Optimal Reachability in Divergent Weighted Timed Games

Benjamin Monmege
Aix-Marseille Université, LIF, CNRS, France

with Damien Busatto-Gaston and Pierre-Alain Reynier

May 19, 2017

published at FoSSaCS 2017

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 $\Longrightarrow$ 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 $\Longrightarrow$ 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 $\Longrightarrow$ 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 $\Longrightarrow$ 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 $\Longrightarrow$ real-time controller

Two-player **timed** game

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

Two-player **weighted** timed game

Motivation: quantitative aspects of real-time synthesis

$$\boxed{\text{Environment}} \;\|\; \boxed{\text{Controller??}} \;\models\; \text{Spec}$$

Two-player game

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

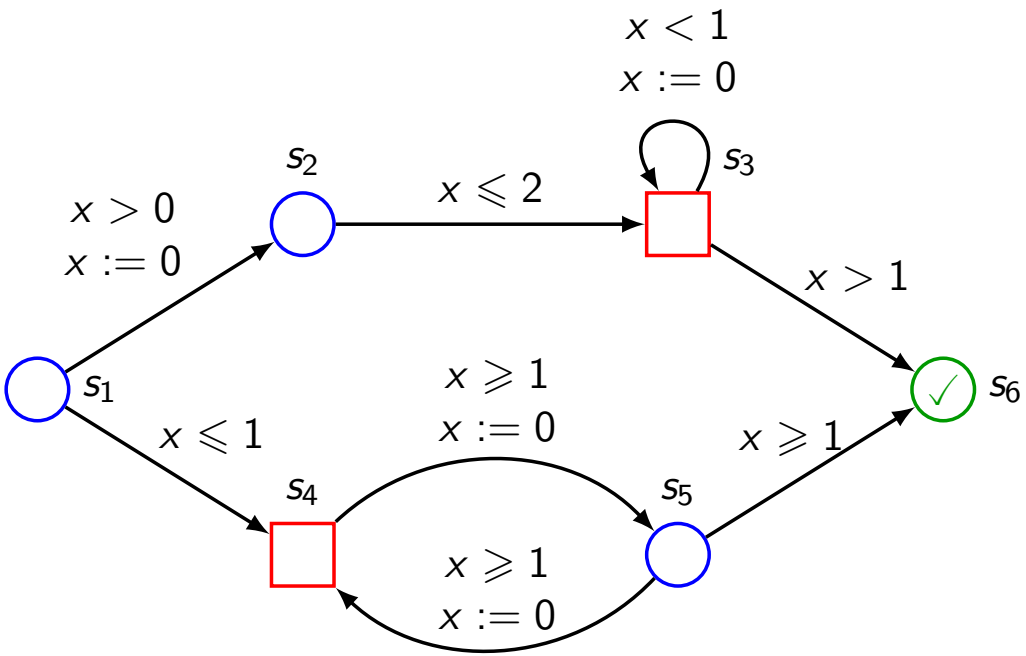
Two-player **timed** game

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

Two-player **weighted** timed game

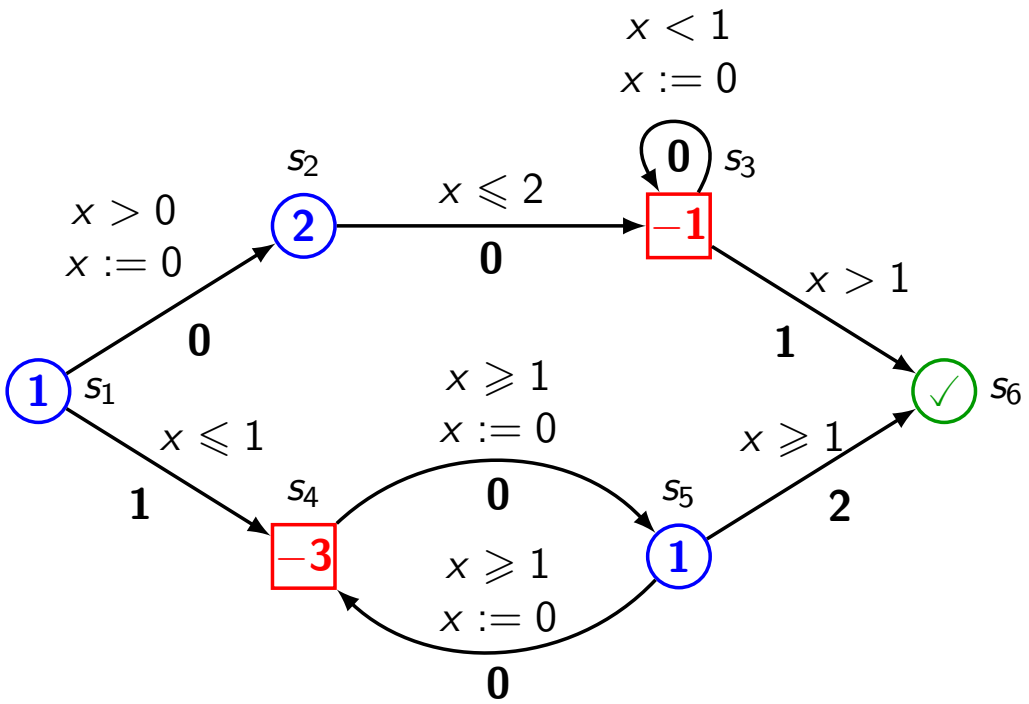
Main motivation since my postdoc here: **negative weights**
$\Longrightarrow$ model, e.g., production/consumption of resources

Weighted timed games



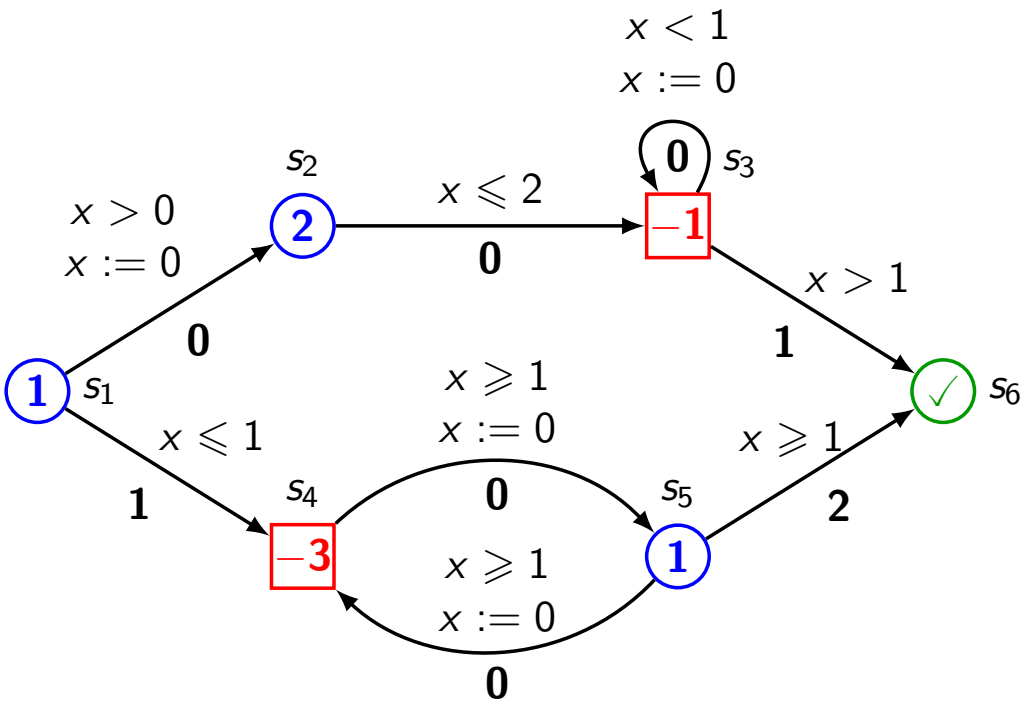
Timed automaton
with state partition between
2 players
+ reachability objective

Weighted timed games



Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

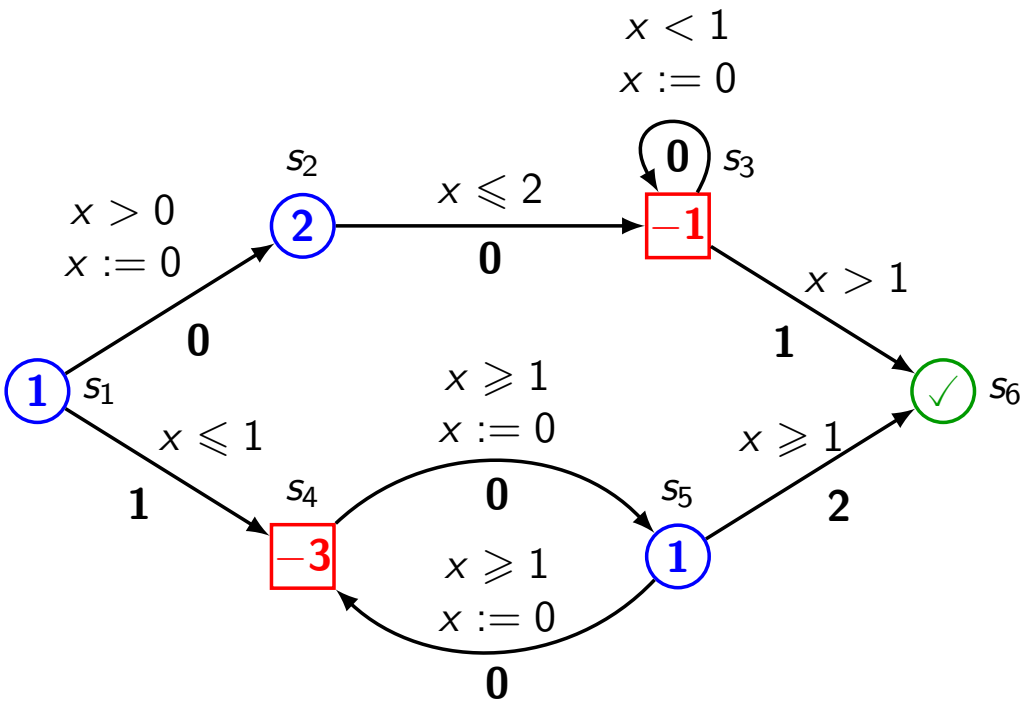
Weighted timed games



$(s_1, 0)$

Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

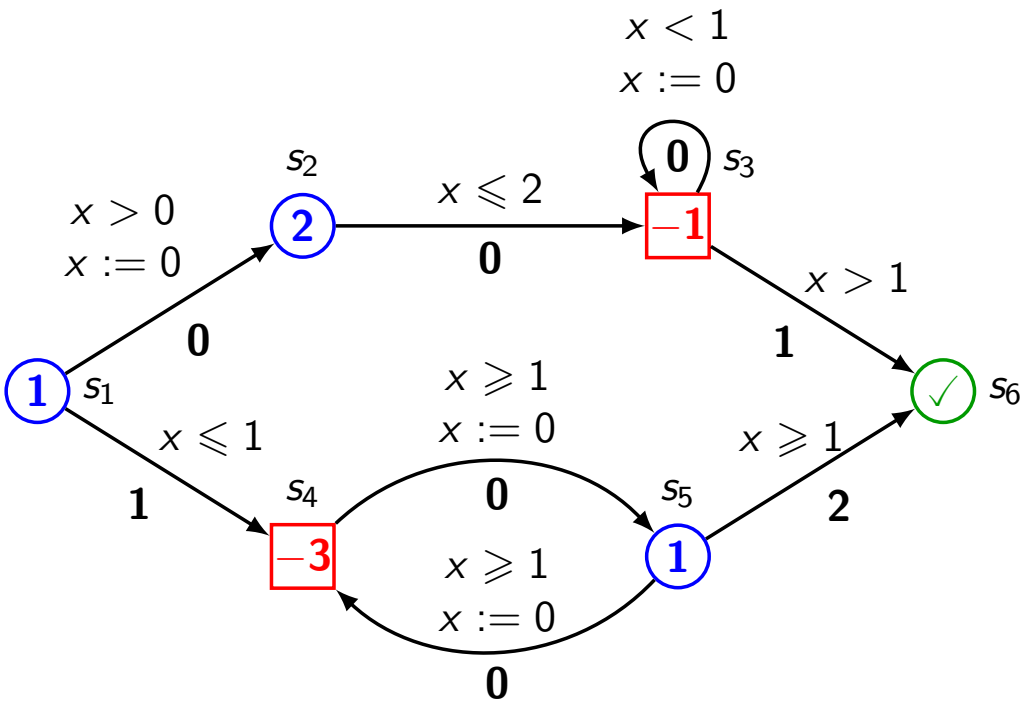
Weighted timed games



$(s_1, 0) \xrightarrow{0.4, \searrow} (s_4, 0.4)$

Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

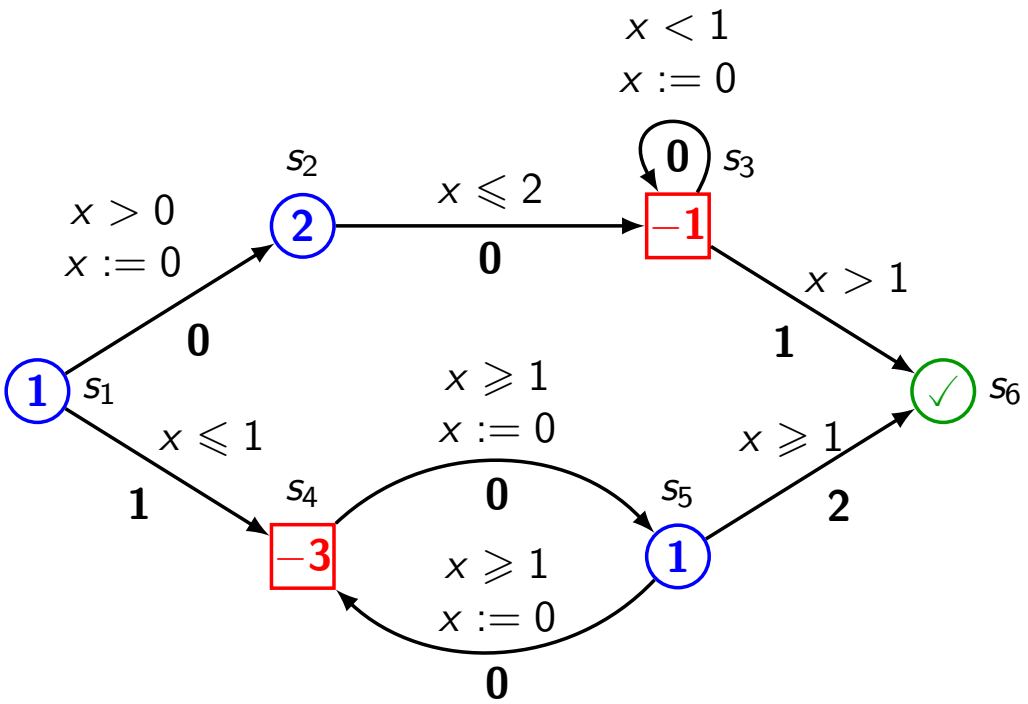
Weighted timed games



$$(s_1, 0) \xrightarrow{0.4, \searrow} (s_4, 0.4) \xrightarrow{0.6, \rightarrow} (s_5, 0)$$

Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

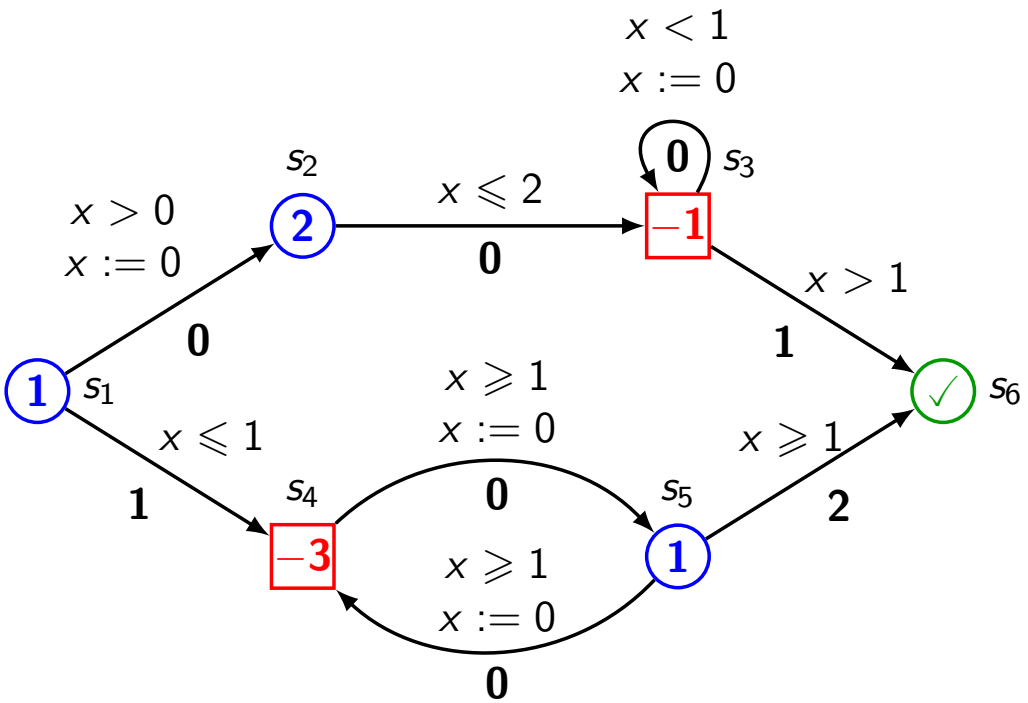
Weighted timed games



Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

$$(s_1, 0) \xrightarrow{0.4, \searrow} (s_4, 0.4) \xrightarrow{0.6, \rightarrow} (s_5, 0) \xrightarrow{1.5, \leftarrow} (s_4, 0) \xrightarrow{1.1, \rightarrow} (s_5, 0) \xrightarrow{2, \nearrow} (\checkmark, 2)$$

