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

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

1 **Computer Science Department, Université de Mons (UMONS), Belgium**
2 **Département d'Informatique, Université Libre de Bruxelles (U.L.B.), Belgium**

Abstract

Classical analysis of two-player quantitative games involves an adversary (modeling the environment of the system) which is purely antagonistic and asks for strict guarantees while Markov decision processes model systems facing 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. We consider both the mean-payoff value problem and the shortest path problem. In both cases, we show how to decide the existence of finite-memory strategies satisfying the problem and how to synthesize one if one exists. We establish algorithms and we study complexity bounds and memory requirements.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases two-player games on graphs, Markov decision processes, quantitative objectives, synthesis, worst-case and expected value, mean-payoff, shortest path

Digital Object Identifier 10.4230/LIPIcs.STACS.2014.199

1 Introduction

Two-player zero-sum quantitative games [14, 28, 3] and Markov decision processes (MDPs) [24, 5] are two popular formalisms for modeling decision making in adversarial and uncertain environments respectively. In the former, two players compete with opposite goals (zero-sum), and we want strategies for player 1 (the system) that ensure a given *minimal performance against all possible strategies* of player 2 (its environment). In the latter, the system plays against a stochastic model of its environment, and we want strategies that ensure a *good expected overall performance*. Those two models are well studied and simple optimal memoryless strategies exist for classical objectives such as mean-payoff [22, 14, 15] or shortest path [1, 12]. But both models have clear weaknesses: strategies that are good for the worst-case may exhibit suboptimal behaviors in probable situations while strategies that are good for the expectation may be terrible in some unlikely but possible situations.

In practice, we want strategies that both ensure (a) some worst-case threshold no matter how the adversary behaves (i.e., against any arbitrary strategy) and (b) a good expectation

* Work partially supported by European project CASSTING (FP7-ICT-601148). Filiot and Randour are respectively F.R.S.-FNRS research associate and research fellow. Raskin is supported by ERC Starting Grant inVEST (279499).

31st Symposium on Theoretical Aspects of Computer Science (STACS'14).
Editors: Ernst W. Mayr and Natacha Portier; pp. 199–213

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

against the expected behavior of the adversary (given as a stochastic model). We study how to construct such finite-memory strategies. We consider finite memory for player 1 as it can be implemented in practice (as opposed to infinite memory). Player 2 is not restricted in his choice of strategies, but we show that simple strategies suffice. Our problem, the **beyond worst-case synthesis problem**, makes sense for any quantitative measure. We focus on two classical ones: the *mean-payoff*, and the *shortest path*.

Example. Consider the weighted game in Fig. 1 to illustrate the *shortest path* context. Circle states belong to player 1, square states to player 2, integer labels are durations in minutes, and fractions are probabilities that model the expected behavior of player 2. Player 1 wants a strategy to go from "home" to "work" such that "work" is *guaranteed* to be reached within 60 minutes (to avoid missing an important meeting), and player 1 would also like to minimize the expected time to reach "work". The strategy that minimizes the expectation is to take the car (expectation is 33 minutes) but it is excluded as there is a possibility to arrive after 60 minutes (in case of heavy traffic). Bicycle is safe but the expectation of this solution is 45 minutes. We can do better with the following strategy: try to take the train, if the train is delayed three time consecutively, then go back home and take the bicycle. This strategy is safe as it always reaches "work" within 59 minutes and its expectation is $\approx 37,56$ minutes (so better than taking directly the bicycle). Our algorithms are able to decide the existence of (and synthesize) such finite-memory strategies.

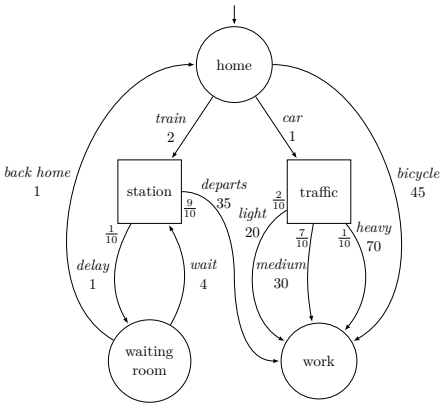

Figure 1 Player 1 wants to minimize its expected time to reach "work", but while ensuring it is less than an hour in all cases.

Contributions. For the mean-payoff, we provide an NP ∩ coNP algorithm (Thm. 7), which would be in P if mean-payoff games were proved to be in P, a long-standing open problem [3, 7]. For the shortest path, we give a pseudo-polynomial time algorithm (Thm. 9), and show that the problem is NP-hard (Thm. 11). For both, synthesized strategies may require up to pseudo-polynomial memory (Thm. 8 and Thm. 10), but accept natural, elegant representations, based on states of the game and simple integer counters. An extended version of this work, including full proofs, can be found in [4].

Related work. Our problems generalize the corresponding problems for two-player zero-sum games and MDPs. In mean-payoff games, optimal memoryless worst-case strategies exist and the best known algorithm is in NP ∩ coNP [14, 28, 3]. For shortest path games, where we consider game graphs with strictly positive weights and try to minimize the cost to target, it can be shown that memoryless strategies also suffice, and the problem is in P. In MDPs, optimal expectation strategies are studied in [24, 15] for both measures: memoryless strategies suffice and they can be computed in P. Our strategies are *strongly risk averse*: they avoid at all cost outcomes below a given threshold (no matter their probability), and inside the set of those *safe* strategies, we maximize expectation. To the best of our knowledge, we are the first to consider such strategies. Other notions of risk have been studied for MDPs: e.g., in [27], the authors want to find policies minimizing the probability (risk) that the total discounted

rewards do not exceed a specified value; in [16], the authors want to achieve a specified value of the long-run limiting average reward at a given probability level (percentile). While those strategies limit risk, they only ensure *low probability* for bad behaviors but not their absence, furthermore, they do not ensure good expectation either. Another body of related work is the study of strategies in MDPs that achieve a trade-off between the expectation and the variance over the outcomes (e.g., [2] for the mean-payoff, [23] for the cumulative reward), giving a statistical measure of the stability of the performance. In our setting, we strengthen this requirement by asking for *strict guarantees on individual outcomes*, while maintaining an appropriate expected payoff.

Future work. Study of other value functions, extension to more general settings (decidable classes of imperfect information games [13], multi-dimension [6, 9], etc), and application to practical cases.

Acknowledgments. We thank G. Latouche and G. Louchard for fruitful discussions about Chernoff bounds in Markov models, and an anonymous reviewer for pointing out interesting related works.

