Decidability of Value Problem for 1-clock Weighted Timed Games

Julie Parreaux
Benjamin Monmege Pierre-Alain Reynier

Aix–Marseille Université

CONCUR 2022

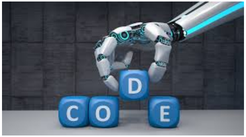
Motivation : game theory for synthesis

Classical approach

Check the correctness of a system

Game theory

Interaction between two antagonistic agents : environment and controller

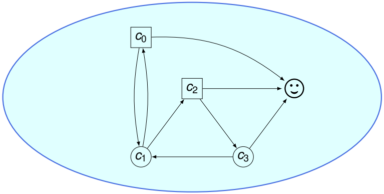


Code synthesis

Correct by construction : synthesis of controller

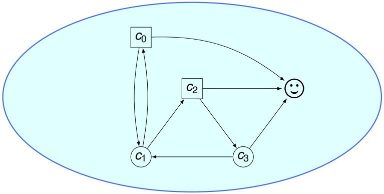
Different classes of games

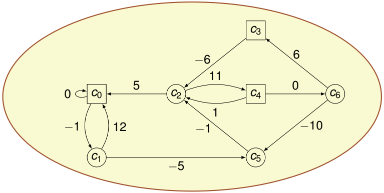
Qualitative games



Different classes of games

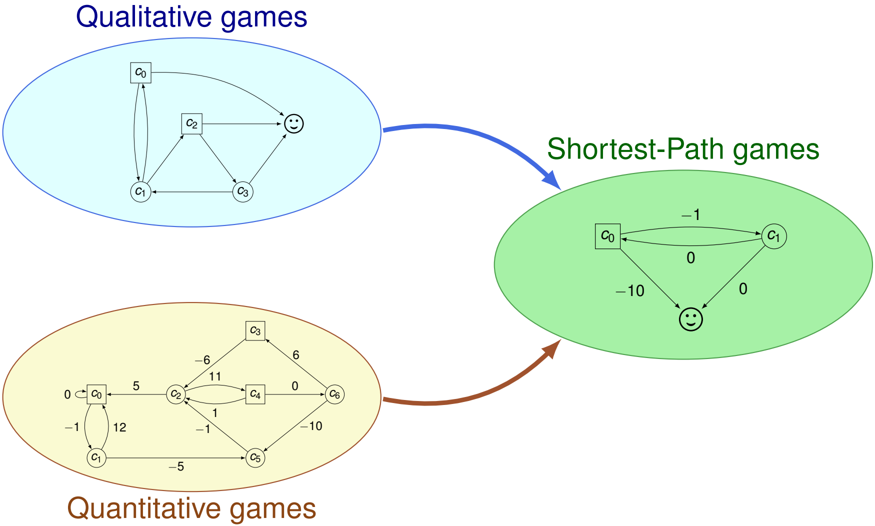
Qualitative games



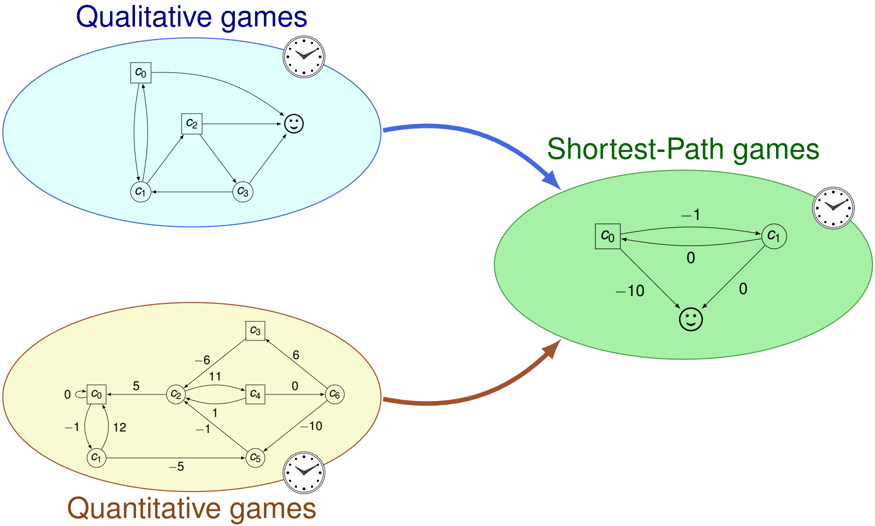


Quantitative games

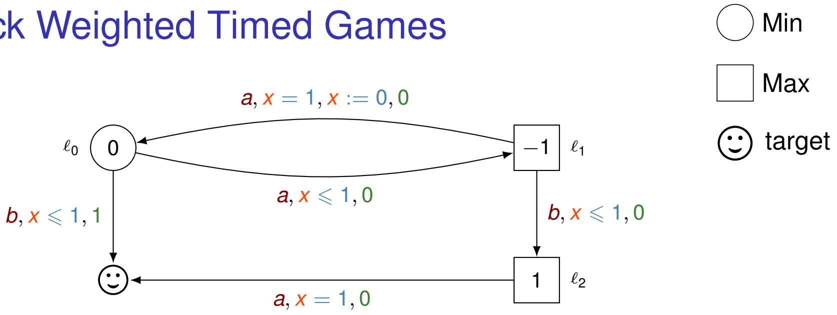
Different classes of games



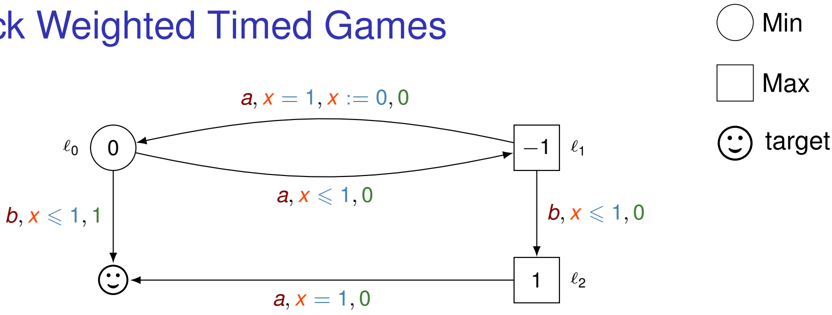
Different classes of games



1-clock Weighted Timed Games



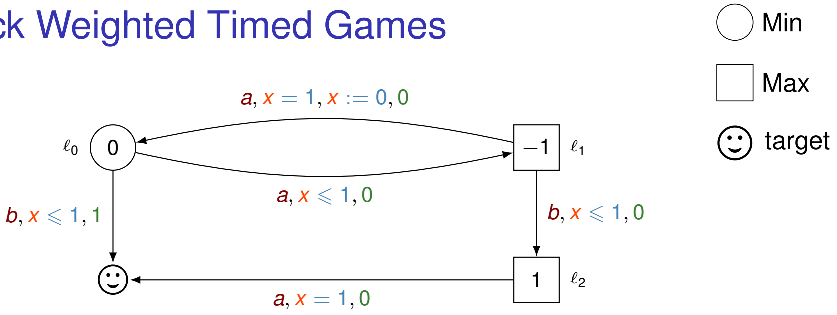
1-clock Weighted Timed Games



Play ρ

$$(\ell_1, 0) \xrightarrow{0.5,\ a} (\ell_0, 0.5) \xrightarrow{0.5,\ a} (\ell_1, 0) \xrightarrow{1/3,\ b} (☺, 1/3)$$