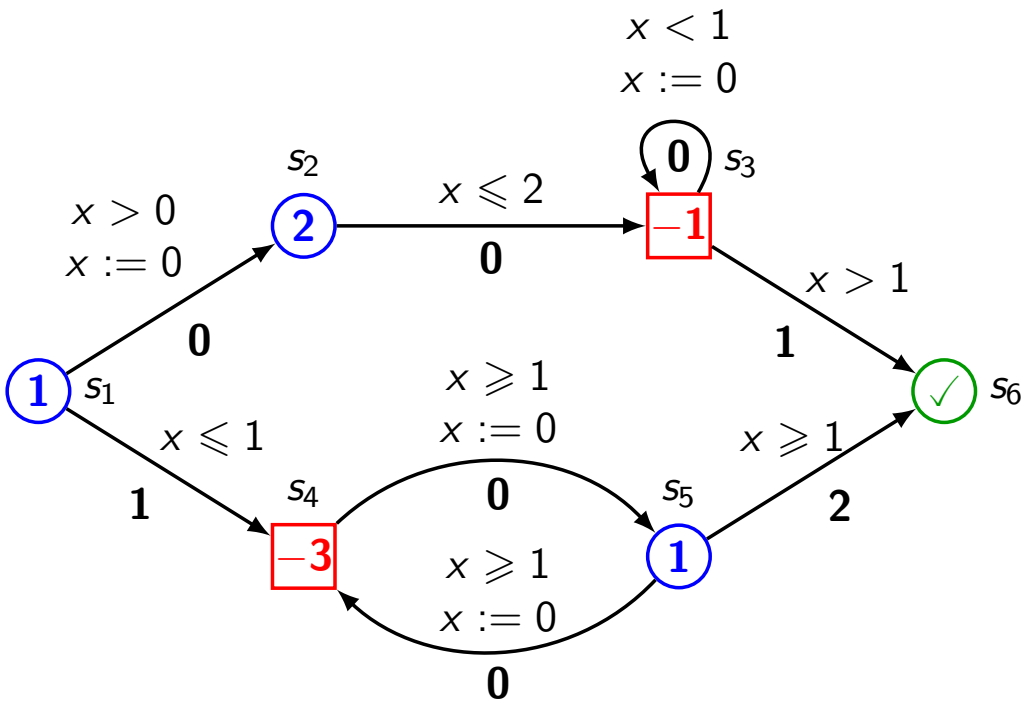
Weighted timed games



Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

$$(s_1,0) \xrightarrow[\mathbf{1}\times 0.4+\mathbf{1}]{0.4,\searrow} (s_4,0.4) \xrightarrow[-\mathbf{3}\times 0.6+\mathbf{0}]{0.6,\rightarrow} (s_5,0) \xrightarrow[+\mathbf{1}\times 1.5+\mathbf{0}]{1.5,\leftarrow} (s_4,0) \xrightarrow[-\mathbf{3}\times 1.1+\mathbf{0}]{1.1,\rightarrow} (s_5,0) \xrightarrow[+\mathbf{1}\times 2+\mathbf{2}]{2,\nearrow} (\checkmark,2) \qquad = 1.8$$

Weighted timed games



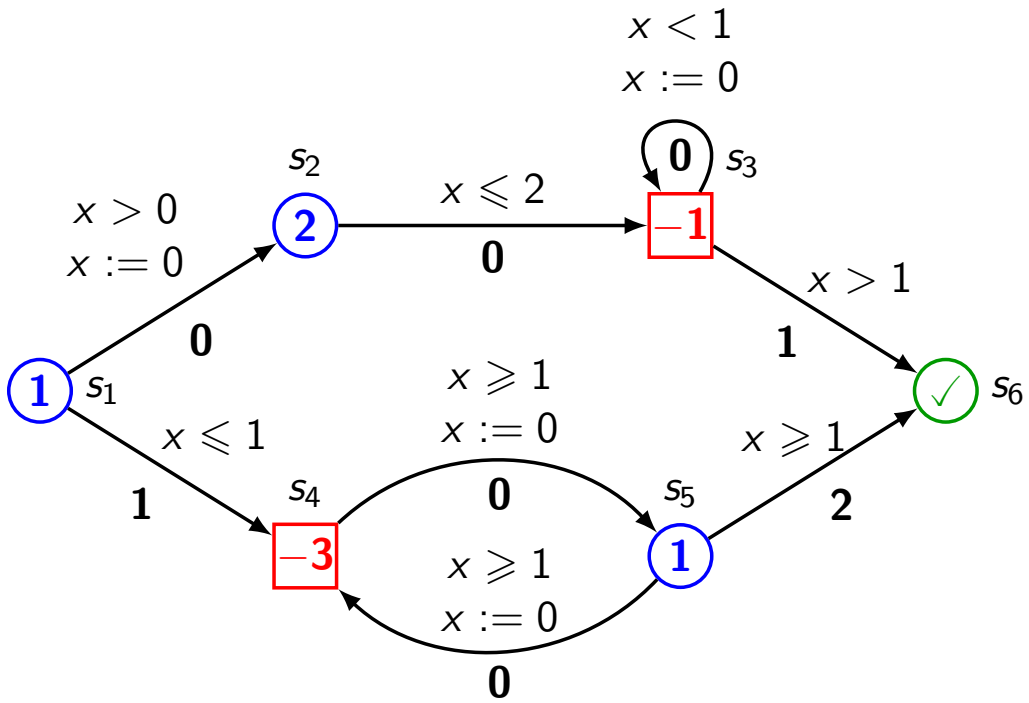
Timed automaton
with state partition between
2 players
+ reachability objective
+ linear rates on states
+ discrete weights on
transitions

$$(s_1,0) \xrightarrow[\mathbf{1}\times 0.4+\mathbf{1}]{0.4,\searrow} (s_4,0.4) \xrightarrow[-\mathbf{3}\times 0.6+\mathbf{0}]{0.6,\rightarrow} (s_5,0) \xrightarrow[+\mathbf{1}\times 1.5+\mathbf{0}]{1.5,\leftarrow} (s_4,0) \xrightarrow[-\mathbf{3}\times 1.1+\mathbf{0}]{1.1,\rightarrow} (s_5,0) \xrightarrow[+\mathbf{1}\times 2+\mathbf{2}]{2,\nearrow} (\checkmark,2) \qquad = 1.8$$

$$(s_1,0) \xrightarrow[\mathbf{1}\times 0.2+\mathbf{0}]{0.2,\nearrow} (s_2,0) \xrightarrow[+\mathbf{2}\times 0.9+\mathbf{0}]{0.9,\rightarrow} (s_3,0.9) \xrightarrow[-\mathbf{1}\times 0.2+\mathbf{0}]{0.2,\circlearrowright} (s_3,0) \xrightarrow[-\mathbf{1}\times 0.9+\mathbf{0}]{0.9,\circlearrowright} (s_3,0) \quad \cdots \quad \cdots \qquad = +\infty$$

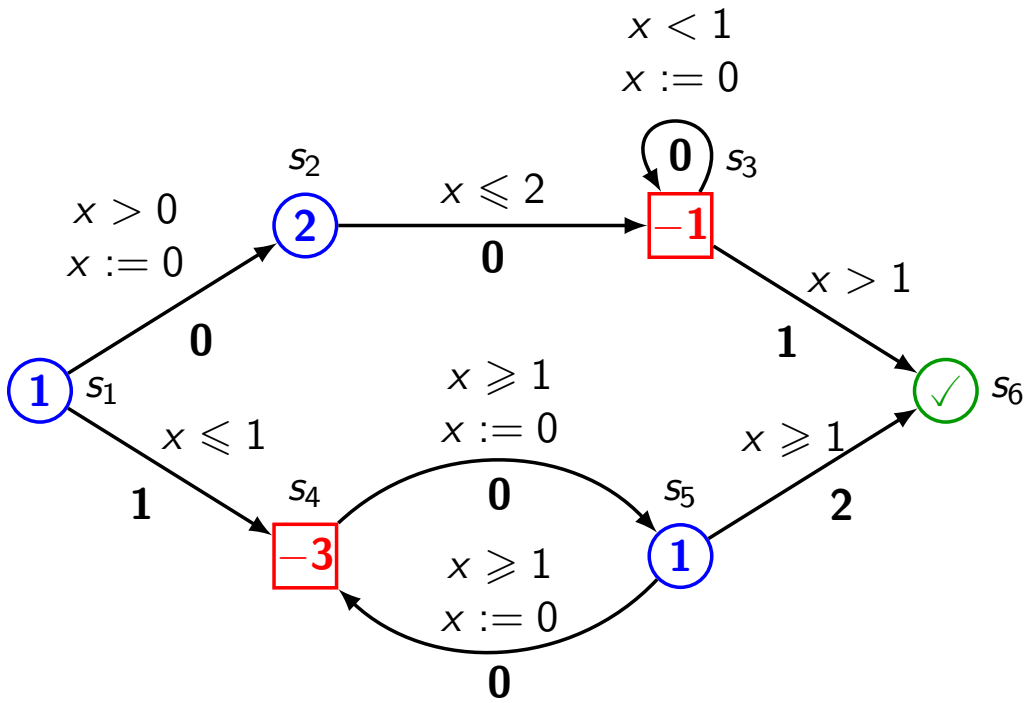
Weight of an execution : $\begin{cases} +\infty & \text{if } \checkmark \text{ not reached} \\ \text{total weight until } \checkmark & \text{otherwise} \end{cases}$

Strategies and objectives



Strategy for a player: map finite executions to a delay and a transition

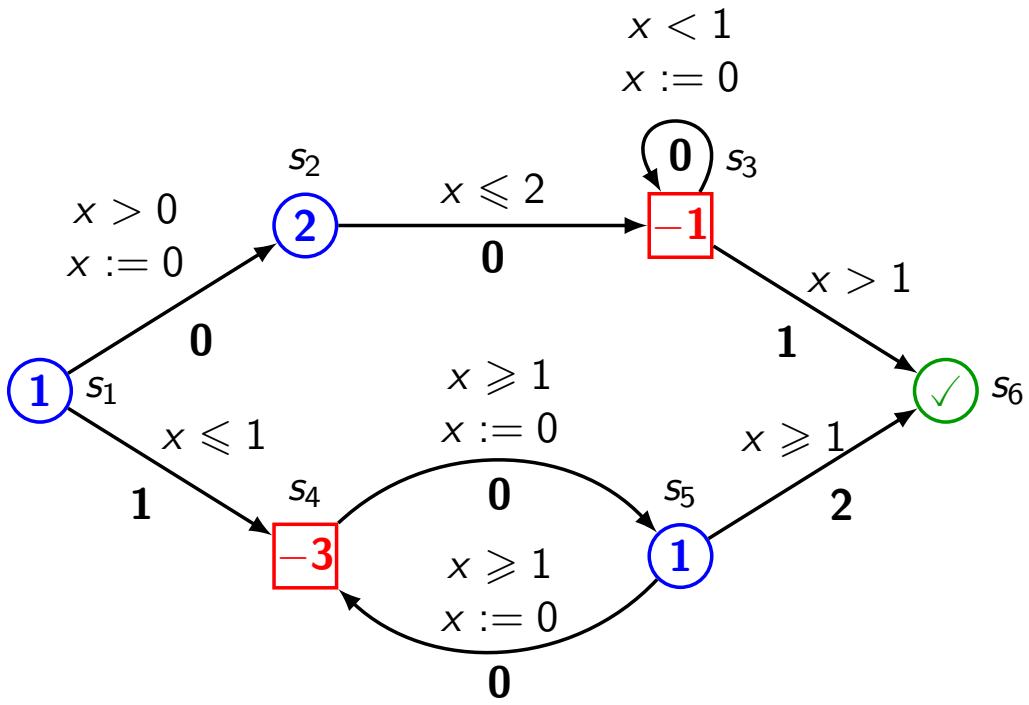
Strategies and objectives



Strategy for a player: map finite executions to a delay and a transition

Objective of player ○: reach ✓ **and** minimise the weight
Objective of player □: avoid ✓ **or, if not possible,** maximise the weight

Strategies and objectives



Strategy for a player: map finite executions to a delay and a transition

Objective of player ○: reach ✓ **and** minimise the weight
Objective of player □: avoid ✓ **or, if not possible,** maximise the weight

Main object of interest:
$\mathrm{Val}(\ell, \nu) = \inf_{\sigma_{\mathrm{Min}} \in \mathrm{Strat}^{\mathrm{Min}}} \sup_{\sigma_{\mathrm{Max}} \in \mathrm{Strat}^{\mathrm{Max}}} \mathrm{Weight}(\mathrm{Exec}(\ell, \nu, \sigma_{\mathrm{Min}}, \sigma_{\mathrm{Max}})) \in \overline{\mathbb{R}}$
What weight can players guarantee? Find (almost) optimal strategy?

The real motivation!

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

Two-player game

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

Two-player **timed** game

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

Two-player **weighted** timed game

Main motivation since my postdoc here: **negative weights**

The real motivation!

$$\boxed{\text{Bar}} \quad \| \quad \boxed{\text{Controller??}} \quad \models \quad \text{Spec}$$

Two-player game

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

Two-player **timed** game

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

Two-player **weighted** timed game

Main motivation since my postdoc here: **negative weights**

The real motivation!

$$\boxed{\text{Bar}} \quad \| \quad \boxed{\text{Customer??}} \quad \models \quad \text{Spec}$$

Two-player game

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

Two-player **timed** game

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

Two-player **weighted** timed game

Main motivation since my postdoc here: **negative weights**

The real motivation!

$$\boxed{\text{Bar}} \quad \| \quad \boxed{\text{Customer??}} \quad \models \quad \text{Beer!}$$

Two-player game

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

Two-player **timed** game

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

Two-player **weighted** timed game

Main motivation since my postdoc here: **negative weights**

The real motivation!

$$\boxed{\text{Bar}} \quad \| \quad \boxed{\text{Customer??}} \quad \models \quad \text{Beer!}$$

Two-player game

"*I want a beer, and I want it fast! But barman needs time...*"

Two-player **timed** game

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

Two-player **weighted** timed game

Main motivation since my postdoc here: **negative weights**

The real motivation!

$$\boxed{\text{Bar}} \quad \| \quad \boxed{\text{Customer??}} \quad \models \quad \text{Beer!}$$

Two-player game

"*I want a beer, and I want it fast! But barman needs time...*"

Two-player **timed** game

"*I want a good beer, but I want it cheap!*"

Two-player **weighted** timed game

Main motivation since my postdoc here: **negative weights**

The real motivation!

$$\boxed{\text{Bar}} \quad \| \quad \boxed{\text{Customer??}} \quad \models \quad \text{Beer!}$$

Two-player game

"*I want a beer, and I want it fast! But barman needs time...*"

Two-player **timed** game

"*I want a good beer, but I want it cheap!*"

Two-player **weighted** timed game

Main motivation since my postdoc here: **drink tasty beers!**

State of the art

Value problem: decide whether $\mathsf{Val}(\ell, \nu) \leqslant K$?

