Computing maximally-permissive strategies in *timed games*

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

[1]INRIA - Mitsubishi Electric, Rennes

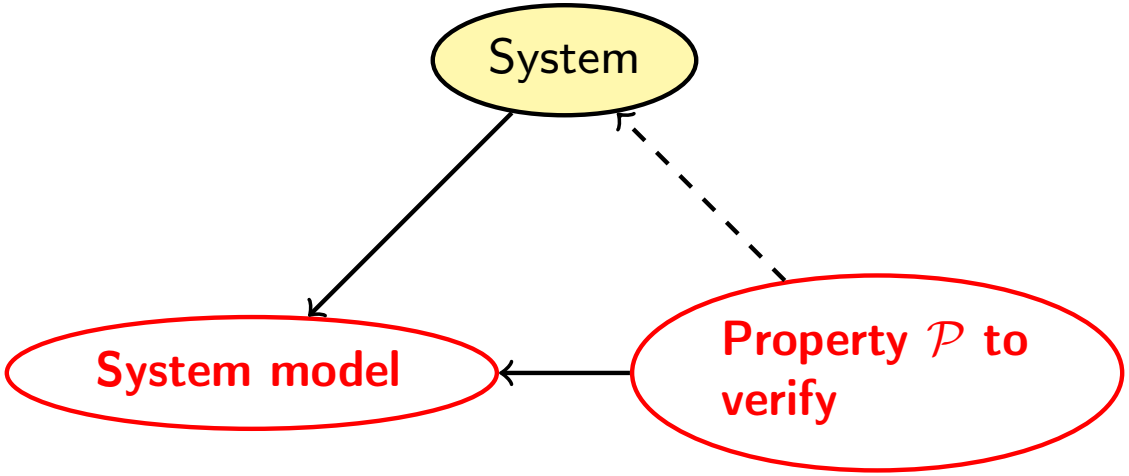
[2]INRIA-IRISA, Rennes

[3]CNRS-IRISA, Rennes

[4]Mitsubishi Electric, Rennes

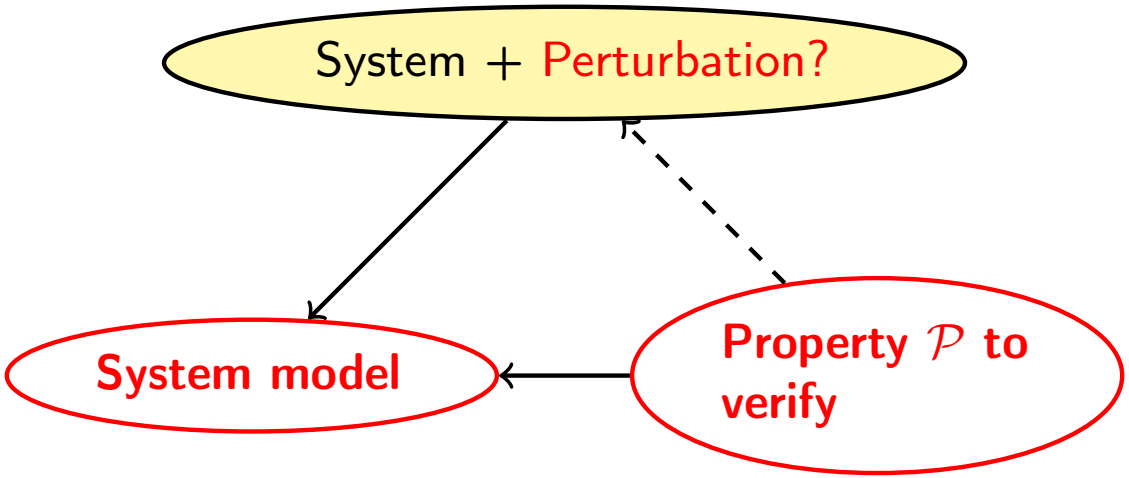
March 29, 2023

Motivations: Verify properties despite perturbations



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

Motivations: Verify properties despite perturbations

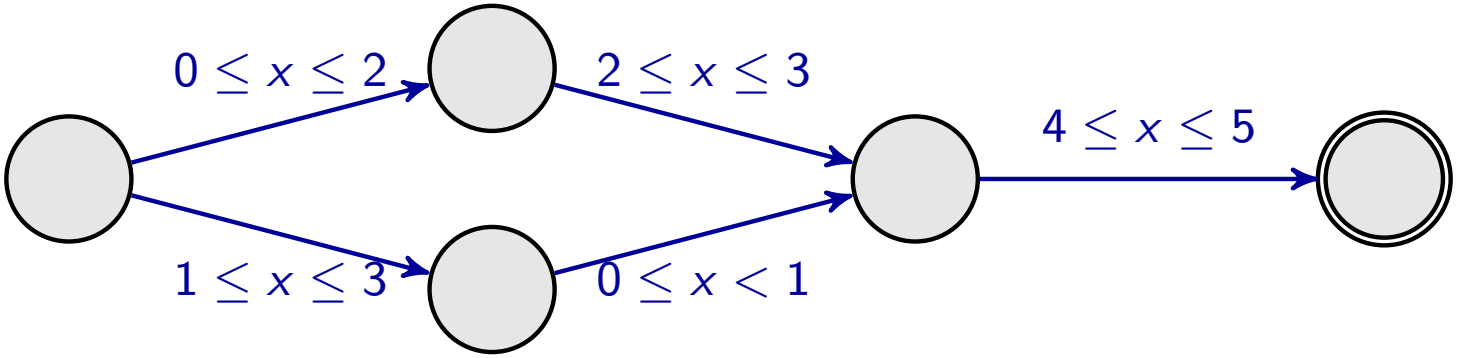


- How to model it? Timed automata + Verification of $\mathcal{P}$.
- Our goal? Verify **with robustness**.

Table of contents

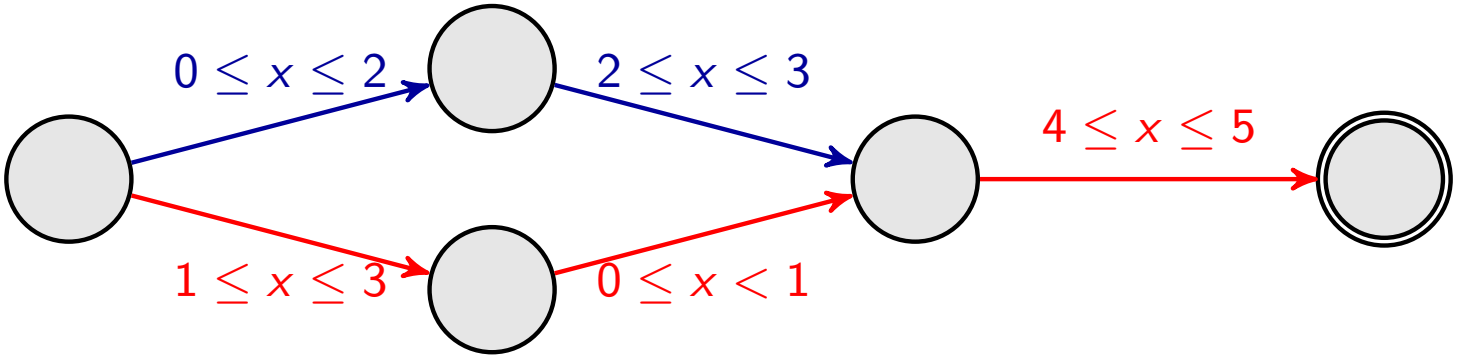
Issues for reachability & robustness

- reachability ?



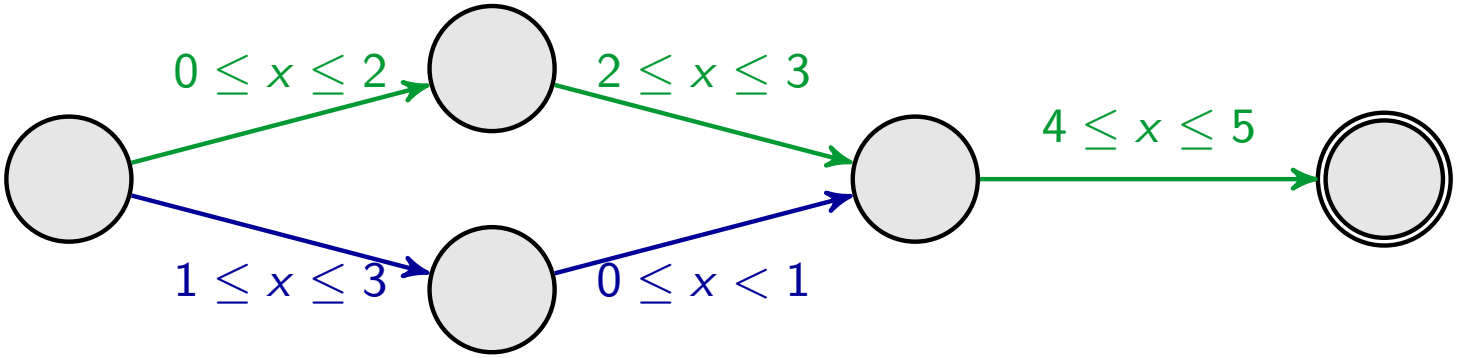
Issues for reachability & robustness

- reachability



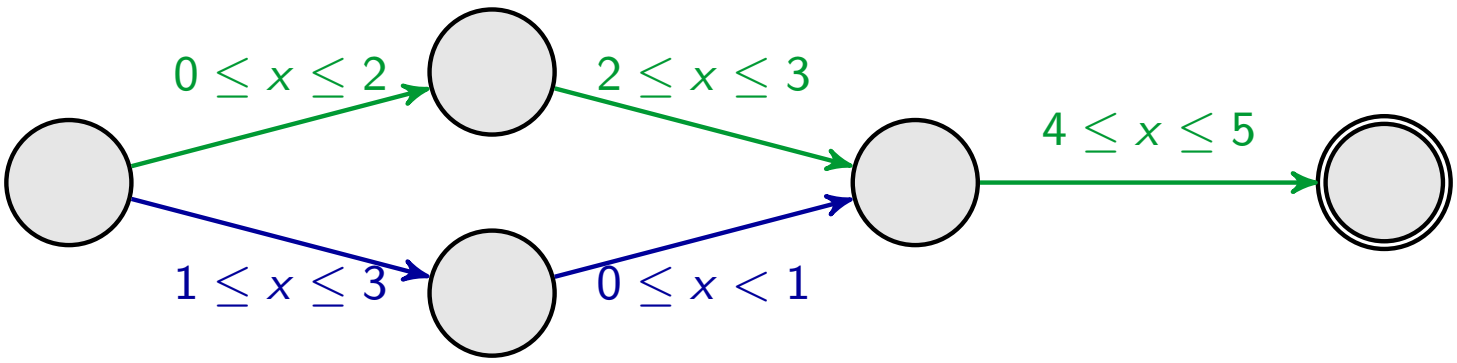
Issues for reachability & robustness

- reachability ✓

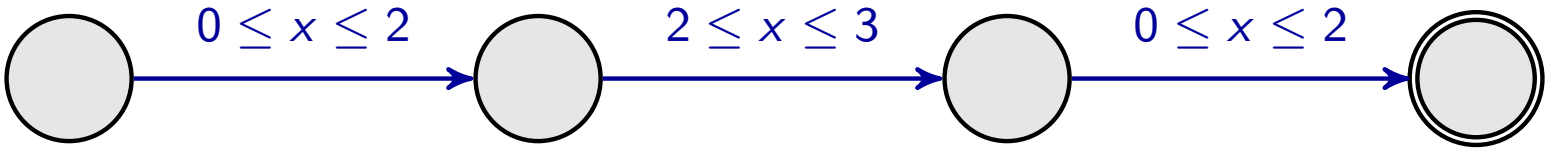


Issues for reachability & robustness

- reachability ✓

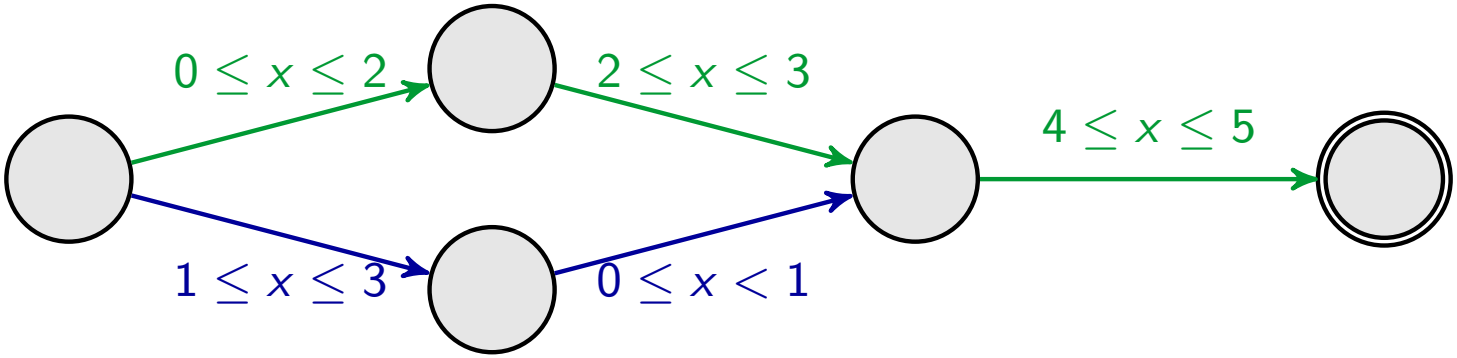


- Robustness ?

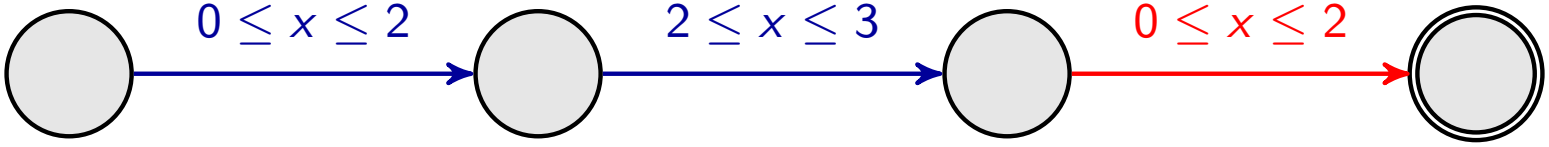


Issues for reachability & robustness

- reachability ✓

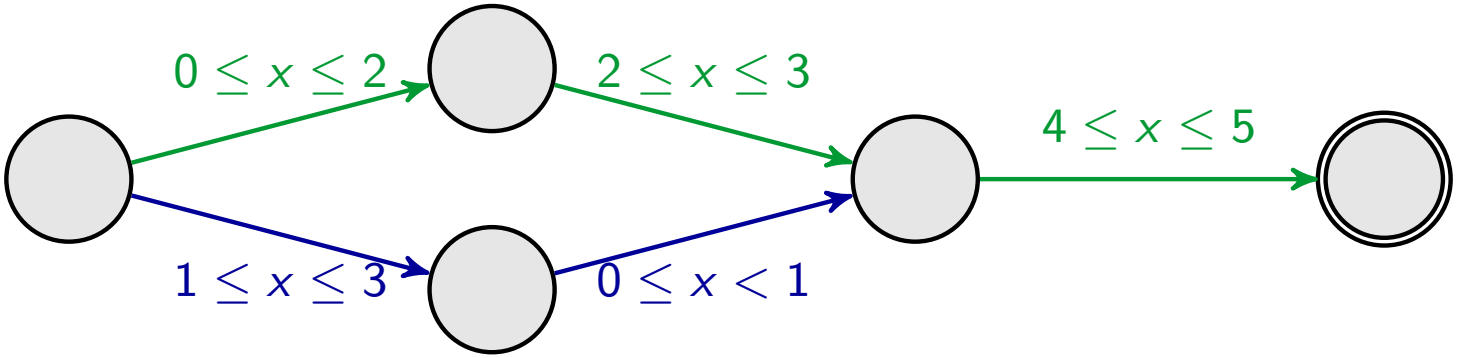


- Robustness ⛔

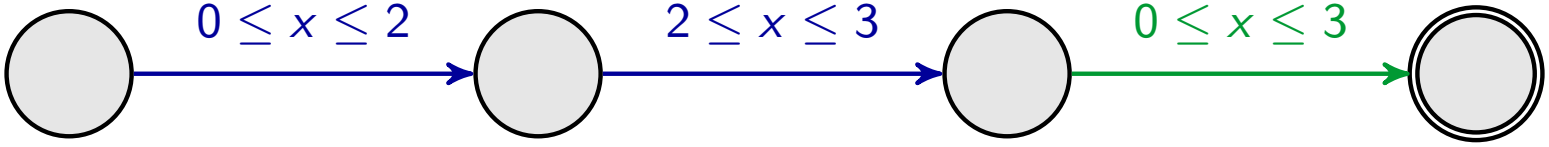


Issues for reachability & robustness

- reachability ✓



- Robustness ✓



Our goal

- Define a semantic of robustness

Our goal

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

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

Our goal

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

For $p \in \mathbb{R}$, a timed automaton 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

Robustness: state of the art

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

Robustness: state of the art

- 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.

Robustness: state of the art

- 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

Robustness: state of the art

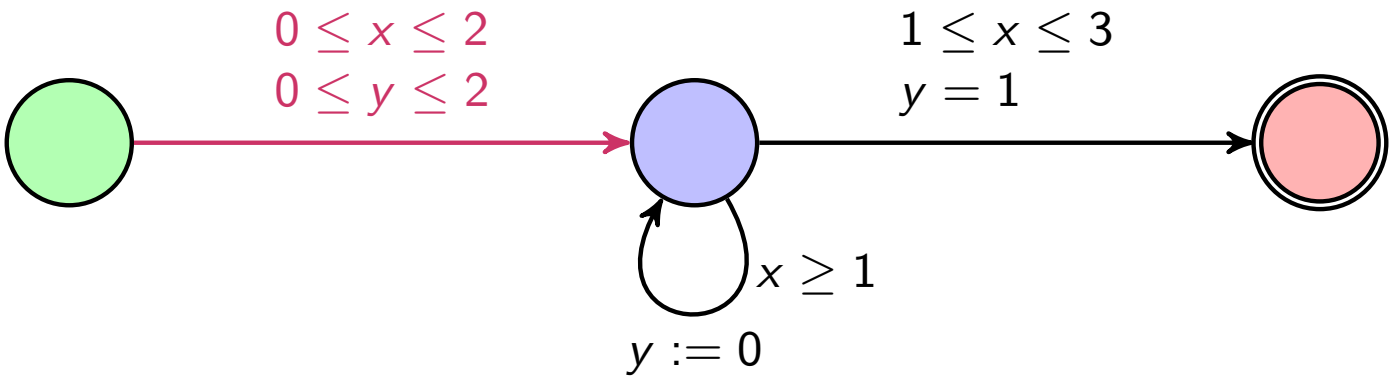
- Topological robustness
  - ▷ Gupta, Hezinger, Jagadeesan "Robust Timed Automata", 1997
  - ▷ Tools: stability theorems.
- Guard enlargement
  - ▷ Sankur "Robustness in Timed Automata", PhD Thesis, 2013
  - ▷ Tools: game theory.
- Delay enlargement
  - ▷ Bouyer, Fang, Markey "Permissive strategies in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

Robustness: state of the art

- 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
  - ▷ An algorithm to compute robustness: ✓
  - ▷ Multiple clocks: ✗.

