Form Methods Syst Des (2017) 50:207–248
DOI 10.1007/s10703-016-0262-7

Percentile queries in multi-dimensional Markov decision processes

Mickael Randour[1] · **Jean-François Raskin**[1] · **Ocan Sankur**[2]

Published online: 5 January 2017

Abstract Markov decision processes (MDPs) with multi-dimensional weights are useful to analyze systems with multiple objectives that may be conflicting and require the analysis of trade-offs. We study the complexity of percentile queries in such MDPs and give algorithms to synthesize strategies that enforce such constraints. Given a multi-dimensional weighted MDP and a quantitative payoff function f, thresholds v_i (one per dimension), and probability thresholds α_i, we show how to compute a single strategy to enforce that for all dimensions i, the probability of outcomes ρ satisfying $f_i(\rho) \geq v_i$ is at least α_i. We consider classical quantitative payoffs from the literature (sup, inf, lim sup, lim inf, mean-payoff, truncated sum, discounted sum). Our work extends to the quantitative case the multi-objective model checking problem studied by Etessami et al. (Log Methods Comput Sci 4(4), 2008) in unweighted MDPs.

Keywords Markov decision processes · Quantitative objectives · Percentile queries

M. Randour is an F.R.S.-FNRS Postdoctoral Researcher, J.-F. Raskin is supported by ERC Starting Grant (279499: inVEST). Work partly supported by European project CASSTING (FP7-ICT-601148).

✉ Mickael Randour
mickael.randour@ulb.ac.be

Jean-François Raskin
jraskin@ulb.ac.be

Ocan Sankur
ocan.sankur@irisa.fr

[1] Département d'Informatique, Université libre de Bruxelles (ULB), CP 212, Boulevard du Triomphe, 1050 Brussels, Belgium

[2] CNRS, Irisa, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France

1 Introduction

Markov decision processes (MDPs) are central mathematical models for reasoning about (optimal) strategies in *uncertain environments*. For example, if rewards (given as numerical values) are assigned to actions in an MDP, we can search for a strategy (policy) that resolves the nondeterminism in a way that the *expected mean reward* of the actions taken by the strategy over time is maximized. See for example [27] for a solution to this problem. If we are risk-averse, we may want to search instead for strategies that ensure that the mean reward over time is larger than a given value with a high probability, i.e., a probability that exceeds a given threshold. See for example [19] for a solution.

Recent works are exploring several natural extensions of those problems. First, there is a series of works that investigate MDPs with multi-dimensional weights [6,14] rather than single-dimensional as it is traditionally the case. Multi-dimensional MDPs are useful to analyze systems with *multiple objectives* that are potentially conflicting and make necessary the analysis of trade-offs. For instance, we may want to build a control strategy that both ensures some good quality of service and minimizes the energy consumption. Second, there are works that aim at synthesizing strategies enforcing *richer properties*. For example, we may want to construct a strategy that both ensures some minimal threshold with certainty (or probability one) and a good expectation [7]. An illustrative survey of such extensions can be found in [29].

Our paper participates in this general effort by providing algorithms and complexity results on the synthesis of strategies that enforce *multiple percentile constraints*. A *multi-percentile query* and the associated synthesis problem is as follows: given a multi-dimensionally weighted MDP M and an initial state s_{init}, synthesize a strategy σ such that it satisfies the conjunction of q constraints:

$$\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\left[f_{l_i} \geq v_i\right] \geq \alpha_i .$$

where each l_i refers to a dimension of the weight vectors, each v_i is a value threshold, and α_i is a probability threshold, and f is a payoff function. Each constraint i expresses that the strategy ensures probability at least α_i to obtain payoff at least v_i in dimension l_i.

We consider seven payoff functions: sup, inf, limsup, liminf, mean-payoff, truncated sum and discounted sum. This wide range covers most classical functions: our exhaustive study provides a *complete picture* for the new multi-percentile framework and we focus on establishing meta-theorems and connections whenever possible. Some of our results are obtained by reduction to the previous work of [18], but for mean-payoff, truncated sum and discounted sum, that are *non-regular payoffs*, we need to develop original techniques.

Let us consider some examples. In an MDP that models a stochastic shortest path problem, we may want to obtain a strategy that ensures that the probability to reach the target within d time units exceeds 50%: this is a single-constraint percentile query. With a *multi-constraint percentile query*, we can impose richer properties on strategies, for instance, enforcing that the duration is $<d_1$ in at least 50% of the cases, and $<d_2$ in 95% of the cases, with $d_1 < d_2$. We may also consider percentile queries in multi-dimensional systems. If in the model, we add information about fuel consumption, we may also enforce that we arrive within d time units in 95% of the cases, and that in half of the cases the fuel consumption is below some threshold c.

Contributions We study percentile problems for a range of classical payoff functions: we establish algorithms and prove complexity and memory bounds. Our algorithms can handle multi-constraint multi-dimensional queries, but we also study interesting subclasses,

Table 1 Some results for percentile queries

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P[27]	P(M)·E($\mathcal{Q}$) [18], PSPACE-h	–
$f \in \mathcal{F}$	P[12]	P	P(M)·E($\mathcal{Q}$) PSPACE-h
$\overline{\mathsf{MP}}$	P[27]	P	P
$\underline{\mathsf{MP}}$	P[27]	P(M)·E($\mathcal{Q}$)	P(M)·E($\mathcal{Q}$)
SP	P(M)·P$_{ps}$($\mathcal{Q}$) [22]	P(M)·P$_{ps}$($\mathcal{Q}$) (one target)	P(M)·E($\mathcal{Q}$)
	PSPACE-h [22]	PSPACE-h [22]	PSPACE-h [22]
ε-gap DS	P$_{ps}$($M, \mathcal{Q}, \varepsilon$) NP-h	P$_{ps}$(M, ε)·E($\mathcal{Q}$) NP-h	P$_{ps}$(M, ε)·E($\mathcal{Q}$) PSPACE-h

Here $\mathcal{F} = \{\inf, \sup, \liminf, \limsup\}$, $\overline{\mathsf{MP}}$ (resp. $\underline{\mathsf{MP}}$) stands for sup. (resp. inf.) mean-payoff, SP for shortest path, and DS for discounted sum. Parameters M and $\mathcal{Q}$ resp. represent model size and query size; P(x), E(x) and P$_{ps}$(x) resp. denote polynomial, exponential and pseudo-polynomial time in parameter x. All results without reference are new

namely, multi-constraint single-dimensional queries, single-constraint queries, and other classes depending on the payoff functions. We present an overview of our results in Table 1. For all payoff functions but the discounted sum, they only require *polynomial time in the size of the model* when the query size is fixed. In most applications, the query size is typically small while the model can be very large. So our algorithms have clear potential to be useful in practice.

We give a non-exhaustive list of contributions and highlight some links with related problems.

(A) We show the PSPACE-hardness of the multiple reachability problem with exponential dependency on the query size (Theorem 2), and the PSPACE-completeness of the almost-sure case, refining the results of [18]. We also prove that in the case of *nested* target sets, the problem admits polynomial-time solution (Theorem 3), and we use it to solve some of the multi-constraint percentile problems.

(B) For payoff functions inf, sup, lim inf and lim sup, we establish a polynomial-time algorithm for the single-dimension case (Theorem 5), and an algorithm that is only exponential in the size of the query for the general case (Theorem 6). We prove the PSPACE-hardness of the problem for sup (Theorem 7), and give a polynomial time algorithm for lim sup (Theorem 8).

(C) In the mean-payoff case, we distinguish $\overline{\mathsf{MP}}$ defined by the limsup of the average weights, and $\underline{\mathsf{MP}}$ by their liminf. For the former, we give a polynomial-time algorithm for the general case (Theorem 10). For the latter, our algorithm is polynomial in the model size and exponential in the query size (Theorem 11).

(D) The truncated sum function computes the *sum* of weights until a target is reached. It models *shortest path* problems. We prove the multi-dimensional percentile problem to be undecidable when both negative and positive weights are allowed (Theorem 12). Therefore, we concentrate on the case of non-negative weights, and establish an algorithm that is polynomial in the model size and exponential in the query size (Theorem 13). We derive from [22] that even the single-constraint percentile problem is PSPACE-hard.

(E) The discounted sum case turns out to be difficult, and linked to a long-standing open problem, not known to be decidable (Lemma 17). Nevertheless, we give algorithms for an approximation of the problem, called ε-gap percentile problem. Our algorithm guarantees correct answers up to an arbitrarily small zone of uncertainty (Theorem 14). We also prove that this ε-gap problem is PSPACE-hard in general, and already NP-hard for single-constraint queries (Lemmas 20 and 21). According to a very recent paper by Haase and Kiefer [23], our reduction even proves PP-hardness of single-contraint queries, which suggests that the problem does not belong to NP at all otherwise the polynomial hierarchy would collapse.

We systematically study the memory requirement of strategies. We build our algorithms using different techniques. Here are a few of them. For inf and sup payoff functions, we reduce percentile queries to multiple reachability queries, and rely on the algorithm of [18]: those are the easiest cases. For lim inf, lim sup and $\overline{\mathsf{MP}}$, we additionally need to resort to maximal end-component decomposition of MDPs. For the following cases, there is no simple reduction to existing problems and we need non-trivial techniques to establish algorithms. For $\underline{\mathsf{MP}}$, we use linear programming techniques to characterize winning strategies, borrowing ideas from [6,18]. For shortest path and discounted sum, we consider unfoldings of the MDP, with particular care to bound their sizes, and for the latter, to analyze the cumulative error due to necessary roundings.

Related work There are several works in the literature that study multi-dimensional MDPs: for discounted sum, see [14], and for mean-payoff, see [6,19]. In the latter papers, the following threshold problem is studied in multi-dimensional MDPs: given a threshold vector $\mathbf{v}$ and a probability threshold ν, does there exist a strategy σ such that $\mathbb{P}^{\sigma}_{s}[\mathbf{r} \geq \mathbf{v}] \geq \nu$, where $\mathbf{r}$ denotes the mean-payoff vector. The work [19] solves this problem for the single dimensional case, and the multi-dimensional for the *non-degenerate* case (w.r.t. the solutions of a linear program). A general algorithm was later given in [6]. This problem asks for a bound on the *joint probability* of the thresholds, that is, the probability of satisfying *all* constraints simultaneously. In contrast, in our problem we bound the *marginal probabilities* separately, which may allow for more modeling flexibility. The problem of maximizing the *expectation vector* was also solved in [6]. Recently, and independently from our work, the problem of bounding the marginal probabilities was considered in [13] for $\underline{\mathsf{MP}}$. The given algorithm consists in a single linear program (while we use a two-step procedure), has the same ingredients as ours, and has the same complexity. In addition, the algorithm of [13] also allows one to add a constraint on the expectation vector.

Multiple reachability objectives in MDPs were considered in [18]: given an MDP and multiple targets T_i, thresholds α_i, decide if there exists a strategy that forces each T_i with a probability larger than α_i. This work is the closest to our work and we show here that their problem is inter-reducible with our problem for the sup measure. In [18] the complexity results are given only for the size of the model and not for the size of the query: we refine those results here and answer questions that were left open in that paper.

Several works consider percentile queries but only for *one* dimension and *one* constraint (while we consider multiple constraints and possibly multiple dimensions) and particular payoff functions. Single-constraint queries for lim sup and lim inf were studied in [12]. The threshold probability problem for truncated sum was studied in MDPs with either all non-negative or all non-positive weights in [26,30]. *Quantile queries* in the single-constraint case were studied for the shortest path with non-negative weights in [34], and for energy-utility objectives in [1]. They have been recently extended to *cost problems* [22], in a direction orthogonal to ours. For fixed finite horizon, [38] considers the problem of ensuring a single-contraint percentile query for the discounted sum, and that of maximizing the expected value subject to a single percentile constraint. Still for the discounted case, there is a series of works studying *threshold problems* [36,37] and *value-at-risk problems* [5]. All can be related to single-constraint percentiles queries.

This paper extends previous work presented in a conference [28]: it gives a full presentation of the proofs, along with intermediate results and additional examples.

2 Preliminaries

Markov decision processes A finite *Markov decision process* (MDP) is a tuple $M = (S, A, \delta)$ where S is the finite set of *states*, A is the finite set of *actions* and $\delta : S \times A \to \mathcal{D}(S)$ is a partial function called the *probabilistic transition function*, where $\mathcal{D}(S)$ denotes the set of rational probability distributions over S. The set of actions that are available in a state $s \in S$ is denoted by $A(s)$. We use $\delta(s, a, s')$ as a shorthand for $\delta(s, a)(s')$. An *absorbing state* s is such that for all $a \in A(s)$, $\delta(s, a, s) = 1$. We assume w.l.o.g. that MDPs are *deadlock-free*: for all $s \in S$, we have that $A(s) \neq \emptyset$ (if not the case, we simply replace the deadlock by an absorbing state with a unique action). An MDP where for all $s \in S$, $|A(s)| = 1$ is a fully-stochastic process called a *Markov chain*.

A *weighted* MDP is a tuple $M = (S, A, \delta, w)$, where w is a *d-dimension weight function* $w : A \to \mathbb{Z}^d$. For any $l \in \{1, \ldots, d\}$, we denote $w_l : A \to \mathbb{Z}$ the projection of w to the l-th dimension, i.e., the function mapping each action a to the l-th element of vector $w(a)$. A *run* of M is an infinite sequence $s_1 a_1 \ldots a_{n-1} s_n \ldots$ of states and actions such that $\delta(s_i, a_i, s_{i+1}) > 0$ for all $i \geq 1$. Finite prefixes of runs are called *histories*.

Fix an MDP $M = (S, A, \delta)$. An *end-component* (EC) of M is an MDP $C = (S', A', \delta')$ with $S' \subseteq S$, $\emptyset \neq A'(s) \subseteq A(s)$ for all $s \in S'$, and $\mathsf{Supp}(\delta(s, a)) \subseteq S'$ for all $s \in S'$, $a \in A'(s)$ (here $\mathsf{Supp}(\cdot)$ denotes the support), $\delta' = \delta|_{S' \times A'}$ and such that C is *strongly connected*, i.e., there is a run between any pair of states in S'. The union of two ECs with non-empty intersection is an EC; one can thus define *maximal* ECs. We let $\mathsf{MEC}(M)$ denote the set of maximal ECs of M, computable in polynomial time [16].

Strategies A *strategy* σ is a function $(SA)^*S \to \mathcal{D}(A)$ such that for all $h \in (SA)^*S$ ending in s, we have $\mathsf{Supp}(\sigma(h)) \subseteq A(s)$. The set of all strategies is Σ. A strategy is *pure* if all histories are mapped to *Dirac distributions*. A strategy σ can be encoded by a *Moore machine*, $(\mathcal{M}, \sigma_a, \sigma_u, \alpha)$ where $\mathcal{M}$ is a finite or infinite set of memory elements, α the *initial distribution on* $\mathcal{M}$, σ_u the *memory update function* $\sigma_u : A \times S \times \mathcal{M} \to \mathcal{M}$, and $\sigma_a : S \times \mathcal{M} \to \mathcal{D}(A)$ the *next action function* where $\mathsf{Supp}(\sigma_a(s, m)) \subseteq A(s)$ for any $s \in S$ and $m \in \mathcal{M}$. We say that σ is *finite-memory* if $|\mathcal{M}| < \infty$, and *K-memory* if $|\mathcal{M}| = K$; it is memoryless if $K = 1$, thus only depends on the last state of the history. We see such strategies as functions $s \mapsto \mathcal{D}(A(s))$ for $s \in S$. A strategy is *infinite-memory* if $|\mathcal{M}|$ is infinite. For a class of problems, we say that strategies use linear (resp. polynomial, exponential) memory if there exist strategies for which K is linear (resp. polynomial, exponential) in the size of M. The entity choosing the strategy is often called the *controller*.

An MDP M, a strategy σ encoded by $(\mathcal{M}, \sigma_a, \sigma_u, \alpha)$, and a state s determine a Markov chain M_s^σ defined on the state space $S \times \mathcal{M}$ as follows. The initial distribution is such that for any $m \in \mathcal{M}$, state (s, m) has probability $\alpha(m)$, and 0 for other states. For any pair of states (s, m) and (s', m'), the probability of the transition $(s, m), a, (s', m')$ is equal to $\sigma_a(s, m)(a) \cdot \delta(s, a, s')$ if $m' = \sigma_u(s, m, a)$, and to 0 otherwise. A *run* of M_s^σ is an infinite sequence of the form $(s_1, m_1), a_1, (s_2, m_2), a_2, \ldots$, where each $(s_i, m_i), a_i, (s_{i+1}, m_{i+1})$ is a transition with nonzero probability in M_s^σ, and $s_1 = s$. When considering the probabilities of events in M_s^σ, we will often consider sets of runs of M. Thus, given $E \subseteq (SA)^\omega$, we denote by $\mathbb{P}^\sigma_{M,s}[E]$ the probability of the runs of M_s^σ whose projection[1] to M is in E, i.e., the probability of event E when M is executed with initial state s and strategy σ. Note that every event has a uniquely defined probability [35] (Carathéodory's extension theorem induces a unique probability measure on the Borel σ-algebra over $(SA)^\omega$).

Almost-sure reachability of ECs Let $\mathsf{Inf}(\rho)$ denote the random variable representing the disjoint union of states and actions that occur infinitely often in the run ρ. By an abuse of notation, we see $\mathsf{Inf}(\rho)$ as a sub-MDP M' if it contains exactly the states and actions of M'. It was shown that for any MDP M, state s, strategy σ, $\mathbb{P}^\sigma_{M,s}[\mathsf{Inf} \text{ is an EC}] = 1$ [16].

Multiple reachability Given a subset T of states, let $\Diamond T$ be the *reachability objective w.r.t. T*, defined as the set of runs visiting a state of T at least once. The *multiple reachability* problem consists, given MDP M, state s_{init}, target sets $T_1, \ldots, T_q$, and probabilities $\alpha_1, \ldots, \alpha_q \in [0, 1] \cap \mathbb{Q}$, in deciding whether there exists a strategy $\sigma \in \Sigma$ such that $\bigwedge_{i=1}^q \mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[\Diamond T_i] \geq \alpha_i$. The *almost-sure multiple reachability* problem restricts to $\alpha_1 = \ldots = \alpha_q = 1$.

Percentile problems We consider *payoff functions* among inf, sup, lim inf, lim sup, mean-payoff, truncated sum (shortest path) and discounted sum. For any run $\rho = s_1 a_1 s_2 a_2 \ldots$, dimension $l \in \{1, \ldots, d\}$, and weight function w,

- $\inf_l(\rho) = \inf_{j\geq 1} w_l(a_j)$, $\sup_l(\rho) = \sup_{j\geq 1} w_l(a_j)$,
- $\liminf_l(\rho) = \liminf_{j\to\infty} w_l(a_j)$, $\limsup_l(\rho) = \limsup_{j\to\infty} w_l(a_j)$,
- $\underline{\mathsf{MP}}_l(\rho) = \liminf_{n\to\infty} \frac{1}{n}\sum_{j=1}^n w_l(a_j)$, $\overline{\mathsf{MP}}_l(\rho) = \limsup_{n\to\infty} \frac{1}{n}\sum_{j=1}^n w_l(a_j)$,
- $\mathsf{DS}_l^{\lambda_l}(\rho) = \sum_{j=1}^\infty \lambda_l^j \cdot w_l(a_j)$, with $\lambda_l \in]0, 1[\cap \mathbb{Q}$ a rational discount factor,
- $\mathsf{TS}_l^T(\rho) = \sum_{j=1}^{n-1} w_l(a_j)$ with s_n the first visit of a state in $T \subseteq S$. If T is never reached, then we assign $\mathsf{TS}_l^T(\rho) = \infty$.

