Meet Your Expectations With Guarantees: Beyond Worst-Case Synthesis in Quantitative Games

V. Bruyère (UMONS) E. Filiot (ULB)
M. Randour (UMONS-ULB) J.-F. Raskin (ULB)

Brussels - 19.11.2013

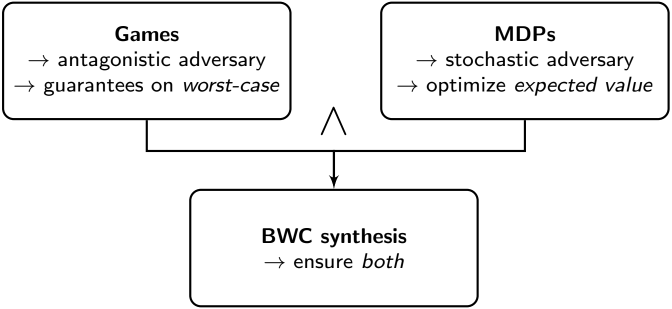
MF&V seminar

UMONS Université de Mons
ULB Université Libre de Bruxelles

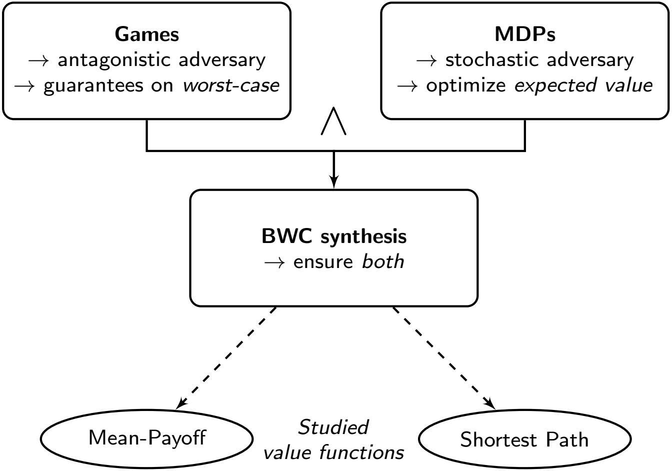
The talk in one slide

Games	**MDPs**
$\rightarrow$ antagonistic adversary	$\rightarrow$ stochastic adversary
$\rightarrow$ guarantees on *worst-case*	$\rightarrow$ optimize *expected value*

The talk in one slide

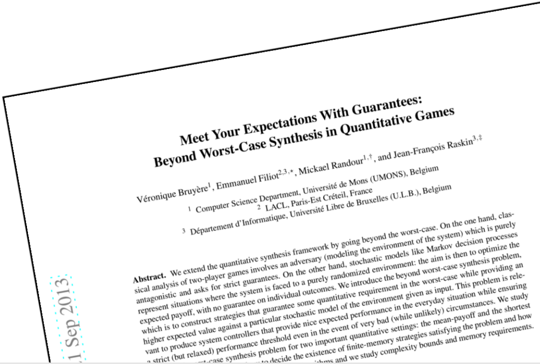


The talk in one slide



Advertisement

Full paper available on arXiv: abs/1309.5439



Meet Your Expectations With Guarantees: Beyond Worst-Case Synthesis in Quantitative Games

Véronique Bruyère[1], Emmanuel Filiot[2,3,*], Mickael Randour[1,†], and Jean-François Raskin[3,‡]

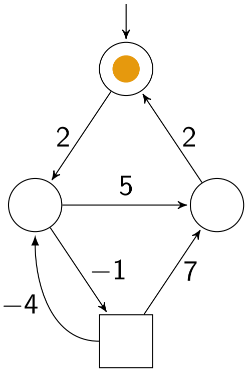
[1] Computer Science Department, Université de Mons (UMONS), Belgium
[2] LACL, Paris-Est Créteil, France
[3] Département d'Informatique, Université Libre de Bruxelles (U.L.B.), Belgium

Abstract. We extend the quantitative synthesis framework by going beyond the worst-case. On the one hand, classical analysis of two-player games involves an adversary (modeling the environment of the system) which is purely antagonistic and asks for strict guarantees. On the other hand, stochastic models like Markov decision processes represent situations where the system is faced to a purely randomized environment: the aim is then to optimize the expected payoff, with no guarantee on individual outcomes. We introduce the beyond worst-case synthesis problem, which is to construct strategies that guarantee some quantitative requirement in the worst-case while providing an higher expected value against a particular stochastic model of the environment given as input. This problem is relevant to produce system controllers that provide nice expected performance in the everyday situation while ensuring a strict (but relaxed) performance threshold even in the event of very bad (while unlikely) circumstances. We study the beyond worst-case synthesis problem for two important quantitative settings: the mean-payoff and the shortest ... to decide the existence of finite-memory strategies satisfying the problem and how ... and we study complexity bounds and memory requirements.

Sep 2013

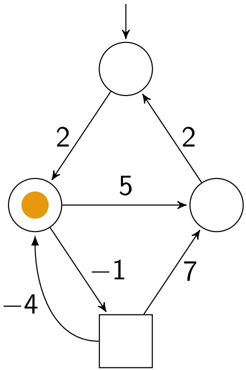
1. Context
2. BWC Synthesis
3. Mean-Payoff
4. Shortest Path
5. Conclusion

Quantitative games on graphs



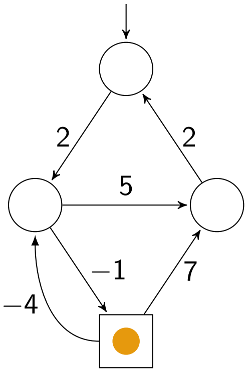
- Graph $\mathcal{G} = (S, E, w)$ with $w \colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states = ◯
  - ▷ $\mathcal{P}_2$ states = □
- Plays have values
  - ▷ $f \colon \text{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i \colon \text{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory ⇒ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\text{Mem}, \text{m}_0, \alpha_\text{u}, \alpha_\text{n})$

Quantitative games on graphs



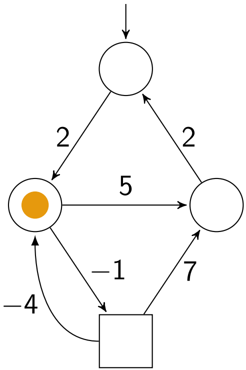
- Graph $\mathcal{G} = (S, E, w)$ with $w \colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states $= \bigcirc$
  - ▷ $\mathcal{P}_2$ states $= \square$
- Plays have values
  - ▷ $f \colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i \colon \mathsf{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory $\Rightarrow$ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_\mathsf{u}, \alpha_\mathsf{n})$

Quantitative games on graphs



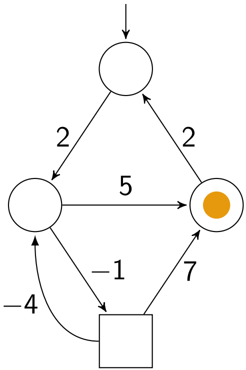
- Graph $\mathcal{G} = (S, E, w)$ with $w\colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states = ○
  - ▷ $\mathcal{P}_2$ states = □
- Plays have values
  - ▷ $f\colon \text{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i\colon \text{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory ⇒ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\text{Mem}, \text{m}_0, \alpha_\text{u}, \alpha_\text{n})$

Quantitative games on graphs



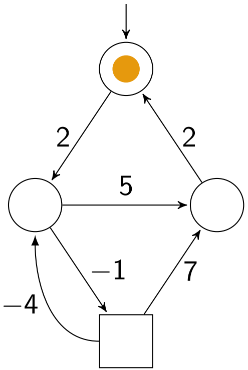
- Graph $\mathcal{G} = (S, E, w)$ with $w\colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states $= \bigcirc$
  - ▷ $\mathcal{P}_2$ states $= \square$
- Plays have values
  - ▷ $f\colon \text{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i\colon \text{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory $\Rightarrow$ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\text{Mem}, \text{m}_0, \alpha_\text{u}, \alpha_\text{n})$

Quantitative games on graphs



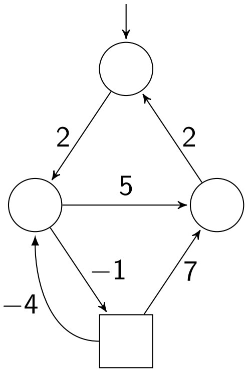
- Graph $\mathcal{G} = (S, E, w)$ with $w\colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states = ◯
  - ▷ $\mathcal{P}_2$ states = □
- Plays have values
  - ▷ $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i\colon \mathsf{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory ⇒ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_{\mathsf{u}}, \alpha_{\mathsf{n}})$

Quantitative games on graphs



- Graph $\mathcal{G} = (S, E, w)$ with $w \colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states $= \bigcirc$
  - ▷ $\mathcal{P}_2$ states $= \square$
- Plays have values
  - ▷ $f \colon \text{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i \colon \text{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory ⇒ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\text{Mem}, \text{m}_0, \alpha_\text{u}, \alpha_\text{n})$

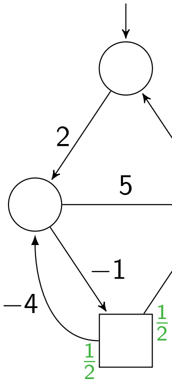
Quantitative games on graphs



Then, $(2, 5, 2)^\omega$

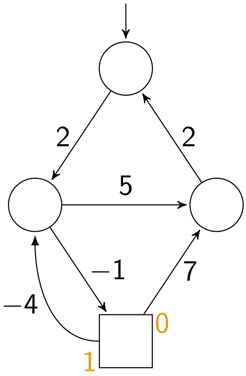
- Graph $\mathcal{G} = (S, E, w)$ with $w \colon E \to \mathbb{Z}$
- Two-player *game* $G = (\mathcal{G}, S_1, S_2)$
  - ▷ $\mathcal{P}_1$ states = ◯
  - ▷ $\mathcal{P}_2$ states = □
- Plays have values
  - ▷ $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
- Players follow *strategies*
  - ▷ $\lambda_i \colon \mathsf{Prefs}_i(G) \to \mathcal{D}(S)$
  - ▷ Finite memory ⇒ stochastic Moore machine $\mathcal{M}(\lambda_i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_\mathsf{u}, \alpha_\mathsf{n})$