2 Beyond Worst-Case Synthesis

Weighted directed graphs. A *weighted directed graph* is a tuple $\mathcal{G} = (S, E, w)$ where (i) S is the set of vertices, called *states*; (ii) $E \subseteq S \times S$ is the set of directed edges; and (iii) $w\colon E \to \mathbb{Z}$ is the weight function. Given $s \in S$, let $\mathsf{Succ}(s) = \{s' \in S \mid (s, s') \in E\}$ be its set of successors. We assume that for all $s \in S$, $\mathsf{Succ}(s) \neq \emptyset$ (no deadlock). We denote by W the largest absolute weight.

A *play* in $\mathcal{G}$ from an initial state $s_{\mathsf{init}} \in S$ is an infinite sequence of states $\pi = s_0 s_1 s_2 \dots$ such that $s_0 = s_{\mathsf{init}}$ and $(s_i, s_{i+1}) \in E$ for all $i \geq 0$. The *prefix* up to the n-th state of π is the finite sequence $\pi(n) = s_0 s_1 \dots s_n$. We denote its last state by $\mathsf{Last}(\pi(n)) = s_n$. The set of plays of $\mathcal{G}$ is denoted by $\mathsf{Plays}(\mathcal{G})$ and the corresponding set of prefixes is denoted by $\mathsf{Prefs}(\mathcal{G})$. Given a play $\pi \in \mathsf{Plays}(\mathcal{G})$, we denote by $\mathsf{Inf}(\pi) \subseteq S$ the set of states that are visited infinitely often along the play.

Given a function $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$, the *value* of a play π is $f(\pi)$. The *mean-payoff* of a prefix $\rho = s_0 s_1 \dots s_n$ is $\mathsf{MP}(\rho) = \frac{1}{n} \sum_{i=0}^{i=n-1} w((s_i, s_{i+1}))$. For plays, $\mathsf{MP}(\pi) = \liminf_{n\to\infty} \mathsf{MP}(\pi(n))$. Given a graph with strictly positive weights ($w\colon E \to \mathbb{N}_0$) and a target set $T \subseteq S$, the *truncated sum up to* T is $\mathsf{TS}_T\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{N} \cup \{\infty\}$, $\mathsf{TS}_T(\pi = s_0 s_1 s_2 \dots) = \sum_{i=0}^{n-1} w((s_i, s_{i+1}))$, with n the first index such that $s_n \in T$, and $\mathsf{TS}_T(\pi) = \infty$ if π never reaches any state in T.

Probability distributions. Given a finite set A, a (rational) *probability distribution* on A is a function $p\colon A \to [0, 1] \cap \mathbb{Q}$ such that $\sum_{a\in A} p(a) = 1$. We denote the set of probability distributions on A by $\mathcal{D}(A)$. The *support* of the probability distribution p on A is $\mathsf{Supp}(p) = \{a \in A \mid p(a) > 0\}$.

Two-player games. We consider two-player turn-based games and denote the two *players* by $\mathcal{P}_1$ and $\mathcal{P}_2$. A finite *two-player game* is a tuple $G = (\mathcal{G}, S_1, S_2)$ composed of (i) a finite weighted graph $\mathcal{G} = (S, E, w)$; and (ii) a partition of its states S into S_1 and S_2 that resp. denote the sets of states belonging to $\mathcal{P}_1$ and $\mathcal{P}_2$. A prefix $\pi(n)$ of a play π belongs to $\mathcal{P}_i$, $i \in \{1, 2\}$, if $\mathsf{Last}(\pi(n)) \in S_i$. The set of prefixes that belong to $\mathcal{P}_i$ is denoted by $\mathsf{Prefs}_i(G)$.

We sometimes denote by $|G|$ the size of a game, defined as a polynomial function of $|S|$, $|E|$ and $V = \lceil \log_2 W \rceil$.

Strategies. A *strategy* for $\mathcal{P}_i$, $i \in \{1, 2\}$, is a function $\lambda_i \colon \mathsf{Prefs}_i(G) \to \mathcal{D}(S)$ such that for all $\rho \in \mathsf{Prefs}_i(G)$, we have $\mathsf{Supp}(\lambda_i(\rho)) \subseteq \mathsf{Succ}(\mathsf{Last}(\rho))$. A strategy is *pure* if its support is a singleton for all prefixes. A strategy λ_i for $\mathcal{P}_i$ has *finite memory* if it can be encoded by a stochastic finite state machine with outputs, called *stochastic Moore machine*, $\mathcal{M}(\lambda_i) = (\mathsf{Mem}, \mathsf{m}_0, \alpha_\mathsf{u}, \alpha_\mathsf{n})$, where (i) Mem is a finite set of memory elements, (ii) $\mathsf{m}_0 \in \mathsf{Mem}$ is the initial memory element, (iii) $\alpha_\mathsf{u} \colon \mathsf{Mem} \times S \to \mathsf{Mem}$ is the update function, and (iv) $\alpha_\mathsf{n} \colon \mathsf{Mem} \times S_i \to \mathcal{D}(S)$ is the next-action function. If the game is in $s \in S_i$ and $\mathsf{m} \in \mathsf{Mem}$ is the current memory, then the strategy chooses s', the next state of the game, according to the distribution $\alpha_\mathsf{n}(\mathsf{m}, s)$. When the game leaves a state $s \in S$, the memory is updated to $\alpha_\mathsf{u}(\mathsf{m}, s)$. Pure strategies have deterministic next-action functions. A strategy is *memoryless* if $|\mathsf{Mem}| = 1$, i.e., it only depends on the current state of the game.

We resp. denote by $\Lambda_i(G)$ and $\Lambda_i^F(G)$ the sets of general (i.e., possibly randomized and infinite-memory) and finite-memory strategies for player $\mathcal{P}_i$ on the game G. We do not write G in this notation when the context is clear. A play π is said to be *consistent* with a strategy $\lambda_i \in \Lambda_i$ if for all $n \geq 0$ such that $\mathsf{Last}(\pi(n)) \in S_i$, we have $\mathsf{Last}(\pi(n+1)) \in \mathsf{Supp}(\lambda_i(\pi(n)))$.