For any payoff function f, $f_l \geq v$ defines the runs ρ that satisfy $f_l(\rho) \geq v$. A *percentile constraint* is of the form $\mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[f_l \geq v] \geq \alpha$, where σ is to be synthesized given threshold value v and probability α. We study *multi-constraint percentile queries* requiring to simultaneously satisfy q constraints each referring to a possibly different dimension. Formally, given a d-dimensional weighted MDP M, initial state $s_{\mathsf{init}} \in S$, payoff function f, dimensions $l_1, \ldots, l_q \in \{1, \ldots, d\}$, value thresholds $v_1, \ldots, v_q \in \mathbb{Q}$ and probability thresholds $\alpha_1, \ldots, \alpha_q \in [0, 1] \cap \mathbb{Q}$, the *multi-constraint percentile problem* asks if there exists a strategy $\sigma \in \Sigma$ such that query

$$\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^\sigma_{M,s_{\mathsf{init}}}\left[f_{l_i} \geq v_i\right] \geq \alpha_i$$

holds. We can actually solve queries $\exists?\sigma$, $\bigvee_{i=1}^m \bigwedge_{j=1}^{n_i} \mathbb{P}^\sigma_{M,s_{\mathsf{init}}}\left[f_{l_{i,j}} \geq v_{i,j}\right] \geq \alpha_{i,j}$. We present our results for conjunctions only since the latter is equivalent to verifying the disjuncts independently: i.e., to $\bigvee_{i=1}^m \exists\sigma \bigwedge_{j=1}^{n_i} \mathbb{P}^\sigma_{M,s_{\mathsf{init}}}\left[f_{l_{i,j}} \geq v_{i,j}\right] \geq \alpha_{i,j}$.

We distinguish *single-dimensional percentile problems* ($d = 1$) from *multi-dimensional* ones ($d>1$). We assume w.l.o.g. that $q \geq d$ otherwise one can simply neglect unused

[1] The projection of a run $(s_1, m_1), a_1, (s_2, m_2), a_2, \ldots$ in M_s^σ to M is simply the run $s_1 a_1 s_2 a_2 \ldots$ in M.

dimensions. For some cases, we will consider the *ε-relaxation* of the problem, which consists in ensuring each value $v_i - \varepsilon$ with probability α_i.

Complexity We assume binary encoding of constants, and define the *model size* $|M|$, a polynomial in $|S|$ and the size of the *encoding* of weights and probabilities (e.g., $\log_2 W$ with W the largest absolute weight), as the size of the representation of M; and the *query size* $|\mathcal{Q}|$, a polynomial in the number of constraints q and the encoding of thresholds, that of the query. The *problem size* refers to the sum of the two.

Memory and randomness Throughout the paper, we will study the memory requirements for strategies w.r.t. different classes of percentile queries. Here, we show, by a simple example, that randomness is always needed for all payoff functions.

Lemma 1 *Randomized strategies are necessary for multi-dimensional percentile queries for any payoff function.*

Proof Let M be a 2-dim. deterministic MDP with $S = \{s_0, s_1, s_2\}$, $A = \{a, b\}$ and the transition function defined as $\delta(s_0, a, s_1) = 1$, $\delta(s_0, b, s_2) = 1$, $\delta(s_1, a, s_1) = 1$ and $\delta(s_2, b, s_2) = 1$. Essentially there are only two possible runs in this MDP: $\rho_1 = s_0(a\, s_1)^\omega$ and $\rho_2 = s_0(b\, s_2)^\omega$. Assume that the weight and the payoff functions are chosen such that $f(\rho_1) = (1, 0)$ and $f(\rho_2) = (0, 1)$: they are incomparable. Consider the query

$$\mathcal{Q} := \mathbb{P}^\sigma_{M,s_0}\big[f_1 \geq 1/2\big] \geq 1/2 \quad \wedge \quad \mathbb{P}^\sigma_{M,s_0}\big[f_2 \geq 1/2\big] \geq 1/2.$$

It is easy to see that $\mathcal{Q}$ can only be satisfied by a strategy that chooses between a and b with equal probability, hence no pure strategy satisfies the query. Note that here f can be chosen anything among sup, lim sup, $\underline{\mathsf{MP}}$, $\overline{\mathsf{MP}}$, DS^{λ_l} with appropriate λ_l, and TS^{T_l} with target sets $T_1 = \{s_1\}$ and $T_2 = \{s_2\}$ respectively for each query. For inf, and lim inf, we may switch the weight vectors for the same result. ◻

3 Multiple reachability and contraction of MECs

Multiple reachability The multiple reachability problem was studied [18] where an algorithm based on a linear program (LP) of size polynomial in the model and exponential in the query was given. As a particular case, it was proved that restricting the target sets to absorbing states yields a polynomial-size LP. We will use this LP later in Fig. 5 in Sect. 5.

Theorem 1 ([18]) *Memoryless strategies suffice for multiple reachability with absorbing target states, and can be decided and computed in polynomial time. With arbitrary targets, exponential-memory strategies (in query size) can be computed in time polynomial in the model and exponential in the query.*

In this section, we improve over this result by showing that the case of almost-sure multiple reachability is PSPACE-complete, with a recursive algorithm and a reduction from QBF satisfiability. This also shows the PSPACE-hardness of the general problem. Moreover, we show that exponential memory is required for strategies, following a construction of [15].

Theorem 2 *The almost-sure multiple reachability problem is* PSPACE*-complete, and strategies need exponential memory in the query size.*

We first show the PSPACE-completeness of the almost-sure multiple reachability problem.

Lemma 2 *The almost-sure multiple reachability problem is* PSPACE*-complete.*

Proof We start by showing PSPACE-membership. Let M be an MDP, s_0 a state, and $T_1, \ldots, T_q$ target sets. We write $T = T_1 \cup \ldots \cup T_q$. Note first that we know how to solve the problem in polynomial time for $q = 1$. Let M' be the MDP obtained by M by making all states in T absorbing. The procedure works as follows. For each state $x \in T$, let us define $I = \{1 \leq i \leq q \mid x \notin T_i\}$; we clearly have $|I|<n$. We recursively verify whether there is a strategy almost-surely satisfying the multiple reachability objective $(T_i)_{i \in I}$. Let $\mathcal{T}$ denote all states of T for which the recursive call returned positively. We now check in polynomial time whether the set $\mathcal{T}$ can be reached almost-surely from s_0. Note that the recursive call depth is linear, so the whole procedure uses polynomial space.

We now prove the equivalence between M and M'. Assume that there is a strategy σ almost-surely reaching $\mathcal{T}$ in M'. This strategy can be followed in M until some state x of $\mathcal{T}$ is reached, which happens almost-surely. But we know, by the recursive callof our procedure, that from any such state $x \in \mathcal{T}$ there exists a strategy almost-surely satisfying the rest of the reachability objectives. Thus, by extending σ in each state $x \in \mathcal{T}$ by these strategies, we construct a solution to the multiple reachability problem in M. Notice that the constructed strategy uses linear memory since it is "memoryless" between each switch.

Conversely, assume that there is a strategy σ satisfying the multiple reachability query in M from s_0. Towards a contradiction, assume that some state $x \in T \setminus \mathcal{T}$ is reached with positive probability in M under σ, thus also in M' under the same strategy. We know by the recursive call of our procedure that the remaining targets cannot be satisfied almost-surely by any strategy from state x in M. It follows that strategy σ fails to satisfy all targets almost-surely from s_0, a contradiction.

To show PSPACE-hardness, we reduce the truth value of a quantified Boolean formula (QBF) to our problem. An instance of QBF is a quantified Boolean formula over $X = \{x_1, x_2, \ldots, x_n\}$

$$\Psi \equiv \exists x_1 \forall x_2 \exists x_2 \ldots \forall x_{n-1} \exists x_n \cdot C_1 \wedge C_2 \wedge \ldots C_m$$

where each clause C_i is the disjunction of 3 literals taken in $\{x, \neg x \mid x \in X\}$. From Ψ, we construct an (acyclic) MDP as shown in Fig. 1. For each variable x_i, there are three states called x_i, f_i and t_i in the MDP. In a state x_i that corresponds to an *existentially quantified* variable, there are two actions that are available: $\top$ and $\bot$. The action $\top$ visits (deterministically) the state t_i while the action $\bot$ visits the state f_i, and then in the two cases, the run proceeds to the state for the next variable. Intuitively, choosing $\top$ in x_i corresponds to the choice of truth value true for x_i, and $\bot$ to truth value false. In a state x_i that corresponds to an *universally quantified* variable, there is only the action $*$ available and the successor is chosen uniformly at random between f_i and t_i. The targets are defined as follows: for each clause C_j, the target set $T_j = \{t_i \mid x_i \in C_j\} \cup \{f_i \mid \neg x_i \in C_j\}$ must be visited with probability one. Clearly, given the value assigned to a variable x_i, we visit exactly the set of target sets T_j that correspond to

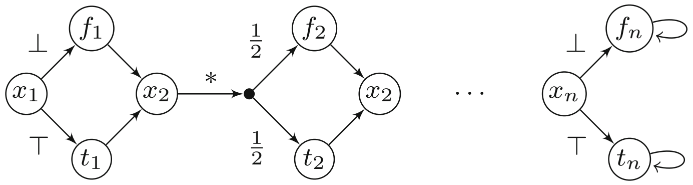


Fig. 1 Reduction for the QBF formula $\exists x_1 \forall x_2 \ldots \exists x_n C_1 \wedge \ldots \wedge C_m$. The objectives are $T_j = \{t_i \mid x_i \in C_j\} \cup \{f_i \mid \neg x_i \in C_i\}$ for all $1 \leq j \leq m$

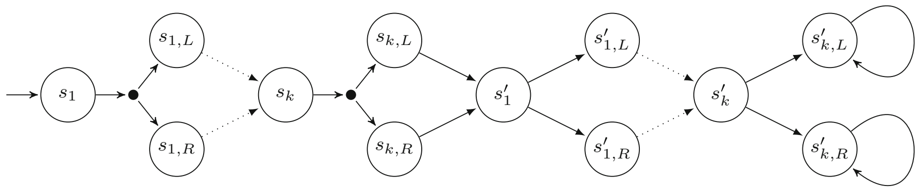


Fig. 2 Family of multiple reachability problems requiring exponential memory

the clauses that are made true by the valuation of x_i. It should be clear now that the histories in the MDP are in bijection with the valuation of the Booelan variables in Ψ and that the set of valuations that satisfies Ψ correspond exactly to the histories that visits all the sets T_j, $1 \leq j \leq n$ with probability one.

Now, we claim that there is a strategy to reach each set T_j, $1 \leq j \leq n$, with probability one if and only if the formula Ψ is true. Indeed, if Ψ is true, we know that there exists for each existentially quantified variable x_i a choice function g_{x_i} which assign a truth value to x_i given the truth values chosen for the variables that appears before x_i in the quantification block. These choice functions naturally translate into a (deterministic memryfull) strategy that mimics the choices of truth values by choosing between $\perp$ and $\top$ accordingly. We get that if the formula is true (all closed are made true) then the associated strategy visits all the target sets with probability one.

For the other direction, we first note that it is not useful for the scheduler to play a randomised strategy. As the MDP is acyclic (except for the two states t_n and f_n that have a self loop), all the target sets are visited with probability one if and only if all the outcomes of the strategy visits all the target sets. So, if the scheduler plays randomly say in state x_i then all the resulting outcomes for action $\perp$ and all the resulting outcomes for action $\top$ must visit all the target sets, so both choices need to be good and there is no need for randomisation and the scheduler can safely choose one of the two arbitrarily. So, pure strategies are sufficient and but we have seen that pure strategies corresponds exactly to the choice functions in the QBF problem. So is clear that from a winning strategy for the scheduler, we can construct a choice function that makes the formula true. □

We establish an exponential lower bound on the memory requirements based on a family of MDPs depicted in Fig. 2 and inspired from [15, Lemma 8].

Lemma 3 *Exponential-memory in the query size is necessary for almost-sure multiple reachability.*

Proof Consider the unweighted MDP M depicted in Fig. 2. The MDP is composed of k gadgets where a state between $s_{i,L}$ and $s_{i,R}$ is stochastically chosen (they are equiprobable), followed by k gadgets where the controller can decide to visit either $s'_{i,L}$ or $s'_{i,R}$. We define an almost-sure multiple reachability problem for target sets

$$T_i = \{s_{1,L}, s'_{1,L}\}, \{s_{1,R}, s'_{1,R}\}, \{s_{2,L}, s'_{2,L}\}, \dots, \{s_{k,L}, s'_{k,L}\}, \{s_{k,R}, s'_{k,R}\}.$$

Hence, this problem requires $q = 2 \cdot k$ constraints to be defined. We claim that a strategy satisfying this problem cannot be expressed by a Moore machine containing less than $2^k = 2^{\frac{q}{2}}$ memory states.

Indeed, it is clear that to ensure almost-sure reachability of all sets T_i, the controller has to chose in state s'_i the exact opposite action of the one stochastically chosen in s_i. Remembering the k choices made in states s_i requires k bits of encoding. Hence, a satisfying strategy requires a Moore machine with 2^k memory states to encode those choices.

It is easy to see that if the controller uses a—possibly randomized—strategy σ with $<2^k$ memory states, then there exists $i \in \{1, \ldots, k\}$ such that $\sigma(s_1 \ldots s_i s_{i,L} \ldots s'_i) = \sigma(s_1 \ldots s_i s_{i,R} \ldots s'_i)$, i.e., the controller chooses to go to $s'_{i,L}$ (resp. $s'_{i,R}$) with identical probability against both stochastic choices in s_i. Assume that the controller chooses to go toward $s'_{i,L}$ with probability $p \in [0, 1]$ and toward $s'_{i,R}$ with probability $1 - p$: this implies that the probability that the target set $\{s_{i,L}, s'_{i,L}\}$ (resp. $\{s_{i,R}, s'_{i,R}\}$) is never visited is equal to $\frac{1}{2} \cdot (1 - p)$ (resp. $\frac{1}{2} \cdot p$). Clearly, it is impossible to have both those probabilities equal to zero simultaneously, which proves that such a strategy cannot satisfy the almost-sure multiple reachability problem defined above, and concludes our proof. □

Despite the above lower bounds, it turns out that the polynomial time algorithm for the case of absorbing targets can be extended: we identify a subclass of the multiple reachability problem that admits a polynomial-time solution. In the *nested multiple reachability* problem, the target sets are nested, i.e., $T_1 \subseteq T_2 \subseteq \ldots \subseteq T_q$. The memory requirement for strategies is reduced as well to linear memory.

Theorem 3 *The nested multiple reachability problem can be solved in polynomial time. Strategies have memory linear in the query size, which is optimal.*

Intuitively, we use $q + 1$ copies of the original MDP, one for each target set, plus one last copy. The idea is then to travel between those copies in a way that reflects the nesting of target sets whenever a target state is visited. The crux to obtain a polynomial-time algorithm is then to reduce the problem to a multiple reachability problem *with absorbing states* over the MDP composed of the $q + 1$ copies, and to benefit from the reduced complexity of this case.

Proof Assume an MDP M, s_0 and the target sets $T_1 \subseteq \ldots \subseteq T_q$ are given. We make $q + 1$ copies of the MDP M, namely, $M_1, \ldots, M_q, M_{q+1}$. We start M_{q+1} at state s_{init}. We redirect some of the edges as follows. For any M_i, state s, action a, and $t \in \mathsf{Supp}(\delta(s, a))$, if $t \in T_j$ for some $j<i$, then we direct this edge to state t in $M_{j'}$ where j' is the smallest index with $t \in T_{j'}$. Hence, along any run, we are in copy M_j if, and only if we have already satisfied all targets $T_j, \ldots, T_q$. Now, we add a fresh absorbing state $\perp_i$ to each copy. From all states of M_i a fresh action $a^\perp$ leads to $\perp_i$. Let us call this new MDP M'. Note that the size of M is $\mathcal{O}(q|M|)$.

For each $i = 1 \ldots q$, we define $T'_i = \{\perp_i, \perp_{i-1}, \ldots, \perp_1\}$. We claim that the multiple reachability problem query $(T_i, \alpha_i)_{1 \leq i \leq q}$ for M is equivalent to $(T'_i, \alpha_i)_{1 \leq i \leq q}$ for M'. But the latter query has absorbing target states, thus the problem can be solved in polynomial time by [18].

Consider a strategy σ for M achieving the objectives $(T_i, \alpha_i)_{1 \leq i \leq q}$. We can assume w.l.o.g. that σ is finite-memory by [18]. Let S_i denote the set of states of M_i. We define strategy σ' for M' as follows. Let us define a mapping $p(\cdot)$ from the histories of M' to those of M, where a state of any copy is projected to the original state in M. The mapping is actually a bijection from histories of M' that do not use the action $a^\perp$ to histories of M. Now, for all histories h of M' that end in copy M_i, if $\mathbb{P}^{\sigma}_{M,p(h)}[\Diamond \cup_{j<i} T_i] = 0$, we set $\sigma'(h) = a^\perp$. Otherwise, we let $\sigma'(h) = \sigma(p(h))$.

We prove that for all $i = 1 \ldots q$, $\mathbb{P}^{\sigma}_{M,s_0}[\Diamond T_i] \leq \mathbb{P}^{\sigma'}_{M',s'_0}[\Diamond T'_i]$. Let $\iota(h)$ denote the copy in which h ends in M'. For all histories h of M from which the probability of satisfying

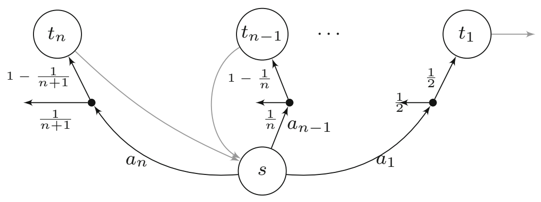


Fig. 3 Linear memory is required for the nested multiple reachability problem

$\Diamond \cup_{j<\iota(p^{-1}(h))} T_i$ is nonzero, we have $\mathbb{P}^{\sigma}_{M,s_0}[h] = \mathbb{P}^{\sigma'}_{M',s'_0}[p^{-1}(h)]$ by definition. Define $H_i = \{h \mid \forall i = 1 \ldots |h|-1, h_i \notin T_i, h_{|h|} \in T_i, \mathbb{P}^{\sigma}_{M,s_0}[h]>0\}$, that is, the histories that visit T_i for the first time at their last state. Clearly $\mathbb{P}^{\sigma}_{M,s_0}[H_i] = \mathbb{P}^{\sigma}_{M,s_0}[\Diamond T_i]$. But the probability of reaching T_i is always nonzero along these histories, so we also have $H_i = H'_i := \{h \in H_i \mid \forall i = 1 \ldots |h|, \mathbb{P}^{\sigma}_{M,h_{1\ldots i}}[\Diamond T_i]>0\}$, and we get $\mathbb{P}^{\sigma'}_{M',s'_0}[H'_i] = \mathbb{P}^{\sigma}_{M,s_0}[H'_i]$. In other words, $\mathbb{P}^{\sigma'}_{M',s'_0}[\Diamond \cup_{j\le i} S_j] \ge \mathbb{P}^{\sigma}_{M,s_0}[\Diamond T_i]$, that is, the target sets $T_1, \ldots, T_q$ are reached in M' with at least the same probabilities as in M. We now need to show that from any history ending in copy M_i, some state $\bot_j$ with $j \le i$ is reached almost-surely in M' under σ'. It will follow that $\mathbb{P}^{\sigma'}_{M',s'_{\mathsf{init}}}[\Diamond T'_i] \ge \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\Diamond T_i]$. To see this, notice that strategy σ is finite-memory, and so is σ'. So there exists $\nu>0$ such that for any state s, and memory element m if the probability of satisfying $\Diamond \cup_{j<i} S_j$ is nonzero from s and m, then it is at least ν. Note that the probability of never satisfying $\Diamond \cup_{j<i} S_j$ while staying in such states is 0. So, whenever the run reaches copy M_i, almost-surely, either some copy M_j with $j<i$ is reached, or we reach a history h such that $\mathbb{P}^{\sigma}_{M,p(h)}[\Diamond \cup_{j<i} T_j] = 0$, in which case we end in $\bot_i$. The inequality follows.