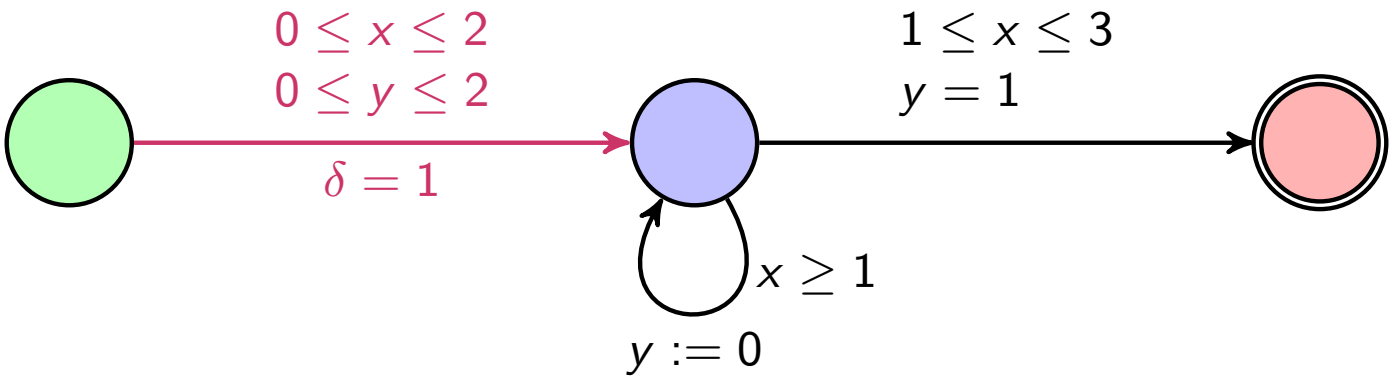
Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement



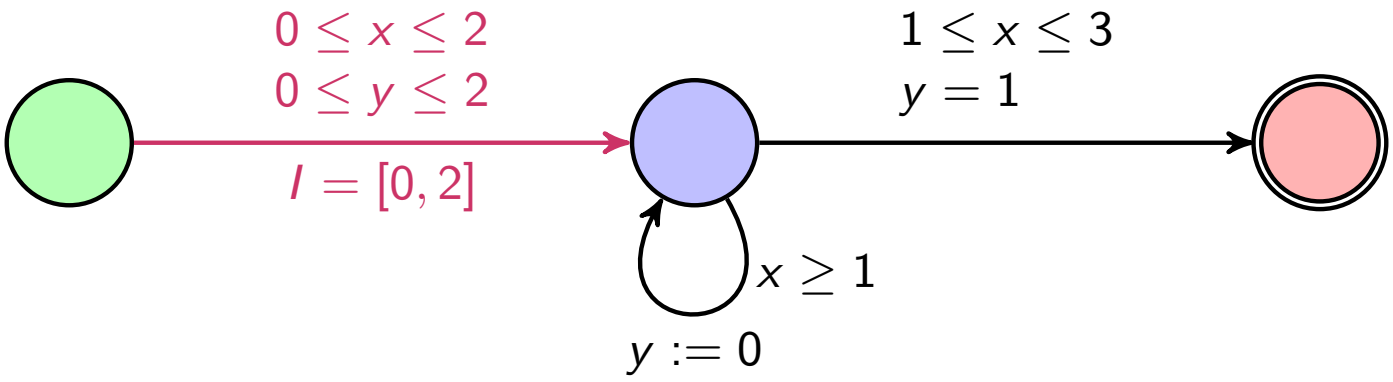
Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement



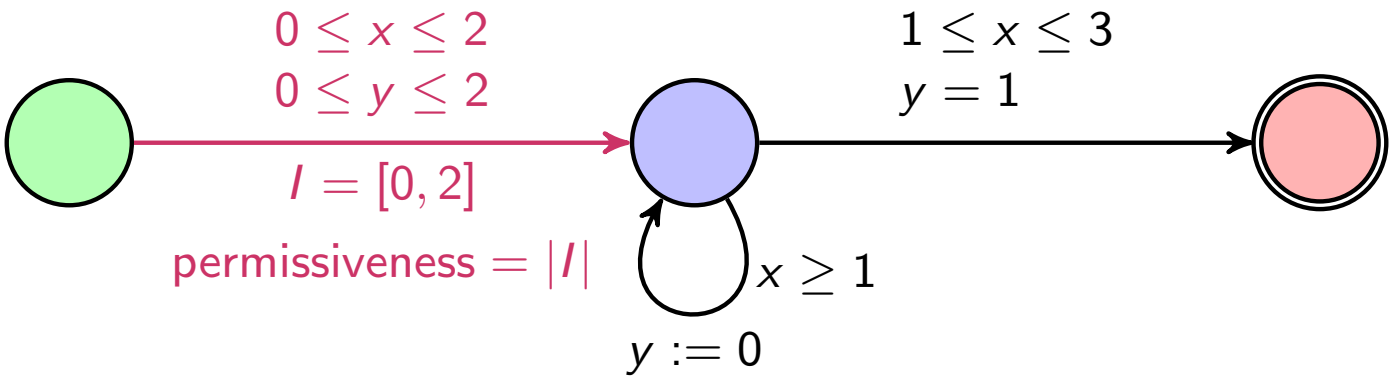
Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement



Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement



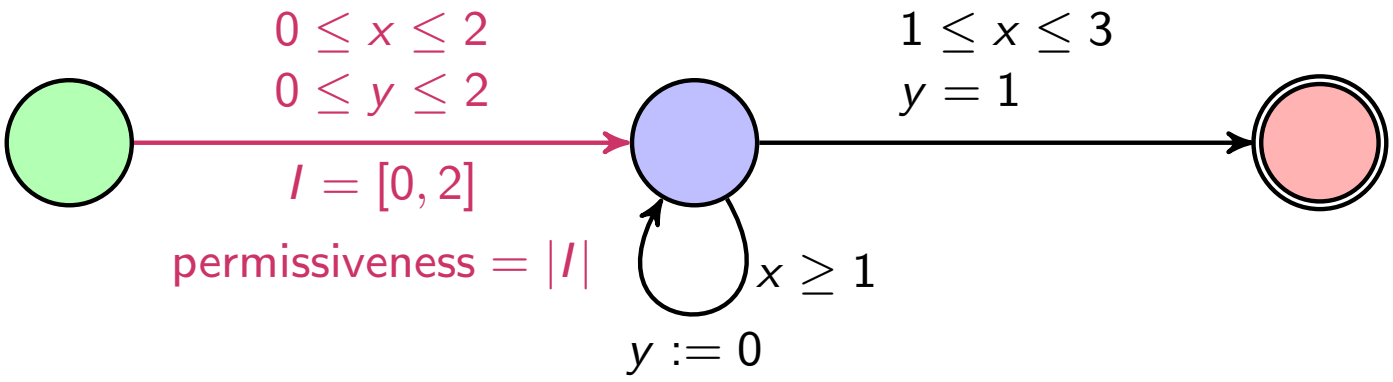
- How to model the "Best case/Worst case"?

Choice of Interval I

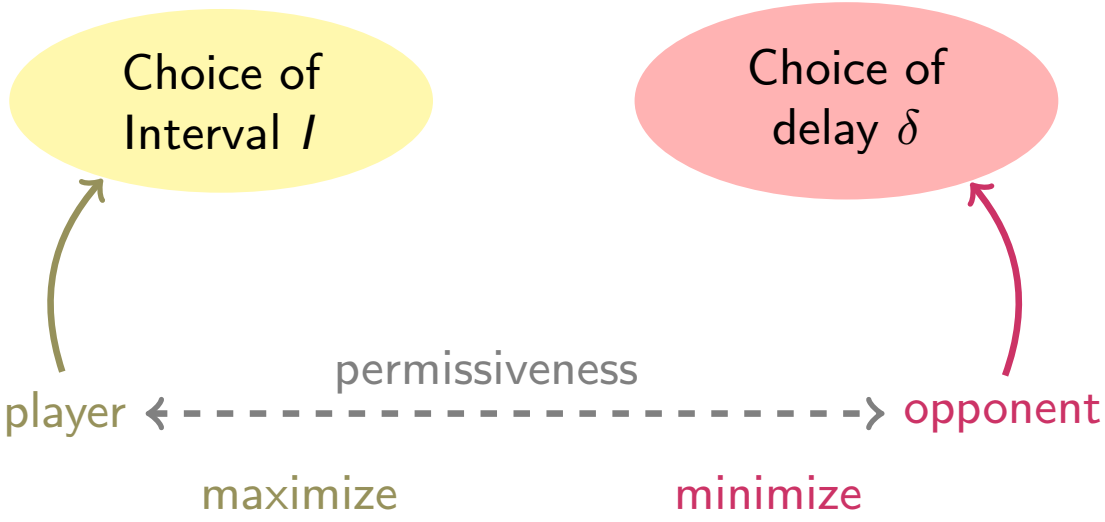
Choice of delay δ

Quantifying robustness: The example of delay enlargement

- Delay/ No delay enlargement

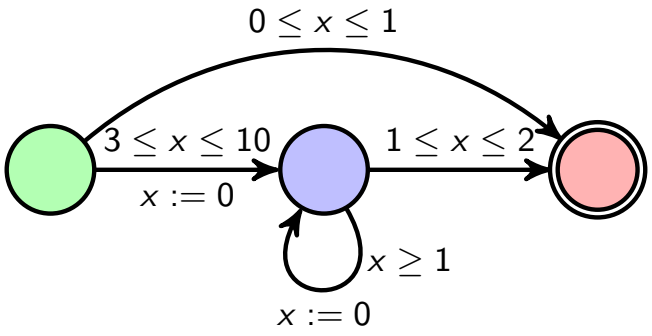


- How to model the "Best case/Worst case"?

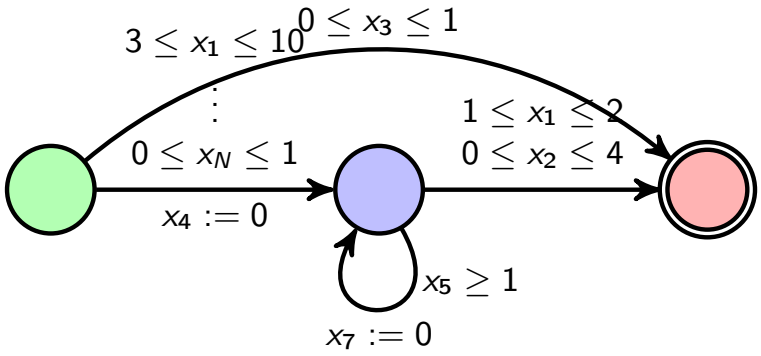


State of the art for delay enlargement: Bouyer et al. vs our model

- Bouyer et al.

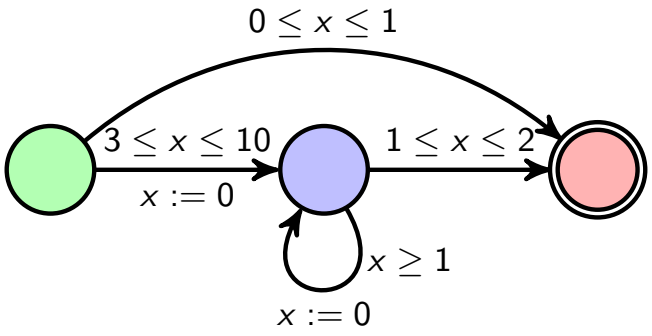


- Our model



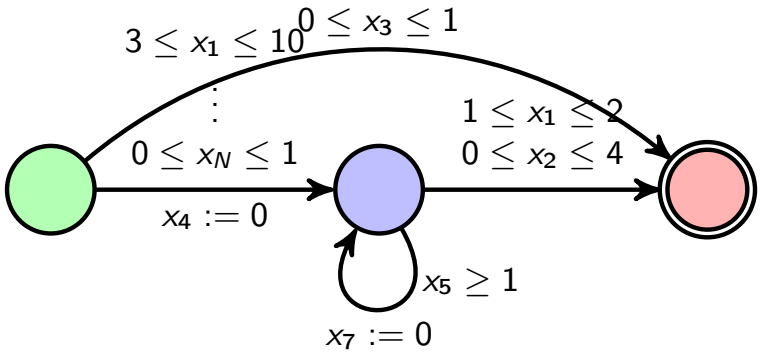
State of the art for delay enlargement: Bouyer et al. vs our model

- Bouyer et al.



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

- Our model



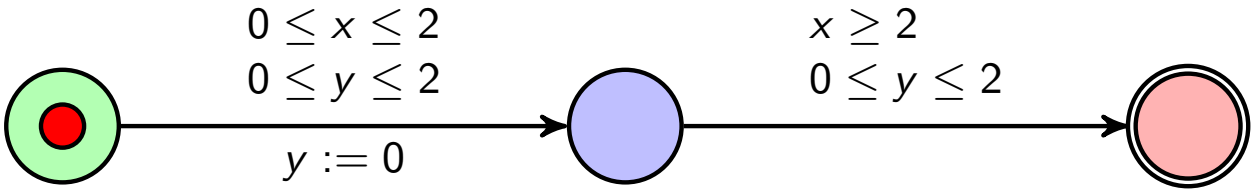
  - ▷ Op: min: ✓
  - ▷ 🕒: ✓
  - ▷ 🕒 . . . 🕒: ✓

Table of contents

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(0, 0)$.



Player

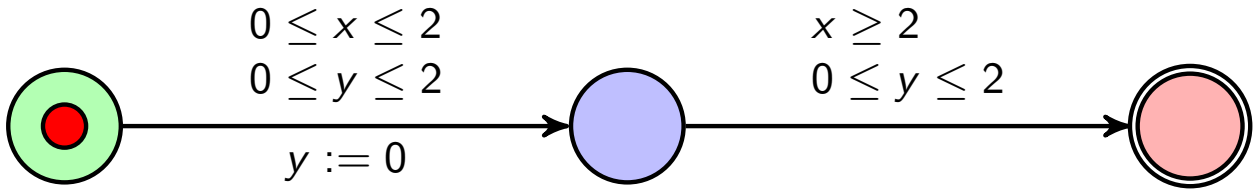
permissiveness

Opponent

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(0, 0)$.



Player

I play $[0, 2]$

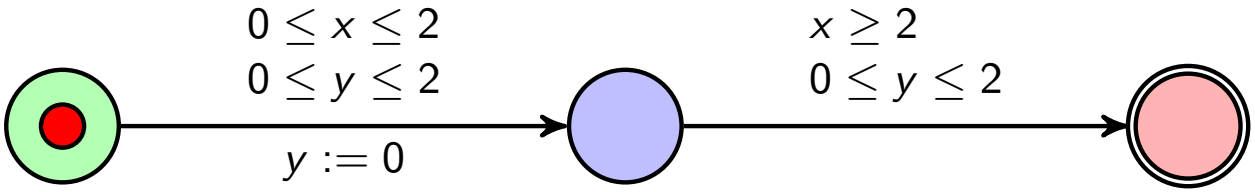
permissiveness

Opponent

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(0, 0)$.



Player

I play $[0, 2]$

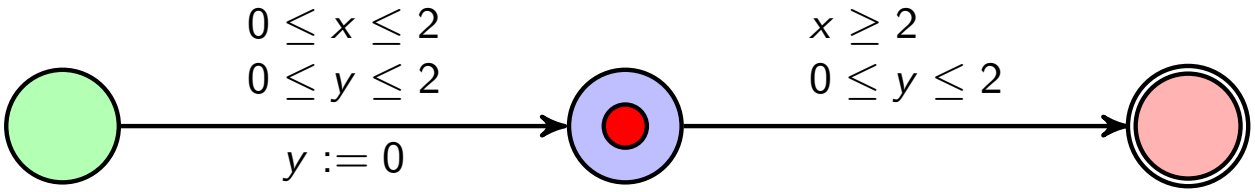
permissiveness $= 2$

Opponent

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(0, 0)$.



Player

I play $[0, 2]$

permissiveness $= 2$

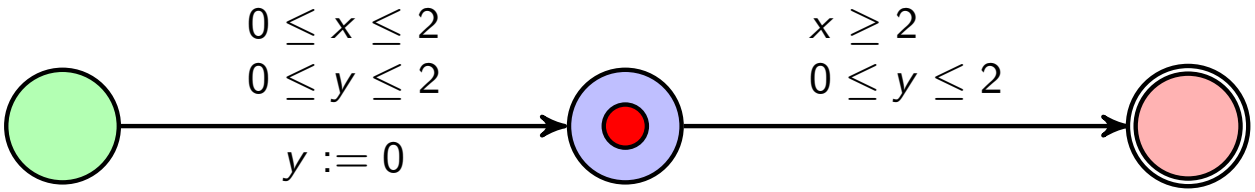
Opponent

I choose $\delta = 0$

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(0, 0)$.



Player

I play $[0, 2]$

I play $\{2\}$

permissiveness $= 2$

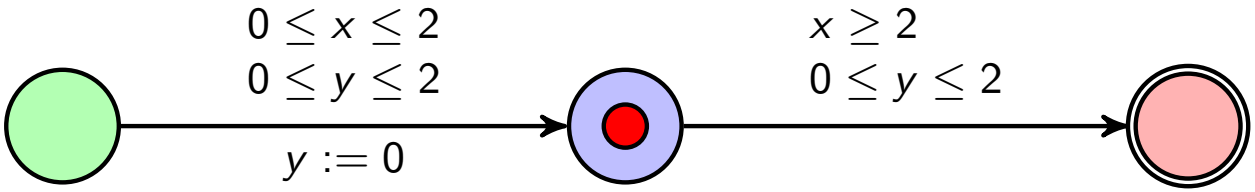
Opponent

I choose $\delta = 0$

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(0, 0)$.



Player

I play $[0, 2]$

I play $\{2\}$

permissiveness $= 2$

$= 0$

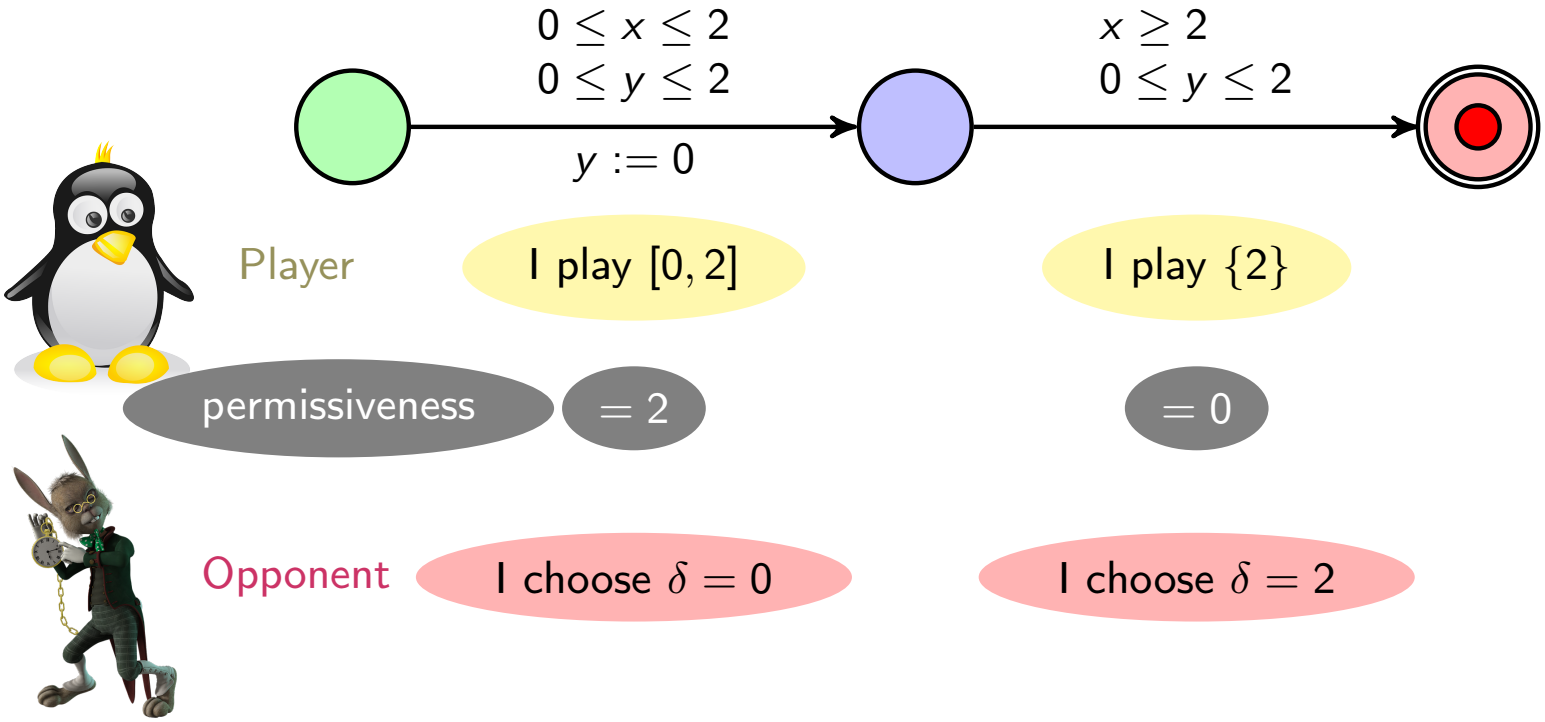
Opponent

I choose $\delta = 0$

An example of permissiveness computation

- Are we 1−robust? Let's lose!

value: $(2, 2)$.



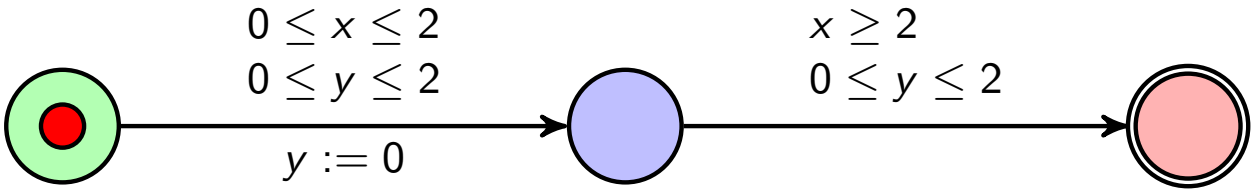
- Final permissiveness

$\min(2, 0) = 0$:

An example of permissiveness computation

- Are we 1−robust? Let's win!

value: $(0, 0)$.



Player

I play $[1, 2]$

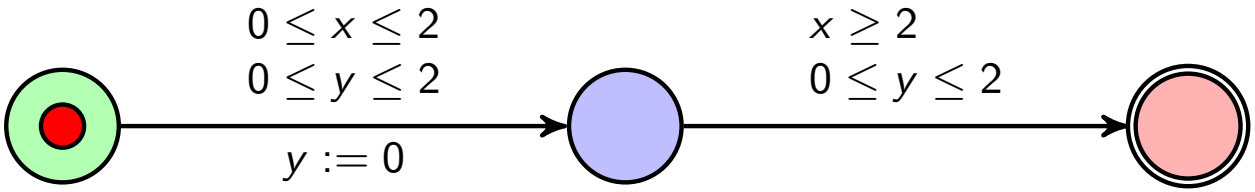
permissiveness

Opponent

An example of permissiveness computation

- Are we 1−robust? Let's win!

value: $(0, 0)$.



Player

I play $[1, 2]$

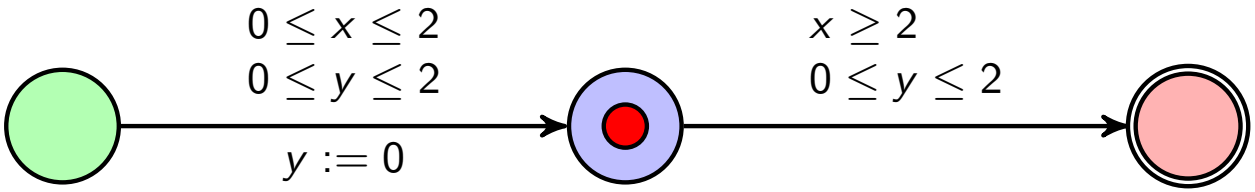
permissiveness $= 1$

Opponent

An example of permissiveness computation

- Are we 1−robust? Let's win!

value: $(1, 0)$.



Player

I play $[1, 2]$

permissiveness $= 1$

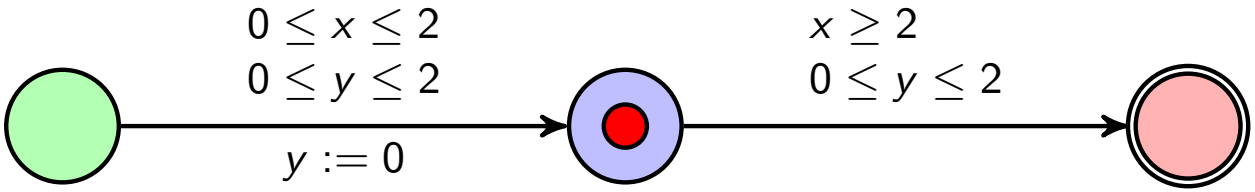
Opponent

I choose $\delta = 1$

An example of permissiveness computation

- Are we 1−robust? Let's win!

value: $(1, 0)$.



