Weighted Timed Games: Decidability, Randomisation and Robustness

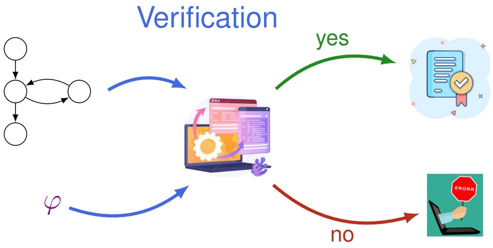
Julie Parreaux

University of Warsaw

Séminaire M2F

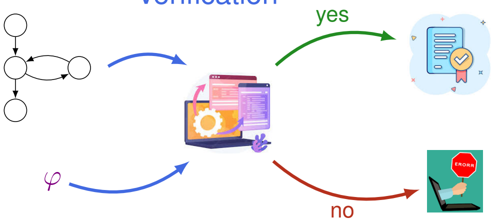
Joint work with Benjamin Monmege and Pierre–Alain Reynier

Correctness and performance of real-time systems



Correctness and performance of real-time systems

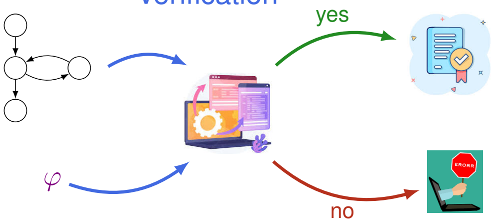
Verification



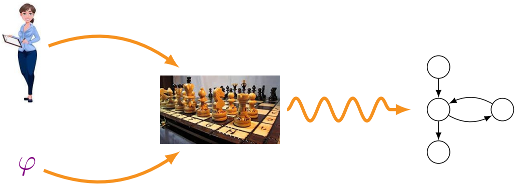
Synthesis

Correctness and performance of real-time systems

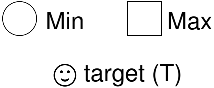
Verification

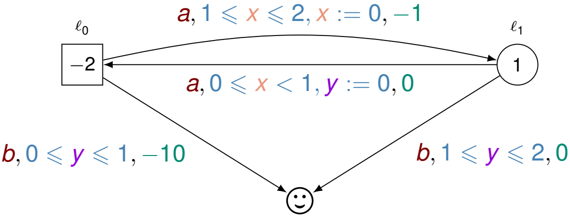


Synthesis

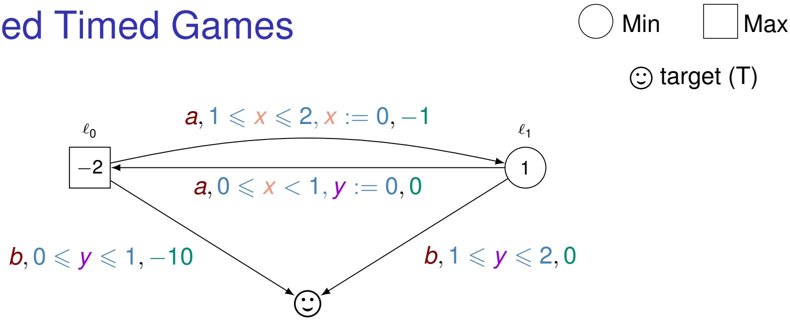


Weighted Timed Games



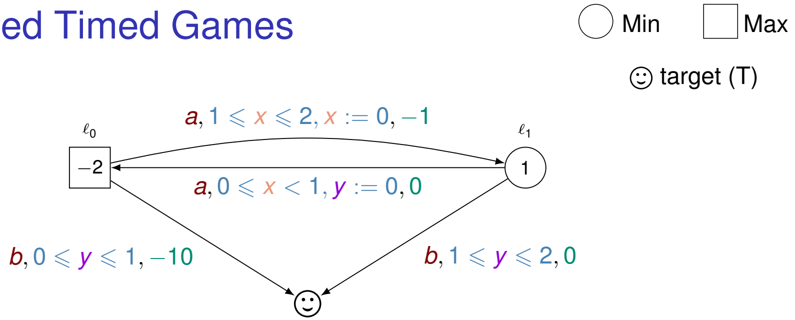


Weighted Timed Games



Play ρ $\left(\ell_1, \begin{bmatrix} x \mapsto 0 \\ y \mapsto 0 \end{bmatrix}\right)$

Weighted Timed Games

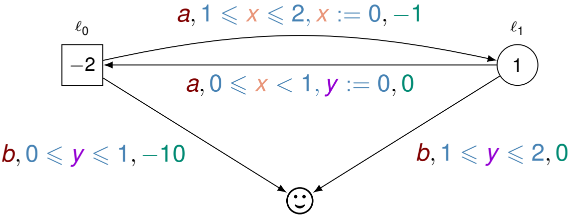


Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Weighted Timed Games

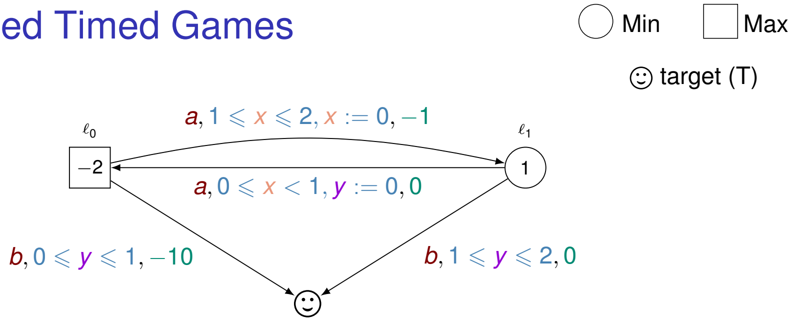
○ Min □ Max

☺ target (T)



Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a}$

Weighted Timed Games

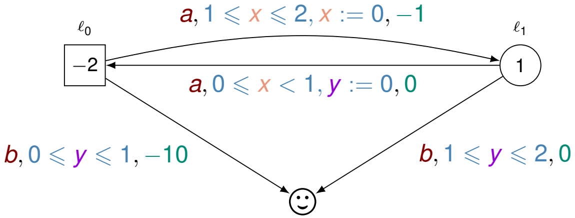


Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix})$

Weighted Timed Games

○ Min ☐ Max

☺ target (T)

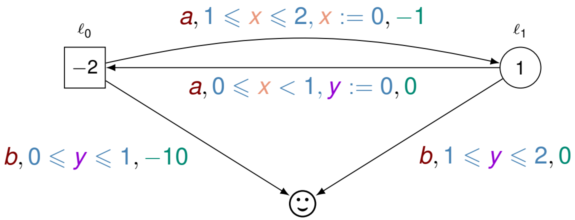


Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$

Weighted Timed Games

○ Min □ Max

☺ target (T)



Play ρ

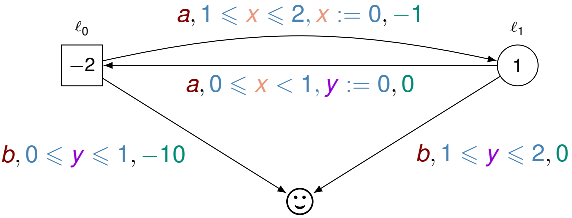
$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$$

$$+ \qquad\qquad\qquad +$$

Weighted Timed Games

○ Min □ Max

☺ target (T)



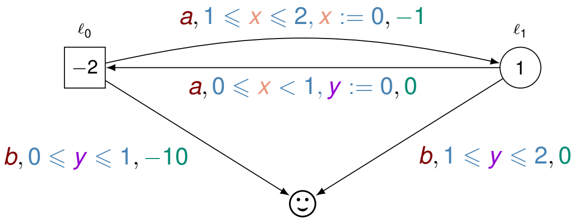
Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$

$0 \quad + \quad\quad\quad +$

Weighted Timed Games

○ Min □ Max

☺ target (T)



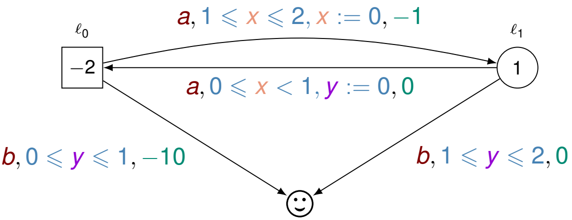
Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$

$1 \times 0.5 + 0 \quad + \quad\quad +$

Weighted Timed Games

○ Min □ Max

☺ target (T)