Conversely, consider any strategy σ' for M' achieving the reachability objectives $(T'_i, \alpha_i)_{1\le i\le q}$. We assume σ' to be memoryless by [18]. We define $\sigma(h) = \sigma'(p^{-1}(h))$ whenever the action σ' prescribes is different than $a^{\bot}$, and otherwise σ' switches to an arbitrary memoryless strategy. Since all histories of M' that end in $\bot_i$ satisfy the objectives $T_i \cup \ldots \cup T_q$, strategy σ achieves the objectives $(T_i, \alpha_i)_{1\le i\le q}$. The memory of σ is $\mathcal{O}(q)$ since σ' is memoryless in M' which is made of q copies of M.

We now show that linear memory is necessary. Consider an MDP M with states $s, t_1, \ldots, t_n$, $\bot$. State s has n actions $a_1, \ldots, a_n$. For each $1 \le i \le n$, action a_i leads from s to t_i with probability $1 - \frac{1}{i+1}$, and with probability $\frac{1}{i+1}$ leads to absorbing state $\bot$. From all states t_i with $i>1$, s is reachable by a deterministic action, but from t_1 one can only reach $\bot$. The MDP is depicted in Fig. 3 ($\bot$ is not shown).

We consider the nested multiple reachability targets $T_1, \ldots, T_n$ with $T_i = \{t_i, \ldots, t_1\}$ for each i, and consider threshold probabilities $\alpha_1, \ldots, \alpha_n$ defined by $\alpha_n = 1 - \frac{1}{n+1}$, and $\alpha_i = \alpha_{i+1}(1 - \frac{1}{i+1})$ for $1 \le i \le n-1$.

Let us first describe a strategy that satisfies these constraints. Define a strategy that deterministically chooses, at each visit to state s, the following actions: $a_n, a_{n-1}, \ldots, a_1$. A simple calculation shows that the constraints are satisfied: the probability of satisfying T_n is at least $1 - \frac{1}{n+1}$ by the first action, that of T_{n-1} is $\frac{1}{n+1}(1 - \frac{1}{n})$ by the sequence $a_n a_{n-1}$ of actions, and so on. We argue that this is the only strategy that satisfies these reachability queries showing that $\mathcal{O}(n)$ memory is necessary.

Consider any strategy σ satisfying the multiple reachability queries. We show that σ must deterministically choose a_n in the first step. In fact, assume that some action a_i with $i \ne n$ is chosen with probability $\eta>0$. The probability of moving to $\bot$ under any such action is at least $\frac{1}{n}$. Thus the probability of going to $\bot$ in the first step (without seeing any t_i) is at least $\eta\frac{1}{n} + (1-\eta)\frac{1}{n+1} > \frac{1}{n+1}$, which is a contradiction. Now, assume that σ deterministically chooses

$a_n a_{n-1} \dots a_{n-i+1}$ in the first i steps. The probability of reaching T_{n-i} in the first i steps is thus 0, while the probability of being in s (and not in $\bot$) after i steps is $\gamma = (1 - \frac{1}{n+1}) \cdots (1 - \frac{1}{n-i+2})$. Assume that σ does not deterministically choose a_{n+i}. Target T_{n-i} is reached by histories that eventually choose some action $a_{n-i}, \dots, a_1$. Let H denote the set of histories stopping at the first action from this set, i.e., $H = s((a_n + \dots + a_{n-i+1})s)^*(a_{n-i} + \dots + a_1)$. Note that at these histories, either we satisfy T_{n-i} or we end in $\bot$, so the probability of satisfying T_{n-i} under σ can be written as $\gamma \sum_{h \in H} \alpha_h p_h$, where α_h is the probability of σ of choosing the actions of h from the current history, and p_h is the probability of the resulting run. We have, for all $h \in H$, $p_h \leq (1 - \frac{1}{n+1})^{\frac{|h|-1}{2}} (1 - \frac{1}{n-i+1})$ since h contains $\frac{|h|-1}{2}$ actions outside $a_1, \dots, a_{n-i}$ and after each such action we must come back to s. For all $h \in H$ with $\frac{|h|-1}{2} > 1$, we must have $\alpha_h = 0$ since otherwise we would get $\gamma \sum_{h \in H} \alpha_h p_h < \gamma(1 - \frac{1}{n-i+1})$. Furthermore, if σ chooses an action some action a_j with $1 \leq j \leq n - i$ in the first step, the probability of going to $\bot$ is $\frac{1}{j+1} > 1 - \frac{1}{n-i+1}$. It follows that $\alpha_h = 1$ for the unique history that chooses action a_{n-i}. □

Contraction of MECs In order to solve percentile queries, we sometimes reduce our problems to multiple reachability by first contracting MECs of given MDPs, which is a known technique [16]. We define a transformation of MDP M to represent the events $\mathsf{Inf}(\rho) \subseteq C$ for $C \in \mathsf{MEC}(M)$ as fresh states. Intuitively, all states of a MEC will now lead to an absorbing state that will abstract the behavior of the MEC.

Consider M with $\mathsf{MEC}(M) = \{C_1, \dots, C_m\}$. We define MDP M' from M as follows. For each C_i, we add state s_{C_i} and action a^* from each state $s \in C_i$ to s_{C_i}. All states s_{C_i} are absorbing, and $A(s_{C_i}) = \{a^*\}$. The probabilities of events $\mathsf{Inf}(\rho) \subseteq C_i$ in M are captured by the reachability of states s_{C_i} in M', as follows. We use the classical temporal logic symbols $\Diamond$ and $\Box$ to represent the *eventually* and *always* operators respectively.

Lemma 4 *Let M be an MDP and $\mathsf{MEC}(M) = \{C_1, \dots, C_m\}$. For any strategy σ for M, there exists a strategy τ for M' such that for all $i \in \{1, \dots, m\}$, $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\Diamond\Box C_i] = \mathbb{P}^{\tau}_{M',s_{\mathsf{init}}}[\Diamond s_{C_i}]$. Conversely, for any strategy τ for M' such that $\sum_{i=1}^{m} \mathbb{P}^{\tau}_{M',s_{\mathsf{init}}}[\Diamond s_{C_i}] = 1$, there exists σ such that for all i, $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\Diamond\Box C_i] = \mathbb{P}^{\tau}_{M',s}[\Diamond s_{C_i}]$.*

Proof Consider any strategy τ in M' with $\sum_{i=1}^{m} \mathbb{P}^{\tau}_{M',s}[\Diamond s_{C_i}] = 1$. We define strategy σ for M by imitating τ except that whenever it chooses action a^* from some state $s \in C_i$, we switch to a memoryless strategy that surely stays inside C_i. The desired equality follows. The other direction was proved in [6, Lemma 4.6]. □

Under some hypotheses, solving multi-constraint percentile problems on ECs yield the result for all MDPs, by the transformation of Lemma 4. We prove a general theorem and then derive particular results as corollaries.

Theorem 4 *Consider all prefix-independent payoffs f such that for all strongly connected MDPs M, and all $(l_i, v_i)_{1 \leq i \leq q} \in \{1, \dots, d\} \times \mathbb{Q}$, there exists a strategy σ such that*

$$\forall i \in \{1, \dots, d\}, \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[f_{l_i} \geq v_i] \geq \sup_{\tau} \mathbb{P}^{\tau}_{M,s_{\mathsf{init}}}[f_{l_i} \geq v_i].$$

If the value $\sup_\tau$ is computable in polynomial time for strongly connected MDPs, then the multi-constraint percentile problem for f is decidable in polynomial time. Moreover, if strategies achieving $\sup_\tau$ for strongly connected MDPs use $\mathcal{O}(g(M, q))$ memory, then the overall strategy use $\mathcal{O}(g(M, q))$ memory.

The hypotheses are crucial. Essentially, we require payoff functions that are prefix-independent and for which strategies can be combined easily inside MECs (in the sense that if two constraints can be satisfied independently, they can be satisfied simultaneously). Prefix-independence also implies that we can forget about what happens before a MEC is reached. Hence, by using the MEC contraction, we can reduce the percentile problem to multiple reachability for absorbing target states.

Proof Consider an MDP M, an initial state s_{init}, and an instance of the multi-constraint percentile problem $(l_i, v_i, \alpha_i)_{1\leq i\leq q}$ for payoff function f.

Let $C_1, \ldots, C_m$ denote the MECs of M. Consider the MDP M' of Lemma 4. For each $1 \leq j \leq m$, let $\mathbf{u(j)}$ denote the component-wise optimal value vector achievable inside C_j and σ_j a witness strategy, which can be computed by hypothesis in polynomial time. Note that because f is prefix-independent and each C_j strongly connected, it follows by [8] that $\sup_\tau \mathbb{P}^\tau_{M,s_{\mathsf{init}}}[f_{l_i} \geq v_i] \in \{0, 1\}$. In fact, for strongly connected MDPs, if a prefix-independent measure can be satisfied with nonzero probability, then there exists a state from which the threshold can be satisfied with probability 1. Moreover, because the MDP is strongly connected, such a state is reachable almost-surely from any other state.

Now, for each $1 \leq i \leq q$, we define $T_i = \{s_{C_j} \mid 1 \leq j \leq m, \sup_\tau \mathbb{P}^\tau_{M,s_{\mathsf{init}}}[f_{l_i} \geq u(j)_i] = 1\}$, where states s_{C_i} were defined in Lemma 4. We solve the multiple reachability with absorbing targets $T_1, \ldots, T_m$ in M', with probabilities $\alpha_1, \ldots, \alpha_q$, by Theorem 1. All computations are in polynomial time. We now establish the connection with the multi-constraint percentile problem.

Assume there is a strategy τ in M' witnessing the multiple reachability problem. Recall that the strategy σ for M of Lemma 4 derived from τ consists in following τ until an action a^* is taken, upon which one switches to an arbitrary strategy inside the current MEC. Let us define strategy σ' in this manner, by switching to the optimal strategy σ_j, where C_j is the current MEC. It follows that, for each $1 \leq i \leq q$, the probability of switching to σ_j for j such that $s_{C_j} \in T_i$ is at least α_i. But such σ_j satisfy $f_{l_i} \geq v_i$ almost-surely in C_j. Because f is prefix-independent, we get $\mathbb{P}^{\sigma'}_{M,s_{\mathsf{init}}}[f_{l_i} \geq v_i] \geq \alpha_i$. Strategy σ' thus just needs one additional bit compared to σ to remember whether it has switched to a strategy inside a MEC.

Conversely, consider any strategy σ satisfying the multi-constraint percentile problem for f. Let τ be the strategy for M' given by Lemma 4. We have,

$$\begin{aligned}\mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[f_{l_i} \geq v_i] &= \textstyle\sum_{j=1}^m \mathbb{P}^\sigma_{M,s}[f_{l_i} \geq v_i \mid \Diamond\Box C_j]\mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[\Diamond\Box C_j]\\ &= \textstyle\sum_{j=1}^m \mathbb{P}^\sigma_{M,s}[f_{l_i} \geq v_i \mid \Diamond\Box C_j]\mathbb{P}^\tau_{M',s_{\mathsf{init}}}[\Diamond s_{C_j}]\end{aligned}$$

Furthermore, we $\mathbb{P}^\sigma_{M,s}[f(w_i) \geq v_i \mid \Diamond\Box C_j]>0$ implies that $\sup_{\sigma'} \mathbb{P}^{\sigma'}_{C_j,s}[f(w_i) \geq v_i] = 1$ as observed above. It follows that

$$\mathbb{P}^\sigma_{M,s}[f(w_i) \geq v_i \mid \Diamond\Box C_j] \leq \sup_{\sigma'} \mathbb{P}^{\sigma'}_{C_j,s}[f(w_i) \geq v_i)].$$

We obtain

$$\alpha_i \leq \mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[f(w_i) \geq v_i] \leq \sum_{j:C_j\in T_i} \mathbb{P}^\tau_{M',s_{\mathsf{init}}}[\Diamond s_{C_j}],$$

which concludes the proof. □

4 Inf, sup, liminf, limsup payoff functions

Single-dimensional queries We give polynomial-time algorithms for the *single*-dimensional multi-constraint percentile problems. For inf and sup we reduce the problem to nested multiple reachability, while lim inf and lim sup are solved by applying Theorem 4.

Theorem 5 *The single-dimensional multi-constraint percentile problems can be solved in polynomial time in the problem size for* inf*,* sup*,* lim inf*, and* lim sup *functions. Computed strategies use memory linear in the query size for* inf *and* sup*, and constant memory for* lim inf *and* lim sup.

Proof Let us fix MDP M, and a starting state s_{init}. We start with sup. The result will be derived by Theorem 3. Consider an instance $(v_i, \alpha_i)_{1 \le i \le q}$ of the problem, where we assume w.l.o.g. that $v_1 \le \ldots \le v_q$. To simplify the argument, let us assume that weights are assigned to states rather than edges; one can always transform the given MDP (in polynomial time) to ensure this. We define T_i as the set of states whose weights are at least v_i. The problem of ensuring that $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\sup \ge v_i] \ge \alpha_i$ by some strategy σ is then equivalent to the nested reachability problem with targets $T_1 \supseteq T_2 \supseteq \ldots \supseteq T_q$. The problem can thus be solved in polynomial time by Theorem 3. The resulting strategies use linear memory by this theorem.

For Inf, consider an instance $(v_i, \alpha_i)_{1 \le i \le q}$ of the problem with $v_1 \le \ldots \le v_q$. We make $q + 1$ copies of M, each named M_i. For any state s of M, we refer as $s(i)$ to the corresponding copy in M_i. The starting state is $s_{\mathsf{init}}(q + 1)$. In each M_i, any edge from $s(i)$ to $t(i)$ of weight $w<v_i$ is redirected to $t(j)$, where $j \le i$ is the least index such that $w<v_j$. Intuitively, if the run is in M_i, this means that the current history h violates all constraints $\inf \ge v_j$ for all $j = i \ldots q$. For each $1 \le i \le q$, let Safe^i_M denote the set of states of M_{i+1} from which $\inf \ge v_i$ can be surely satisfied. These sets can be computed in polynomial time. Now, we add an absorbing state $\top_i$ for each copy M_i, and a fresh action $a^{\top}$ deterministically leads to $\top_i$ from all states Safe^i_M. Note that M' has size $\mathcal{O}(q|M|)$. Define $T_i = \{\top_i, \ldots, \top_q\}$. Now the multiple reachability instance $(T_i, \alpha_i)_{1 \le i \le q}$ on M' (with absorbing target states) is equivalent to the multi-constraint percentile problem for inf. In fact, from any strategy σ satisfying the reachability probabilities in M', one can clearly construct a strategy for M by following σ until some state $\top_i$ is reached, and then switching to a strategy that is surely safe for the objective $\inf \ge v_i$. Conversely, given a strategy σ for M satisfying the multi-constraint percentile query, we define strategy σ' for M' by following σ, and as soon as some state Safe^i_M is reached, going to $\top_i$. We argue that for each $1 \le i \le q$, the probability of reaching $\cup_{j=i}^{q} \mathsf{Safe}^j_M$ is at least α_i in M under σ. In fact, otherwise, with probability more than $1 - \alpha_i$, the play always stays outside this set. Because $\inf \ge v_i$ is a safety property, this means that the property is violated with probability more than $1 - \alpha_i$, which is a contradiction. The resulting strategy uses linear memory since M' is made of $q + 1$ copies of M.

For liminf and limsup, consider an instance $(v_i, \alpha_i)_{1 \le i \le q}$ of the problem, where we assume $v_1 \le \ldots \le v_q$. We are going to use Theorem 4.

The problem is easy to solve for an end-component C: for each $i = 1 \ldots q$, one removes all edges with weight smaller than v_i, and checks if there is an end-component C' included in C. Consider the largest i with this property. We know that from any state of C, C' can be reached almost-surely, and one can stay inside C' surely. Then by such a strategy, all constraints $\liminf \ge v_j$ for $j = 1 \ldots i$ are satisfied almost-surely, while other constraints are violated almost-surely by any strategy that stays inside C. Optimal strategies inside strongly connected MDPs are thus memoryless. We satisfy the hypotheses of Theorem 4, which yields a polynomial-time algorithm.

The limsup case is solved similarly: In each end-component C, if i_0 denotes the largest v_{i_0} such that some edge of C has weight $\geq v_{i_0}$, then all constraints $\limsup \geq v_j$ for $j = 1 \dots i_0$ can be satisfied almost-surely, and no other constraint is satisfied by any strategy.

The memory usage follows from Theorem 4. □

Multi-dimensional queries We show that all multi-dimensional cases can be solved in time polynomial in the model size and exponential in the query size by a reduction to multiple LTL objectives studied in [18]. Our algorithm actually solves a more general class of queries, where the payoff function can be different for each query.

Theorem 6 *The multi-dimensional percentile problems for payoffs* sup, inf, lim sup *and* lim inf *can be solved in time polynomial in the model size and exponential in the query size, yielding strategies with memory exponential in the query.*

Given an MDP M, for all $i \in \{1 \dots q\}$ and value v_i, we denote $A_{l_i}^{\geq v_i}$ the set of actions of M whose rewards are at least v_i. We fix an MDP M. For any constraint $\phi_i \equiv f(w_{l_i}) \geq v_i$, we define an LTL formula denoted Φ_i as follows. For $f_{l_i} = \inf$, $\Phi_i = \Box A_{l_i}^{\geq v_i}$, for $f_{l_i} = \sup$, $\Phi_i = \Diamond A_{l_i}^{\geq v_i}$, for $f_{l_i} = \liminf$, $\Phi_i = \Diamond\Box A_{l_i}^{\geq v_i}$, and for $f_{l_i} = \limsup$, $\Phi_i = \Box\Diamond A_{l_i}^{\geq v_i}$. The percentile problem is then reduced to queries of the form $\wedge_{i=1}^{q} \mathbb{P}_{M,s_{\mathsf{init}}}^{\sigma}[\Phi_i] \geq \alpha_i$, for which an algorithm was given in [18] that takes time polynomial in $|M|$ and doubly exponential in q. We improve this complexity since our formulae have bounded sizes.

Lemma 5 *For all constraints $\phi_1, \dots, \phi_q$, and probabilities $\alpha_1, \dots, \alpha_q$, there exists a strategy σ such that $\bigwedge_{1 \leq i \leq q} \mathbb{P}_{M,s_{\mathsf{init}}}^{\sigma}[\phi_i] \geq \alpha_i$ if, and only if, there exists a strategy τ such that $\bigwedge_{1 \leq i \leq q} \mathbb{P}_{M,s_{\mathsf{init}}}^{\tau}[\Phi_i] \geq \alpha_i$. This can be decided in time polynomial in the model and exponential in the query, and computed strategies use exponential memory in the query.*

Proof The correspondence between LTL formulae and weighted objectives are clear by construction. The complexity follows from [18]. In fact, if D_i denotes the subset construction applied to Büchi automata recognizing Φ_i, then the multiple objective LTL problem can be solved in time polynomial in the size of the product of M with $D_1, \dots, D_q$. But for each formula, a Büchi automaton of size 2 can be constructed; it follows that the algorithm of [18] has complexity polynomial in $|M|$ and exponential in q. The computed strategy is memoryless on the product of M and $D_1, \dots, D_q$, thus the corresponding strategy for M has memory $D_1 \times \dots \times D_q$ which is a single exponential in q. □

The exponential dependency on the query size cannot be avoided in general unless $\mathsf{P} = \mathsf{PSPACE}$, as shown in the following theorem.

Theorem 7 *The multi-dimensional percentile problem is* PSPACE*-hard for* sup.

Proof Multiple reachability with arbitrary target sets can be encoded as the multi-dimensional multi-constraint percentile problem for sup with weights from $\{0, 1\}$, as we show now. Given MDP M and targets $T_1, \dots, T_q$, we define M' by duplicating states as follows. For each state s, we create a new state s^{bis}. All actions leaving s now leave from s^{bis}, and a single action a^s deterministically leads from s to s^{bis}. It is clear that there is a bijection between the strategies of M and those of M' and that they induce the same reachability probabilities for any subset of states of M. We define a q-dimensional weight function on M' that takes values in $\{0, 1\}$. At any state s, $w_i(a^s) = 1$ if, and only if $s \in T_i$. All other actions have value 0. In other terms, the weit function assigns 1 to dimension i if the target set T_i

is seen. Since the payoff function is sup, along any history the dimensions that have the value 1 are exactly the target sets that have been satisfied. For any probabilities $\alpha_1, \ldots, \alpha_q$, $\exists\sigma, \forall i = 1 \ldots q, \mathbb{P}^{\sigma}_{M', s_{\mathsf{init}}}[\sup_i \geq 1] \geq \alpha_i$, if, and only if $\exists\sigma, \mathbb{P}^{\sigma}_{M,s}[\Diamond T_i] \geq \alpha_i]$. PSPACE-hardness follows from Theorem 2. □

Nevertheless, the complexity can be improved for lim sup functions, for which we give a polynomial-time algorithm by an application of Theorem 4.

