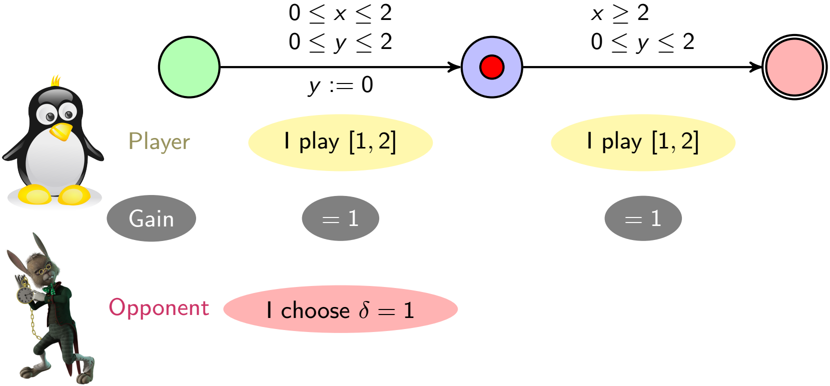
Gain $= 1$

Opponent I choose $\delta = 1$

An example of gain computation

- Are we 1−robust? Let's win!

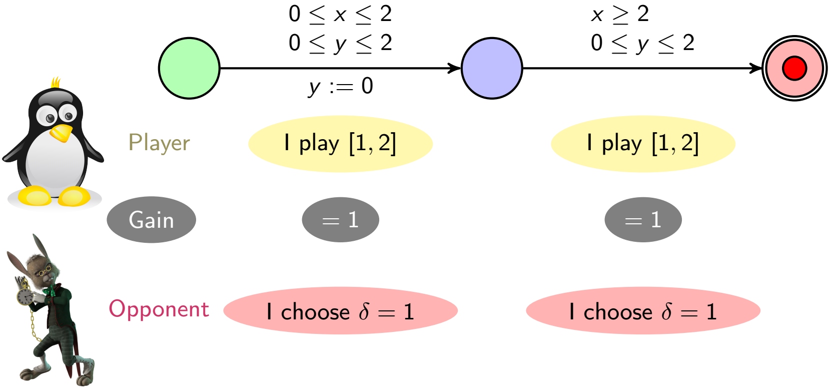
The clock values $(1, 0)$**.**



An example of gain computation

- Are we 1−robust? Let's win!

The clock values $(2, 1)$.



- Final gain

$\min(1, 1) = 1$:

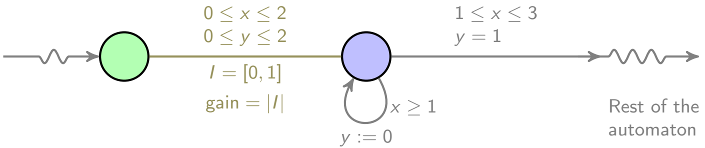
What is the gain?

- The gain: a way to quantify robustness
  - ▷ Gain $\searrow$ = Robustness $\searrow$
  - ▷ A recursive calculus of a function $\mathcal{T}_i(q, v)$.

What is the gain?

- The gain: a way to quantify robustness
  - ▷ Gain $\searrow$ = Robustness $\searrow$
  - ▷ A recursive calculus of a function $\mathcal{T}_i(q, v)$.

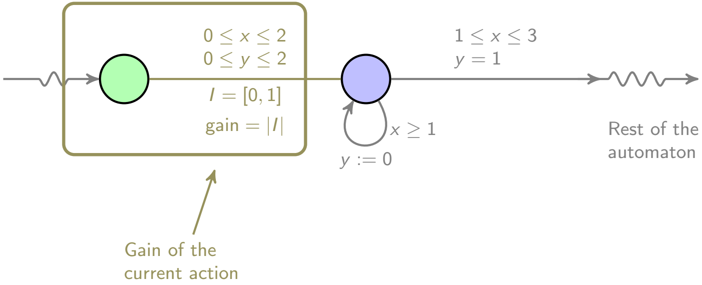
- A recursive algorithm to compute the gain



What is the gain?

- The gain: a way to quantify robustness
  - ▷ Gain ↘ = Robustness ↘
  - ▷ A recursive calculus of a function $\mathcal{T}_i(q, v)$.

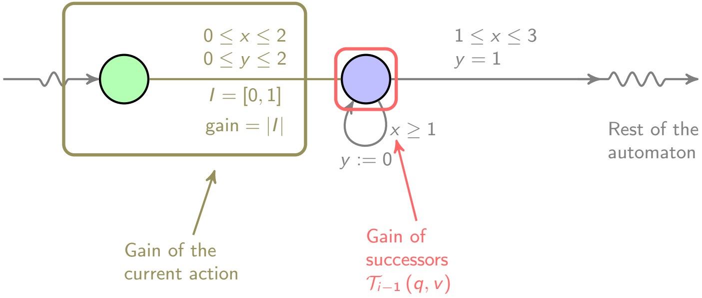
- A recursive algorithm to compute the gain



What is the gain?

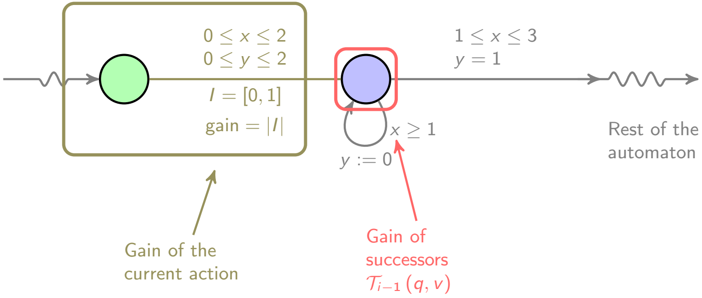
- The gain: a way to quantify robustness
  - ▷ Gain $\searrow$ = Robustness $\searrow$
  - ▷ A recursive calculus of a function $\mathcal{T}_i(q, v)$.

- A recursive algorithm to compute the gain



What is the gain?

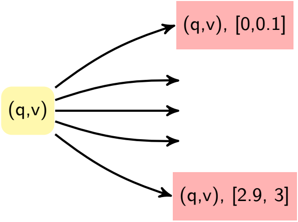
- The gain: a way to quantify robustness
  - ▷ Gain ↘ = Robustness ↘
  - ▷ A recursive calculus of a function $\mathcal{T}_i(q, v)$.
- A recursive algorithm to compute the gain



Gain of the automaton: minimum of current gain and the gain of the successors

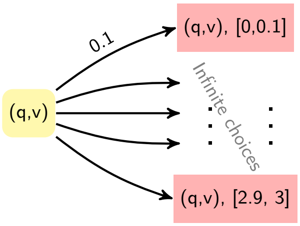
What is the gain ?

Guard $0 \leq x \leq 3$



What is the gain ?