Play ρ

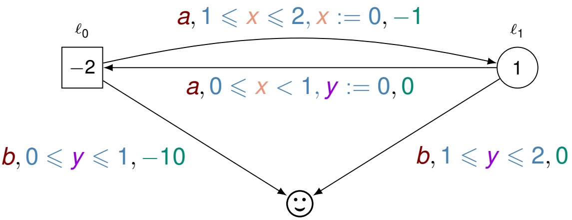
$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix}) \rightsquigarrow -\frac{8}{3}$$

$$1 \times 0.5 + 0 \quad + \quad -2 \times 1.25 - 1 \quad + \quad 1 \times \tfrac{1}{3} + 0$$

Weighted Timed Games

○ Min ☐ Max

☺ target (T)



Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$

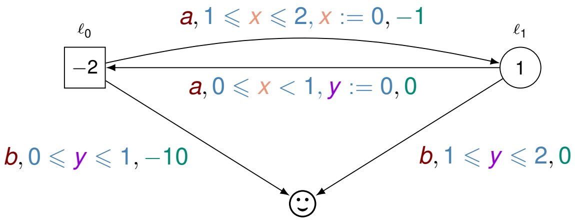
Deterministic strategy

Choose an edge and a delay

Weighted Timed Games

○ Min □ Max

☺ target (T)



Play ρ $\quad (\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$

Deterministic strategy

Choose an edge and a delay

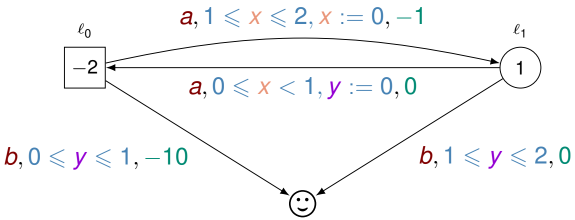
From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

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

Weighted Timed Games

○ Min □ Max

☺ target (T)



Play ρ $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{0.5,\ a} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{1.25,\ a} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{1/3,\ b} (☺, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$

Deterministic strategy

Choose an edge and a delay

From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

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

What features on strategies are needed for Min?

Features on strategies needed for Min

σ Min

τ Max

Features on strategies needed for Min

σ Min

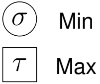
τ Max

Deterministic value

$\mathsf{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathsf{Play}(c, \sigma, \tau))$

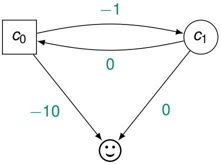
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



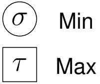
Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$$



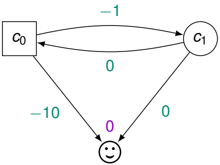
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



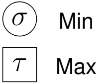
Deterministic value

$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$



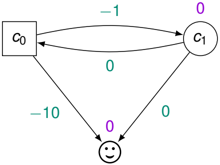
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



Deterministic value

$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$



Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

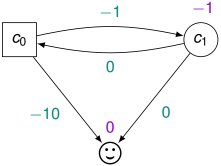
Features on strategies needed for Min

σ Min

τ Max

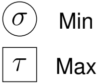
Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$$



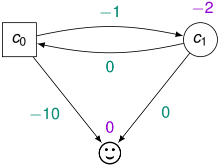
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



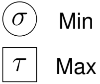
Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$$



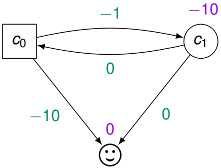
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



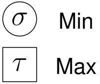
Deterministic value

$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$



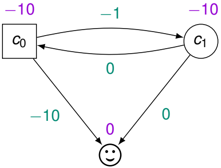
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))$$



Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

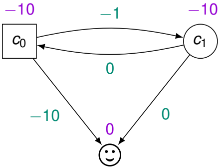
Features on strategies needed for Min

σ Min

τ Max

Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))}_{\mathrm{dVal}^{\sigma}(c)}$$

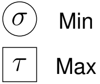


Optimal strategy for Min

$$\mathrm{dVal}^{\sigma}(c) \leqslant \mathrm{dVal}(c)$$

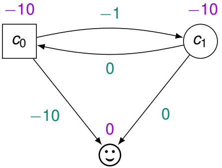
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



Deterministic value

$$\mathrm{dVal}(c) = \inf_\sigma \underbrace{\sup_\tau \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))}_{\mathrm{dVal}^\sigma(c)}$$



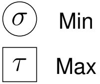
Optimal strategy for Min

$$\mathrm{dVal}^\sigma(c) \leqslant \mathrm{dVal}(c)$$

Finite memory

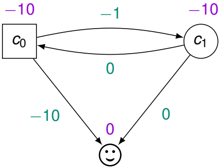
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



Deterministic value

$$\mathsf{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathsf{Play}(c, \sigma, \tau))}_{\mathsf{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

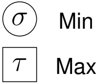
$$\mathsf{dVal}^{\sigma}(c) \leqslant \mathsf{dVal}(c)$$

Finite memory

Switching strategy:

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min

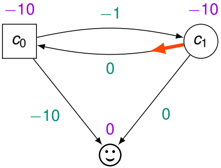


Deterministic value

$$\mathrm{dVal}(c) = \inf_\sigma \underbrace{\sup_\tau \mathbf{cost}(\mathrm{Play}(c,\sigma,\tau))}_{\mathrm{dVal}^\sigma(c)}$$

Optimal strategy for Min

$$\mathrm{dVal}^\sigma(c) \leqslant \mathrm{dVal}(c)$$



Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

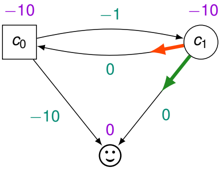
Features on strategies needed for Min

σ Min

τ Max

Deterministic value

$$\mathsf{dVal}(c) = \inf_\sigma \underbrace{\sup_\tau \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^\sigma(c)}$$



Optimal strategy for Min

$$\mathsf{dVal}^\sigma(c) \leqslant \mathsf{dVal}(c)$$

Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

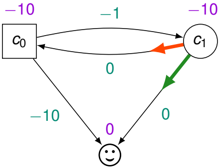
Features on strategies needed for Min

σ Min

τ Max

Deterministic value

$$\text{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \textbf{cost}(\text{Play}(c, \sigma, \tau))}_{\text{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$$\text{dVal}^{\sigma}(c) \leqslant \text{dVal}(c)$$

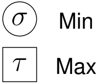
Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

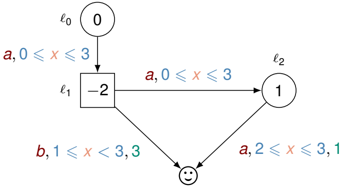
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



Deterministic value

$$\mathsf{dVal}(c) = \inf_\sigma \underbrace{\sup_\tau \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^\sigma(c)}$$



Optimal strategy for Min

$\mathsf{dVal}^\sigma(c) \leqslant \mathsf{dVal}(c)$

Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

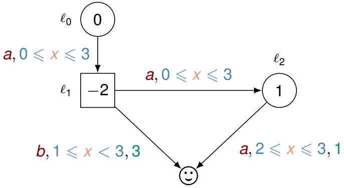
Features on strategies needed for Min

σ Min

τ Max

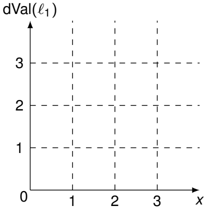
Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))}_{\mathrm{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$$\mathrm{dVal}^{\sigma}(c) \leqslant \mathrm{dVal}(c)$$



Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

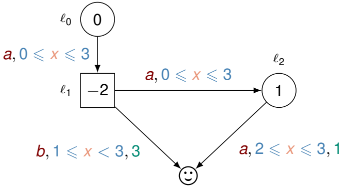
Features on strategies needed for Min

σ Min

τ Max

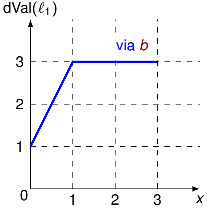
Deterministic value

$$\mathsf{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$$\mathsf{dVal}^{\sigma}(c) \leqslant \mathsf{dVal}(c)$$



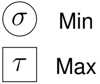
Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

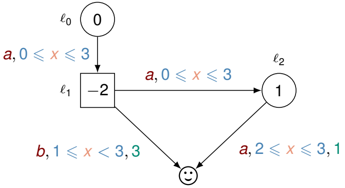
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games, T. Brihaye, G. Geeraerts, A. Haddad and B. Monmege, 2016, Acta Informatica

Features on strategies needed for Min



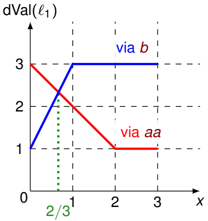
Deterministic value

$$\mathrm{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathrm{Play}(c, \sigma, \tau))}_{\mathrm{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$\mathrm{dVal}^{\sigma}(c) \leqslant \mathrm{dVal}(c)$

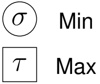


Finite memory

Switching strategy:

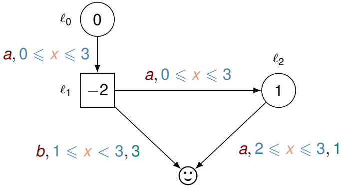
- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

Features on strategies needed for Min



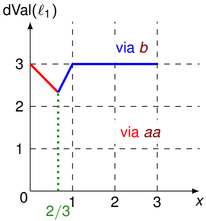
Deterministic value

$$\mathsf{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$$\mathsf{dVal}^{\sigma}(c) \leqslant \mathsf{dVal}(c)$$

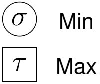


Finite memory

Switching strategy:

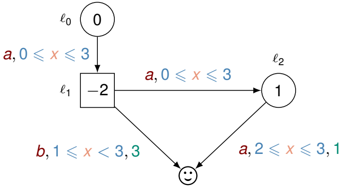
- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

Features on strategies needed for Min



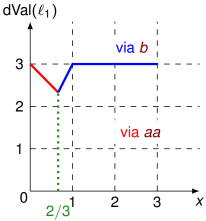
Deterministic value

$$\mathsf{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$\mathsf{dVal}^{\sigma}(c) \leqslant \mathsf{dVal}(c)$



Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

Infinite precision

From ℓ_0, Min wants to reach the valuation $2/3$

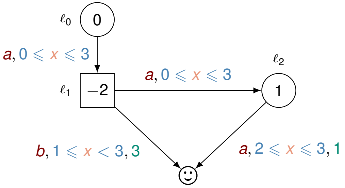
Features on strategies needed for Min

σ Min

τ Max

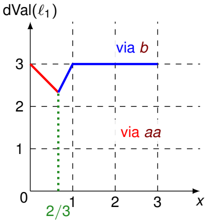
Deterministic value

$$\mathsf{dVal}(c) = \inf_\sigma \underbrace{\sup_\tau \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^\sigma(c)}$$



Optimal strategy for Min

$\mathsf{dVal}^\sigma(c) \leqslant \mathsf{dVal}(c)$



Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

Infinite precision

From ℓ_0, Min wants to reach the valuation $2/3$

- if $x \leqslant 2/3$: Min plays $2/3 - x$

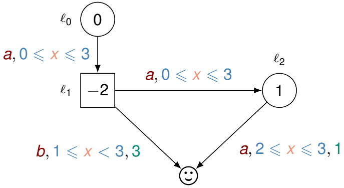
Features on strategies needed for Min

σ Min

τ Max

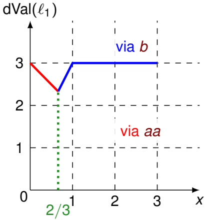
Deterministic value

$$\mathsf{dVal}(c) = \inf_{\sigma} \underbrace{\sup_{\tau} \mathbf{cost}(\mathsf{Play}(c,\sigma,\tau))}_{\mathsf{dVal}^{\sigma}(c)}$$



Optimal strategy for Min

$$\mathsf{dVal}^{\sigma}(c) \leqslant \mathsf{dVal}(c)$$



Finite memory

Switching strategy:

- σ_1: reach cycle with a weight $\leqslant -1$
- σ_2: reach ☺
- K: number of turns before switch

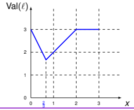
Infinite precision

From ℓ_0, Min wants to reach the valuation $2/3$

- if $x \leqslant 2/3$: Min plays $2/3 - x$
- otherwise, Min plays 0

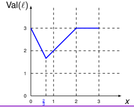
Problems on weighted timed games

Deterministic value problem



Problems on weighted timed games

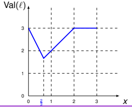
Deterministic value problem



Trading memory with probabilities

Problems on weighted timed games

Deterministic value problem

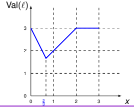


Trading memory with probabilities

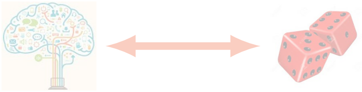
Robust optimal strategies

Problems on weighted timed games

Deterministic value problem



Trading memory with probabilities



Robust optimal strategies

Deterministic value problem

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

Deterministic value problem

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

	WTG			
$\mathbb{N}$	undecidable			
$\mathbb{Z}$	undecidable			

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, and A. Trivedi, 2014, CONCUR

Deterministic value problem

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

	WTG	0-clock		
$\mathbb{N}$	undecidable			
$\mathbb{Z}$	undecidable			

Deterministic value problem

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

	WTG	0-clock		
$\mathbb{N}$	undecidable	PTIME		
$\mathbb{Z}$	undecidable	pseudo-polynomial		

On Short Paths Interdiction Problems: Total and Node-Wise Limited Interdiction, L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao, 2008, Theory of Computing Systems

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

Deterministic value problem

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

	WTG	0-clock	divergent	
$\mathbb{N}$	undecidable	PTIME		
$\mathbb{Z}$	undecidable	pseudo-polynomial		

Deterministic value problem

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

	WTG	0-clock	divergent	
$\mathbb{N}$	undecidable	PTIME		
$\mathbb{Z}$	undecidable	pseudo-polynomial		

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

Optimal Reachability for Weighted Timed Game., R. Alur, M. Bernadsky, and P. Madhusudan, 2004, ICALP

Optimal Strategies in Priced Timed Game Automata, P. Bouyer, F. Cassez, E.l Fleury, and K. Larsen, 2004, FSTTCS

Optimal Reachability in Divergent Weighted Timed Games., D. Busatto-Gaston, B. Monmege, and P.-A. Reynier, 2017, FOSSACS

Deterministic value problem

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

	WTG	0-clock	divergent	
$\mathbb{N}$	undecidable	PTIME	EXPTIME	
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

Optimal Reachability for Weighted Timed Game., R. Alur, M. Bernadsky, and P. Madhusudan, 2004, ICALP

Optimal Strategies in Priced Timed Game Automata, P. Bouyer, F. Cassez, E.l Fleury, and K. Larsen, 2004, FSTTCS

Optimal Reachability in Divergent Weighted Timed Games., D. Busatto-Gaston, B. Monmege, and P.-A. Reynier, 2017, FOSSACS

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

Almost optimal strategies in one clock priced timed games, P. Bouyer, K. Larsen, N. Markey, and J. Rasmussen, 2006, FSTTCS

Two-Player Reachability-Price Games on Single Clock Timed Automata., M. Rutkowski, 2011, QAPL

A Faster Algorithm for Solving One-Clock Priced Timed Games, T. Dueholm Hansen, R. Ibsen-Jensen, and P. Bro Miltersen, 2013, CONCUR

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

PSPACE lower bound

The deterministic value problem is PSPACE-hard for 1-clock WTG

One-Clock Priced Timed Games are PSPACE-hard., J. Fearnley, R. Ibsen-Jensen, and R. Savani, 2020, LICS

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

PSPACE lower bound

The deterministic value problem is PSPACE-hard for 1-clock WTG

Theorem (CONCUR'22): the problem is decidable for 1-clock WTG

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

PSPACE lower bound

The deterministic value problem is PSPACE-hard for 1-clock WTG

Theorem (CONCUR'22): the problem is decidable for 1-clock WTG

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

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

PSPACE lower bound

The deterministic value problem is PSPACE-hard for 1-clock WTG

Theorem (CONCUR'22): the problem is decidable for 1-clock WTG

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

- Back-time algorithm: compute $c \mapsto \mathrm{Val}(c)$ from $x = 1$ to 0

Deterministic value problem

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

	WTG	0-clock	divergent	1-clock
$\mathbb{N}$	undecidable	PTIME	EXPTIME	EXPTIME
$\mathbb{Z}$	undecidable	pseudo-polynomial	3-EXPTIME	

Property of divergence

All SCCs of the WTG contain only cycles with a weight $\leqslant -1$ or $\geqslant 1$

PSPACE lower bound

The deterministic value problem is PSPACE-hard for 1-clock WTG

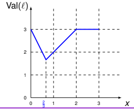
Theorem (CONCUR'22): the problem is decidable for 1-clock WTG

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

- Back-time algorithm: compute $c \mapsto \mathrm{Val}(c)$ from $x = 1$ to 0
- Value iteration algorithm: deterministic value is a fixed point of a given operator

Problems on weighted timed games

Deterministic value problem



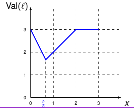
Trading memory with probabilities

Robust optimal strategies

Problems on weighted timed games

Decidability for 1-clock WTG

Deterministic value problem



Trading memory with probabilities

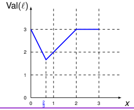
Robust optimal strategies

Problems on weighted timed games

Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Deterministic value problem



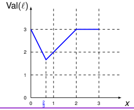
Trading memory with probabilities

Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Trading memory with probabilities