Theorem 8 *The multi-dimensional percentile problem for* lim sup *is solvable in polynomial time. Computed strategies use constant-memory.*

Proof The problem is easy to solve if M is strongly connected. In fact, if for some i, M contains no action whose weight at dimension l_i is at least v_i, then no strategy satisfies $\limsup_{l_i} \geq v_i$ with positive probability. Conversely, let $I \subseteq \{1, \ldots, q\}$ such that for each $i \in I$, M contains an edge e with $w_{l_i}(e) \geq v_i$. Then, there is a strategy σ satisfying $\wedge_{i \in I} \mathbb{P}^{\sigma}_{M,s_0}[\limsup_{l_i} \geq v_i] = 1$. In fact, because M is strongly connected each state and action can be eventually reached almost-surely from any state. In particular, the strategy which assigns uniform probabilities to all available actions visits all states infinitely often almost-surely.

Thus, we satisfy the hypotheses of Theorem 4, and a polynomial-time algorithm follows. □

The exact query complexity of the lim inf and inf cases are left open.

5 Mean-payoff

We consider the multi-constraint percentile problem both for $\underline{\mathsf{MP}}$ and $\overline{\mathsf{MP}}$. We will see that strategies require infinite memory in both cases, in which case it is known that the two payoff functions differ. The *single-constraint* percentile problem was first solved in [19]. The case of multiple dimensions was mentioned as a challenging problem but left open. We solve this problem thus generalizing the previous work.

5.1 The single-dimensional case

We start with a polynomial-time algorithm for the single-dimensional case obtained via Theorem 4, thus extending the results of [19] to multi-constraint percentile queries.

Theorem 9 *The single dimensional multi-constraint percentile problems for payoff functions* $\underline{\mathsf{MP}}$ *and* $\overline{\mathsf{MP}}$ *are equivalent and solvable in polynomial time. Computed strategies use constant memory.*

Proof Let $C_1, \ldots, C_m$ be the MECs of a given MDP M. If we define $v^*(C_i) = \sup_{\sigma \in \Sigma} \mathbb{E}^{\sigma}_{C_i,s}[\underline{\mathsf{MP}}] = \sup_{\sigma \in \Sigma} \mathbb{E}^{\sigma}_{C_i,s}[\overline{\mathsf{MP}}]$, then for each $1 \leq i \leq m$, there exists a strategy σ_i, computable in polynomial time, with the property $\mathbb{P}^{\sigma_i}_{M,s}[\underline{\mathsf{MP}} = v^*(C_i)] = 1$ [27]. In other terms, optimal strategies exist for single dimensional mean-payoff, and the optimal value can be achieved almost-surely inside strongly connected MDPs. In contrast, no value greater than the optimal value can be achieved with positive probability. The polynomial-time algorithm then follows from Theorem 4.

The equivalence between $\underline{\mathsf{MP}}$ and $\overline{\mathsf{MP}}$ follows from the fact that they are equivalent inside MECs since memoryless strategies exist, and that the strategy of Theorem 4 almost-surely eventually switches to an optimal strategy for a MEC. □

5.2 Percentiles on multi-dimensional $\overline{\mathsf{MP}}$

Let $\mathbb{E}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_i]$ be the *expectation* of $\overline{\mathsf{MP}}_i$ under strategy σ, and $\mathsf{Val}^*_{M,s_{\mathsf{init}}}(\overline{\mathsf{MP}}_i) = \sup_\sigma \mathbb{E}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_i]$, computable in polynomial time [27]. We solve the problem inside ECs, then apply Theorem 4. It is known that for strongly connected MDPs, for each i, some strategy σ satisfies $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_i = \mathsf{Val}^*_{M,s_{\mathsf{init}}}(\overline{\mathsf{MP}}_i)] = 1$, and that for all strategies τ, $\mathbb{P}^{\tau}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_i > v] = 0$ for all $v > \mathsf{Val}^*_{M,s_{\mathsf{init}}}(\overline{\mathsf{MP}}_i)$. By switching between these optimal strategies for each dimension, with growing intervals, we prove that for strongly connected MDPs, a single strategy can simultaneously optimize $\overline{\mathsf{MP}}_i$ on *all* dimensions.

We first recall the following result on the convergence speed of optimal memoryless strategies in MDPs.

Lemma 6 ([32]) *Let M be any single-dimensional weighted MDP, v^* be defined as the optimal value $v^* = \sup_\sigma \mathbb{E}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}]$, and σ an optimal memoryless strategy with $v^* = \mathbb{E}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}]$. For all $\varepsilon > 0$ and $\eta > 0$, there exists $K_0 > 0$ such that for all $K \geq K_0$, $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\{s_1a_1s_2a_2 \ldots \mid \frac{1}{K}\sum_{i=1}^{K} w(a_i) \geq v^* - \varepsilon\}] \geq 1 - \eta$.*

We now show that strongly connected multi-dimensional MDPs, a single strategy can simultaneously optimize $\overline{\mathsf{MP}}$, on *all* dimensions.

Lemma 7 *For any strongly connected MDP M, there is an infinite-memory strategy σ such that $\forall i \in \{1, \ldots, d\}$, $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_i \geq \mathsf{Val}^*_{M,s_{\mathsf{init}}}(\overline{\mathsf{MP}}_i)] = 1$.*

Proof Let us write $v_i^* = \mathsf{Val}^*_{M,s_{\mathsf{init}}}(\overline{\mathsf{MP}}_i)$, and let σ_i be a memoryless optimal strategy for this dimension. We define a strategy that switches between these strategies σ_i with growing time intervals. We fix $\eta \in (0, 1)$, and define the sequence $\varepsilon_i = \frac{1}{i}$. Let $t_1 = 1$. For $i \geq 2$, if K_0 the bound given by Lemma 6 for ε_i and η, we choose $t_i \geq K_0$ such that $t_i \geq i^2 \sum_{j=1}^{i-1} t_j$. Strategy σ is defined by running σ_j during t_i steps where $j = (i \mod d) + 1$. Let us define $\alpha_i = \sum_{j=1}^{i} t_j$.

We now prove that σ achieves the optimal value at each dimension with probability 1. Let A_i denote the random variable of the i-th action of an execution for a given MDP, initial state, and strategy. Fix any dimension $k \in \{1, \ldots, d\}$. For any i such that $(i \mod d) + 1 = k$, between steps $\alpha_{i-1} + 1$ and α_i, strategy σ_k is memoryless, and by Lemma 6, we have

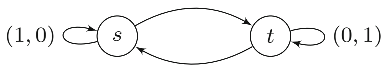
$$\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\left[\frac{1}{t_i}\sum_{j=\alpha_{i-1}+1}^{\alpha_i} w(A_j) \geq v_k^* - \varepsilon_i\right] \geq \eta.$$

Observe that $\frac{t_i}{\alpha_i} = \frac{i^2}{i^2+1}$, and $\frac{\alpha_{i-1}}{\alpha_i} = \frac{1}{i^2+1}$. So, with probability η, we get

$$\begin{aligned}\tfrac{1}{\alpha_i}\textstyle\sum_{j=1}^{\alpha_i} w(A_j) &= \tfrac{1}{\alpha_i}\textstyle\sum_{j=1}^{\alpha_{i-1}} w(A_j) + \tfrac{1}{\alpha_i}\sum_{j=\alpha_{i-1}+1}^{\alpha_i} w(A_j)\\ &\geq \tfrac{\alpha_{i-1}}{\alpha_i}\min_{a\in A} w(a) + \tfrac{t_i}{\alpha_i}(v_i^* - \varepsilon_i).\\ &\geq \tfrac{1}{i^2+1}\min_{a\in A} w(a) + \tfrac{i^2}{i^2+1}(v_i^* - \varepsilon_i).\end{aligned}$$

This means that for any $\varepsilon > 0$, there exists i_0 such that for all $i \geq i_0$ with $(i \mod d) + 1 = k$, we have

$$\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\left[\frac{1}{\alpha_i}\sum_{j=1}^{\alpha_i} w(A_j) \geq v_i^* - \varepsilon\right] \geq \eta,$$



$$\begin{tikzcd}(1,0) & s \arrow[l, phantom] \arrow[loop left] \arrow[r, bend left] & t \arrow[l, bend left] \arrow[loop right] & (0,1)\end{tikzcd}$$

Fig. 4 Infinite-memory strategies are necessary for $\overline{\mathsf{MP}}$

so $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_k \geq v^*_k - \varepsilon] = 1$ for all $\varepsilon{>}0$. It follows that $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\overline{\mathsf{MP}}_k \geq v^*_k] = 1$ for all dimensions k. ◻

Thanks to the above lemma, we fulfill the hypotheses of Theorem 4, and we obtain the following theorem.

Theorem 10 *The multi-dimensional percentile problem for* $\overline{\mathsf{MP}}$ *is solvable in polynomial time. Strategies use infinite-memory, which is necessary.*

To see that infinite-memory strategies are necessary for $\overline{\mathsf{MP}}$, consider the MDP of Fig. 4 where thresholds $v_1 = v_2 = 1$ can be achieved almost-surely by the above theorem, but not by any finite-memory strategy. The proof is identical to the case of maximizing the expectation in [9, Lemma 7] where it is proved for the case of deterministic MDPs (i.e., automata).

5.3 Percentiles on multi-dimensional $\underline{\mathsf{MP}}$

In contrast with the $\overline{\mathsf{MP}}$ case, our algorithm for $\underline{\mathsf{MP}}$ is more involved, and requires new techniques. In fact, the case of end-components is already non-trivial for $\underline{\mathsf{MP}}$, since there is no single strategy that satisfies all percentile constraints in general, and one cannot hope to apply Theorem 4 as we did in previous sections. We rather need to consider the set of strategies σ_I satisfying *maximal* subsets of percentile constraints; these are called *maximal strategies*. We then prove that any strategy satisfying all percentile queries can be written as a *linear combination* of maximal strategies, that is, there exists a strategy which chooses and executes each σ_I following a probability distribution.

For general MDPs, we first consider each MEC separately and write down the linear combination with unknown coefficients. We know that any strategy in a MDP eventually stays forever in a MEC. Thus, we adapt the linear program of [18] that encodes the reachability probabilities with multiple targets, which are the MECs here. We combine these reachability probabilities with the unknown linear combination coefficients, and obtain a linear program (Fig. 5), which we prove to be equivalent to our problem.

Single EC Fix a strongly connected d-dimensional MDP M and pairs of thresholds $(v_i, \alpha_i)_{1\leq i\leq q}$. We denote each event by $A_i \equiv \underline{\mathsf{MP}}_i \geq v_i$. In [6], the problem of maximizing the *joint* probability of the events A_i was solved in polynomial time. In particular, we have the following for strongly connected MDPs.

Lemma 8 ([6]) *If M is strongly connected, then there exists a strategy σ such that $\mathbb{P}^{\sigma}_{M,s}[\wedge_{1\leq i\leq q} A_i]{>}0$ if, and only if there exists σ' such that $\mathbb{P}^{\sigma'}_{M,s}[\wedge_{1\leq i\leq q} A_i] = 1$. Moreover, this can be decided in polynomial time, and for positive instances, for any $\varepsilon{>}0$, a memoryless strategy τ can be computed in polynonomial time in M, $\log(v_i)$ and $\log(\frac{1}{\varepsilon})$, such that $\mathbb{P}^{\tau}_{M,s}[\wedge_{1\leq i\leq q}\underline{\mathsf{MP}}_i \geq v_i - \varepsilon] = 1$.*

We give an overview of our algorithm. Using Lemma 8, we define strategy σ_I achieving $\mathbb{P}^{\sigma_I}_{M,s}[\wedge_{i\in I} A_i] = 1$ for any maximal subset $I \subseteq \{1, \ldots, q\}$ for which such a strategy exists. Then, to build a strategy for the multi-constraint problem, we look for a linear combination of these σ_I: given $\sigma_{I_1}, \ldots, \sigma_{I_m}$, we choose each $i_0 \in \{1, \ldots, m\}$ following a probability distribution to be computed, and we run $\sigma_{I_{i_0}}$.

$$\mathbf{1}_{s_{\mathsf{init}}}(s) + \sum_{s' \in S, a \in A(s')} y_{s',a}\delta(s', a, s) = \sum_{a \in A'(s)} y_{s,a}, \qquad \forall s \in S, \tag{2}$$

$$\sum_{s \in S_{\mathrm{MEC}}} y_{s,a^*} = 1, \tag{3}$$

$$\sum_{s \in C} y_{s,a^*} = \sum_{I \in \mathcal{I}^C} \lambda_I^C, \quad \forall C \in \mathsf{MEC}(M), \tag{4}$$

$$\lambda_I^C \geq 0, \quad \forall C \in \mathsf{MEC}(M), \forall I \in \mathcal{I}^C, \tag{5}$$

$$\sum_{C \in \mathsf{MEC}(M)} \sum_{I \in \mathcal{I}^C : i \in I} \lambda_I^C \geq \alpha_i, \qquad \forall i = 1 \ldots d. \tag{6}$$

Fig. 5 Linear program (L) for the multi-constraint percentiles for $\underline{\mathsf{MP}}$

We now formalize this idea. Let $\mathcal{I}$ be the set of maximal I (for set inclusion) such that some σ_I satisfies $\mathbb{P}^{\sigma_I}_{M,s}[\wedge_{i \in I} A_i] = 1$. Note that for all $I \in \mathcal{I}$, and $j \notin I$, $\mathbb{P}^{\sigma_I}_{M,s}[\wedge_{i \in I} A_i \wedge A_j] = 0$. Assuming otherwise would contradict the maximality of I, by Lemma 8. We consider the events $\mathcal{A}_I = \wedge_{i \in I} A_i \wedge_{i \notin I} \neg A_i$ for maximal I.

We are looking for a non-negative family $(\lambda_I)_{I \in \mathcal{I}}$ whose sum equals 1 and such that $\forall i \in \{1, \ldots, q\}$, $\sum_{I \in \mathcal{I} \text{ s.t. } i \in I} \lambda_I \geq \alpha_i$. This will ensure that if each σ_I is chosen with probability λ_I (among the set $\{\sigma_I\}_{I \in \mathcal{I}}$); with probability at least α_i, some strategy satisfying A_i with probability 1 is chosen. So each A_i is satisfied with probability at least α_i. This can be written in the matrix notation as

$$\mathcal{M}\boldsymbol{\lambda} \geq \boldsymbol{\alpha}, \quad 0 \leq \boldsymbol{\lambda}, \quad \mathbf{1} \cdot \boldsymbol{\lambda} = 1, \tag{1}$$

where $\mathcal{M}$ is a $q \times |\mathcal{I}|$ matrix with $\mathcal{M}_{i,I} = 1$ if $i \in I$, and 0 otherwise.

Lemma 9 *For any strongly connected MDP M, and an instance $(v_i, \alpha_i)_{1 \leq i \leq q}$ of the multi-constraint percentile problem for $\underline{\mathsf{MP}}$, (1) has a solution if, and only if there exists a strategy σ satisfying the multi-constraint percentile problem.*

Proof Assume (1) and consider the strategy $\sum_{I \in \mathcal{I}} \lambda_I \sigma_I$, which means that at the beginning of the run, we choose each set I with probability λ_I, and run σ_I. Clearly, the probability of satisfying A_i is at least the probability of running a strategy σ_I such that $i \in I$, which is $\sum_{I \in \mathcal{I} : i \in I} \lambda_I$. The result follows.

Conversely, let σ denote a strategy satisfying $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[A_i] \geq \alpha_i$ for all i. Let us consider all events $\mathcal{A}_I$ including non-maximal I. The events $\mathcal{A}_I$ are disjoint and we have $A_i = \cup_{I : i \in I} \mathcal{A}_I$. It follows that $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[A_i] = \sum_{I : i \in I} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathcal{A}_I] = \sum_{I : i \in I} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathcal{A}_I] \cdot \mathbb{P}^{\sigma_I}_{M,s_{\mathsf{init}}}[\mathcal{A}_I]$ since $\mathbb{P}^{\sigma_I}_{M,s_{\mathsf{init}}}[\mathcal{A}_I] = 1$ by definition. In order to derive a probability distribution on *maximal* subsets only, we define a partition of $2^{\{1,\ldots,q\}}$ by assigning each non-maximal J to a maximal $I \in \mathcal{I}$ with $J \subseteq I$. Formally, we consider sets $\alpha(I) \subseteq 2^{\{1,\ldots,q\}}$ with $I \in \alpha(I)$, such that for all $J \in \alpha(I)$, $J \subseteq I$, and $\{\alpha(I)\}_{I \in \mathcal{I}}$ defines a partition of $2^{\{1,\ldots,q\}}$. For any $I \in \mathcal{I}$, we set $\lambda_I = \sum_{J \in \alpha(I)} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathcal{A}_J]$. This yields a solution of (1). □

Now (1) has size $O(q \cdot 2^q)$, and each subset I can be checked in time polynomial in the model size. The computation of $\mathcal{I}$, the set of maximal subsets, can be carried out in a top-down fashion; one might thus avoid enumerating all subsets in practice. We get the following result.

Lemma 10 *For strongly connected MDPs, the multi-dimensional percentile problem for $\underline{\mathsf{MP}}$ can be solved in time polynomial in M and exponential in q. Strategies require* *infinite-memory in general. On positive instances, 2^q-memory randomized strategies can be computed for the ε-relaxation of the problem in time polynomial in $|M|$, 2^q, and* $\max_i \big(\log(v_i), \log(\alpha_i)\big), \log(\frac{1}{\varepsilon})$.

Proof The first statement is clear from the two previous lemmas, since (1) can be solved in time polynomial in M and exponential in q. For the ε-relaxation problem, notice that once we compute the set $\mathcal{I}$ and solve (1), for any set $I \in \mathcal{I}$, we compute in polynomial time a randomized strategy σ_I ensuring $A_i^\varepsilon = \wedge_{i\in I}\underline{\mathsf{MP}}_i \geq v_i - \varepsilon$. This can be done as in [6]. Then the strategy choosing randomly each σ_I with probability σ_I ensures all bounds up to ε (i.e., $v_i - \varepsilon$).

The need for infinite memory was proved in [6, Section 5] for the problem of ensuring thresholds $\mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[\underline{\mathsf{MP}}_1 \geq v_1 \wedge \ldots \underline{\mathsf{MP}}_2 \geq v_2] \geq \alpha$ for thresholds v_1, v_2 and probability α. It was proved that on the MDP of Fig. 4, $v_1 = v_2 = 0.5$ and $\alpha = 1$ can be ensured by an infinite-memory strategy and that finite-memory strategies can only achieve these thresholds with probability 0. Now, if the multi-constraint percentile query $\mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[\underline{\mathsf{MP}}_1 \geq v_1] \geq 0.6 \wedge \mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[\underline{\mathsf{MP}}_2 \geq v_2] \geq 0.6$ has a solution by a strategy σ, then we must have that $\mathbb{P}^\sigma_{M,s_{\mathsf{init}}}[\underline{\mathsf{MP}}_1 \geq v_1 \wedge \underline{\mathsf{MP}}_2 \geq v_2] \geq 0.2$ (this simply follows from the fact that $0.6+0.6 = 1.2$). Therefore σ must use infinite-memory. □

General MDPs Given MDP M, let us consider M' given by Lemma 4. We start by analyzing each maximal EC C of M as above, and compute the sets $\mathcal{I}^C$ of maximal subsets. We define a variable λ_I^C for each $I \in \mathcal{I}^C$, and also $y_{s,a}$ for each state s and action $a \in A'(s)$. Recall that $A'(s) = A(s) \cup \{a^*\}$ for states s that are inside a MEC, and $A'(s) = A(s)$ otherwise. Let S_{MEC} be the set of states of M that belong to a MEC. We consider the linear program (L) of Fig. 5.

