Simple Priced Timed Games are not That Simple

Thomas Brihaye (UMons), Gilles Geeraerts (ULB), Axel Haddad (UMons), Lefaucheux Engel (MPI-SWS), Benjamin Monmege (LIF)

October 15, 2019

Smart Houses on a Grid (Jadevej Case)

Eight houses
Electric local grid

Each house:

- Solar panels
- Electric heating
- Storage of energy

Smart Houses on a Grid (Jadevej Case)

Eight houses
Electric local grid

Each house:

- Solar panels
- Electric heating
- Storage of energy

Goal: for each house, optimize its behavior to reduce its energy bill

How to compute the expenses of a house?

Smart Houses on a Grid (Jadevej Case)

Eight houses
Electric local grid

Each house:

- Solar panels
- Electric heating
- Storage of energy

Goal: for each house, optimize its behavior to reduce its energy bill

How to compute the expenses of a house?

Solar panel ON
- Selling energy: +2€/t.u.
- Consumption: 0€/t.u.
- Storing energy: 0€/t.u.

Solar panel OFF
- Selling energy: +2€/t.u.
- Consumption: −2€/t.u.

Solar panel OFF
- Selling energy: +1€/t.u.
- Consumption: −1€/t.u.

+ fixed cost to start selling or buying energy

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Real-time requirements / environment $\Rightarrow$ real-time controller

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Real-time requirements / environment $\Rightarrow$ real-time controller

Among all *valid* controller, choose a *good/cheap/efficient* one

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Two-player game

Real-time requirements / environment $\Rightarrow$ real-time controller

Among all *valid* controller, choose a *good/cheap/efficient* one

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Two-player game

Real-time requirements / environment $\Rightarrow$ real-time controller

Two-player **timed** game

Among all *valid* controller, choose a *good/cheap/efficient* one

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Two-player game

Real-time requirements / environment $\Rightarrow$ real-time controller

Two-player **timed** game

Among all *valid* controller, choose a *good/cheap/efficient* one

Two-player **priced** timed game

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller}} \quad \models \quad \text{Spec}$$

Two-player game

Real-time requirements / environment $\Rightarrow$ real-time controller

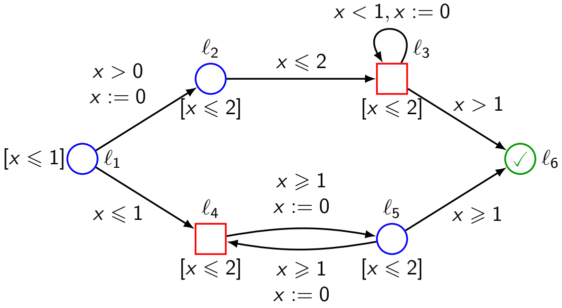
Two-player **timed** game

Among all *valid* controller, choose a *good/cheap/efficient* one

Two-player **priced** timed game

Production/consumption of resources $\Rightarrow$ Negative weights

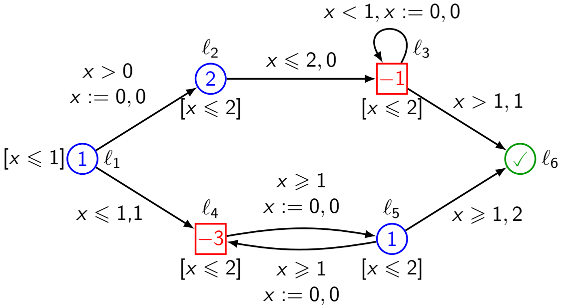
Priced Timed Games (PTG)



Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

Priced Timed Games (PTG)

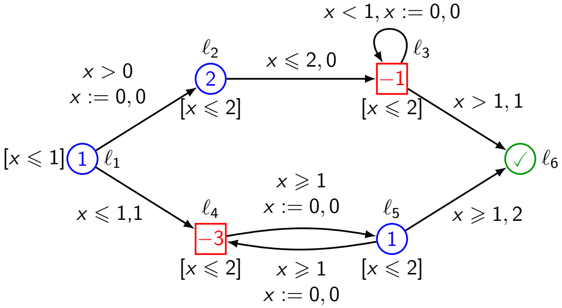


Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

$(\ell_1, 0)$

Priced Timed Games (PTG)

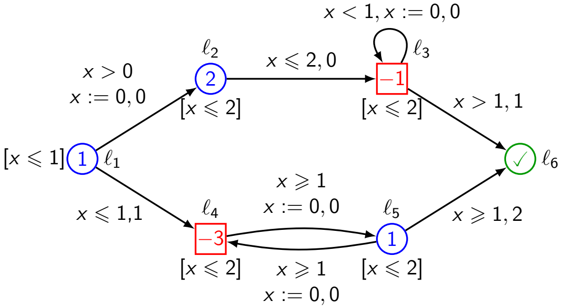


Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

$$(\ell_1, 0) \xrightarrow{0.4, \searrow} (\ell_4, 0.4)$$

Priced Timed Games (PTG)