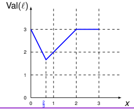
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Trading memory with probabilities

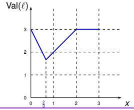
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



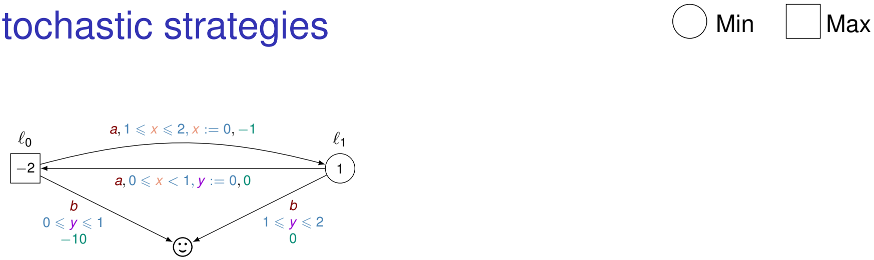
Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Trading memory with probabilities

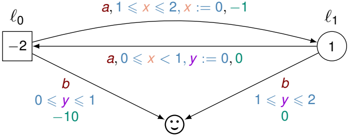
Robust optimal strategies

Stochastic strategies



Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies



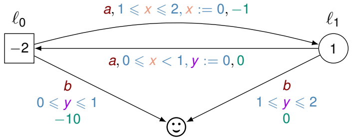
Stochastic strategy

Distribution over possible choices

Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies

Min Max



Stochastic strategy

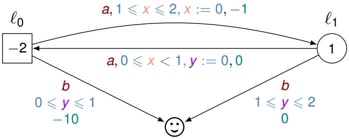
Distribution over possible choices

1. Edge *a*: finite distribution

Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies

Min Max



Stochastic strategy

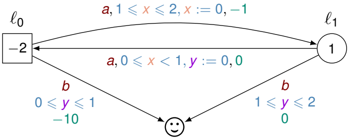
Distribution over possible choices

1. Edge *a*: finite distribution
2. Delay for *a*: infinite distribution

Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies

○ Min □ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between *a* or *b* with $\mathcal{B}(\frac{1}{2})$

Stochastic strategy

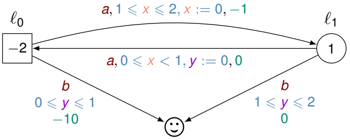
Distribution over possible choices

1. Edge *a*: finite distribution
2. Delay for *a*: infinite distribution

Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies

◯ Min ☐ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between a or b with $\mathcal{B}(\frac{1}{2})$

- a: choose t with $\mathcal{U}([0, 1))$

Stochastic strategy

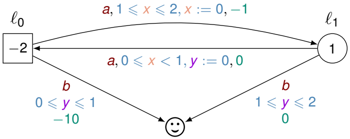
Distribution over possible choices

1. Edge a: finite distribution
2. Delay for a: infinite distribution

Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies

○ Min ☐ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between *a* or *b* with $\mathcal{B}(\frac{1}{2})$

- *a*: choose *t* with $\mathcal{U}([0, 1))$
- *b*: choose *t* with $\delta_{1.5}$

Stochastic strategy

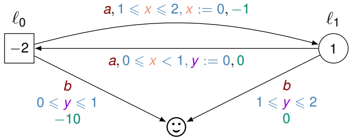
Distribution over possible choices

1. Edge *a*: finite distribution
2. Delay for *a*: infinite distribution

Stochastic Timed Automata, N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, and M. Jurdzinzki, 2014, Logical Methods in Computer Science

Stochastic strategies

η Min θ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between *a* or *b* with $\mathcal{B}(\frac{1}{2})$

- *a*: choose *t* with $\mathcal{U}([0, 1))$
- *b*: choose *t* with $\delta_{1.5}$

Stochastic strategy

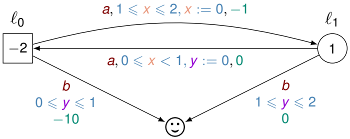
Distribution over possible choices

1. Edge *a*: finite distribution
2. Delay for *a*: infinite distribution

When we fix two strategies

Stochastic strategies

η Min θ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between a or b with $\mathcal{B}(\frac{1}{2})$

- a: choose t with $\mathcal{U}([0, 1))$
- b: choose t with $\delta_{1.5}$

Stochastic strategy

Distribution over possible choices

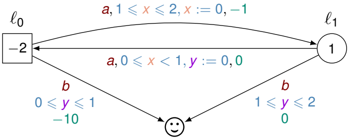
1. Edge a: finite distribution
2. Delay for a: infinite distribution

When we fix two strategies

- Infinite Markov Chain

Stochastic strategies

η Min θ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between *a* or *b* with $\mathcal{B}(\frac{1}{2})$

- *a*: choose *t* with $\mathcal{U}([0, 1))$
- *b*: choose *t* with $\delta_{1.5}$

Stochastic strategy

Distribution over possible choices

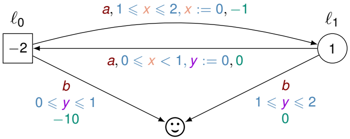
1. Edge *a*: finite distribution
2. Delay for *a*: infinite distribution

When we fix two strategies

- Infinite Markov Chain
- Replace **cost**(Play(c, η, θ)) by $\mathbb{E}_c^{\eta,\theta}$(**cost**)

Stochastic strategies

η Min θ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between a or b with $\mathcal{B}(\frac{1}{2})$

- a: choose t with $\mathcal{U}([0, 1))$
- b: choose t with $\delta_{1.5}$

Stochastic strategy

Distribution over possible choices

1. Edge a: finite distribution
2. Delay for a: infinite distribution

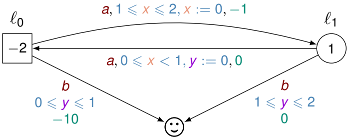
When we fix two strategies

- Infinite Markov Chain
- Replace **cost**(Play(c, η, θ)) by $\mathbb{E}_c^{\eta,\theta}$(**cost**)

Measurability conditions on η and θ

Stochastic strategies

η Min θ Max



From $(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$

Choose between a or b with $\mathcal{B}(\frac{1}{2})$

- a: choose t with $\mathcal{U}([0, 1))$
- b: choose t with $\delta_{1.5}$

Stochastic strategy

Distribution over possible choices

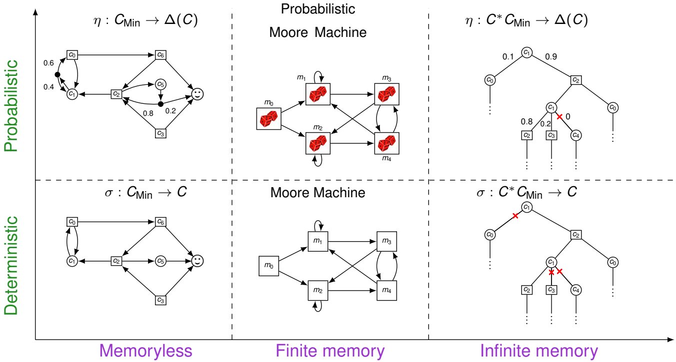
1. Edge a: finite distribution
2. Delay for a: infinite distribution

When we fix two strategies

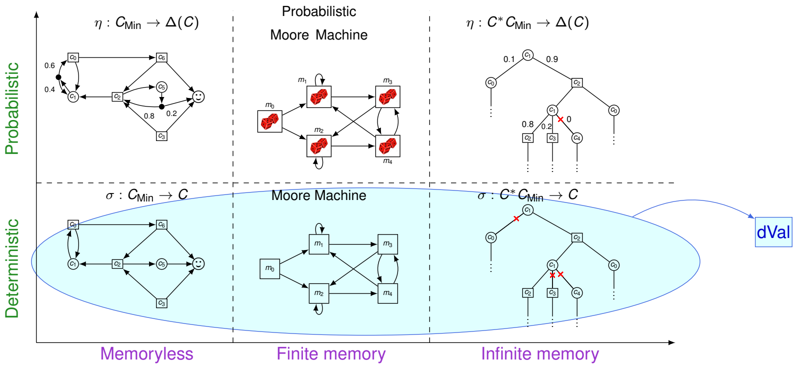
- Infinite Markov Chain
- Replace $\mathbf{cost}(\mathrm{Play}(c, \eta, \theta))$ by $\mathbb{E}_c^{\eta,\theta}(\mathbf{cost})$