State of the art

Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)

State of the art

Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)
- 2 players: undecidable even with non-negative weights and 3 clocks (Brihaye, Bruyère, and Raskin 2005; Bouyer, Brihaye, and Markey 2006)

State of the art

Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)
- 2 players: undecidable even with non-negative weights and 3 clocks (Brihaye, Bruyère, and Raskin 2005; Bouyer, Brihaye, and Markey 2006)
- **1 clock** and **non-negative weights**: exp. algo. (Bouyer, Larsen, Markey, and Rasmussen 2006)

State of the art

Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)
- 2 players: undecidable even with non-negative weights and 3 clocks (Brihaye, Bruyère, and Raskin 2005; Bouyer, Brihaye, and Markey 2006)
- **1 clock** and **non-negative weights**: exp. algo. (Bouyer, Larsen, Markey, and Rasmussen 2006)
- **1 clock** and $\leqslant$ **2 location-weights from** $\{-d, 0, +d\}$: pseudo-poly. time algo. (Brihaye, Geeraerts, Narayanan Krishna, Manasa, Monmege, and Trivedi 2014) (corner-point abstraction)

State of the art

Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)
- 2 players: undecidable even with non-negative weights and 3 clocks (Brihaye, Bruyère, and Raskin 2005; Bouyer, Brihaye, and Markey 2006)
- **1 clock** and **non-negative weights**: exp. algo. (Bouyer, Larsen, Markey, and Rasmussen 2006)
- **1 clock** and $\leqslant$ **2 location-weights from** $\{-d, 0, +d\}$: pseudo-poly. time algo. (Brihaye, Geeraerts, Narayanan Krishna, Manasa, Monmege, and Trivedi 2014) (corner-point abstraction)
- **non-negative weights and strictly non-Zeno-cost cycles**: 2-exp. algo. (Bouyer, Cassez, Fleury, and Larsen 2004; Alur, Bernadsky, and Madhusudan 2004)

State of the art

Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)
- 2 players: undecidable even with non-negative weights and 3 clocks (Brihaye, Bruyère, and Raskin 2005; Bouyer, Brihaye, and Markey 2006)
- **1 clock** and **non-negative weights**: exp. algo. (Bouyer, Larsen, Markey, and Rasmussen 2006)
- **1 clock** and $\leqslant$ **2 location-weights from** $\{-d, 0, +d\}$: pseudo-poly. time algo. (Brihaye, Geeraerts, Narayanan Krishna, Manasa, Monmege, and Trivedi 2014) (corner-point abstraction)
- **non-negative weights and strictly non-Zeno-cost cycles**: 2-exp. algo. (Bouyer, Cassez, Fleury, and Larsen 2004; Alur, Bernadsky, and Madhusudan 2004)

$$\mathcal{F}(\boldsymbol{x})_{(\ell,\nu)} = \begin{cases} \sup_{(\ell,\nu) \xrightarrow{d,t} (\ell',\nu')} d \times \mathrm{Weight}(\ell) + \mathrm{Weight}(t) + \boldsymbol{x}_{(\ell',\nu')} & \text{if } \ell \in L_{\mathrm{Max}} \\ \inf_{(\ell,\nu) \xrightarrow{d,t} (\ell',\nu')} d \times \mathrm{Weight}(\ell) + \mathrm{Weight}(t) + \boldsymbol{x}_{(\ell',\nu')} & \text{if } \ell \in L_{\mathrm{Min}} \end{cases}$$

State of the art

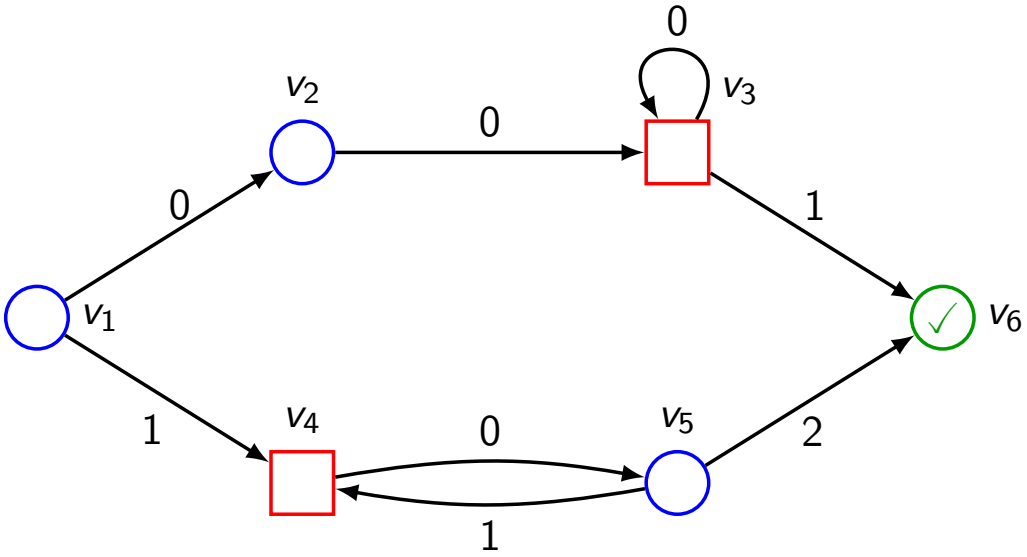
Value problem: decide whether $\mathrm{Val}(\ell, \nu) \leqslant K$?

- 1 player (**weighted timed automaton**): **PSPACE**-complete
  - Algorithm based on regions (Bouyer, Brihaye, Bruyère, and Raskin 2007);
  - **PSPACE**-hard with at least 2 clocks (Fearnley and Jurdziński 2013)
- 2 players: undecidable even with non-negative weights and 3 clocks (Brihaye, Bruyère, and Raskin 2005; Bouyer, Brihaye, and Markey 2006)
- **1 clock** and **non-negative weights**: exp. algo. (Bouyer, Larsen, Markey, and Rasmussen 2006)
- **1 clock** and $\leqslant$ **2 location-weights from** $\{-d, 0, +d\}$: pseudo-poly. time algo. (Brihaye, Geeraerts, Narayanan Krishna, Manasa, Monmege, and Trivedi 2014) (corner-point abstraction)
- **non-negative weights and strictly non-Zeno-cost cycles**: 2-exp. algo. (Bouyer, Cassez, Fleury, and Larsen 2004; Alur, Bernadsky, and Madhusudan 2004)

$$\mathcal{F}(\boldsymbol{x})_{(\ell,\nu)} = \begin{cases} \sup_{(\ell,\nu) \xrightarrow{d,t} (\ell',\nu')} d \times \mathrm{Weight}(\ell) + \mathrm{Weight}(t) + \boldsymbol{x}_{(\ell',\nu')} & \text{if } \ell \in L_{\mathrm{Max}} \\ \inf_{(\ell,\nu) \xrightarrow{d,t} (\ell',\nu')} d \times \mathrm{Weight}(\ell) + \mathrm{Weight}(t) + \boldsymbol{x}_{(\ell',\nu')} & \text{if } \ell \in L_{\mathrm{Min}} \end{cases}$$

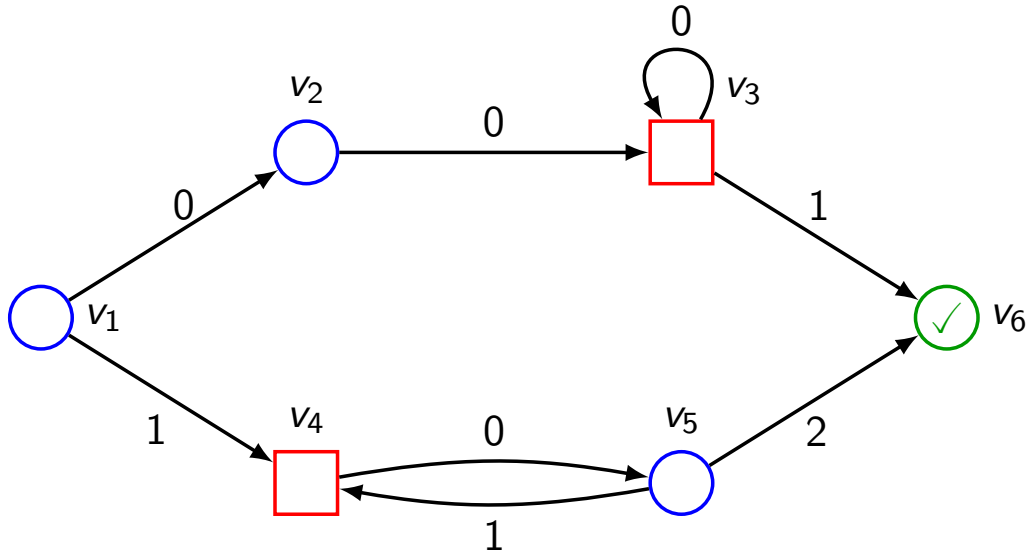
Missing: decidable fragment of games without constraints on number of clocks, and with negative weights

A special case first: Weighted (untimed) games



Weighted oriented graph
with vertices partition
between 2 players
+ reachability objective

A special case first: Weighted (untimed) games



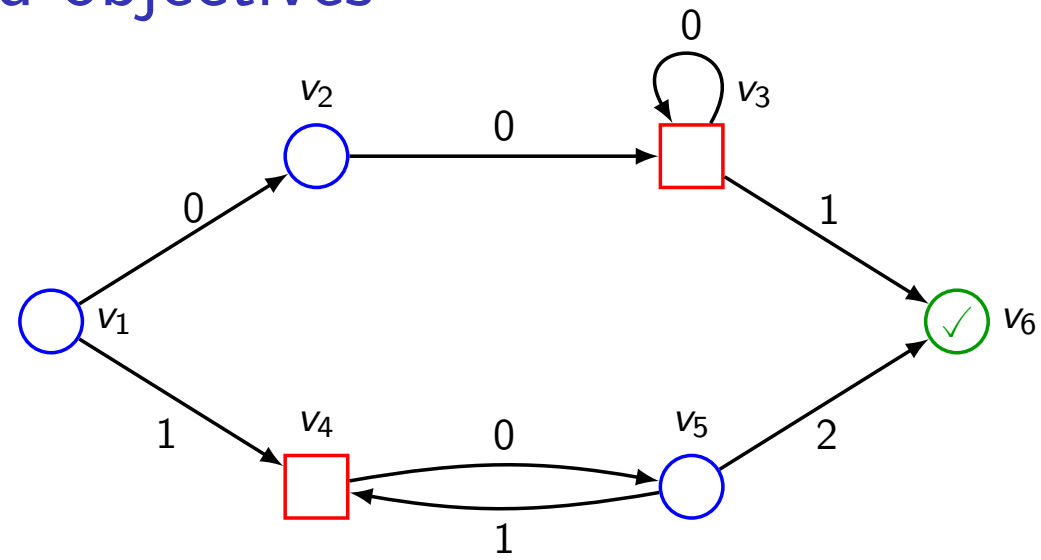
Weighted oriented graph
with vertices partition
between 2 players
+ reachability objective

$$v_1 \searrow v_4 \rightarrow v_5 \leftarrow v_4 \rightarrow v_5 \nearrow \checkmark$$
$$1 \qquad +1 \qquad +2 \qquad = 4$$

$$v_1 \nearrow v_2 \rightarrow v_3 \circlearrowright v_3 \circlearrowright v_3 \quad \cdots$$
$$\cdots \qquad = +\infty \ (\checkmark \text{ not reached})$$

Weight of a path: $\begin{cases} +\infty & \text{if } \checkmark \text{ not reached} \\ \text{total weight until } \checkmark & \text{otherwise} \end{cases}$

Strategies and objectives



Player ○'s objective: reach ✓ **and** minimise the cost
Player □'s objective: avoid ✓ **or, if not possible,** maximise the weight

Strategy for a player: map finite paths to edges

Main object of interest:
$\mathrm{Val}(v) = \inf_{\sigma_{\mathrm{Min}} \in \mathrm{Strat}^{\mathrm{Min}}} \sup_{\sigma_{\mathrm{Max}} \in \mathrm{Strat}^{\mathrm{Max}}} \mathrm{Weight}(\mathrm{Exec}(v, \sigma_{\mathrm{Min}}, \sigma_{\mathrm{Max}})) \in \overline{\mathbb{Z}}$
What weight can players guarantee? Find optimal strategy?

State of the art

Value computation

- non-negative weights: polynomial algo. (Khachiyan, Boros, Borys, Elbassioni, Gurvich, Rudolf, et al. 2008)
  "Dijkstra on 2 players games"

State of the art

Value computation

- non-negative weights: polynomial algo. (Khachiyan, Boros, Borys, Elbassioni, Gurvich, Rudolf, et al. 2008)
  "Dijkstra on 2 players games"
- with negative weights : pseudo-poly algo. (Brihaye, Geeraerts, Haddad, and Monmege 2016)
  "value iteration algorithm"

$$\mathcal{F}(\boldsymbol{x})_v = \begin{cases} \min\limits_{e=(v,a,v')\in E} \mathrm{Weight}(e) + \boldsymbol{x}_{v'} & \text{if } v \in V_{\mathrm{Min}} \\ \max\limits_{e=(v,a,v')\in E} \mathrm{Weight}(e) + \boldsymbol{x}_{v'} & \text{if } v \in V_{\mathrm{Max}} \,. \end{cases}$$

State of the art

Value computation

- non-negative weights: polynomial algo. (Khachiyan, Boros, Borys, Elbassioni, Gurvich, Rudolf, et al. 2008)
  "Dijkstra on 2 players games"
- with negative weights : pseudo-poly algo. (Brihaye, Geeraerts, Haddad, and Monmege 2016)
  "value iteration algorithm"

$$\mathcal{F}(\boldsymbol{x})_v = \begin{cases} \min\limits_{e=(v,a,v')\in E} \text{Weight}(e) + \boldsymbol{x}_{v'} & \text{if } v \in V_{\text{Min}} \\ \max\limits_{e=(v,a,v')\in E} \text{Weight}(e) + \boldsymbol{x}_{v'} & \text{if } v \in V_{\text{Max}} \,. \end{cases}$$

- Value $-\infty$: detection is as hard as solving parity games ($\mathbf{NP} \cap \mathbf{co\text{-}NP}$)

Contribution 1: divergent weighted games

Property of the underlying graph:

Divergence

Every cycle have total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no cycles of weight $= 0$.

Contribution 1: divergent weighted games

Property of the underlying graph:

Divergence

Every cycle have total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no cycles of weight $= 0$.

Theorem:

We can compute the value of a divergent weighted game in poly. time.

Contribution 1: divergent weighted games

Property of the underlying graph:

Divergence

Every cycle have total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no cycles of weight $= 0$.

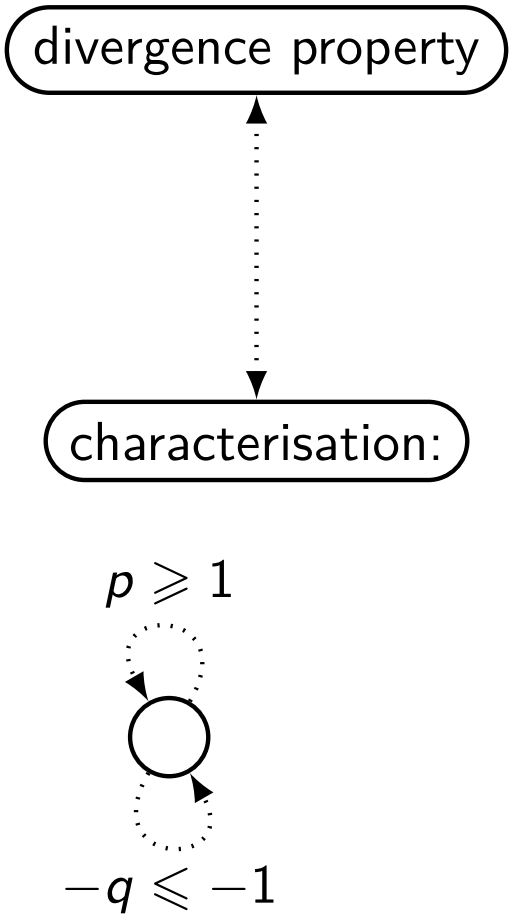
Theorem:

We can compute the value of a divergent weighted game in poly. time.

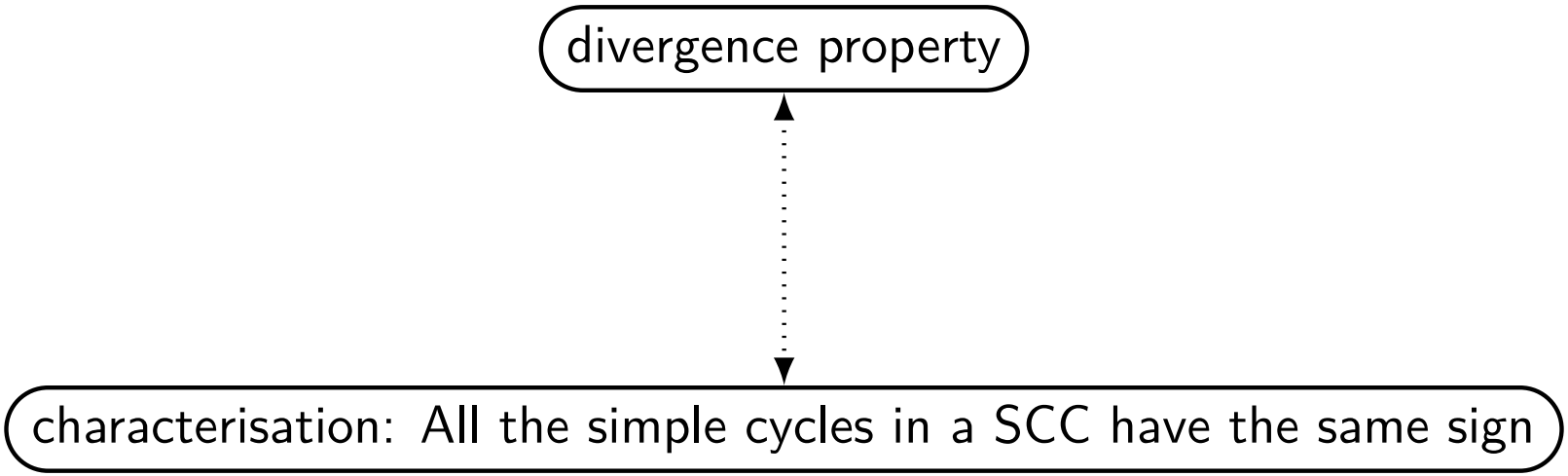
Theorem:

We can decide in PTIME if a weighted game is divergent.