Player

I play $[1, 2]$

I play $[1, 2]$

permissiveness $= 1$

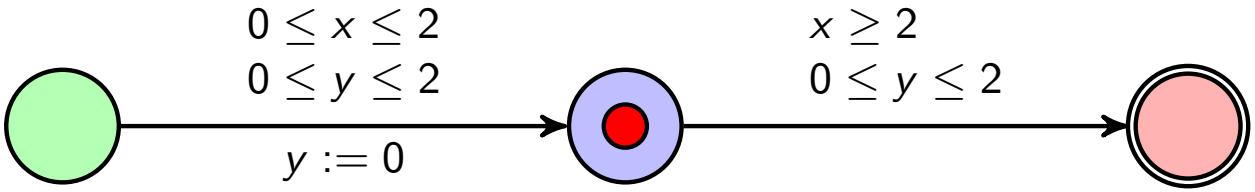
Opponent

I choose $\delta = 1$

An example of permissiveness computation

- Are we 1−robust? Let's win!

value: $(1, 0)$.



Player

I play $[1, 2]$

I play $[1, 2]$

permissiveness $= 1$

$= 1$

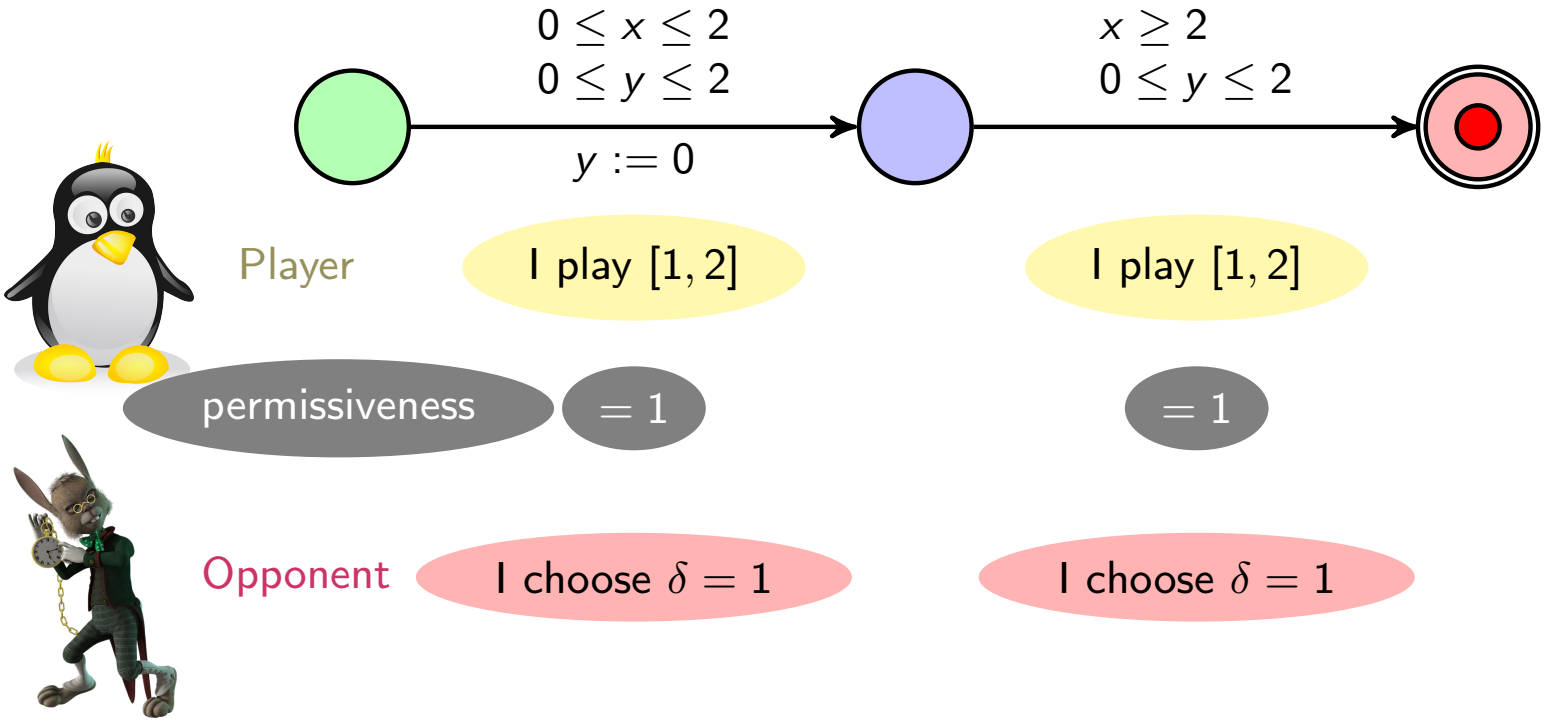
Opponent

I choose $\delta = 1$

An example of permissiveness computation

- Are we 1−robust? Let's win!

value: $(2, 1)$.



- Final permissiveness

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

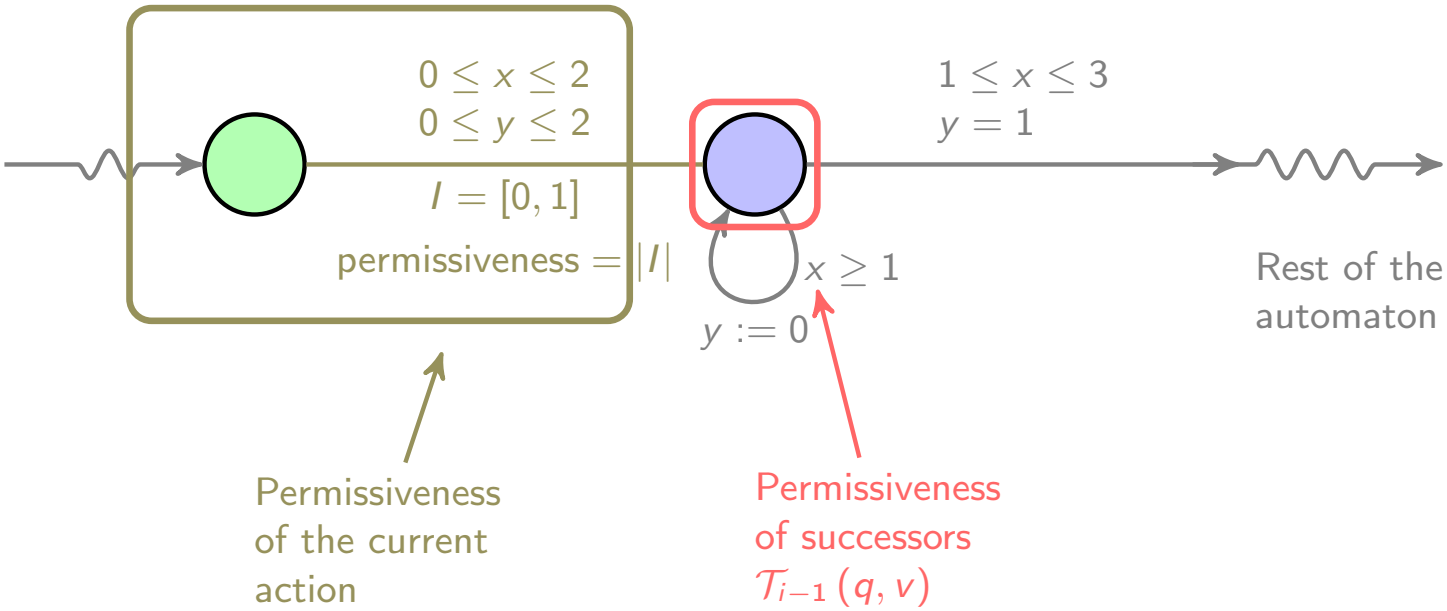
What is the permissiveness?

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

What is the permissiveness?

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

- An algorithm to compute the permissiveness

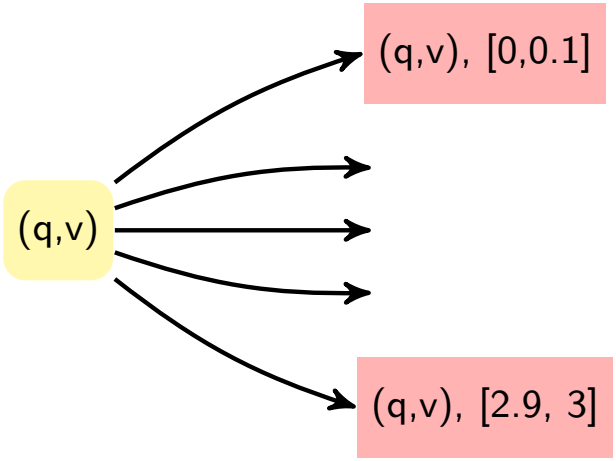


Permissiveness of the automaton: minimum of current permissiveness and the permissiveness of the successors

What is the permissiveness?

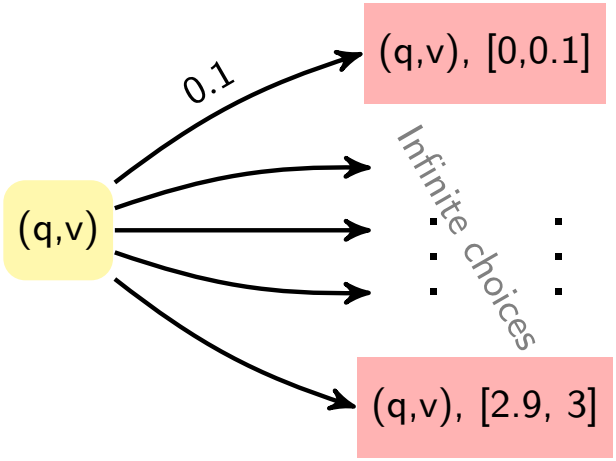
What is the permissiveness?

Guard $0 \leq x \leq 3$



What is the permissiveness?

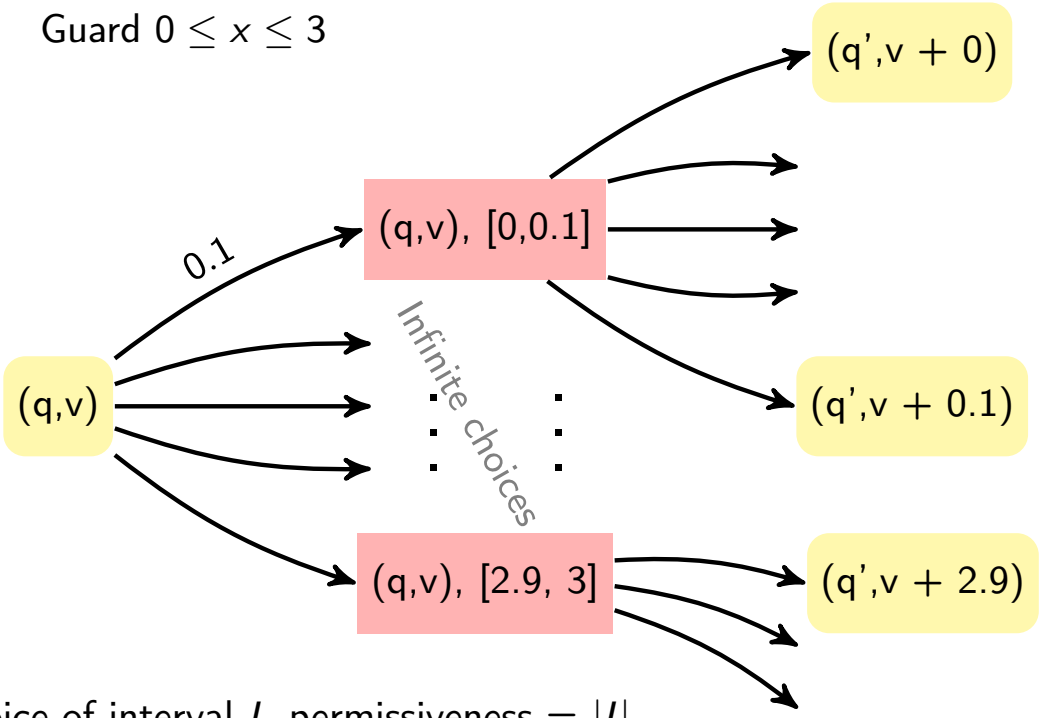
Guard $0 \leq x \leq 3$



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

Current action/permissiveness

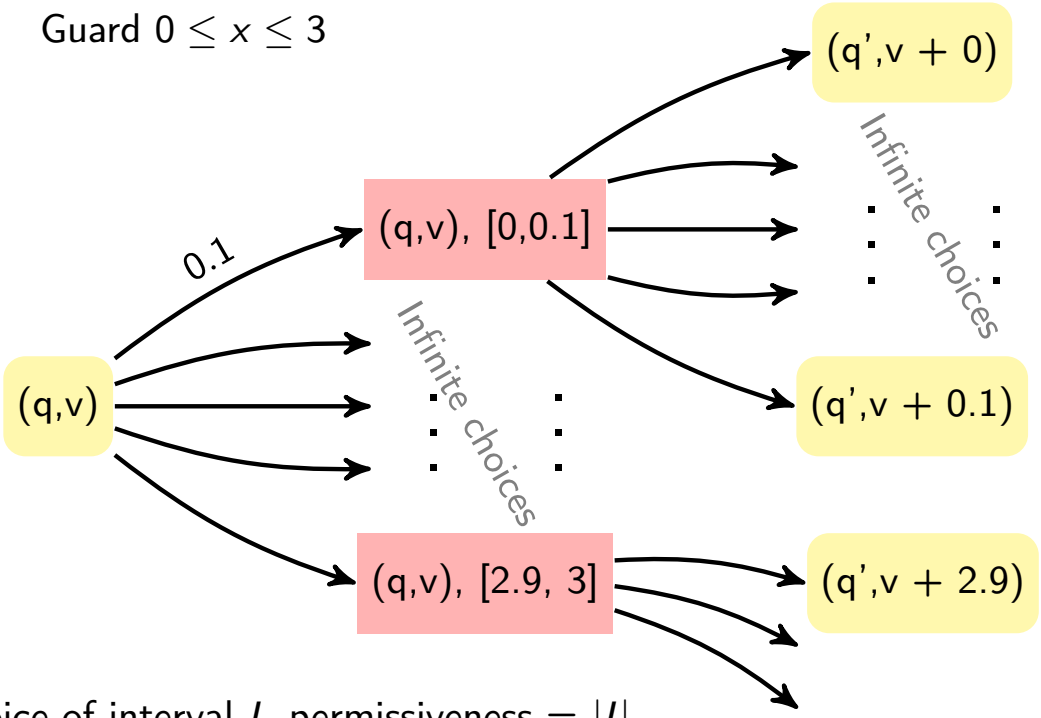
What is the permissiveness?



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

Current action/permissiveness

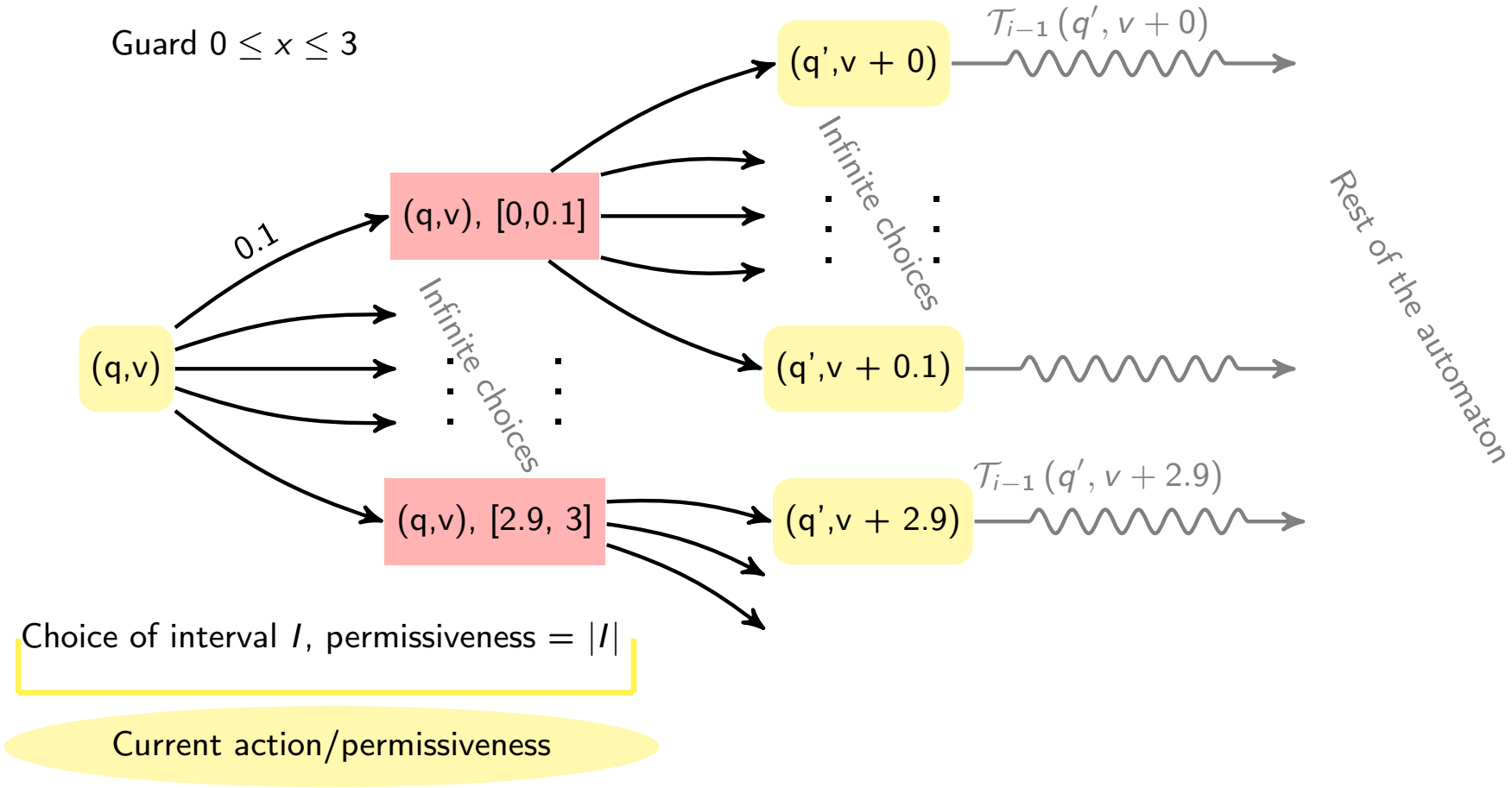
What is the permissiveness?



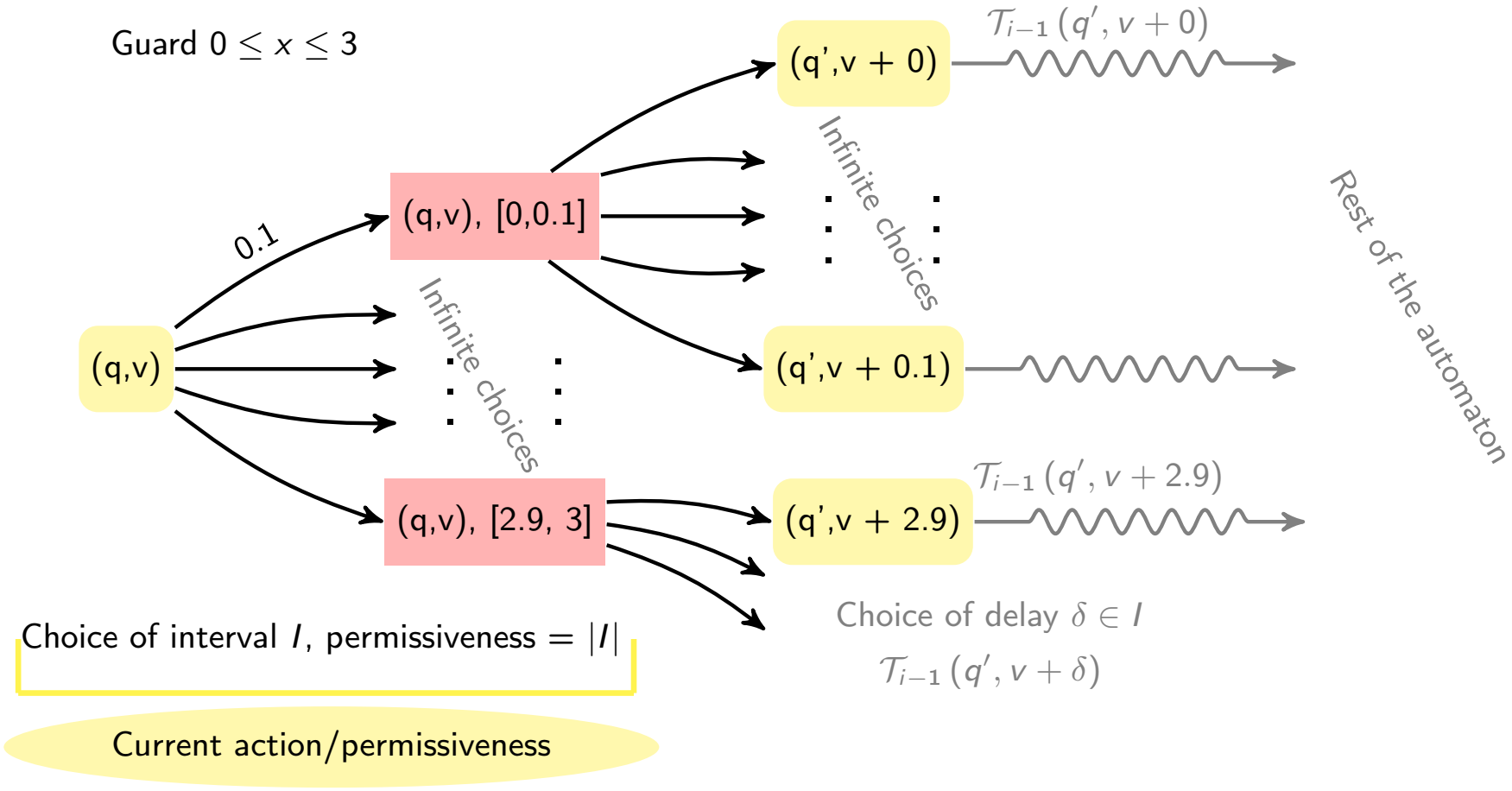
Choice of interval I, permissiveness $= |I|$