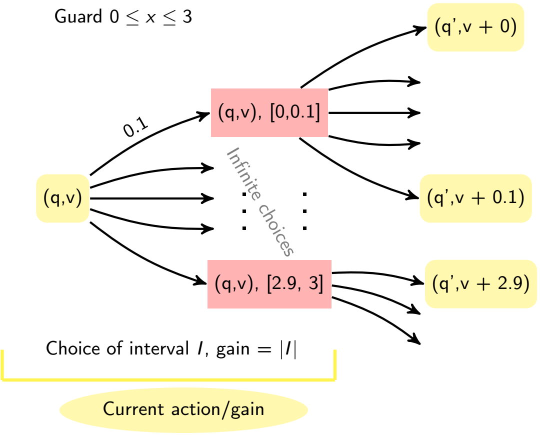
Guard $0 \leq x \leq 3$



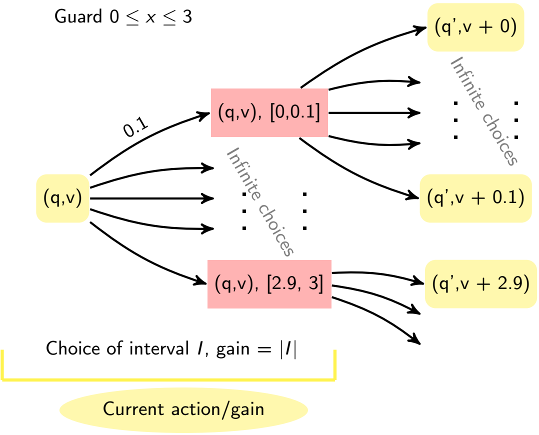
Choice of interval I, gain $= |I|$

Current action/gain

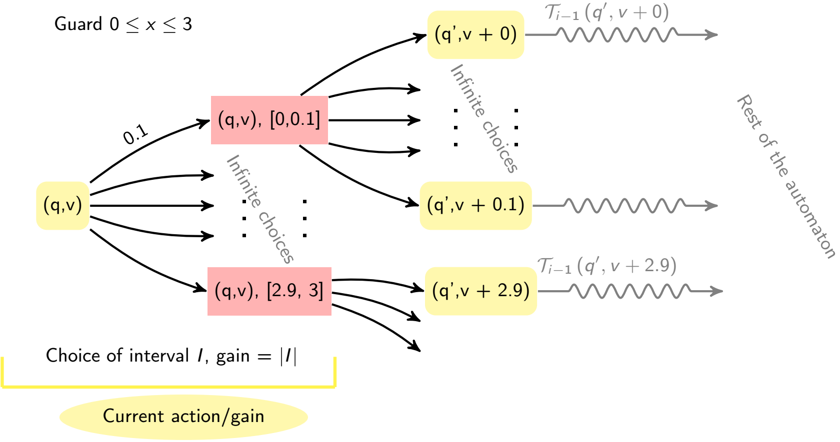
What is the gain ?



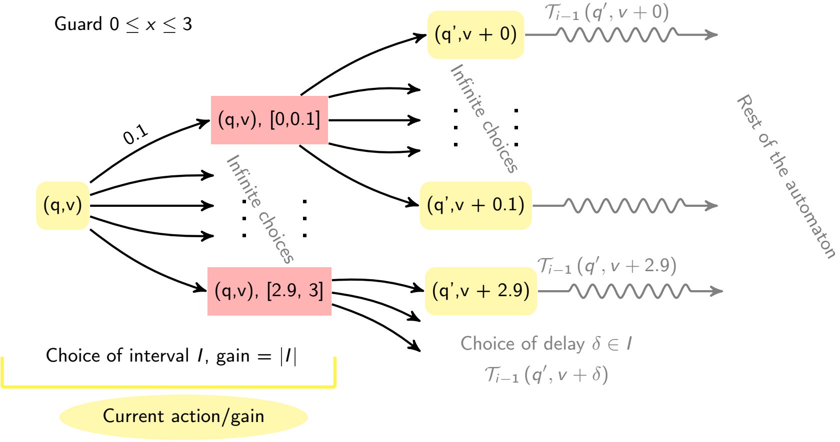
What is the gain ?



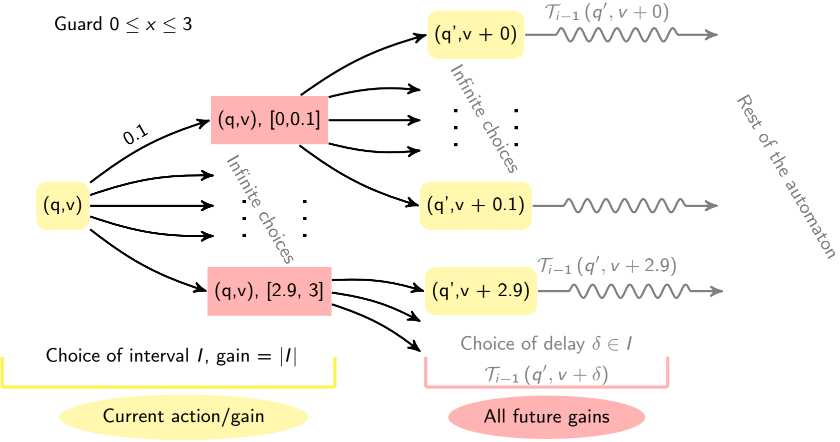
What is the gain ?



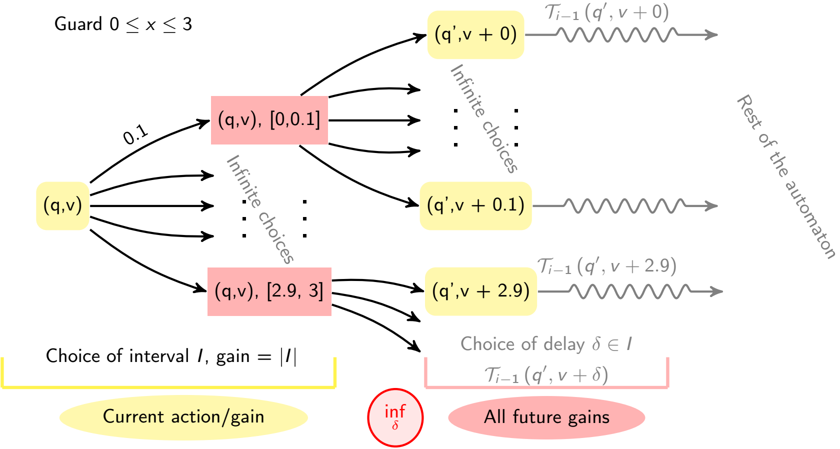
What is the gain ?