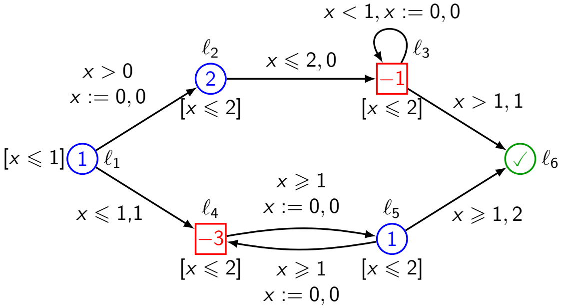


$$(\ell_1, 0) \xrightarrow{0.4, \searrow} (\ell_4, 0.4) \xrightarrow{0.6, \rightarrow} (\ell_5, 0)$$

Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

Priced Timed Games (PTG)

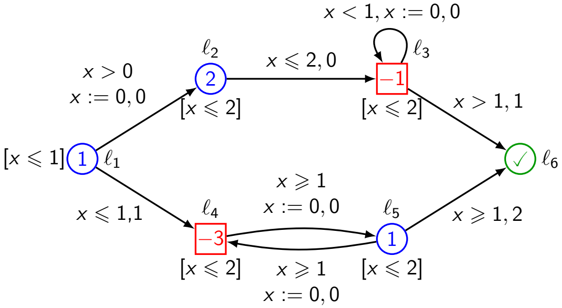


Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

$$(\ell_1, 0) \xrightarrow{0.4, \searrow} (\ell_4, 0.4) \xrightarrow{0.6, \rightarrow} (\ell_5, 0) \xrightarrow{1.5, \leftarrow} (\ell_4, 0) \xrightarrow{1.1, \rightarrow} (\ell_5, 0) \xrightarrow{2, \nearrow} (\checkmark, 2)$$

Priced Timed Games (PTG)



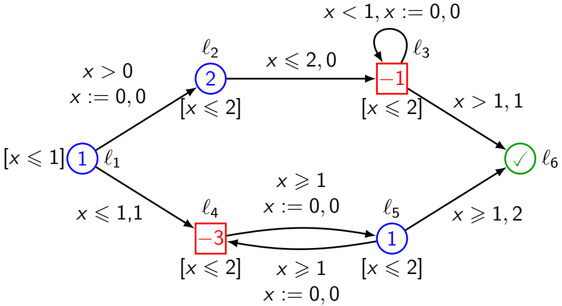
Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

$$(\ell_1, 0) \xrightarrow{0.4, \searrow} (\ell_4, 0.4) \xrightarrow{0.6, \rightarrow} (\ell_5, 0) \xrightarrow{1.5, \leftarrow} (\ell_4, 0) \xrightarrow{1.1, \rightarrow} (\ell_5, 0) \xrightarrow{2, \nearrow} (\checkmark, 2)$$

$$0.4 + 1 \quad -3 \times 0.6 \quad +1.5 \quad -3 \times 1.1 \quad +2 \times 2 + 2 \quad = 3.8$$

Priced Timed Games (PTG)



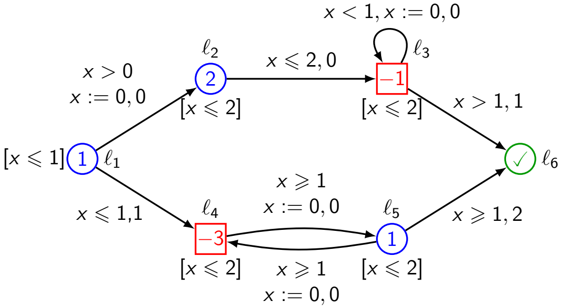
Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

Semantics in terms of
infinite game with weights

$$(\ell_1, 0) \xrightarrow{0.4, \searrow} (\ell_4, 0.4) \xrightarrow{0.6, \rightarrow} (\ell_5, 0) \xrightarrow{1.5, \leftarrow} (\ell_4, 0) \xrightarrow{1.1, \rightarrow} (\ell_5, 0) \xrightarrow{2, \nearrow} (\checkmark, 2)$$
$$0.4 + 1 \quad -3 \times 0.6 \quad +1.5 \quad -3 \times 1.1 \quad +2 \times 2 + 2 \quad = 3.8$$

$$(\ell_1, 0) \xrightarrow{0.2, \nearrow} (\ell_2, 0) \xrightarrow{0.7, \rightarrow} (\ell_3, 0.7) \xrightarrow{0.2, \circlearrowleft} (\ell_3, 0) \xrightarrow{0.9, \circlearrowleft} (\ell_3, 0) \quad \cdots$$
$$0.2 \quad +0.7 \quad -0.2 \quad -0.9 \quad \cdots \quad = +\infty \ (\checkmark \text{ not reached})$$

Priced Timed Games (PTG)



Timed Automaton
with partition of states
between 2 players
+ reachability objective
+ rates in locations
+ costs over transitions