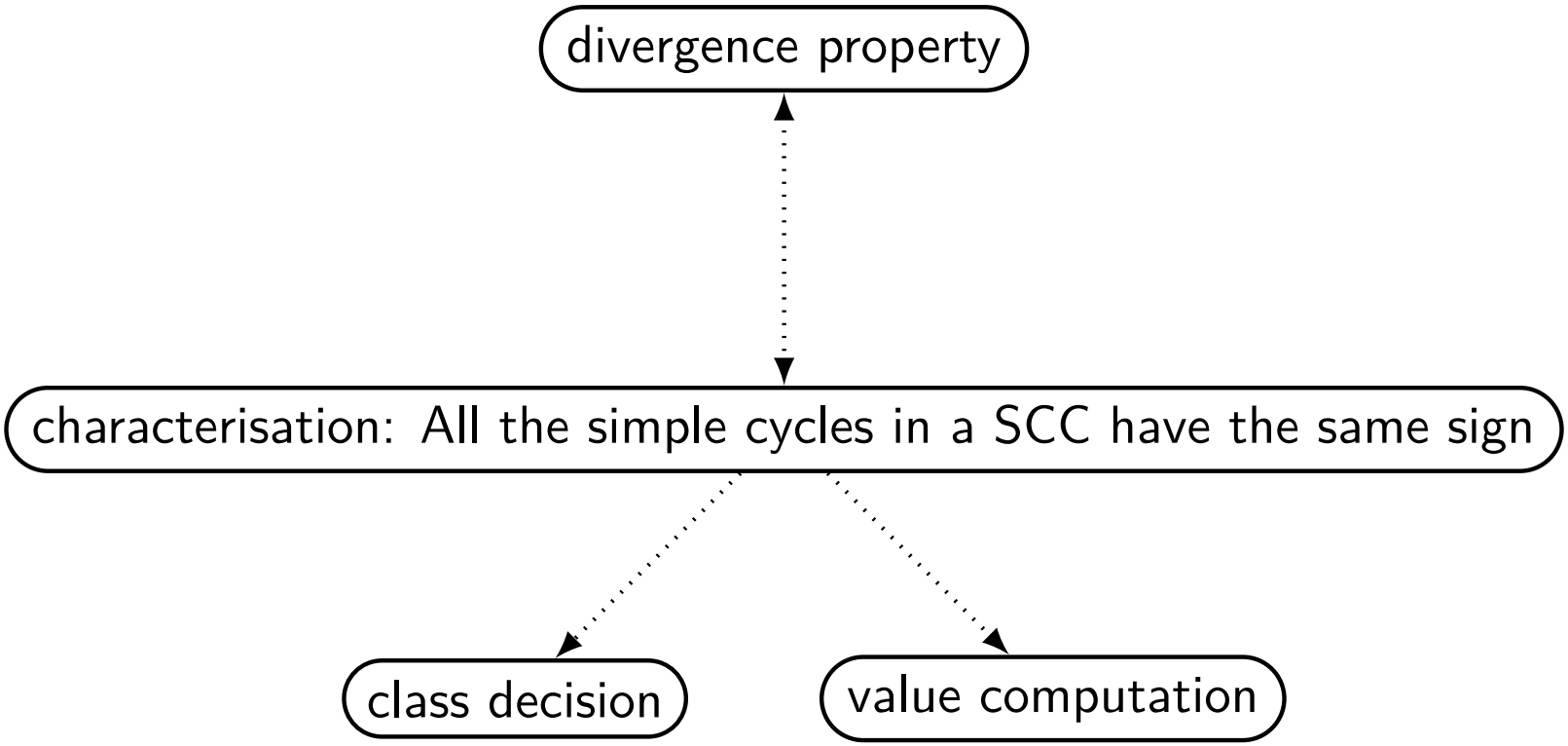
Divergent weighted games analysis



Divergent weighted games analysis



Divergent weighted games analysis



Value computation in a divergent weighted game

- Detect and remove $+\infty$ vertices (outside of the attractor of player $\bigcirc$ toward $\checkmark$)

Value computation in a divergent weighted game

- Detect and remove $+\infty$ vertices (outside of the attractor of player $\bigcirc$ toward $\checkmark$)
- SCC decomposition

Value computation in a divergent weighted game

- Detect and remove $+\infty$ vertices (outside of the attractor of player $\bigcirc$ toward $\checkmark$)
- SCC decomposition
- Value computation SCC by SCC, bottom-up

Value computation in a divergent weighted game

- Detect and remove $+\infty$ vertices (outside of the attractor of player $\bigcirc$ toward $\checkmark$)
- SCC decomposition
- Value computation SCC by SCC, bottom-up

positive SCC

- The "value iteration" algorithm converges in linear time

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi : v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\Longrightarrow$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi : v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$
  - $|\pi| > (2K + W(n-1))n \Longrightarrow \exists v''$ appearing $\geqslant 2K + W(n-1)$ times

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi : v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$
  - $|\pi| > (2K + W(n-1))n \implies \exists v''$ appearing $\geqslant 2K + W(n-1)$ times
  - π can be decomposed into $\geqslant 2K + W(n-1)$ cycles (positive!) and a finite play π' visiting each vertex at most once

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi : v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$
  - $|\pi| > (2K + W(n-1))n \implies \exists v''$ appearing $\geqslant 2K + W(n-1)$ times
  - π can be decomposed into $\geqslant 2K + W(n-1)$ cycles (positive!) and a finite play π' visiting each vertex at most once
  - $\text{Weight}(\pi) \geqslant 2K + W(n-1) - (n-1)W = 2K$

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\Longrightarrow$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi \colon v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$
  - $|\pi| > (2K + W(n-1))n \Longrightarrow \exists v''$ appearing $\geqslant 2K + W(n-1)$ times
  - π can be decomposed into $\geqslant 2K + W(n-1)$ cycles (positive!) and a finite play π' visiting each vertex at most once
  - $\text{Weight}(\pi) \geqslant 2K + W(n-1) - (n-1)W = 2K$
  - $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n \geqslant 2K$, so $\boldsymbol{x}_v^{n+p} \geqslant 2K + \boldsymbol{x}_{v'}^n \geqslant K$.

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi : v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$
  - $|\pi| > (2K + W(n-1))n \implies \exists v''$ appearing $\geqslant 2K + W(n-1)$ times
  - π can be decomposed into $\geqslant 2K + W(n-1)$ cycles (positive!) and a finite play π' visiting each vertex at most once
  - $\text{Weight}(\pi) \geqslant 2K + W(n-1) - (n-1)W = 2K$
  - $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n \geqslant 2K$, so $\boldsymbol{x}_v^{n+p} \geqslant 2K + \boldsymbol{x}_{v'}^n \geqslant K$.
  - But $K \geqslant \boldsymbol{x}_v^n$, so $\boldsymbol{x}_v^{n+p} \geqslant \boldsymbol{x}_v^n$: contradiction.

Positive SCC: value iteration converges in linear time

- $W := \max_{e \in E} |\text{Weight}(e)|$
- No negative cycles in the SCC $\implies$ no vertices of value $-\infty$
- $K := \max_v |\boldsymbol{x}_v^n| < \infty$, with $n = |V|$
- Let $p > (2K + W(n-1))n$. Values obtained after $n + p$ steps are identical to those obtained after n steps only!
  - If $\boldsymbol{x}_v^{n+p} < \boldsymbol{x}_v^n$ $\exists v', \pi : v \to^p v'$ of weight $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n$
  - $|\pi| > (2K + W(n-1))n \implies \exists v''$ appearing $\geqslant 2K + W(n-1)$ times
  - π can be decomposed into $\geqslant 2K + W(n-1)$ cycles (positive!) and a finite play π' visiting each vertex at most once
  - $\text{Weight}(\pi) \geqslant 2K + W(n-1) - (n-1)W = 2K$
  - $\boldsymbol{x}_v^{n+p} - \boldsymbol{x}_{v'}^n \geqslant 2K$, so $\boldsymbol{x}_v^{n+p} \geqslant 2K + \boldsymbol{x}_{v'}^n \geqslant K$.
  - But $K \geqslant \boldsymbol{x}_v^n$, so $\boldsymbol{x}_v^{n+p} \geqslant \boldsymbol{x}_v^n$: contradiction.
- Therefore, stabilisation after n steps only

Value computation in a weighted game

- Detect and remove $+\infty$ vertices (outside of the attractor of player ○ toward ✓)
- SCC decomposition
- Value computation SCC by SCC, bottom-up

positive SCC

- The "value iteration" algorithm converges in linear time

Value computation in a weighted game

- Detect and remove $+\infty$ vertices (outside of the attractor of player ○ toward ✓)
- SCC decomposition
- Value computation SCC by SCC, bottom-up

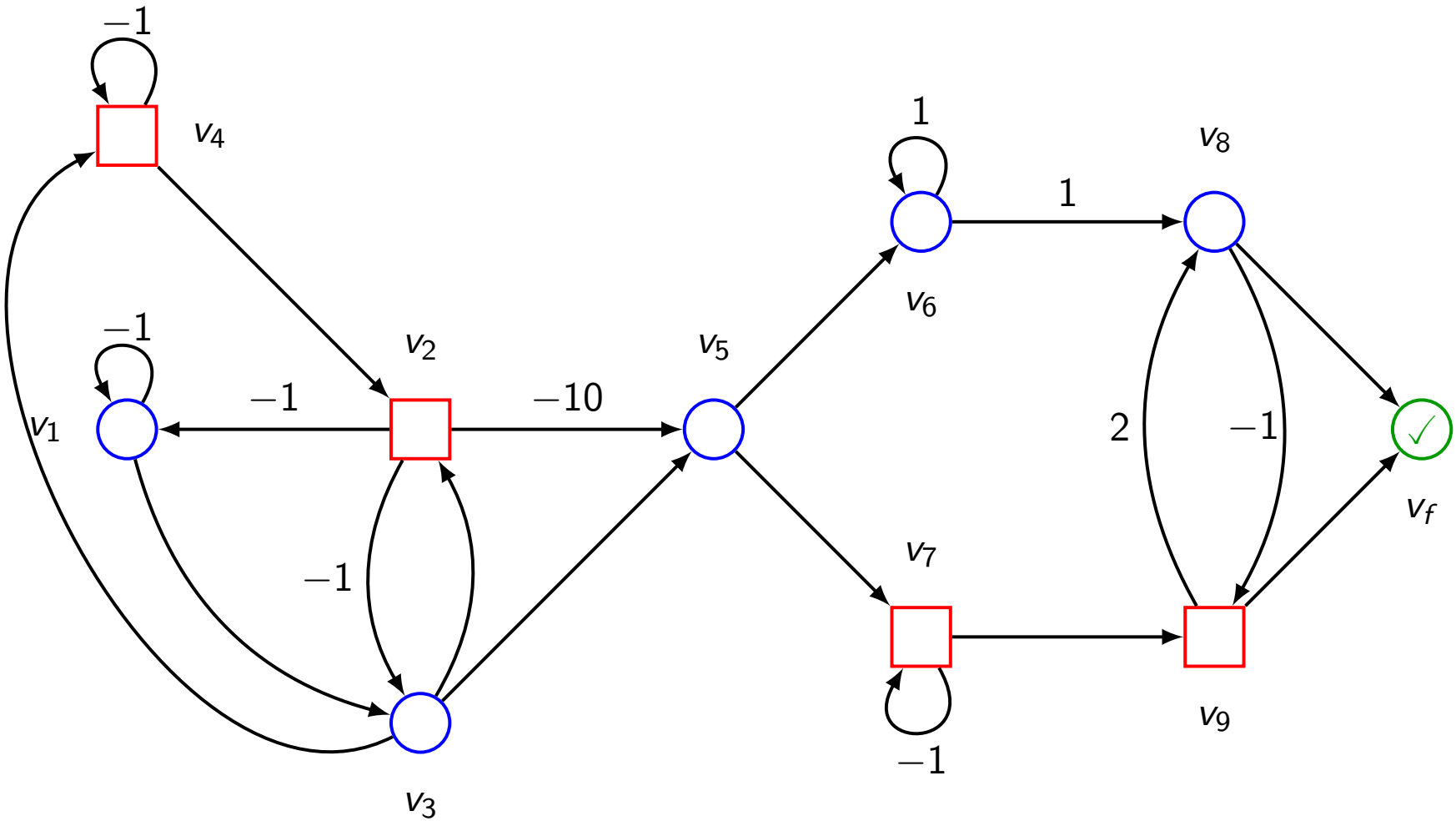
positive SCC

- The "value iteration" algorithm converges in linear time

negative SCC

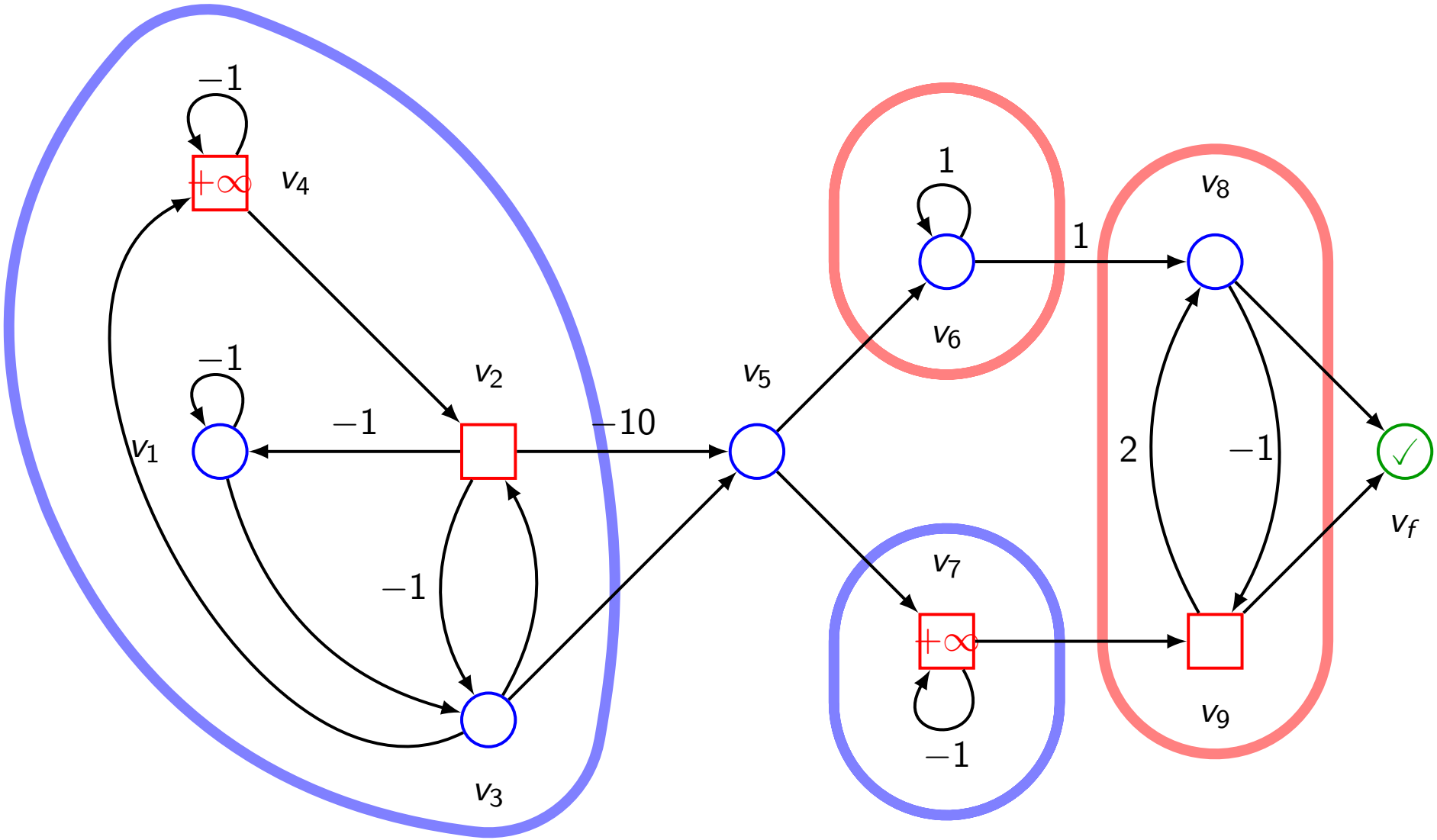
- Outside of the attractor of player □ toward ✓ $\Rightarrow -\infty$
- The "value iteration" algorithm converges in linear time with initialisation at $-\infty$