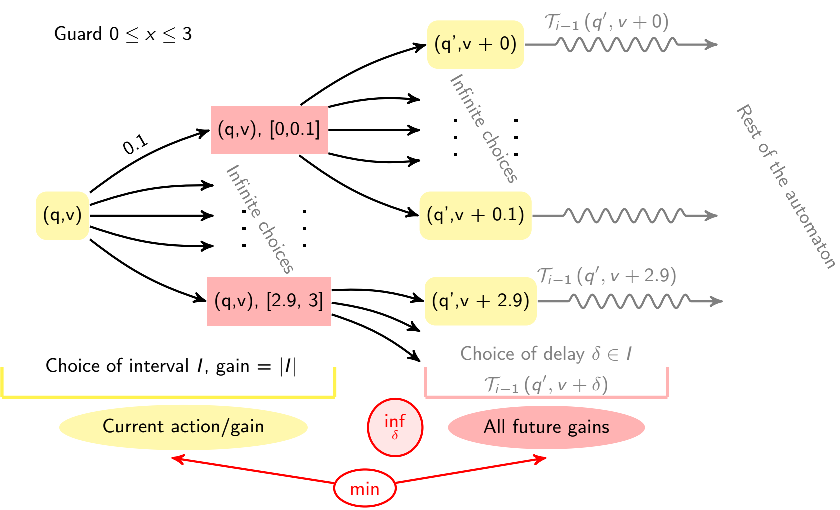
What is the gain ?



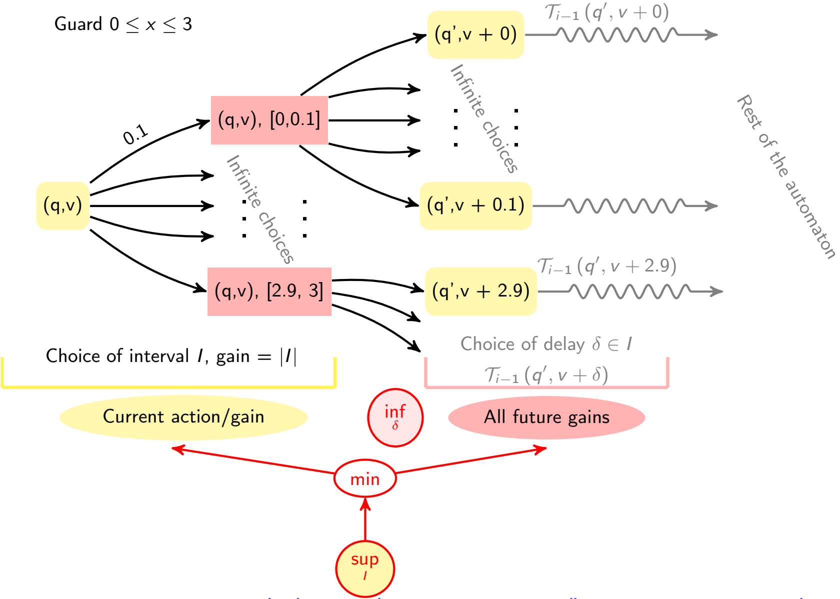
What is the gain ?



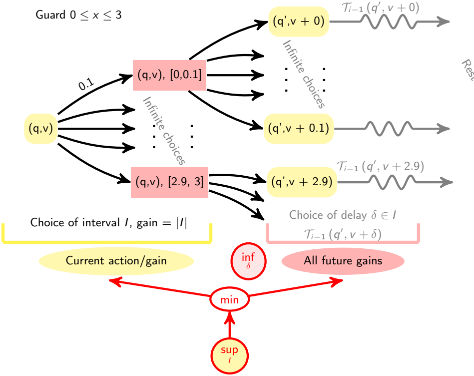
What is the gain ?



What is the gain ?



What is the gain ?



Issues: How to compute the gain?

- inf / sup: **infinite** choices & **opposite** strategies
- determine a finite number of strategies to test of the two players:

$$\inf \Rightarrow \min \text{ and } \sup \Rightarrow \max .$$

Table of contents

For linear automata: Strategy of the opponent

We will restrict to linear automata, *i.e* we don't consider

Thm: Bouyer et al. Strategy of the opponent

Choice of player

$[a, b]$

Best respond of opponent

$\delta = b$

For linear automata: Strategy of the opponent

We will restrict to linear automata, *i.e* we don't consider

Issues: Our model: opponent's best strategy

Choice of player

$[a, b]$

Best respond of opponent ?

$\cancel{\delta = b}$?

For linear automata: Counter-example and results for our model

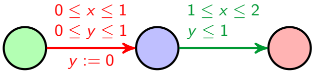
Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	$\cancel{\delta = b}$?

For linear automata: Counter-example and results for our model

Issues: opponent's strategy

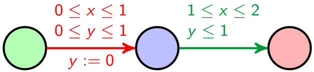
Choice of player	Best respond of opponent ?
$[a, b]$	$\cancel{\delta = b}$?



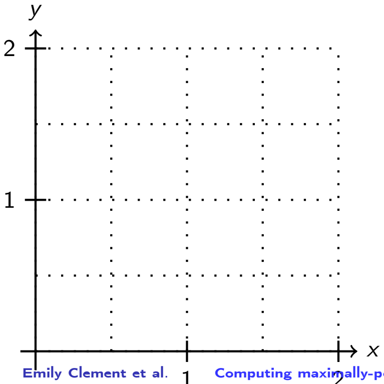
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	~~$\delta = b$~~ ?



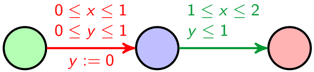
- Graph of the counter example:



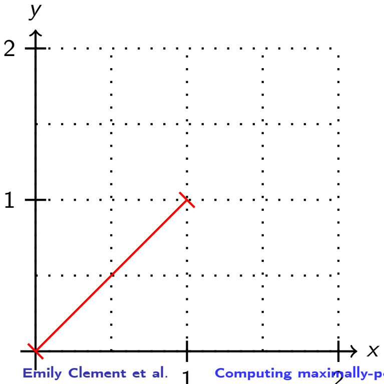
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	$\cancel{\delta = b}$?



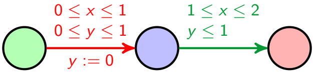
- Graph of the counter example:



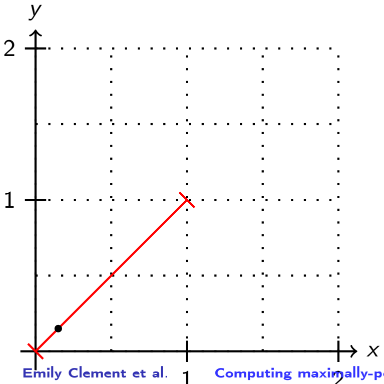
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	$\cancel{\delta = b}$?



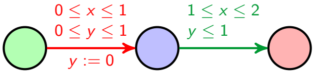
- Graph of the counter example:



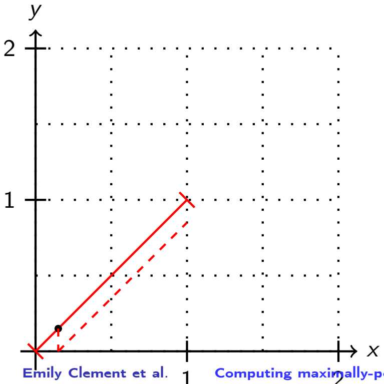
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	$\cancel{\delta = b}$?