Current action/permissiveness

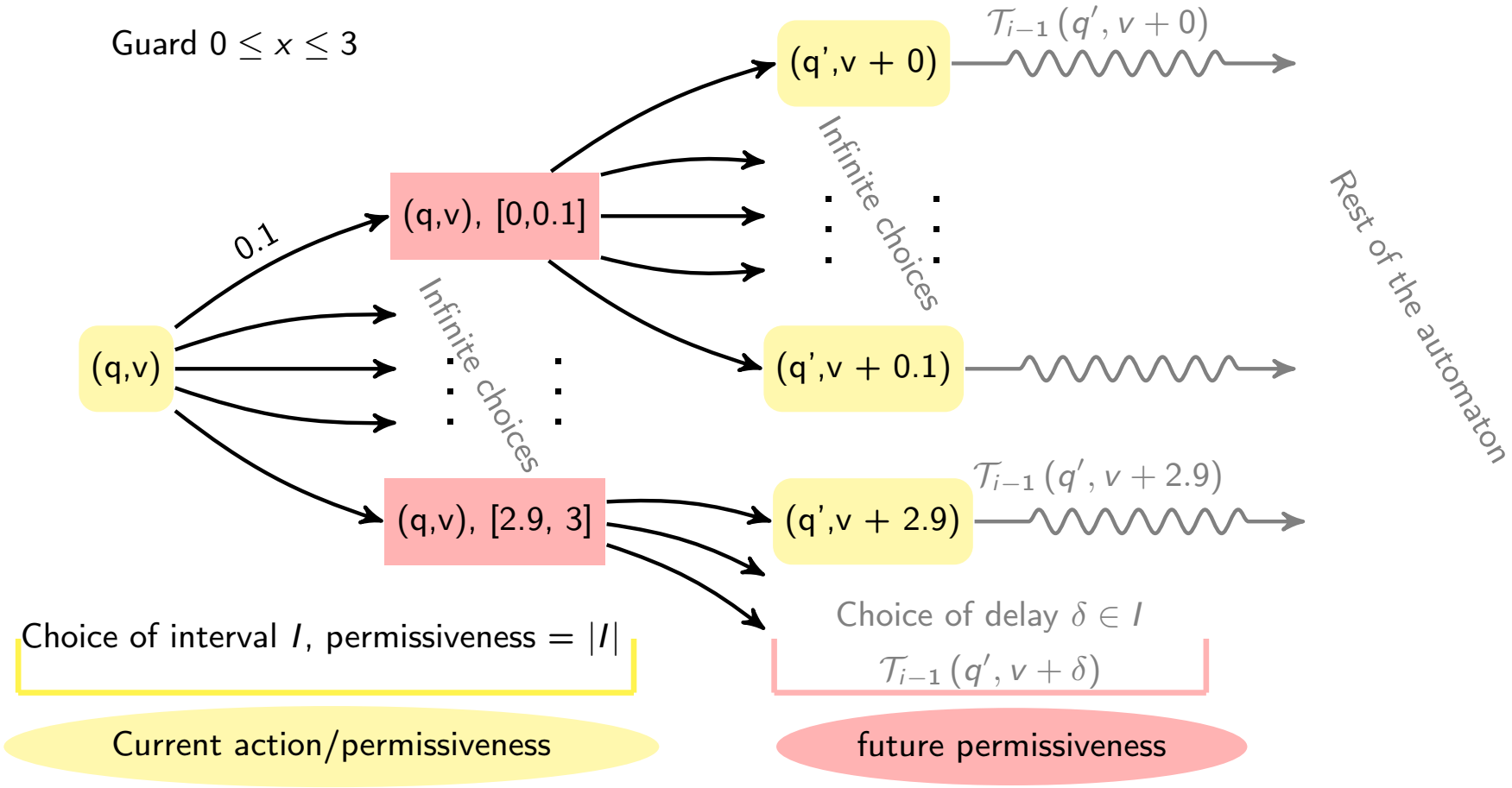
What is the permissiveness?



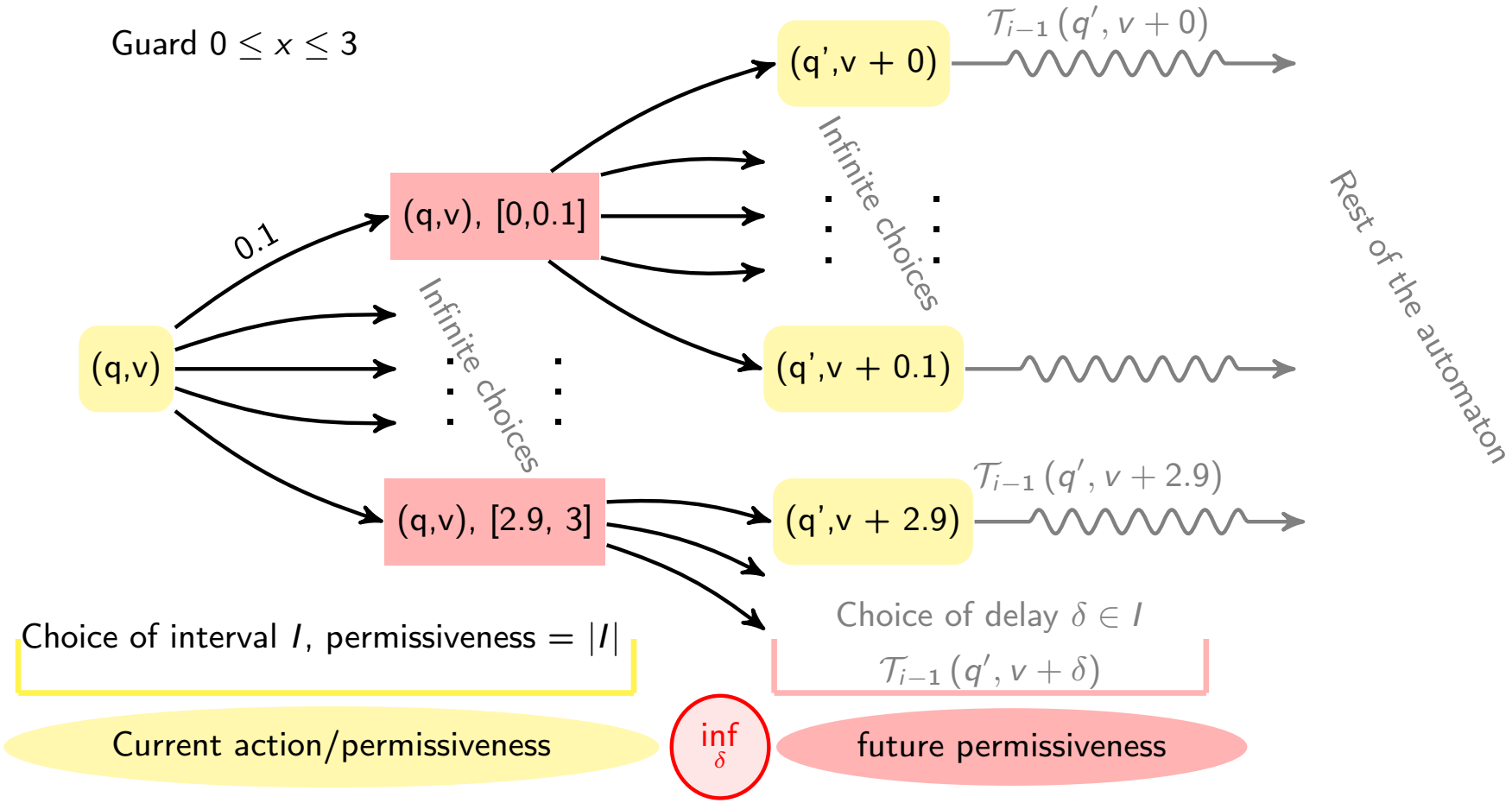
What is the permissiveness?



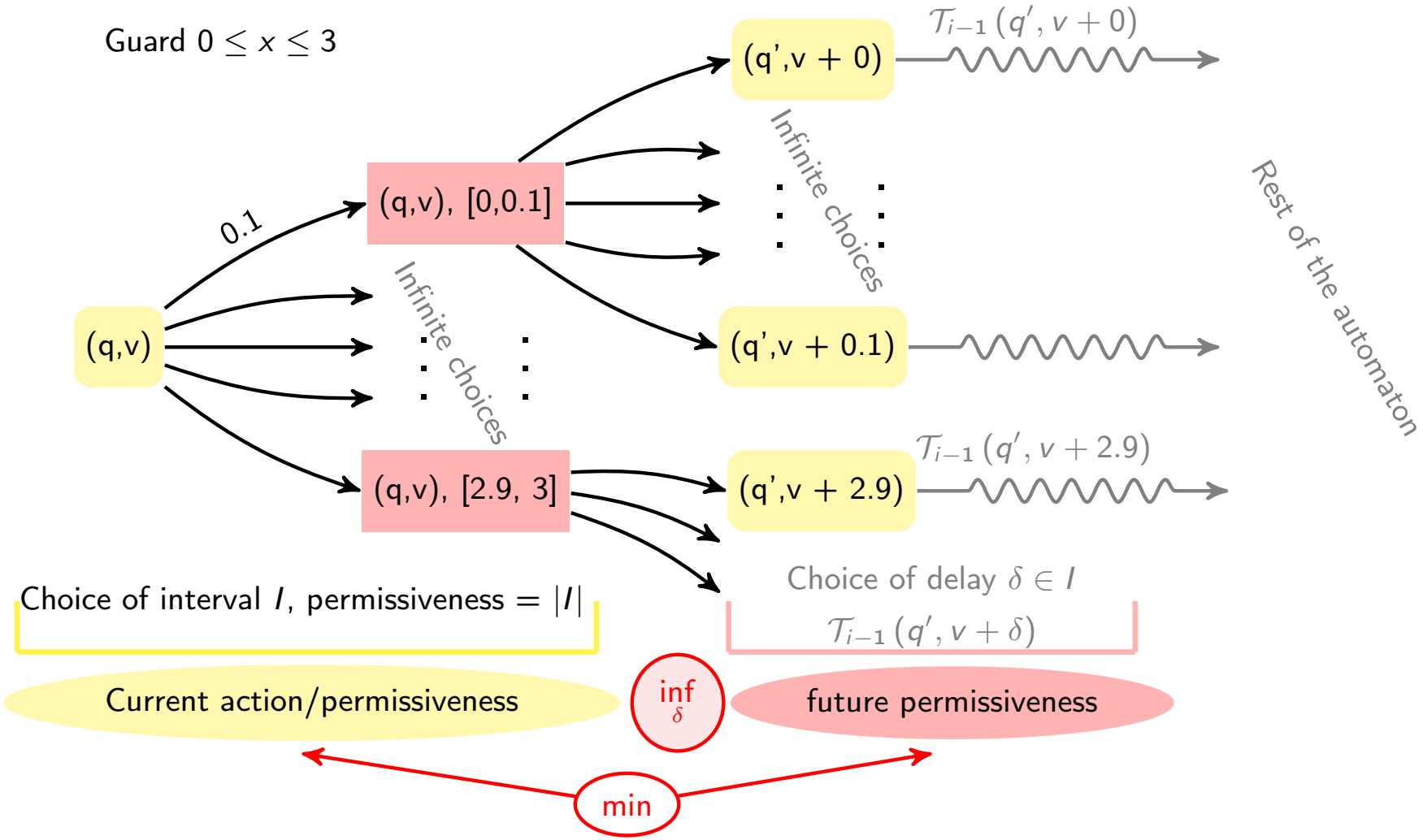
What is the permissiveness?



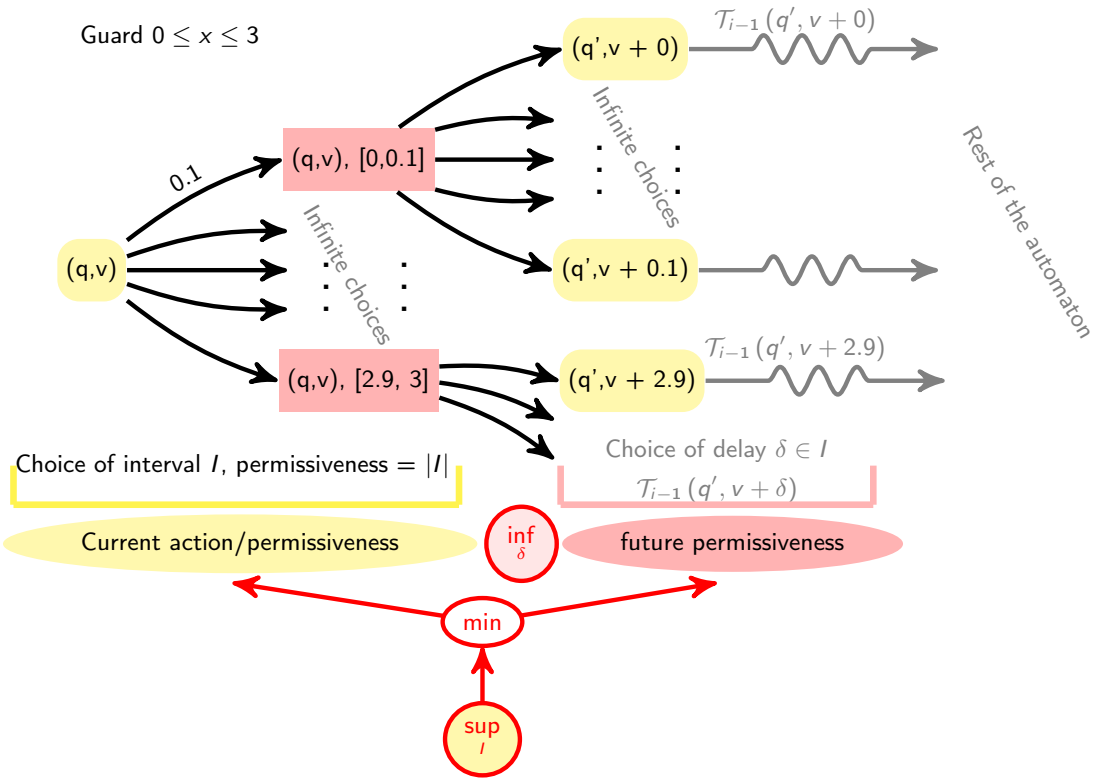
What is the permissiveness?



What is the permissiveness?



What is the permissiveness?



Issues: How to compute the permissiveness ?

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

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

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 answer 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 ?

~~$\delta = b$~~ ?

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

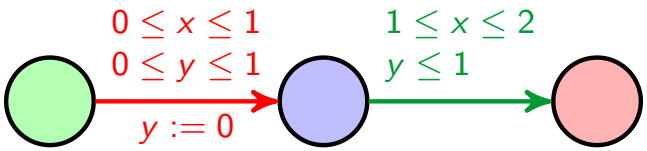
Issues: opponent's strategy

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

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

Issues: opponent's strategy

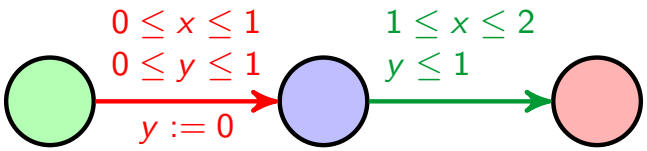
Choice of player	Best respond of opponent ?
$[a, b]$	$\cancel{\delta = b}$?



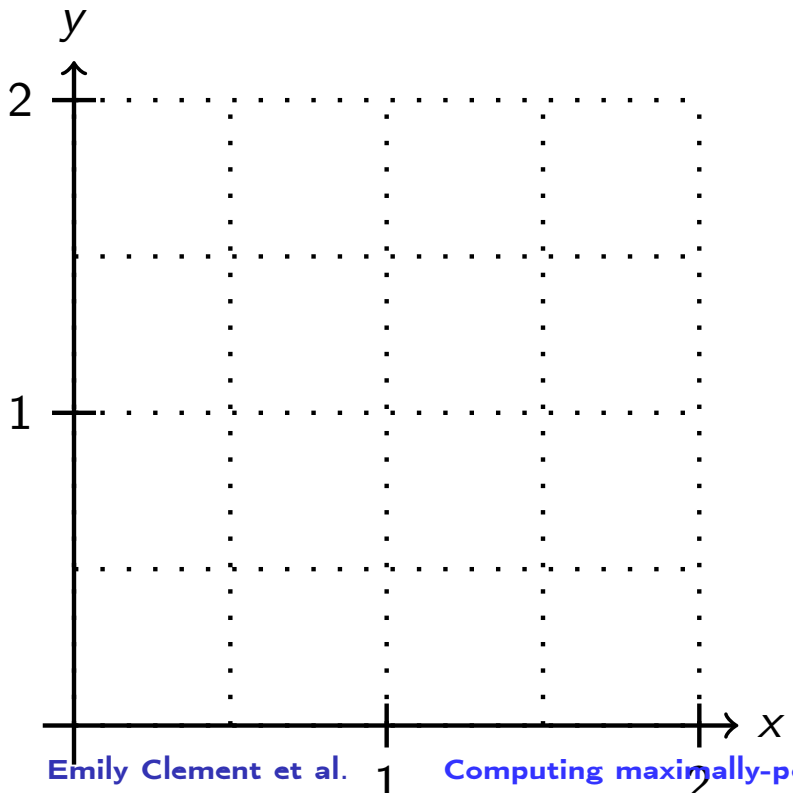
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



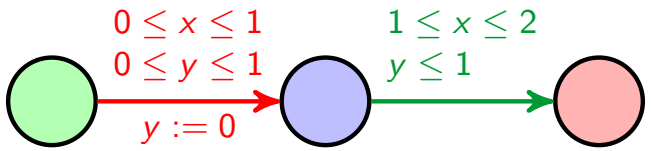
- Graph of the counter example:



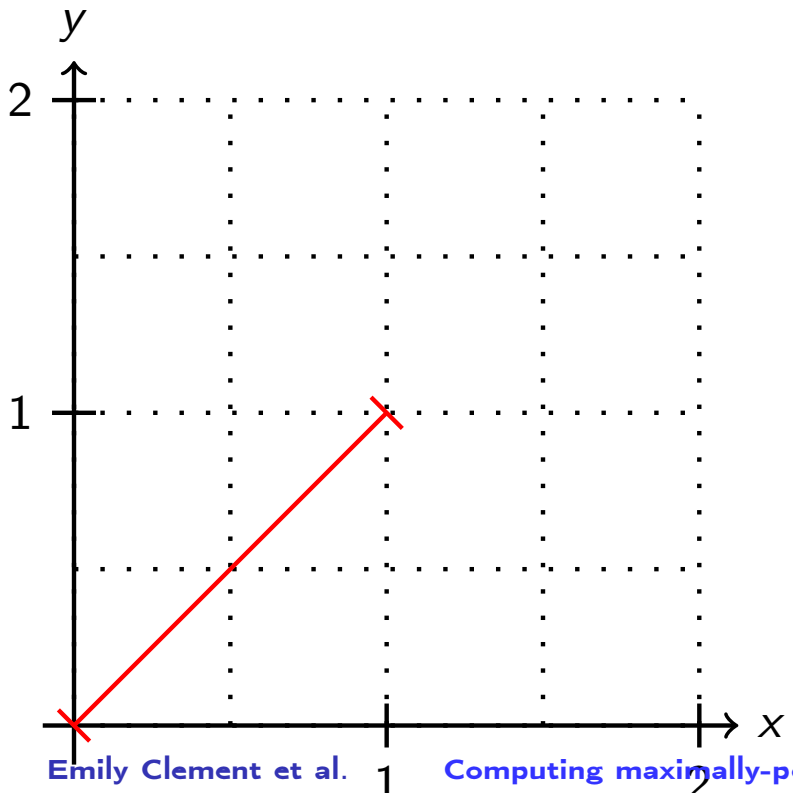
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



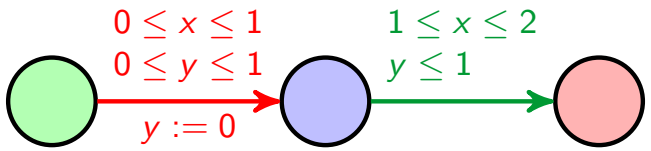
- Graph of the counter example:



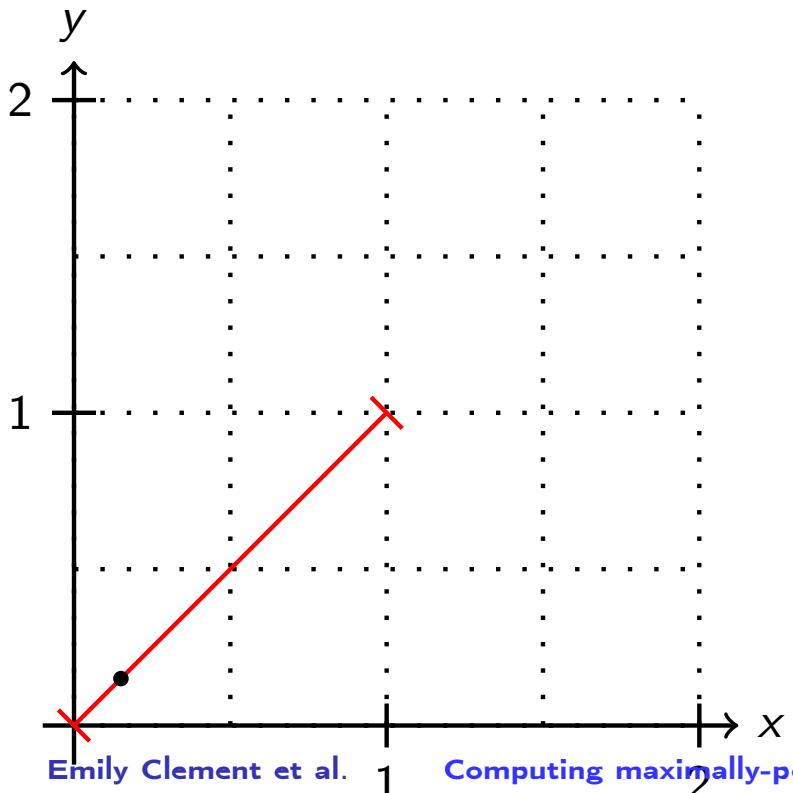
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