1-clock Weighted Timed Games

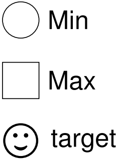


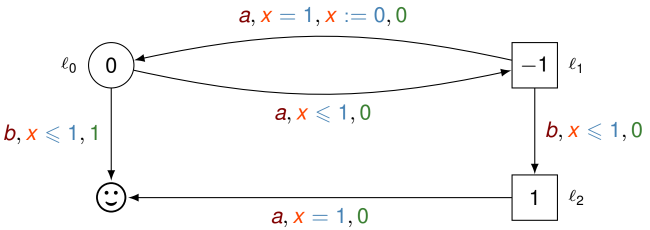
Play ρ

$$(\ell_1, 0) \xrightarrow{0.5,\ a} (\ell_0, 0.5) \xrightarrow{0.5,\ a} (\ell_1, 0) \xrightarrow{1/3,\ b} (☺, 1/3) \rightsquigarrow -0.5$$

$$0 \times 0.5 + 0 \qquad -1 \times 0.5 + 0 \qquad 0 \times \tfrac{1}{3} + 1$$

1-clock Weighted Timed Games





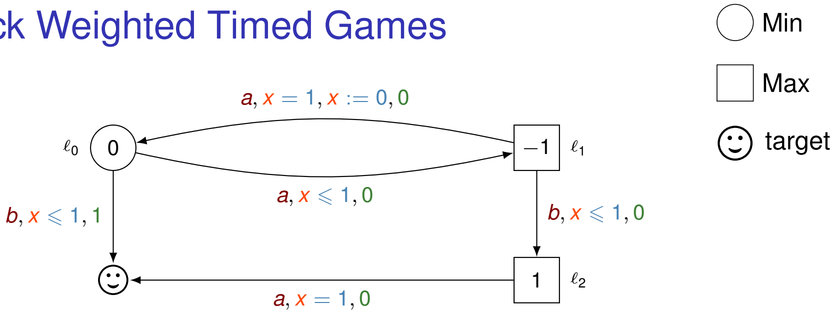
Play ρ

$$(\ell_1, 0) \xrightarrow{0.5,\ a} (\ell_0, 0.5) \xrightarrow{0.5,\ a} (\ell_1, 0) \xrightarrow{1/3,\ b} (☺, 1/3)$$

Strategy

Choose an edge and a delay

1-clock Weighted Timed Games



Play ρ

$$(\ell_1, 0) \xrightarrow{0.5,\ a} (\ell_0, 0.5) \xrightarrow{0.5,\ a} (\ell_1, 0) \xrightarrow{1/3,\ b} (☺, 1/3)$$

Strategy

Choose an edge and a delay

In $(\ell_0, 0)$

Choose a with $t = \frac{1}{3}$

Value problem

σ Min τ Max

Deciding if Val(c) $\leqslant \lambda$?

Value problem

σ Min τ Max

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$

Value problem

σ Min τ Max

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

σ Min τ Max

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game

Value Iteration

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

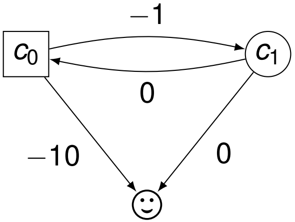
Value problem

σ Min τ Max

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$



State of the art

☺ Decidable for finite game

Value Iteration

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

σ Min τ Max

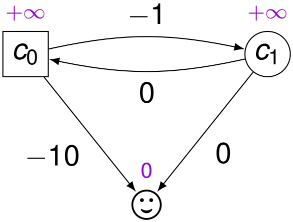
Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

Value

$$\mathrm{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathrm{Payoff}(\mathrm{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

σ Min τ Max

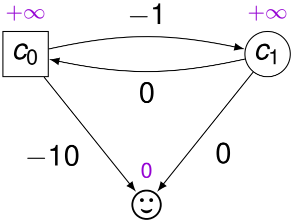
Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

Value

$$\mathrm{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathrm{Payoff}(\mathrm{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

$$\mathrm{Val}(c) = \begin{cases} \min_{c'} \left(\mathrm{wt}(c, c') + \mathrm{Val}(c')\right) & \text{for Min} \\ \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

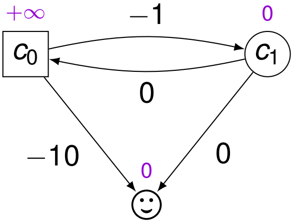
σ Min τ Max

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

$$\mathsf{Val}(c) = \begin{cases} \min_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Min} \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

σ Min τ Max

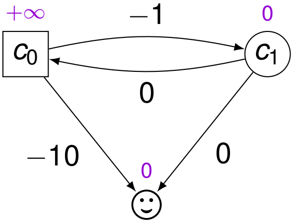
Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

$$\mathsf{Val}(c) = \begin{cases} \min_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Max} \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

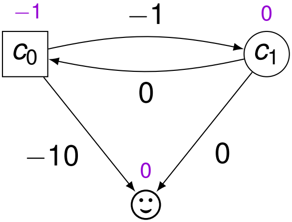
σ Min τ Max

Value

$$\mathrm{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathrm{Payoff}(\mathrm{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

$$\mathrm{Val}(c) = \begin{cases} \min_{c'} (\mathrm{wt}(c,c') + \mathrm{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathrm{wt}(c,c') + \mathrm{Val}(c')) & \text{for Max} \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

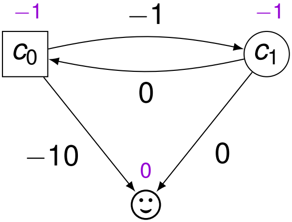
σ Min τ Max

Value

$$\mathrm{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathrm{Payoff}(\mathrm{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

$$\mathrm{Val}(c) = \begin{cases} \min_{c'} (\mathrm{wt}(c, c') + \mathrm{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathrm{wt}(c, c') + \mathrm{Val}(c')) & \text{for Max} \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

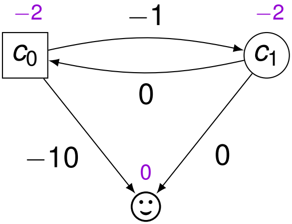
Value problem

σ Min τ Max

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$



State of the art

☺ Decidable for finite game

Value Iteration

$$\mathsf{Val}(c) = \begin{cases} \min_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Max} \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

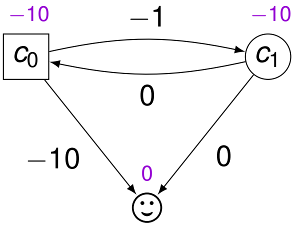
σ Min τ Max

Value

$$\mathrm{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathrm{Payoff}(\mathrm{Play}(c, \sigma, \tau))$$

State of the art

☺ Decidable for finite game



Value Iteration

$$\mathrm{Val}(c) = \begin{cases} \min_{c'} (\mathrm{wt}(c, c') + \mathrm{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathrm{wt}(c, c') + \mathrm{Val}(c')) & \text{for Max} \end{cases}$$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games., T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege, 2017, Acta Informatica

Value problem

σ Min τ Max

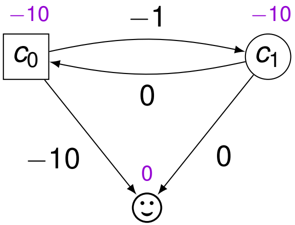
Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

Value

$$\mathrm{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathrm{Payoff}(\mathrm{Play}(c, \sigma, \tau))$$

State of the art

- ☺ Decidable for finite game
- ☹ Undecidable for at least 2 clocks



Value Iteration

$$\mathrm{Val}(c) = \begin{cases} \min_{c'} (\mathrm{wt}(c, c') + \mathrm{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathrm{wt}(c, c') + \mathrm{Val}(c')) & \text{for Max} \end{cases}$$

On Optimal Timed Strategies, T. Brihaye, V. Bruyère and J.-F. Raskin, 2005, FORMATS

On the Value Problem in Weighted Timed Games, P. Bouyer, S. Jaziri, and N. Markey, 2015, CONCUR.