Markov decisions processes. A finite *Markov decision process* (MDP) is a tuple $\mathrm{P} = (\mathcal{G}, S_1, S_\Delta, \Delta)$ where (i) $\mathcal{G} = (S, E, w)$ is a finite weighted graph, (ii) S_1 and S_Δ define a partition of the set of states S into states of $\mathcal{P}_1$ and *stochastic states*, and (iii) $\Delta \colon S_\Delta \to \mathcal{D}(S)$ is the transition function that, given a stochastic state $s \in S_\Delta$, defines the probability distribution $\Delta(s)$ over the possible successors of s, such that for all states $s \in S_\Delta$, $\mathsf{Supp}(\Delta(s)) \subseteq \mathsf{Succ}(s)$. In contrast to some other classical definitions of MDPs in the literature, we explicitly allow that, for some states $s \in S_\Delta$, $\mathsf{Supp}(\Delta(s)) \subsetneq \mathsf{Succ}(s)$: some edges of the graph $\mathcal{G}$ are assigned probability zero by the transition function. We define the subset of edges $E_\Delta = \{(s_1, s_2) \in E \mid s_1 \in S_\Delta \ Rightarrow s_2 \in \mathsf{Supp}(\Delta(s_1))\}$, representing all edges that either start in a state of $\mathcal{P}_1$, or are chosen with non-zero probability by the transition function Δ. The notions of prefixes belonging to $\mathcal{P}_1$ and of strategies for $\mathcal{P}_1$ are naturally extended to MDPs.

End-components. We define *end-components* (ECs) of an MDP as subgraphs in which $\mathcal{P}_1$ can ensure to stay despite stochastic states [11]. Let $P = (\mathcal{G}, S_1, S_\Delta, \Delta)$ be an MDP, with $\mathcal{G} = (S, E, w)$ its underlying graph. An EC in P is a set $U \subseteq S$ such that (i) the subgraph $(U, E_\Delta \cap (U \times U))$ is strongly connected, with E_Δ defined as before, i.e., stochastic edges with probability zero are treated as non-existent; and (ii) for all $s \in U \cap S_\Delta$, $\mathsf{Supp}(\Delta(s)) \subseteq U$, i.e., in stochastic states, all outgoing edges either stay in U or belong to $E \setminus E_\Delta$ (the probability of leaving U from a state $s \in S_\Delta$ is zero).

Markov chains. A finite *Markov chain* (MC) is a tuple $M = (\mathcal{G}, \delta)$ where (i) $\mathcal{G} = (S, E, w)$ is a finite weighted graph; and (ii) $\delta \colon S \to \mathcal{D}(S)$ is the transition function that, given $s \in S$, defines the distribution $\delta(s)$, such that for all $s \in S$, $\mathsf{Supp}(\delta(s)) \subseteq \mathsf{Succ}(s)$. In an MC, an *event* is a measurable set of plays $\mathcal{A} \subseteq \mathsf{Plays}(\mathcal{G})$. Every event has a uniquely defined probability [26] (Carathéodory's extension theorem induces a unique probability measure on the Borel σ-algebra over $\mathsf{Plays}(\mathcal{G})$). We denote by $\mathbb{P}^M_{s_\mathsf{init}}(\mathcal{A})$ the probability that a play belongs to $\mathcal{A}$ when the MC M starts in $s_\mathsf{init} \in S$ and is executed for an infinite number of

steps. Given a measurable function $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$, we denote by $expect^{M}_{s_{\mathsf{init}}}(f)$ the *expected value* or *expectation* of f over a play starting in s_{init}.

Outcomes. Let $M = (\mathcal{G}, \delta)$ be a Markov chain, with $\mathcal{G} = (S, E, w)$ its underlying graph. Given an initial state $s_{\mathsf{init}} \in S$, we define the set of its possible *outcomes* as

$$\mathsf{Outs}_M(s_{\mathsf{init}}) = \{\pi = s_0 s_1 s_2 \ldots \in \mathsf{Plays}(\mathcal{G}) \mid s_0 = s_{\mathsf{init}} \wedge \forall\, n \in \mathbb{N},\ s_{n+1} \in \mathsf{Supp}(\delta(s_n))\}\,.$$

Let $G = (\mathcal{G}, S_1, S_2)$ be a two-player game, with $\mathcal{G} = (S, E, w)$ its graph. Given two strategies, $\lambda_1 \in \Lambda_1$ and $\lambda_2 \in \Lambda_2$, and an initial state $s_{\mathsf{init}} \in S$, we extend the notion of outcomes as follows:

$$\mathsf{Outs}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2) = \{\pi = s_0 s_1 s_2 \ldots \in \mathsf{Plays}(\mathcal{G}) \mid s_0 = s_{\mathsf{init}} \wedge \pi \text{ is consistent with } \lambda_1 \text{ and } \lambda_2\}\,.$$

When fixing the strategies, we obtain an MC denoted by $G[\lambda_1, \lambda_2]$. This MC is finite if both λ_1 and λ_2 are finite-memory strategies. The outcomes of G and $G[\lambda_1, \lambda_2]$ are not *sensu stricto* of the same nature as the graph of the MC is obtained through the product of the memory elements of the strategies given as Moore machines and the states of the game. Still, there exists a bijection between outcomes of the MC and their *traces* in the initial game, thanks to the projection operator on S. For the sake of readability, we equivalently refer to outcomes and their traces.

Let $P = (\mathcal{G}, S_1, S_\Delta, \Delta)$ be an MDP, with $\mathcal{G} = (S, E, w)$ its graph. Again, we can fix the strategy λ_1 of $\mathcal{P}_1$ and obtain the MC $P[\lambda_1]$. Its set of outcomes starting in $s_{\mathsf{init}} \in S$ is denoted $\mathsf{Outs}_P(s_{\mathsf{init}}, \lambda_1)$. Finally, back to the two-player game G, if we fix the strategy λ_i of only one player $\mathcal{P}_i$, $i \in \{1, 2\}$, we obtain not an MC, but an MDP for the remaining player $\mathcal{P}_{3-i}$. This MDP is denoted by $G[\lambda_i]$.

Subgraphs and subgames. Given a graph $\mathcal{G} = (S, E, w)$ and a subset $A \subseteq S$, we define the induced subgraph $\mathcal{G} \restriction A = (A, E \cap (A \times A), w)$ naturally. Subgames are defined similarly by considering their induced subgraphs: they are only properly defined if the induced subgraphs contain no deadlock.

Worst-case synthesis. Given a game $G = (\mathcal{G}, S_1, S_2)$, with $\mathcal{G} = (S, E, w)$, an initial state $s_{\mathsf{init}} \in S$, a function $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$, and a threshold $\mu \in \mathbb{Q}$, the *worst-case threshold problem* asks to decide if $\mathcal{P}_1$ has a strategy $\lambda_1 \in \Lambda_1$ such that $\forall\, \lambda_2 \in \Lambda_2, \forall\, \pi \in \mathsf{Outs}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2)$, $f(\pi) \geq \mu$. For the mean-payoff, pure memoryless optimal[1] strategies exist for both players [22, 14]. Hence, deciding the winner is in NP $\cap$ coNP, and it was furthermore shown to be in UP $\cap$ coUP [28, 21, 18]. Whether the problem is in P is a long-standing open problem [3, 7]. For the shortest path (truncated sum value function), it can be shown that the decision problem takes polynomial time, as a winning strategy of $\mathcal{P}_1$ should avoid all cycles (because they yield strictly positive costs), hence usage of attractors and comparison of the worst possible sum of costs with the threshold suffices.