Example



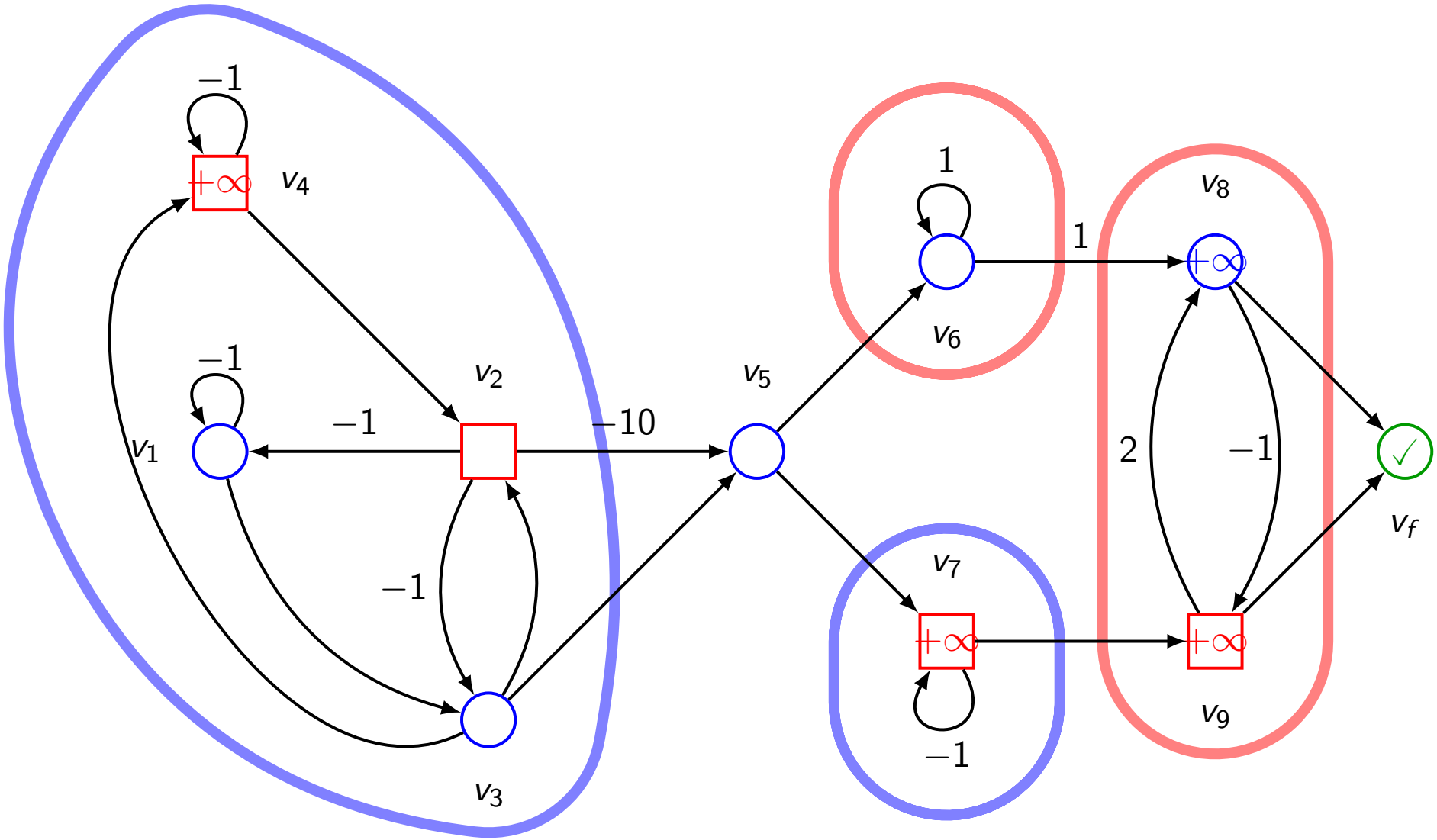
Min = ○, Max = □

Example



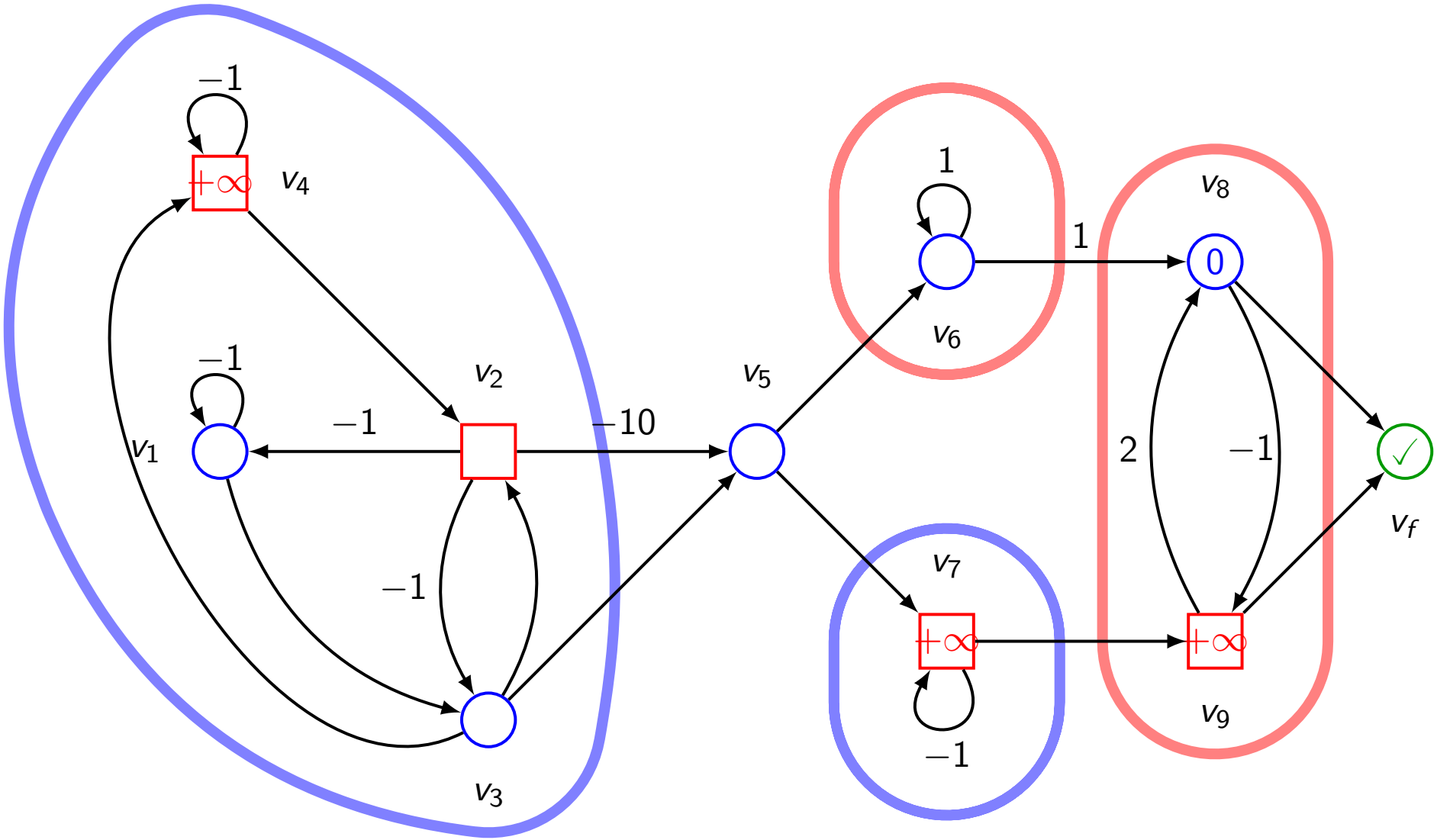
Min = ○, Max = □

Example



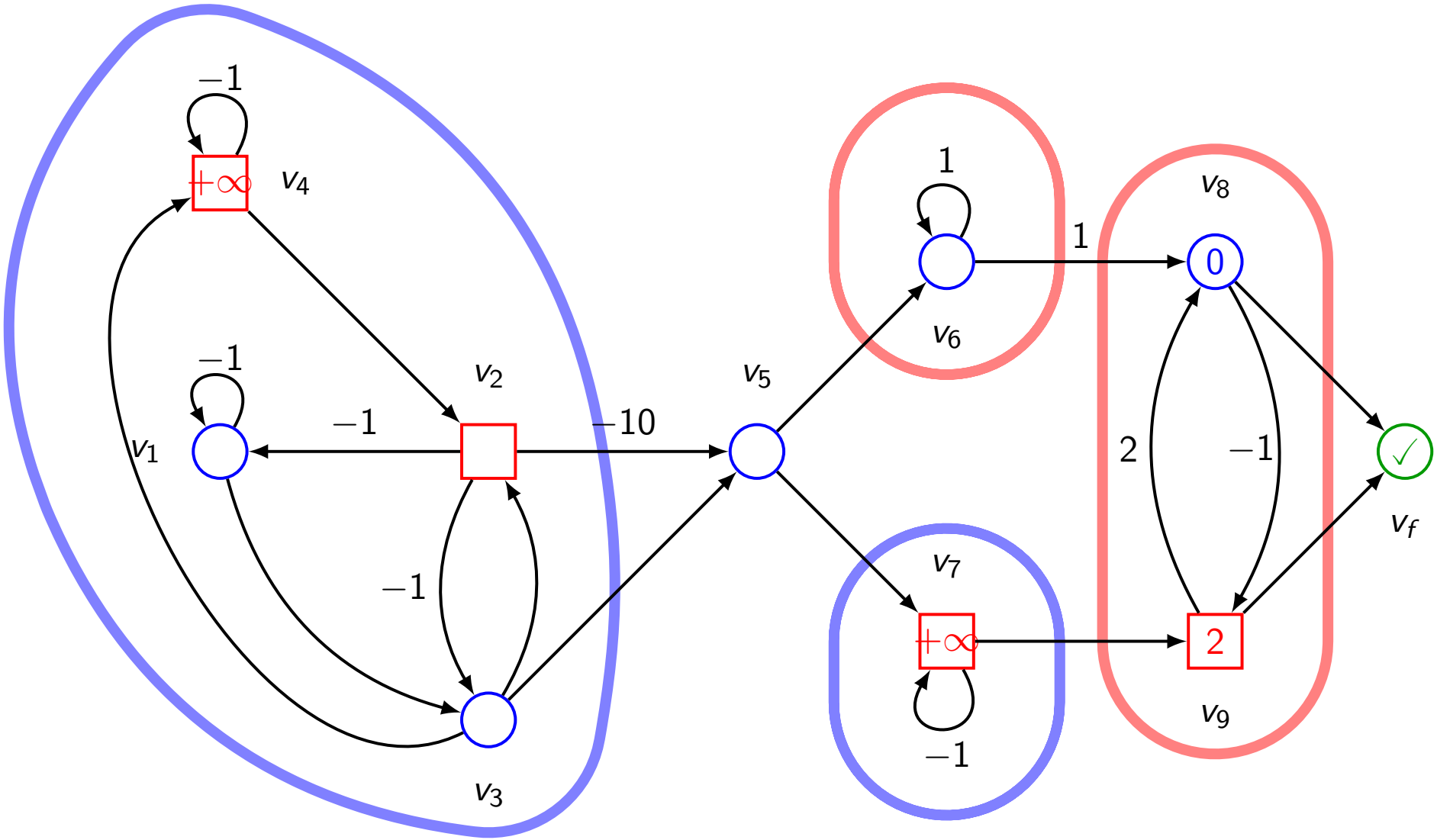
Min = ○, Max = □

Example



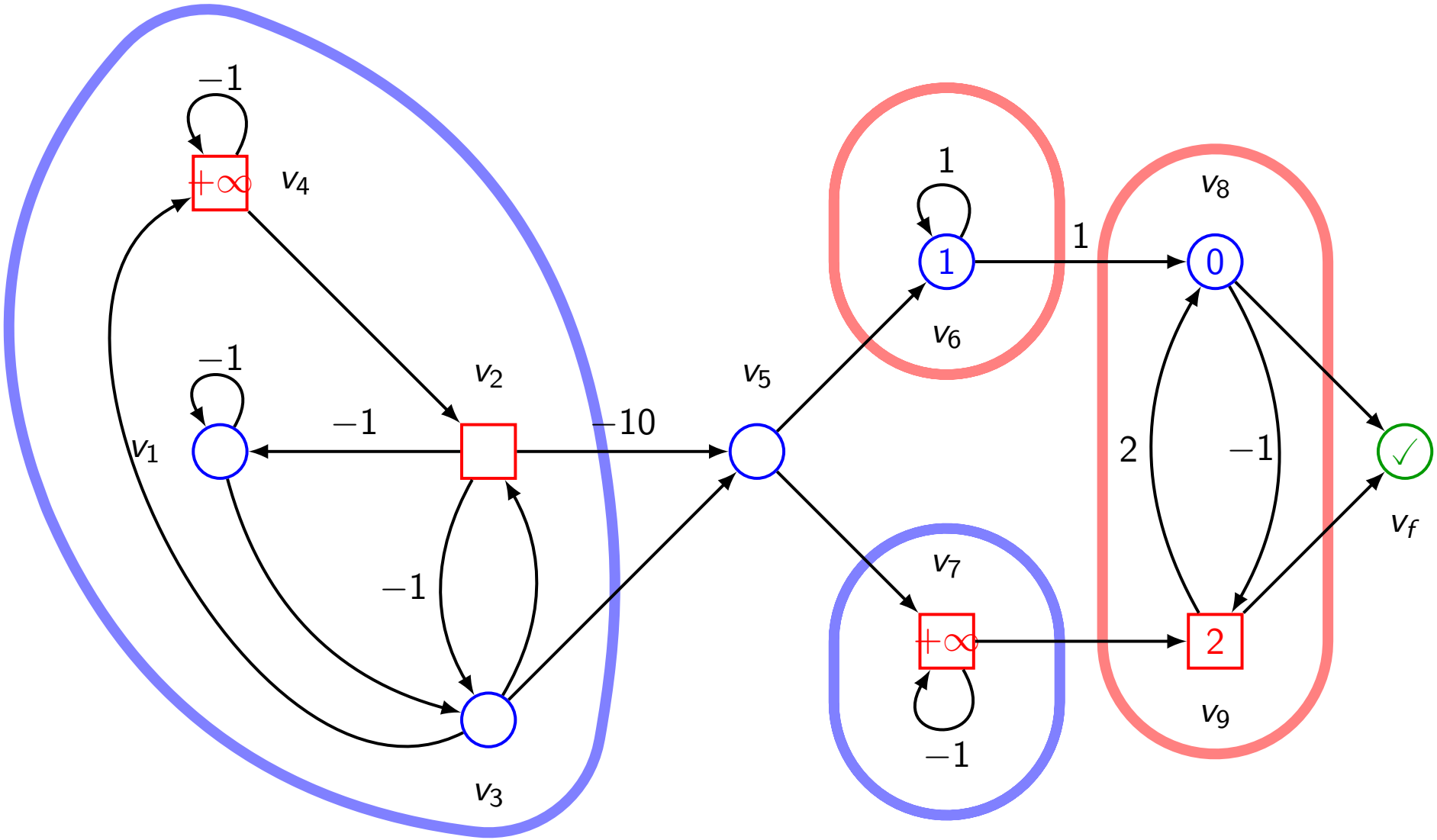
Min = ○, Max = □

Example



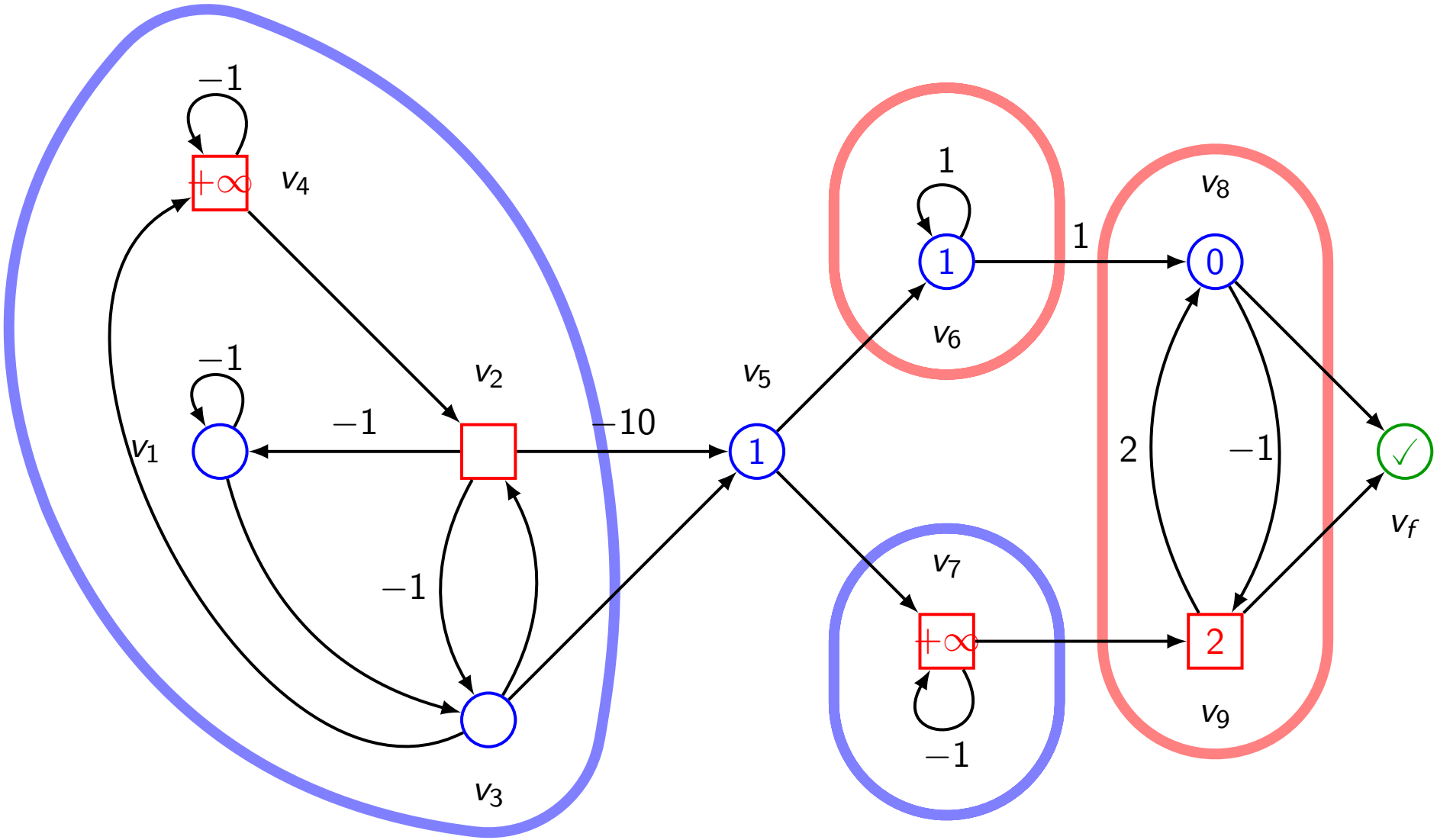
Min = ○, Max = □

Example



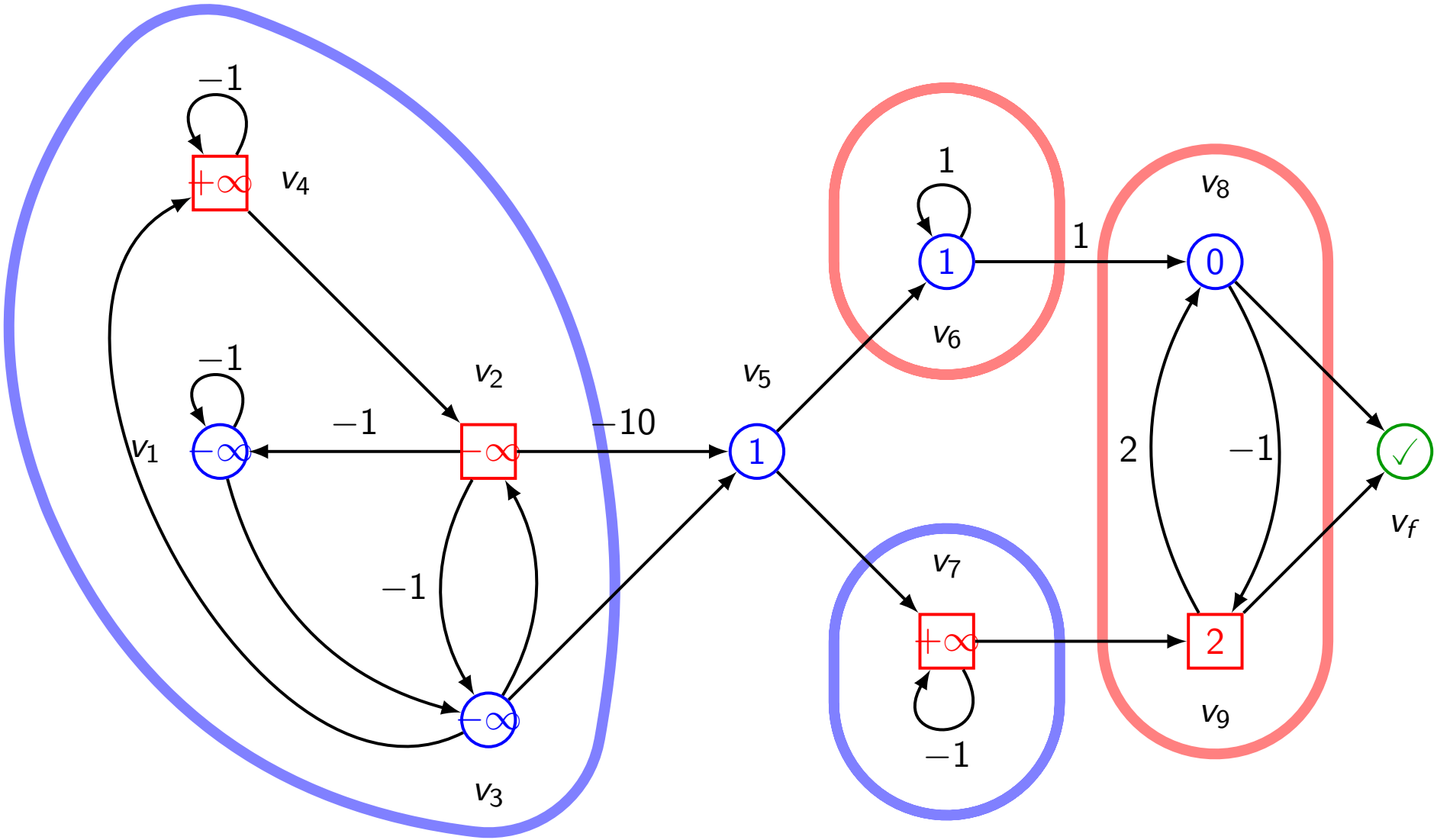
Min = ○, Max = □

Example



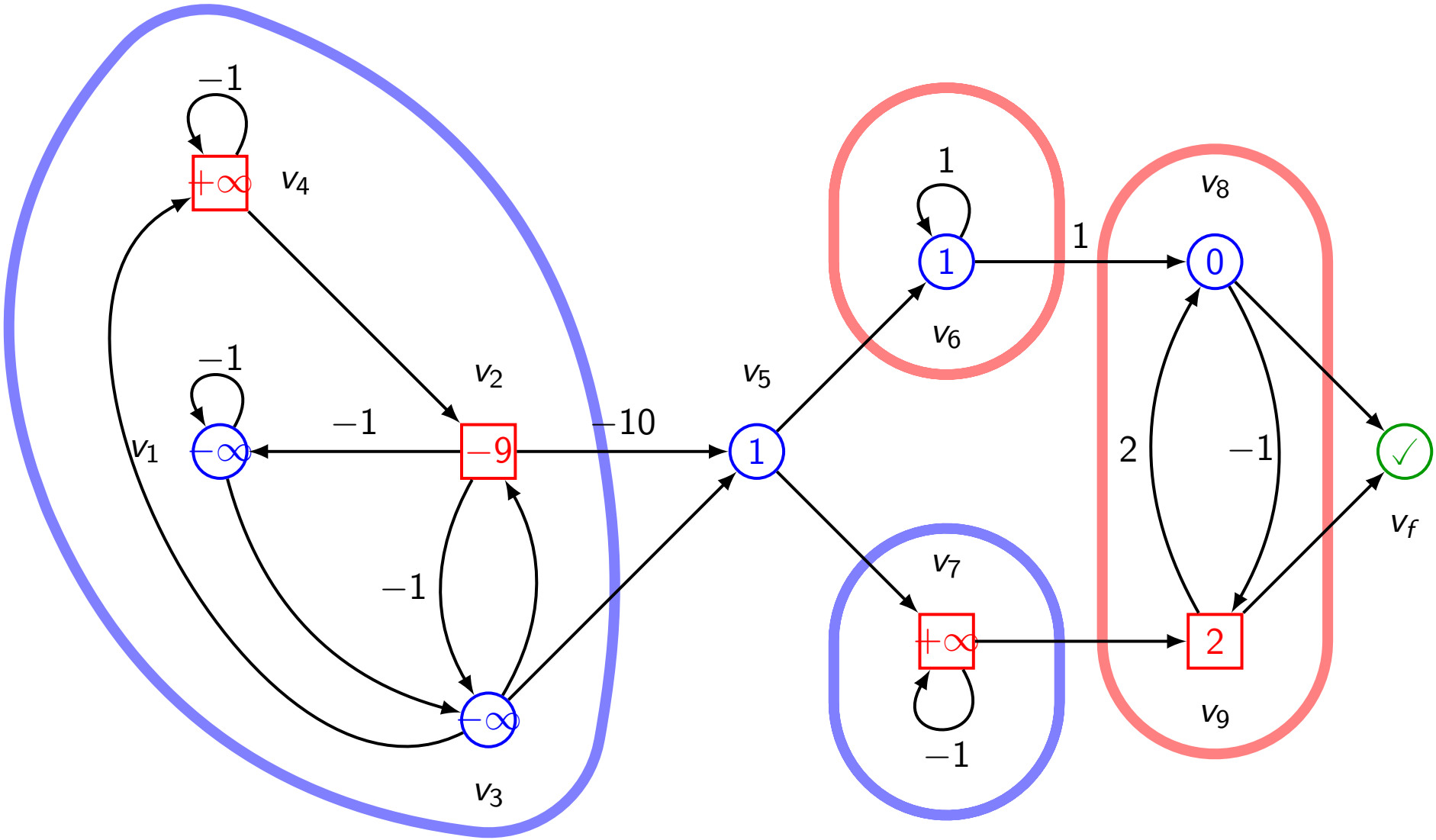
Min = ○, Max = □

Example



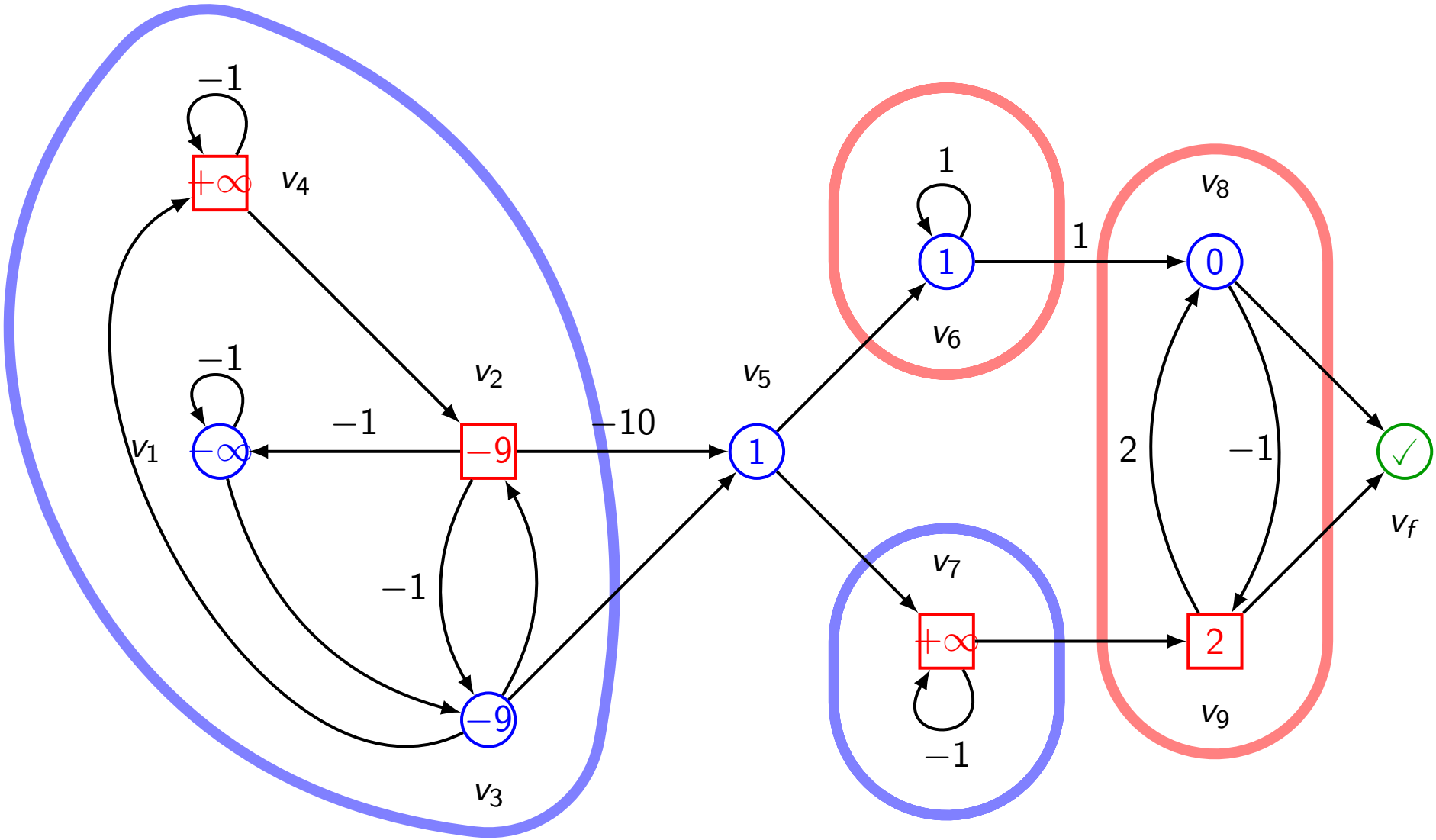
Min = ○, Max = □

Example



Min = ○, Max = □

Example



Min = ○, Max = □

Contribution 2: Divergent weighted timed games

Property of the underlying timed automaton:

Divergence

Every execution following a cycle of the region automaton has a total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no execution following a cycle has weight in $(-1, 1)$

Contribution 2: Divergent weighted timed games

Property of the underlying timed automaton:

Divergence

Every execution following a cycle of the region automaton has a total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no execution following a cycle has weight in $(-1, 1)$

Theorem:

The value problem on divergent weighted timed games is in 2-**EXP**, and is **EXP**-hard.