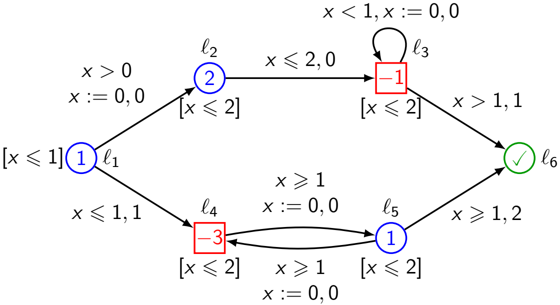
Semantics in terms of
infinite game with weights

$$(\ell_1,0)\xrightarrow[0.4+1]{0.4,\searrow}(\ell_4,0.4)\xrightarrow[-3\times 0.6]{0.6,\rightarrow}(\ell_5,0)\xrightarrow[+1.5]{1.5,\leftarrow}(\ell_4,0)\xrightarrow[-3\times 1.1]{1.1,\rightarrow}(\ell_5,0)\xrightarrow[+2\times 2+2]{2,\nearrow}(\checkmark,2) \quad = 3.8$$

$$(\ell_1,0)\xrightarrow[0.2]{0.2,\nearrow}(\ell_2,0)\xrightarrow[+0.7]{0.7,\rightarrow}(\ell_3,0.7)\xrightarrow[-0.2]{0.2,\circlearrowleft}(\ell_3,0)\xrightarrow[-0.9]{0.9,\circlearrowleft}(\ell_3,0)\ \cdots \quad = +\infty \ (\checkmark \text{ not reached})$$

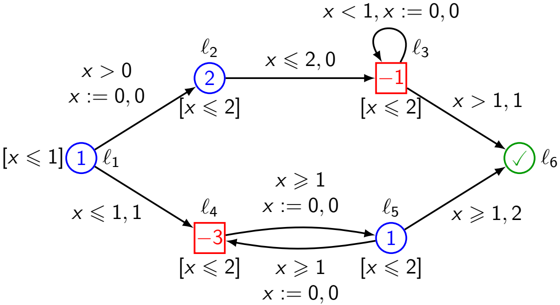
$$\text{Cost of a play: } \begin{cases} +\infty & \text{if } \checkmark \text{ not reached} \\ \text{total payoff up to } \checkmark & \text{otherwise} \end{cases}$$

Strategies and objectives



Strategy for each player: mapping of finite plays to a delay and an action

Strategies and objectives

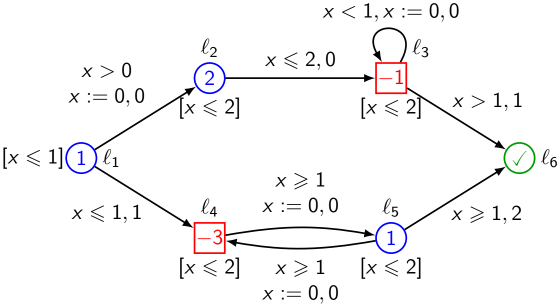


Strategy for each player: mapping of finite plays to a delay and an action

Goal of player ○: reach ✓ **and** minimize the cost

Goal of player □: avoid ✓ **or, if not possible,** maximize the cost

Strategies and objectives



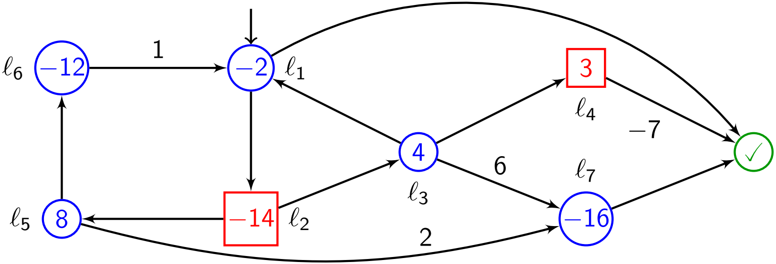
Strategy for each player: mapping of finite plays to a delay and an action

Goal of player ○: reach ✓ **and** minimize the cost
Goal of player □: avoid ✓ **or, if not possible,** maximize the cost

Main object of interest:
$\overline{\mathsf{Val}}(\ell, \nu)$ = minimal cost player ○ can guarantee
$\underline{\mathsf{Val}}(\ell, \nu)$ = maximal cost player □ can guarantee
What can players guarantee as a payoff? And design *good* strategies.

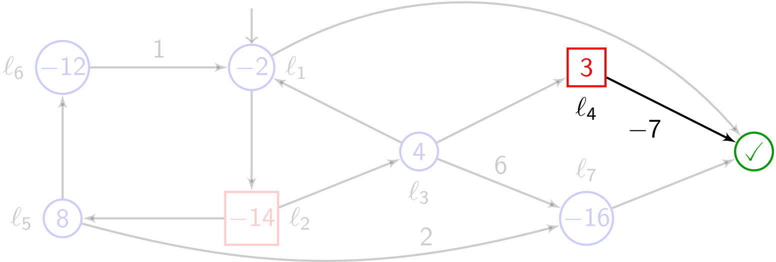
Simple Priced Timed Games

Simple Priced Timed Game (SPTG): One-clock PTG with no guards or resets and one global invariant bounding the clock by 1.