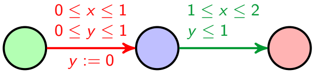
- Graph of the counter example:



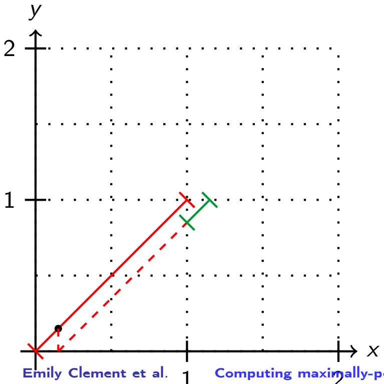
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	~~$\delta = b$~~ ?



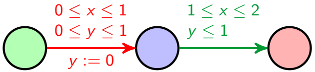
- Graph of the counter example:



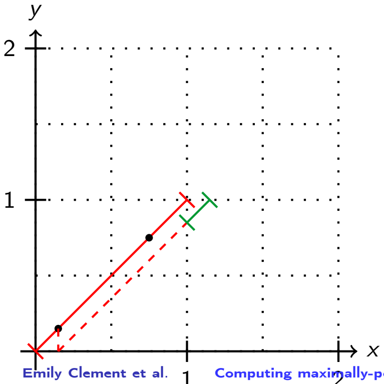
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	~~$\delta = b$~~ ?



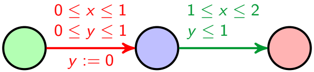
- Graph of the counter example:



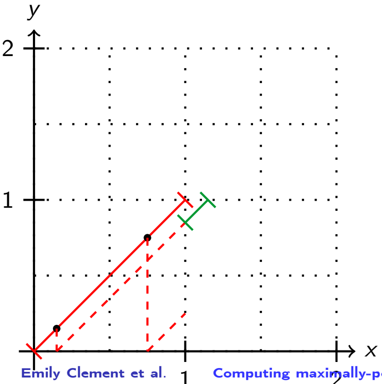
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

Choice of player	Best respond of opponent ?
$[a, b]$	~~$\delta = b$~~ ?



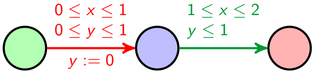
- Graph of the counter example:



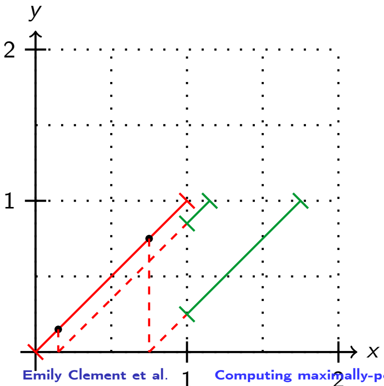
For linear automata: Counter-example and results for our model

Thm: opponent's strategy

Choice of player	Best respond of opponent
$[a, b]$	$\delta \in \{a, b\}$

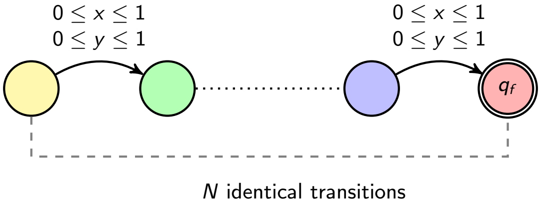


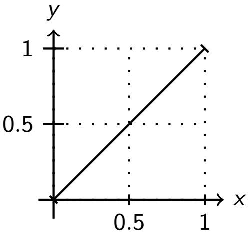
- Graph of the counter example:



For linear automata: examples of computation of the gain

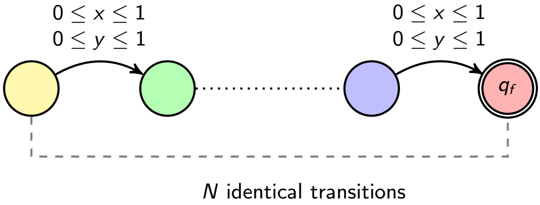
- Identical guards

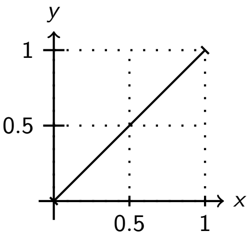




For linear automata: examples of computation of the gain

- Identical guards





- The strategy of the player and the opponent

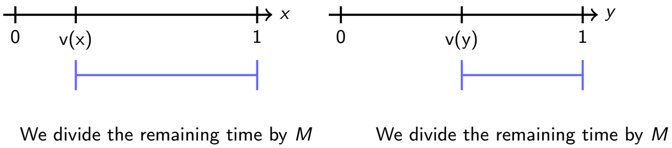


Figure: Time left for each clock

For linear automata: examples of computation of the gain

•Identical guards

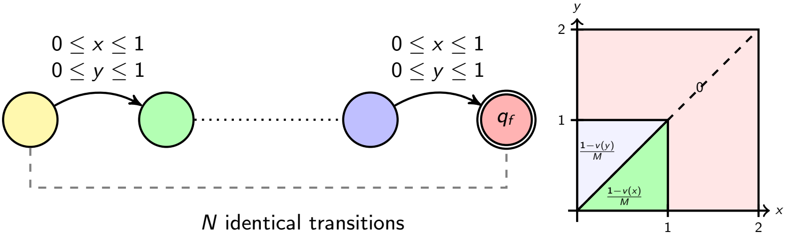
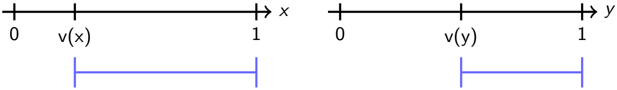


Figure: Gain in q_{N-M}

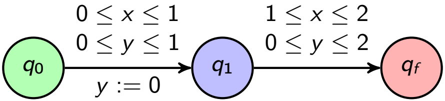
•The strategy of the player and the opponent



We divide the remaining time by M We divide the remaining time by M

Figure: Time left for each clock

A more complicated example



- The computation of the gain

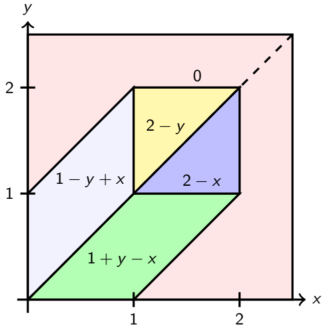


Figure: Gain in q_1

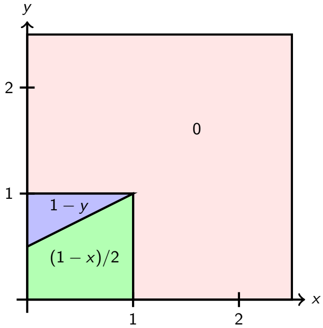
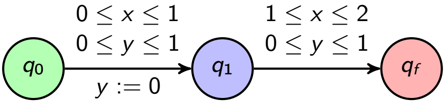
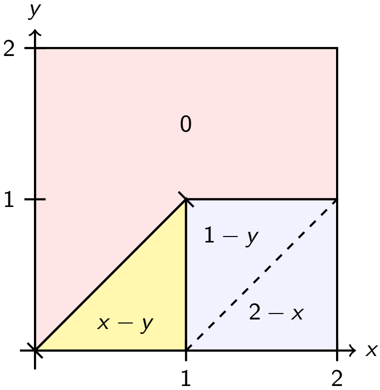