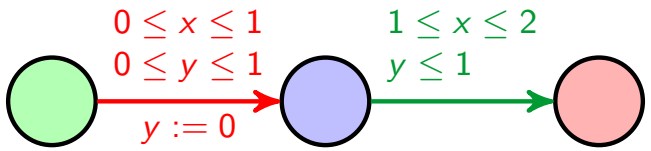
- Graph of the counter example:



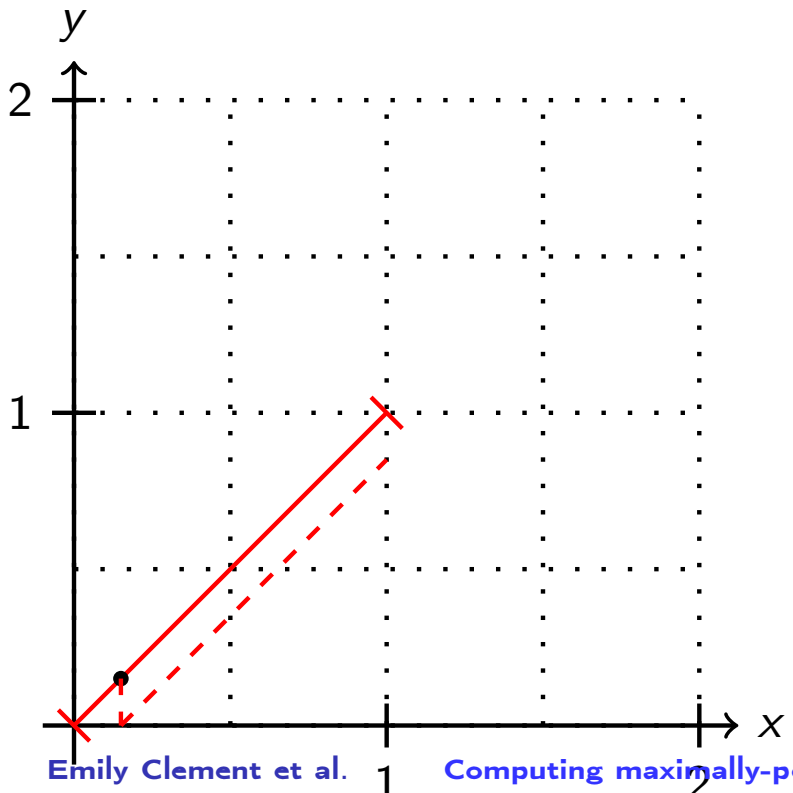
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



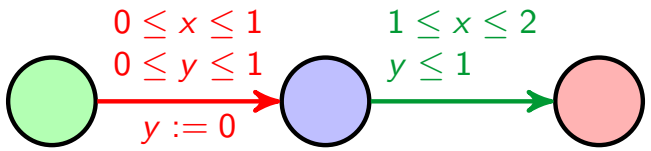
- Graph of the counter example:



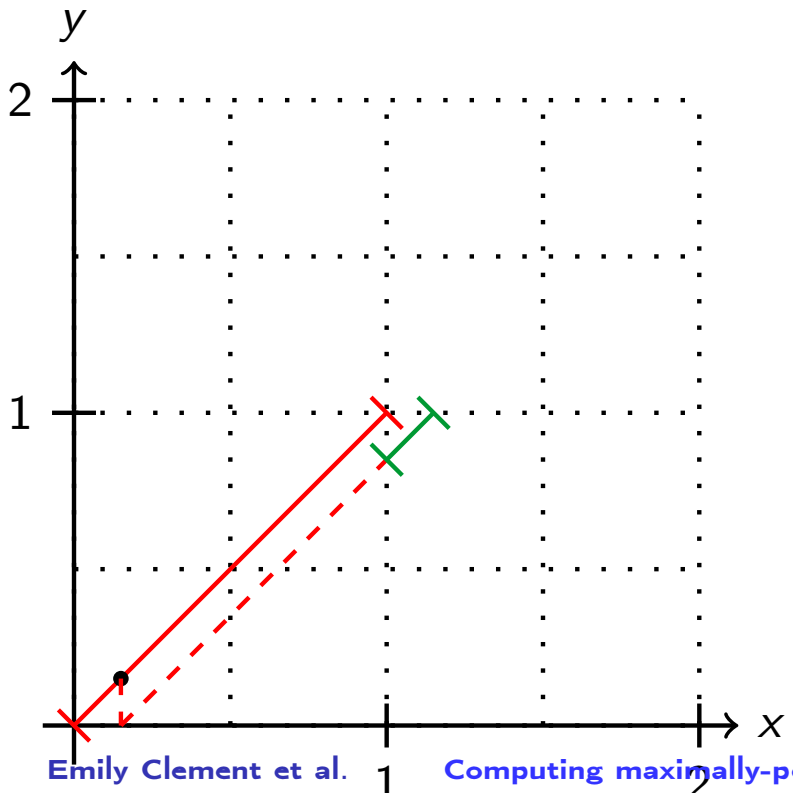
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



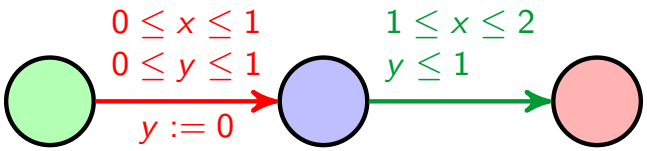
- Graph of the counter example:



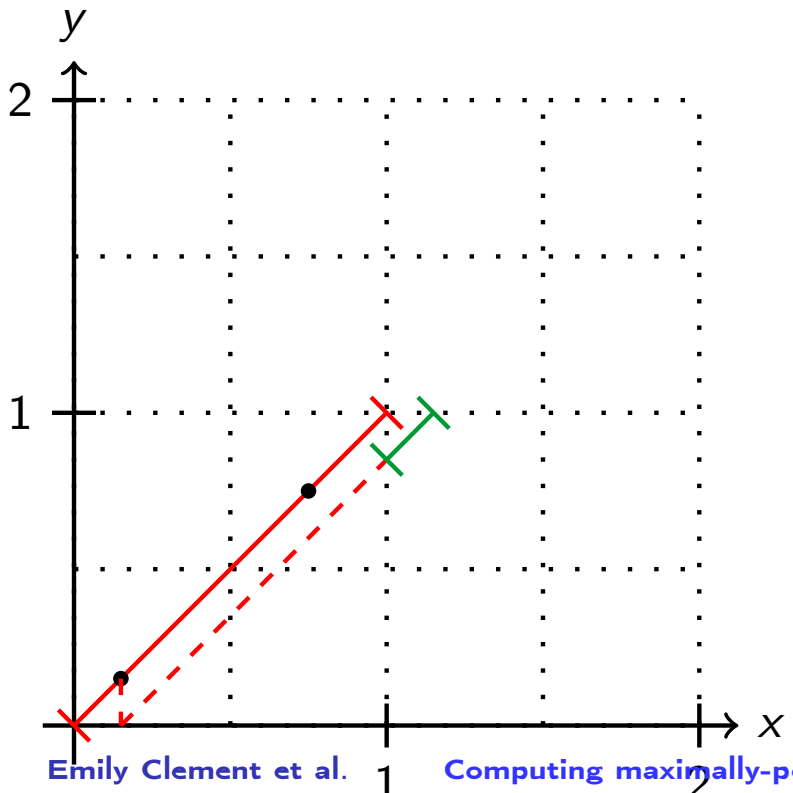
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



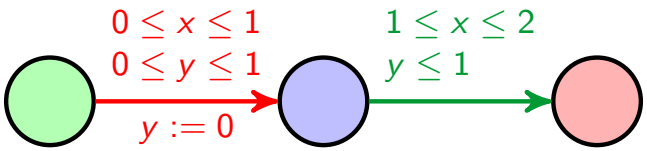
- Graph of the counter example:



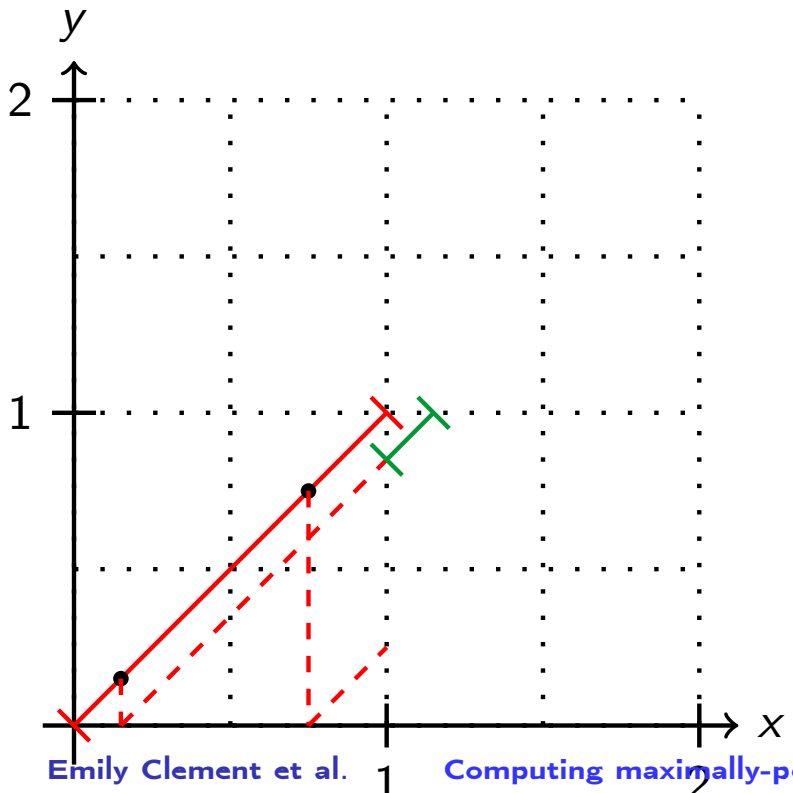
For linear automata: Counter-example and results for our model

Issues: opponent's strategy

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



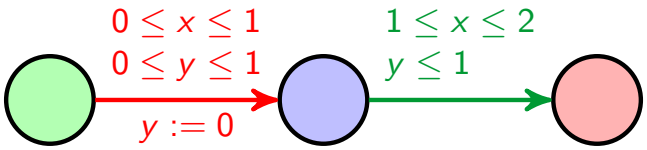
- Graph of the counter example:



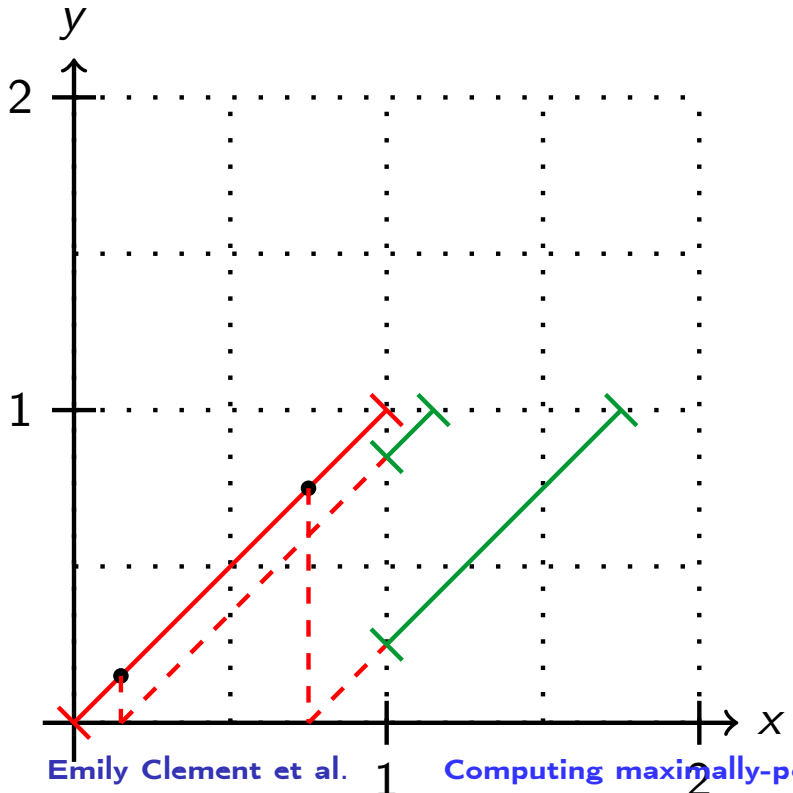
For linear automata: Counter-example and results for our model

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 permissiveness

- Identical guards

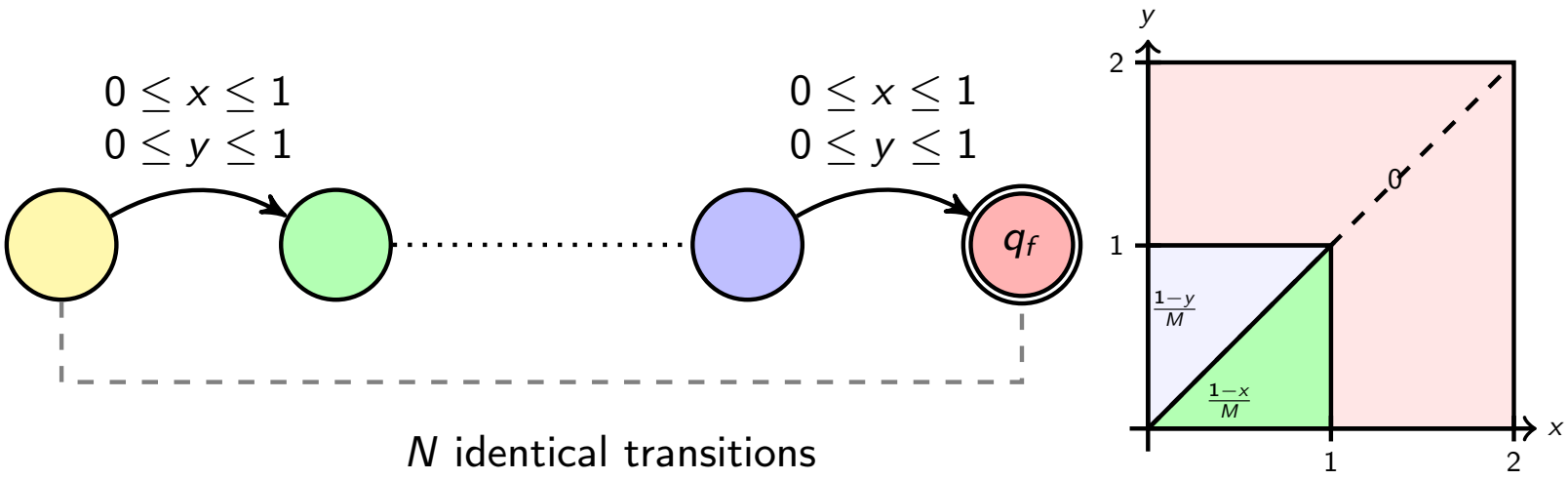


Figure: Permissiveness in q_{N-M}

For linear automata: examples of computation of the permissiveness

- Identical guards

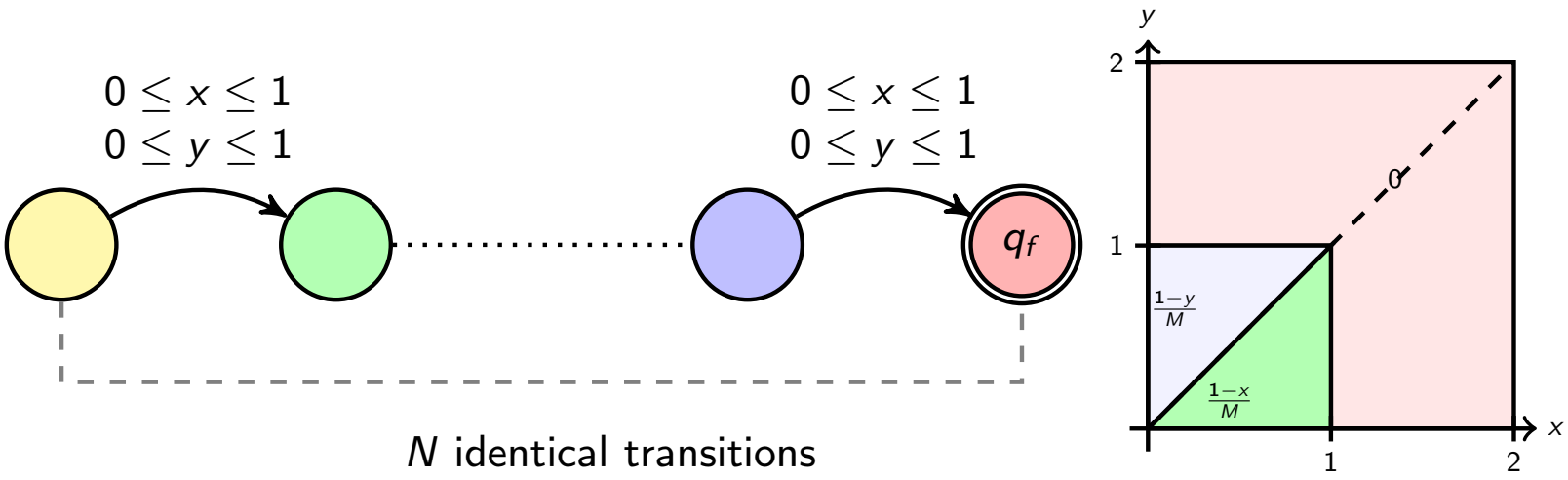


Figure: Permissiveness in q_{N-M}

- The strategy of the player and the opponent

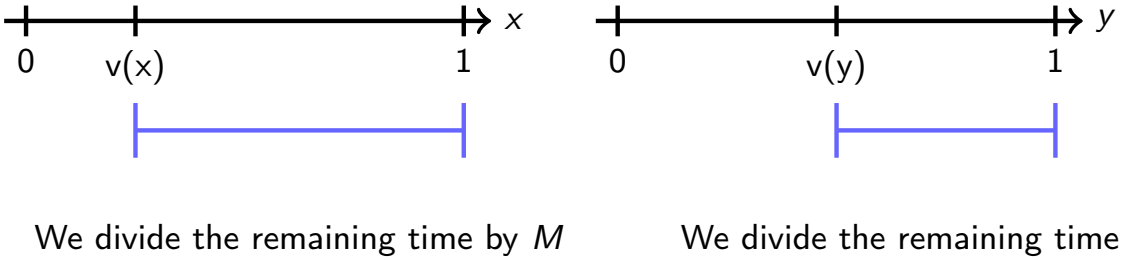
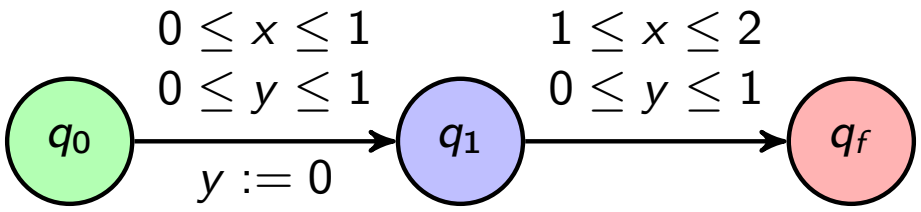
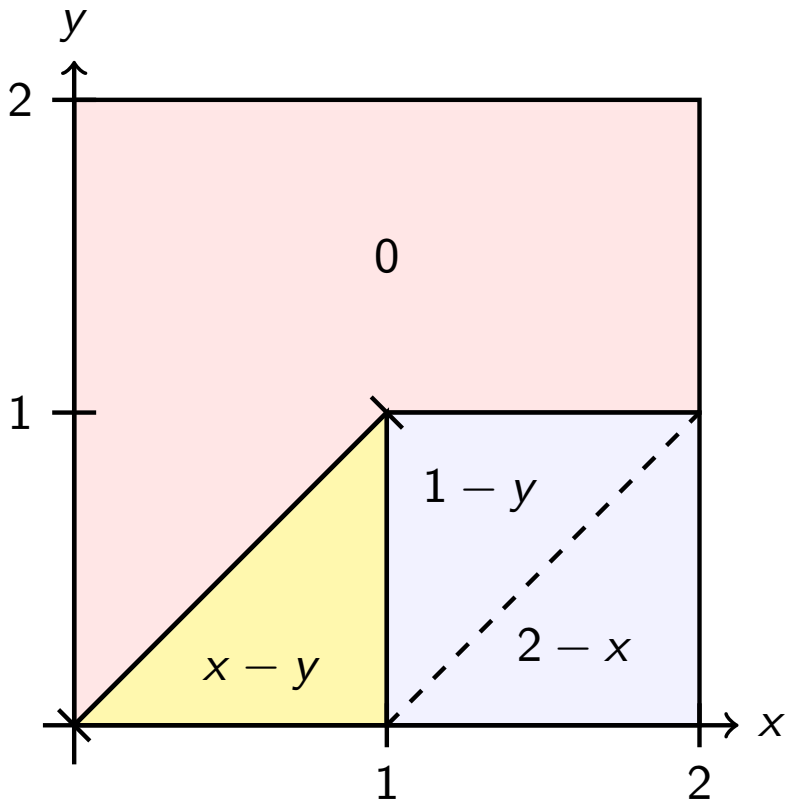


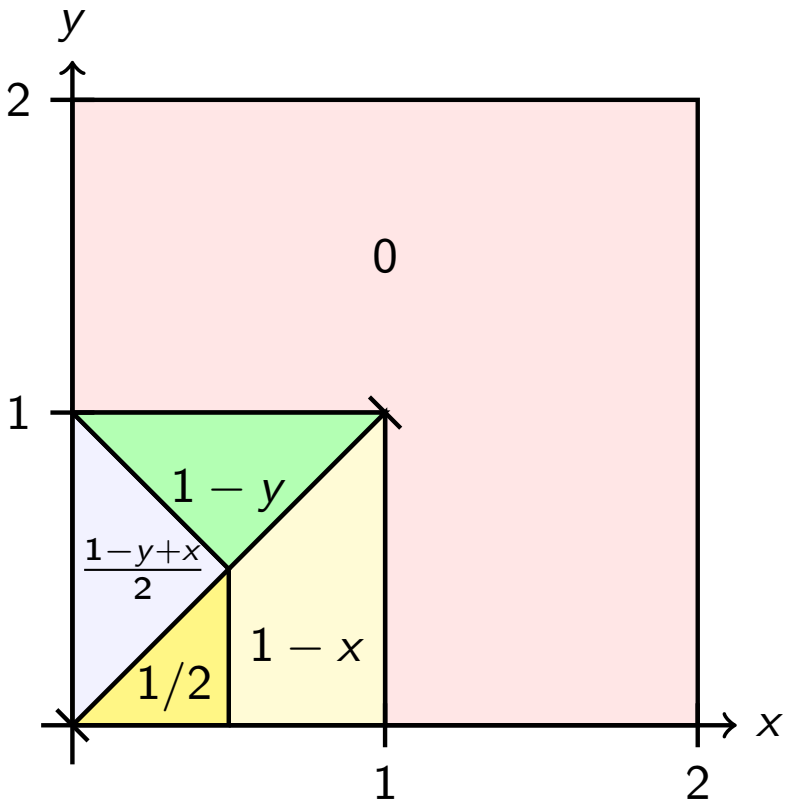
Figure: Time left for each clock

A more complicated example





(a) Permissiveness in q_1



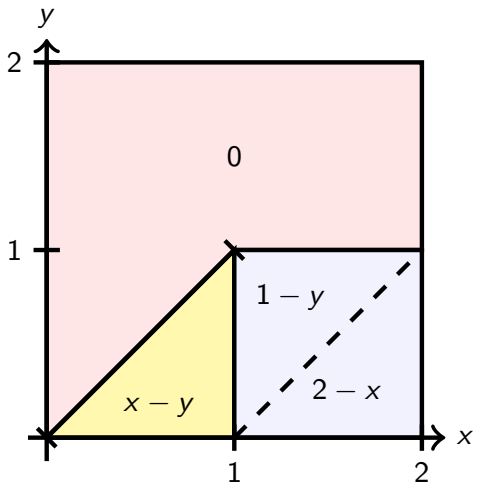
(b) Permissiveness in q_0

For linear automata: strategy of the player

- Recall: Choice of player

Choices of a, b that maximizes $\min\left(b - a, T\left(q, v + a\left[r\right]\right), T\left(q, v + a\left[r\right]\right)\right)$ (under linear contraints)?