Simple Priced Timed Games

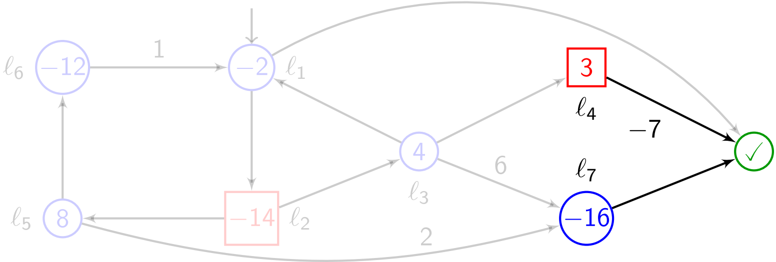
Simple Priced Timed Game (SPTG): One-clock PTG with no guards or resets and one global invariant bounding the clock by 1.



$$\overline{\mathsf{Val}}(\ell_4, \nu) = 3(1-\nu) - 7 = -3\nu - 4$$

Simple Priced Timed Games

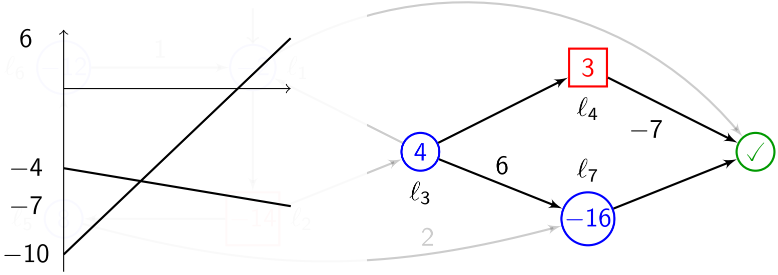
Simple Priced Timed Game (SPTG): One-clock PTG with no guards or resets and one global invariant bounding the clock by 1.



$$\overline{\mathsf{Val}}(\ell_4, \nu) = -3\nu - 4, \qquad \overline{\mathsf{Val}}(\ell_7, \nu) = -16(1 - \nu)$$

Simple Priced Timed Games

Simple Priced Timed Game (SPTG): One-clock PTG with no guards or resets and one global invariant bounding the clock by 1.

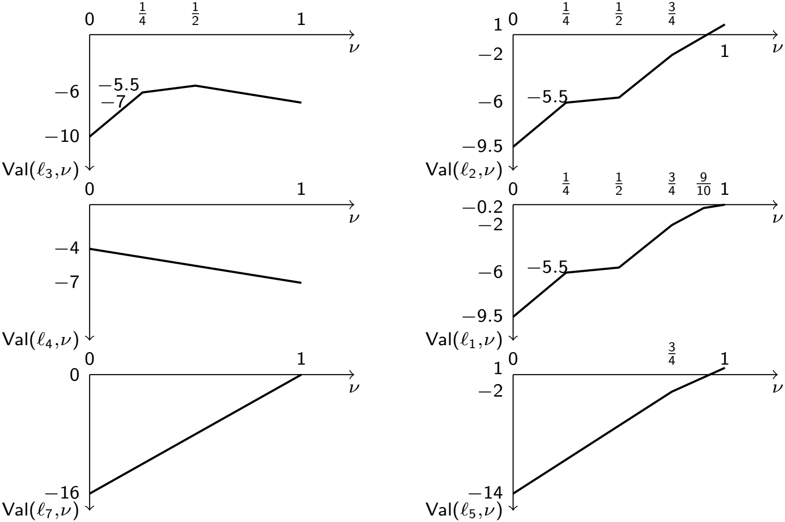


$$\overline{\mathsf{Val}}(\ell_4, \nu) = -3\nu - 4, \qquad \overline{\mathsf{Val}}(\ell_7, \nu) = -16(1-\nu),$$

$$\overline{\mathsf{Val}}(\ell_3, \nu) = \inf_{0 \leqslant t \leqslant 1-\nu}[4t + \min(-3(\nu+t) - 4, 6 - 16(1-(\nu+t)))] = \min(-3\nu - 4, 16\nu - 10)$$

Simple Priced Timed Games

Simple Priced Timed Game (SPTG): One-clock PTG with no guards or resets and one global invariant bounding the clock by 1.



State of the art

$\mathsf{F}_{\leqslant K}$✓: Decide whether $\overline{\mathsf{Val}}(\ell, \nu) \leqslant K$?

State of the art

$\mathsf{F}_{\leqslant K}$✓: Decide whether $\overline{\mathsf{Val}}(\ell, \nu) \leqslant K$?

- One-player case (**Priced timed automata**): optimal reachability problem is PSPACE-complete
  - Algorithm based on regions [Bouyer, Brihaye, Bruyère, and Raskin, 2007];
  - and hardness shown for timed automata with at least 2 clocks [Fearnley and Jurdziński, 2013, Haase, Ouaknine, and Worrell, 2012]
- 2-player PTGs: undecidable [Brihaye, Bruyère, and Raskin, 2005, Bouyer, Brihaye, and Markey, 2006a], even with only non-negative weights and 3 clocks
- PTGs with non-negative weights and strictly non-Zeno cost cycles or with one clock: exponential algorithm [Bouyer, Cassez, Fleury, and Larsen, 2004, Alur, Bernadsky, and Madhusudan, 2004, Bouyer, Larsen, Markey, and Rasmussen, 2006b, Rutkowski, 2011, Hansen, Ibsen-Jensen, and Miltersen, 2013]

This talk: PTGs with one-clock

Solving PTGs with non-negative weights

[Bouyer, Larsen, Markey, and Rasmussen, 2006b, Rutkowski, 2011]: iterative elimination of locations

Solving PTGs with non-negative weights

[Bouyer, Larsen, Markey, and Rasmussen, 2006b, Rutkowski, 2011]: iterative elimination of locations

- precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Removing resets
  - Bounding clock by 1
  - Removing guards/invariants

Solving PTGs with non-negative weights

[Bouyer, Larsen, Markey, and Rasmussen, 2006b, Rutkowski, 2011]: iterative elimination of locations

- precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Removing resets
  - Bounding clock by 1
  - Removing guards/invariants

- for SPTGs: compute value functions $\overline{\mathsf{Val}}(\ell, \nu)$.