We prove the following main lemma in this section.

Lemma 11 *The LP (L) has a solution if, and only if the multi-constraint percentiles problem for* $\underline{\mathsf{MP}}$ *has a solution. Moreover, the equation has size polynomial in M and exponential in q. From any solution of (L) randomized finite memory strategies can be computed for the ε-relaxation problem.*

The linear program follows the ideas of [6,18]. Note that the first two lines of (L) corresponds to the multiple reachability LP of [18] for absorbing target states. The equations encode strategies that work in two phases. Variables $y_{s,a}$ correspond to the expected number of visits of state-action s, a in the first phase. Variable y_{s,a^*} describes the probability of switching to the second phase at state s. The second phase consists in surely staying in the current MEC, so we require $\sum_{s\in S_{\mathsf{MEC}}} y_{s,a^*} = 1$ (and we will have $y_{s,a^*} = 0$ if s does not belong to a MEC). In the second phase, we immediately switch to some strategy σ_I^C where C denotes the current MEC. Thus, variable λ_I^C corresponds to the probability with which we enter the second phase in C and switch to strategy σ_I^C [see (4)]. Intuitively, given a solution $(\lambda_I)_I$ computed for one EC by (1), we have the correspondence $\lambda_I^C = \sum_{s\in C} y_{s,a^*} \cdot \lambda_I$. The interpretation of (6) is that each event A_i is satisfied with probability at least α_i.

The two following lemmas prove Lemma 11.

Lemma 12 *If (L) has a solution then there exists a strategy for the multi-constraint percentile problem. Moreover, from any solution of (L) one can derive in time polynomial in M,* $\log\big(\frac{1}{\varepsilon}\big)$, *and exponential in q, a $\mathcal{O}(2^q)$-memory randomized strategy solving the ε-relaxation of the multi-constraint percentile problem.*

Proof Let $\bar{y_{s,a}}, \bar{y_{s,a^*}}, \bar{\lambda_I^C}$ be a solution of (L). By [18, Theorem 3.2], there exists a memoryless strategy ρ for M' such that $\mathbb{P}^{\rho}_{M',s_{\mathsf{init}}}[\Diamond s_C] = \sum_{s\in C} y_{s,a^*}$ for each MEC C, and $\sum_{C\in\mathsf{MEC}(M)} \mathbb{P}^{\rho}_{M',s_{\mathsf{init}}}[\Diamond s_C] = 1$ by the second line. In this strategy, y_{s,a^*} is the probability of going to s_{C_i} from s.

For each MEC C, we define the strategy σ^C for M which, from the states of C, executes each strategy σ_I for $I \in \mathcal{I}^C$ with probability $\frac{\lambda_I^C}{\sum_{J\in\mathcal{I}^C}\lambda_J^C} = \frac{\lambda_I^C}{\sum_{s\in C} y_{s,a^*}}$, if the denominators are positive, and with an arbitrary distribution otherwise. We combine these in a strategy σ for M which starts by simulating ρ until ρ chooses takes the action a^*, at which point σ switches to σ^C.

By construction the probability of σ of switching to σ_I^C is

$$\sum_{s\in C} y_{s,a^*} \cdot \frac{\lambda_I^C}{\sum_{s\in C} y_{s,a^*}} = \lambda_I^C,$$

for any C and $I \in \mathcal{I}^C$. Moreover, thanks to the fact that $\sum_{s\in S_{\mathsf{MEC}}} y_{s,a^*} = 1$, we know that σ will eventually switch to some σ^C almost-surely. Because for all I and C such that $i \notin I$, the probability of σ_I^C of satisfying A_i inside C is 0 (see above), we get that the probability of satisfying A_i under σ is equal to the probability of switching to some λ_I^C. But thanks to the last line of the program, this quantity is at least α_i. Hence, σ satisfies the multi-constraint percentile problem.

We obtain a strategy for the relaxed problem as follows. Each strategy σ_I^C may be infinite-memory a priori but for any $\varepsilon>0$, we can compute by Lemma 8, memoryless randomized strategies τ_I^C ensuring $\mathcal{A}_I^\varepsilon$ with probability 1. Now, since ρ is also memoryless, the combined strategy only needs $2^q + 1$ memory elements (to store the phase, and which I it has chosen once in a MEC). The result follows straightforwardly. □

Lemma 13 *If strategy σ solves the multi-constraint percentile problem for $\underline{\mathsf{MP}}$, then (L) has a solution.*

Proof Let $C_1, \ldots, C_m$ denote the MECs, and define $y_{C_i} = \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C_i]$ for each i. Clearly, we have $\sum_i y_{C_i} = 1$. Let ρ denote the strategy on M' of Lemma 4 given for σ. For any action $a \in C_i$, let $y_{s,a}$ denote the expected number of times action a is taken at s under ρ in M' starting at s_{init}. Now, [18, Lemma 3.3] ensures that these variables have finite values and satisfy the first two lines of (L).

We define strategy σ' for M which follows ρ until action a^* is taken, at which point it switches to each strategy σ_I^C with probability $\mathbb{P}^{\sigma}_{M,s_0}[\mathcal{A}_I \mid \mathsf{Inf}(\rho) = C]$ (these include non-maximal sets I). We have that

$$\mathbb{P}^{\sigma'}_{M,s_{\mathsf{init}}}[A_i] = \sum_{C\in\mathsf{MEC}(M)} \mathbb{P}^{\sigma'}_{M,s_{\mathsf{init}}}[A_i \mid \mathsf{Inf}(\rho) = C]\mathbb{P}^{\sigma'}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C].$$

By definition of σ', the first term in the sum equals $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[A_i \mid \mathsf{Inf}(\rho) = C]$. The second term in the sum is equal to $\mathbb{P}^{\rho}_{M',s_{\mathsf{init}}}[\Diamond s_C] = \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C]$. It follows that $\mathbb{P}^{\sigma'}_{M,s_{\mathsf{init}}}[A_i] = \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[A_i]$.

Now, to obtain a solution of (L), it remains to get rid of the strategies σ_I for non-maximal subsets I for each MEC. We thus modify once more σ' to obtain σ'' as follows. Whenever σ' switches to some strategy σ_I^C, where I is not maximal, we rather switch to some σ_J^C for some -arbitrarily chosen- maximal $J \supset I$. It is clear that $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C] = \mathbb{P}^{\sigma''}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C]$ and $\mathbb{P}^{\sigma''}_{M,s_{\mathsf{init}}}[\mathcal{A}_I \mid \mathsf{Inf}(\rho) = C] \geq \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathcal{A}_I \mid \mathsf{Inf}(\rho) = C]$ for all $I \in \mathcal{I}^C$.

Now, for each C and $I \in \mathcal{I}^C$, we define $\lambda_I^C = \mathbb{P}^{\sigma''}_{M,s_{\mathsf{init}}}[\mathcal{A}_I \wedge \mathsf{Inf}(\rho) = C]$. It is easy to verify that $0 \leq \lambda_I^C$ and $\sum_{I \in \mathcal{I}^C} \lambda_I^C = \sum_{s \in C} y_{s,a^*}$, so (4) and (5) are satisfied. We have $\sum_{I \in \mathcal{I}^{C_i}} \lambda_I^{C_i} = \mathbb{P}^{\sigma''}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C_i] = \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[\mathsf{Inf}(\rho) = C_i] = y_{C_i}$ for all $i = 1 \ldots d$. Moreover, $\mathbb{P}^{\sigma''}_{M,s_{\mathsf{init}}}[A_i] = \sum_{C \in \mathsf{MEC}(M)} \sum_{I \in \mathcal{I}^C : i \in I} \mathbb{P}^{\sigma''}_{M,s_{\mathsf{init}}}[\mathcal{A}_I \wedge \mathsf{Inf}(\rho) = C]$. This is at least equal to $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}[A_i]$ as we saw above, which is at least α_i by assumption; hence (6) is also satisfied. □

Our results for the multi-dimensional problems with $\underline{\mathsf{MP}}$ are summed up in the next theorem.

Theorem 11 *The multi-dimensional percentile problem for $\underline{\mathsf{MP}}$ can be solved in time polynomial in the model, and exponential in the query. Infinite-memory strategies are necessary, but exponential-memory (in the query) suffices for the ε-relaxation and can be computed with the same complexity.*

6 Shortest path

We study shortest path problems in MDPs, which generalize the classical graph problem. In MDPs, the problem consists in finding a strategy ensuring that a target set is reached with bounded truncated sum with high probability. This problem has been studied in the context of games and MDPs (e.g., [2,7,17]). We consider percentile queries of the form $\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\big[\mathsf{TS}^{T_i}_{l_i} \leq v_i\big] \geq \alpha_i$ (inner inequality $\leq$ is more natural but $\geq$ could be used by negating all weights). Observe that each constraint i may relate to a different target set $T_i \subseteq S$.

6.1 MDPs with arbitrary weights

We prove that without further restriction, the multi-dimensional percentile problem is undecidable, even for a fixed number of dimensions. Our proof is inspired by the approach of Chatterjee et al. [10] for the undecidability of two-player multi-dimensional total-payoff games but requires additional techniques to adapt to the stochastic case.

Theorem 12 *The multi-dimensional percentile problem is undecidable for the truncated sum payoff function, for MDPs with both negative and positive weights and four dimensions, even with a unique target set.*

Proof We reduce the halting problem for two-counter machines (2CMs) to a multi-dimensional percentile problem for the truncated sum payoff function over an MDP with weights in $\mathbb{Z}^4$, with a unique target set.

Counters of a 2CM take values $(v_1, v_2) \in \mathbb{N}^2$ along an execution, and can be incremented or decremented (if positive). A counter can be tested for equality to zero, and the machine can branch accordingly. The halting problem for 2CMs is well-known to be undecidable [25].

Consider a 2CM $\mathcal{M}$. From this 2CM, we construct an MDP $M = (S, A, \delta, w)$ and a target set of states $T \subset S$, with an initial state $s_{\mathsf{init}} \in S$ such that there exists a strategy $\sigma \in \Sigma$ satisfying the four-dimensional percentile query

$$\mathcal{Q} := \bigwedge_{i=1}^{4} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\big[\mathsf{TS}^{T}_{l_i} \leq 0\big] = 1.$$

if and only if the machine does not halt.

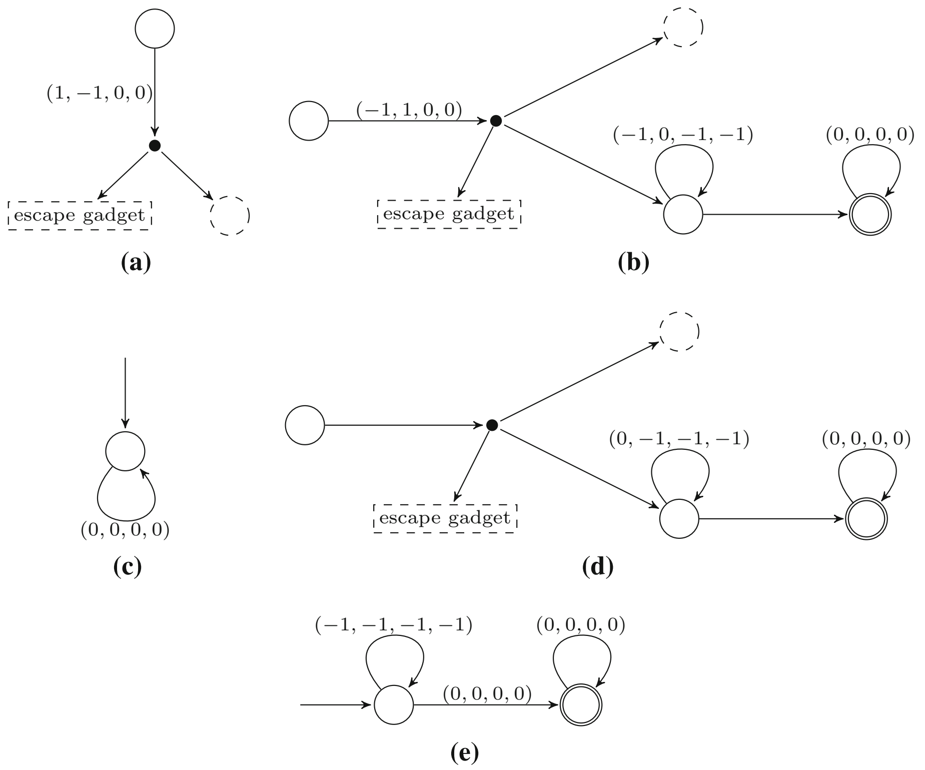


Fig. 6 Gadgets encoding 2CM halting problem in a multi-dimensional percentile problem for truncated sum payoff function. **a** Increment C_1. **b** Decrement C_1. **c** Halting. **d** Checking that C_1 is equal to zero. **e** Escape gadget reachable by every action of the MDP

Intuitively, this MDP is built such that strategies that do not faithfully simulate the 2CM $\mathcal{M}$ cannot satisfy the percentile query. To ensure that this is the case, we will implement checks through probabilistic transitions that will produce bad runs with positive probability against unfaithful strategies.

The MDP M is built as follows. The states of M are copies of the control states of $\mathcal{M}$ (plus some special states discussed in the following). Actions in the MDP represent transitions between these control states. The weight function maps actions to 4-dimensional vectors of the form $(c_1, -c_1, c_2, -c_2)$, that is, two dimensions for the first counter C_1 and two for the second counter C_2. Each increment of counter C_1 (resp. C_2) in $\mathcal{M}$ is implemented in M as an action of weight $(1, -1, 0, 0)$ (resp. $(0, 0, 1, -1)$). For decrements, we have weights respectively $(-1, 1, 0, 0)$ and $(0, 0, -1, 1)$ for C_1 and C_2. Therefore, the current value of counters (v_1, v_2) along an execution of the 2CM $\mathcal{M}$ is represented in the MDP as the current sum of weights, $(v_1, -v_1, v_2, -v_2)$. Hence, along a faithful execution, the 1st and 3rd dimensions are always non-negative, while the 2nd and 4th are always non-positive. The two dimensions per counter are used to enforce faithful simulation of non-negativeness of counters and zero test.

We now discuss how this MDP M ensures faithful simulation of the 2CM $\mathcal{M}$ by the controller through the use of the gadgets represented in Fig. 6. Small filled circles represent equiprobable stochastic transitions, double circles depict states of the target set T.

- An *escape gadget* has a positive probability to be reached whenever an instruction of the 2CM is simulated. When in this gadget, the controller can decrease the sum on all dimensions below zero by cycling long enough before deciding to reach the target, hence making the run acceptable for the considered percentile query.
- *Increment and decrement* of counter values are easily simulated using the first four dimensions.
- *Values of counters may never go below zero*. To ensure this, we make every decrement action probabilistic with three equiprobable outcomes. Consider the decrement of C_1 as depicted in Fig. 6b. Either the simulation continues (dashed control state), or it branches to the escape gadget, or it branches to the bottom right part of the decrement gadget. In that case, the controller can cycle long enough in the first state to ensure a negative sum of weights in all dimensions except for the second one, before reaching the target (runs *have to* reach the target or their truncated sum will be ∞). If the controller is not faithful and a negative value is reached on counter C_1 when decrementing, this branching will induce a run which is losing because the second dimension will be strictly positive (recall it has value $-c_1$). Notice that the controller can never cheat, otherwise this branching will happen with strictly positive probability (i.e., after a finite prefix). On the contrary, if the controller never cheats, this branching is harmless and induces acceptable runs w.r.t. the percentile query. The gadget is similar for decrements of C_2 using the fourth dimension.
- *Zero tests are correctly executed*. In the same spirit, we allow a probabilistic branching after the controller claims a counter is equal to zero. Consider Fig. 6d for counter C_1. If a zero test is passed while $c_1 > 0$, the sum on the first dimension will stay strictly positive and the run will not be acceptable. On the contrary, if the controller is faithful, this branching is again harmless as it is possible to make all sums negative except for the first dimension for which it would already be equal to zero. We use a similar gadget for C_2 based on the third dimension. We also need to ensure that the controller cannot cheat by claiming that a counter is strictly positive while it is not. To achieve this, we follow each claim that C_i is strictly positive by a decrement on C_i (using our gadget) followed by an increment (idem). Thus, if C_i is equal to zero while the controller claims the opposite, the decrement gadget will yield non-acceptable runs, as seen before. On the other hand, if the controller is faithful, visiting those two gadgets is safe (and does not modify the counters for the rest of the simulation).
- *Halting*. The end of a 2CM execution is modeled in the MDP by an halting state. This state does not belong to the target set T: any run corresponding to an halting execution will have its truncated sum equal to ∞ on all dimensions, which makes it bad for the percentile query.

Now, we have argued that if the simulation is not faithful, bad runs will be produced with strictly positive probability, and the percentile query will not be satisfied. Furthermore, if the machine halts, then with a strictly positive probability, the halting state will be reached (because the machine halts after a *finite* number of operations), which also results in bad runs. Hence if the percentile query is satisfied by a strategy $\sigma \in \Sigma$, then this strategy describes a faithful infinite execution of $\mathcal{M}$.

It remains to show that if the 2CM does not halt, the percentile query is satisfiable. Clearly, the halting state will never be reached, and gadgets cannot produce bad runs as the simulation is faithful. However, runs that never reach any target state (i.e., runs that never branch away from the simulation) are still bad runs as they yield an infinite truncated sum. Nonetheless, observe that each action taken in the MDP yields a strictly positive probability to branch to the escape gadget or to branch inside decrement and zero-test gadgets. Hence, if the 2CM does not halt, such actions are taken infinitely often and with probability one, the simulation eventually branches toward the target states (with a good truncated sum as argued before). We conclude that the strategy that simulates a never-halting machine does yield good runs with probability one.

Consequently, we have that the studied multi-dimensional percentile problem is equivalent to the 2CM halting problem, and thus, undecidable. □

6.2 MDPs with non-negative weights

In the light of this result, we will restrict our setting to non-negative weights, a setting closer to the original interpretation of the shortest path problem (we could equivalently consider non-positive weights with inequality $\geq$ inside percentile constraints). We first discuss recent related work.

Comparison with quantiles and cost problems In [34], Ummels and Baier study *quantile queries* over non-negatively weighted MDPs. They are equivalent to minimizing $v \in \mathbb{N}$ in a single-constraint percentile query $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\big[\mathsf{TS}^{T} \leq v\big] \geq \alpha$ such that there still exists a satisfying strategy, for some fixed α. Very recently, Haase and Kiefer extended quantile queries by introducing *cost problems* [22]. They can be seen as single-constraint percentile queries where inequality $\mathsf{TS}^{T} \leq v$ is replaced by an arbitrary Boolean combination of inequalities φ. Hence, it can be written as $\mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\big[\mathsf{TS}^{T} \models \varphi\big] \geq \alpha$. Cost problems are studied on single-dimensional MDPs and all the inequalities relate to the same target T, in contrast to our setting which allows both for multiple dimensions and multiple target sets. The single probability threshold bounds the probability of the whole event φ.

Both settings are incomparable. Cost problems consist in a unique query that checks that with probability α, paths satisfy the Boolean combination φ. In percentile queries, we have several constraints and we check that *each* inequality is satisfied with the corresponding probability α_i: paths do not need to satisfy *all* inequalities at the same time. In full generality, for a fixed probability threshold α, it is easier to satisfy a unique constraint over a disjunction of inequalities than to satisfy a disjunction of constraints over single inequalities: in the second case, α percent of the paths must satisfy the *same* unique inequality, not in the first one. Similarly, it is harder to satisfy a unique constraint over a conjunction of inequalities than to satisfy a conjunction of constraints over single inequalities: in the first case, α percent of the paths must satisfy *all* inequalities, not in the second one.