- Intuition for the form of $v \mapsto T\left(q, v\right)$



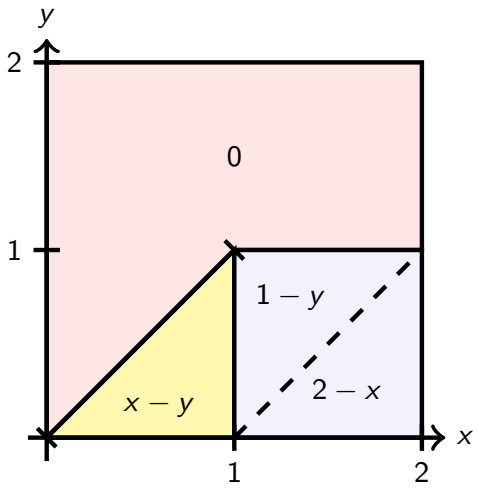
- ▷ **Piece-wise** affine function, with computable pieces and coefficients
- ▷ Computable splitting of **zones**

For linear automata: strategy of the player

- Recall: Choice of player

Choices of a, b that maximizes $\min(b - a, T(q, v + a[r]), T(q, v + a[r]))$ (under linear contraints)?

- Intuition for the form of $v \mapsto T(q, v)$



- ▷ **Piece-wise** affine function, with computable pieces and coefficients
- ▷ Computable splitting of **zones**

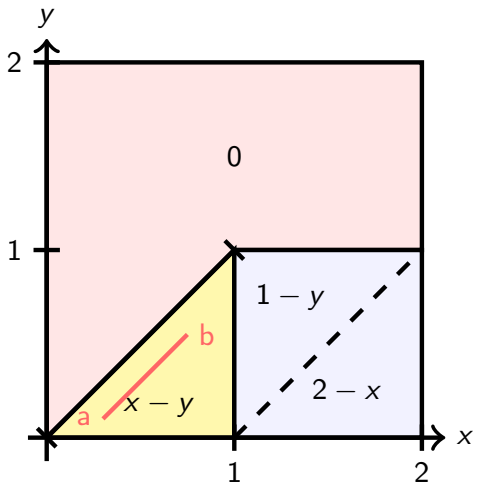
- Our method
  - ▷ Compare all the **choices**, zone by zones.
  - ▷ Maximize a and b of $\min(b - a, f(a), g(b))$, where f and g are **affine functions**.
  - ▷ Conclude by comparing.

For linear automata: strategy of the player

- Recall: Choice of player

Choices of a, b that maximizes $\min(b - a, T(q, v + a[r]), T(q, v + a[r]))$ (under linear contraints)?

- Intuition for the form of $v \mapsto T(q, v)$



- ▷ **Piece-wise** affine function, with computable pieces and coefficients
- ▷ Computable splitting of **zones**

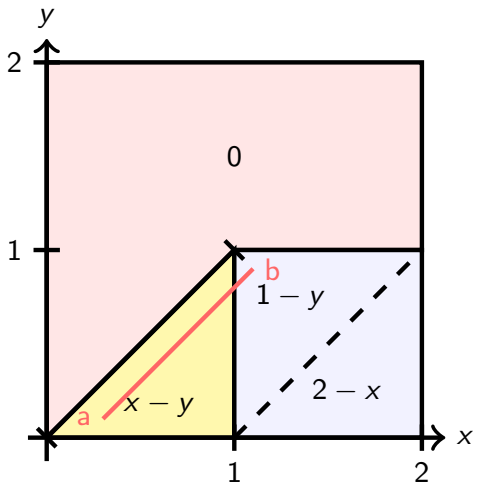
- Our method
  - ▷ Compare all the **choices**, zone by zones.
  - ▷ Maximize a and b of $\min(b - a, f(a), g(b))$, where f and g are **affine functions**.
  - ▷ Conclude by comparing.

For linear automata: strategy of the player

- Recall: Choice of player

Choices of a, b that maximizes $\min(b - a, T(q, v + a[r]), T(q, v + a[r]))$ (under linear contraints)?

- Intuition for the form of $v \mapsto T(q, v)$



- ▷ **Piece-wise** affine function, with computable pieces and coefficients
- ▷ Computable splitting of **zones**

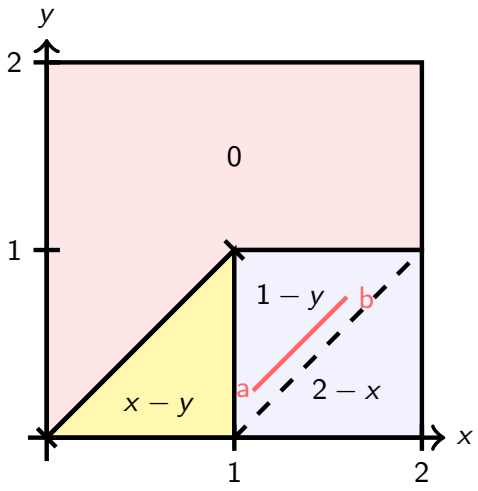
- Our method
  - ▷ Compare all the **choices**, zone by zones.
  - ▷ Maximize a and b of $\min(b - a, f(a), g(b))$, where f and g are **affine functions**.
  - ▷ Conclude by comparing.

For linear automata: strategy of the player

- Recall: Choice of player

Choices of a, b that maximizes $\min(b - a, T(q, v + a[r]), T(q, v + a[r]))$ (under linear contraints)?

- Intuition for the form of $v \mapsto T(q, v)$

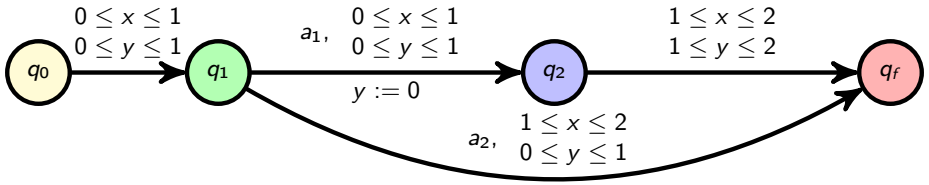


- ▷ **Piece-wise** affine function, with computable pieces and coefficients
- ▷ Computable splitting of **zones**

- Our method
  - ▷ Compare all the **choices**, zone by zones.
  - ▷ Maximize a and b of $\min(b - a, f(a), g(b))$, where f and g are **affine functions**.
  - ▷ Conclude by comparing.

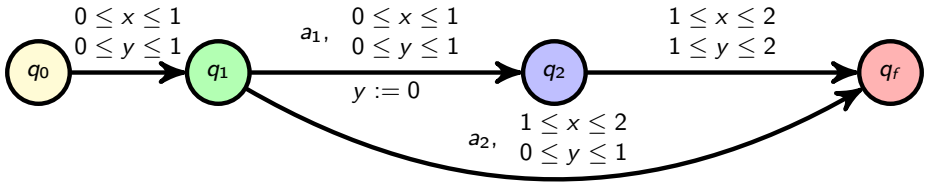
The case of non-linear automata

- A counter-example to our opponent strategy



The case of non-linear automata

- A counter-example to our opponent strategy



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

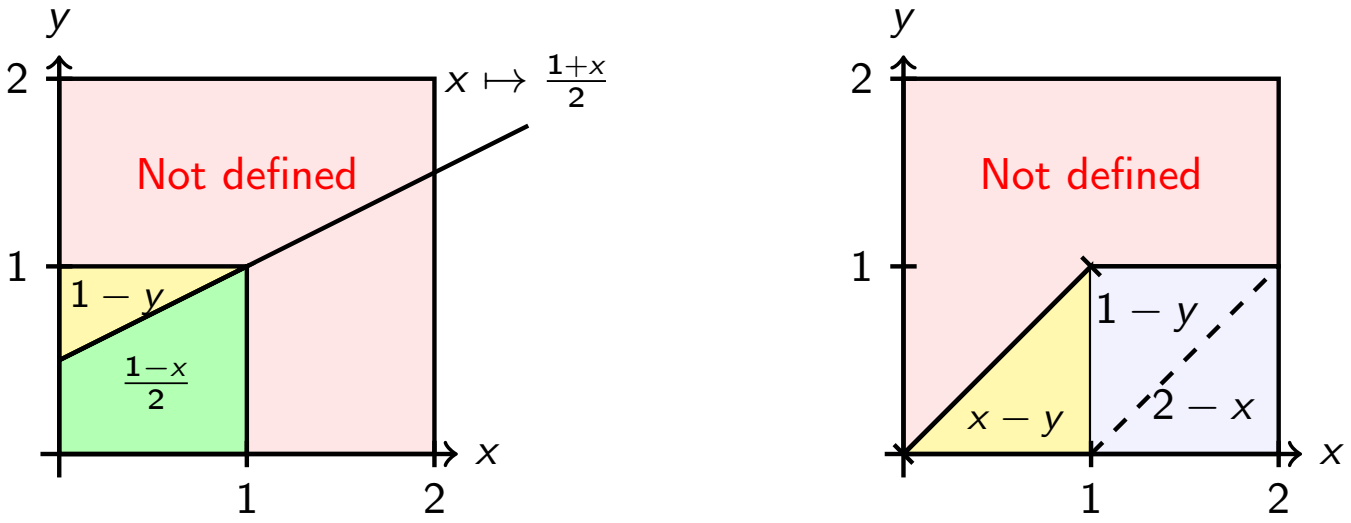
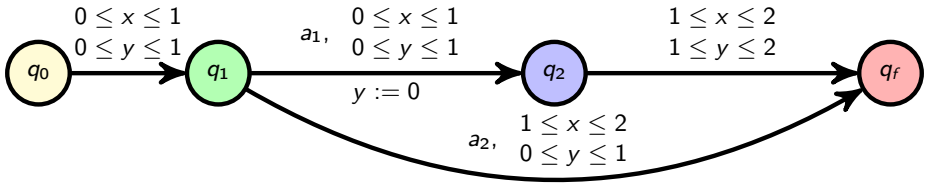


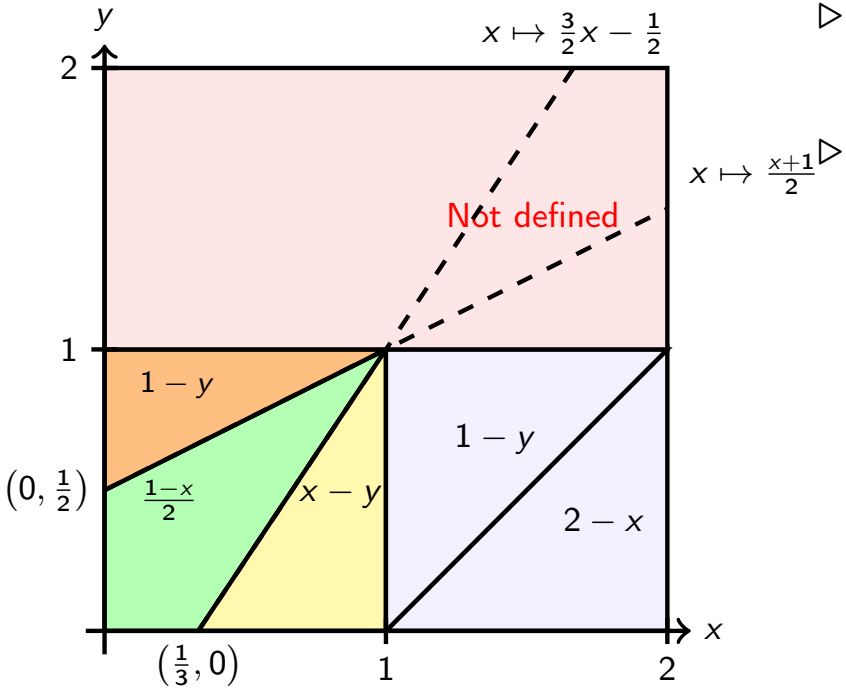
Figure: $v \mapsto T(q_1, v)$ depending on the action chosen (a_1 at left, a_2 at right)

The case of non-linear automata (2)

- A counter-example to our opponent strategy



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



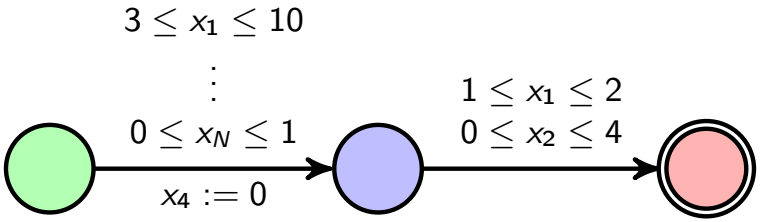
▷ Limits between 2 zones: $\Delta = 1/3 - x$.

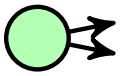
▷ Choosing a in the green zone and b in the yellow zone, the opponent choice of a delay with the following permissiveness function:

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

Achieved, ongoing and future works

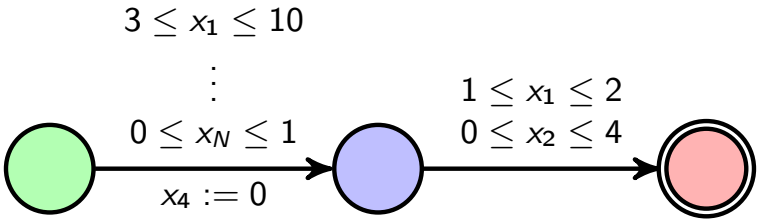
- Achieved and ongoing works



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

Achieved, ongoing and future works

- Achieved and ongoing works



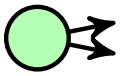
▷ With: Op: max.

▷  : ✓

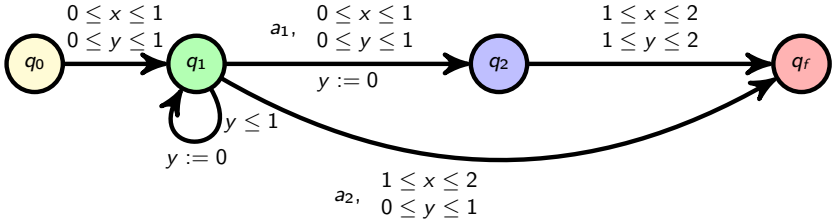
▷  : ✓

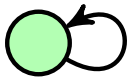
▷ Strategy of  : ✓

▷ Strategy of  : 

▷  : 

- Future work



▷ 

▷ Timed games

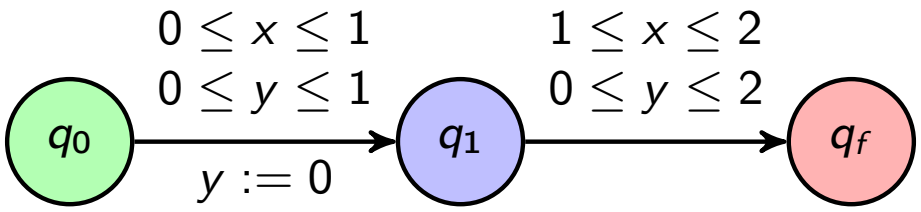
▷ C++ implementation/TChecker

▷ General permissiveness function.

▷  $\longrightarrow$ Stochastic opponent 

▷ Op: $\sum$ or Ponderate $\sum$.

Other complicated examples



- The computation of the permissiveness

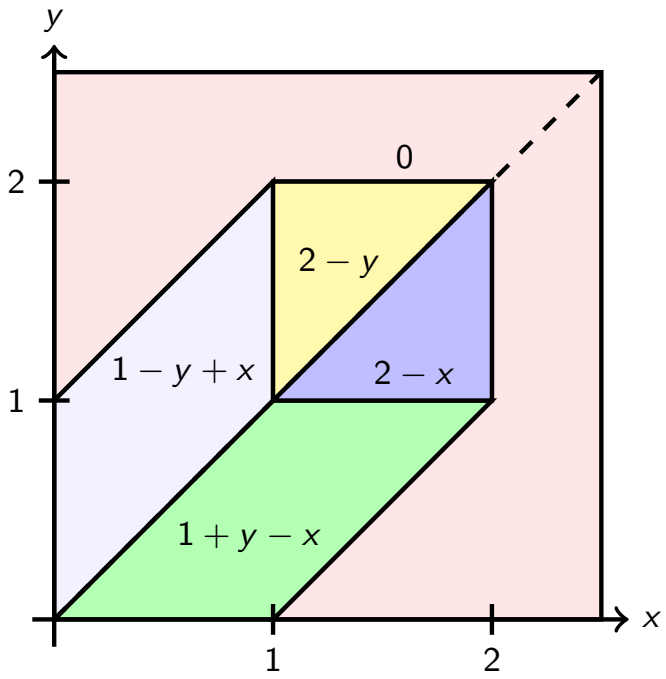


Figure: Permissiveness in q_1

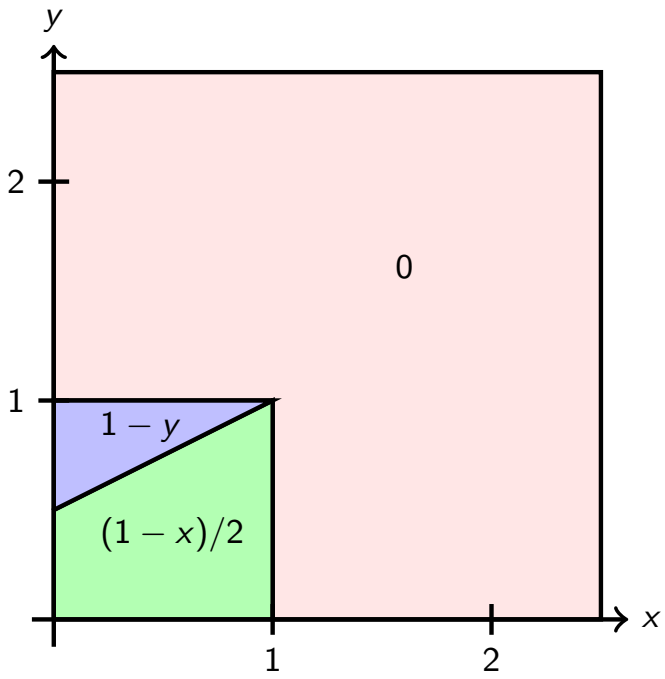


Figure: Permissiveness in q_0

What is the penalty?

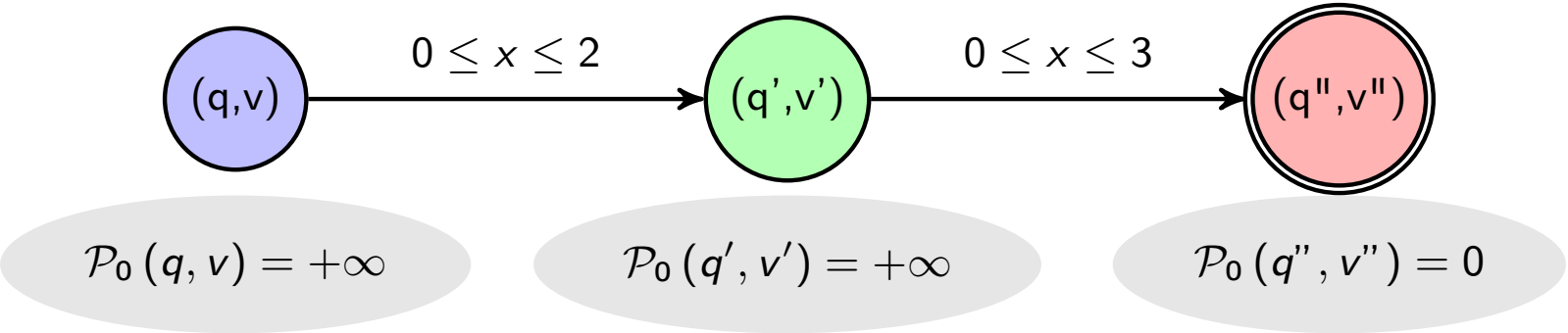
- The penalty for our model

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 $i = 0$, Initialization



What is the penalty?