Markov decision processes



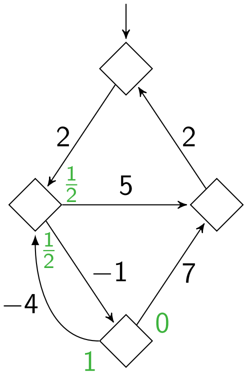
- MDP $P = (\mathcal{G}, S_1, S_\Delta, \Delta)$ with $\Delta\colon S_\Delta \to \mathcal{D}(S)$
  - ▷ $\mathcal{P}_1$ states = ◯
  - ▷ stochastic states = □
- MDP = game + strategy of $\mathcal{P}_2$
  - ▷ $P = G[\lambda_2]$

Markov decision processes



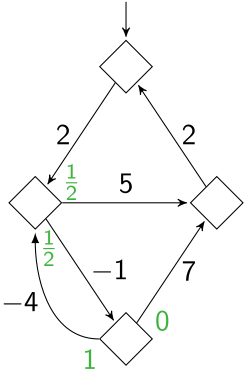
- MDP $P = (\mathcal{G}, S_1, S_\Delta, \Delta)$ with $\Delta \colon S_\Delta \to \mathcal{D}(S)$
  - $\mathcal{P}_1$ states $= \bigcirc$
  - stochastic states $= \square$
- MDP $=$ game $+$ strategy of $\mathcal{P}_2$
  - $P = G[\lambda_2]$
- **Important**: we allow $E \setminus E_\Delta \neq \emptyset$, $E_\Delta = \{(s_1, s_2) \in E \mid s_1 \in S_\Delta \Rightarrow \Delta(s_1)(s_2) > 0\}$

Markov chains



- MC $M = (\mathcal{G}, \delta)$ with $\delta \colon S \rightarrow \mathcal{D}(S)$
- MC = MDP + strategy of $\mathcal{P}_1$
  = game + both strategies
  - ▷ $M = P[\lambda_1] = G[\lambda_1, \lambda_2]$

Markov chains



- MC $M = (\mathcal{G}, \delta)$ with $\delta \colon S \to \mathcal{D}(S)$
- MC = MDP + strategy of $\mathcal{P}_1$
  = game + both strategies
  - ▷ $M = P[\lambda_1] = G[\lambda_1, \lambda_2]$
- Event $\mathcal{A} \subseteq \mathsf{Plays}(\mathcal{G})$
  - ▷ probability $\mathbb{P}^M_{s_{\mathsf{init}}}(\mathcal{A})$
- Measurable $f \colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$
  - ▷ *expected value* $\mathbb{E}^M_{s_{\mathsf{init}}}(f)$

Classical interpretations

- **System** trying to ensure a specification $= \mathcal{P}_1$
  - ▷ whatever the actions of its **environment**

Classical interpretations

- **System** trying to ensure a specification $= \mathcal{P}_1$
  - ▷ whatever the actions of its **environment**
- The environment can be seen as
  - ▷ *antagonistic*
    - two-player game, *worst-case* threshold problem for $\mu \in \mathbb{Q}$
    - $\exists ?\, \lambda_1 \in \Lambda_1, \forall\, \lambda_2 \in \Lambda_2, \forall\, \pi \in \mathsf{Outs}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2),\ f(\pi) \geq \mu$

Classical interpretations

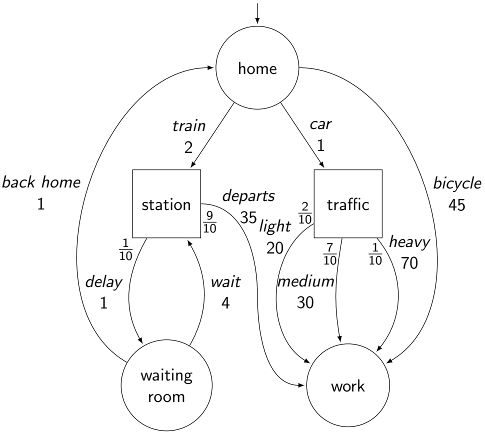
- **System** trying to ensure a specification $= \mathcal{P}_1$
  - ▷ whatever the actions of its **environment**
- The environment can be seen as
  - ▷ *antagonistic*
    - two-player game, *worst-case* threshold problem for $\mu \in \mathbb{Q}$
    - $\exists?\, \lambda_1 \in \Lambda_1, \forall\, \lambda_2 \in \Lambda_2, \forall\, \pi \in \mathsf{Outs}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2),\ f(\pi) \geq \mu$
  - ▷ *fully stochastic*
    - MDP, *expected value* threshold problem for $\nu \in \mathbb{Q}$
    - $\exists?\, \lambda_1 \in \Lambda_1,\ \mathbb{E}^{P[\lambda_1]}_{s_{\mathsf{init}}}(f) \geq \nu$

What if you want both?

In practice, we want both

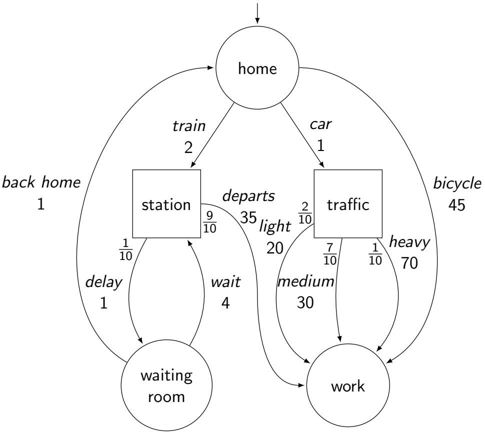
1. nice expected performance in the everyday situation,
2. strict (but relaxed) performance guarantees even in the event of very bad circumstances.

Example: going to work



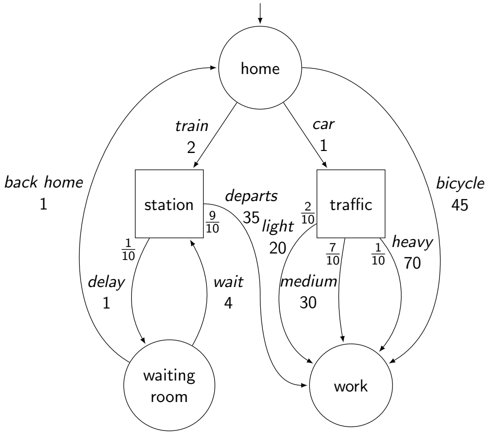
- ▷ Weights = minutes
- ▷ Goal: *minimize our expected time* to reach "work"
- ▷ **But**, important meeting in one hour! Requires *strict guarantees* on the worst-case reaching time.

Example: going to work



- ▷ Optimal expectation strategy: take the car.
  - $\mathbb{E} = 33$, $\mathrm{WC} = 71 > 60$.
- ▷ Optimal worst-case strategy: bicycle.
  - $\mathbb{E} = \mathrm{WC} = 45 < 60$.

Example: going to work



- ▷ Optimal expectation strategy: take the car.
  - $\mathbb{E} = 33$, $\mathrm{WC} = 71 > 60$.
- ▷ Optimal worst-case strategy: bicycle.
  - $\mathbb{E} = \mathrm{WC} = 45 < 60$.
- ▷ **Sample BWC strategy**: try train up to 3 delays then switch to bicycle.
  - $\mathbb{E} \approx 37.56$, $\mathrm{WC} = 59 < 60$.

Beyond worst-case synthesis

Formal definition

Given a game $G = (\mathcal{G}, S_1, S_2)$, with $\mathcal{G} = (S, E, w)$ its underlying graph, an initial state $s_{\text{init}} \in S$, a finite-memory stochastic model $\lambda_2^{\text{stoch}} \in \Lambda_2^F$ of the adversary, represented by a stochastic Moore machine, a measurable value function $f\colon \text{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$, and two rational thresholds $\mu, \nu \in \mathbb{Q}$, the *beyond worst-case (BWC) problem* asks to decide if $\mathcal{P}_1$ has a finite-memory strategy $\lambda_1 \in \Lambda_1^F$ such that

$$\begin{cases} \forall\, \lambda_2 \in \Lambda_2,\ \forall\, \pi \in \text{Outs}_G(s_{\text{init}}, \lambda_1, \lambda_2),\ f(\pi) > \mu & (1) \\ \mathbb{E}_{s_{\text{init}}}^{G[\lambda_1, \lambda_2^{\text{stoch}}]}(f) > \nu & (2) \end{cases}$$

and the *BWC synthesis problem* asks to synthesize such a strategy if one exists.

Beyond worst-case synthesis

Formal definition

Given a game $G = (\mathcal{G}, S_1, S_2)$, with $\mathcal{G} = (S, E, w)$ its underlying graph, an initial state $s_{\mathsf{init}} \in S$, a finite-memory stochastic model $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^F$ of the adversary, represented by a stochastic Moore machine, a measurable value function $f\colon \mathsf{Plays}(\mathcal{G}) \rightarrow \mathbb{R} \cup \{-\infty, \infty\}$, and two rational thresholds $\mu, \nu \in \mathbb{Q}$, the *beyond worst-case (BWC) problem* asks to decide if $\mathcal{P}_1$ has a finite-memory strategy $\lambda_1 \in \Lambda_1^F$ such that

$$\begin{cases} \forall\, \lambda_2 \in \Lambda_2, \forall\, \pi \in \mathsf{Outs}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2),\; f(\pi) > \mu & (1) \\ \mathbb{E}_{s_{\mathsf{init}}}^{G[\lambda_1, \lambda_2^{\mathsf{stoch}}]}(f) > \nu & (2) \end{cases}$$

and the *BWC synthesis problem* asks to synthesize such a strategy if one exists.

Notice the highlighted parts!