Value problem

σ Min τ Max

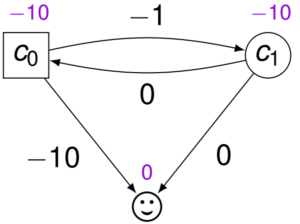
Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

Value

$$\mathsf{Val}(c) = \inf_{\sigma} \sup_{\tau} \mathsf{Payoff}(\mathsf{Play}(c, \sigma, \tau))$$

State of the art

- ☺ Decidable for finite game
- ☹ Undecidable for at least 2 clocks



Value Iteration

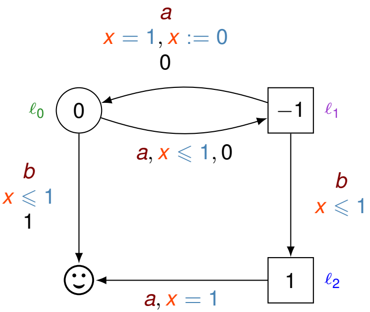
$$\mathsf{Val}(c) = \begin{cases} \min_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Min} \\ \max_{c'} (\mathsf{wt}(c, c') + \mathsf{Val}(c')) & \text{for Max} \end{cases}$$

Open problem

And for 1 clock ?

Value Iteration for 1-clock WTG

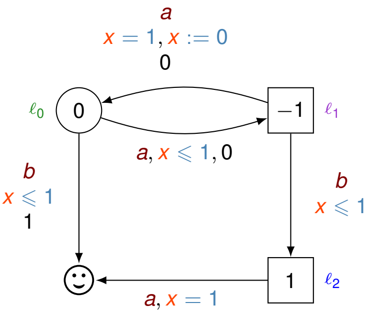
◯ Min ☐ Max



Value Iteration for 1-clock WTG

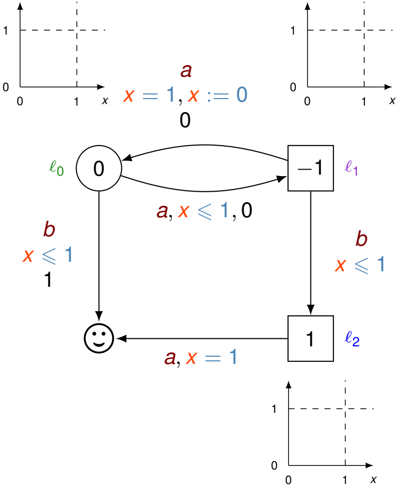
Min Max

Value Iteration



Value Iteration for 1-clock WTG

Min Max

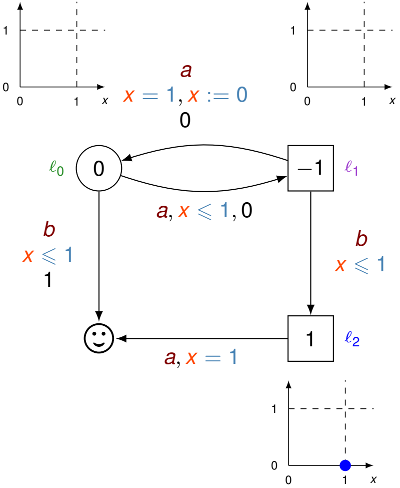


Value Iteration

- On piecewise affine functions

Value Iteration for 1-clock WTG

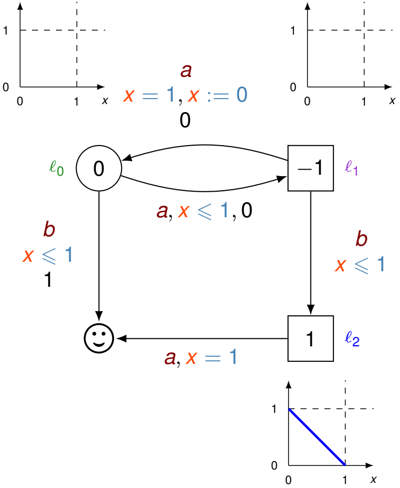
Min Max



Value Iteration

- On piecewise affine functions

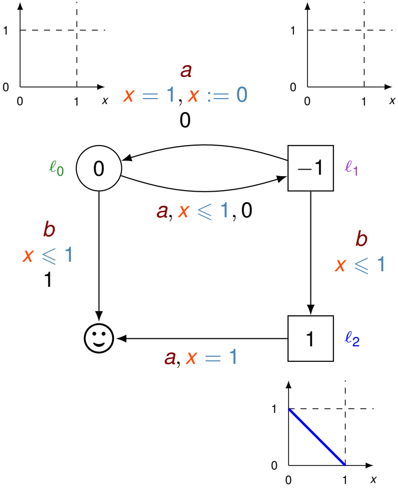
Value Iteration for 1-clock WTG



Value Iteration

- On piecewise affine functions

Value Iteration for 1-clock WTG

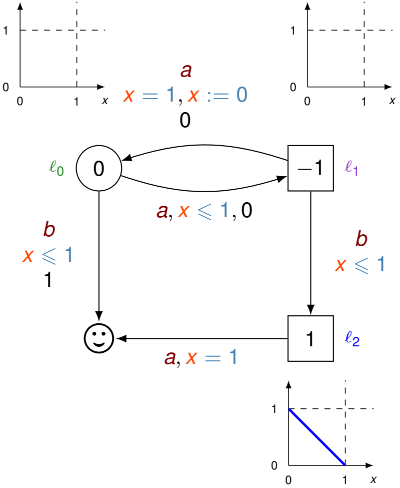


Value Iteration

- On piecewise affine functions

Strategies for Max

Value Iteration for 1-clock WTG



Value Iteration

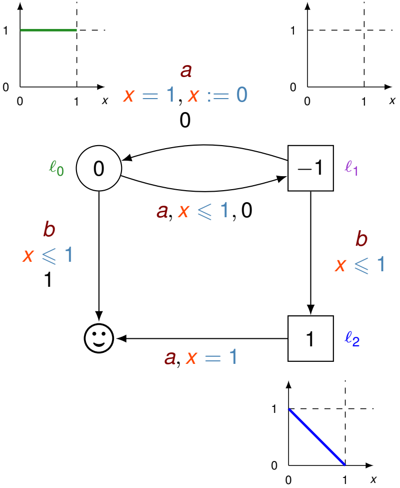
- On piecewise affine functions

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

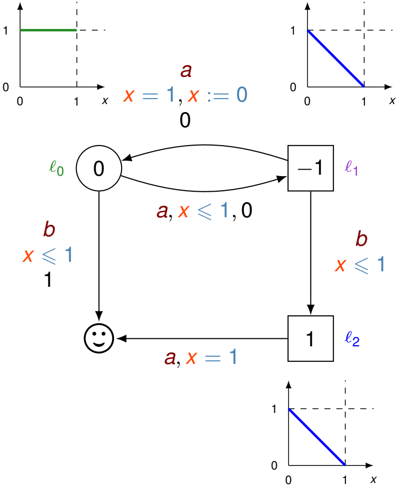
- On piecewise affine functions

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

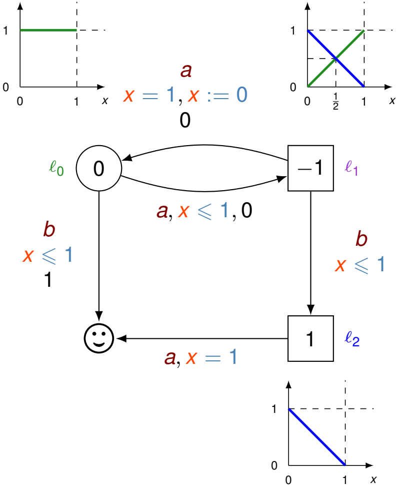