[1] A strategy for $\mathcal{P}_i$, $i \in \{1, 2\}$, is said to be *optimal* if it ensures a threshold higher or equal to the threshold ensured by any other strategy of the same player. The threshold ensured by an optimal strategy is called the *optimal value*.

Expected value synthesis. Given an MDP $P = (\mathcal{G}, S_1, S_\Delta, \Delta)$, with $\mathcal{G} = (S, E, w)$, an initial state $s_{\mathsf{init}} \in S$, a measurable function $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$, and a threshold $\nu \in \mathbb{Q}$, the *expected value threshold problem* asks to decide if $\mathcal{P}_1$ has a strategy $\lambda_1 \in \Lambda_1$ such that $\mathbb{E}_{s_{\mathsf{init}}}^{P[\lambda_1]}(f) \geq \nu$. Optimal expected mean-payoff in MDPs can be achieved by memoryless strategies, and the corresponding decision problem can be solved in polynomial time through linear programming [15]. The truncated sum value function has been studied in the literature under the name of *shortest path problem*: again, memoryless strategies suffice to be optimal and the problem is solvable in polynomial time [1, 12].

Beyond worst-case synthesis. We study the synthesis of finite-memory strategies that ensure, *simultaneously*, a value greater than a threshold μ in the worst-case (i.e., against any strategy of the adversary), and an expected value greater than a threshold ν against a given finite-memory stochastic model of the adversary (e.g., representing commonly observed behavior of the environment).

▶ **Definition 1.** Given a game $G = (\mathcal{G}, S_1, S_2)$, with $\mathcal{G} = (S, E, w)$, 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 $f\colon \mathsf{Plays}(\mathcal{G}) \to \mathbb{R} \cup \{-\infty, \infty\}$, and two 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.

We take the convention to ask for values strictly greater than the thresholds to ease the formulation of our results in the following. Indeed, for some thresholds, it is possible to synthesize strategies that ensure ε-close values, for any $\varepsilon > 0$, while it is not feasible to achieve the exact threshold. Notice that we can assume $\nu > \mu$, otherwise the problem reduces to the classical worst-case analysis.

3 Mean-Payoff Value Function

We present algorithm BWC_MP (Alg. 1) for the BWC synthesis problem and we highlight its cornerstones. Results on memory requirements follow. A sample game is presented in Fig. 2.

Inputs and outputs. The algorithm takes as input: a game G^i, a finite-memory stochastic model of the adversary λ_2^i, a worst-case threshold μ^i, an expected value threshold ν^i, and an initial state s_{init}^i. Its output is Yes if and only if there exists a finite-memory strategy of $\mathcal{P}_1$ satisfying the BWC problem. We present how to synthesize such a satisfying strategy in the following.

Preprocessing. The first part of the algorithm (lines 1-7) is the preprocessing of the game G^i and the stochatic model λ_2^i given as inputs in order to apply the second part of the algorithm (lines 8–11) on a modified game G and stochastic model $\lambda_2^{\mathsf{stoch}}$, simpler to manipulate. We ensure that the answer to the BWC problem on the modified game is Yes if and only if it is also Yes on the input game, and that winning strategies of $\mathcal{P}_1$ in G can be transferred to winning strategies in G^i.

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

Require: $G^i = \left(\mathcal{G}^i, S_1^i, S_2^i\right)$ a game, $\mathcal{G}^i = \left(S^i, E^i, w^i\right)$ 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 $s_{\mathsf{init}}^i \in S^i$ 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_{init}^i, for the thresholds pair (μ^i, ν^i) and the mean-payoff value function

{*Preprocessing*}

1: **if** $\mu^i \neq 0$ **then** define $\forall\, e \in E^i$, $w_{\mathsf{new}}^i(e) := b \cdot w^i(e) - a$, and consider thresholds $(0, \nu := b \cdot \nu^i - a)$

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

3: **if** $s_{\mathsf{init}}^i \notin S_{WC}$ **then return** NO **else**

4: Let $G^w := G^i \restriction S_{WC}$ be the subgame induced by worst-case winning states

5: 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 $s_{\mathsf{init}} := (s_{\mathsf{init}}^i, \mathsf{m}_0)$ the corresponding initial state

6: Let $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M(G)$ be the memoryless transcription of λ_2^i on G

7: 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*}

8: Compute $\mathcal{U}_{\mathrm{w}}$ the set of maximal winning end-components of P

9: Build $P' = (\mathcal{G}', S_1, S_\Delta, \Delta)$, where $\mathcal{G}' = (S, E, w')$ and w' is defined s.t. $\forall\, e = (s_1, s_2) \in E$, $w'(e) := w(e)$ if $\exists\, U \in \mathcal{U}_{\mathrm{w}}$ s.t. $\{s_1, s_2\} \subseteq U$, or $w'(e) := 0$ otherwise

10: Compute the maximal expected value ν^* from s_{init} in P'

11: **if** $\nu^* > \nu$ **then return** YES **else return** NO

First, we modify the weights of $\mathcal{G}^i$ in order to consider the equivalent BWC problem with thresholds $(0, \nu)$. This classical trick is used to get rid of explicitely considering the worst-case threshold in the following, as it is equal to zero. Second, observe that any strategy that is winning for the BWC problem must also be winning for the classical worst-case problem. Such a strategy cannot allow visits of any state from which $\mathcal{P}_1$ cannot ensure winning against an antagonistic adversary: entering such a state would be losing no matter the prefix. Indeed, mean-payoff is prefix-independent: for all $\rho \in \mathsf{Prefs}(\mathcal{G})$, $\pi \in \mathsf{Plays}(\mathcal{G})$ we have that $\mathsf{MP}(\rho \cdot \pi) = \mathsf{MP}(\pi)$. Hence, we reduce our study to G^w, the subgame induced by worst-case winning states in G^i (lines 2 and 4). Obviously, if from the initial state s_{init}^i, $\mathcal{P}_1$ cannot win the worst-case problem, then the answer to the BWC problem is NO (lines 3). Third, we build the game G which states are defined by the product of the states of G^w and the memory elements of $\mathcal{M}(\lambda_2^i)$ (line 5). Intuitively, we expand the initial game by integrating the memory of the stochastic model of $\mathcal{P}_2$ in the graph. This does not modify the power of the adversary. Fourth, the finite-memory stochastic model λ_2^i on G^i clearly translates to a memoryless stochastic model $\lambda_2^{\mathsf{stoch}}$ on G (line 6). This helps us obtain elegant proofs for the second part of the algorithm.