Mean-payoff value function

- $\mathsf{MP}(\pi) = \liminf_{n\to\infty} \left[\frac{1}{n} \cdot \sum_{i=0}^{i=n-1} w\big((s_i, s_{i+1})\big) \right]$

- Sample play $\pi = 2,\ -1,\ -4,\ 5,\ (2,\ 2,\ 5)^{\omega}$
  - ▷ $\mathsf{MP}(\pi) = 3 \leadsto$ *prefix-independent*

Mean-payoff value function

- $\mathrm{MP}(\pi) = \liminf_{n\to\infty} \left[\frac{1}{n} \cdot \sum_{i=0}^{i=n-1} w\big((s_i, s_{i+1})\big)\right]$
- Sample play $\pi = 2,\ -1,\ -4,\ 5,\ (2,\ 2,\ 5)^{\omega}$
  - ▷ $\mathrm{MP}(\pi) = 3 \rightsquigarrow$ *prefix-independent*

Games: worst-case threshold problem [LL69, EM79, ZP96, Jur98, GS09]

Memoryless optimal strategies exist for both players and the problem is in NP ∩ coNP.

MDPs: expected value threshold problem [Put94, FV97]

Memoryless optimal strategies exist and the problem is in P.

BWC MP problem: overview

Theorem (algorithm & complexity)

The BWC problem for the mean-payoff is in $\mathbf{NP} \cap \mathbf{coNP}$ *and at least as hard as deciding the winner in mean-payoff games.*

- ▷ Additional modeling power for free!

BWC MP problem: overview

Theorem (algorithm & complexity)

The BWC problem for the mean-payoff is in **NP ∩ coNP** *and at least as hard as deciding the winner in mean-payoff games.*

- ▷ Additional modeling power for free!

Theorem (memory bounds)

Memory of **pseudo-polynomial** *size may be necessary and is always sufficient to satisfy the BWC problem for the mean-payoff: polynomial in the size of the game and the stochastic model, and polynomial in the weight and threshold values.*

Algorithm: overview

Algorithm 1 BWC_MP($G^i, \lambda_2^i, \mu^i, \nu^i, s^i_{\mathsf{init}}$)

Require: $G^i = (\mathcal{G}^i, S^i_1, S^i_2)$ a game, $\mathcal{G}^i = (S^i, E^i, w^i)$ its underlying graph, $\lambda_2^i \in \Lambda_2^F(G^i)$ a finite-memory stochastic model of the adversary, $\mathcal{M}(\lambda_2^i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_\mathsf{u}, \alpha_\mathsf{n})$ its Moore machine, $\mu^i = \frac{a}{b}, \nu^i \in \mathbb{Q}$, $\mu^i < \nu^i$, resp. the worst-case and the expected value thresholds, and $s^i_{\mathsf{init}} \in S^i$ the initial state

Ensure: The answer is YES if and only if $\mathcal{P}_1$ has a finite-memory strategy $\lambda_1 \in \Lambda_1^F(G^i)$ satisfying the BWC problem from s^i_{init}, for the thresholds pair (μ^i, ν^i) and the mean-payoff value function

{*Preprocessing*}
if $\mu^i \neq 0$ **then**
  Modify the weight function of $\mathcal{G}^i$ s.t. $\forall e \in E^i$, $w^i_{\mathsf{new}}(e) := b \cdot w^i(e) - a$, and consider the new thresholds pair $(0, \nu := b \cdot \nu^i - a)$
Compute $S_{WC} := \{s \in S^i \mid \exists \lambda_1 \in \Lambda_1(G^i), \forall \lambda_2 \in \Lambda_2(G^i), \forall \pi \in \mathsf{Outs}_{G^i}(s, \lambda_1, \lambda_2), \mathsf{MP}(\pi) > 0\}$
if $s^i_{\mathsf{init}} \notin S_{WC}$ **then**
  return NO
else
  Let $G^w := G^i \downharpoonright S_{WC}$ be the subgame induced by worst-case winning states
  Build $G := G^w \otimes \mathcal{M}(\lambda_2^i) = (\mathcal{G}, S_1, S_2)$, $\mathcal{G} = (S, E, w)$, $S \subseteq (S_{WC} \times \mathsf{Mem})$, the game obtained by product with the Moore machine, and $s_{\mathsf{init}} := (s^i_{\mathsf{init}}, \mathsf{m}_0)$ the corresponding initial state
  Let $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M(G)$ be the memoryless transcription of λ_2^i on G
  Let $P := G[\lambda_2^{\mathsf{stoch}}] = (\mathcal{G}, S_1, S_\Delta = S_2, \Delta = \lambda_2^{\mathsf{stoch}})$ be the MDP obtained from G and $\lambda_2^{\mathsf{stoch}}$

{*Main algorithm*}
Compute $\mathcal{U}_\mathsf{w}$ the set of maximal winning end-components of P
Build $P' = (\mathcal{G}', S_1, S_\Delta, \Delta)$, where $\mathcal{G}' = (S, E, w')$ and w' is defined as follows:

$$\forall e = (s_1, s_2) \in E,\ w'(e) := \begin{cases} w(e) \text{ if } \exists U \in \mathcal{U}_\mathsf{w} \text{ s.t. } \{s_1, s_2\} \subseteq U \\ 0 \text{ otherwise} \end{cases}$$

Compute the maximal expected value ν^* from s_{init} in P'
if $\nu^* > \nu$ **then**
  return YES
else
  return NO

Algorithm: overview

Algorithm 1 BWC_MP$(G^i, \lambda_2^i, \mu^i, \nu^i, s^i_{\mathsf{init}})$

Require: $G^i = (\mathcal{G}^i, S^i_1, S^i_2)$ a game, $\mathcal{G}^i = (S^i, E^i, w^i)$ its underlying graph, $\lambda_2^i \in \Lambda_2^F(G^i)$ a finite-memory stochastic model of the adversary, $\mathcal{M}(\lambda_2^i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_\mathsf{u}, \alpha_\mathsf{n})$ its Moore machine, $\mu^i = \frac{a}{b}, \nu^i \in \mathbb{Q}$, $\mu^i < \nu^i$, resp. the worst-case and the expected value thresholds, and $s^i_{\mathsf{init}} \in S^i$ the initial state

Ensure: The answer is YES if and only if $\mathcal{P}_1$ has a finite-memory strategy $\lambda_1 \in \Lambda_1^F(G^i)$ satisfying the BWC problem from s^i_{init}, for the thresholds pair (μ^i, ν^i) and the mean-payoff value function

Boolean output + by-product strategy

{*Preprocessing*}
if $\mu^i \neq 0$ **then**
  Modify the weight function of $\mathcal{G}^i$ s.t. $\forall e \in E^i$, $w^i_{\mathsf{new}}(e) := b \cdot w^i(e) - a$, and consider the new thresholds pair $(0, \nu := b \cdot \nu^i - a)$
Compute $S_{WC} := \{s \in S^i \mid \exists \lambda_1 \in \Lambda_1(G^i), \forall \lambda_2 \in \Lambda_2(G^i), \forall \pi \in \mathsf{Outs}_{G^i}(s, \lambda_1, \lambda_2), \mathsf{MP}(\pi) > 0\}$
if $s^i_{\mathsf{init}} \notin S_{WC}$ **then**
  return NO
else
  Let $G^w := G^i \upharpoonright S_{WC}$ be the subgame induced by worst-case winning states
  Build $G := G^w \otimes \mathcal{M}(\lambda_2^i) = (\mathcal{G}, S_1, S_2)$, $\mathcal{G} = (S, E, w)$, $S \subseteq (S_{WC} \times \mathsf{Mem})$, the game obtained by product with the Moore machine, and $s_{\mathsf{init}} := (s^i_{\mathsf{init}}, \mathsf{m}_0)$ the corresponding initial state
  Let $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M(G)$ be the memoryless transcription of λ_2^i on G
  Let $P := G[\lambda_2^{\mathsf{stoch}}] = (\mathcal{G}, S_1, S_\Delta = S_2, \Delta = \lambda_2^{\mathsf{stoch}})$ be the MDP obtained from G and $\lambda_2^{\mathsf{stoch}}$

{*Main algorithm*}
Compute $\mathcal{U}_\mathsf{w}$ the set of maximal winning end-components of P
Build $P' = (\mathcal{G}', S_1, S_\Delta, \Delta)$, where $\mathcal{G}' = (S, E, w')$ and w' is defined as follows:

$$\forall e = (s_1, s_2) \in E,\ w'(e) := \begin{cases} w(e) \text{ if } \exists U \in \mathcal{U}_\mathsf{w} \text{ s.t. } \{s_1, s_2\} \subseteq U \\ 0 \text{ otherwise} \end{cases}$$

Compute the maximal expected value ν^* from s_{init} in P'
if $\nu^* > \nu$ **then**
  return YES
else
  return NO

Algorithm: overview

Algorithm 1 BWC_MP$(G^i, \lambda_2^i, \mu^i, \nu^i, s^i_{\mathsf{init}})$

Require: $G^i = (\mathcal{G}^i, S_1^i, S_2^i)$ a game, $\mathcal{G}^i = (S^i, E^i, w^i)$ its underlying graph, $\lambda_2^i \in \Lambda_2^F(G^i)$ a finite-memory stochastic model of the adversary, $\mathcal{M}(\lambda_2^i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_{\mathsf{u}}, \alpha_{\mathsf{n}})$ its Moore machine, $\mu^i = \frac{a}{b}, \nu^i \in \mathbb{Q}$, $\mu^i < \nu^i$, resp. the worst-case and the expected value thresholds, and $s^i_{\mathsf{init}} \in S^i$ the initial state

Ensure: The answer is YES if and only if $\mathcal{P}_1$ has a finite-memory strategy $\lambda_1 \in \Lambda_1^F(G^i)$ satisfying the BWC problem from s^i_{init}, for the thresholds pair (μ^i, ν^i) and the mean-payoff value function