Measurability conditions on η and θ

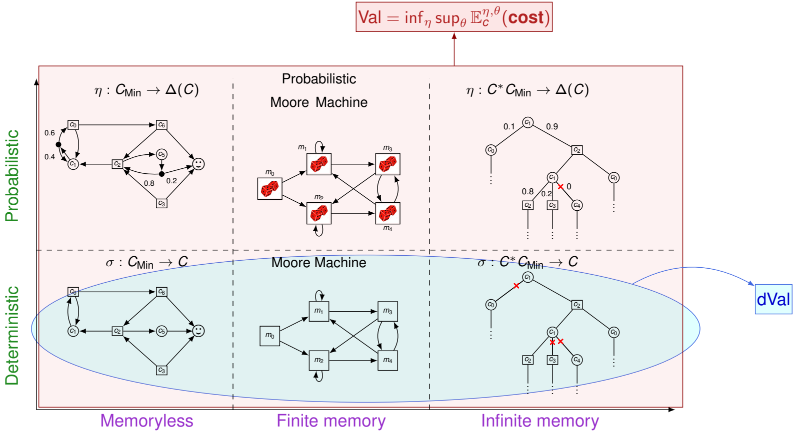
Stochastic values



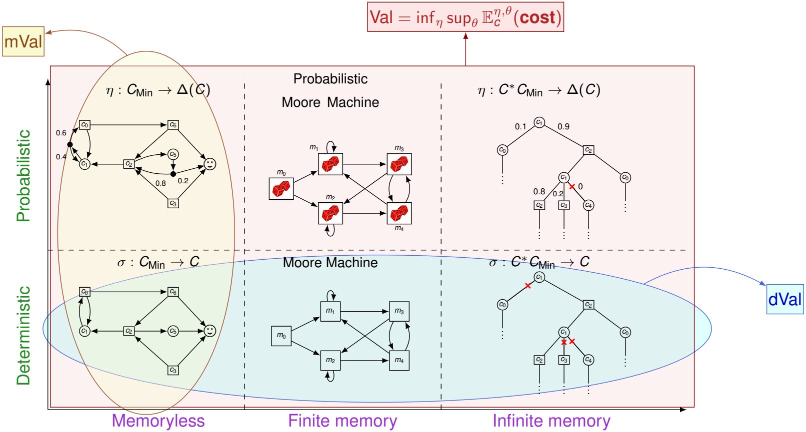
Stochastic values



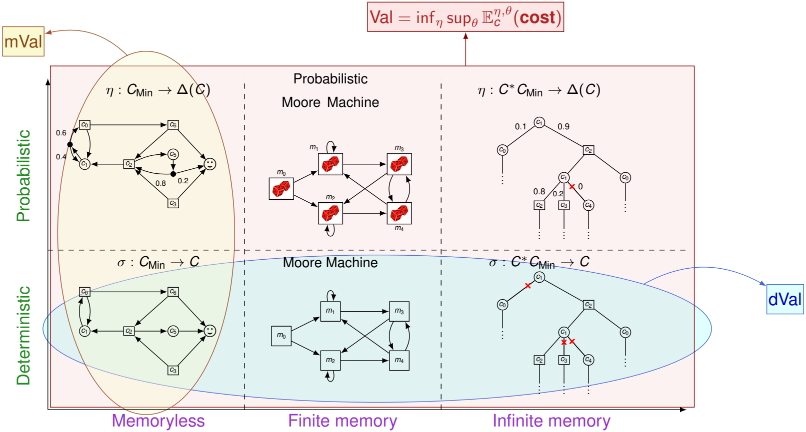
Stochastic values



Stochastic values

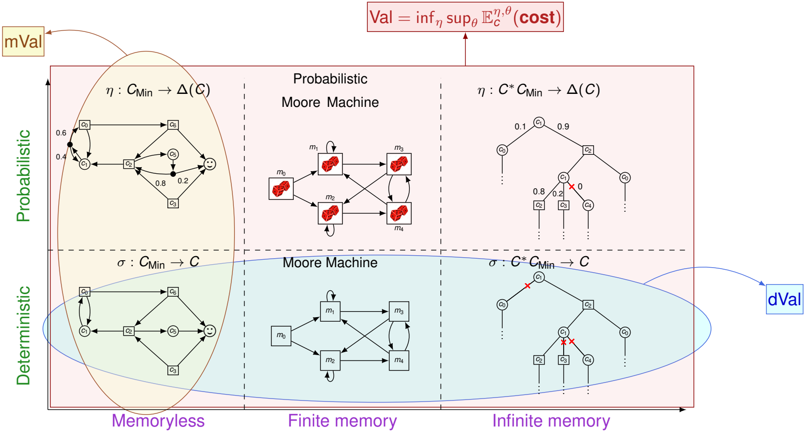


Stochastic values



Theorem (CONCUR'20, ICALP'21): Trading memory with probabilities

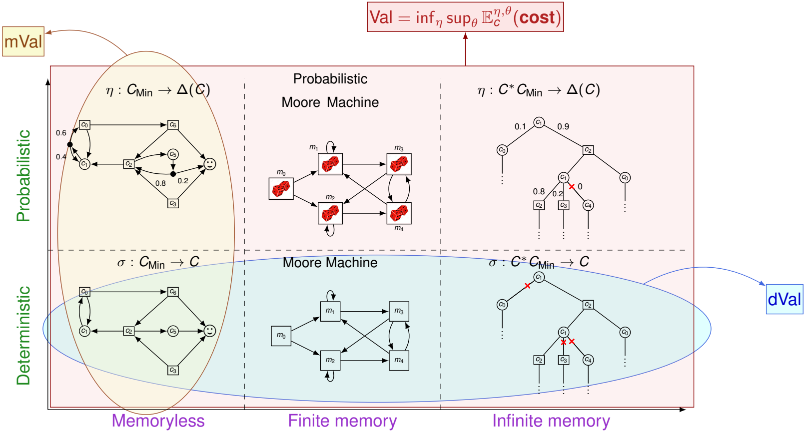
Stochastic values



Theorem (CONCUR'20, ICALP'21): Trading memory with probabilities

$$\text{dVal} = \text{Val} = \text{mVal}$$

Stochastic values

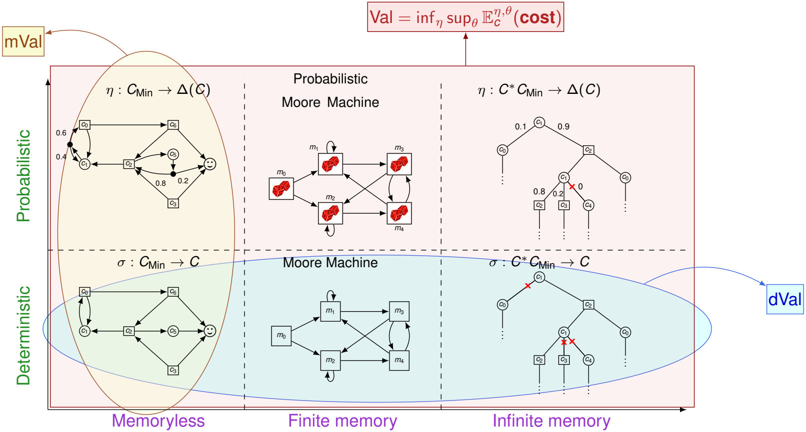


Theorem (CONCUR'20, ICALP'21): Trading memory with probabilities

$$\text{dVal} = \text{Val} = \text{mVal}$$

- 0-clock weighted timed games

Stochastic values

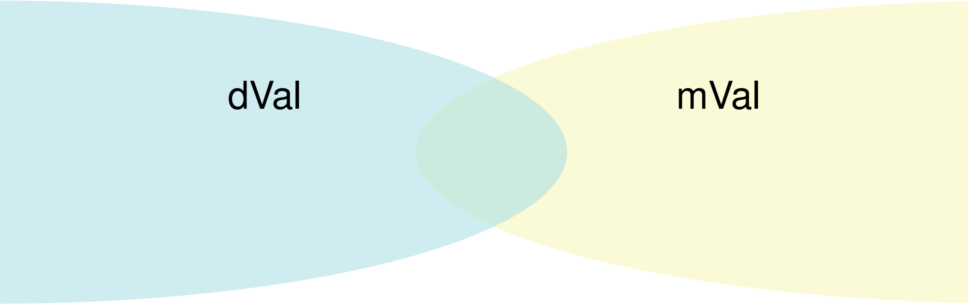


Theorem (CONCUR'20, ICALP'21): Trading memory with probabilities

$$\mathsf{dVal} \quad = \quad \mathsf{Val} \quad = \quad \mathsf{mVal}$$

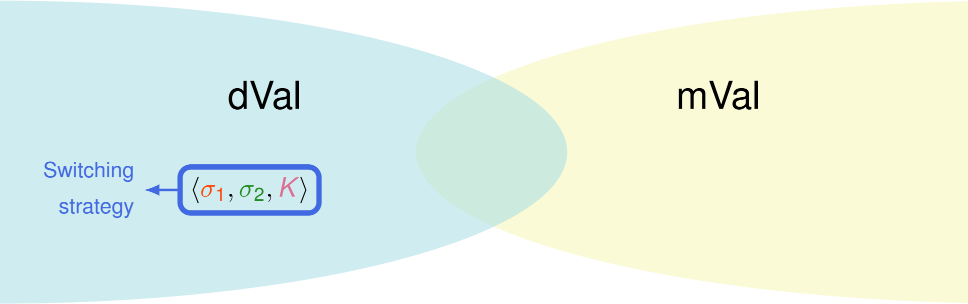
- 0-clock weighted timed games
- divergent weighted timed games