Contribution 2: Divergent weighted timed games

Property of the underlying timed automaton:

Divergence

Every execution following a cycle of the region automaton has a total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no execution following a cycle has weight in $(-1, 1)$

Theorem:

The value problem on divergent weighted timed games is in 2-**EXP**, and is **EXP**-hard.

Theorem:

Deciding if a weighted timed game is divergent is **PSPACE**-complete

Contribution 2: Divergent weighted timed games

Property of the underlying timed automaton:

Divergence

Every execution following a cycle of the region automaton has a total weight either $\leqslant -1$ or $\geqslant 1$, i.e., no execution following a cycle has weight in $(-1, 1)$

Theorem:

The value problem on divergent weighted timed games is in 2-**EXP**, and is **EXP**-hard.

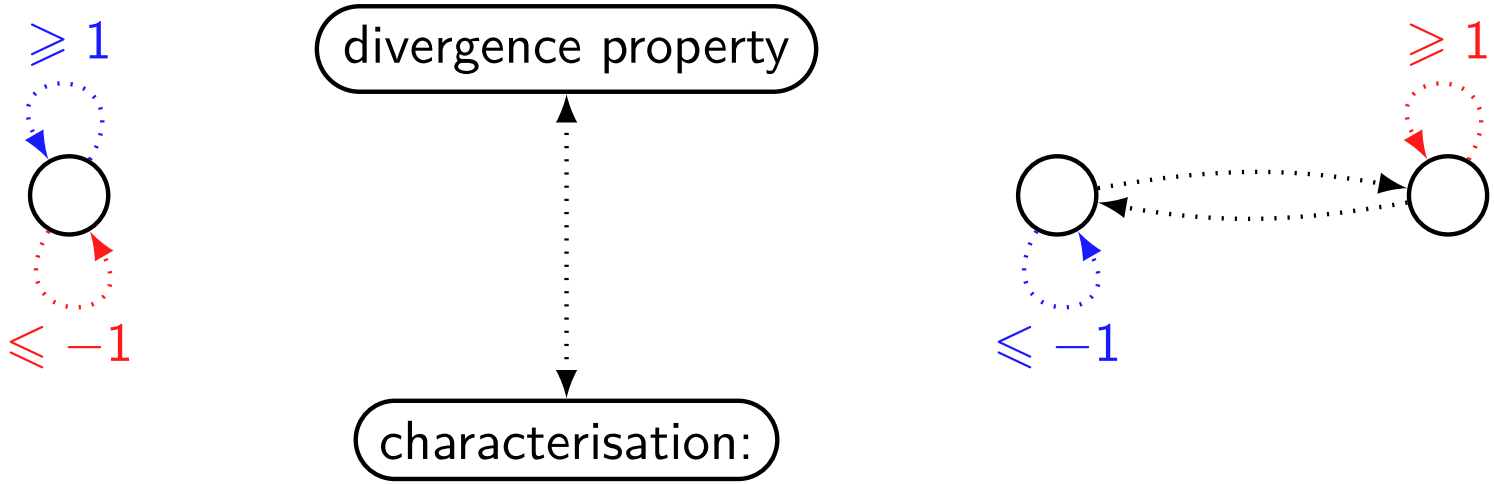
Theorem:

Deciding if a weighted timed game is divergent is **PSPACE**-complete

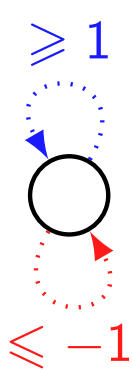
Observation

The set of weights of executions following a fixed region automaton cycle is an interval

Divergent WTG analysis: use of corner-point abstraction

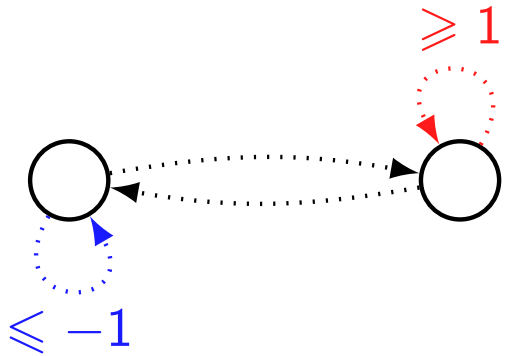


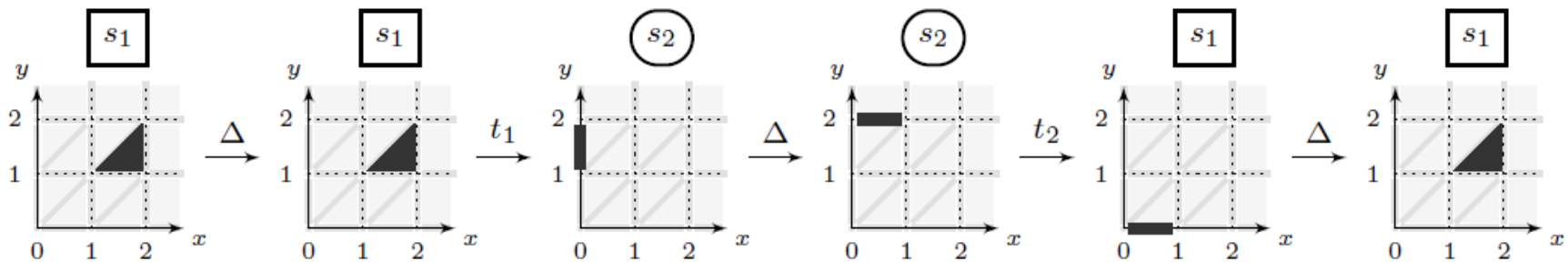
Divergent WTG analysis: use of corner-point abstraction