Still, our queries share common subclasses with cost problems: atomic formulae φ exactly correspond to our single-constraint queries. Moreover, cost problems for such formulae are inter-reducible with quantile queries [22, Proposition 2]. Cost problems with atomic formulae are PSPACE-hard, so this also holds for *single-constraint* percentile queries. The best known algorithm in this case is in $\mathsf{EXPTIME}$. In the following, we establish an algorithm that still only requires exponential time while allowing for *multi-constraint multi-dimensional multi-target* percentile queries.

Overview Our main contributions for the shortest path are summarized in Theorem 13. In the following, we detail each of them and discuss some subclasses of queries with interesting complexities.

Theorem 13 *The percentile problem for the shortest path with non-negative weights can be solved in time polynomial in the model size and exponential in the query size (exponential in the number of constraints and pseudo-polynomial in the largest threshold). The prob-*

lem is PSPACE*-hard even for single-constraint queries. Exponential-memory strategies are sufficient and in general necessary.*

Algorithm The sketch of our algorithm is as follows. Consider a d-dimensional MDP M and a q-query percentile problem, with potentially different targets for each query. Let v_{max} be the maximum of the thresholds v_i. Because weights are non-negative, extending a finite history never decreases the sum of its weights. Thus, any history ending with a sum exceeding v_{max} in all dimensions is surely losing under any strategy.

Based on this, we build an MDP M' by unfolding M and integrating the sum for each dimension in states of M'. We ensure its finiteness thanks to the above observation and we reduce its overall size to a *single*-exponential by defining a suitable equivalence relation between states of M': we only care about the current sum in each dimension, and we can forget about the actual path that led to it. Precisely, the states of M' are in $S \times \{0, \ldots, v_{\mathsf{max}} + 1\}^d$. Now, for each constraint, we compute a set of target states in M' that exactly captures all runs satisfying the inequality of the constraint. Thus, we are left with a multiple reachability problem on M': we look for a strategy σ' that ensures that each of these sets R_i is reached with probability α_i. This query can be answered in time polynomial in $|M'|$ but exponential in the number of sets R_i, i.e., in q (Theorem 1).

We prove the correctness of our algorithm in the next lemma.

Lemma 14 *The percentile problem for the shortest path can be solved in time polynomial in the size of the MDP and the thresholds values, and exponential in the number of dimensions of the weight function and the number of constraints of the problem.*

Proof Let $M = (S, A, \delta, w)$ be the considered MDP, with $w : A \to \mathbb{N}^d$ its d-dimensional non-negative weight function, $s_{\mathsf{init}} \in S$ the initial state. We consider a q-constraint query: we are looking for a strategy $\sigma \in \Sigma$ such that

$$\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M, s_{\mathsf{init}}}\left[\mathsf{TS}^{T_i}_{l_i} \leq v_i\right] \geq \alpha_i$$

for the given thresholds $v_i \in \mathbb{N}$, $\alpha_i \in [0, 1] \cap \mathbb{Q}$ and target sets $T_i \subseteq S$. The algorithm is as follows.

Let v_{max} be the maximum of the thresholds v_i, $i \in \{1, \ldots, q\}$. Observe that given any prefix of a run, extending it can never decrease the sum of weights (as all weights are non-negative) and that any run for which the truncated sum exceeds v_{max} in all dimensions is not interesting for the controller.

Based on those observations, we unfold the MDP M, creating a tree-like structure in the nodes of which we integrate the current sum of weights. That is, nodes are labeled by elements of $S \times \mathbb{N}^d$. We stop a branch as soon as the sum reaches $v_{\mathsf{max}} + 1$ in all dimensions (we do not care about what happens after the sum hits this value as it is now a bad outcome for all the percentile constraints). Now, this unfolding is not exactly a tree because we allow actions of weight zero in the original MDPs. Hence we may have to introduce cycles in the unfolding: whenever a branch visits a node with a label identical to one of its ancestors, we stop this branch and introduce a cycle to the corresponding ancestor. Those two cutting criteria guarantee an unfolding which is finite and of maximum height $h = \mathcal{O}(|S| \cdot (v_{\mathsf{max}} + 2) \cdot d)$. That is because every cycle (in the original MDP) that does not result in a cycle in the unfolding has to increase at least one dimension, by at least one, and has at most length $|S|$.

Now consider the overall size of this unfolding. Recall that we want to build an unfolding which is at most exponential. If no special care is taken, the size of the unfolding could be as high as $\mathcal{O}(b^h)$, where b denotes the branching degree of M, defined as

$$b = \max_{s \in S} \left|\{(a, s') \mid a \in A(s), s' \in S, \delta(s, a, s') > 0\}\right|.$$

In particular, the overall size could be exponential in v_{max}, that is, doubly-exponential in its encoding. To avoid that, we reduce the size of the unfolding by merging equivalent nodes.

What are equivalent nodes? First, we declare two nodes to be equivalent if they relate to the same state and describe identical sums on all dimensions. Second, observe that for any node of the unfolding, the sum on any dimension can theoretically grow up to $h \cdot W$, with W the largest weight appearing on any action of M. That is, it can grow larger than $(v_{\mathsf{max}} + 1)$ as we stop only when *all* dimensions are larger than this bound. Nonetheless, w.r.t. satisfaction of the percentile query, we do not need to recall exactly what is the value reached after exceeding v_{max} as in any case, such a sum in a given dimension is not acceptable for any related constraint. Hence, we can also merge nodes by replacing any label larger than $(v_{\mathsf{max}} + 1)$ by label $(v_{\mathsf{max}} + 1)$.

By merging nodes equivalent according to this definition, we ensure that the overall size of the unfolding is at most $u = \mathcal{O}(|S| \cdot (v_{\mathsf{max}} + 2)^d)$. Indeed, the possible values for sums on any dimension in the unfolding run from 0 to $(v_{\mathsf{max}} + 1)$. Observe that this overall size u is, as desired, polynomial in the number of states $|S|$ and in the largest threshold v_{max}, and exponential in the number of dimensions d.

Interestingly, this merging process can be executed on the fly while building the unfolding hence does not hinder the total execution time of the algorithm (i.e., one does not have to fully build the doubly-exponential unfolding to construct the single-exponential merged one).

Now notice that this unfolding is itself an MDP, denoted M'. For each constraint $i \in \{1, \dots, q\}$, we can compute the set R_i of nodes that are labeled by a state in the corresponding target set T_i and have a label less than or equal to v_i on the corresponding sum dimension l_i. Hence, such a set R_i actually captures all branches satisfying the inequality of constraint i (a branch is captured if it possesses a node of R_i). Observe that we only consider dimensions related to constraint i when computing the set R_i (e.g., it is not a problem to exceed v_{max} in other dimensions). This computation takes time $\mathcal{O}(u \cdot q)$ in the worst case.

Now we are left with a multiple reachability problem on M': we have to decide the existence of a strategy σ' satisfying the query

$$\mathcal{Q}' := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{M', s'_{\mathsf{init}}}\left[\Diamond R_i\right] \geq \alpha_i.$$

If such a strategy σ' exists in M', it is easy to see that the equivalent strategy σ in the original MDP M satisfies the shortest path percentile query $\mathcal{Q}$. Indeed, the probability of reaching set R_i in M' following strategy σ' is exactly the probability of satisfying constraint i in M following the equivalent strategy σ. On the contrary, if no strategy satisfies the multiple reachability query $\mathcal{Q}'$, it implies that the original percentile query $\mathcal{Q}$ cannot be satisfied either.

Solving this query $\mathcal{Q}'$ can be done in polynomial time in the size of the unfolding MDP M' but exponential in the number of sets R_i, i.e., in the number of constraints (Theorem 1). Hence, the overall time required by the algorithm is polynomial in $|S|$ and in the maximum threshold v_{max}, and exponential in the number of dimensions d and in the number of constraints q.

□

It is worthwhile to mention that the exponential dependency on the number of constraints can be lifted when they all consider the same target set.

Remark 1 Percentile problems with unique target are solvable in time polynomial in the number of constraints but still exponential in the number of dimensions.

Proof Consider the unfolding algorithm described in the proof of Lemma 14. Assume that all constraints of the percentile query relate to the same target set $T \subseteq S$. In that case, all branches can be stopped as soon as they reach a state of T. Thus, when computing the sets R_i for the multiple reachability problem on the unfolding M', all nodes that belong to these sets are actually leaves of the unfolding. Hence they are absorbing states of M'. From Theorem 1, it follows that the multiple reachability problem can be solved in polynomial time, which eliminates the exponential dependency on the number of constraints for solving the shortest path percentile problem with a single target set. □

For single-dimensional queries with a unique target set (but still potentially multi-constraint), our algorithm remains pseudo-polynomial as it requires polynomial time in the thresholds values (i.e., exponential in their encoding).

Corollary 1 *The single-dimensional percentile problem with a unique target set can be solved in pseudo-polynomial time.*

Lower bound By equivalence with cost problems for atomic cost formulae, it follows from [22, Theorem 7] that no truly-polynomial-time algorithm exists for the single-constraint percentile problem unless $\mathsf{P} = \mathsf{PSPACE}$.

Lemma 15 *The single-constraint percentile problem for the shortest path is* PSPACE*-hard.*

Memory We now formally prove the need for (and sufficiency of) exponential memory in shortest path percentile queries.

Lemma 16 *Exponential-memory strategies are both sufficient and, in general, necessary to satisfy percentile queries for the shortest path.*

Proof First, the algorithm given in Lemma 14 solves the percentile problem by answering a multiple reachability problem over an unfolded MDP of exponential size. As stated in Theorem 1, memory of size polynomial in the MDP (here, the unfolded one) and exponential in the number of constraints (which is untouched by our algorithm) is sufficient to satisfy such queries. Hence, it follows that exponential-memory strategies suffice for shortest path percentile queries.

Second, let us show that multiple reachability problems over an MDP M can be reduced to shortest path percentile problems over the very same MDP, enriched with a trivial weight function. Consider an unweighted MDP $M = (S, A, \delta)$ and a multiple reachability query for sets $T_i \subseteq S$ and thresholds $\alpha_i \in [0, 1] \cap \mathbb{Q}$, with $i \in \{1, \ldots, q\}$. Let $M' = (S, A, \delta, w)$ be a single-dimensional weighted version of the MDP M, where all actions are assigned weight zero. Then we trivially have that a strategy σ satisfies the multiple reachability query on M if and only if it satisfies the percentile query $\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M',s_{\mathsf{init}}}\big[\mathsf{TS}^{T_i} \leq 0\big] \geq \alpha_i$ on M'. Indeed, runs that are bad for this percentile query are exactly the ones that are assigned truncated sum ∞ because they do not reach the considered target sets. This concludes the reduction.

Finally, we know by Lemma 3 that exponential memory is needed in general for multiple reachability queries. This lower bound thus straightforwardly carries over to shortest path percentile queries. □

7 Discounted sum

The *discounted sum* accumulates weights using a discount factor to model that short-term rewards or costs are more important than long-term ones. It is well-studied in automata [3] and MDPs [11,14,27]. We consider queries of the form $\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\big[\mathsf{DS}^{\lambda_i}_{l_i} \geq v_i\big] \geq \alpha_i$, for discount factors $\lambda_i \in \,]0, 1[\,\cap \mathbb{Q}$ and the usual thresholds. That is, we study multi-dimensional MDPs and possibly distinct discount factors for each constraint.

Unfortunately, our setting encompasses a much simpler question which is still not known to be decidable. We discuss this question and its reduction to our problem in Sect. 7.1. We also argue that solving this problem would require an important breakthrough. Then, in Sect. 7.2, we establish a conservative algorithm that, in some sense, can approximate the answer to the percentile problem.

7.1 Precise discounted sum is hard

Consider the *precise discounted sum problem*: given a rational t, and a rational discount factor $\lambda \in \,]0, 1[$, does there exist an infinite binary sequence $\tau = \tau_1\tau_2\tau_3 \ldots \in \{0, 1\}^{\omega}$ such that $\sum_{j=1}^{\infty} \lambda^j \cdot \tau_j = t$? In [4], this problem is related to several long-standing open questions, such as decidability of the *universality problem for discounted-sum automata* [3]. A slight generalization to paths in graphs is also mentioned by Chatterjee et al. [11] as a key open problem.

Lemma 17 *The precise discounted sum problem can be reduced to an almost-sure percentile problem over a two-dimensional MDP with only one state.*

Proof Assume we have a precise discounted sum problem with discount factor $\lambda \in \,]0, 1[\,\cap \mathbb{Q}$ and target value $t \in \mathbb{Q}$. Let M be an MDP with only one state s and two actions, a and b (that both cycle on s with probability one, obviously). Consider the two-dimensional weight function $w : A \to \mathbb{Z}^2$ such that $w(a) = (0, 0)$ and $w(b) = (1, -1)$. The role of action a (resp. b) is to represent the choice of 0 (resp. 1) in the binary sequence.

We define the percentile problem asking for the existence of a strategy $\sigma \in \Sigma$ such that $\mathbb{P}^{\sigma}_{M,s}\big[\mathsf{DS}^{\lambda}_{1} \geq t\big] = 1 \quad \wedge \quad \mathbb{P}^{\sigma}_{M,s}\big[\mathsf{DS}^{\lambda}_{2} \geq -t\big] = 1$. By definition of the weight function, the second term of the conjunction is equivalent to $\mathbb{P}^{\sigma}_{M,s}\big[\mathsf{DS}^{\lambda}_{1} \leq t\big] = 1$. Hence, if a satisfying strategy σ exists, it does satisfy $\mathbb{P}^{\sigma}_{M,s}\big[\mathsf{DS}^{\lambda}_{1} = t\big] = 1$. We claim that the answer to the precise discounted sum problem is Yes if and only if the answer to the percentile problem is Yes.

First, assume a satisfying strategy σ exists. In general, our percentile problems do not require strategies to be pure. However, even if σ is randomized, we can extract a run ρ induced by this strategy and such that $\mathsf{DS}^{\lambda}_{1}(\rho) = t$ (such a run exists otherwise the strategy would not satisfy the percentile query). This run can be seen as a sequence of actions $\rho_A \in \{a, b\}^{\omega}$, which we translate in a corresponding sequence $\tau \in \{0, 1\}^{\omega}$ satisfying the precise discounted sum problem. Conversely, assume there exists a sequence $\tau \in \{0, 1\}^{\omega}$ satisfying the precise discounted sum problem. Then it defines a (possibly infinite-memory) pure strategy σ that ensures a discounted sum equal to t, and thus the percentile query is satisfied. □

This suggests that answering percentile problems for the discounted sum would require an important breakthrough.

7.2 Approximation algorithm

Approaching an answer As shown in Sect. 7.1, an exact algorithm is currently out of reach. Fortunately, we are still able to establish an algorithm that can "approximate" a solution. Since we consider decision problems, the notion of approximation should not be understood *sensu stricto*. We will formalize the output of the algorithm in the following but we first give an intuitive sketch.

The ε-gap problem Our algorithm takes as input a percentile query and an arbitrarily small *precision factor* $\varepsilon > 0$ and has three possible outputs: Yes, No and Unknown. If it answers Yes, then a satisfying strategy exists and can be synthesized. If it answers No, then no such strategy exists. Finally, the algorithm may output Unknown for a specified "zone" close to the threshold values involved in the problem and of width which depends on ε. It is possible to incrementally reduce the uncertainty zone, but it cannot be eliminated as the case $\varepsilon = 0$ would answer the precise discounted sum problem, which is not known to be decidable.

We actually solve an *ε-gap problem*, a particular case of *promise problems* [21], where the set of inputs is partitioned in three subsets: yes-inputs, no-inputs and the rest of them. The promise problem then asks to answer Yes for all yes-inputs and No for all no-inputs, while the answer may be arbitrary for the remaining inputs. In our setting, the set of inputs for which no guarantee is given can be taken arbitrarily small, parametrized by value $\varepsilon > 0$: this is an ε-gap problem. This notion is later formalized in Theorem 15.

Related work: single-constraint case There are papers considering models related to *single-constraint* percentile queries. Consider a single-dimensional MDP and a single-constraint query, with thresholds v and α. The *threshold problem* fixes v and maximizes α [36,37]. The *value-at-risk problem* fixes α and maximizes v [5]. This is similar to *quantiles* in the shortest path setting [34].

Paper [5] is the first to provide an exponential-time algorithm to approximate the optimal value v^* under a fixed α in the general setting. The authors also rely on approximation. While we do not consider optimization, we do extend the setting to *multi-constraint*, *multi-dimensional*, *multi-discount* problems, and we are able to remain in the same complexity class, namely EXPTIME.

Overview Our main contributions for the discounted sum are summarized in Theorem 14. In the following, we provide a thorough discussion for each of them, and prove several intermediate results of interest.

Theorem 14 *The ε-gap percentile problem for the discounted sum can be solved in time pseudo-polynomial in the model size and the precision factor, and exponential in the query size: polynomial in the number of states, the weights, the discount factors and the precision factor, and exponential in the number of constraints. It is* PSPACE*-hard for two-dimensional MDPs and already* NP*-hard for single-constraint queries. Exponential-memory strategies are both sufficient and in general necessary to satisfy ε-gap percentile queries.*

Cornerstones of the algorithm Our approach is similar to the shortest path: we want to build an unfolding capturing the needed information w.r.t. the discounted sums, and then reduce the percentile problem to a multiple reachability problem over this unfolding. However, several challenges have to be overcome.

First, we need a *finite* unfolding. This was easy in the shortest path due to non-decreasing sums and corresponding upper bounds. Here, it is not the case as we put no restriction on weights. Nonetheless, thanks to the discount factor, weights contribute less and less to the sum along a run. In particular, cutting all runs after a pseudo-polynomial length changes the overall sum by at most $\varepsilon/2$.

Second, we reduce the overall size of the unfolding. For the shortest path we took advantage of integer labels to define equivalence. Here, the space of values taken by the discounted sums is too large for a straightforward equivalence. To reduce it, we introduce a *rounding* scheme of the numbers involved. This idea is inspired by [5]. We bound the error due to cumulated roundings by $\varepsilon/2$.

So, we control the amount of information lost to guarantee exact answers except inside an arbitrarily small ε-zone. Given a q-constraint query $\mathcal{Q}$ for thresholds v_i, α_i, dimensions l_i and discounts λ_i, we define the *x-shifted query $\mathcal{Q}_x$*, for $x \in \mathbb{Q}$, as the exact same problem for thresholds $v_i + x, \alpha_i$, dimensions l_i and discounts λ_i. Our algorithm satisfies the following theorem, which formalizes the ε-gap percentile problem mentioned in Theorem 14.

Theorem 15 *There is an algorithm that, given an MDP, a percentile query $\mathcal{Q}$ for the discounted sum and a precision factor $\varepsilon>0$, solves the following ε-gap problem in exponential time. It answers*

- *Yes if there is a strategy satisfying the $(2 \cdot \varepsilon)$-shifted percentile query $\mathcal{Q}_{2\cdot\varepsilon}$;*
- *No if there is no strategy satisfying the $(-2 \cdot \varepsilon)$-shifted percentile query $\mathcal{Q}_{-2\cdot\varepsilon}$;*
- *and arbitrarily otherwise.*

We first state a more precise result and proceed with the technical discussion of the algorithm in the following paragraphs.

Theorem 16 *There is an algorithm satisfying the following properties.*

1. *It takes as input an MDP, a percentile query $\mathcal{Q}$ for the discounted sum and a precision factor $\varepsilon>0$.*
2. *If it outputs Yes, then there exists a strategy satisfying the percentile query $\mathcal{Q}$.*
3. *If it outputs No, then there exists no such strategy.*
4. *If it outputs Unknown, then there exists a strategy satisfying at least the $(-2 \cdot \varepsilon)$-shifted percentile query $\mathcal{Q}_{-2\cdot\varepsilon}$ and there exists no strategy satisfying the $(2\cdot\varepsilon)$-shifted percentile query $\mathcal{Q}_{2\cdot\varepsilon}$.*
5. *It runs in time polynomial in the size of the MDP, the weights, the discount factors and the precision factor, and exponential in the number of constraints.*

It suffices to prove Theorem 16 for Theorem 15 to follow as an immediate corollary for the ε-gap formulation of the problem.