{*Preprocessing*}
if $\mu^i \neq 0$ **then**
  Modify the weight function of $\mathcal{G}^i$ s.t. $\forall e \in E^i$, $w^i_{\mathsf{new}}(e) := b \cdot w^i(e) - a$, and consider the new thresholds pair $(0, \nu := b \cdot \nu^i - a)$
Compute $S_{WC} := \{s \in S^i \mid \exists \lambda_1 \in \Lambda_1(G^i), \forall \lambda_2 \in \Lambda_2(G^i), \forall \pi \in \mathsf{Outs}_{G^i}(s, \lambda_1, \lambda_2), \mathsf{MP}(\pi) > 0\}$
if $s^i_{\mathsf{init}} \notin S_{WC}$ **then**
  return NO
else
  Let $G^w := G^i \downharpoonright S_{WC}$ be the subgame induced by worst-case winning states
  Build $G := G^w \otimes \mathcal{M}(\lambda_2^i) = (\mathcal{G}, S_1, S_2)$, $\mathcal{G} = (S, E, w)$, $S \subseteq (S_{WC} \times \mathsf{Mem})$, the game obtained by product with the Moore machine, and $s_{\mathsf{init}} := (s^i_{\mathsf{init}}, \mathsf{m}_0)$ the corresponding initial state
  Let $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M(G)$ be the memoryless transcription of λ_2^i on G
  Let $P := G[\lambda_2^{\mathsf{stoch}}] = (\mathcal{G}, S_1, S_\Delta = S_2, \Delta = \lambda_2^{\mathsf{stoch}})$ be the MDP obtained from G and $\lambda_2^{\mathsf{stoch}}$

Preprocessing

{*Main algorithm*}
Compute $\mathcal{U}_{\mathsf{w}}$ the set of maximal winning end-components of P
Build $P' = (\mathcal{G}', S_1, S_\Delta, \Delta)$, where $\mathcal{G}' = (S, E, w')$ and w' is defined as follows:

$$\forall e = (s_1, s_2) \in E,\ w'(e) := \begin{cases} w(e) \text{ if } \exists U \in \mathcal{U}_{\mathsf{w}} \text{ s.t. } \{s_1, s_2\} \subseteq U \\ 0 \text{ otherwise} \end{cases}$$

Compute the maximal expected value ν^* from s_{init} in P'
if $\nu^* > \nu$ **then**
  return YES
else
  return NO

Algorithm: overview

Algorithm 1 BWC_MP$(G^i, \lambda_2^i, \mu^i, \nu^i, s^i_{\mathsf{init}})$

Require: $G^i = (\mathcal{G}^i, S_1^i, S_2^i)$ a game, $\mathcal{G}^i = (S^i, E^i, w^i)$ its underlying graph, $\lambda_2^i \in \Lambda_2^F(G^i)$ a finite-memory stochastic model of the adversary, $\mathcal{M}(\lambda_2^i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_{\mathsf{u}}, \alpha_{\mathsf{n}})$ its Moore machine, $\mu^i = \frac{a}{b}, \nu^i \in \mathbb{Q}$, $\mu^i < \nu^i$, resp. the worst-case and the expected value thresholds, and $s^i_{\mathsf{init}} \in S^i$ the initial state

Ensure: The answer is YES if and only if $\mathcal{P}_1$ has a finite-memory strategy $\lambda_1 \in \Lambda_1^F(G^i)$ satisfying the BWC problem from s^i_{init}, for the thresholds pair (μ^i, ν^i) and the mean-payoff value function

{Preprocessing}

if $\mu^i \neq 0$ **then**
  Modify the weight function of $\mathcal{G}^i$ s.t. $\forall e \in E^i$, $w^i_{\mathsf{new}}(e) := b \cdot w^i(e) - a$, and consider the new thresholds pair $(0, \nu := b \cdot \nu^i - a)$
Compute $S_{WC} := \{s \in S^i \mid \exists \lambda_1 \in \Lambda_1(G^i), \forall \lambda_2 \in \Lambda_2(G^i), \forall \pi \in \mathsf{Outs}_{G^i}(s, \lambda_1, \lambda_2), \mathsf{MP}(\pi) > 0\}$
if $s^i_{\mathsf{init}} \notin S_{WC}$ **then**
  return NO
else
  Let $G^w := G^i \downharpoonright S_{WC}$ be the subgame induced by worst-case winning states
  Build $G := G^w \otimes \mathcal{M}(\lambda_2^i) = (\mathcal{G}, S_1, S_2)$, $\mathcal{G} = (S, E, w)$, $S \subseteq (S_{WC} \times \mathsf{Mem})$, the game obtained by product with the Moore machine, and $s_{\mathsf{init}} := (s^i_{\mathsf{init}}, \mathsf{m}_0)$ the corresponding initial state
  Let $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M(G)$ be the memoryless transcription of λ_2^i on G
  Let $P := G[\lambda_2^{\mathsf{stoch}}] = (\mathcal{G}, S_1, S_\Delta = S_2, \Delta = \lambda_2^{\mathsf{stoch}})$ be the MDP obtained from G and $\lambda_2^{\mathsf{stoch}}$

Main algorithm

{Main algorithm}

Compute $\mathcal{U}_{\mathsf{w}}$ the set of maximal winning end-components of P
Build $P' = (\mathcal{G}', S_1, S_\Delta, \Delta)$, where $\mathcal{G}' = (S, E, w')$ and w' is defined as follows:

$$\forall e = (s_1, s_2) \in E,\ w'(e) := \begin{cases} w(e) \text{ if } \exists U \in \mathcal{U}_{\mathsf{w}} \text{ s.t. } \{s_1, s_2\} \subseteq U \\ 0 \text{ otherwise} \end{cases}$$

Compute the maximal expected value ν^* from s_{init} in P'
if $\nu^* > \nu$ **then**
  return YES
else
  return NO

Preprocessing: three steps

1. Modify weights and use thresholds ($\mu = 0$, ν)
   - ▷ simple trick to ease the following technicalities

Preprocessing: three steps

1. Modify weights and use thresholds ($\mu = 0$, ν)
   - ▷ simple trick to ease the following technicalities

2. **Remove all worst-case losing states**

$$S_{WC} := \left\{ s \in S^i \mid \exists\, \lambda_1 \in \Lambda_1(G^i), \forall\, \lambda_2 \in \Lambda_2(G^i), \forall\, \pi \in \mathsf{Outs}_{G^i}(s, \lambda_1, \lambda_2), \mathsf{MP}(\pi) > 0 \right\}$$

$$G^w := G^i \downharpoonright S_{WC}$$

- ▷ BWC satisfying strategies must avoid $S \setminus S_{WC}$: an antagonistic adversary can force WC losing outcomes from there (due to prefix-independence)
- ▷ Answer No if $s_{\mathsf{init}} \notin S_{WC}$

Preprocessing: three steps

1. Modify weights and use thresholds ($\mu = 0$, ν)
   - ▷ simple trick to ease the following technicalities

2. **Remove all worst-case losing states**

$$S_{WC} := \left\{ s \in S^i \mid \exists\, \lambda_1 \in \Lambda_1(G^i),\, \forall\, \lambda_2 \in \Lambda_2(G^i),\, \forall\, \pi \in \mathsf{Outs}_{G^i}(s, \lambda_1, \lambda_2),\, \mathsf{MP}(\pi) > 0 \right\}$$

$$G^w := G^i \downharpoonright S_{WC}$$

- ▷ BWC satisfying strategies must avoid $S \setminus S_{WC}$: an antagonistic adversary can force WC losing outcomes from there (due to prefix-independence)
- ▷ Answer No if $s_{\mathsf{init}} \notin S_{WC}$
- ▷ In G^w, $\mathcal{P}_1$ has a **memoryless WC winning strategy** from all states

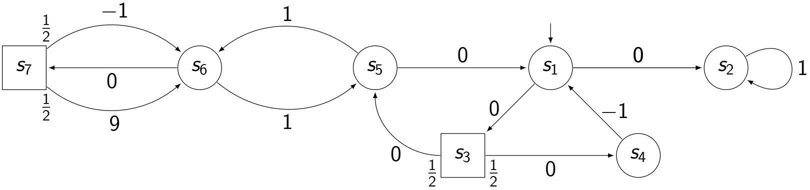
Preprocessing: three steps

3. Build $G := G^w \otimes \mathcal{M}(\lambda_2^i)$, the game obtained by **product with the Moore machine**
   - Corresponding stochastic model $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M(G)$ is **memoryless**

Preprocessing: three steps

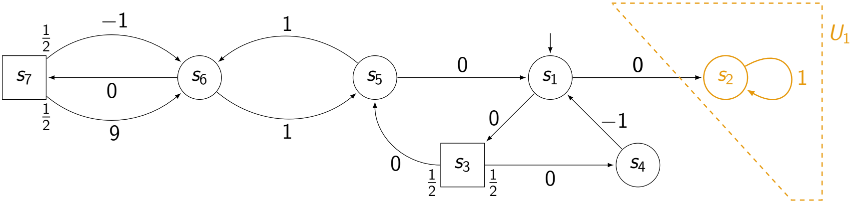
3. Build $G := G^{w} \otimes \mathcal{M}(\lambda_2^{i})$, the game obtained by **product with the Moore machine**
   - Corresponding stochastic model $\lambda_2^{\text{stoch}} \in \Lambda_2^{M}(G)$ is **memoryless**
   - Obtain the MDP $P := G[\lambda_2^{\text{stoch}}]$, **sharing the same graph**
     - helps for elegant proofs

Main algorithm: end-components



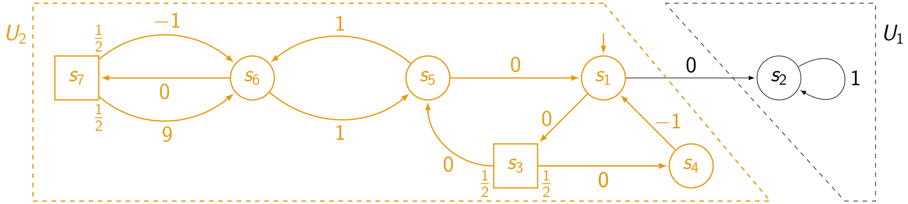
- An **EC** of the MDP $P = G[\lambda_2^{\text{stoch}}]$ is a subgraph in which $\mathcal{P}_1$ can ensure to stay despite stochastic states [dA97], i.e., a set $U \subseteq S$ s.t.
  - (i) $(U, E_\Delta \cap (U \times U))$ is strongly connected,
  - (ii) $\forall\, s \in U \cap S_\Delta$, $\text{Supp}(\Delta(s)) \subseteq U$, i.e., in stochastic states, all outgoing edges either stay in U or belong to $E \setminus E_\Delta$.
- Beware arbitrary adversaries may use edges in $E \setminus E_\Delta$!