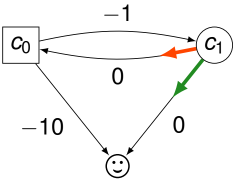
Trading memory with probabilities



Trading memory with probabilities

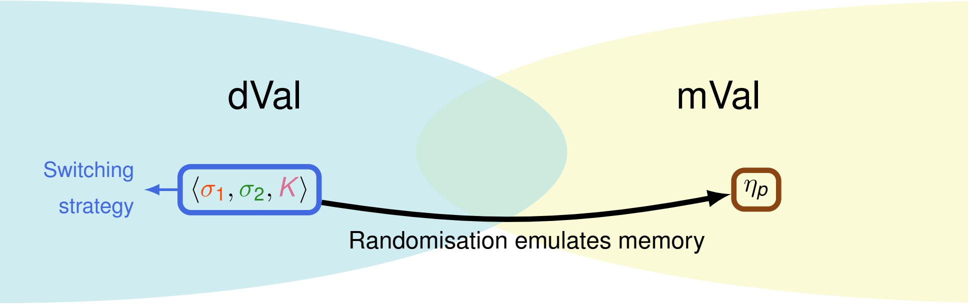
○ Min □ Max

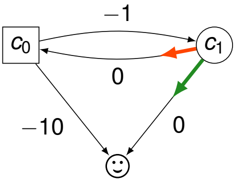




Trading memory with probabilities

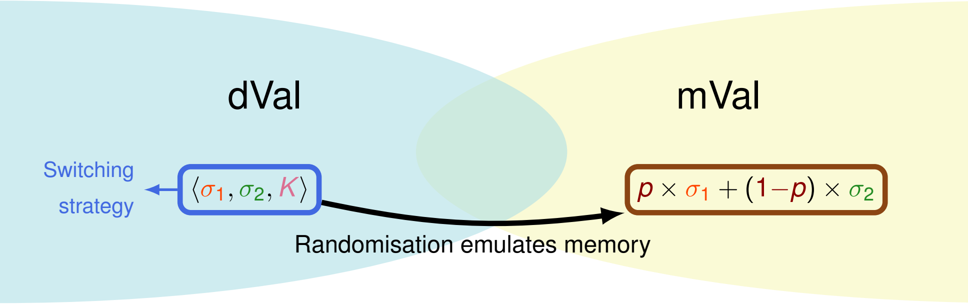
Min Max

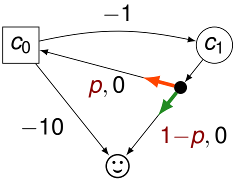




Trading memory with probabilities

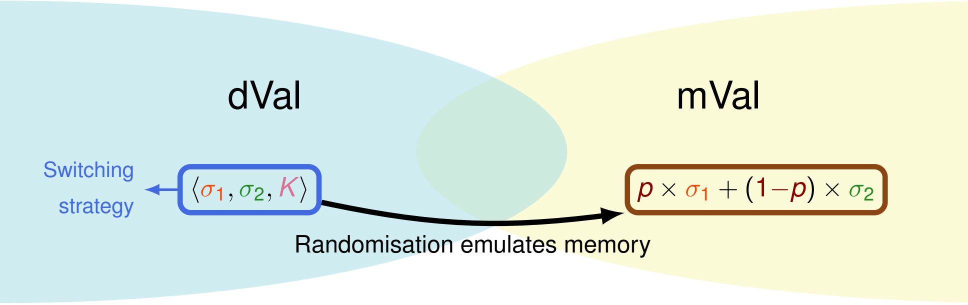
○ Min □ Max

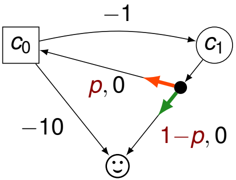




Trading memory with probabilities

○ Min □ Max

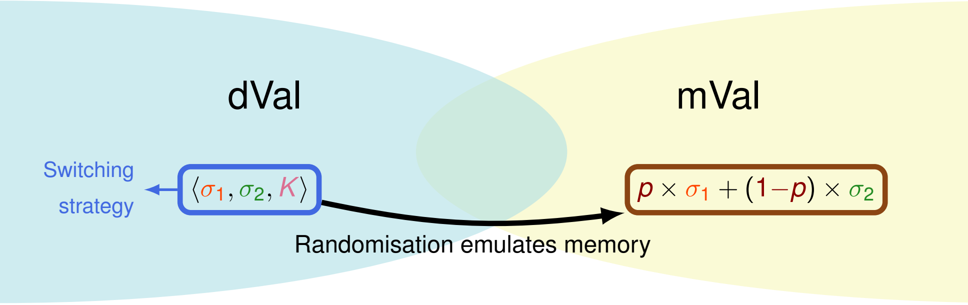


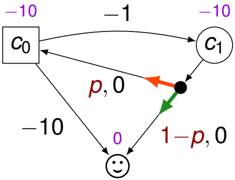


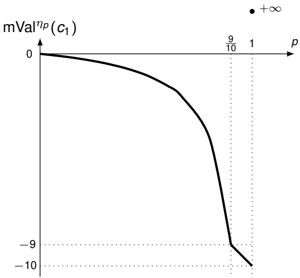
- Max has a best response deterministic memoryless strategy: τ

Trading memory with probabilities

○ Min □ Max



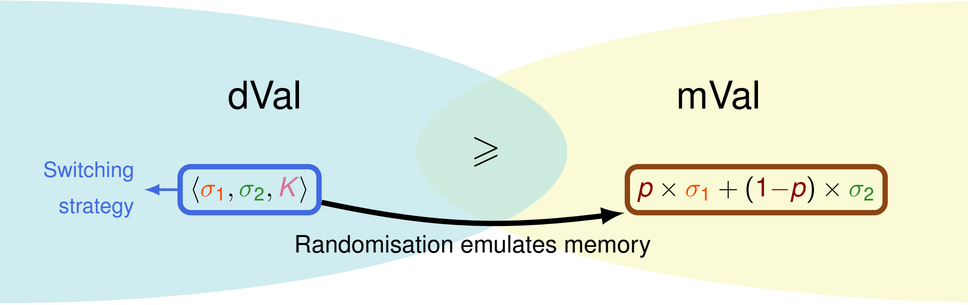


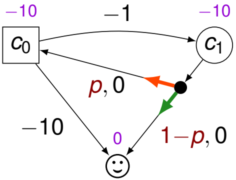


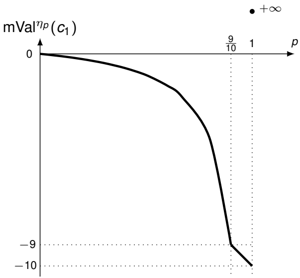
- Max has a best response deterministic memoryless strategy: τ

Trading memory with probabilities

○ Min □ Max







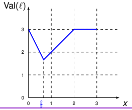
- Max has a best response deterministic memoryless strategy: τ

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Trading memory with probabilities

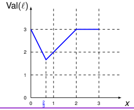
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Trading memory with probabilities

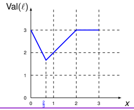
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities

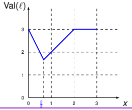
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities

Probabilities are useless in 1-clock WTG, divergent WTG, and 0-clock WTG

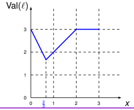
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities