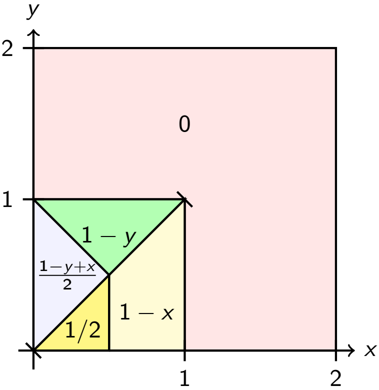
Figure: Gain in q_0

Other examples





(a) Gain in q_1



(b) Gain in q_0

For linear automata: strategy of the player

- Intuition for the form of $v \mapsto T(q, v)$
  - ▷ Computable cutting of zones

For linear automata: strategy of the player

- Intuition for the form of $v \mapsto T\,(q, v)$
  - ▷ Computable cutting of zones
  - ▷ Piece-wise affine function, with computable pieces and coefficients

For linear automata: strategy of the player

- Intuition for the form of $v \mapsto T(q, v)$
  - ▷ Computable cutting of zones
  - ▷ Piece-wise affine function, with computable pieces and coefficients
- Our method
  - ▷ Compare all the choices, zone by zones

For linear automata: strategy of the player

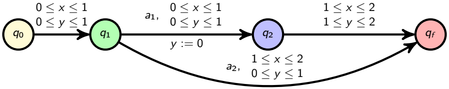
- Intuition for the form of $v \mapsto T(q, v)$
  - ▷ Computable cutting of zones
  - ▷ Piece-wise affine function, with computable pieces and coefficients
- Our method
  - ▷ Compare all the choices, zone by zones
  - ▷ Solve the maximization / a and b of $\min(b - a, f(a), g(b))$, where f and g are affine functions.

For linear automata: strategy of the player

- Intuition for the form of $v \mapsto T(q, v)$
  - ▷ Computable cutting of zones
  - ▷ Piece-wise affine function, with computable pieces and coefficients
- Our method
  - ▷ Compare all the choices, zone by zones
  - ▷ Solve the maximization / a and b of $\min(b - a, f(a), g(b))$, where f and g are affine functions.
  - ▷ Compare all the choices/gains and conclude.

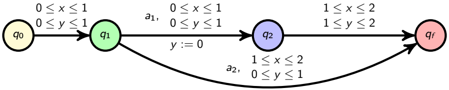
The case of non-linear automata

- A counter example of our opponent strategy



The case of non-linear automata

- A counter example of our opponent strategy



- Gain and counter example: the gain function $v \mapsto T(q_1, v)$

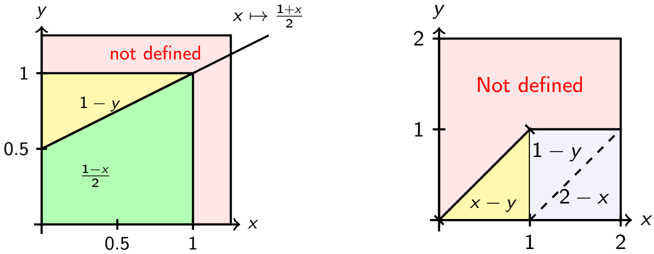
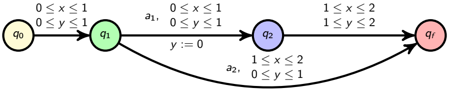


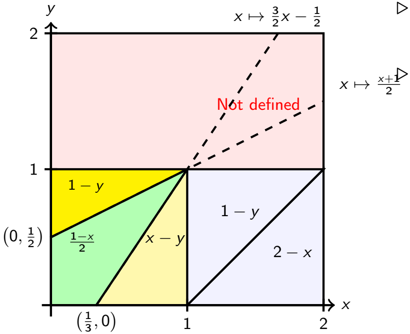
Figure: The gain function $v \mapsto T(q_1, v)$ depending on the action chosen (a_1 at left, a_2 at right)

The case of non-linear automata

- A counter example of our opponent strategy



- Gain and counter example: the gain function $v \mapsto T(q_1, v)$

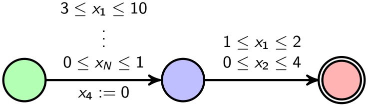


- ▷ Limits between 2 zones: $\Delta = 1/3 - x$.
- ▷ Choosing a in the green zone and b in the yellow zone, the opponent must decide of a delay with the following gain function:

$$\min\left(\inf_{\delta \in [a, \Delta]} \frac{-x - \delta + 1}{2}, \inf_{\delta \in [\Delta, b]} (x + \delta)\right)$$

Achieved, ongoing and future works

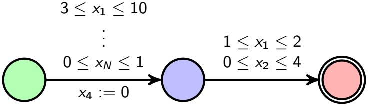
- Achieved and ongoing works



- Future work

Achieved, ongoing and future works

- Achieved and ongoing works

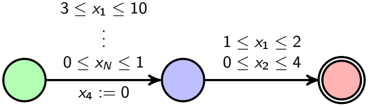


▷ With: Op: max.

- Future work

Achieved, ongoing and future works

- Achieved and ongoing works



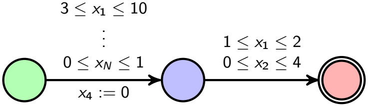
  ▷ With: Op: max.

  ▷ 🕒: ✓

- Future work

Achieved, ongoing and future works

- Achieved and ongoing works

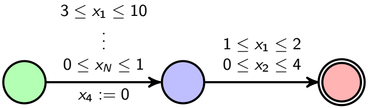


  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷ ...: ✓

- Future work

Achieved, ongoing and future works

- Achieved and ongoing works

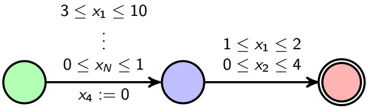


  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷  . . . : ✓
  - ▷ Strategy of : ✓

- Future work

Achieved, ongoing and future works

- Achieved and ongoing works

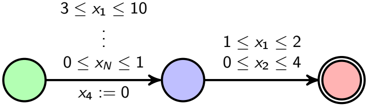


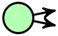
  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷  . . . : ✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 

- Future work

Achieved, ongoing and future works

- Achieved and ongoing works

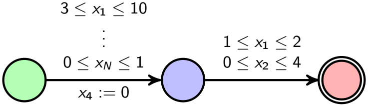


  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷  . . . : ✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 
  - ▷ : 

- Future work

Achieved, ongoing and future works

- Achieved and ongoing works



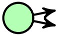
  ▷ With: Op: max.