Main algorithm: end-components



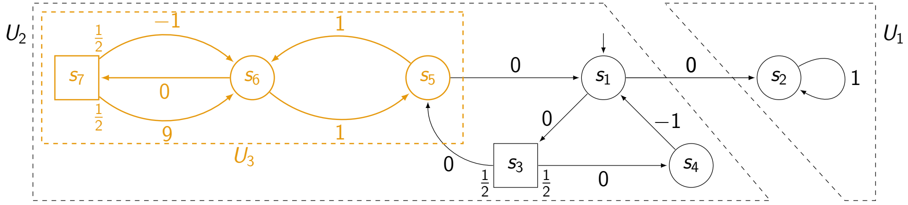
ECs: $\mathcal{E} = \{ U_1$

Main algorithm: end-components



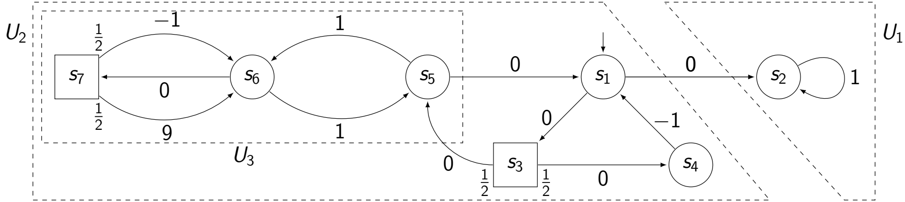
ECs: $\mathcal{E} = \{U_1, U_2$

Main algorithm: end-components



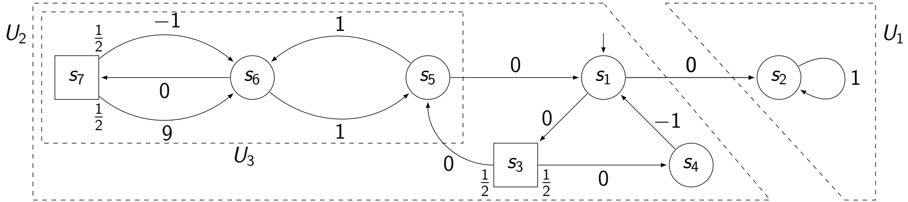
ECs: $\mathcal{E} = \{U_1, U_2, U_3$

Main algorithm: end-components



ECs: $\mathcal{E} = \{U_1, U_2, U_3, \{s_5, s_6\}, \{s_6, s_7\}, \{s_1, s_3, s_4, s_5\}\}$

Main algorithm: end-components



ECs: $\mathcal{E} = \{U_1, U_2, U_3, \{s_5, s_6\}, \{s_6, s_7\}, \{s_1, s_3, s_4, s_5\}\}$

Lemma (Long-run appearance of ECs [CY95, dA97])

Let $\lambda_1 \in \Lambda_1(P)$ be an **arbitrary strategy** *of $\mathcal{P}_1$. Then, we have that*

$$\mathbb{P}^{P[\lambda_1]}_{s_{\mathsf{init}}}\left(\{\pi \in \mathsf{Outs}_{P[\lambda_1]}(s_{\mathsf{init}}) \mid \mathsf{Inf}(\pi) \in \mathcal{E}\}\right) = 1.$$

▷ **The expectation on $P[\lambda_1]$ depends uniquely on ECs**

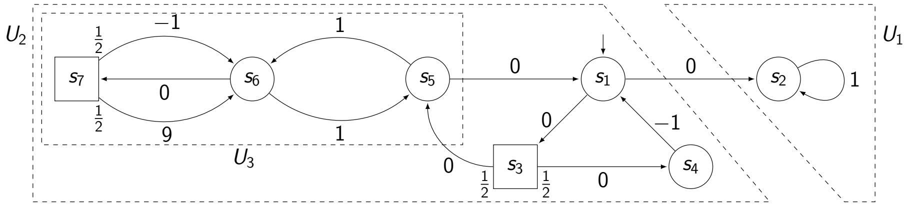
How to satisfy the BWC problem?

- *Expected value requirement*: reach ECs with the highest achievable expectations and stay in them (optimal expected value in EC [FV97])

How to satisfy the BWC problem?

- *Expected value requirement*: reach ECs with the highest achievable expectations and stay in them (optimal expected value in EC [FV97])
- *Worst-case requirement*: some ECs may need to be eventually avoided because risky!

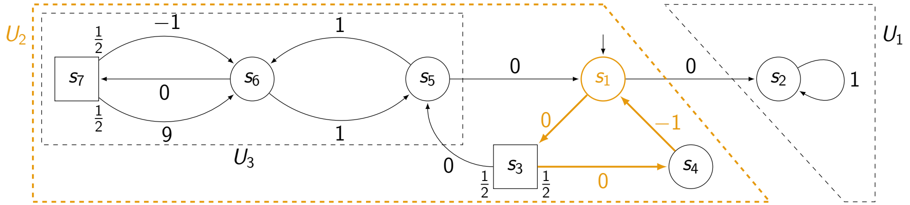
Classification of ECs



▷ $U \in \mathcal{W}$, **the winning ECs**, if $\mathcal{P}_1$ can win in $G_\Delta \downharpoonleft U$, from all states:

$$\exists\, \lambda_1 \in \Lambda_1(G_\Delta \downharpoonleft U), \forall\, \lambda_2 \in \Lambda_2(G_\Delta \downharpoonleft U), \forall\, s \in U, \forall\, \pi \in \mathsf{Outs}_{(G_\Delta \downharpoonleft U)}(s, \lambda_1, \lambda_2), \mathsf{MP}(\pi) > 0$$

Classification of ECs



- ▷ $U \in \mathcal{W}$, **the winning ECs**, if $\mathcal{P}_1$ can win in $G_\Delta \downharpoonleft U$, from all states:

$$\exists \lambda_1 \in \Lambda_1(G_\Delta \downharpoonleft U), \forall \lambda_2 \in \Lambda_2(G_\Delta \downharpoonleft U), \forall s \in U, \forall \pi \in \mathrm{Outs}_{(G_\Delta \downharpoonleft U)}(s, \lambda_1, \lambda_2), \mathrm{MP}(\pi) > 0$$

- ▷ $\mathcal{W} = \{U_1, U_3, \{s_5, s_6\}, \{s_6, s_7\}\}$
- ▷ U_2 **losing**: from state s_1, $\mathcal{P}_2$ can force the outcome $\pi = (s_1 s_3 s_4)^\omega$ of $\mathrm{MP}(\pi) = -1/3 < 0$

Winning ECs: usefulness

Lemma (Long-run appearance of winning ECs)

Let $\lambda_1^f \in \Lambda_1^F$ *be a* **finite-memory** *strategy of* $\mathcal{P}_1$ *that* **satisfies** *the BWC problem for thresholds* $(0, \nu) \in \mathbb{Q}^2$*. Then, we have that*

$$\mathbb{P}^{P[\lambda_1^f]}_{s_{\mathsf{init}}}\left(\left\{\pi \in \mathsf{Outs}_{P[\lambda_1^f]}(s_{\mathsf{init}}) \mid \mathsf{Inf}(\pi) \in \mathcal{W}\right\}\right) = 1.$$

Winning ECs: usefulness

Lemma (Long-run appearance of winning ECs)

Let $\lambda_1^f \in \Lambda_1^F$ *be a* **finite-memory** *strategy of* $\mathcal{P}_1$ *that* **satisfies** *the BWC problem for thresholds* $(0, \nu) \in \mathbb{Q}^2$*. Then, we have that*

$$\mathbb{P}^{P[\lambda_1^f]}_{s_{\text{init}}}\left(\left\{\pi \in \mathsf{Outs}_{P[\lambda_1^f]}(s_{\text{init}}) \mid \mathsf{Inf}(\pi) \in \mathcal{W}\right\}\right) = 1.$$

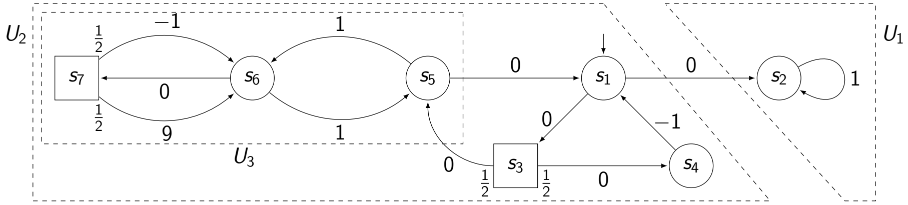
- ▷ A good finite-memory strategy for the BWC problem should *maximize* the expected value achievable through *winning* ECs

Winning ECs: computation

- Deciding if an EC is winning or not is in NP $\cap$ coNP (worst-case threshold problem)
- $|\mathcal{E}| \leq 2^{|S|} \leadsto$ exponential # of ECs

Winning ECs: computation

- Deciding if an EC is winning or not is in NP $\cap$ coNP (worst-case threshold problem)
- $|\mathcal{E}| \leq 2^{|S|} \rightsquigarrow$ exponential # of ECs
- Considering the maximal ECs **does not** suffice! See $U_3 \subset U_2$



Winning ECs: computation

- Deciding if an EC is winning or not is in NP $\cap$ coNP (worst-case threshold problem)
- $|\mathcal{E}| \leq 2^{|S|} \rightsquigarrow$ exponential # of ECs
- Considering the maximal ECs **does not** suffice! See $U_3 \subset U_2$

But,

- possible to define a recursive algorithm computing the **maximal winning ECs**, such that $|\mathcal{U}_{\mathrm{W}}| \leq |S|$, in NP $\cap$ coNP.
- Uses polynomial number of of calls to
  - max. EC decomp. of sub-MDPs (each in $\mathcal{O}(|S|^2)$ [CH12]),
  - worst-case threshold problem (NP $\cap$ coNP).
- Critical complexity gain for the overall algorithm $\mathrm{BWC\_MP}$!