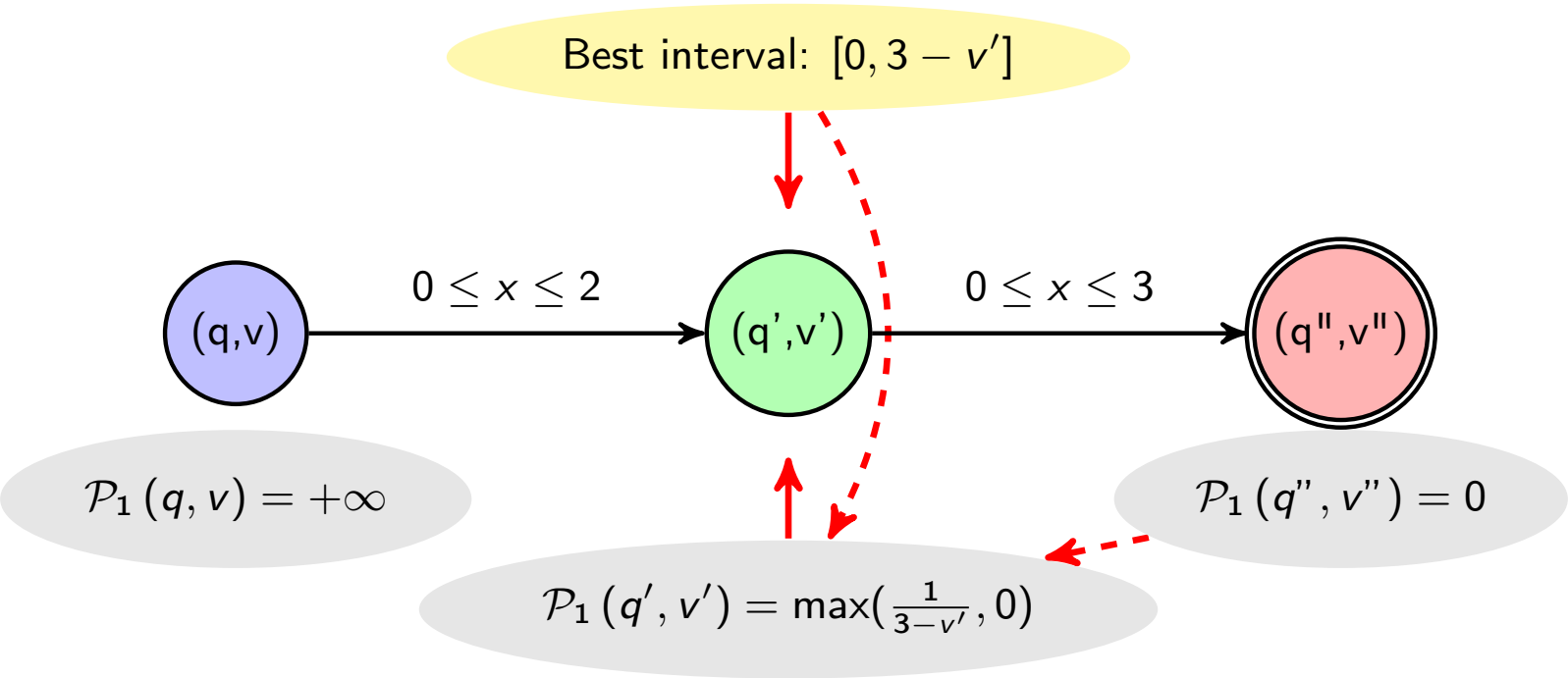
- The penalty for our model

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



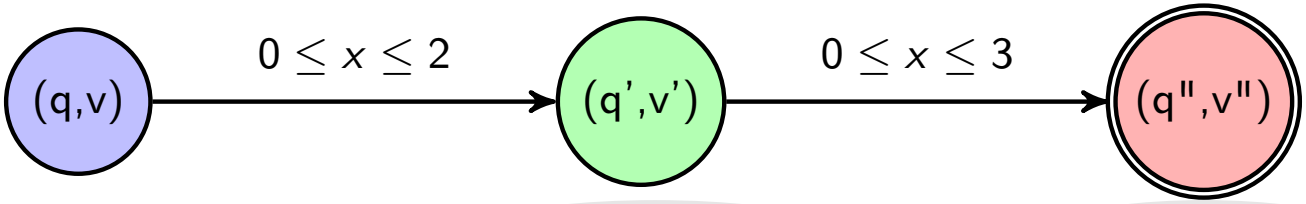
What is the penalty?

- The penalty for our model

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

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

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



$$\mathcal{P}_1(q, v) = \inf_{[a,b]} \max\left(\frac{1}{b-a}, \sup_{\delta \in [a,b]} \mathcal{P}_1(q', v' + \delta)\right)$$

$$\mathcal{P}_2(q', v') = \frac{1}{3 - v'}$$

$$\mathcal{P}_1(q'', v'') = 0$$

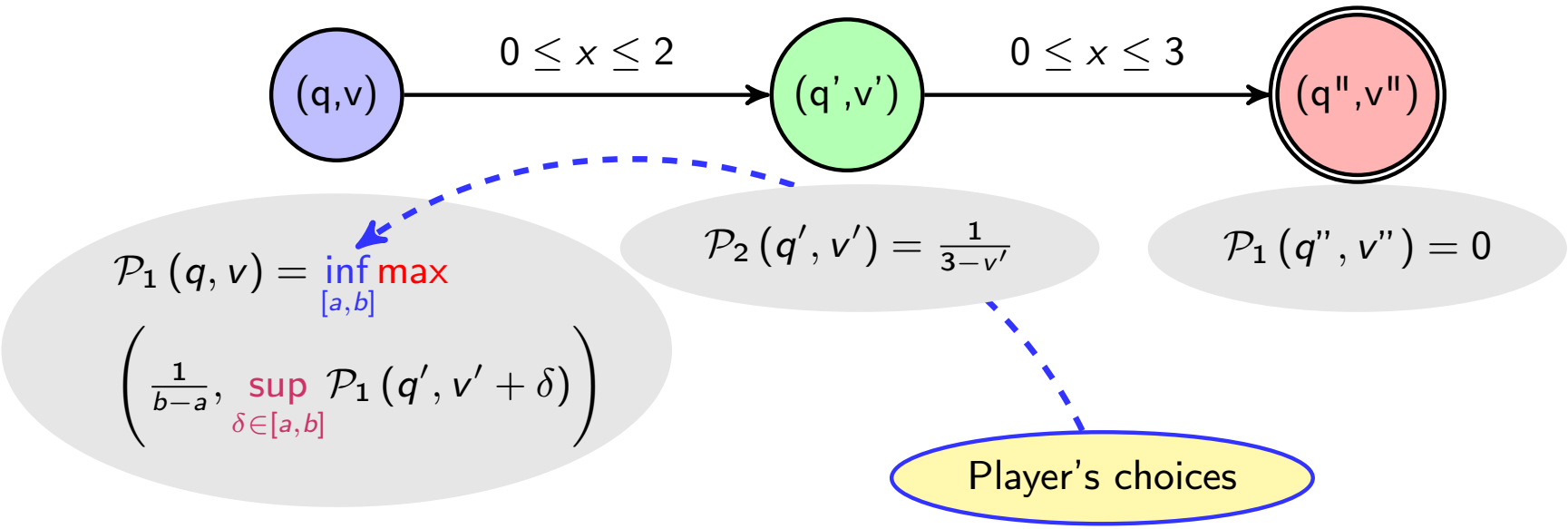
What is the penalty?

- 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)



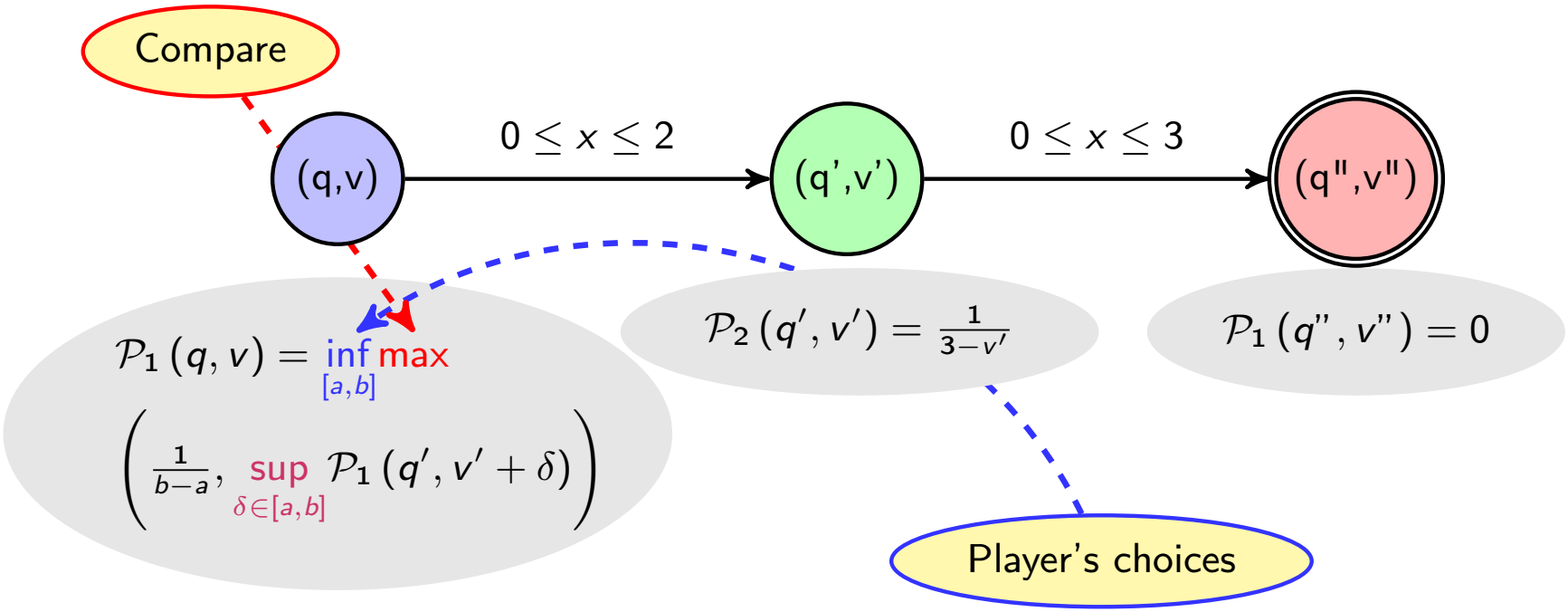
What is the penalty?

- The penalty for our model

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

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

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



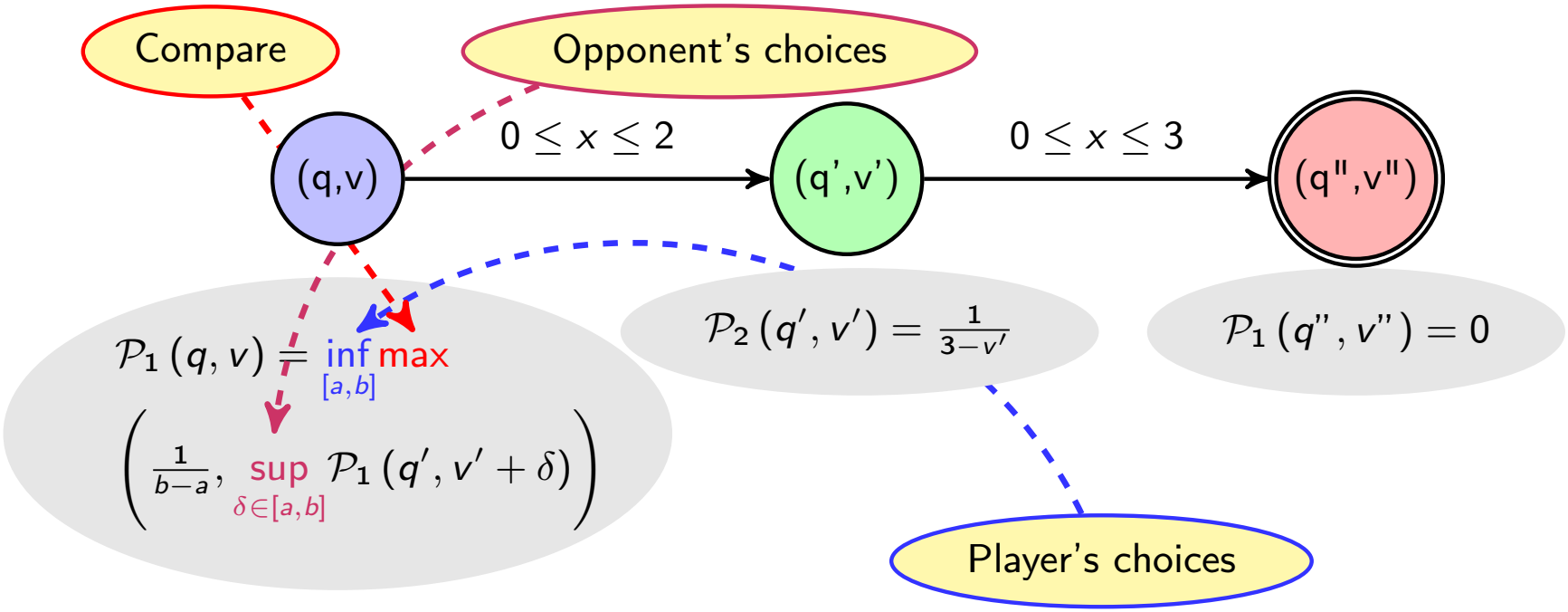
What is the penalty?

- The penalty for our model

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

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

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



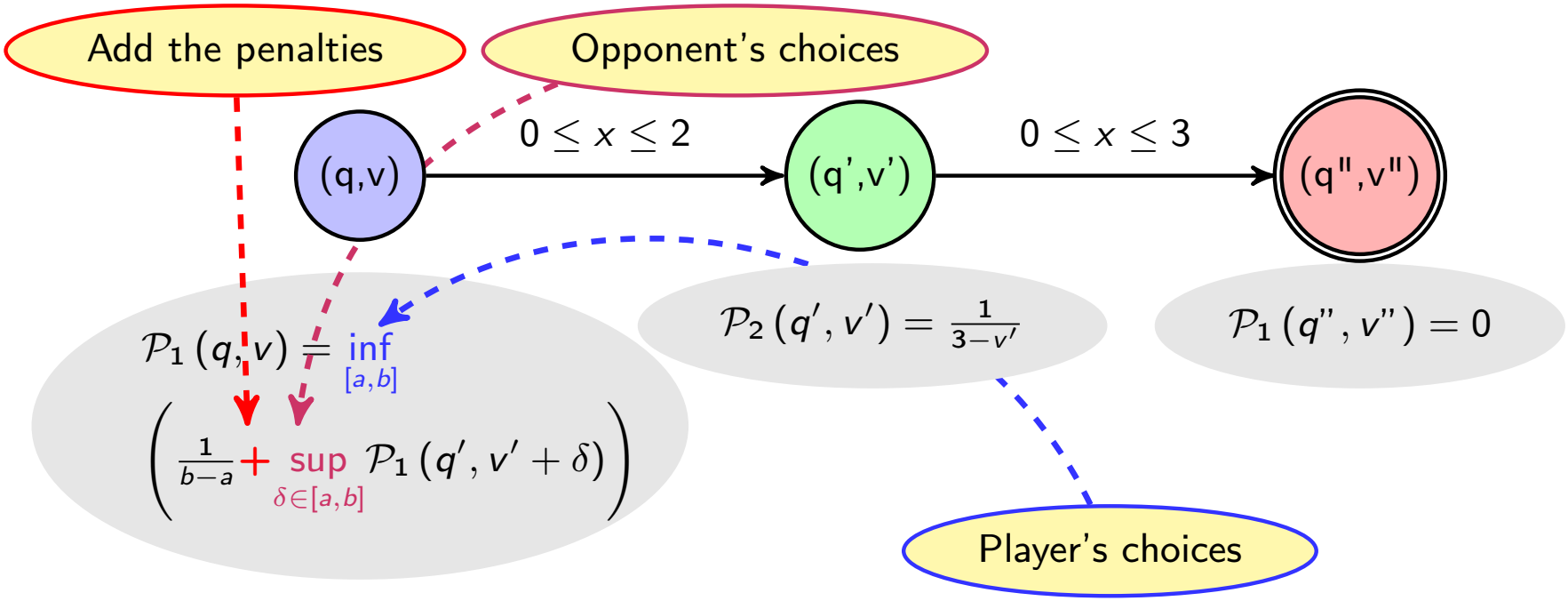
What is the penalty?

- The penalty for our model

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

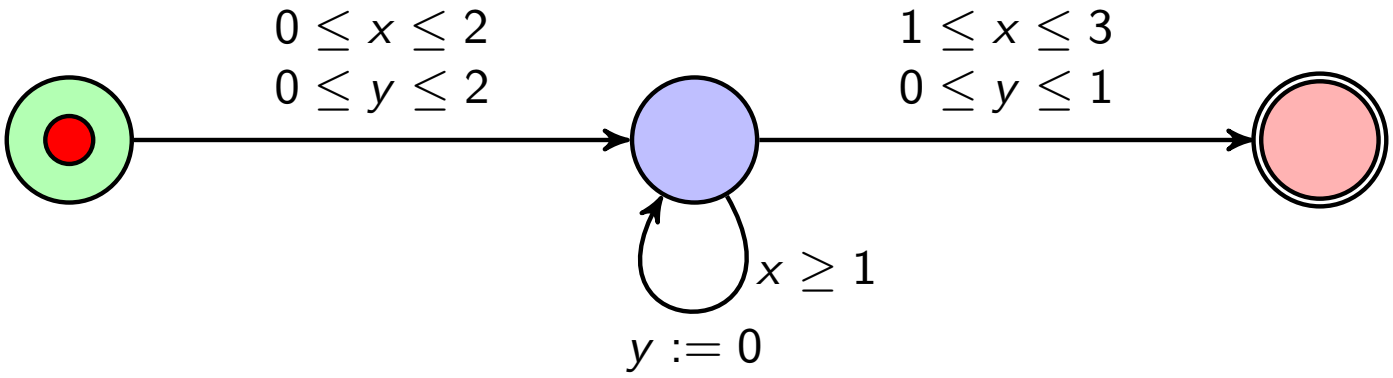
$$\boxed{\mathcal{P}_i(q, v) = \min_{a \in \mathsf{Act}} \inf_{I \in \mathcal{D}(v, \mathsf{inv}(q))} \max\left(\frac{1}{|I|}, \sup_{\delta \in I} \mathcal{P}_{i-1}(succ(v, q, \delta, a))\right)}$$