Details on the precomputation

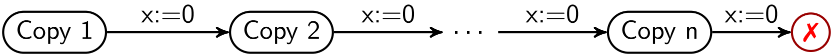
- Removing resets

Details on the precomputation

- Removing resets
  - Bound on the number of useful resets
  - Build and study copies of the PTG

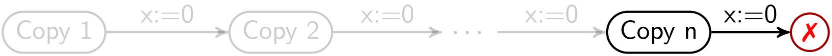
Details on the precomputation

- Removing resets
  - Bound on the number of useful resets
  - Build and study copies of the PTG



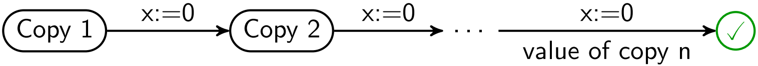
Details on the precomputation

- Removing resets
  - Bound on the number of useful resets
  - Build and study copies of the PTG



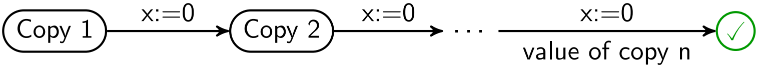
Details on the precomputation

- Removing resets
  - Bound on the number of useful resets
  - Build and study copies of the PTG



Details on the precomputation

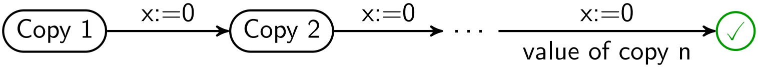
- Removing resets
  - Bound on the number of useful resets
  - Build and study copies of the PTG



- Bounding clock by 1 and removing guards/invariants

Details on the precomputation

- Removing resets
  - Bound on the number of useful resets
  - Build and study copies of the PTG



- Bounding clock by 1 and removing guards/invariants
  - Maximal meaningful value of the clock
  - Build a copy of the PTG for each time unit below this maximum
  - Study each PTG successively

Solving SPTGs: Recursive elimination of states

- Player ○ prefers to stay as long as possible in locations with **minimal price**: add a final location allowing him to stay until the end, and make the location urgent

Solving SPTGs: Recursive elimination of states

- Player ○ prefers to stay as long as possible in locations with **minimal price**: add a final location allowing him to stay until the end, and make the location urgent

- Player □ prefers to leave as soon as possible in locations with **minimal price**: make the location urgent

Solving SPTGs: Recursive elimination of states

- Player ○ prefers to stay as long as possible in locations with **minimal price**: add a final location allowing him to stay until the end, and make the location urgent
- Player □ prefers to leave as soon as possible in locations with **minimal price**: make the location urgent

Problem: intuition not always true... you may have to change decision!

Solving SPTGs: Recursive elimination of states

- Player ○ prefers to stay as long as possible in locations with **minimal price**: add a final location allowing him to stay until the end, and make the location urgent

- Player □ prefers to leave as soon as possible in locations with **minimal price**: make the location urgent

Problem: intuition not always true... you may have to change decision!

Exponential recursive algorithm + construction of the value functions from right ($x = 1$) to left ($x = 0$)

Solving SPTGs: Recursive elimination of states

- Player ○ prefers to stay as long as possible in locations with **minimal price**: add a final location allowing him to stay until the end, and make the location urgent

- Player □ prefers to leave as soon as possible in locations with **minimal price**: make the location urgent

Problem: intuition not always true... you may have to change decision!

Exponential recursive algorithm + construction of the value functions from right ($x = 1$) to left ($x = 0$)

Shape of the value functions: continuous, non-increasing, piecewise affine functions with at most exponential number of cutpoints.

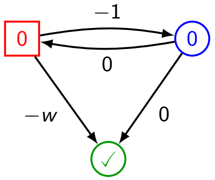
More complex when negative costs

- Value $-\infty$: detection is as hard as mean-payoff. No hope for complexity better than $\mathbf{NP} \cap \mathbf{co\text{-}NP}$, or pseudo-polynomial

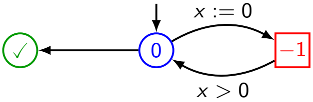
More complex when negative costs

- Value $-\infty$: detection is as hard as mean-payoff. No hope for complexity better than **NP** $\cap$ **co-NP**, or pseudo-polynomial

- Memory complexity
  - Player $\bigcirc$ needs memory, even in the untimed setting



  - Player $\square$ may require infinite memory



Known results with negative costs [Brihaye, Geeraerts, Krishna, Manasa, Monmege, and Trivedi, 2014]

- $\mathsf{F}_{\leqslant K}$ ✓ undecidable for 2 or more clocks

Proof by reduction of 2-counter machines.