Probabilities are useless in 1-clock WTG, divergent WTG, and 0-clock WTG

Robust optimal strategies

Robustness in weighted timed games

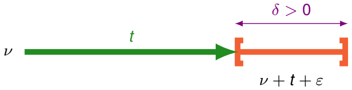
$$\nu \xrightarrow{t} \nu + t$$

Robustness in weighted timed games

Give to Max the power to perturb the delay chosen by Min

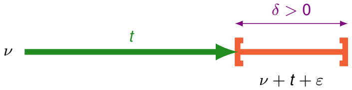
Robustness in weighted timed games

Give to Max the power to perturb the delay chosen by Min



Robustness in weighted timed games

Give to Max the power to perturb the delay chosen by Min

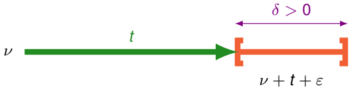


Robust semantics

Check the guard after the perturbation:

Robustness in weighted timed games

Give to Max the power to perturb the delay chosen by Min

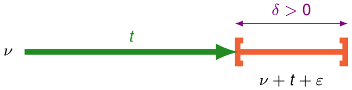


Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Robustness in weighted timed games

Give to Max the power to perturb the delay chosen by Min



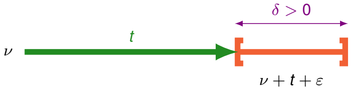
Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

Robustness in weighted timed games

Give to Max the power to perturb the delay chosen by Min



Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

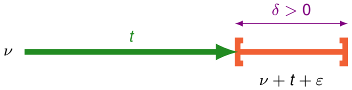
- δ is fixed and known

Robustness in weighted timed games

χ Min

ζ Max

Give to Max the power to perturb the delay chosen by Min



Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

- δ is fixed and known

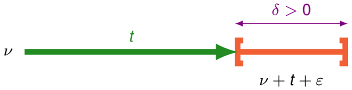
$$\mathrm{rVal}^{\delta}(c) = \inf_{\substack{\chi \\ \delta\text{-robust}}} \sup_{\substack{\zeta \\ \delta\text{-robust}}} \mathbf{cost}(\mathrm{Play}(c, \chi, \zeta))$$

Robustness in weighted timed games

χ Min

ζ Max

Give to Max the power to perturb the delay chosen by Min



Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

- δ is fixed and known

$$\mathrm{rVal}^{\delta}(c) = \inf_{\substack{\chi \\ \delta\text{-robust}}} \sup_{\substack{\zeta \\ \delta\text{-robust}}} \mathbf{cost}(\mathrm{Play}(c, \chi, \zeta))$$

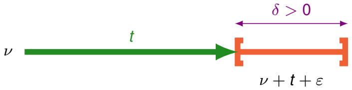
✓ Encoding fixed-δ semantics into exact one

Robustness in weighted timed games

χ Min

ζ Max

Give to Max the power to perturb the delay chosen by Min



Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

- δ is fixed and known

$$\mathrm{rVal}^{\delta}(c) = \inf_{\substack{\chi \\ \delta\text{-robust}}} \sup_{\substack{\zeta \\ \delta\text{-robust}}} \mathbf{cost}(\mathrm{Play}(c, \chi, \zeta))$$

✓ Encoding fixed-δ semantics into exact one

⚠ Need a new clock

Robustness in weighted timed games

χ Min

ζ Max

Give to Max the power to perturb the delay chosen by Min

Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

- δ is fixed and known
- δ tends to 0

$$\mathrm{rVal}^{\delta}(c) = \inf_{\substack{\chi \\ \delta\text{-robust}}} \sup_{\substack{\zeta \\ \delta\text{-robust}}} \mathbf{cost}(\mathrm{Play}(c, \chi, \zeta))$$

✔ Encoding fixed-δ semantics into exact one

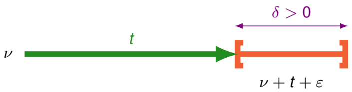
⚠ Need a new clock

Robustness in weighted timed games

χ Min

ζ Max

Give to Max the power to perturb the delay chosen by Min



Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

- δ is fixed and known

$$\mathrm{rVal}^{\delta}(c) = \inf_{\substack{\chi \\ \delta\text{-robust}}} \sup_{\substack{\zeta \\ \delta\text{-robust}}} \mathbf{cost}(\mathrm{Play}(c, \chi, \zeta))$$

- δ tends to 0

$$\mathrm{rVal}(c) = \lim_{\substack{\delta \to 0 \\ \delta > 0}} \mathrm{rVal}^{\delta}(c)$$

✓ Encoding fixed-δ semantics into exact one

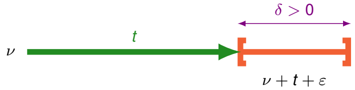
⚠ Need a new clock

Robustness in weighted timed games

χ Min

ζ Max

Give to Max the power to perturb the delay chosen by Min



Robust semantics

Check the guard after the perturbation: $\forall \varepsilon \in [0, \delta], \nu + t + \varepsilon$ satisfies the guard

Two problems induced by our knowledge on δ

- δ is fixed and known

$$\mathsf{rVal}^\delta(c) = \inf_{\substack{\chi \\ \delta\text{-robust}}} \sup_{\substack{\zeta \\ \delta\text{-robust}}} \mathbf{cost}(\mathsf{Play}(c, \chi, \zeta))$$

Encoding fixed-δ semantics into exact one

Need a new clock

- δ tends to 0

$$\mathsf{rVal}(c) = \lim_{\substack{\delta \to 0 \\ \delta > 0}} \mathsf{rVal}^\delta(c)$$

rVal^δ is monotonic in δ

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

	WTG			
rVal^{δ}	undecidable			
rVal	undecidable			

Robust Weighted Timed Automata and Games, P. Bouyer, N. Markey, and O. Sankur, 2013, FORMATS

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable			
rVal	undecidable			

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable			decidable (in $\mathbb{N}$)
rVal	undecidable			

Revisiting Robustness in Priced Timed Game, S. Guha, S. Krishna, L. Manasa, and A. Trivedi, 2015, FSTTCS

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable			

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

Robust value problems

Deciding if $\mathrm{rVal}^\delta(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

	WTG	acyclic	divergent	1-clock
rVal^δ	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

Robust value problems

Deciding if $\mathsf{rVal}^{\delta}(c)$ (resp. $\mathsf{rVal}(c)$) is at most equal to λ?

	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

Cells

Optimal reachability for weighted timed games, R. Alur, M. Bernadsky and P. Madhusudan, 2004, ICALP

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

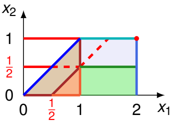
	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

Cells

$y = \sum_i a_i\, x_i + b$



Optimal reachability for weighted timed games, R. Alur, M. Bernadsky and P. Madhusudan, 2004, ICALP

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

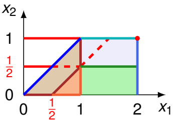
	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

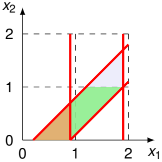
A combination of two existing methods

Cells

$y = \sum_i a_i \, x_i + b$



Shrunk DBM



Shrinking timed automata, O. Sankur, P. Bouyer, and N. Markey, 2011, FSTTCS

Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

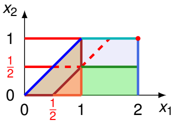
	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

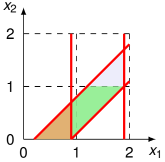
Cells

$y = \sum_i a_i x_i + b$



Shrunk cells

Shrunk DBM



Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

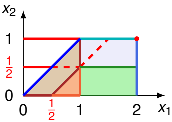
	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

Cells

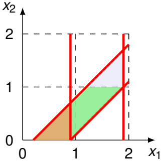
$y = \sum_i a_i \, x_i + b$



Shrunk cells

$y = \sum_i a_i \, x_i + b$

Shrunk DBM



Robust value problems

Deciding if $\mathrm{rVal}^{\delta}(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

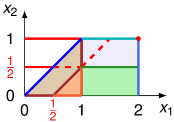
	WTG	acyclic	divergent	1-clock
rVal^{δ}	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable		

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

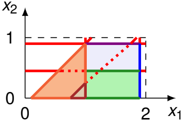
Cells

$y = \sum_i a_i \, x_i + b$

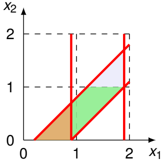


Shrunk cells

$y = \sum_i a_i \, x_i + b + c\,\delta$



Shrunk DBM



Robust value problems

Deciding if $\mathrm{rVal}^\delta(c)$ (resp. $\mathrm{rVal}(c)$) is at most equal to λ?