Technical discussion Let $M = (S, A, \delta, w)$ be a d-dimensional MDP. We consider the q-constraint percentile query $\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M,s_{\mathsf{init}}}\left[\mathsf{DS}^{\lambda_i}_{l_i} \geq v_i\right] \geq \alpha_i$, where for $i \in \{1, \ldots, q\}$, we have that $v_i \in \mathbb{Q}$, $\alpha_i \in [0, 1] \cap \mathbb{Q}$, $\lambda_i \in]0, 1[\cap \mathbb{Q}$ and $l_i \in \{1, \ldots, d\}$. Let ε be an arbitrarily small precision factor. We assume w.l.o.g. that $\varepsilon \in \mathbb{Q}_0$, i.e., we always use rational precision factors.

We now describe the algorithm and establish intermediate results related to the construction operated by the algorithm. We conclude by proving that all properties stated in Theorem 16 are satisfied.

Our first step is building an unfolding of M, in the classical way. We denote it by U. Each node of U is labeled by the corresponding state of M and the discounted sum *related to each query*, computed over the descending path from the root to the node. Observe that we have q numerical dimensions in U and not d as in the shortest path. This will prove useful because we may have different discount factors for each constraint, hence the same dimension may induce different discounted sums depending on the considered constraint. This building scheme induces an infinite tree U with nodes labeled by elements of $S \times \mathbb{Q}^q$.

In order to obtain a finite tree, we compute a bound h on the height such that we do not lose too much information by cutting all branches at level h (assuming the root node is at level 1). Let U_h denote the cut of U at level h.

Lemma 18 *There exists a pseudo-polynomial height h such that for any infinite branch of U, its discounted sum on any dimension and w.r.t. any of the discount factors is at most $\varepsilon/2$ far from the discounted sum of its prefix branch in U_h.*

Proof Consider any branch of U_h, for some $h \in \mathbb{N}_0$. We denote the corresponding prefix of a run by $\pi = s_1 a_1 s_2 a_2 \dots a_{h-1} s_h$. Its discounted sum w.r.t. discount factor λ_i and dimension l_i is $\mathsf{DS}_{l_i}^{\lambda_i}(\pi) = \sum_{j=1}^{h-1} \lambda_i^j \cdot w_{l_i}(a_j)$. This branch could be extended in U to any infinite branch that represents a prolonging run $\rho = s_1 a_1 \dots a_{h-1} s_h a_{h+1} s_{h+1} \dots$ of which π is a prefix. We have that $\mathsf{DS}_{l_i}^{\lambda_i}(\rho) = \sum_{j=1}^{\infty} \lambda_i^j \cdot w_{l_i}(a_j)$ and we want to pick h such that

$$\left|\mathsf{DS}_{l_i}^{\lambda_i}(\rho) - \mathsf{DS}_{l_i}^{\lambda_i}(\pi)\right| \leq \frac{\varepsilon}{2},$$

for any prolonging run ρ. That is, we want

$$\left|\sum_{j=h}^{\infty} \lambda_i^j \cdot w_{l_i}(a_j)\right| \leq \frac{\varepsilon}{2}.$$

Let $\lambda = \max_i \lambda_i$ be the largest discount factor (i.e., the one for which the discounting effect if the slowest) and let W be the largest absolute weight appearing in the MDP M. We obtain that

$$\left|\sum_{j=h}^{\infty} \lambda_i^j \cdot w_{l_i}(a_j)\right| \leq W \cdot \sum_{j=h}^{\infty} \lambda^j = W \cdot \left(\sum_{j=0}^{\infty} \lambda^j - \sum_{j=0}^{h-1} \lambda^j\right) = W \cdot \frac{\lambda^h}{1-\lambda}.$$

It thus suffices to take h large enough to have that $W \cdot \frac{\lambda^h}{1-\lambda} \leq \frac{\varepsilon}{2}$. We assume that $W>0$ otherwise the discounted sum is always zero and the percentile problem is trivial. We also recall that $0<\lambda<1$. Hence the inequality becomes $\lambda^h \leq \frac{\varepsilon\cdot(1-\lambda)}{2\cdot W}$. Applying the binary logarithm, we get the following inequality:

$$h \cdot \log_2(\lambda) \leq \log_2(\varepsilon) + \log_2(1-\lambda) - \log_2(W) - 1.$$

Since $\lambda<1$, we have that $\log_2(\lambda)<0$ and we finally obtain that

$$h \geq \frac{\log_2(\varepsilon) + \log_2(1-\lambda) - \log_2(W) - 1}{\log_2(\lambda)}.$$

Observe that this expression is always positive as $\varepsilon<1$, $\lambda<1$ and $W \geq 1$. In the following, let us assume we take the ceiling of this expression as the value h. What is the size of h w.r.t. the input of the algorithm? Since we are taking the binary logarithm of all involved values, it may seem that h only needs to be polynomial in the encoding of the values. However, when $\lambda \sim 1$, we have that $\log_2 \lambda \sim 1 - \lambda$. Therefore, h can be polynomial in the value of λ, that is, exponential in its encoding. ◻

We now have a finite tree U_h, of pseudo-polynomial height, and such that all discounted sums labeled in its leaves are at most $\varepsilon/2$ far from the one of any prolonging run. In other words, once such a leaf has been reached, the controller may use any arbitrary strategy and its discounted sum will not vary by more than $\varepsilon/2$. This implies that we only care about devising a strategy for the h first steps, as we will use later.

Consider the overall size of the tree U_h. As discussed for the shortest path, this size can be as high as $\mathcal{O}(b^h)$, where b denotes the branching degree of M, defined as $b = \max_{s \in S} \left|\{(a, s') \mid a \in A(s), s' \in S, \delta(s, a, s') > 0\}\right|$. Thus, the overall size could be pseudo-exponential. Again, we want to reduce this tree U_h to a compressed tree of truly-exponential size by merging equivalent nodes.

However, in this case it does not suffice to look for nodes with the exact same labels. Indeed, the range of possible labels is in general pseudo-exponential. Observe that the set of labels of any tree U_h is a finite subset of $S \times [-W \cdot h, W \cdot h]^q$ (this characterization can be narrowed but it suffices for our needs). We introduce a value $\gamma \in \mathbb{Q}$ and maps the set of possible labels to $S \times \{-W \cdot h, -W \cdot h + \gamma, -W \cdot h + 2 \cdot \gamma, \ldots, W \cdot h - \gamma, W \cdot h\}^q$ by rounding the values appearing in U_h to multiples of γ (we assume w.l.o.g. that $W \cdot h$ is such a multiple). To that end, we define the function $\mathsf{Round}_\gamma : \mathbb{Q} \rightarrow \mathbb{Q}$ that rounds any rational $x \in \mathbb{Q}$ to the closest multiple of γ, i.e., the closest value in the new set of labels. The idea of rounding numbers to reduce the complexity is inspired by [5], but the technique differs.

Assume we apply this label mapping on U_h, for some fixed γ. Then, we define $U_{h,\sim_\gamma}$ as the MDP obtained by merging nodes having identical labels after the mapping. This is the unfolded MDP we are looking for *if γ is chosen adequately*, and it can be built on the fly by rounding each node (and potentially merging) at the moment it is created. Intuitively, γ should not be too large to be able to keep the resulting rounding error low, but it should be large enough to induce a range of labels which is at most of exponential size. The following lemma states the existence of such a value $\gamma \in \mathbb{Q}$.

Lemma 19 *There exists a value $\gamma \in \mathbb{Q}$ such that*

1. $\left|S \times \{-W \cdot h, -W \cdot h + \gamma, \ldots, W \cdot h - \gamma, W \cdot h\}^q\right|$ *is at most exponential;*
2. *For all branch π in U_h, for all λ_i, l_i, $i \in \{1, \ldots, q\}$, we have that*

$$\left|\mathsf{DS}_{l_i}^{\lambda_i}(\pi) - \mathsf{RDS}_{l_i}^{\lambda_i}(\pi)\right| \leq \frac{\varepsilon}{2},$$

where $\mathsf{RDS}_{l_i}^{\lambda_i}(\pi)$ denotes the rounded discounted sum of the corresponding branch π' in $U_{h,\sim_\gamma}$ (i.e., the label of the corresponding leaf in $U_{h,\sim_\gamma}$).

Proof We choose $\gamma = \dfrac{\varepsilon}{h-1}$ and prove the two assumptions. Observe that we assume $h > 1$ otherwise U_h contains only the root node with all discounted sums equal to zero and no rounding is needed.

First, consider *Assumption 1*. The size of the set is $|S| \cdot \left(\dfrac{2 \cdot W \cdot h + 1}{\gamma}\right)^q$. Hence it suffices to prove that $(2 \cdot W \cdot h + 1) \cdot \gamma^{-1}$ is at most exponential. Since both h and W are at most exponential (in the encoding of values), this boils down to proving that $\gamma^{-1} = \dfrac{h-1}{\varepsilon}$ is at most exponential, which is the case.

Second, let us prove *Assumption 2*. Recall that our rounding scheme maps each value to the closest multiple of γ whenever the label of a node is computed. It is important to understand that this rounding is executed on the fly, and not after building the tree U_h fully (otherwise we would require pseudo-exponential time). Consequently, when a discounted sum for a node of level $2 \leq n \leq h$ is computed, we have to take into account that the label of its father of level $n-1$ has already been rounded: the rounding errors add up along a branch.

We claim that the total error over a branch of height h is bounded by the expression $(h-1) \cdot \frac{\gamma}{2}$. That is, for all height-h branch π of U_h, for all λ_i, l_i,

$$\left|\mathsf{DS}_{l_i}^{\lambda_i}(\pi) - \mathsf{RDS}_{l_i}^{\lambda_i}(\pi)\right| \leq (h-1) \cdot \frac{\gamma}{2}.$$

We prove it by induction. Let $\pi = s_1 a_1 s_2 \ldots s_h$ in the following.

The base case is $h = 2$. We ask whether

$$\left|\lambda_i \cdot w_{l_i}(a_1) - \mathsf{Round}_\gamma\big(\lambda_i \cdot w_{l_i}(a_1)\big)\right| \leq \frac{\gamma}{2}.$$

This is clearly true by definition of Round_γ, which maps any rational to the closest multiple of γ.

Now assume our claim is true up to level $2 \leq h-1$. We prove it is still satisfied for level h. Let us rewrite $\left|\mathsf{DS}_{l_i}^{\lambda_i}(\pi) - \mathsf{RDS}_{l_i}^{\lambda_i}(\pi)\right|$ as follows:

$$\begin{aligned}\Big|&\mathsf{DS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1}) + \lambda_i^{h-1} \cdot w_{l_i}(a_{h-1})\\ &- \mathsf{Round}_\gamma\left(\mathsf{RDS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1}) + \lambda_i^{h-1} \cdot w_{l_i}(a_{h-1})\right)\Big|.\end{aligned}$$

Using the equality $\mathsf{Round}_\gamma(n \cdot \gamma + x) = n \cdot \gamma + \mathsf{Round}_\gamma(x)$ for $n \in \mathbb{N}$ and $x \in \mathbb{Q}$, along with the fact that $\mathsf{RDS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1})$ is already rounded by construction, we rewrite this as:

$$\begin{aligned}\Big|&\mathsf{DS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1}) + \lambda_i^{h-1} \cdot w_{l_i}(a_{h-1}) - \mathsf{RDS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1})\\ &- \mathsf{Round}_\gamma\left(\lambda_i^{h-1} \cdot w_{l_i}(a_{h-1})\right)\Big|.\end{aligned}$$

By the subadditivity of $|\cdot|$, we bound this expression by

$$\begin{aligned}&\left|\mathsf{DS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1}) - \mathsf{RDS}_{l_i}^{\lambda_i}(s_1 \ldots s_{h-1})\right|\\ &+ \left|\lambda_i^{h-1} \cdot w_{l_i}(a_{h-1}) - \mathsf{Round}_\gamma\left(\lambda_i^{h-1} \cdot w_{l_i}(a_{h-1})\right)\right|.\end{aligned}$$

Finally, using the induction hypothesis for the first term and the definition of Round_γ for the second one, we can bound this sum by

$$(h-2) \cdot \frac{\gamma}{2} + \frac{\gamma}{2} = (h-1) \cdot \frac{\gamma}{2},$$

which proves our initial claim.

Now, by our choice of γ, this implies that the total rounding error over any branch is at most $\frac{\varepsilon}{2}$, which proves the correctness of *Assumption 2*. □

Let us sum up the situation: given an MDP, a percentile query and a precision factor $\varepsilon > 0$, we are able to construct an unfolded MDP $U_{h,\sim_\gamma}$ of at most exponential size such that all leaves have labels in $S \times \{-W \cdot h, -W \cdot h + \gamma, \ldots, W \cdot h - \gamma, W \cdot h\}^q$, where each of the q numerical dimensions approximate the discounted sum of corresponding infinite branches within an error bounded by ε ($\varepsilon/2$ due to truncating the branches and $\varepsilon/2$ due to the rounding of values).

The last step of our algorithm is as follows. Consider the $2 \cdot q$ following target sets of nodes in $U_{h,\sim_\gamma}$.

- $\forall\, i \in \{1, \dots, q\}$, Sure_i is the set of leaves for which the label on numerical dimension i is greater than or equal to $v_i + \varepsilon$. Essentially, we have that $\mathsf{RDS}^{\lambda_i}_{l_i}(\pi) \geq v_i + \varepsilon$, where π denotes a corresponding descending branch.
- $\forall\, i \in \{1, \dots, q\}$, Maybe_i is the set of leaves for which the label on numerical dimension i is greater than or equal to $v_i - \varepsilon$. Essentially, we have that $\mathsf{RDS}^{\lambda_i}_{l_i}(\pi) \geq v_i - \varepsilon$, where π denotes a corresponding descending branch.

Observe that $\mathsf{Sure}_i \subseteq \mathsf{Maybe}_i$ for all query i. Our algorithm proceeds as follows.

(A) We execute the multiple reachability problem checking the existence of a strategy σ' such that

$$\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{U_{h,\sim_\gamma}, s'_{\mathsf{init}}}\big[\Diamond \mathsf{Sure}_i\big] \geq \alpha_i,$$

with s'_{init} the root node of the unfolded MDP. If the answer is Yes, then we answer Yes to the percentile problem. Otherwise, we proceed to the next step.

(B) We execute the multiple reachability problem checking the existence of a strategy σ' such that

$$\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{U_{h,\sim_\gamma}, s'_{\mathsf{init}}}\big[\Diamond \mathsf{Maybe}_i\big] \geq \alpha_i,$$

with s'_{init} the root node of the unfolded MDP. If the answer is Yes, then we answer $\mathsf{Unknown}$ to the percentile problem. Otherwise, we answer No.

The intuition is threefold. First, if a leaf of Sure_i is reached, then whatever the strategy that is played afterwards, any prolonging run will have a discounted sum at least equal to v_i w.r.t. the corresponding discount factor λ_i and dimension l_i. Hence, all prolonging runs are acceptable for constraint i. Second, if a leaf of Maybe_i is reached, then some prolonging runs may satisfy the constraint while other do not: we need to compute the unfolding for a smaller precision factor ε in order to obtain useful information from nodes that are currently in $\mathsf{Maybe}_i \setminus \mathsf{Sure}_i$. Third, if a leaf does not belong to Maybe_i, then any prolonging run is guaranteed to falsify constraint i as adding error ε does not suffice to make the discounted sum at least equal to v_i. We are finally able to prove Theorem 16.

Proof (Proof of Theorem 16) We consider each of *properties 2–5* separately.

Property 2 Our algorithm answers Yes if and only if there exists a strategy σ' satisfying the multiple reachability query $\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{U_{h,\sim_\gamma}, s'_{\mathsf{init}}}\big[\Diamond \mathsf{Sure}_i\big] \geq \alpha_i$. We define the strategy σ on the original MDP M that plays as follows: it chooses the $(h-1)$ first actions according to σ' and then plays an arbitrary memoryless strategy. By Lemma 18, Lemma 19, and by definition of Sure_i, this strategy guarantees that for all $i \in \{1, \dots, q\}$, a discounted sum (w.r.t. λ_i and l_i) at least equal to v_i is achieved with probability at least equal to α_i. Hence this finite-memory strategy σ satisfies the discounted sum percentile query on the original MDP M.

Property 3 Our algorithm answers No if and only if there exists no strategy σ' satisfying the multiple reachability query $\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{U_{h,\sim_\gamma}, s'_{\mathsf{init}}}\big[\Diamond \mathsf{Maybe}_i\big] \geq \alpha_i$. By contradiction, assume the multiple reachability query cannot be satisfied, yet there exists a strategy σ in the original MDP M that satisfies the percentile query for the discounted sum. That is, for all i and associated λ_i, l_i, this strategy achieves discounted sum at least v_i with probability at least α_i. By Lemmas 18 and 19, we know that such a strategy reaches with probability at least α_i leaves in $U_{h,\sim_\gamma}$ that are labeled with a value at least equal to $v_i - \varepsilon$ in numerical dimension

i. That is, σ reaches each set Maybe_i with probability at least α_i, which contradicts the hypothesis and proves the property.

Property 4 Applying the same argument as for *property 1*, if there exists a strategy σ' for the multiple reachability query $\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{U_{h,\sim_\gamma}, s'_{\mathsf{init}}}[\Diamond \mathsf{Maybe}_i] \geq \alpha_i$, then this strategy can be translated into a strategy σ over M that ensures the percentile query where all value thresholds v_i are replaced by their shifted version $v_i - 2 \cdot \varepsilon$. Indeed, observe that the threshold gap between sets Maybe_i and Sure_i is exactly $2 \cdot \varepsilon$. Conversely, we apply the argument of *property 2* to deduce that if there exists no strategy for the multiple reachability query $\bigwedge_{i=1}^{q} \mathbb{P}^{\sigma'}_{U_{h,\sim_\gamma}, s'_{\mathsf{init}}}[\Diamond \mathsf{Sure}_i] \geq \alpha_i$ (which is the case otherwise the answer of the algorithm would have been Yes), then there is no strategy for the percentile query shifted by $2 \cdot \varepsilon$.

Property 5. It remains to study the complexity of our algorithm. Recall that the unfolded MDP $U_{h,\sim_\gamma}$ can be constructed in time

$$\mathcal{O}\left(|S| \cdot \left(\frac{2 \cdot W \cdot h + 1}{\gamma}\right)^q\right),$$

while h is polynomial in $\lambda = \max_i \lambda_i$, $\log_2(\varepsilon)$ and $\log_2(W)$ and γ is polynomial in both ε and h. Moreover, multiple reachability queries executed by the algorithm only require polynomial time in $|U_{h,\sim_\gamma}|$ as all target states are absorbing (they are leaves in the unfolding). Overall, this shows that our algorithm requires time that is polynomial in $|S|$, W, λ and ε, and exponential in q. This proves the property and finally concludes our proof of correctness for the algorithm. □

Lower bounds The ε-gap percentile problem is PSPACE-hard by reduction from subset-sum games [33]. Those are two-player games defined by a finite list of pairs of natural numbers $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$, and a target $t \in \mathbb{N}$. Players take turns choosing between a_j and b_j. After n rounds, if the sum of the chosen numbers equals t, then player 1 wins, otherwise player 2 wins. Deciding if player 1 has a winning strategy in a subset-sum game is PSPACE-complete [33].

Lemma 20 *The ε-gap problem defined in Theorem 15 is* PSPACE*-hard, already for two-dimensional MDPs and fixed values of discount and precision factors.*

Two tricks are important. First, counterbalancing the discount effect via adequate weights. Second, simulating an equality constraint. This cannot be achieved directly because it requires to handle $\varepsilon = 0$. Still, by choosing weights carefully we restrict possible discounted sums to integer values only. Then we choose the thresholds and $\varepsilon > 0$ such that no run can take a value within the uncertainty zone. This circumvents the limitation due to uncertainty.

Proof Consider a subset-sum game defined by pairs $(a_1, b_1), \ldots, (a_n, b_n) \in \mathbb{N}^2$, and target $t \in \mathbb{N}$. Assume that we have an algorithm, called $\mathsf{Algo}_\varepsilon$, that solves the ε-gap problem of Theorem 15. We claim that this algorithm can also decide if player 1 has a winning strategy in the subset-sum game, through a polynomial-time reduction of the subset-sum game to a discounted sum percentile problem.