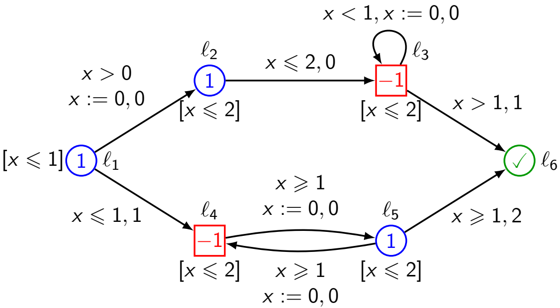
Known results with negative costs [Brihaye, Geeraerts, Krishna, Manasa, Monmege, and Trivedi, 2014]

- $\mathsf{F}_{\leqslant K}\checkmark$ undecidable for 2 or more clocks

Proof by reduction of 2-counter machines.

- Pseudo-polynomial algorithm for One-clock Bi-valued PTG

Assumption: rates of locations $\{p^-, p^+\}$ included in $\{0, +d, -d\}$ ($d \in \mathbf{N}$) (no assumption on costs of transitions)



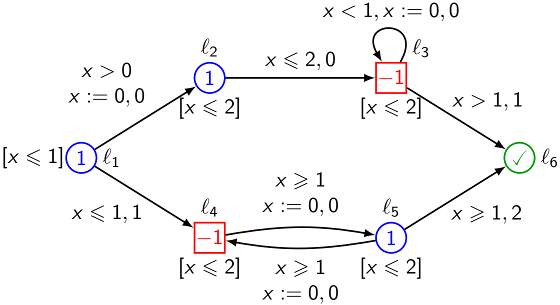
Known results with negative costs [Brihaye, Geeraerts, Krishna, Manasa, Monmege, and Trivedi, 2014]

- $\mathsf{F}_{\leqslant K}\checkmark$ undecidable for 2 or more clocks

Proof by reduction of 2-counter machines.

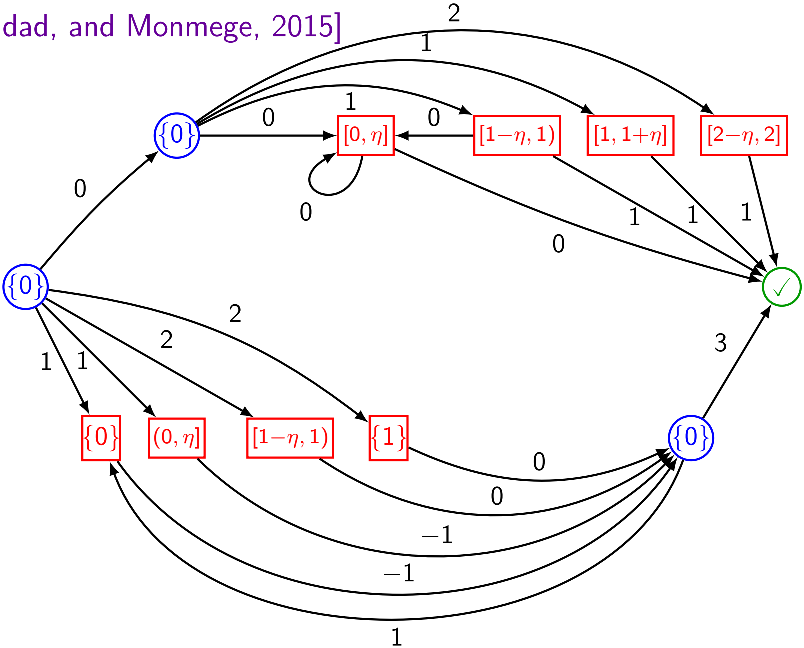
- Pseudo-polynomial algorithm for One-clock Bi-valued PTG

Assumption: rates of locations $\{p^-, p^+\}$ **included in** $\{0, +d, -d\}$ **($d \in \mathbf{N}$)** (no assumption on costs of transitions)

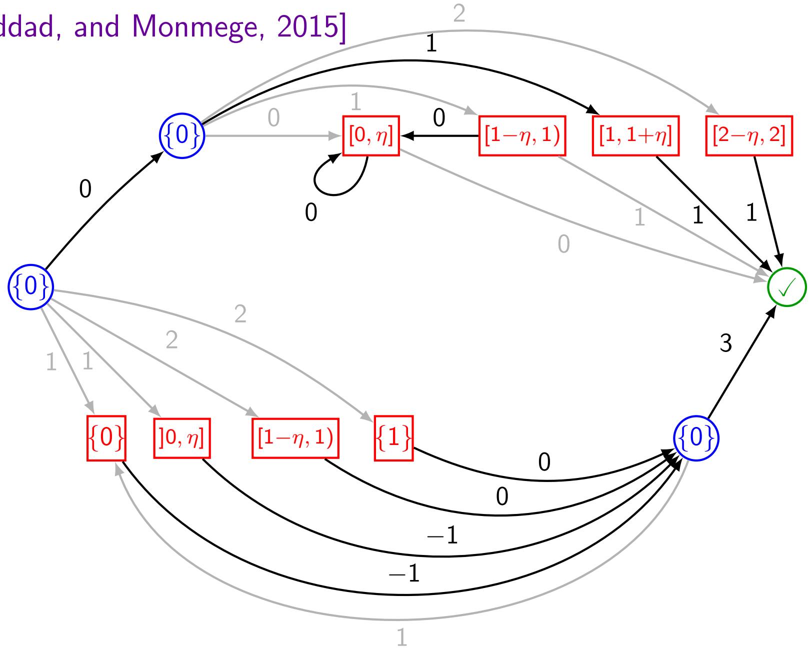


Method: Corner point abstraction.