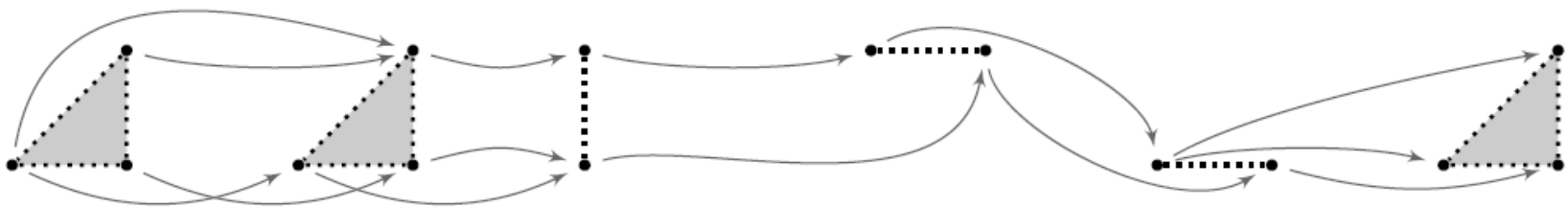


divergence property

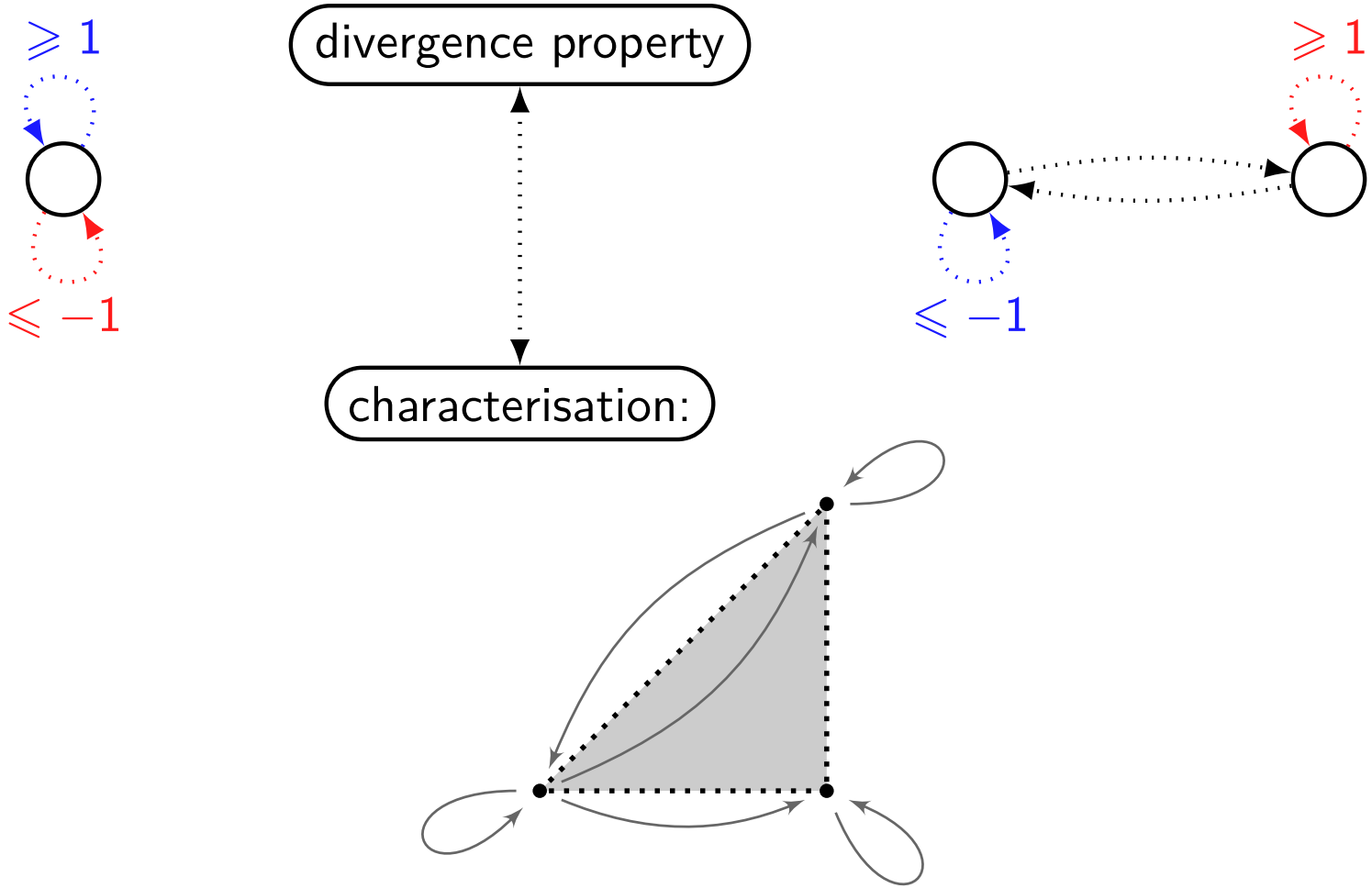
characterisation:



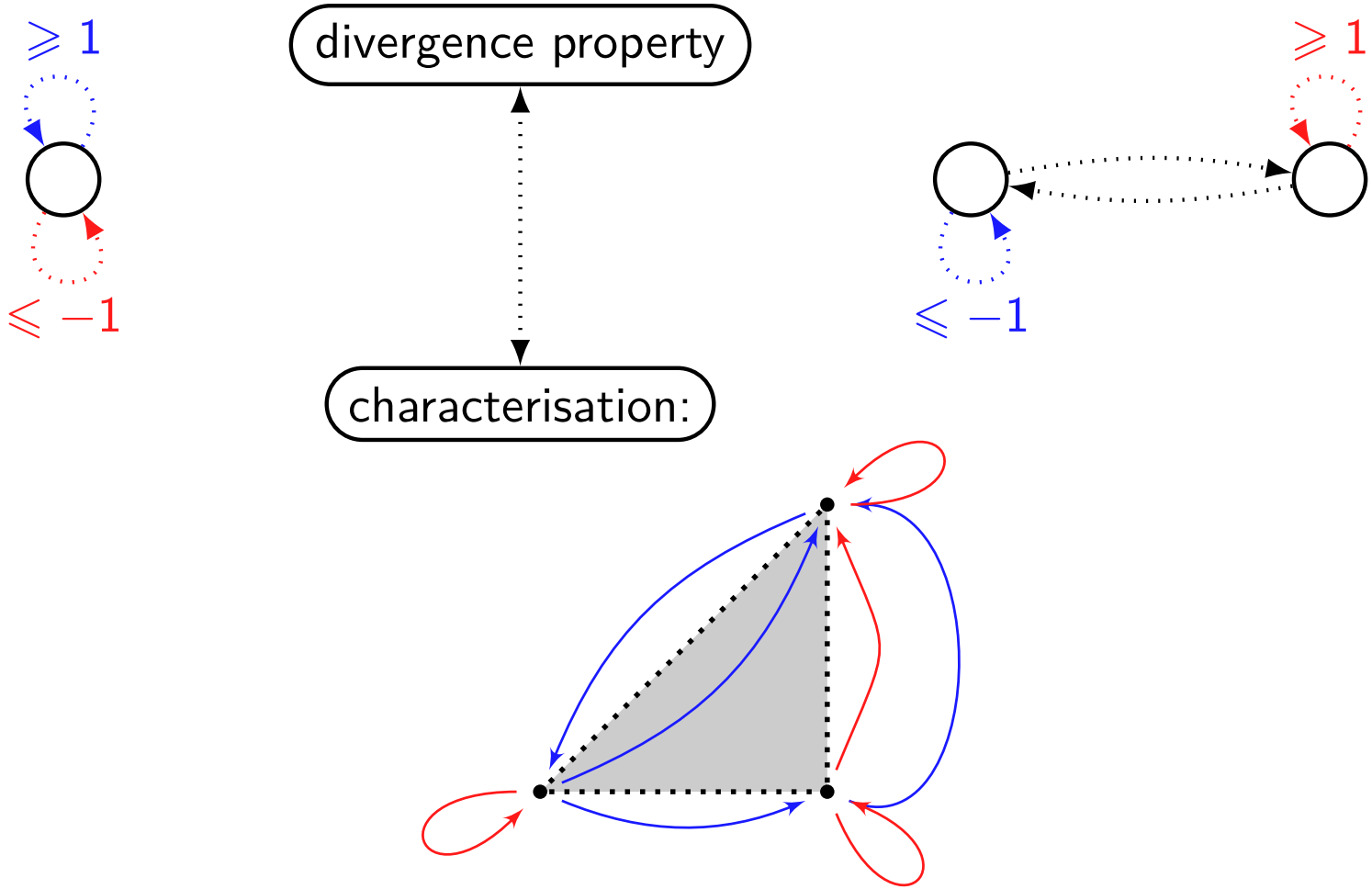




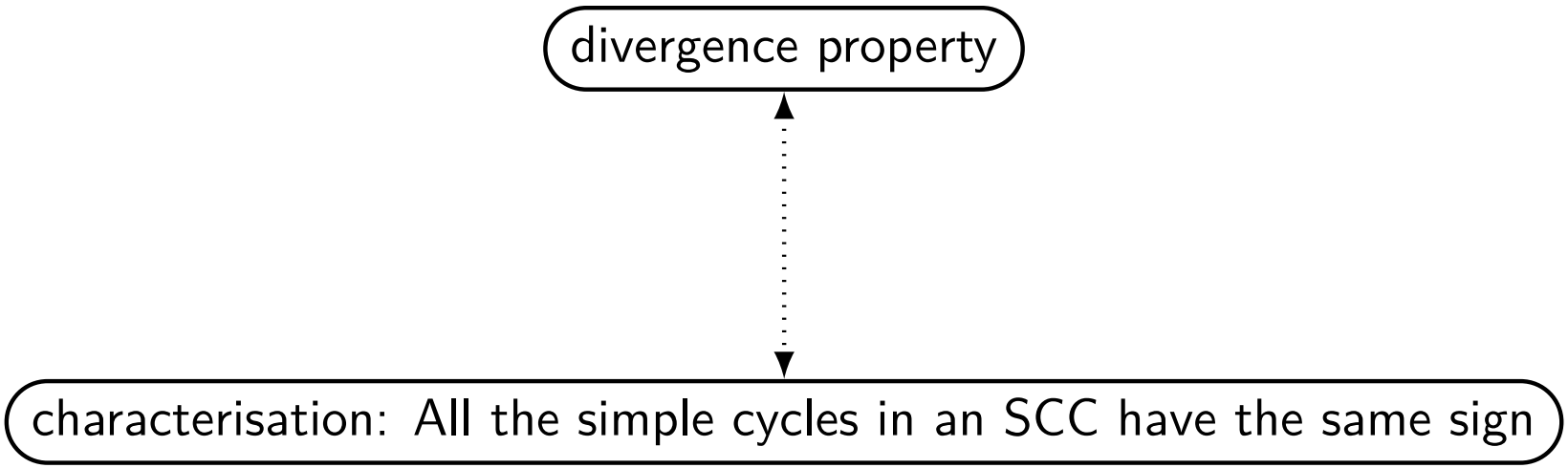
Divergent WTG analysis: use of corner-point abstraction



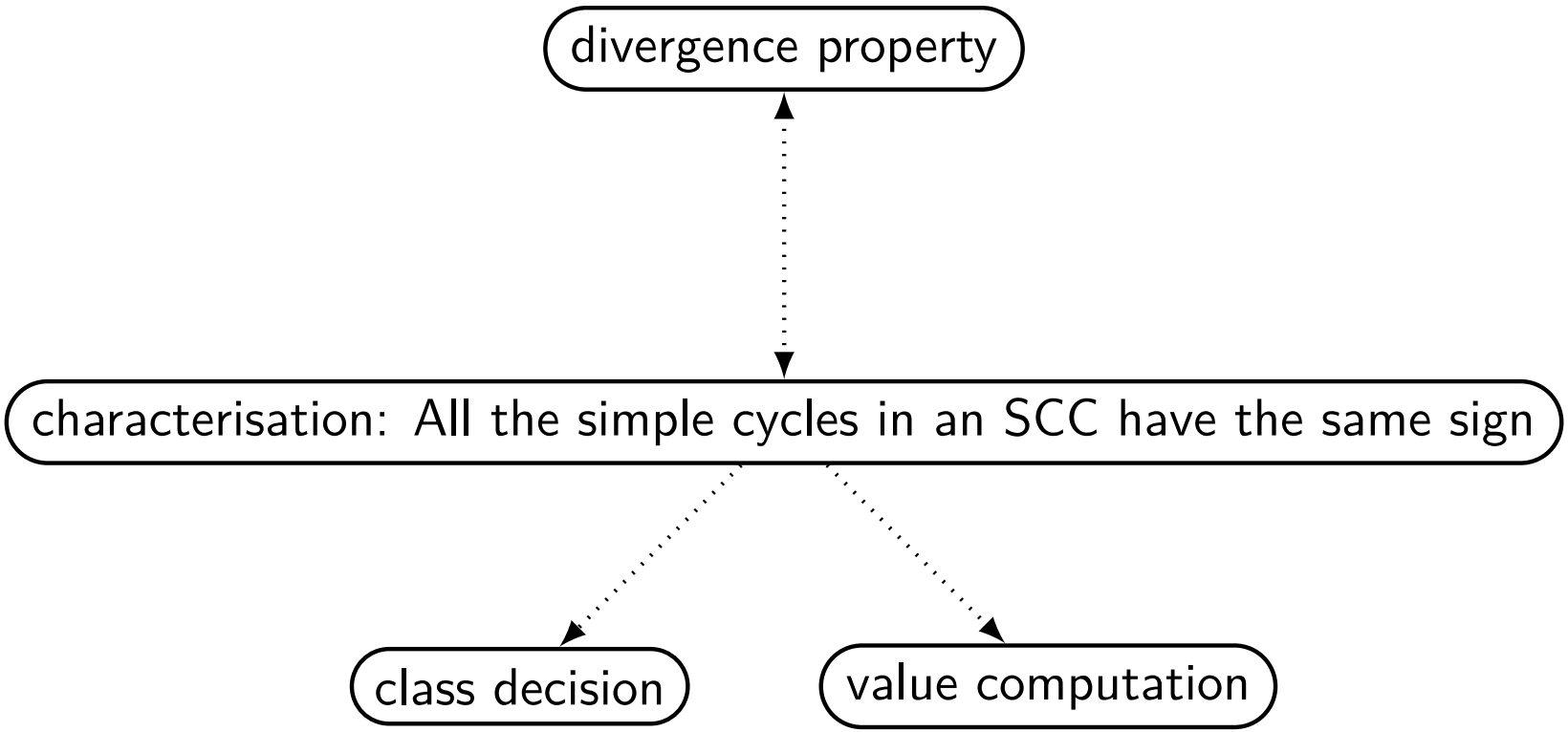
Divergent WTG analysis: use of corner-point abstraction



Divergent WTG analysis: use of corner-point abstraction



Divergent WTG analysis: use of corner-point abstraction



Value computation in divergent weighted timed games

- Remove $+\infty$ states
- SCC decomposition
- Value computation SCC after SCC, bottom-up

Value computation in divergent weighted timed games

- Remove $+\infty$ states
- SCC decomposition
- Value computation SCC after SCC, bottom-up

positive SCC

- Weighted timed games with **non-negative weights and strictly non-Zeno-cost cycles** (Bouyer, Cassez, Fleury, and Larsen 2004; Alur, Bernadsky, and Madhusudan 2004)
- Adaptation of iterative algorithm (in the presence of negative weights too) still converges in a number of steps linear in the size of the region automaton

Value computation in divergent weighted timed games

- Remove $+\infty$ states
- SCC decomposition
- Value computation SCC after SCC, bottom-up

positive SCC

- Weighted timed games with **non-negative weights and strictly non-Zeno-cost cycles** (Bouyer, Cassez, Fleury, and Larsen 2004; Alur, Bernadsky, and Madhusudan 2004)
- Adaptation of iterative algorithm (in the presence of negative weights too) still converges in a number of steps linear in the size of the region automaton

negative SCC

- Outside of the attractor of player $\square$ toward $\checkmark$ $\Rightarrow -\infty$
- The iterative algorithm converges on the other states in a number of steps linear in the size of the region automaton (initialisation with $-\infty$)

Conclusion

Value computation in weighted games:

- general case: **pseudo-polynomial** algorithm
- divergent weighted games: **P**-complete

Deciding if a weighted game is divergent: in **P**

Conclusion

Value computation in weighted games:

- general case: **pseudo-polynomial** algorithm
- divergent weighted games: **P**-complete

Deciding if a weighted game is divergent: in **P**

Value computation in weighted timed games:

- general case: value problem **undecidable**
- divergent weighted timed games: **2-exponential** algorithm