- On piecewise affine functions

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

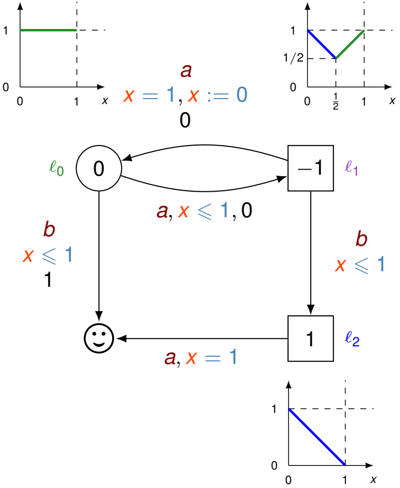
- On piecewise affine functions

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions

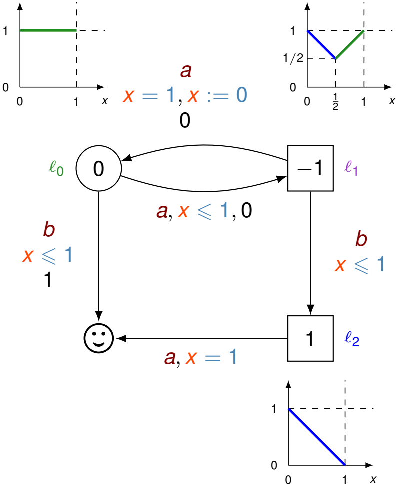
Strategies for Max

$\tau(\ell_2, x) = (a, 1 - x)$

$\tau(\ell_1, x) =$

Value Iteration for 1-clock WTG

○ Min □ Max



Value Iteration

- On piecewise affine functions

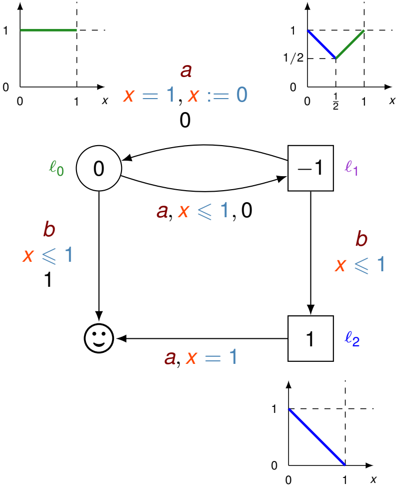
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions

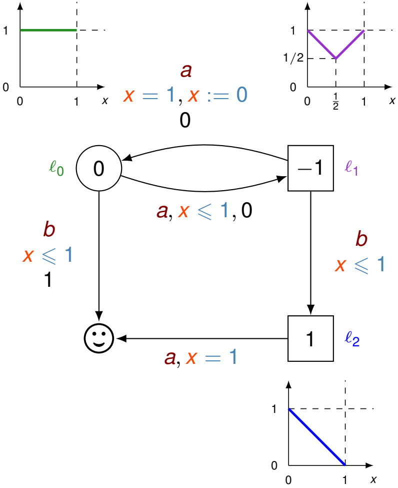
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions

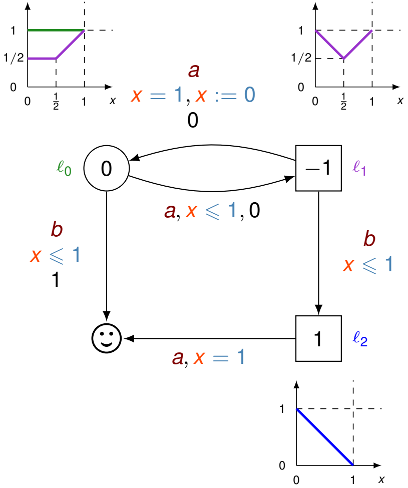
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

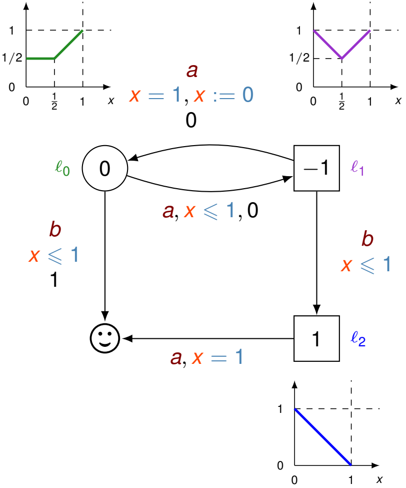
- On piecewise affine functions

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG



Value Iteration

- On piecewise affine functions

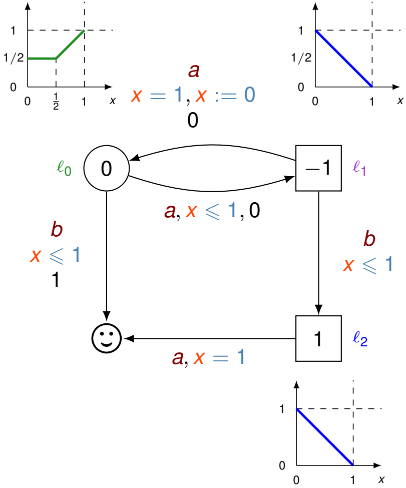
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions

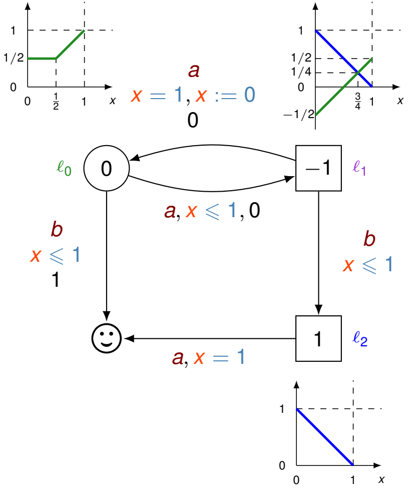
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

◯ Min ☐ Max



Value Iteration

- On piecewise affine functions

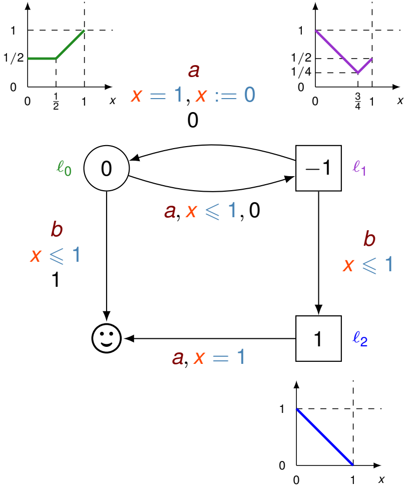
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

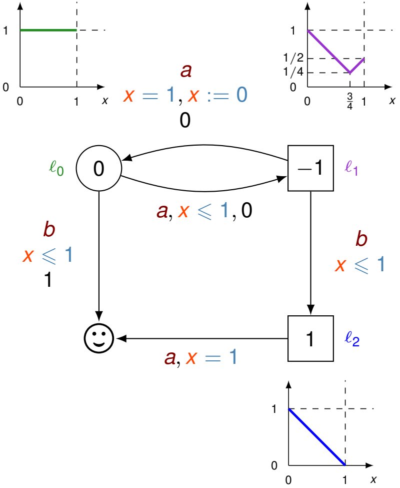
- On piecewise affine functions

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG



Value Iteration

- On piecewise affine functions

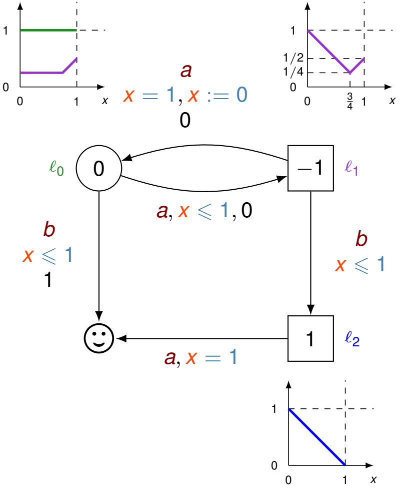
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions

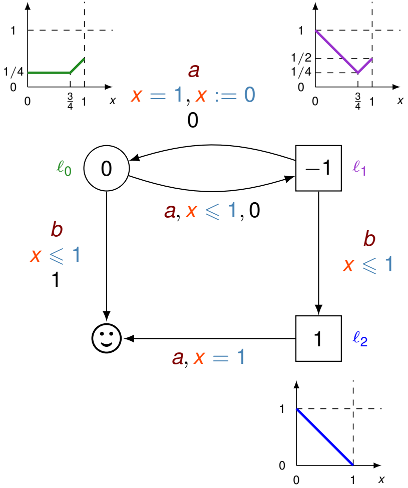
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions

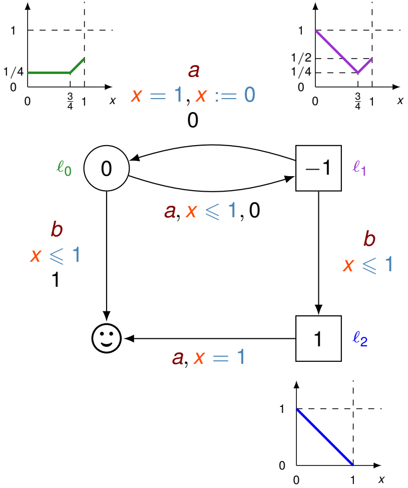
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG

Min Max



Value Iteration

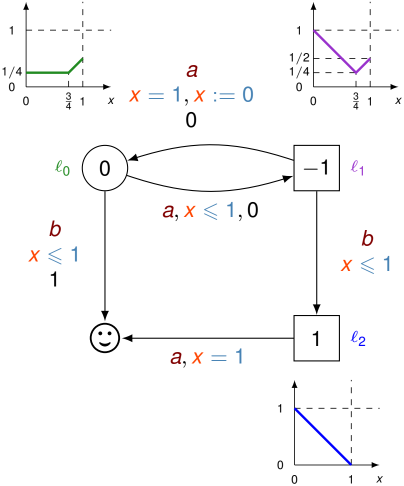
- On piecewise affine functions
- May not converge in finite time

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau(\ell_1, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1/2 \\ (b, 0) & \text{if } x \leqslant 1/2 \end{cases}$$

Value Iteration for 1-clock WTG



Value Iteration

- On piecewise affine functions
- May not converge in finite time

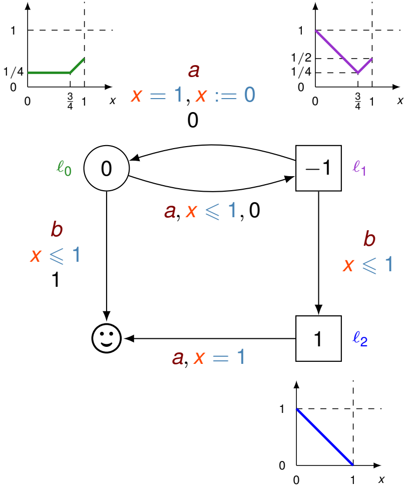
Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau({\ell_1}^2, x) = \begin{cases} (a, 1 - x) & \text{if } x > 3/4 \\ (b, 0) & \text{if } x \leqslant 3/4 \end{cases}$$

Value Iteration for 1-clock WTG

○ Min □ Max



Value Iteration

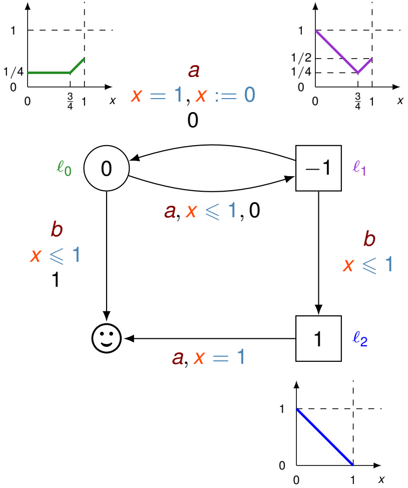
- On piecewise affine functions
- May not converge in finite time

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau({\ell_1}^i, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1 - \frac{1}{2^i} \\ (b, 0) & \text{if } x \leqslant 1 - \frac{1}{2^i} \end{cases}$$

Value Iteration for 1-clock WTG



Value Iteration

- On piecewise affine functions
- May not converge in finite time

Strategies for Max

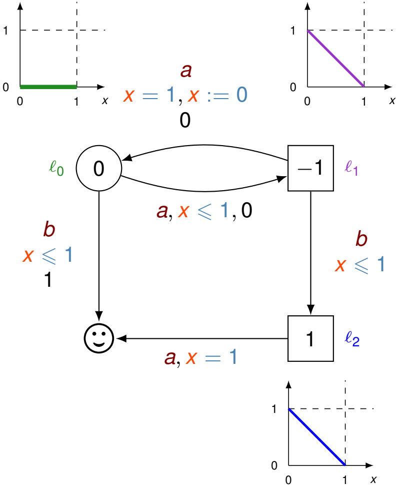
$$\tau(\ell_2, x) = (a, 1 - x)$$

$$\tau({\ell_1}^i, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1 - \frac{1}{2^i} \\ (b, 0) & \text{if } x \leqslant 1 - \frac{1}{2^i} \end{cases}$$

Max may need memory to play ε-optimally

Value Iteration for 1-clock WTG

Min Max



Value Iteration

- On piecewise affine functions
- May not converge in finite time
- Converges to Val

Strategies for Max

$$\tau(\ell_2, x) = (a, 1 - x)$$

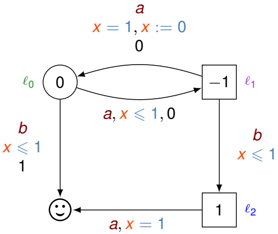
$$\tau({\ell_1}^i, x) = \begin{cases} (a, 1 - x) & \text{if } x > 1 - \frac{1}{2^i} \\ (b, 0) & \text{if } x \leqslant 1 - \frac{1}{2^i} \end{cases}$$

Max may need memory to play ε-optimally

Value problem for 1-clock WTG

Min Max

Deciding if $\mathsf{Val}(c) \leqslant \lambda$?

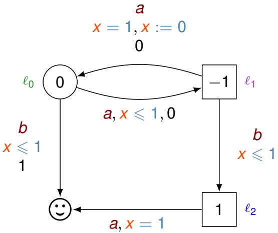


Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?

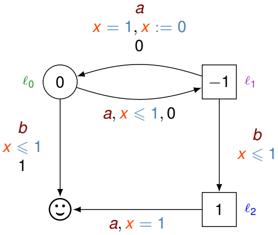
State of the art: 1-clock WTG



Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



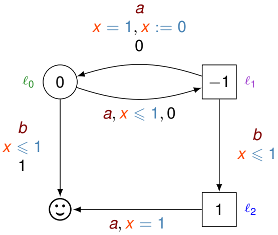
State of the art: 1-clock WTG

☹ Undecidable for 2 clocks

Value problem for 1-clock WTG

◯ Min ☐ Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



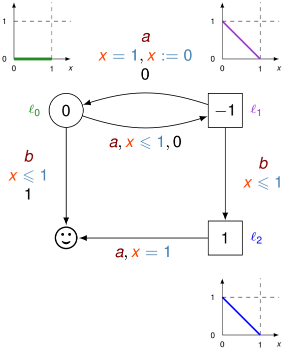
State of the art: 1-clock WTG

- ☹ Undecidable for 2 clocks
- ☹ Value Iteration

Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



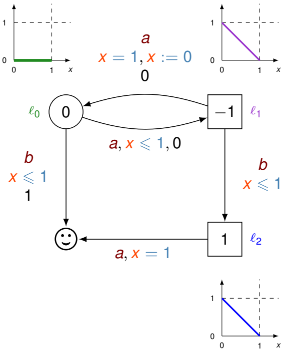
State of the art: 1-clock WTG

- ☹ Undecidable for 2 clocks
- ☹ Value Iteration: not in finite time

Value problem for 1-clock WTG

Min Max

Deciding if Val(c) $\leqslant \lambda$?