Solving min-cost reachability games [Brihaye, Geeraerts, Haddad, and Monmege, 2015]



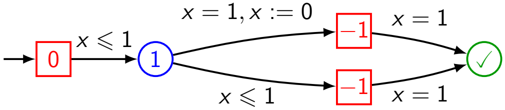
Solving min-cost reachability games [Brihaye, Geeraerts, Haddad, and Monmege, 2015]



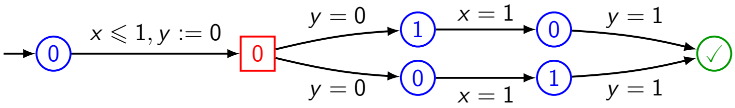
1BPTG: maximal fragment for corner-point abstraction

Players may need to play far from corners...

- With 3 weights in $\{-1, 0, +1\}$: play at $1/2$...



- With 2 weights in $\{-1, 0, +1\}$ but 2 clocks: value $1/2$...



Solving PTGs with arbitrary weights

- Precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Bounding clock by 1, removing guards/invariants
  - Removing resets
- For SPTGs: compute value functions $\overline{\mathsf{Val}}(\ell, x)$.

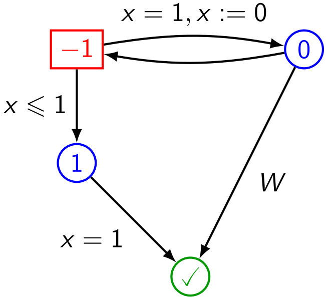
Solving PTGs with arbitrary weights

- Precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Bounding clock by 1, removing guards/invariants
    $\longrightarrow$ Previously used techniques work!
  - Removing resets
- For SPTGs: compute value functions $\overline{\mathsf{Val}}(\ell, x)$.

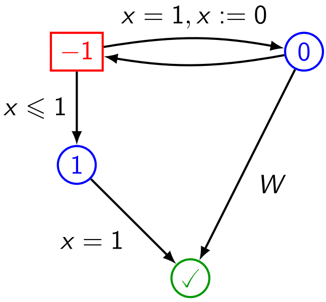
Solving PTGs with arbitrary weights

- Precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Bounding clock by 1, removing guards/invariants
    $\longrightarrow$ Previously used techniques work!
  - Removing resets
    $\longrightarrow$ Previously, bound the number of resets...
- For SPTGs: compute value functions $\overline{\mathsf{Val}}(\ell, x)$.

Bounding the number of resets needed is not possible

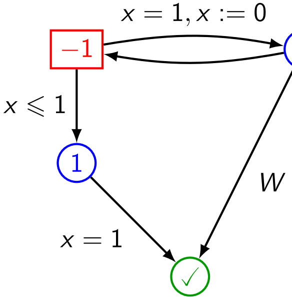


Bounding the number of resets needed is not possible



Player $\bigcirc$ can guarantee (i.e., ensure to be below) value ε for all $\varepsilon > 0$...

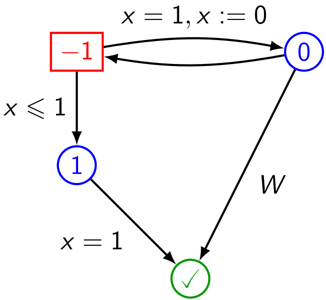
Bounding the number of resets needed is not possible



Player ○ can guarantee (i.e., ensure to be below) value ε for all $\varepsilon > 0$...

... but cannot obtain 0: hence, no optimal strategy...

Bounding the number of resets needed is not possible



Player ○ can guarantee (i.e., ensure to be below) value ε for all $\varepsilon > 0$...

... but cannot obtain 0: hence, no optimal strategy...

... moreover, to obtain ε, ○ needs to loop at least $W + \lceil 1/\log \varepsilon \rceil$ times before reaching ✓!

Solving PTGs with arbitrary weights

- precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Bounding clock by 1, removing guards/invariants
    $\longrightarrow$ Previously used techniques work!
  - Removing resets
    $\longrightarrow$ Previously, bound the number of resets...
- for SPTGs: compute value functions $\overline{\mathsf{Val}}(\ell, x)$.

Solving PTGs with arbitrary weights

- precomputation: polynomial-time cascade of simplification of PTGs into SPTGs
  - Bounding clock by 1, removing guards/invariants
    $\longrightarrow$ Previously used techniques work!
  - Removing resets
    $\longrightarrow$ Previously, bound the number of resets...
- for SPTGs: compute value functions $\overline{\mathsf{Val}}(\ell, x)$.

Challenges with arbitrary weights:

- Proof of correctness does not generalise: initially two distinct proofs for $\bigcirc$ and $\square$
- Proof of termination does not generalise: difficult because of the double recursion...

Make a symmetric treatment of ○ and □

Theorem

PTGs are determined ($\overline{\mathsf{Val}} = \underline{\mathsf{Val}} = \mathsf{Val}$), and value functions are continuous (over regions).

Determinacy follows from Gale-Stewart determinacy result.

Make a symmetric treatment of $\bigcirc$ and $\square$

Theorem

PTGs are determined ($\overline{\mathsf{Val}} = \underline{\mathsf{Val}} = \mathsf{Val}$), and value functions are continuous (over regions).