Winning ECs: what can we expect?

We know we can only benefit from the expectation of winning ECs.
But how can we compute it?

Winning ECs: what can we expect?

We know we can only benefit from the expectation of winning ECs. But how can we compute it?

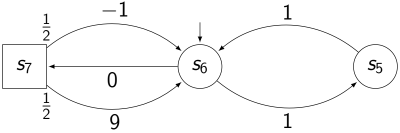
Theorem (BWC satisfaction from winning ECs)

Let $U \in \mathcal{W}$ a winning EC, $s_{\mathsf{init}} \in U$ an initial state inside the EC, and $\nu^* \in \mathbb{Q}$ the maximal expected value achievable by $\mathcal{P}_1$ in $P \downharpoonleft U$. Then, for all $\varepsilon > 0$, there exists a finite-memory strategy of $\mathcal{P}_1$ that satisfies the BWC problem for the thresholds pair $(0, \nu^* - \varepsilon)$.

▷ We can be **arbitrarily close to the optimal expectation** of the EC while ensuring the worst-case!

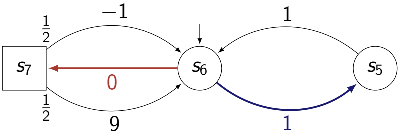
Inside a WEC: combined strategy

Consider the WEC $U_3 \subseteq S$ and $E \setminus E_\Delta = \emptyset$



Inside a WEC: combined strategy

Consider the WEC $U_3 \subseteq S$ and $E \setminus E_\Delta = \emptyset$



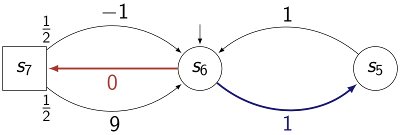
Two particular memoryless strategies exist:

1. Optimal expected value strategy $\lambda_1^e \in \Lambda_1^{PM}(P)$, yielding $\mathbb{E} = 2$
2. Optimal worst-case strategy $\lambda_1^{wc} \in \Lambda_1^{PM}(G)$, ensuring MP $= 1 > 0$

Remark: $\nu^* = 2 > \mu^* = 1$

Inside a WEC: combined strategy

Consider the WEC $U_3 \subseteq S$ and $E \setminus E_\Delta = \emptyset$



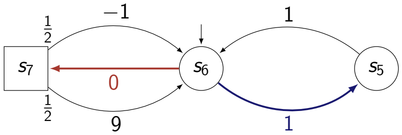
We define $\lambda_1^{cmb} \in \Lambda_1^{PF}$ as follows, for some well-chosen $K, L \in \mathbb{N}$.

(a) Play λ_1^{e} for K steps and memorize $\mathsf{Sum} \in \mathbb{Z}$, the sum of weights encountered during these K steps.

(b) If $\mathsf{Sum} > 0$, then go to *(a)*.
Else, play λ_1^{wc} during L steps then go to *(a)*.

Inside a WEC: combined strategy

Consider the WEC $U_3 \subseteq S$ and $E \setminus E_\Delta = \emptyset$



- *Phase (a)*: try to increase the expectation and approach the optimal one
- *Phase (b)*: compensate, if needed, losses that occured in *(a)*

Combined strategy: parameters

Key result: $\exists\, K, L \in \mathbb{N}$ for any thresholds pair $(0,\, \nu^* - \varepsilon)$

- plays = sequences of periods starting with phase *(a)*

Combined strategy: parameters

Key result: $\exists\, K, L \in \mathbb{N}$ for any thresholds pair $(0,\, \nu^* - \varepsilon)$

- plays = sequences of periods starting with phase *(a)*
- *Worst-case requirement*
  - ▷ $\forall\, K$, $\exists\, L(K)$ s.t. *(a)* + *(b)* has MP $\geq 1/(K + L) > 0$
  - ▷ Periods *(a)* induce MP $\geq 1/K$ (not followed by *(b)*)
  - ▷ Weights are integers and period length bounded $\rightsquigarrow$ inequality remains strict for play

Combined strategy: parameters

Key result: $\exists\, K, L \in \mathbb{N}$ for any thresholds pair $(0, \nu^* - \varepsilon)$

- plays = sequences of periods starting with phase *(a)*
- *Worst-case requirement*
  - ▷ $\forall\, K$, $\exists\, L(K)$ s.t. *(a)* + *(b)* has MP $\geq 1/(K + L) > 0$
  - ▷ Periods *(a)* induce MP $\geq 1/K$ (not followed by *(b)*)
  - ▷ Weights are integers and period length bounded $\rightsquigarrow$ inequality remains strict for play
- *Expected value requirement*
  - ▷ When $K \to \infty$, $\mathbb{E}_{(a)} \to \nu^*$
  - ▷ We need the *overall contribution* of *(b)* to tend to zero when $K \to \infty$
    - $\mathbb{P}_{(b)}$ decreases faster than increase of $L(K)$: exponential vs. polynomial
    - proved using results related to Chernoff bounds and Hoeffding's inequality on MCs [Tra09, GO02]: bound on the probability of being far from the optimal after K steps of *(a)*

Witness-and-secure strategy

What if $E \setminus E_{\Delta} \neq \emptyset$?

- arbitrary adversaries can produce bad behaviors
- add the possibility to **react** using a worst-case winning strategy (existing everywhere thanks to the preprocessing)
  - ▷ guarantees worst-case
  - ▷ no impact on expected value (probability zero)

Back to the algorithm

So we know we should only use WECs and we know how to play ε-optimally when starting in a WEC. *What remains to settle?*

Back to the algorithm

So we know we should only use WECs and we know how to play ε-optimally when starting in a WEC. *What remains to settle?*

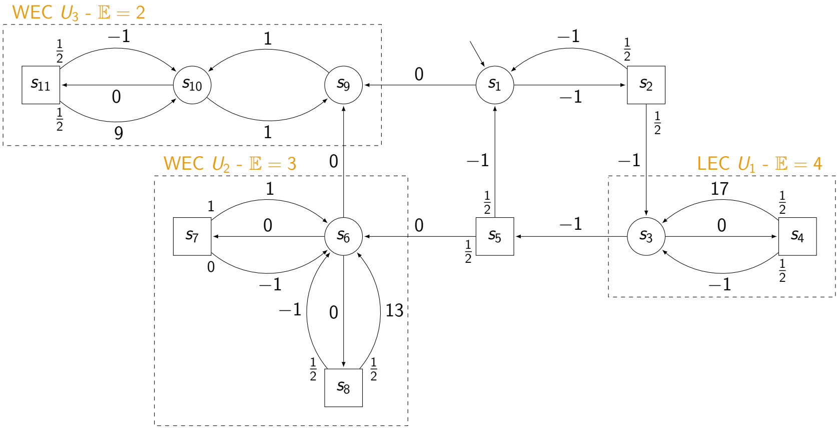
- ▷ Determine **which** WECs to reach and **how**!

Back to the algorithm

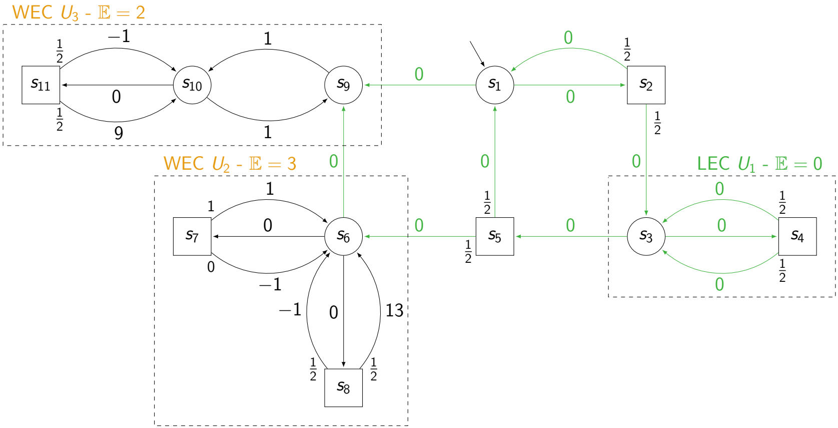
So we know we should only use WECs and we know how to play ε-optimally when starting in a WEC. *What remains to settle?*

- ▷ Determine **which** WECs to reach and **how**!
- ▷ Key idea: define a **global strategy** that will go towards the highest valued WECs and avoid LECs

Global strategy via modified MDP



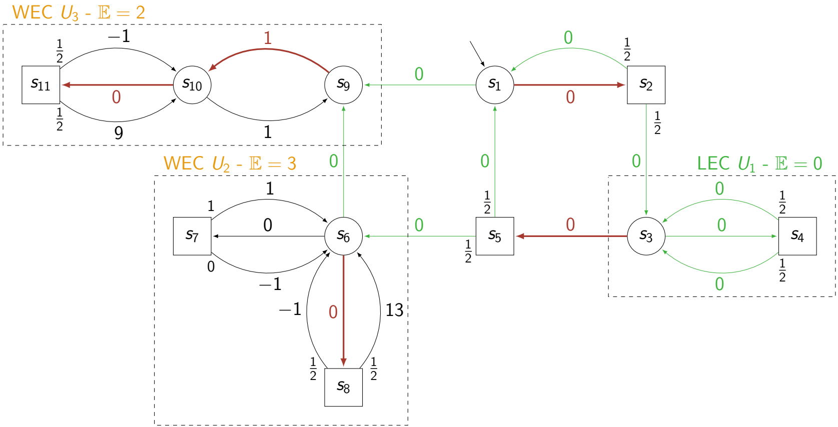
Global strategy via modified MDP



1. Modify weights:

$$\forall\, e = (s_1, s_2) \in E,\ w'(e) := \begin{cases} w(e) \text{ if } \exists\, U \in \mathcal{U}_{\mathrm{w}} \text{ s.t. } \{s_1, s_2\} \subseteq U, \\ 0 \text{ otherwise.} \end{cases}$$

Global strategy via modified MDP



2 Compute memoryless optimal expectation strategy λ_1^e on P'

- ▷ the probability to be in a good WEC (here, U_2) after N steps tends to one when $N \to \infty$

Global strategy via modified MDP

3 $\lambda_1^{glb} \in \Lambda_1^{PF}(G)$:

(a) Play $\lambda_1^{e} \in \Lambda_1^{PM}(G)$ for N steps.