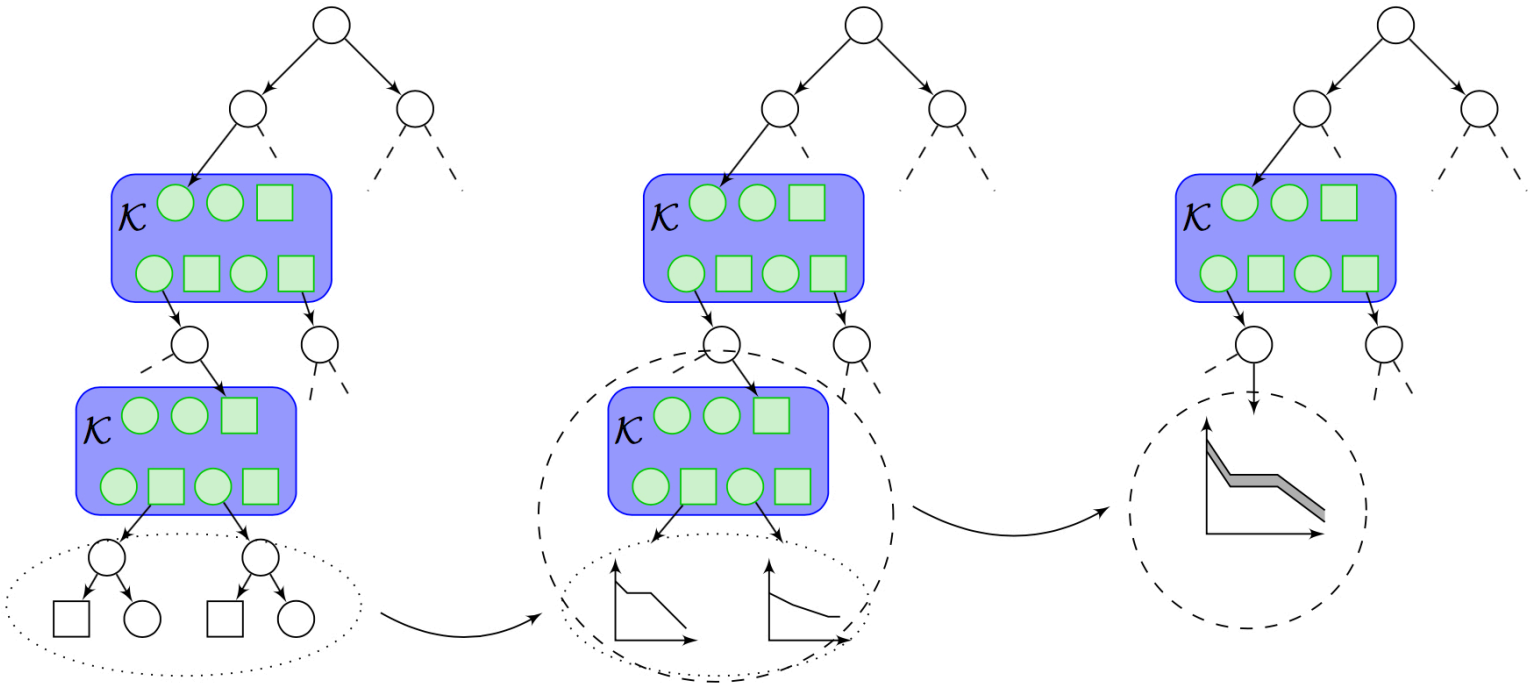
Deciding if a weighted timed game is divergent: **PSPACE**-complete

What about cycles of weight $= 0$?

- Adding cycles of weight $= 0$ to divergent WTG $\Longrightarrow$ **Undecidable!**

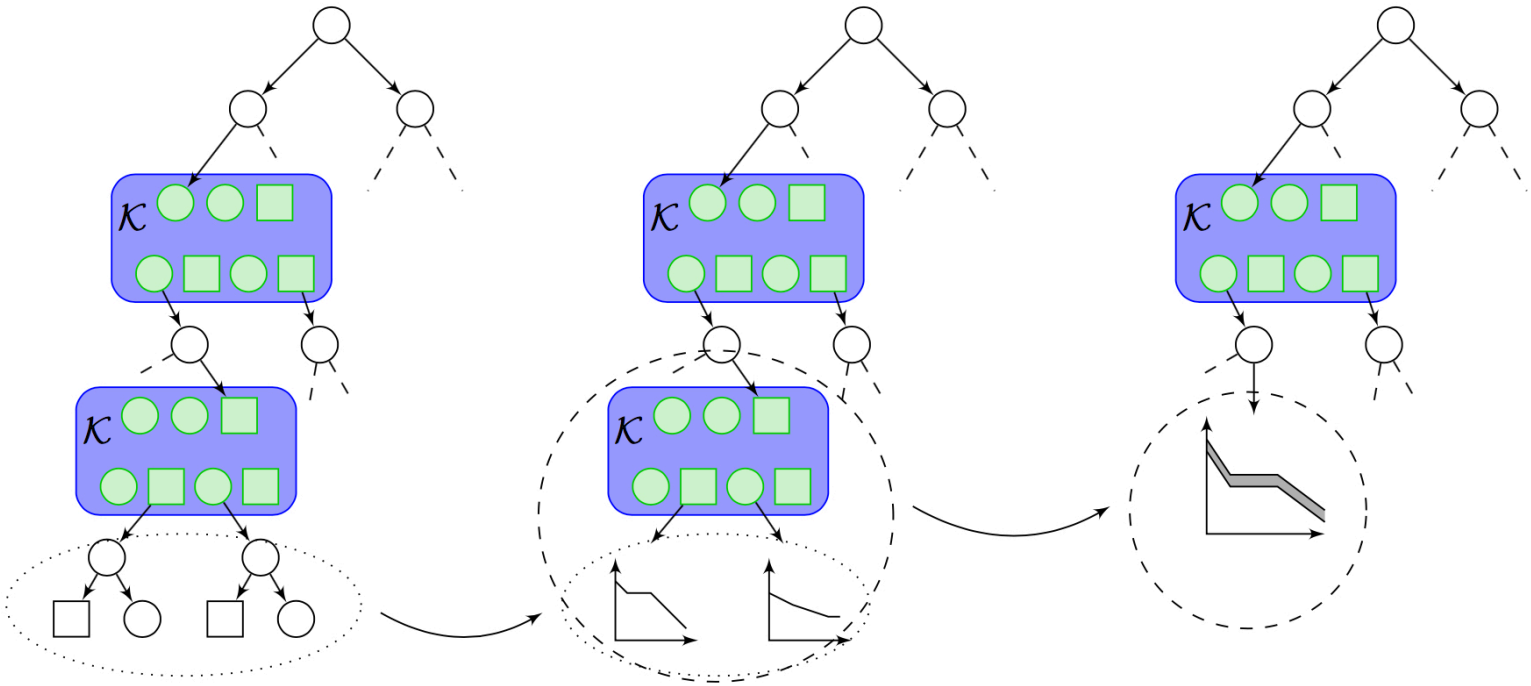
What about cycles of weight $= 0$?

- Adding cycles of weight $= 0$ to divergent WTG $\Longrightarrow$ **Undecidable!**
- Already with only non-negative weights (Bouyer, Jaziri, and Markey 2015): but possible to approximate the value...



What about cycles of weight $= 0$?

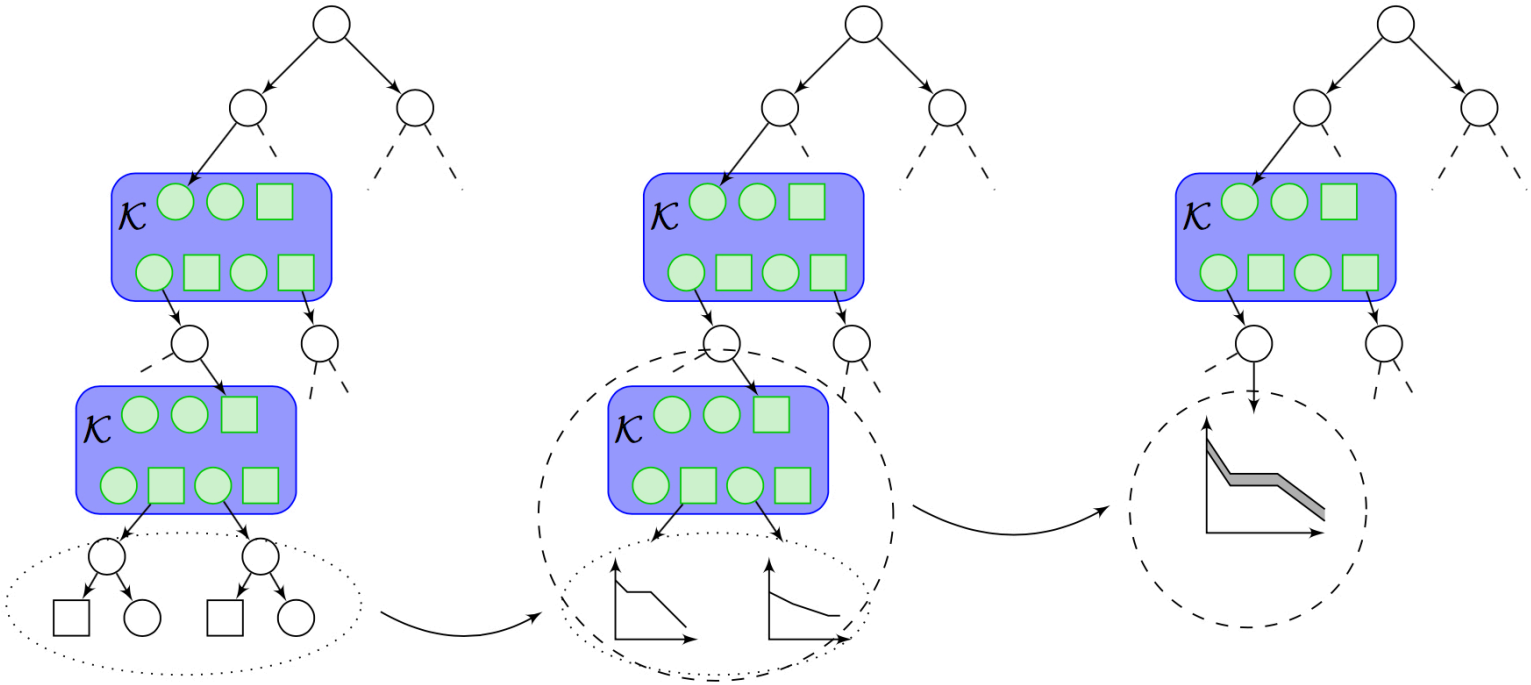
- Adding cycles of weight $= 0$ to divergent WTG $\Longrightarrow$ **Undecidable!**
- Already with only non-negative weights (Bouyer, Jaziri, and Markey 2015): but possible to approximate the value...



- Extension in the presence of negative weights?

What about cycles of weight $= 0$?

- Adding cycles of weight $= 0$ to divergent WTG $\Longrightarrow$ **Undecidable!**
- Already with only non-negative weights (Bouyer, Jaziri, and Markey 2015): but possible to approximate the value...

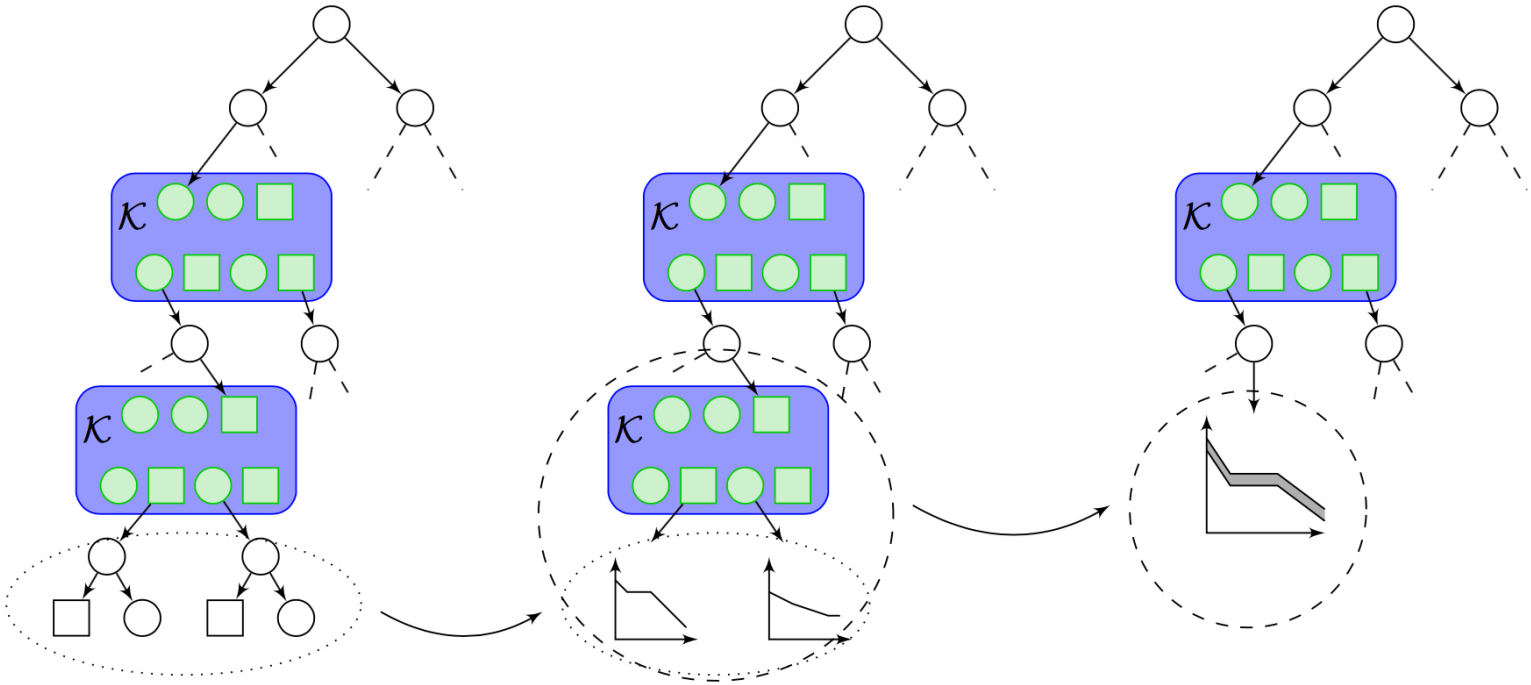


- Extension in the presence of negative weights?
- Almost-divergent WTG: every SCC of the region automaton is

$$\text{either } (\geqslant 1 \text{ or } = 0), \qquad \text{or } (\leqslant -1 \text{ or } = 0)$$

What about cycles of weight $= 0$?

- Adding cycles of weight $= 0$ to divergent WTG $\Longrightarrow$ **Undecidable!**
- Already with only non-negative weights (Bouyer, Jaziri, and Markey 2015): but possible to approximate the value...



- Extension in the presence of negative weights?
- Almost-divergent WTG: every SCC of the region automaton is

$$\text{either } (\geqslant 1 \text{ or } = 0), \qquad \text{or } (\leqslant -1 \text{ or } = 0)$$

New result

Approximation is decidable for almost-divergent WTG.

Thank you!

References I


Alur, Rajeev, Mikhail Bernadsky, and P. Madhusudan (2004). "Optimal Reachability for Weighted Timed Games". In: *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*. Vol. 3142. LNCS. Springer, pp. 122–133.


Bouyer, Patricia, Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin (2007). "On the Optimal Reachability Problem of Weighted Timed Automata". In: *Formal Methods in System Design* 31.2, pp. 135–175.


Bouyer, Patricia, Thomas Brihaye, and Nicolas Markey (2006). "Improved Undecidability Results on Weighted Timed Automata". In: *Information Processing Letters* 98.5, pp. 188–194.


Bouyer, Patricia, Franck Cassez, Emmanuel Fleury, and Kim G. Larsen (2004). "Optimal Strategies in Priced Timed Game Automata". In: *Proceedings of the 24th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'04)*. Vol. 3328. LNCS. Springer, pp. 148–160.

References II


Bouyer, Patricia, Samy Jaziri, and Nicolas Markey (2015). "On the Value Problem in Weighted Timed Games". In: *Proceedings of the 26th International Conference on Concurrency Theory (CONCUR'15)*. Vol. 42. Leibniz International Proceedings in Informatics. Leibniz-Zentrum für Informatik, pp. 311–324. DOI: 10.4230/LIPIcs.CONCUR.2015.311.


Bouyer, Patricia, Kim G. Larsen, Nicolas Markey, and Jacob Illum Rasmussen (2006). "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)*. Vol. 4337. LNCS. Springer, pp. 345–356.


Brihaye, Thomas, Véronique Bruyère, and Jean-François Raskin (2005). "On Optimal Timed Strategies". In: *Proceedings of the Third international conference on Formal Modeling and Analysis of Timed Systems (FORMATS'05)*. Vol. 3829. LNCS. Springer, pp. 49–64.

References III


Brihaye, Thomas, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege (2016). "Pseudopolynomial Iterative Algorithm to Solve Total-Payoff Games and Min-Cost Reachability Games". In: *Acta Informatica*. DOI: 10.1007/s00236-016-0276-z.


Brihaye, Thomas, Gilles Geeraerts, Shankara Narayanan Krishna, Lakshmi Manasa, Benjamin Monmege, and Ashutosh Trivedi (2014). "Adding Negative Prices to Priced Timed Games". In: *Proceedings of the 25th International Conference on Concurrency Theory (CONCUR'14)*. Vol. 8704. Springer, pp. 560–575. DOI: 10.1007/978-3-662-44584-6_38.


Fearnley, John and Marcin Jurdziński (2013). "Reachability in Two-Clock Timed Automata Is PSPACE-Complete". In: *Proceedings of the 40th international conference on Automata, Languages, and Programming (ICALP'13)*. Vol. 7966. LNCS. Springer, pp. 212–223.

References IV


Khachiyan, Leonid, Endre Boros, Konrad Borys, Khaled Elbassioni, Vladimir Gurvich, Gabor Rudolf, et al. (2008). "On Short Paths Interdiction Problems: Total and Node-Wise Limited Interdiction". In: *Theory of Computing Systems* 43.2, pp. 204–233. DOI: 10.1007/s00224-007-9025-6.