State of the art: 1-clock WTG

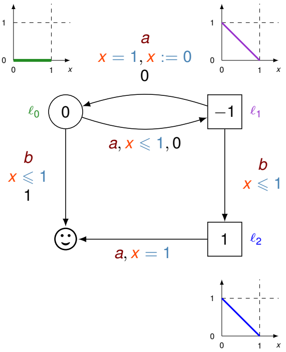
- ☹ Undecidable for 2 clocks
- ☹ Value Iteration: not in finite time
- ☺ Decidable with non-negative weights

Almost Optimal Strategies in One Clock Priced Timed Games, P. Bouyer, K. Larsen, N. Markey, and J. Rasmussen, 2006, FSTTCS

Value problem for 1-clock WTG

Min Max

Deciding if Val(c) $\leqslant \lambda$?



State of the art: 1-clock WTG

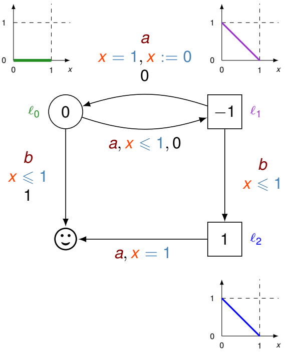
- Undecidable for 2 clocks
- Value Iteration: not in finite time
- Decidable with non-negative weights
- Decidable without cycle with reset

Simple Priced Timed Games Are Not That Simple, T. Brihaye, G. Geeraerts, A. Haddad, E. Lefaucheux, and B. Monmege, 2015, FSTTCS

Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



State of the art: 1-clock WTG

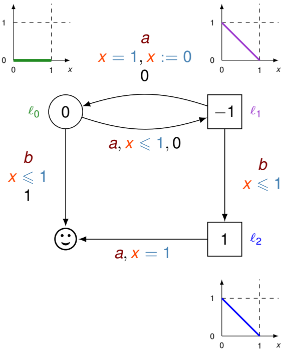
- ☹ Undecidable for 2 clocks
- ☹ Value Iteration: not in finite time
- ☺ Decidable with non-negative weights
- ☺ Decidable without cycle with reset

Back-time algorithm

Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



State of the art: 1-clock WTG

- Undecidable for 2 clocks
- Value Iteration: not in finite time
- Decidable with non-negative weights
- Decidable without cycle with reset

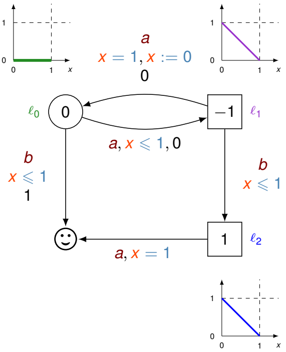
Back-time algorithm

Compute $c \mapsto \mathrm{Val}(c)$ from $x = 1$ to 0

Value problem for 1-clock WTG

○ Min □ Max

Deciding if Val(c) $\leqslant \lambda$?



State of the art: 1-clock WTG

- ☹ Undecidable for 2 clocks
- ☹ Value Iteration: not in finite time
- ☺ Decidable with non-negative weights
- ☺ Decidable without cycle with reset

Back-time algorithm

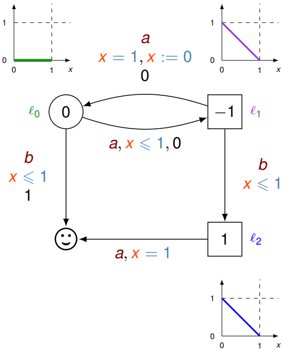
Compute $c \mapsto \text{Val}(c)$ from $x = 1$ to 0

⚠ Min needs an unbounded number of resets

Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



State of the art: 1-clock WTG

- Undecidable for 2 clocks
- Value Iteration: not in finite time
- Decidable with non-negative weights
- Decidable without cycle with reset

Back-time algorithm

Compute $c \mapsto \mathrm{Val}(c)$ from $x = 1$ to 0

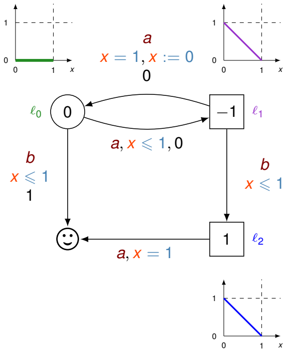
Min needs an unbounded number of resets

Decidable for 1-clock WTG

Value problem for 1-clock WTG

Min Max

Deciding if $\mathrm{Val}(c) \leqslant \lambda$?



State of the art: 1-clock WTG

- ☹ Undecidable for 2 clocks
- ☹ Value Iteration: not in finite time
- ☺ Decidable with non-negative weights
- ☺ Decidable without cycle with reset

Min needs an unbounded number of resets

Back-time algorithm

Compute $c \mapsto \mathrm{Val}(c)$ from $x = 1$ to 0

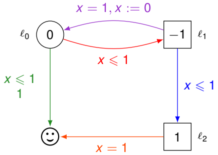
Decidable for 1-clock WTG

$c \mapsto \mathrm{Val}(c)$ is computable in exponential time

Contribution

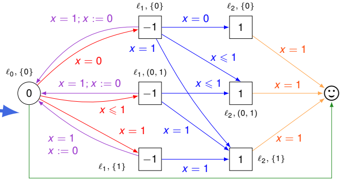
$c \mapsto \mathrm{Val}(c)$ is
computable

Contribution

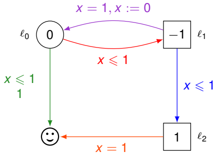


$c \mapsto \mathrm{Val}(c)$ is computable

Contribution

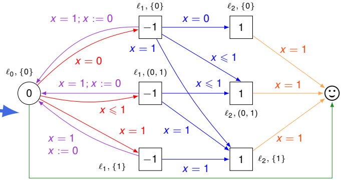


Encoding regions



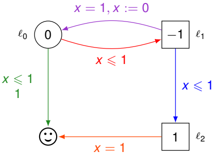
$c \mapsto \mathrm{Val}(c)$ is computable

Contribution

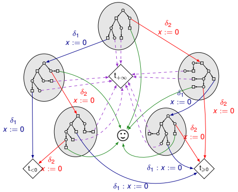


Encoding regions

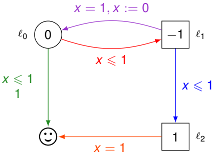
Finite unfolding



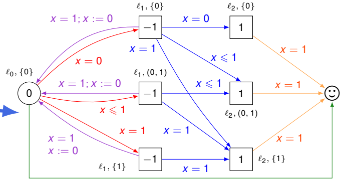
$c \mapsto \mathrm{Val}(c)$ is computable



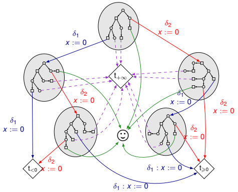
Contribution



Encoding regions

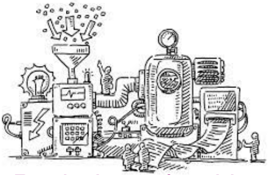


Finite unfolding



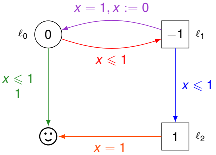
$c \mapsto \mathrm{Val}(c)$ is computable