Analysis of end-components. The second part of the algorithm (lines 8-11) operates on a game G such that from all states, $\mathcal{P}_1$ has a strategy to achieve a strictly positive mean-payoff (recall $\mu = 0$). We consider the MDP $P = G[\lambda_2^{\mathsf{stoch}}]$ and notice that the underlying graphs of G and P are the same thanks to $\lambda_2^{\mathsf{stoch}}$ being memoryless. The next steps rely on the analysis of *end-components* in the MDP, i.e., strongly connected subgraphs in which $\mathcal{P}_1$ can ensure to stay when playing against the stochastic adversary. The motivation to this analysis arises from the following well-known result.

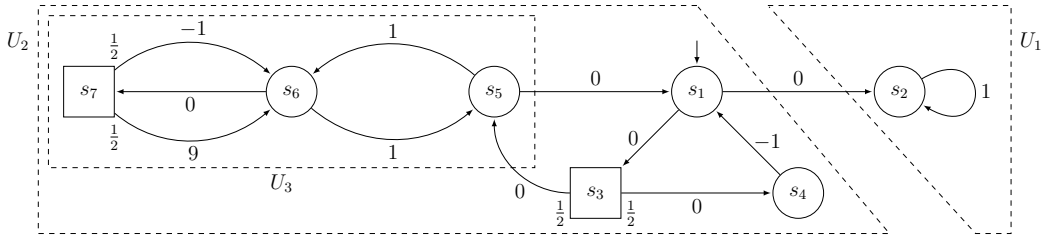

Figure 2 End component U_2 is losing. The set of maximal winning ECs is $\mathcal{U}_{\mathrm{w}} = \{U_1, U_3\}$. The set of winning ECs is $\mathcal{W} = \mathcal{U}_{\mathrm{w}} \cup \{\{s_5, s_6\}, \{s_6, s_7\}\}$.

▶ **Lemma 2** ([10, 11]). *Let $P = (\mathcal{G}, S_1, S_\Delta, \Delta)$ be an MDP, $\mathcal{G} = (S, E, w)$ its underlying graph, $\mathcal{E} \subseteq 2^S$ the set of its ECs, $s_{\mathsf{init}} \in S$ the initial state, and $\lambda_1 \in \Lambda_1(P)$ an arbitrary strategy of $\mathcal{P}_1$. Then,*

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

Recall that the mean-payoff is prefix-independent, therefore the value of any outcome only depends on the states that are seen infinitely often. Hence, the expected mean-payoff in $P[\lambda_1]$ depends *uniquely* on the value obtained in the ECs. Inside an EC, we can compute the maximal expected value that can be achieved by $\mathcal{P}_1$, and this value is the same in all states of the EC [15].

To satisfy the expected value requirement (eq. (2)), an acceptable strategy has to favor reaching ECs with a sufficient expectation, but under the constraint that it also ensures the worst-case requirement (eq. (1)): some ECs with high expected values may still need to be avoided because they do not permit to guarantee this constraint. This is the cornerstone of the classification of ECs that follows.

Classification of end-components. Let $\mathcal{E} \subseteq 2^S$ be the set of all ECs in P. By definition, only edges in E_Δ, as defined in Sect. 2, are involved to determine which sets of states form an EC in P. For any EC $U \in \mathcal{E}$, there may exist edges from $E \setminus E_\Delta$ starting in U, such that $\mathcal{P}_2$ can force leaving U when using an arbitrary strategy. Still these edges will never be used by the stochastic model $\lambda_2^{\mathsf{stoch}}$. This remark is important to the definition of strategies of $\mathcal{P}_1$ that guarantee the worst-case requirement, as $\mathcal{P}_1$ needs to be able to react to the hypothetic use of such an edge. It is also the case *inside* an EC.

Now, we want to consider the ECs in which $\mathcal{P}_1$ can ensure that the worst-case requirement will be fulfilled (without having to leave the EC): we call them *winning*. The others need to be eventually avoided, hence have zero impact on the expectation of a finite-memory strategy satisfying the BWC problem. So we call the latter *losing*. Formally, let $U \in \mathcal{E}$ be an EC. It is *winning* if, in the subgame $G \restriction U$, from all states, $\mathcal{P}_1$ has a strategy to ensure a strictly positive mean-payoff against any strategy of $\mathcal{P}_2$ *that only chooses edges which are assigned non-zero probability by* $\lambda_2^{\mathsf{stoch}}$, or equivalently, edges in E_Δ. We denote $\mathcal{W} \subseteq \mathcal{E}$ the set of such ECs. Non-winning ECs are *losing*: in those, whatever the strategy of $\mathcal{P}_1$ played against the stochastic model $\lambda_2^{\mathsf{stoch}}$ (or any strategy with the same support), there exists at least one outcome for which the mean-payoff is not strictly positive (even if its probability is zero, its mere existence is not acceptable for the worst-case requirement).

Maximal winning end-components. Based on these definitions, observe that line 8 of algorithm BWC_MP does not actually compute the set $\mathcal{W}$ containing all winning ECs, but

the set $\mathcal{U}_{\mathrm{w}} \subseteq \mathcal{W}$, defined as $\mathcal{U}_{\mathrm{w}} = \{U \in \mathcal{W} \mid \forall U' \in \mathcal{W}, U \subseteq U' \Rightarrow U = U'\}$, i.e., the set of *maximal* winning ECs.

The intuition on *why we can* restrict to this subset is as follows. If an EC $U_1 \in \mathcal{W}$ is included in another EC $U_2 \in \mathcal{W}$, then the maximal expected value achievable in U_2 is at least equal to the one achievable in U_1. Indeed, $\mathcal{P}_1$ can reach U_1 with probability one (by virtue of U_2 being an EC and $U_1 \subseteq U_2$) and stay in it with probability one (by virtue of U_1 being an EC): the expectation is equal to what can be obtained in U_1 thanks to the prefix-independence. Hence it is sufficient to consider maximal winning ECs in our computations.

As for *why we do it*, the complexity gain is critical. The number of winning ECs can be exponential in the size of the input, as $|\mathcal{W}| \leq |\mathcal{E}| \leq 2^{|S|}$. Yet, the number of maximal ones is bounded by $|\mathcal{U}_{\mathrm{w}}| \leq |S|$ as they are disjoint by definition: for any two winning ECs with a non-empty intersection, their union is also an EC, and is still winning because $\mathcal{P}_1$ can essentially stick to the EC of his choice.