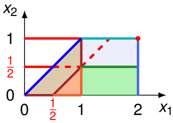
	WTG	acyclic	divergent	1-clock
rVal^δ	undecidable	decidable	decidable	decidable (in $\mathbb{N}$)
rVal	undecidable	decidable	decidable	

Theorem (SUBMITTED): Decidability of the robust value problem in acyclic WTG

A combination of two existing methods

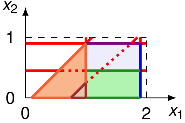
Cells

$y = \sum_i a_i\, x_i + b$

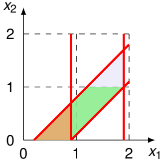


Shrunk cells

$y = \sum_i a_i\, x_i + b + c\,\delta$



Shrunk DBM

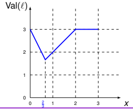


Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities

Probabilities are useless in 1-clock WTG, divergent WTG, and 0-clock WTG

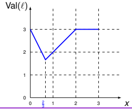
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



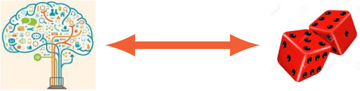
Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities



Probabilities are useless in 1-clock WTG, divergent WTG, and 0-clock WTG

Definition of robust values

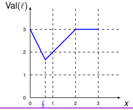
Robust optimal strategies

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



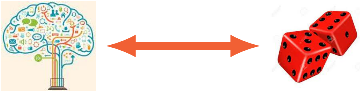
Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities



Probabilities are useless in 1-clock WTG, divergent WTG, and 0-clock WTG

Definition of robust values

Robust optimal strategies

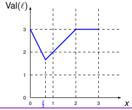
Decidability of rVal(c) $< +\infty$ in all WTGs

Problems on weighted timed games

Fixpoint characterisation

Switching strategies in divergent WTG

Deterministic value problem



Decidability for 1-clock WTG

Software prototype for 1-clock WTG

Definition of stochastic values

Memory is useless in divergent WTG and 0-clock WTG

Trading memory with probabilities

Probabilities are useless in 1-clock WTG, divergent WTG, and 0-clock WTG

Definition of robust values

Computing robust values in divergent (and acyclic) WTG

Robust optimal strategies

Decidability of rVal(c) $< +\infty$ in all WTGs

Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

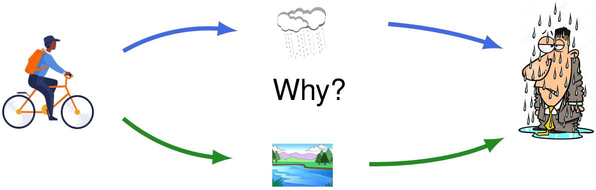
Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Why?

Counterfactual causality

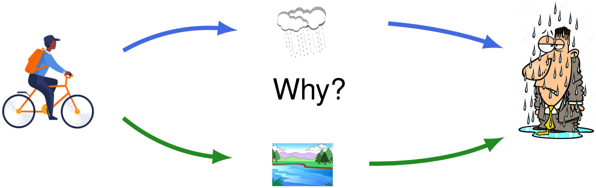
Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)



Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Why the specification does not hold in the counterexample?



Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Why the specification does not hold in the counterexample?

Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Why the specification does not hold in the counterexample?

Counterfactual causality

¬Cause (in the system) implies ¬Effect (in *closed* execution)

Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Why the specification does not hold in the counterexample?

Counterfactual causality

¬Cause (in the system) implies ¬Effect (in *closed* execution)

Definition (GandALF'23): Definition of counterfactual causes in transitions systems and games

Counterfactual causality

Joint work with Christel Baier and Jakob Piribauer at Dresden (Germany)

Why the specification does not hold in the counterexample?

Counterfactual causality

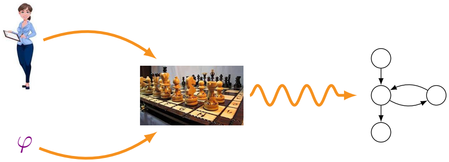
¬Cause (in the system) implies ¬Effect (in *closed* execution)

Definition (GandALF'23): Definition of counterfactual causes in transitions systems and games

Using distance over executions (strategies) to define *close*

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)



Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

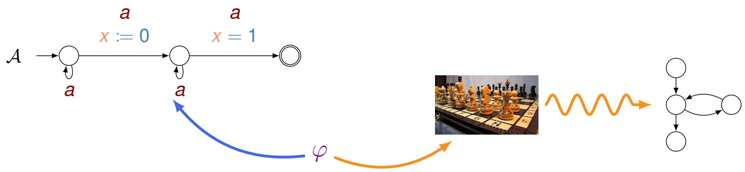
Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)



Timed Church synthesis

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

$A \times \mathbb{Q}_{\geqslant 0}$

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

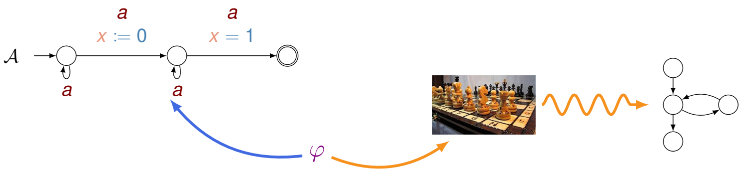
Timed Church synthesis

$A \times \mathbb{Q}_{\geqslant 0}$

B

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)



Timed Church synthesis

(a_1, t_1)

$A \times \mathbb{Q}_{\geqslant 0}$

B

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

(a_1, t_1)

$A \times \mathbb{Q}_{\geqslant 0}$

b_1

B

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

(a_1, t_1) (a_2, t_2)

$A \times \mathbb{Q}_{\geqslant 0}$

b_1

B

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

(a_1, t_1) (a_2, t_2)

$A \times \mathbb{Q}_{\geqslant 0}$

B

b_1 b_2

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

$A \times \mathbb{Q}_{\geqslant 0}$ (a_1, t_1) (a_2, t_2) $\cdots$

B b_1 b_2 $\cdots$

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

(a_1, t_1) (a_2, t_2) $\cdots$

$A \times \mathbb{Q}_{\geqslant 0}$

B

b_1 b_2 $\cdots$

Existence winning strategy		

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

(a_1, t_1) (a_2, t_2) $\cdots$

$A \times \mathbb{Q}_{\geqslant 0}$

B

b_1 b_2 $\cdots$

Existence winning strategy	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$	

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

$(a_1, t_1) \quad (a_2, t_2) \quad \cdots$

$A \times \mathbb{Q}_{\geqslant 0}$

B $\quad b_1 \quad b_2 \quad \cdots$

Existence winning strategy	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

$A \times \mathbb{Q}_{\geqslant 0}$ (a_1, t_1) (a_2, t_2) $\cdots$

B b_1 b_2 $\cdots$

Existence winning strategy	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$
		
		

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)

Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

(a_1, t_1) (a_2, t_2) $\cdots$

$A \times \mathbb{Q}_{\geqslant 0}$

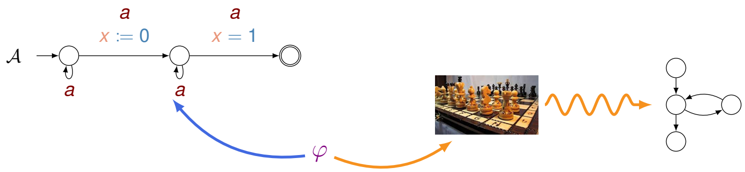
B b_1 b_2 $\cdots$

Existence winning strategy	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$
		
	Undecidable	

Timed Games and Deterministic Separability, L. Clemente, S. Lasota, and R. Piórkowski, 2020, ICALP

Timed Church synthesis (work on progress)

Joint work with Sławomir Lasota at Warsaw (Poland)



Timed Church synthesis

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

(a_1, t_1) (a_2, t_2) $\cdots$

$A \times \mathbb{Q}_{\geqslant 0}$

B

b_1 b_2 $\cdots$

Existence winning strategy	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$	 wins $\Leftrightarrow$ $w \in \mathcal{L}(\mathcal{A})$
	Undecidable	Undecidable
	Undecidable	Undecidable

How to synthesis a real-time system usable in the real word?

How to synthesis a real-time system usable in the real word?

Memory from the specification

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

Robustness

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

Robustness

Synthesis of robust systems

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

Robustness

Synthesis of robust systems

- Parametric timed automata

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

Robustness

Synthesis of robust systems

- Parametric timed automata
- Stochastic robustness

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

Robustness

Synthesis of robust systems

- Parametric timed automata
- Stochastic robustness

Quantify the robustness of a system

How to synthesis a real-time system usable in the real word?

Memory from the specification

Decidable classes for timed Church synthesis

- Reduce expressiveness of winning condition
- Reduce power of one player

Determinacy

Deterministic separability for timed automata

Emulate the memory

Memory versus probabilities

- Characterisation of winning strategy without memory
- New algorithms to solve games

Memory versus clocks

Memory versus control

Robustness

Synthesis of robust systems

- Parametric timed automata
- Stochastic robustness

Quantify the robustness of a system

Thank you. Questions?