Determinacy follows from Gale-Stewart determinacy result.

Advantage: both players are dual...

Make a symmetric treatment of $\bigcirc$ and $\square$

Theorem

PTGs are determined ($\overline{\mathsf{Val}} = \underline{\mathsf{Val}} = \mathsf{Val}$), and value functions are continuous (over regions).

Determinacy follows from Gale-Stewart determinacy result.

Advantage: both players are dual...

Theorem

For every SPTG, all value functions are piecewise affine, with at most an exponential number of cutpoints (in number of locations).

Make a symmetric treatment of $\bigcirc$ and $\square$

Theorem

PTGs are determined ($\overline{\mathsf{Val}} = \underline{\mathsf{Val}} = \mathsf{Val}$), and value functions are continuous (over regions).

Determinacy follows from Gale-Stewart determinacy result.

Advantage: both players are dual...

Theorem

For every SPTG, all value functions are piecewise affine, with at most an exponential number of cutpoints (in number of locations).

Theorem

The value function of an SPTG can be computed in exponential time.

Toward more complex PTGs

What about resets ?

Toward more complex PTGs

What about resets ?

Immediate extension: reset acyclic 1-clock PTGs

Toward more complex PTGs

What about resets ?

Immediate extension: reset acyclic 1-clock PTGs

Current solution: cycles with resets have cost bounded away from 0

Subsume 1-clock robust games [Brenguier, Cassez, and Raskin, 2014]

Future Work

- Final extension of the result for all 1-clock PTGs?
- Use the result for 1-clock to approximate/compute the value of multiple-clocks PTGs with adequate structural properties
- Implementation and test of different algorithms on real instances

Future Work

- Final extension of the result for all 1-clock PTGs?
- Use the result for 1-clock to approximate/compute the value of multiple-clocks PTGs with adequate structural properties
- Implementation and test of different algorithms on real instances

Thank you for your attention

References I

Rajeev Alur, Mikhail Bernadsky, and P. Madhusudan. Optimal reachability for weighted timed games. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*, volume 3142 of *Lecture Notes in Computer Science*, pages 122–133. Springer, 2004.

Patricia Bouyer, Franck Cassez, Emmanuel Fleury, and Kim G. Larsen. Optimal strategies in priced timed game automata. In *Proceedings of the 24th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'04)*, volume 3328 of *Lecture Notes in Computer Science*, pages 148–160. Springer, 2004.

Patricia Bouyer, Thomas Brihaye, and Nicolas Markey. Improved undecidability results on weighted timed automata. *Information Processing Letters*, 98(5):188–194, 2006a.

Patricia Bouyer, Kim G. Larsen, Nicolas Markey, and Jacob Illum Rasmussen. Almost optimal strategies in one-clock priced timed games. In *Proceedings of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'06)*, volume 4337 of *Lecture Notes in Computer Science*, pages 345–356. Springer, 2006b.

Patricia Bouyer, Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On the optimal reachability problem of weighted timed automata. *Formal Methods in System Design*, 31(2):135–175, 2007.

References II

Romain Brenguier, Franck Cassez, and Jean-François Raskin. Energy and mean-payoff timed games. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control, HSCC'14, Berlin, Germany, April 15-17, 2014*, pages 283–292, 2014.

Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On optimal timed strategies. In *Proceedings of the Third international conference on Formal Modeling and Analysis of Timed Systems (FORMATS'05)*, volume 3829 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2005.

Thomas Brihaye, Gilles Geeraerts, Shankara Narayanan Krishna, Lakshmi Manasa, Benjamin Monmege, and Ashutosh Trivedi. Adding Negative Prices to Priced Timed Games. In *Proceedings of the 25th International Conference on Concurrency Theory (CONCUR'13)*, volume 8704 of *Lecture Notes in Computer Science*, pages 560–575. Springer, 2014.

Thomas Brihaye, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege. To reach or not to reach? Efficient algorithms for total-payoff games. In Luca Aceto and David de Frutos Escrig, editors, *Proceedings of the 26th International Conference on Concurrency Theory (CONCUR'15)*, volume 42 of *LIPIcs*, pages 297–310. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, September 2015.

John Fearnley and Marcin Jurdziński. Reachability in two-clock timed automata is PSPACE-complete. In *Proceedings of ICALP'13*, volume 7966 of *Lecture Notes in Computer Science*, pages 212–223. Springer, 2013.

References III

Christoph Haase, Joël Ouaknine, and James Worrell. On the relationship between reachability problems in timed and counter automata. In *Proceedings of RP'12*, pages 54–65, 2012.

Thomas Dueholm Hansen, Rasmus Ibsen-Jensen, and Peter Bro Miltersen. A faster algorithm for solving one-clock priced timed games. In *Proceedings of the 24th International Conference on Concurrency Theory (CONCUR'13)*, volume 8052 of *Lecture Notes in Computer Science*, pages 531–545. Springer, 2013.

Michał Rutkowski. Two-player reachability-price games on single-clock timed automata. In *Proceedings of the Ninth Workshop on Quantitative Aspects of Programming Languages (QAPL'11)*, volume 57 of *Electronic Proceedings in Theoretical Computer Science*, pages 31–46, 2011.