▶ **Lemma 3.** *The set $\mathcal{U}_{\mathrm{w}}$ of maximal winning ECs can be computed in NP ∩ coNP.*

Roughly sketched, our recursive subalgorithm computes the maximal EC decomposition of an MDP (in polynomial time [8]), then checks for each EC U in the decomposition (their number is polynomial) if U is winning or not, which requires a call to an NP ∩ coNP oracle solving the worst-case threshold problem on the corresponding subgame. If U is losing, it may still be the case that a sub-EC $U' \subsetneq U$ is winning. We recurse on the MDP reduced to U, where states from which $\mathcal{P}_2$ can win in U have been removed: the stack of calls is at most polynomial.

Ensure reaching winning end-components. We now refine Lemma 2 for *finite-memory* strategies that *satisfy* the BWC problem.

▶ **Lemma 4.** *Let $G = (\mathcal{G}, S_1, S_2)$ be a two-player game, $\lambda_2^{\mathsf{stoch}} \in \Lambda_2^M$ a memoryless stochastic model of $\mathcal{P}_2$, $P = G[\lambda_2^{\mathsf{stoch}}]$ the resulting MDP and $s_{\mathsf{init}} \in S$ the initial state. 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*

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

Equivalently, the probability that $\mathsf{Inf}(\pi) = U$ for some $U \in \mathcal{E} \setminus \mathcal{W}$ is zero. The equality is crucial. It may be the case, with non-zero probability, that $\mathsf{Inf}(\pi) = U' \subsetneq U$ for some $U' \in \mathcal{W}$ and $U \in \mathcal{E} \setminus \mathcal{W}$ (hence the recursive algorithm to compute $\mathcal{U}_{\mathrm{w}}$). It is clear that $\mathcal{P}_1$ should not visit all the states of a losing EC forever, as then he would not be able to guarantee the worst-case threshold.

Our goal is to build an MDP P', sharing the same graph and ECs as P, such that an optimal strategy for the expectation problem on P' will naturally avoid losing ECs and prescribe which winning ECs are the most interesting to reach for a BWC strategy on the initial game G and MDP P. The expected value obtained in P by any BWC satisfying strategy of $\mathcal{P}_1$ only depends on the weights of edges involved in winning ECs, or equivalently, in maximal winning ECs (as the set of outcomes that are not trapped in them has measure zero). We build P' by modifying the weights of P (line 9): we keep them unchanged in edges that belong to some $U \in \mathcal{U}_{\mathrm{w}}$, and we put them to zero everywhere else, which is lower than the expectation granted by winning ECs (strictly positive by definition).

Reach the highest valued winning end-components. We compute the maximal expected value ν^* that can be achieved by $\mathcal{P}_1$ in the MDP P', from the initial state (line 10). It takes polynomial time and memoryless strategies suffice to achieve the maximal value [15]. Basically, we build a strategy that favors reaching ECs with high associated expectations in P'. We argue that the ECs reached with probability one by this strategy are necessarily winning ECs. Clearly, if a winning EC is reachable instead of a losing one, it will be favored because of the weights definition in P' (expectation is strictly higher in winning ECs). It remains to check if winning ECs are reachable with probability one from any state in S. They are, due to the preprocessing. Indeed, all states are winning for the worst-case requirement. Clearly, from any state in $A = S \setminus \bigcup_{U \in\ ecsSet} U$, $\mathcal{P}_1$ cannot ensure to stay in A (otherwise it would form an EC) and must be able to win the worst-case from reached ECs. Now for any state in $B = \bigcup_{U \in \mathcal{E}} U \setminus \bigcup_{U \in \mathcal{U}_\mathrm{w}} U$, i.e., states in losing ECs and not in any sub-EC winning, $\mathcal{P}_1$ cannot win the worst-case by staying in B, by definition of losing EC. Since $\mathcal{P}_1$ can ensure the worst-case by hypothesis, he must be able to reach $C = \bigcup_{U \in \mathcal{U}_\mathrm{w}} U$ from any state in B, as claimed.

Inside winning end-components. Based on that, winning ECs are reached with probability one. Consider what we can say about such ECs assuming that $E_\Delta = E$, i.e., if all possible edges are mapped to non-zero probabilities. We establish a finite-memory *combined strategy* of $\mathcal{P}_1$ that ensures (i) worst-case satisfaction while yielding (ii) an expected value ε-close to the maximal expectation inside the component. For two well-chosen parameters $K, L \in \mathbb{N}$, it is informally defined as follows: in phase *(a)*, play a memoryless expected value optimal strategy for K steps and memorize $\mathsf{Sum} \in \mathbb{Z}$, the sum of weights along these steps; in phase *(b)*, if $\mathsf{Sum} > 0$, go to *(a)*, otherwise play a memoryless worst-case optimal strategy for L steps, then go to *(a)*. In phases *(a)*, $\mathcal{P}_1$ tries to increase its expectation and approach its optimal one, while in phase *(b)*, he compensates, if needed, losses that occured in phase *(a)*. The two memoryless strategies exist on the subgame induced by the EC: by definition of ECs, based on E_Δ, the stochastic model of $\mathcal{P}_2$ will never be able to force leaving the EC against the combined strategy. A key result of our paper is the existence of values for K and L such that (i) and (ii) are verified, as stated in the next theorem.

▶ **Theorem 5.** *Inside a WEC with $\nu^* \in \mathbb{Q}$ the maximal expectation achievable by $\mathcal{P}_1$, for all $\varepsilon > 0$, there exists a finite-memory strategy of $\mathcal{P}_1$ that satisfies the BWC problem for thresholds* $(0, \nu^* - \varepsilon)$.

We see plays as sequences of periods, each starting with phase *(a)*. First, for any K, we can define $L(K)$ such that any period composed of phases *(a)* + *(b)* ensures a mean-payoff at least $1/(K + L) > 0$. Periods containing only phase *(a)* trivially induce a mean-payoff at least $1/K$. Both rely on the weights being integers. As the length of any period is bounded, the inequality remains strict for the mean-payoff of any play, granting (i). Now, consider parameter K. Clearly, when $K \to \infty$, the expectation over a phase *(a)* tends to the optimal one. Nevertheless, phases *(b)* also contribute to the overall expectation of the combined strategy, and (in general) lower it so that it is strictly less than the optimal for any $K, L \in \mathbb{N}$. Hence to prove (ii), we not only need that the probability of playing phase *(b)* decreases when K increases, but also that it decreases faster than the increase of L, needed to ensure (i), so that overall, the contribution of phases *(b)* tends to zero when $K \to \infty$. This is indeed the case and can be proved using results bounding the probability of observing a mean-payoff significantly (more than some ε) different than the optimal expectation along a phase *(a)* of length $K \in \mathbb{N}$: this probability decreases exponentially when K increases [25, 19] (related to

the notions of Chernoff bounds and Hoeffding's inequality in MCs), while L only needs to be polynomial in K.