  ▷ : ✓

  ▷  … : ✓

  ▷ Strategy of : ✓

  ▷ Strategy of : 

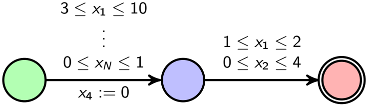
  ▷ : 

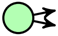
  ▷ Implementation (Python) 

- Future work

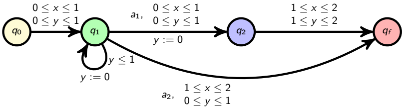
Achieved, ongoing and future works

- Achieved and ongoing works



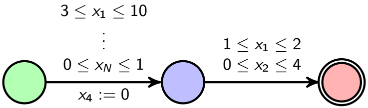
  - ▷ With: Op: max.
  - ▷ ⏲: ✓
  - ▷ ⏲ ... ⏲: ✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 
  - ▷ : 
  - ▷ Implementation (Python) 

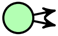
- Future work



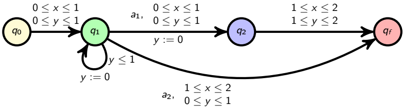
Achieved, ongoing and future works

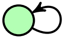
- Achieved and ongoing works



  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷  . . . : ✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 
  - ▷ : 
  - ▷ Implementation (Python) 

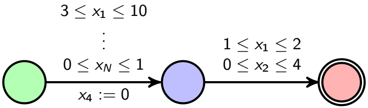
- Future work

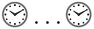
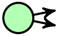


  - ▷ 

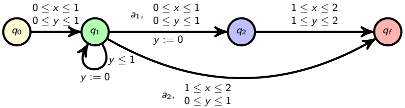
Achieved, ongoing and future works

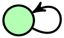
- Achieved and ongoing works



  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷ : ✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 
  - ▷ : 
  - ▷ Implementation (Python) 

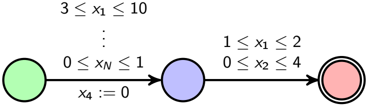
- Future work

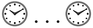
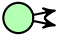


  - ▷ 
  - ▷ C++ implementation

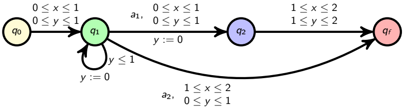
Achieved, ongoing and future works

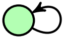
- Achieved and ongoing works



  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷ : ✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 
  - ▷ : 
  - ▷ Implementation (Python) 

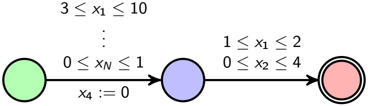
- Future work



  - ▷ 
  - ▷ C++ implementation
  - ▷ General penalty function.

Achieved, ongoing and future works

- Achieved and ongoing works



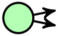
▷ With: Op: max.

▷ : ✓

▷ ...: ✓

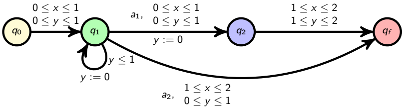
▷ Strategy of : ✓

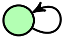
▷ Strategy of : 

▷ : 

▷ Implementation (Python) 

- Future work



▷ 

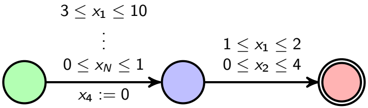
▷ C++ implementation

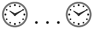
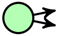
▷ General penalty function.

▷  ⟶ Stochastic opponent 

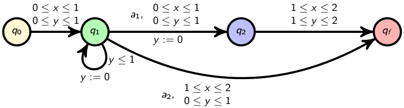
Achieved, ongoing and future works

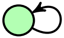
- Achieved and ongoing works



  - ▷ With: Op: max.
  - ▷ : ✓
  - ▷ :✓
  - ▷ Strategy of : ✓
  - ▷ Strategy of : 
  - ▷ : 
  - ▷ Implementation (Python) 

- Future work



  - ▷ 
  - ▷ C++ implementation
  - ▷ General penalty function.
  - ▷  $\longrightarrow$ Stochastic opponent 
  - ▷ Op: $+$, Ponderate $\sum$.

What is the penalty?

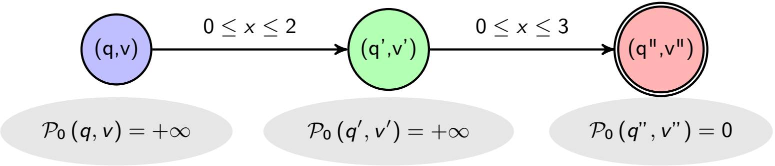
- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

$$\boxed{\mathcal{P}_i\left(q, v\right) = \min_{a \in \mathsf{Act}} \inf_{I \in \mathcal{D}(v, \mathsf{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta \in I} \mathcal{P}_{i-1}\left(succ\left(v, q, \delta, a\right)\right)\right)}$$

- Example of computation of $\mathcal{P}_i\left(q, v\right)$ (Our model)

Step $i = 0$, Initialization



What is the penalty?

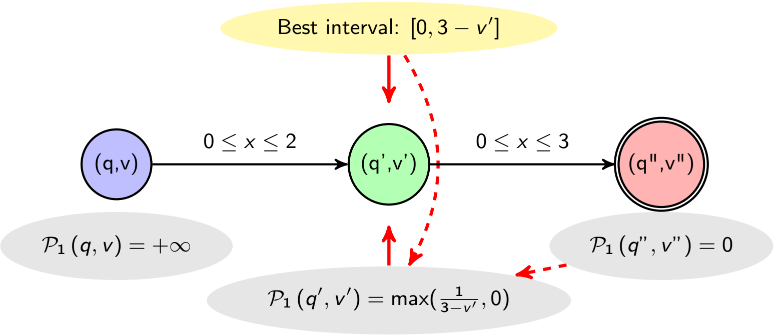
- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

$$\boxed{\mathcal{P}_i\,(q,v) = \min_{a\in\mathsf{Act}}\ \inf_{I\in\mathcal{D}(v,\mathsf{inv}(q))}\ \max\left(\frac{1}{|I|}, \sup_{\delta\in I}\mathcal{P}_{i-1}\,(succ\,(v,q,\delta,a))\right)}$$

- Example of computation of $\mathcal{P}_i\,(q,v)$ (Our model)

Step $i = 1$, Comparison



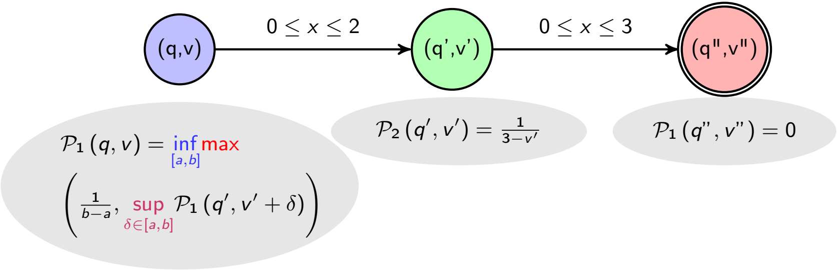
What is the penalty?

- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

$$\boxed{\mathcal{P}_i\left(q, v\right) = \min_{a \in \mathsf{Act}} \inf_{I \in \mathcal{D}(v, \mathsf{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta \in I} \mathcal{P}_{i-1}\left(succ\left(v, q, \delta, a\right)\right)\right)}$$

- Example of computation of $\mathcal{P}_i\left(q, v\right)$ (Our model)



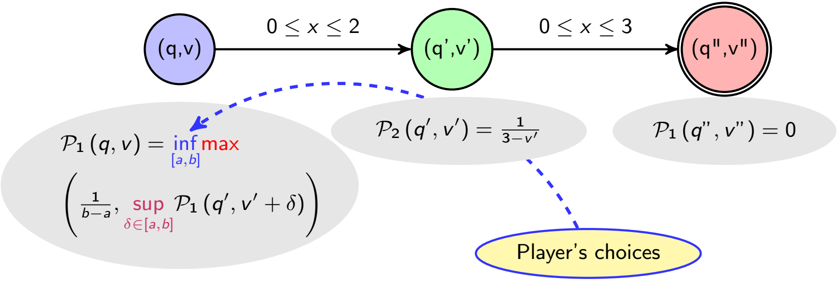
What is the penalty?

- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

$$\boxed{\mathcal{P}_i(q,v) = \min_{a \in \mathsf{Act}} \inf_{I \in \mathcal{D}(v, \mathsf{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta \in I} \mathcal{P}_{i-1}(succ(v,q,\delta,a))\right)}$$

- Example of computation of $\mathcal{P}_i(q, v)$ (Our model)



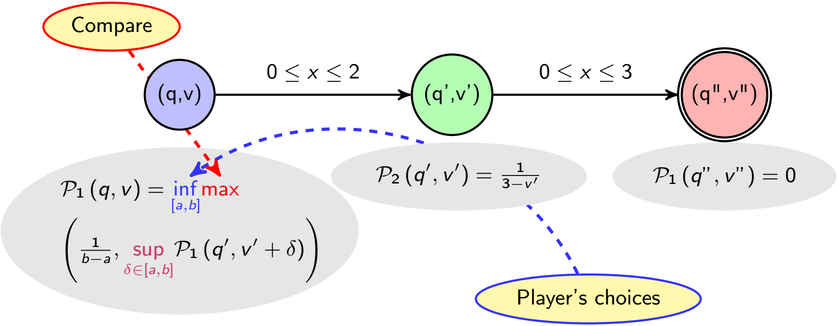
What is the penalty?

- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

$$\boxed{\mathcal{P}_i\left(q,v\right) = \min_{a\in \mathsf{Act}} \inf_{I\in\mathcal{D}(v,\mathsf{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta\in I} \mathcal{P}_{i-1}\left(succ\left(v,q,\delta,a\right)\right)\right)}$$

- Example of computation of $\mathcal{P}_i\left(q,v\right)$ (Our model)



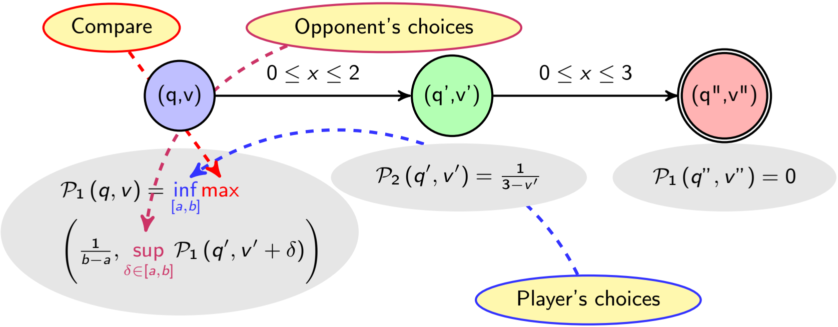
What is the penalty?

- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

$$\boxed{\mathcal{P}_i\left(q, v\right) = \min_{a \in \mathsf{Act}} \inf_{I \in \mathcal{D}(v, \mathsf{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta \in I} \mathcal{P}_{i-1}\left(succ\left(v, q, \delta, a\right)\right)\right)}$$

- Example of computation of $\mathcal{P}_i\left(q, v\right)$ (Our model)



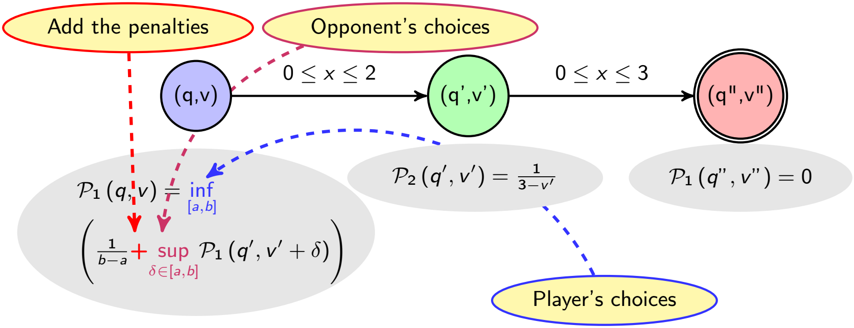
What is the penalty?

- The penalty for our model

For any winning configuration (q, v), for $i > 0$,

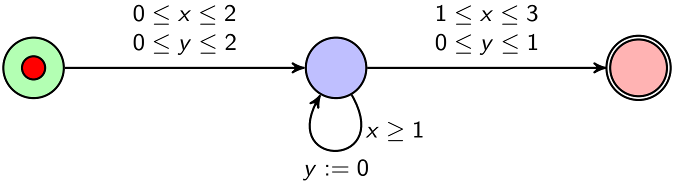
$$\boxed{\mathcal{P}_i(q, v) = \min_{a \in \mathrm{Act}} \inf_{I \in \mathcal{D}(v, \mathrm{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta \in I} \mathcal{P}_{i-1}(succ(v, q, \delta, a))\right)}$$

- Example of computation of $\mathcal{P}_i(q, v)$ (Bouyer et al.)



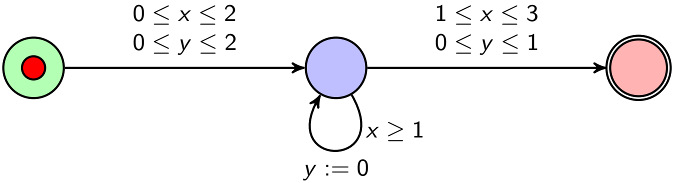
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(0, 0)$.