(b) Let $s \in S$ be the reached state.

(b.1) If $s \in U \in \mathcal{U}_{\mathrm{w}}$, play corresponding $\lambda_1^{wns} \in \Lambda_1^{PF}(G)$ forever.

(b.2) Else play $\lambda_1^{wc} \in \Lambda_1^{PM}(G)$ forever.

- ▷ Parameter $N \in \mathbb{N}$ can be chosen so that overall expectation is arbitrarily close to optimal in P', or equivalently, optimal for BWC strategies in P
- ▷ Algorithm $\mathrm{BWC\_MP}$ answers $\textsc{Yes}$ iff $\nu^* > \nu$

Correctness and completeness

Algorithm $\mathrm{BWC\_MP}$ is

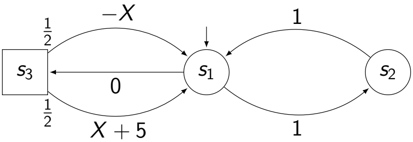
- **correct**: if answer is $\mathrm{Y{\scriptsize ES}}$, then λ_1^{glb} satisfies the BWC problem for the given thresholds
- **complete**: if answer is $\mathrm{N{\scriptsize O}}$, then the BWC problem cannot be satisfied by a finite-memory strategy

BWC MP problem: bounds

- *Complexity*
  - ▷ algorithm in NP ∩ coNP (P if MP games proved in P)
  - ▷ lower bound via reduction from MP games

BWC MP problem: bounds

- *Complexity*
  - algorithm in NP ∩ coNP (P if MP games proved in P)
  - lower bound via reduction from MP games



- *Memory*
  - pseudo-polynomial upper bound via global strategy
  - matching lower bound via family $(G(X))_{X \in \mathbb{N}_0}$ requiring polynomial memory in $W = X + 5$ to satisfy the BWC problem for thresholds $(0, \nu \in]1, 5/4[)$
    - ⇝ need to use (s_1, s_3) infinitely often for $\mathbb{E}$ but need pseudo-poly. memory to counteract $-X$ for the WC requirement

Shortest path - truncated sum

- Assume strictly positive integer weights, $w\colon E \to \mathbb{N}_0$
- Let $T \subseteq S$ be a *target set* that $\mathcal{P}_1$ wants to reach with a path of bounded value (cf. introductory example)
  - ▷ inequalities are reversed, $\nu < \mu$
- $\mathsf{TS}_T(\pi = s_0 s_1 s_2 \dots) = \sum_{i=0}^{n-1} w((s_i, s_{i+1}))$, with n the first index such that $s_n \in T$, and $\mathsf{TS}_T(\pi) = \infty$ if $\forall\, n$, $s_n \notin T$

Shortest path - truncated sum

- Assume strictly positive integer weights, $w\colon E \to \mathbb{N}_0$
- Let $T \subseteq S$ be a *target set* that $\mathcal{P}_1$ wants to reach with a path of bounded value (cf. introductory example)
  - ▷ inequalities are reversed, $\nu < \mu$
- $\mathsf{TS}_T(\pi = s_0 s_1 s_2 \dots) = \sum_{i=0}^{n-1} w((s_i, s_{i+1}))$, with n the first index such that $s_n \in T$, and $\mathsf{TS}_T(\pi) = \infty$ if $\forall\, n,\ s_n \notin T$

Games: worst-case threshold problem

Memoryless optimal strategies as cycles are to be avoided, and the problem is in P, solvable using attractors and computation of the worst cost.

MDPs: expected value threshold problem [BT91, dA99]

Memoryless optimal strategies exist and the problem is in P.

BWC SP problem: overview

Theorem (algorithm)

The BWC problem for the shortest path can be solved in **pseudo-polynomial** *time: polynomial in the size of the game graph, the Moore machine for the stochastic model of the adversary and the encoding of the expected value threshold, and polynomial in the value of the worst-case threshold.*

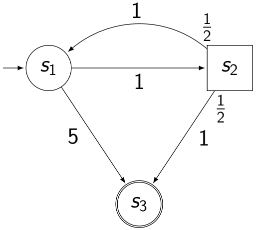
Theorem (memory bounds)

Pseudo-polynomial *memory may be necessary and is always sufficient to satisfy the BWC problem for the shortest path.*

Theorem (complexity lower bound)

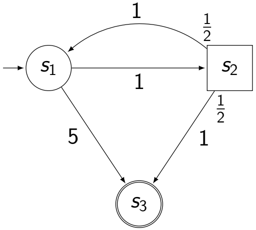
The BWC problem for the shortest path is **NP-hard**.

Pseudo-polynomial algorithm: sketch



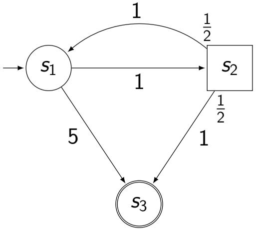
1. Start from $G = (\mathcal{G}, S_1, S_2)$, $\mathcal{G} = (S, E, w)$, $T = \{s_3\}$, $\mathcal{M}(\lambda_2^{\text{stoch}})$, $\mu = 8$, and $\nu \in \mathbb{Q}$

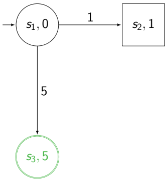
Pseudo-polynomial algorithm: sketch



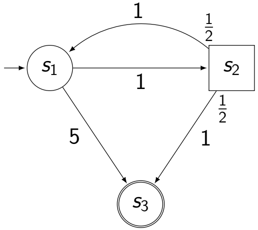
1. Start from $G = (\mathcal{G}, S_1, S_2)$, $\mathcal{G} = (S, E, w)$, $T = \{s_3\}$, $\mathcal{M}(\lambda_2^{\mathsf{stoch}})$, $\mu = 8$, and $\nu \in \mathbb{Q}$
2. Build G' by unfolding $\mathcal{G}$, tracking the current sum *up to the worst-case threshold* μ, and integrating it in the states of $\mathcal{G}'$.