Now, consider what happens if $E_\Delta \subsetneq E$. If $\mathcal{P}_2$ uses an arbitrary strategy, he can take edges of probability zero, i.e., in $E \setminus E_\Delta$, either staying in the EC, or leaving it. In both cases, this must be taken into account in order to satisfy eq. (1) as it may involve dangerous weights (recall that zero-probability edges are not considered when an EC is classified as winning or not). Fortunately, if this were to occur, $\mathcal{P}_1$ could switch to a worst-case winning memoryless strategy, which exists in all states thanks to the preprocessing (line 4). This has no impact on the expectation as it occurs with probability zero against $\lambda_2^{\mathsf{stoch}}$. The strategy to follow in winning ECs adds this reaction procedure to the combined strategy: we call it the *witness-and-secure strategy*.

Global strategy synthesis. In summary, losing ECs should be avoided and will be by a strategy that optimizes the expectation on the MDP P'; in winning ECs, $\mathcal{P}_1$ can obtain the expectation of the EC (at some arbitrarily small ε close) *and* ensure the worst-case threshold. We finally compare the value ν^* with the threshold ν (line 11): (i) if $\nu^* > \nu$, there exists a finite-memory strategy satisfying the BWC problem, and (ii) if not, there does not exist such a strategy.

▶ **Lemma 6.** *Algorithm* BWC_MP *is correct and complete.*

To prove (i), we establish a finite-memory strategy in G, called *global strategy*, of $\mathcal{P}_1$ that ensures a strictly positive mean-payoff against any antagonistic adversary, and ensures an expected mean-payoff ε-close to ν^* (hence, strictly greater than ν) against the stochastic adversary modeled by $\lambda_2^{\mathsf{stoch}}$ (i.e., in P). The intuition is as follows. We play the memoryless optimal strategy of the MDP P' for a sufficiently long time, defined by a parameter $N \in \mathbb{N}$, in order to be with probability close to one in a winning EC (the convergence is exponential by results on absorption times in MCs [20]). Then, if inside a winning EC, we switch to the witness-and-secure strategy which ensures both thresholds. If not yet in a winning EC, we switch to a worst-case winning strategy in G, existing by hypothesis. Thus the mean-payoff of plays that do not reach winning ECs is strictly positive. Since in winning ECs we are ε-close to the maximal expected value of the EC, we conclude that it is possible to play the optimal expectation strategy of MDP P' for sufficiently long to obtain an overall expected value which is arbitrarily close to ν^*, and still guarantee the worst-case threshold in all outcomes. To prove (ii), it suffices to understand that only ECs have an impact on the expectation, and that losing ECs cannot be used forever without endangering the worst-case requirement. Given a winning strategy on G, we can build a corresponding winning strategy on G^i by reintegrating the memory elements of the Moore machine in the memory of the strategy of $\mathcal{P}_1$.

Complexity bounds. The input size depends on the sizes of the game and the Moore machine for the stochastic model, and the encodings of weights and thresholds. All computations require (deterministic) polynomial time except for external calls solving the worst-case threshold problem, which is in NP ∩ coNP [28, 21] and not known to be in P. Hence, the overall complexity is in NP ∩ coNP and may collapse to P if the worst-case problem were to be proved in P: the BWC framework for mean-payoff surprisingly provides additional modeling power without negative impact on the complexity class. We establish that the BWC problem is at least as difficult as the worst-case problem thanks to a polynomial time reduction from the latter to the former. Thus, membership to NP ∩ coNP can be seen as optimal regarding our current knowledge of the worst-case problem.

▶ **Theorem 7.** *The beyond worst-case problem for the mean-payoff value function is in NP ∩ coNP and at least as hard as mean-payoff games.*

Memory requirements. The global strategy suffices if satisfaction of the BWC problem is possible. All the involved strategies (global, witness-and-secure, combined) are alternations between pure memoryless strategies, based on parameters N, K and $L \in \mathbb{N}$, which only need to be polynomial in the size of the game and the stochastic model, and in the values, granting the upper bound of Thm. 8. This bound is tight as polynomial memory in the value of weights is needed in general. Consider a family of games, $(G(X))_{X\in\mathbb{N}_0}$, based on the subgame $G \restriction U_3$ in Fig. 2, but with weights $-X$ and $X+5$ instead of -1 and 9 respectively. When choosing $\mu = 0$ and $\nu \in]1, 5/4[$, the BWC problem is satisfiable and it cannot be achieved by the memoryless strategy that always chooses edge (s_6, s_5). It is thus mandatory to choose (s_6, s_7) infinitely often in order to win. Moreover, after some point, everytime this edge is chosen, a satisfying strategy must *eventually* counteract the potential negative weight $-X$ by taking edge (s_1, s_2) for $\lfloor X/2 \rfloor + 1$ times. Hence polynomial memory in W is needed.

▶ **Theorem 8.** *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.*

4 Truncated Sum Value Function - Shortest Path Problem

Let us consider a game graph such that $w\colon E \to \mathbb{N}_0$ assigns *strictly positive* integer weights to all edges, and a target set $T \subseteq S$ that $\mathcal{P}_1$ wants to reach with a path of bounded value. In other words, we study the BWC problem for the *shortest path* [1, 12]. More precisely, given an initial state $s_{\mathsf{init}} \in S$, the goal of $\mathcal{P}_1$ is to ensure to reach T with a path of *truncated sum* strictly lower than $\mu \in \mathbb{N}$ against all possible behaviors of $\mathcal{P}_2$ while guaranteeing, at the same time, an expected cost to target strictly lower than $\nu \in \mathbb{Q}$ against the stochastic model of the adversary specified by the stochastic Moore machine $\mathcal{M}(\lambda_2^{\mathsf{stoch}})$. Regarding Def. 1, the inequalities are reversed. Hence we assume $\nu < \mu$.