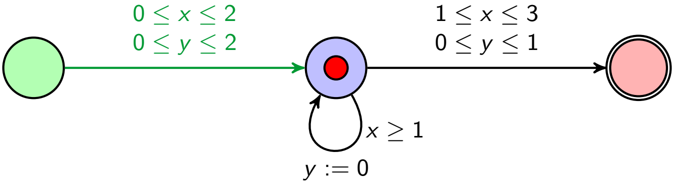
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(0, 0)$.
  - We propose a delay $\delta = 1.5$



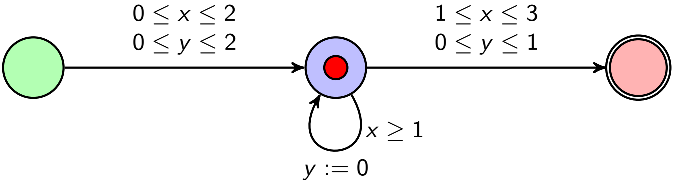
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(1.5, 1.5)$.



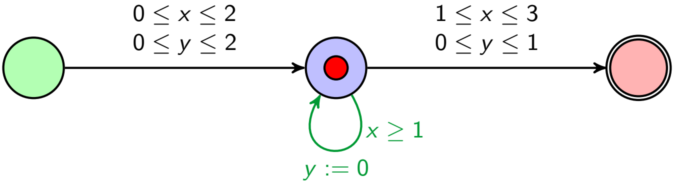
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(1.5, 1.5)$.
  - We propose a delay $\delta = 1$



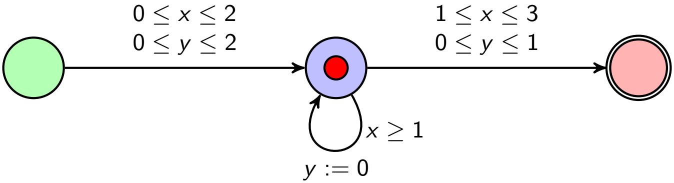
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(2.5, 0)$.



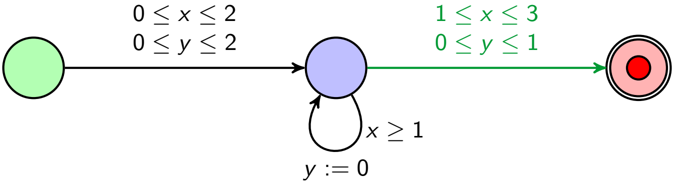
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(2.5, 0)$.
  - We propose a delay $\delta = 0.5$



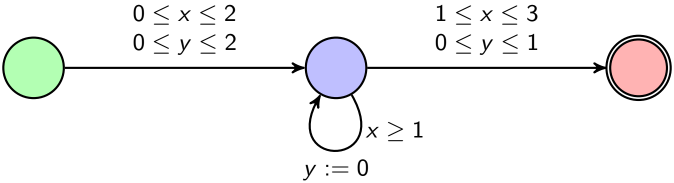
Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(3, 0.5)$.



Example of multi-clocks automaton

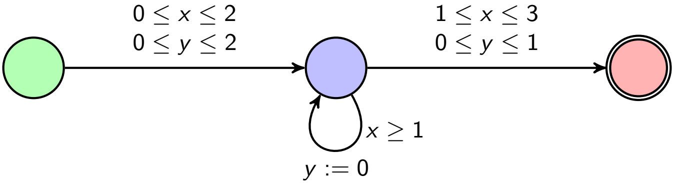
- A two-clock automaton
  - Clock (x, y) values $(3, 0.5)$.



Our goal:

Example of multi-clocks automaton

- A two-clock automaton
  - Clock (x, y) values $(3, 0.5)$.

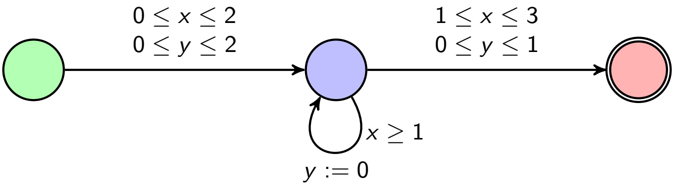


Our goal:

- Reachability ✓

Example of multi-clocks automaton

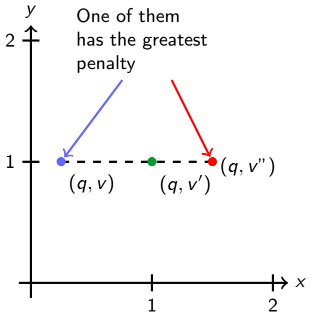
- A two-clock automaton
  - Clock (x, y) values $(3, 0.5)$.



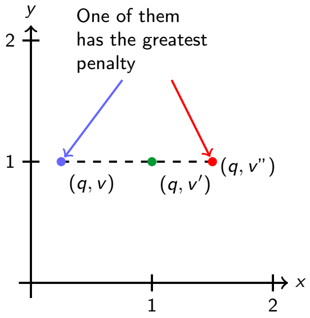
Our goal:

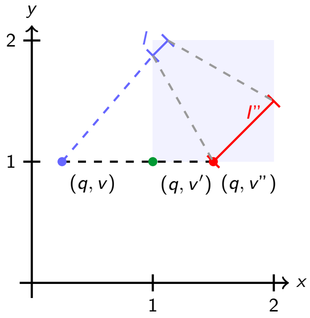
- Reachability ✓
- Robustness ✓

Proof of our strategy



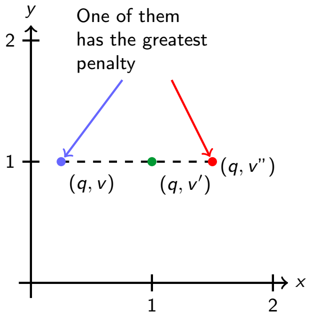
Proof of our strategy





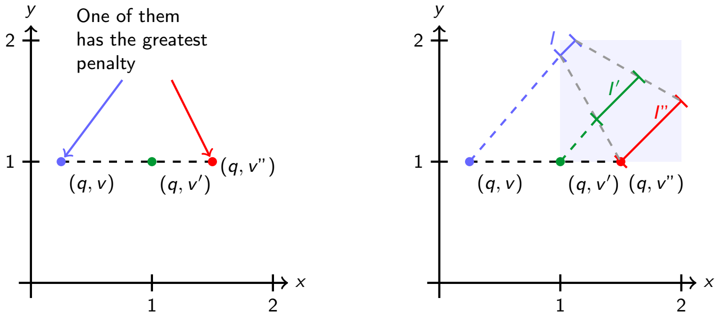
- Step of the proof
  - ▷ Take two arbitrary (enabled) interval I, I''.

Proof of our strategy



- Step of the proof
  - ▷ Take two arbitrary (enabled) interval $I, I"$.
  - ▷ Construct I' s.t the inequality works.

Proof of our strategy



- Step of the proof
  - ▷ Take two arbitrary (enabled) interval $I, I"$.
  - ▷ Construct I' s.t the inequality works.
  - ▷ Take the optimum intervals $I, I"$.