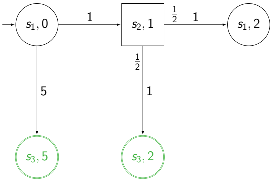
Pseudo-polynomial algorithm: sketch



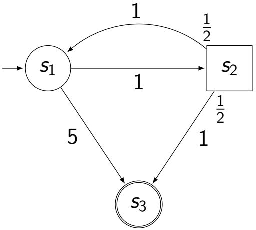


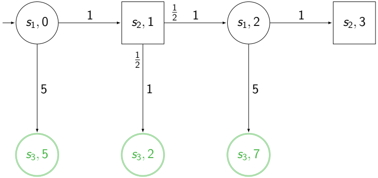
Pseudo-polynomial algorithm: sketch



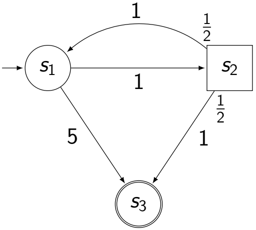


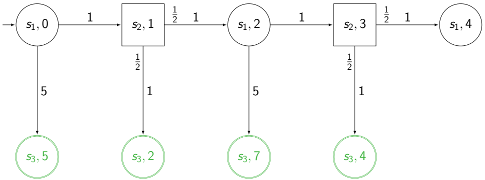
Pseudo-polynomial algorithm: sketch



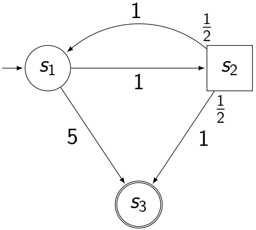


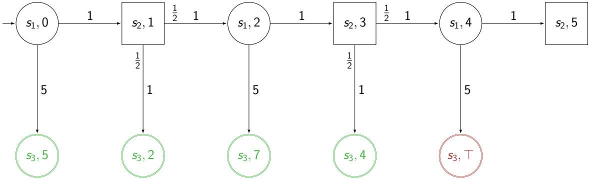
Pseudo-polynomial algorithm: sketch



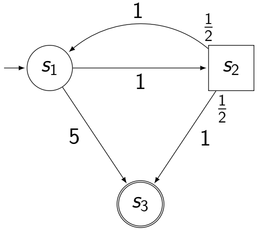


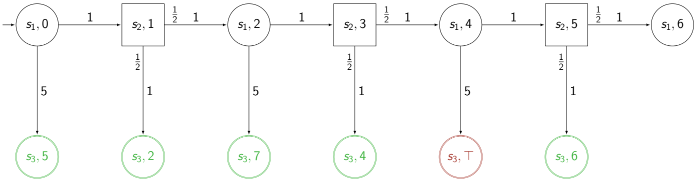
Pseudo-polynomial algorithm: sketch



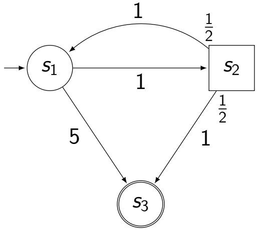


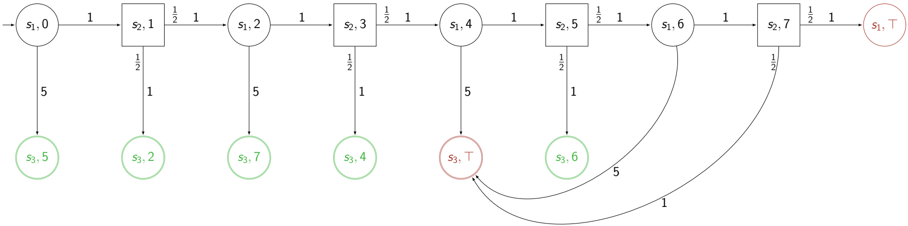
Pseudo-polynomial algorithm: sketch





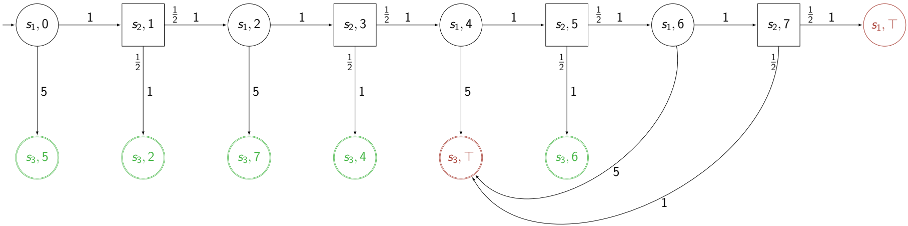
Pseudo-polynomial algorithm: sketch





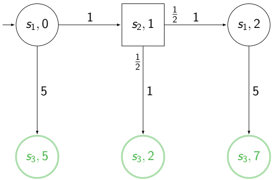
Pseudo-polynomial algorithm: sketch

3. Compute R, the attractor of T with cost $< \mu = 8$
4. Consider $G_\mu = G' \downharpoonright R$



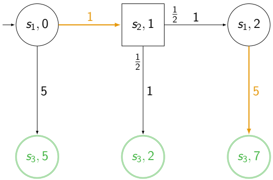
Pseudo-polynomial algorithm: sketch

3. Compute R, the attractor of T with cost $< \mu = 8$
4. Consider $G_\mu = G' \downharpoonright R$



Pseudo-polynomial algorithm: sketch

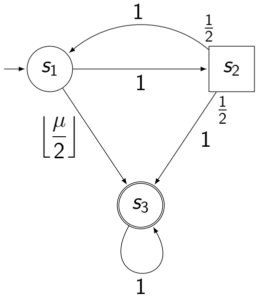
5. Consider $P = G_\mu \otimes \mathcal{M}(\lambda_2^{\text{stoch}})$
6. Compute memoryless optimal expectation strategy
7. If $\nu^* < \nu$, answer YES, otherwise answer NO



Here, $\nu^* = 9/2$

Memory bounds

- ▷ Upper bound provided by synthesized strategy
- ▷ Lower bound given by family of games $(G(\mu))_{\mu \in \{13+k\cdot 4 | k \in \mathbb{N}\}}$ requiring memory linear in μ
  - $\rightsquigarrow$ play (s_1, s_2) exactly $\left\lfloor \frac{\mu}{4} \right\rfloor$ times and then switch to (s_1, s_3) to minimize expected value while ensuring the worst-case



Complexity lower bound: NP-hardness

- Truly-polynomial algorithm very unlikely...
- Reduction from the K^{th} **largest subset problem**
  - ▷ commonly thought to be outside NP as natural certificates are larger than polynomial [JK78, GJ79]

Complexity lower bound: NP-hardness

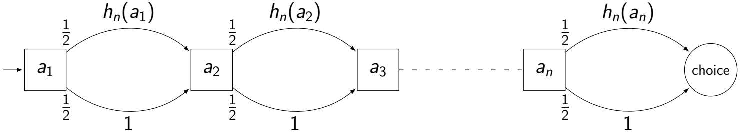
- Truly-polynomial algorithm very unlikely. . .
- Reduction from the K^{th} **largest subset problem**
  - ▷ commonly thought to be outside NP as natural certificates are larger than polynomial [JK78, GJ79]

K^{th} largest subset problem

Given a finite set A, a size function $h \colon A \to \mathbb{N}_0$ assigning strictly positive integer values to elements of A, and two naturals $K, L \in \mathbb{N}$, decide if there exist K distinct subsets $C_i \subseteq A$, $1 \leq i \leq K$, such that $h(C_i) = \sum_{a \in C_i} h(a) \leq L$ for all K subsets.

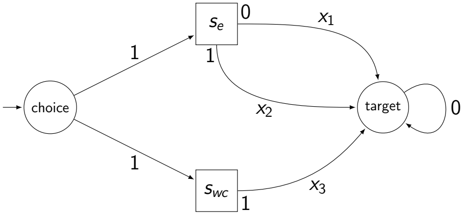
- Build a game composed of *two gadgets*

Random subset selection gadget



- ▷ Stochastically generates paths representing subsets of A: an element is selected in the subset if the upper edge is taken when leaving the corresponding state
- ▷ **All subsets are equiprobable**

Choice gadget



- s_e leads to lower expected values but may be dangerous for the worst-case requirement
- s_{wc} is always safe but induces an higher expected cost

Crux of the reduction

Establish that there exist values for thresholds and weights s.t.

(i) an optimal (i.e., minimizing the expectation while guaranteeing a given worst-case threshold) strategy for $\mathcal{P}_1$ consists in choosing state s_e only when the randomly generated subset $C \subseteq A$ satisfies $h(C) \leq L$;

(ii) this strategy satisfies the BWC problem *if and only if* there exist K distinct subsets that verify this bound.

In a nutshell

- BWC framework combines worst-case and expected value requirements
  - ▷ a natural wish in many practical applications
  - ▷ few existing theoretical support

In a nutshell

- BWC framework combines worst-case and expected value requirements
  - ▷ a natural wish in many practical applications
  - ▷ few existing theoretical support
- Mean-payoff: additional modeling power for no complexity cost (decision-wise)
- Shortest path: harder than the worst-case, pseudo-polynomial with NP-hardness result

In a nutshell

- BWC framework combines worst-case and expected value requirements
  - ▷ a natural wish in many practical applications
  - ▷ few existing theoretical support
- Mean-payoff: additional modeling power for no complexity cost (decision-wise)
- Shortest path: harder than the worst-case, pseudo-polynomial with NP-hardness result
- In both cases, pseudo-polynomial memory is both sufficient and necessary
  - ▷ but strategies have natural representations based on states of the game and simple integer counters

Beyond BWC synthesis?

Possible future works include

- study of other quantitative objectives,
- extension of our results to more general settings (multi-dimension [CDHR10, CRR12], decidable classes of games with imperfect information [DDG+10], etc),
- application of the BWC problem to various practical cases.

Beyond BWC synthesis?

Possible future works include

- study of other quantitative objectives,
- extension of our results to more general settings (multi-dimension [CDHR10, CRR12], decidable classes of games with imperfect information [DDG+10], etc),
- application of the BWC problem to various practical cases.

Thanks!
Do not hesitate to discuss with us!

References I

D.P. Bertsekas and J.N. Tsitsiklis.
An analysis of stochastic shortest path problems.
Mathematics of Operations Research, 16:580–595, 1991.

K. Chatterjee, L. Doyen, T.A. Henzinger, and J.-F. Raskin.
Generalized mean-payoff and energy games.
In Proc. of FSTTCS, LIPIcs 8, pages 505–516. Schloss Dagstuhl - LZI, 2010.

K. Chatterjee, L. Doyen, M. Randour, and J.-F. Raskin.
Looking at mean-payoff and total-payoff through windows.
In Proc. of ATVA, LNCS 8172, pages 118–132. Springer, 2013.

K. Chatterjee and M. Henzinger.
An $\mathcal{O}(n^2)$ time algorithm for alternating Büchi games.
In Proc. of SODA, pages 1386–1399. SIAM, 2012.

K. Chatterjee, M. Randour, and J.-F. Raskin.
Strategy synthesis for multi-dimensional quantitative objectives.
In Proc. of CONCUR, LNCS 7454, pages 115–131. Springer, 2012.

C. Courcoubetis and M. Yannakakis.
The complexity of probabilistic verification.
J. ACM, 42(4):857–907, 1995.

References II

L. de Alfaro.
Formal verification of probabilistic systems.
PhD thesis, Stanford University, 1997.

L. de Alfaro.
Computing minimum and maximum reachability times in probabilistic systems.
In Proc. of CONCUR, LNCS 1664, pages 66–81. Springer, 1999.

A. Degorre, L. Doyen, R. Gentilini, J.-F. Raskin, and S. Torunczyk.
Energy and mean-payoff games with imperfect information.
In Proc. of CSL, LNCS 6247, pages 260–274. Springer, 2010.

A. Ehrenfeucht and J. Mycielski.
Positional strategies for mean payoff games.
Int. Journal of Game Theory, 8(2):109–113, 1979.

J. Filar and K. Vrieze.
Competitive Markov Decision Processes.
Springer, 1997.

M.R. Garey and D.S. Johnson.
Computers and intractability: a guide to the Theory of NP-Completeness.
Freeman New York, 1979.

P.W. Glynn and D. Ormoneit.
Hoeffding's inequality for uniformly ergodic Markov chains.
Statistics & Probability Letters, 56(2):143–146, 2002.

References III

T. Gawlitza and H. Seidl.
Games through nested fixpoints.
In Proc. of CAV, LNCS 5643, pages 291–305. Springer, 2009.

D.B. Johnson and S.D. Kashdan.
Lower bounds for selection in X + Y and other multisets.
Journal of the ACM, 25(4):556–570, 1978.

M. Jurdziński.
Deciding the winner in parity games is in UP $\cap$ co-UP.
Inf. Process. Lett., 68(3):119–124, 1998.

T.M. Liggett and S.A. Lippman.
Stochastic games with perfect information and time average payoff.
Siam Review, 11(4):604–607, 1969.

M.L. Puterman.
Markov Decision Processes: Discrete Stochastic Dynamic Programming.
John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

M. Tracol.
Fast convergence to state-action frequency polytopes for MDPs.
Oper. Res. Lett., 37(2):123–126, 2009.

U. Zwick and M. Paterson.
The complexity of mean payoff games on graphs.
Theoretical Computer Science, 158:343–359, 1996.

References IV