A pseudo-polynomial time algorithm. First, we construct, from the original game G and the worst-case threshold μ, a new game G_μ such that there is a bijection between the strategies of $\mathcal{P}_1$ in G_μ and the strategies of $\mathcal{P}_1$ in the original game G that are winning for the worst-case requirement: we unfold the original graph $\mathcal{G}$, tracking the current value of the truncated sum *up to the worst-case threshold* μ, and integrating this value in the states of an expanded graph $\mathcal{G}'$. In the corresponding game G', we compute the set of states R from which $\mathcal{P}_1$ can reach the target set with cost lower than μ and we define the subgame $G_\mu = G' \restriction R$ such that any path in the graph of G_μ satisfies the worst-case requirement. Second, from G_μ and the stochastic Moore machine $\mathcal{M}(\lambda_2^{\mathsf{stoch}})$, we construct an MDP in which we search for a *playerOne* strategy that ensures reachability of T with an expected cost strictly lower than ν. If it exists, it is guaranteed that it will also satisfy the worst-case requirement against any strategy of $\mathcal{P}_2$ thanks to the bijection evoked earlier.

▶ **Theorem 9.** *The beyond worst-case problem for the shortest path can be solved in pseudo-polynomial time: polynomial in the size of the underlying 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.*

Memory requirements. The construction of Thm. 9 yields an upper bound that is polynomial in the size of the game and the stochastic model, and in the value of the worst-case threshold. Indeed, the synthesized strategy is memoryless in the MDP P that is obtained by taking the product of the expanded game G_μ, such that $|G_\mu| \leq |G| \cdot (\mu + 1)$, with the Moore machine $\mathcal{M}(\lambda_2^{\mathsf{stoch}})$.

We exhibit a family of games (Fig. 3) for which winning requires memory linear in μ, proving that the pseudo-polynomial bound is tight. Let $\mu \in \{13 + k \cdot 4 \mid k \in \mathbb{N}\}$. From s_1, $\mathcal{P}_1$ can ensure reaching the target set $T = \{s_3\}$ at a guaranteed cost of $\lfloor \frac{\mu}{2} \rfloor$. Yet, in order to *minimize* the expected cost of reaching T, $\mathcal{P}_1$ should try to reach it via state s_2, as the cost will be diminished. Hence, $\mathcal{P}_1$ should play edge (s_1, s_2) repeatedly, up to the point where playing (s_1, s_3) becomes mandatory to preserve the worst-case requirement (i.e., when the running sum of weights becomes equal to $\lfloor \frac{\mu}{2} \rfloor$ as the total cost for the worst outcome will be $2 \cdot \lfloor \frac{\mu}{2} \rfloor < \mu$). To implement this strategy, $\mathcal{P}_1$ has to play (s_1, s_2) exactly $\lfloor \frac{\mu}{4} \rfloor$ times and then switch to (s_1, s_3). This requires memory linear in the value μ. The expected value threshold ν can be chosen sufficiently low so that $\mathcal{P}_1$ is compelled to use this optimal strategy to satisfy the BWC problem.

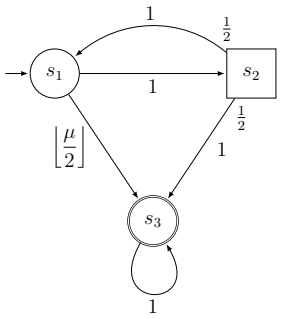

Figure 3 Family of games requiring linear memory in μ.

▶ **Theorem 10.** *Memory of pseudo-polynomial size may be necessary and is always sufficient to satisfy the BWC problem for the shortest path: polynomial in the size of the game and the stochastic model, and polynomial in the worst-case threshold value.*

NP-hardness of the decision problem. We establish that it is very unlikely that a truly-polynomial (i.e., also polynomial in the size of the encoding of the worst-case threshold) time algorithm exists, as the decision problem is NP-hard. Actually, it is likely that the problem is not in NP at all, since we prove a reduction from the K^{th} *largest subset problem* which is known to be NP-hard and commonly thought to be outside NP as natural certificates for the problem are larger than polynomial [17].

The K^{th} largest subset problem is as follows. 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. The reduction is as follows. We build a game composed of two gadgets. The *random subset selection gadget* stochastically generates paths representing subsets of A, with the property that all subsets are equiprobable. The *choice gadget* follows. In it, $\mathcal{P}_1$ decides either to go to a state s_e, which leads to lower expectations but may be dangerous for the worst-case requirement, or to go to a state s_{wc}, always safe with regard to the worst-case but inducing an higher expected cost. The crux of the proof is to define values of the thresholds and the weights such that an optimal (i.e., minimizing the expectation while guaranteeing a given worst-case threshold) strategy for $\mathcal{P}_1$ consists in choosing s_e only when the generated subset $C \subseteq A$ satisfies $h(C) \leq L$, as asked by the K^{th} largest subset problem; and such that this strategy satisfies the BWC problem if and only if there exist K distinct subsets that verify this bound, i.e., if and only if the answer to the K^{th} largest subset problem is Yes.

▶ **Theorem 11.** *The beyond worst-case problem for the shortest path is NP-hard.*

References

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

2 T. Brázdil, K. Chatterjee, V. Forejt, and A. Kucera. Trading performance for stability in Markov decision processes. In *Proc. of LICS*, pages 331–340. IEEE Computer Society, 2013.

3 L. Brim, J. Chaloupka, L. Doyen, R. Gentilini, and J.-F. Raskin. Faster algorithms for mean-payoff games. *Formal Methods in System Design*, 38(2):97–118, 2011.

4 V. Bruyère, E. Filiot, M. Randour, and J.-F. Raskin. Meet your expectations with guarantees: beyond worst-case synthesis in quantitative games. *CoRR*, abs/1309.5439, 2013. `http://arxiv.org/abs/1309.5439`.

5 K. Chatterjee and L. Doyen. Games and Markov decision processes with mean-payoff parity and energy parity objectives. In *Proc. of MEMICS*, LNCS. Springer, 2011.

6 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.

7 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.

8 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.

9 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.

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

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

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

13 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.

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

15 J. Filar and K. Vrieze. *Competitive Markov decision processes*. Springer, 1997.

16 J.A. Filar, D. Krass, and K.W. Ross. Percentile performance criteria for limiting average Markov decision processes. *Transactions on Automatic Control*, pages 2–10, 1995.

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

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

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

20 C.M. Grinstead and J.L. Snell. *Introduction to probability*. American Mathematical Society, 1997.

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

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

23 S. Mannor and J.N. Tsitsiklis. Mean-variance optimization in Markov decision processes. In *Proc. of ICML*, pages 177–184. Omnipress, 2011.

24 M.L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

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

26 M.Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proc. of FOCS*, pages 327–338. IEEE Computer Society, 1985.

27 C. Wu and Y. Lin. Minimizing risk models in Markov decision processes with policies depending on target values. *Journal of Mathematical Analysis and Applications*, 231(1):47–67, 1999.

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