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



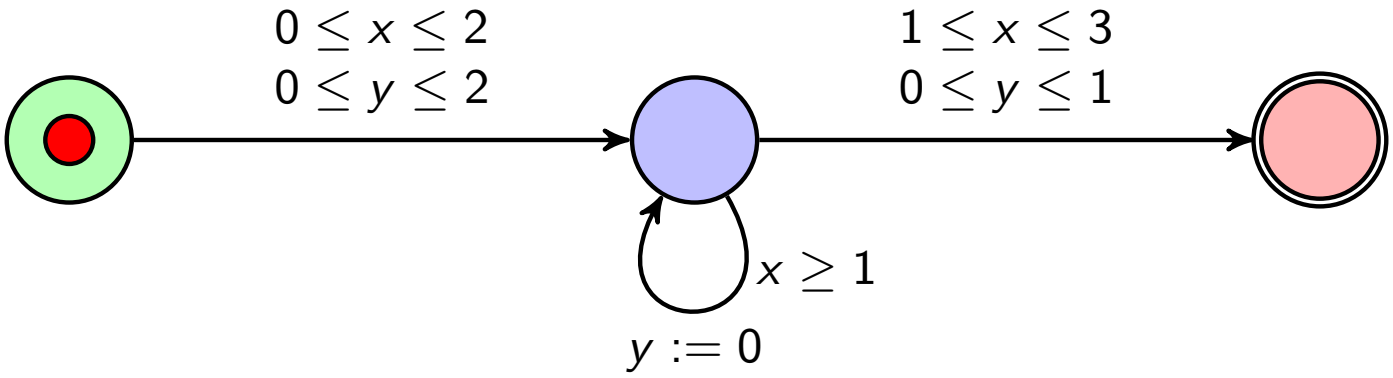
Example of multi-clocks automaton

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



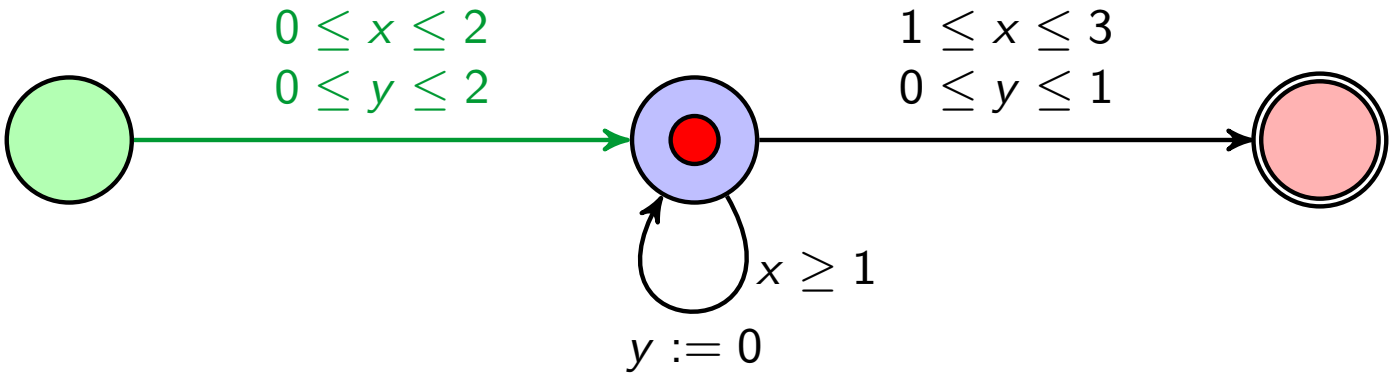
Example of multi-clocks automaton

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



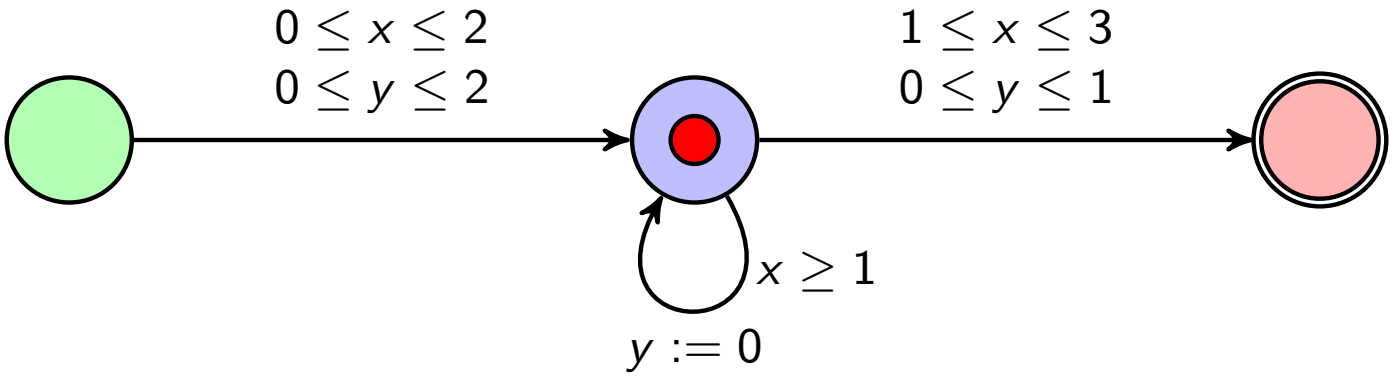
Example of multi-clocks automaton

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



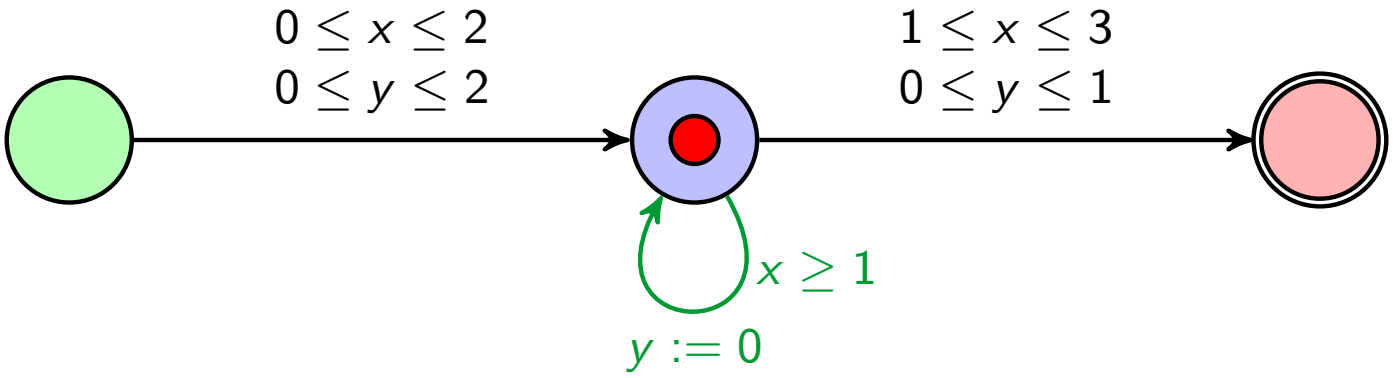
Example of multi-clocks automaton

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



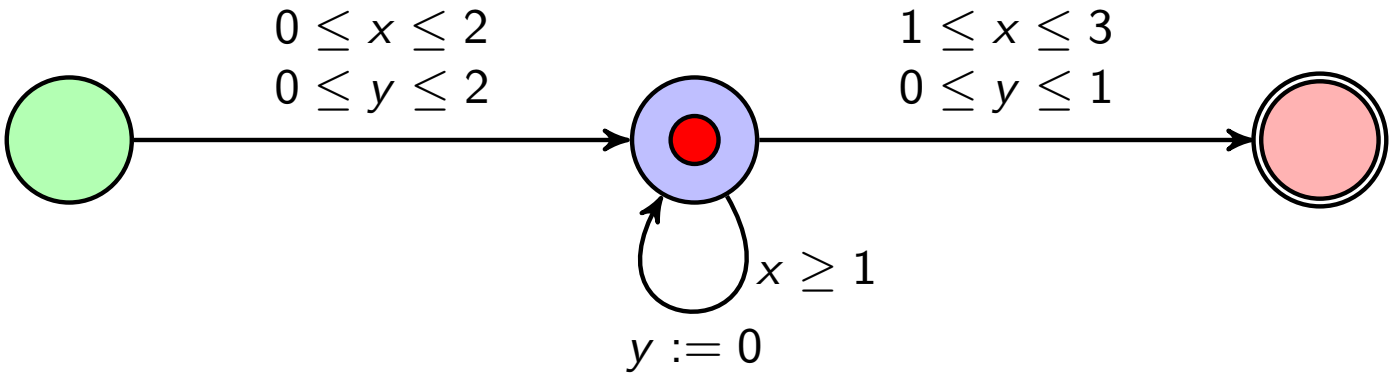
Example of multi-clocks automaton

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



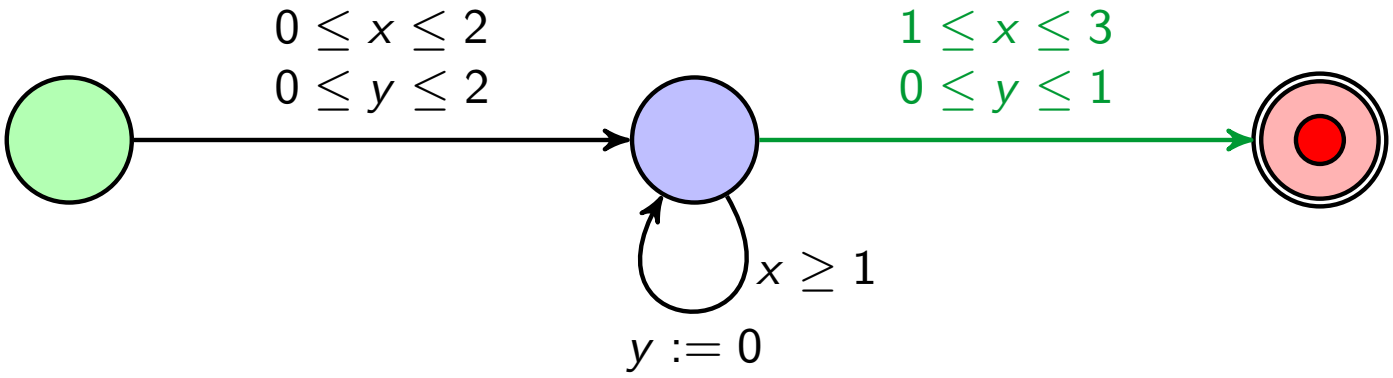
Example of multi-clocks automaton

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



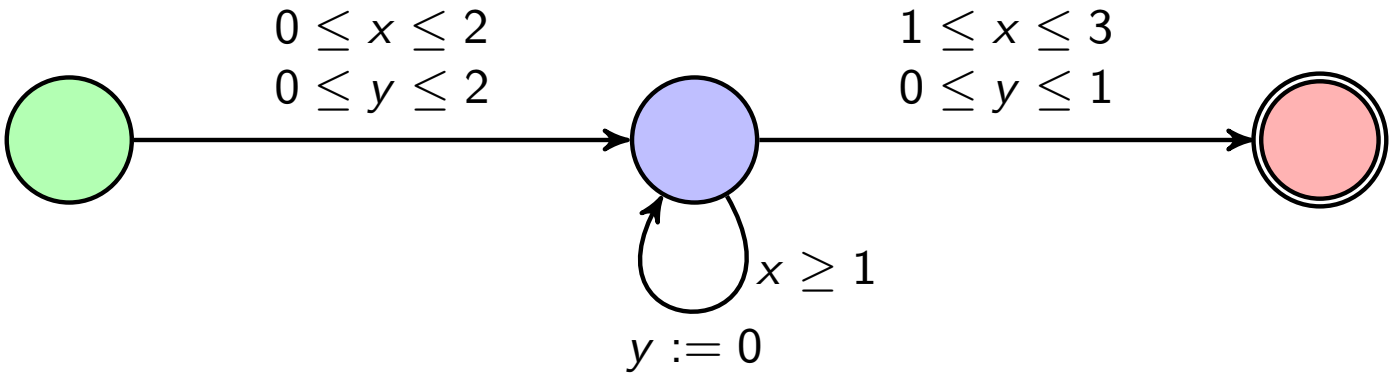
Example of multi-clocks automaton

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



Example of multi-clocks automaton

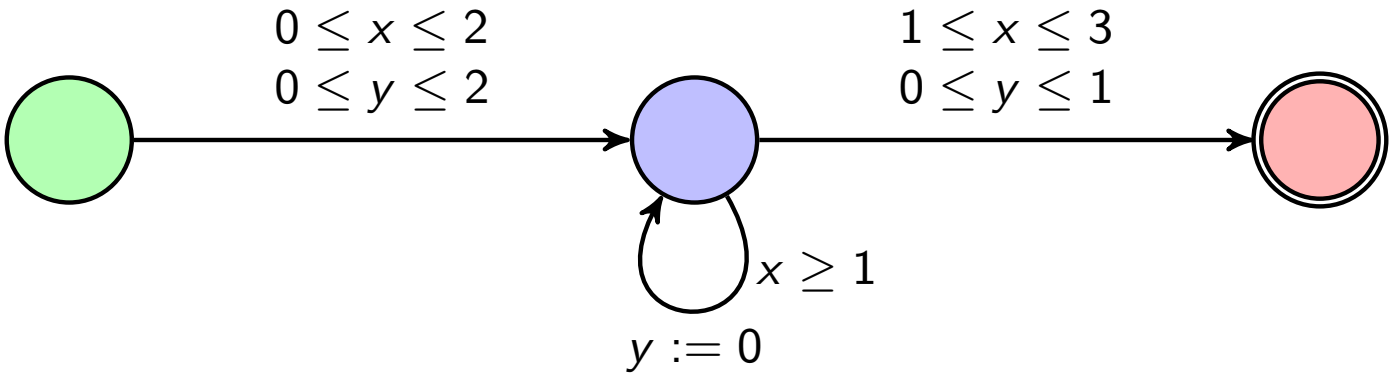
- A two-clock automaton
  - Clock (x, y) values $(3, 0.5)$.



Our goal:

Example of multi-clocks automaton

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

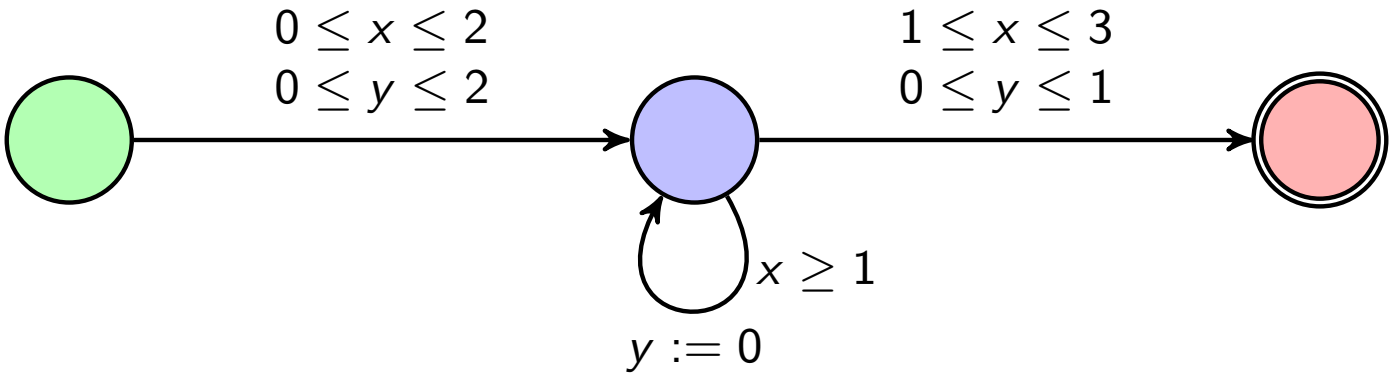


Our goal:

- Reachability ✓

Example of multi-clocks automaton

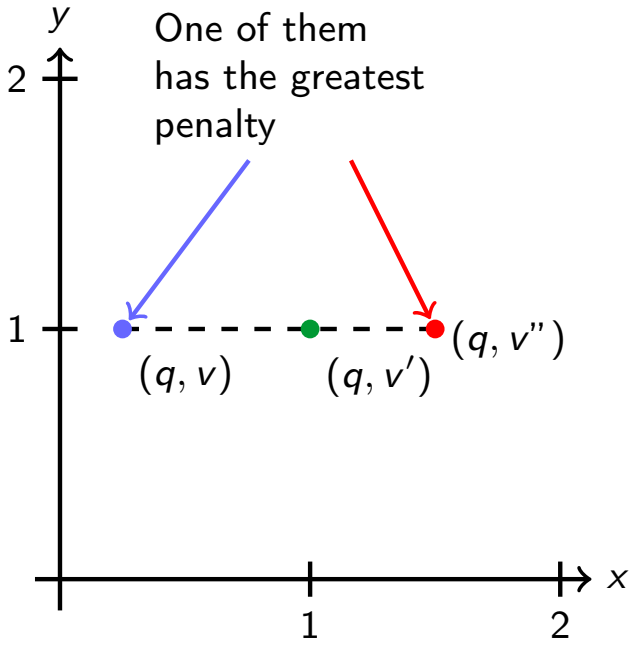
- A two-clock automaton
  - Clock (x, y) values $(3, 0.5)$.



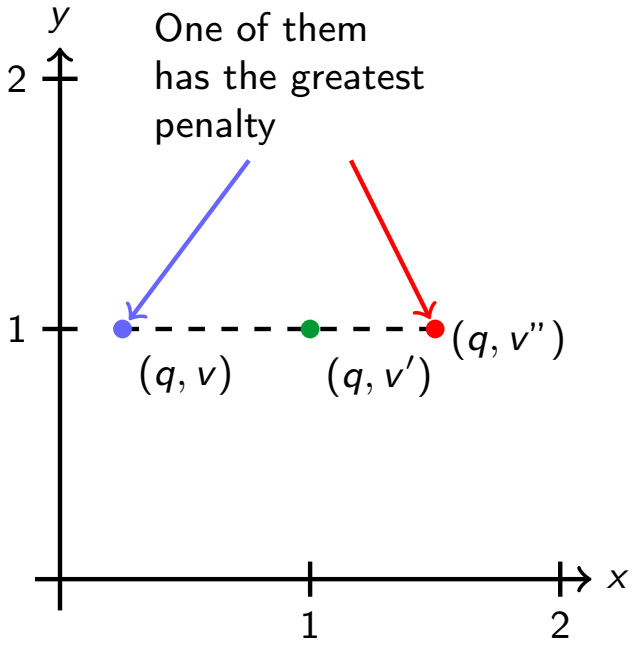
Our goal:

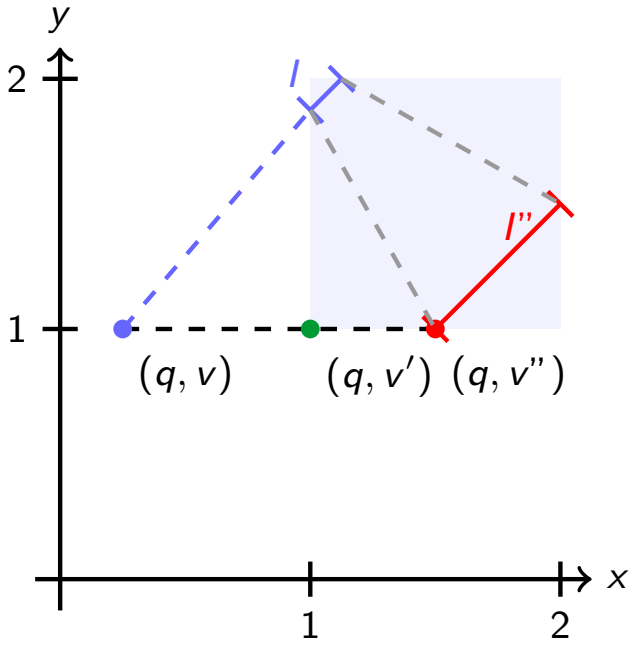
- Reachability ✓
- Robustness ✓

Proof of our strategy



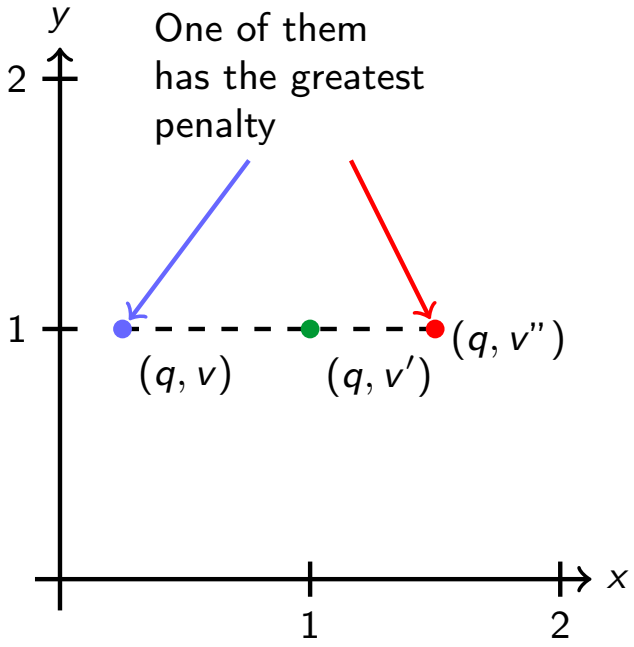
Proof of our strategy

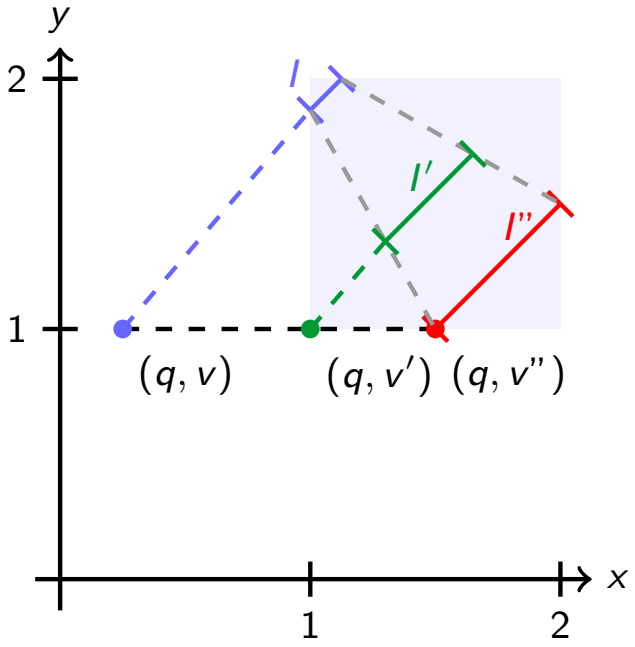




- Step of the proof
  - ▷ Take two arbitrary (enabled) interval I, I''.

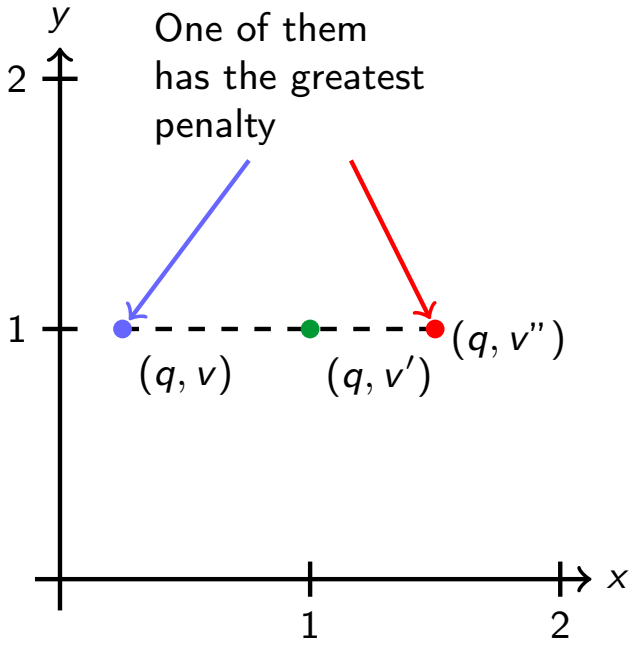
Proof of our strategy

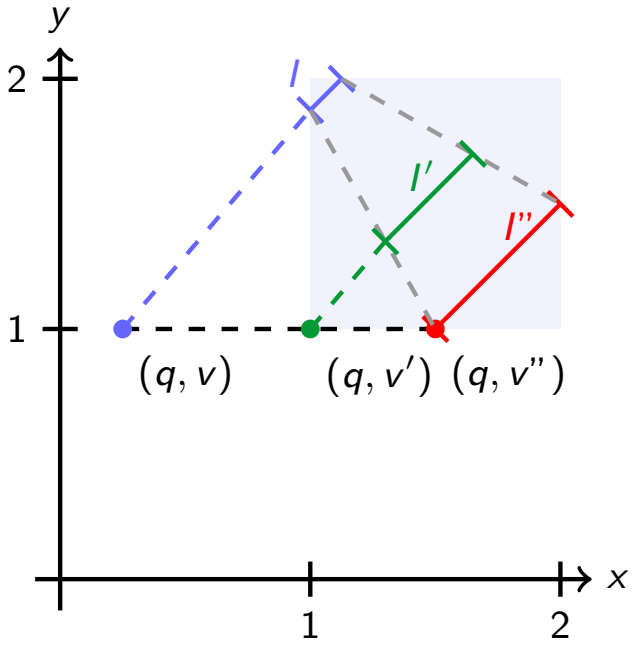




- Step of the proof
  - ▷ Take two arbitrary (enabled) interval I, I''.
  - ▷ Construct I' s.t the inequality works.

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''.