Solving

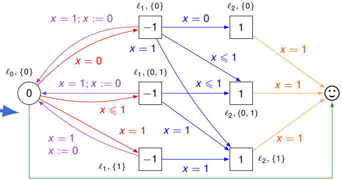


Back-time algorithm

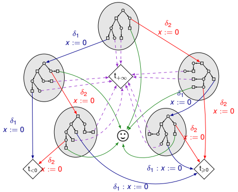
Contribution



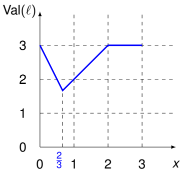
Encoding regions

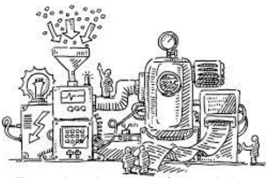


Finite unfolding



$c \mapsto \mathrm{Val}(c)$ is computable



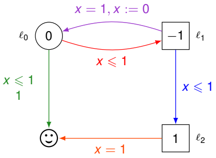


Back-time algorithm

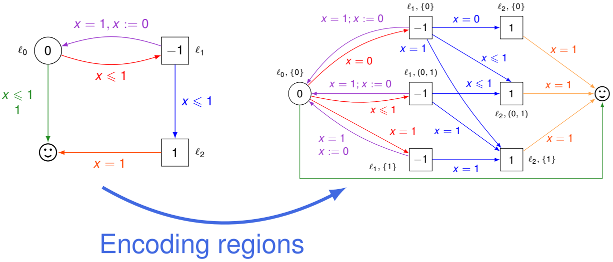
Solving

Ideas of the proof

Ideas of the proof



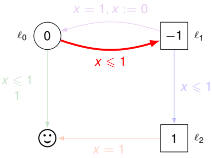
Ideas of the proof

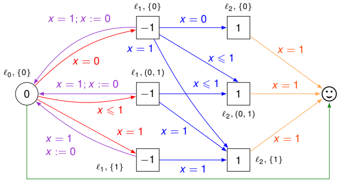


Simple Priced Timed Games Are Not That Simple, T. Brihaye, G. Geeraerts, A. Haddad, E. Lefaucheux, and B. Monmege, 2015, FSTTCS

Ideas of the proof

$(\ell_0, 0) \xrightarrow{1-\varepsilon} (\ell_1, 1-\varepsilon)$

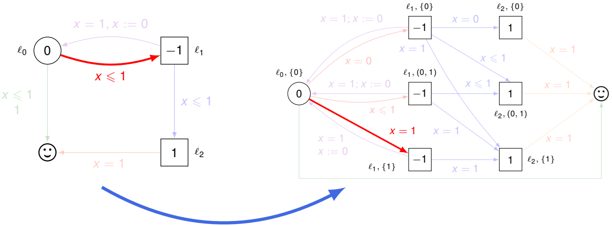




Simple Priced Timed Games Are Not That Simple, T. Brihaye, G. Geeraerts, A. Haddad, E. Lefaucheux, and B. Monmege, 2015, FSTTCS

Ideas of the proof

$(\ell_0, 0) \xrightarrow{1-\varepsilon} (\ell_1, 1-\varepsilon)$

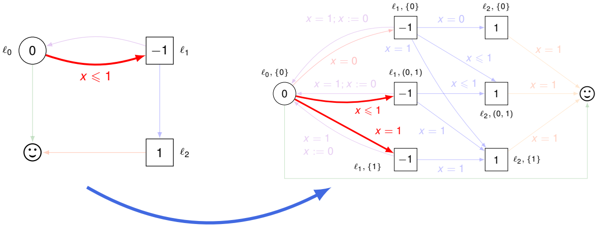


Encoding regions

Simple Priced Timed Games Are Not That Simple, T. Brihaye, G. Geeraerts, A. Haddad, E. Lefaucheux, and B. Monmege, 2015, FSTTCS

Ideas of the proof

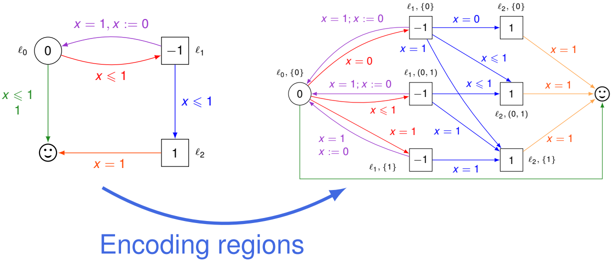
$(\ell_0, 0) \xrightarrow{1-\varepsilon} (\ell_1, 1-\varepsilon)$ ⟶ $((\ell_0, \{0\}), 0) \xrightarrow{1} ((\ell_1, (0,1)), 1)$



Encoding regions

Ideas of the proof

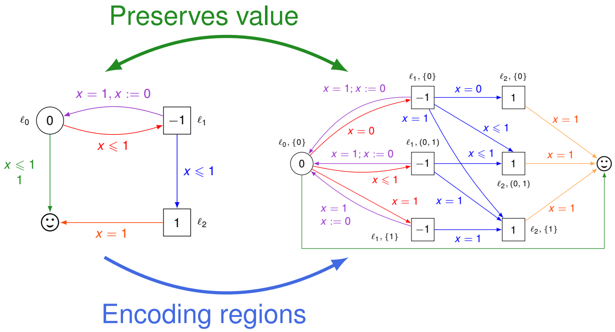
$(\ell_0, 0) \xrightarrow{1-\varepsilon} (\ell_1, 1-\varepsilon)$ ⟶ $((\ell_0, \{0\}), 0) \xrightarrow{1} ((\ell_1, (0,1)), 1)$



Main argument

Max has a memoryless optimal strategy in the region game

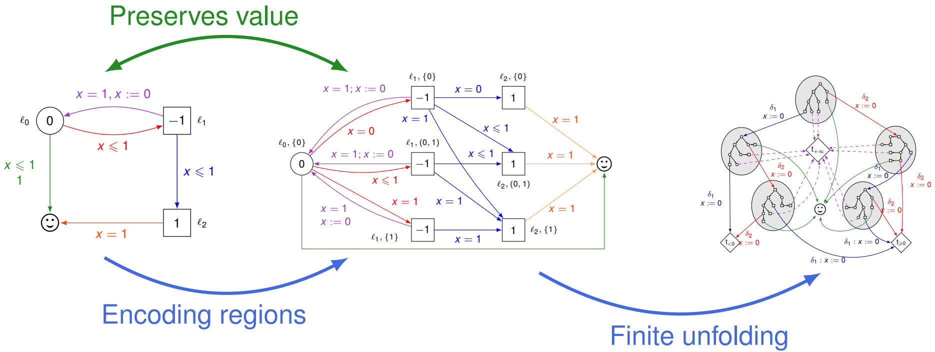
Ideas of the proof



Main argument

Max has a memoryless optimal strategy in the region game

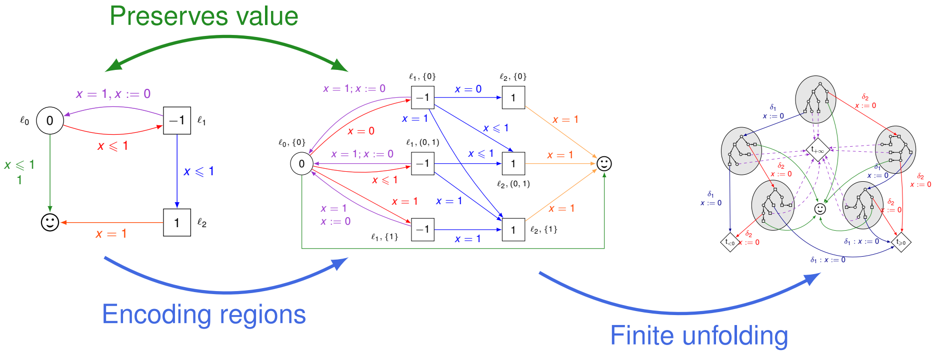
Ideas of the proof



Main argument

Max has a memoryless optimal strategy in the region game

Ideas of the proof

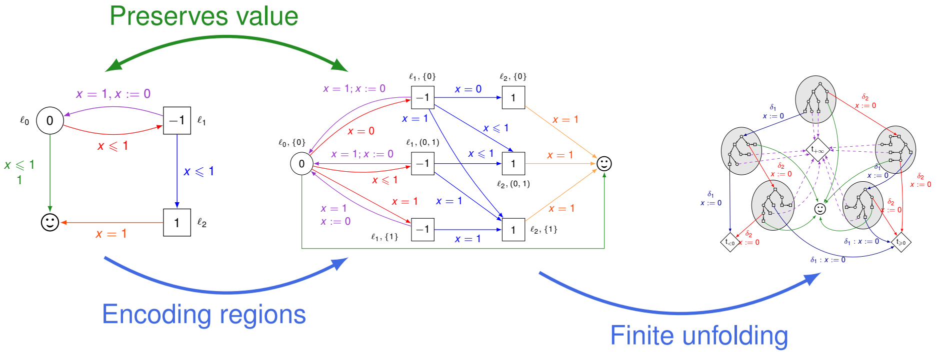


Main argument

Max has a memoryless optimal strategy in the region game

- Bounds the number of reset

Ideas of the proof

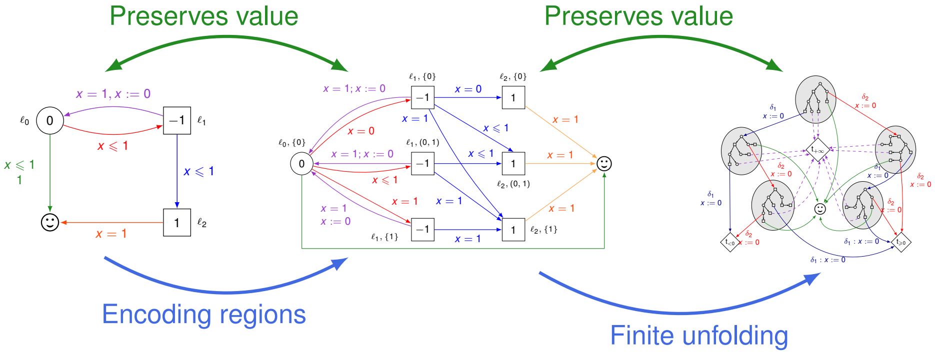


Main argument

Max has a memoryless optimal strategy in the region game

- Bounds the number of reset
- Acyclic WTG

Ideas of the proof



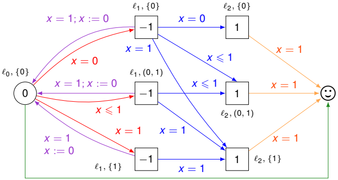
Main argument

Max has a memoryless optimal strategy in the region game

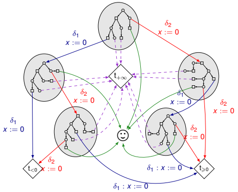
- Bounds the number of reset
- Acyclic WTG

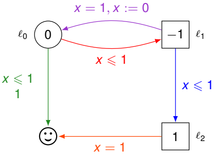
About complexity

Encoding
Regions

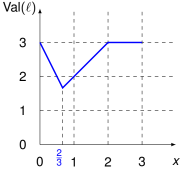


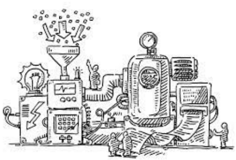
Finite
unfolding





$c \mapsto \mathrm{Val}(c)$ is
computable in
exponential time

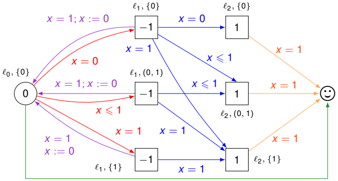




Back-time algorithm

Solving

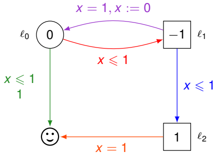
About complexity



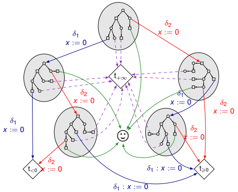
Encoding
Regions

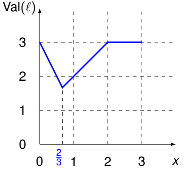
polynomial

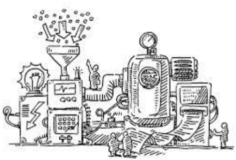
Finite
unfolding



$c \mapsto \mathrm{Val}(c)$ is
computable in
exponential time







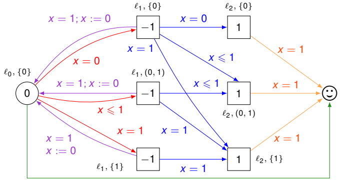
Back-time algorithm

Solving

About complexity

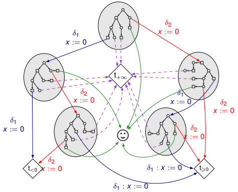
Encoding
Regions

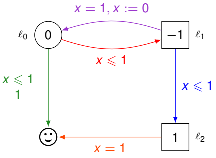
polynomial



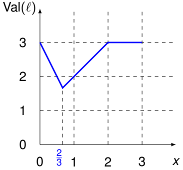
Finite
unfolding

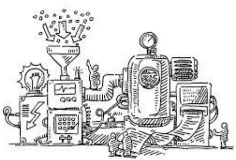
exponential





$c \mapsto \mathrm{Val}(c)$ is computable in exponential time





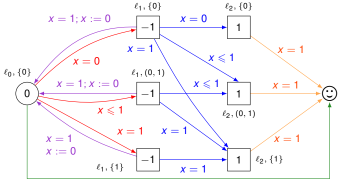
Back-time algorithm

Solving

About complexity

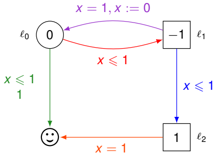
Encoding
Regions

polynomial

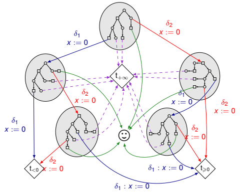


Finite
unfolding

exponential

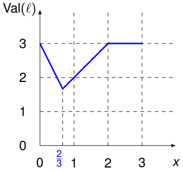


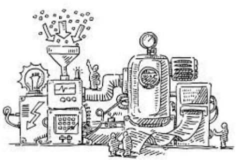
$c \mapsto \mathrm{Val}(c)$ is computable in exponential time



pseudo-
polynomial

Solving





Back-time algorithm

One-Clock Priced Timed Games with Negative Weights, T. Brihaye, G. Geeraerts, A. Haddad, E. Lefaucheux, B. Monmege, Log. Methods Comput. Sci., 2022

To conclude: Value problem in WTG

	Negative weights	Non-negative weights
1 clock		
2 clocks		
$\geqslant$ 3 clocks		

To conclude: Value problem in WTG

<table>
<tr><th></th><th>Negative weights</th><th>Non-negative weights</th></tr>
<tr><td>1 clock</td><td></td><td></td></tr>
<tr><td>2 clocks</td><td rowspan="2">Undecidable</td><td></td></tr>
<tr><td>$\geqslant$ 3 clocks</td><td>Undecidable</td></tr>
</table>

On Optimal Timed Strategies, T. Brihaye, V. Bruyère and J.-F. Raskin, 2005, FORMATS

Adding Negative Prices to Priced Timed Games, T. Brihaye, G. Geeraerts, S. Krishna, L. Manasa, B. Monmege, A. Trivedi, 2014, CONCUR 2014

To conclude: Value problem in WTG

	Negative weights	Non-negative weights
1 clock		Exponential
2 clocks	Undecidable	
$\geqslant$ 3 clocks		Undecidable

Almost Optimal Strategies in One Clock Priced Timed Games, P. Bouyer, K. Larsen, N. Markey, and J. Rasmussen, 2006, FSTTCS

To conclude: Value problem in WTG

	Negative weights	Non-negative weights
1 clock	Exponential	Exponential
2 clocks	Undecidable	
$\geqslant$ 3 clocks		Undecidable

To conclude: Value problem in WTG

	Negative weights	Non-negative weights
1 clock	Exponential	Exponential
2 clocks	Undecidable	Open
$\geqslant$ 3 clocks		Undecidable

To conclude: Value problem in WTG

	Negative weights	Non-negative weights
1 clock	Exponential PSPACE-hard	Exponential PSPACE-hard
2 clocks	Undecidable	Open
$\geqslant$ 3 clocks	Undecidable	Undecidable

One-clock priced timed games are PSPACE-hard., J. Fearnley, R. Ibsen-Jensen, and R. Savani, 2020, LICS

To conclude: Value problem in WTG

	Negative weights	Non-negative weights
1 clock	Exponential	Exponential
	PSPACE-hard	
2 clocks	Undecidable	Open
$\geqslant$ 3 clocks		Undecidable

Thank you! Questions ?