We construct a 2-dimensional MDP $M = (S, A, \delta, w)$. Our construction is illustrated in Fig. 7. Filled circles represent equiprobable stochastic transitions. Controllable states simulate choices of player 1 in the game: the controller can choose between a_j and b_j when j is odd. Conversely, stochastic transitions simulate choices of player 2: when j is even, a_j and b_j are chosen with the same probability $1/2$. Each action corresponding to choosing a_j (resp. b_j) has a 2-dimensional weight $(\lambda^{-j} \cdot a_j, -\lambda^{-j} \cdot a_j)$ (resp. $(\lambda^{-j} \cdot b_j, -\lambda^{-j} \cdot b_j)$).

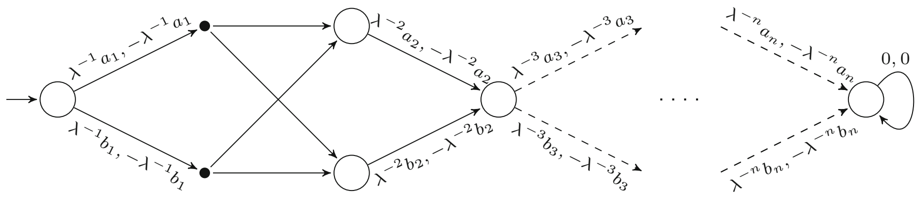

$\lambda^{-1}a_1, -\lambda^{-1}a_1$
$\lambda^{-1}b_1, -\lambda^{-1}b_1$
$\lambda^{-2}a_2, -\lambda^{-2}a_2$
$\lambda^{-2}b_2, -\lambda^{-2}b_2$
$\lambda^{-3}a_3, -\lambda^{-3}a_3$
$\lambda^{-3}b_3, -\lambda^{-3}b_3$
$\lambda^{-n}a_n, -\lambda^{-n}a_n$
$\lambda^{-n}b_n, -\lambda^{-n}b_n$
0, 0

Fig. 7 Encoding of subset-sum game into 2-dimensional percentile problem for the discounted sum

The discount factor can be fixed arbitrarily, say $\lambda = 1/2$ for the sake of concreteness. Note that those weights only require an encoding which is polynomial in the size of the input. We add a self-loop with weight (0, 0) on the terminal state.

Observe that any run in this MDP has a discounted sum which is exactly equal to the sum of the chosen elements a_j, b_j, thanks to the countereffect of λ^{-j} in the weights definition. Hence we also have that all runs have integer discounted sums.

Our goal is to find a 2-dimensional percentile query that can express the winning condition of the subset-sum game, taking into account that algorithm $\mathsf{Algo}_\varepsilon$ can only solve the ε-gap problem.

Intuitively, we would like to express that the discounted sum must be exactly equal to t, in all possible runs. First observe that given the structure of the MDP, the terminal state and its zero loop is guaranteed to be reached in n steps. Therefore, any strategy ensuring the required property almost-surely (i.e., with probability one) also ensures it surely (i.e., over all possible runs). Ideally, we would like to execute the 2-constraint percentile problem asking for the existence of a strategy that satisfies query

$$\mathcal{Q}^A := \ \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_1 \geq t\big] = 1 \quad \wedge \quad \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_2 \geq -t\big] = 1.$$

Let us call it *Problem A*. Any strategy satisfying $\mathcal{Q}^A$ would be a winning strategy for player-1, and conversely. Still, this would only be useful if we could take $\varepsilon = 0$, which we cannot.

Instead, consider *Problem B*, asking for the existence of a strategy satisfying

$$\mathcal{Q}^B := \ \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_1 \geq t - 1/2\big] = 1 \quad \wedge \quad \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_2 \geq -t - 1/2\big] = 1.$$

Furthermore, let us choose the precision factor $\varepsilon = 1/6$. Recall we assume that $\mathsf{Algo}_\varepsilon$ solves the ε-gap problem. Consider the execution of $\mathsf{Algo}_\varepsilon$ over query $\mathcal{Q}^B$. By definition of the ε-gap problem (Theorem 15), we have that:

(1) if there exists a strategy σ satisfying

$$\mathcal{Q}^B_{2\cdot\varepsilon} := \ \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_1 \geq t - 1/6\big] = 1 \quad \wedge \quad \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_2 \geq -t - 1/6\big] = 1,$$

then the answer of $\mathsf{Algo}_\varepsilon$ is Yes;

(2) if there exists no strategy σ satisfying

$$\mathcal{Q}^B_{-2\cdot\varepsilon} := \ \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_1 \geq t - 5/6\big] = 1 \quad \wedge \quad \mathbb{P}^\sigma_{M,s}\big[\mathsf{DS}^\lambda_2 \geq -t - 5/6\big] = 1,$$

then the answer of $\mathsf{Algo}_\varepsilon$ is No;

(3) otherwise the answer can be either Yes or No.

Now let us review the possible answers of $\mathsf{Algo}_\varepsilon$.

Assume the answer is Yes. By (2), we have that there exists a strategy σ that satisfies $\mathcal{Q}^B_{-2\cdot\varepsilon}$ otherwise the answer would have been No. Since all runs have integer discounted

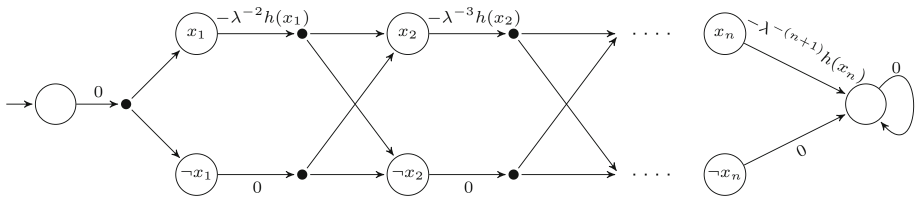


Fig. 8 Reduction from K-th largest subset problem to ε-gap problem for a single-constraint discounted sum percentile problem over a Markov chain

sums, this necessarily implies that σ also satisfies $\mathcal{Q}^A$. Indeed, we have that $t = \lceil t - 5/6 \rceil$ and $-t = \lceil -t - 5/6 \rceil$. Hence player-1 has a winning strategy in the subset-sum game.

Assume the answer is No. By (1), we have that there exists no strategy σ that satisfies $\mathcal{Q}^B_{2\cdot\varepsilon}$ otherwise the answer would have been Yes. Obviously, there exists no more strategy satisfying $\mathcal{Q}^A$ since it is harder to satisfy (its thresholds are higher). Hence player-1 has no winning strategy in the subset-sum game.

Finally, we see that the answer of $\mathsf{Algo}_\varepsilon$ is Yes if and only if the answer to *Problem A* is also Yes. Since algorithm $\mathsf{Algo}_\varepsilon$ can decide *Problem A*, we also have that it can decide if player-1 has a winning strategy in the subset-sum game, which concludes our proof. □

For single-constraint ε-gap problems, we prove NP-hardness, even for Markov chains. Our proof is by reduction from the K-th largest subset problem [20], inspired by [7, Theorem 11]. A recent paper by Haase and Kiefer [23] shows that this K-th largest subset problem is actually PP-complete. This suggests that the single-constraint problem does not belong to NP at all, otherwise the polynomial hierarchy would collapse to P^{NP} by Toda's theorem [31].

Lemma 21 *The ε-gap problem defined in Theorem 15 is* NP*-hard for single-constraint queries. This holds even for Markov chains, i.e., MDPs with only one available action in every state.*

Proof The K-th largest subset problem is as follows. Given a finite set $X = \{x_1, \ldots, x_n\}$ (hence $n = |X|$), a size function $h : X \to \mathbb{N}$ assigning non-negative integer values to elements of X, and two naturals $K, L \in \mathbb{N}$, decide if there exist K distinct subsets $Y_i \subseteq X$, $1 \leq i \leq K$, such that $h(Y_i) = \sum_{x \in Y_i} h(x) \leq L$ for all K subsets. The NP-hardness of this problem was proved in [24].

We assume w.l.o.g. that $K \leq 2^n$ otherwise the answer to the problem is trivially No since we cannot find a sufficient number of *distinct* subsets.

Given an instance of the K-th largest subset problem, we build a Markov chain as depicted in Fig. 8. Observe that this is indeed a Markov *chain* as there is a unique action available in all states. As usual, the filled circles represent equiprobable transitions. In the first step, element x_1 is either selected (upper transition) or not selected (lower one), with equal probability. This is repeated for every element up to reaching the terminal state with a zero loop. Hence, there is a bijection between runs in this Markov chain and subsets of X. Moreover, all subsets are equiprobable: they have probability $1/2^n$ to be selected.

The discount factor can be chosen arbitrarily. For the sake of concreteness, assume it is $\lambda = 1/2$. Now, observe that the weight function is defined such that the discounted sum over a run representing a subset $Y \subseteq X$ is exactly equal to $-h(Y) = -\sum_{x \in Y} h(x)$. To achieve this, we again use the trick of multiplying values $-h(x_i)$ by $\lambda^{-(i+1)}$ (the shift is due to the first transition). By definition of our weight function, it is clear that all runs take integer values. Also, the size of the Markov chain is polynomial in the size of the original problem.

Consider the single-constraint percentile query asking if

$$\mathbb{P}_{M,s}\big[\mathsf{DS}^{\lambda} \geq -L - 1/2\big] \geq \frac{K}{2^n},$$

with s the initial state of the Markov chain. Note that we drop the existential quantification on strategies since there exists a unique - and trivial - strategy in a Markov chain. Recall that we only have access to an algorithm, say $\mathsf{Algo}_{\varepsilon}$, that solves the ε-gap problem, not the exact one. Consider $\varepsilon = 1/6$ and let us review the possible answers given by the execution of $\mathsf{Algo}_{\varepsilon}$ on this query.

Assume $\mathsf{Algo}_{\varepsilon}$ answers Yes. By definition of the ε-gap problem (Theorem 15), we have that

$$\mathbb{P}_{M,s}\big[\mathsf{DS}^{\lambda} \geq -L - 5/6\big] \geq \frac{K}{2^n} \quad \Rightarrow \quad \mathbb{P}_{M,s}\big[\mathsf{DS}^{\lambda} \geq -L\big] \geq \frac{K}{2^n}.$$

The implication follows from the fact that all runs take integer values and by equality $\lceil -L - 5/6 \rceil = -L$ since $L \in \mathbb{N}$. This implies that there are at least K distinct runs representing subsets $Y_i \subseteq X$ for which $-h(Y_i) \geq -L \Leftrightarrow h(Y_i) \leq L$. Hence the answer to the K-th largest subset problem is also Yes.

Now assume $\mathsf{Algo}_{\varepsilon}$ answers No. By definition of the ε-gap problem, we have that

$$\mathbb{P}_{M,s}\big[\mathsf{DS}^{\lambda} \geq -L - 1/6\big] < \frac{K}{2^n} \quad \Rightarrow \quad \mathbb{P}_{M,s}\big[\mathsf{DS}^{\lambda} \geq -L\big] < \frac{K}{2^n},$$

using the fact that the second inequality is harder to satisfy. This implies that there are strictly $<K$ distinct runs representing subsets $Y_i \subseteq X$ for which $-h(Y_i) \geq -L \Leftrightarrow h(Y_i) \leq L$. Hence the answer to the K-th largest subset problem is also No.

In summary, we have that $\mathsf{Algo}_{\varepsilon}$ answers Yes if and only if the answer to the K-th largest subset problem is also Yes. This concludes our proof. □

Memory For the precise discounted sum and generalizations, infinite memory is needed [11]. For ε-gap problems, the exponential upper bound follows from the algorithm while the lower bound is shown via a family of problems that emulate the ones used for multiple reachability (Theorem 2).

Lemma 22 *Exponential-memory strategies are both sufficient and, in general, necessary to satisfy ε-gap percentile problems for the discounted sum.*

Proof First, the algorithm of Theorem 16 solves the ε-gap percentile problem by answering a multiple reachability problem over an unfolded MDP of exponential size. As stated in Theorem 1, memory of size polynomial in the MDP (here, the unfolded one) and exponential in the number of contraints (which is untouched by our algorithm) is sufficient to satisfy such queries. Moreover, once the first h steps have been played according to such a strategy, any arbitrary strategy may be used, in particular a memoryless one suffices. Hence, it follows that exponential-memory strategies suffice for the discounted sum ε-gap percentile problem.

Second, for the lower bound we use a family of MDPs based on the one defined to prove the exponential memory requirements of multiple reachability problems (Lemma 3). Consider the unweighted MDP depicted in Fig. 2. Recall it is composed of k stochastic gadgets followed by k controllable gadgets. We transform this MDP into a $2 \cdot k$-dimensional MDP M as follows. First, we remove the self-loops on states $s'_{k,L}$ and $s'_{k,R}$ and replace them by actions going to a terminal state s_t with probability one: this is for technical convenience. Second, we associate actions to $2 \cdot k$-dimensional weight vectors:

– the action leaving s_1 has weight $-\lambda^{-1}$ in all $2 \cdot k$ dimensions,
– actions leaving a state $s_{i,L}$ have weight $\lambda^{-2 \cdot i}$ in dimension i and weight zero in all other dimensions,
– actions leaving a state $s_{i,R}$ have weight $\lambda^{-2 \cdot i}$ in dimension $k + i$ and weight zero in all other dimensions,
– actions leaving a state $s'_{i,L}$ have weight $\lambda^{-(k+2 \cdot i)}$ in dimension i and weight zero in all other dimensions,
– actions leaving a state $s'_{i,R}$ have weight $\lambda^{-(k+2 \cdot i)}$ in dimension $k + i$ and weight zero in all other dimensions,
– all remaining actions have weight zero in all dimensions.

As usual, the discount factor can be taken equal to $1/2$. While this may seem technical, the goal is simple: emulating the multiple reachability problem used in Lemma 3. Each dimension $l \in \{1, \ldots, 2 \cdot k\}$ will get a -1 by the first action. Then, a dimension $l \in \{1, \ldots, k\}$ (resp. $l \in \{k + 1, \ldots, 2 \cdot k\}$) will get a 1 when $s_{l,L}$ or $s'_{l,L}$ (resp. when $s_{l-k,R}$ or $s'_{l-k,R}$) is visited. All other actions have no impact on the discounted sum over dimension l. Therefore, one can easily check if a run ρ has visited a set $\{s_{i,L}, s'_{i,L}\}$ (resp. $\{s_{i,R}, s'_{i,R}\}$): it suffices to check if the discounted sum on dimension i (resp. $k + i$) is at least zero.

Now consider the percentile query

$$\mathcal{Q} := \bigwedge_{l=1}^{2 \cdot k} \mathbb{P}^{\sigma}_{M,s_1}\left[\mathsf{DS}^{\lambda}_{l} \geq -1/2\right] = 1,$$

and in particular, its ε-gap version, with $\varepsilon = 1/6$. Applying the same reasoning as for proofs of Lemmas 20 and 21, we can prove that the answer to this ε-gap problem is Yes if and only if all target sets

$$T_l = \{s_{1,L}, s'_{1,L}\}, \{s_{1,R}, s'_{1,R}\}, \{s_{2,L}, s'_{2,L}\}, \ldots, \{s_{k,L}, s'_{k,L}\}, \{s_{k,R}, s'_{k,R}\}$$

are reached almost-surely. By Lemma 3, we know that this requires a strategy encoded as a Moore machine with no $<2^k$ memory states. This shows the exponential lower bound for the ε-gap problem and concludes our proof. □

8 Conclusion

Through this paper, we studied the strategy synthesis problem for *multi-percentile queries* on multi-dimensional MDPs: we considered a wide range of payoff functions from the literature (sup, inf, limsup, liminf, mean-payoff, truncated sum, discounted sum), and established a complete picture of the multi-percentile framework, including algorithms, lower bounds on complexity, and memory requirements. Our results are summed up in Table 1.

It is especially interesting to observe that for all payoff functions but the discounted sum, our algorithms require *polynomial time in the size of the model* when the query size is fixed. This is of utmost practical interest as in most applications, the query size (i.e., specification) is typically small while the model (i.e., the system) can be very large. Hence, our algorithms have clear potential to be useful in practice. As future work, we aim to assess their practical efficiency through implementation in tool suites and case studies.

References

1. Baier C, Daum M, Dubslaff C, Klein J, Klüppelholz S (2014) Energy-utility quantiles. In: NASA formal methods, LNCS 8430, Springer, pp 285–299
2. Bertsekas DP, Tsitsiklis JN (1991) An analysis of stochastic shortest path problems. Math Oper Res 16:580–595
3. Boker U, Henzinger TA (2014) Exact and approximate determinization of discounted-sum automata. Log Methods Comput Sci 10(1)
4. Boker U, Henzinger TA, Otop J (2015) The target discounted-sum problem. In: IEEE Proceedings of LICS, pp 750–761
5. Brázdil T, Chen T, Forejt V, Novotný P, Simaitis A (2013) Solvency Markov decision processes with interest. In: Proceedings of FSTTCS, volume 24 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, pp 487–499
6. Brázdil T, Brozek V, Chatterjee K, Forejt V, Kucera A (2014) Markov decision processes with multiple long-run average objectives. Log Methods Comput Sci 10(13):1–29
7. Bruyère V, Filiot E, Randour M, Raskin J-F (2014) Meet your expectations with guarantees: beyond worst-case synthesis in quantitative games. In: Proceedings of STACS, volume 25 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, pp 199–213
8. Chatterjee K (2007) Concurrent games with tail objectives. Theore Comput Sci 388(1 3):181–198
9. Chatterjee K, Doyen L, Henzinger TA, Raskin J-F (2010) Generalized mean-payoff and energy games. In: Proceedings of FSTTCS, volume 8 of LIPIcs, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp 505–516
10. Chatterjee K, Doyen L, Randour M, Raskin J-F (2015) Looking at mean-payoff and total-payoff through windows. Inf Comput 242:25–52
11. Chatterjee K, Forejt V, Wojtczak D (2013) Multi-objective discounted reward verification in graphs and MDPs. In: Proceedings of LPAR, LNCS 8312, Springer, pp 228–242
12. Chatterjee K, Henzinger TA (2009) Probabilistic systems with limsup and liminf objectives. In: Brattka MAV, Löwe VGB (eds), Infinity in logic and computation, LNCS 5489, Springer, pp 32–45
13. Chatterjee K, Komárková Z, Kretínský J (2015) Unifying two views on multiple mean-payoff objectives in Markov decision processes. In: Proceedings of LICS, pp 244–256
14. Chatterjee K, Majumdar R, Henzinger TA (2006) Markov decision processes with multiple objectives. In: Proceedings of STACS, LNCS 3884, Springer, pp 325–336
15. Chatterjee K, Randour M, Raskin J-F (2014) Strategy synthesis for multi-dimensional quantitative objectives. Acta Inform 51(3–4):129–163
16. de Alfaro L (1997) Formal verification of probabilistic systems. Ph.D. thesis, Stanford University
17. de Alfaro L (1999) Computing minimum and maximum reachability times in probabilistic systems. In: Proceedings of CONCUR, LNCS 1664, Springer, pp 66–81
18. Etessami K, Kwiatkowska M, Vardi MY, Yannakakis M (2008) Multi-objective model checking of Markov decision processes. Log Methods in Comput Sci 4(4)
19. Filar JA, Krass D, Ross KW (1995) Percentile performance criteria for limiting average Markov decision processes. IEEE Trans Autom Control 40(1):2–10
20. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, New York
21. Goldreich O (2006) On promise problems: a survey. In: Goldreich O, Rosenberg A, Selman AL (eds), Theoretical computer science, essays in memory of Shimon Even, LNCS 3895, Springer, pp 254–290
22. Haase C, Kiefer S (2015) The odds of staying on budget. In: Proceedings of ICALP, LNCS 9135, Springer, pp 234–246
23. Haase C, Kiefer S (2016) The complexity of the Kth largest subset problem and related problems. Inf Process Lett 116(2):111–115
24. Johnson DB, Kashdan SD (1978) Lower bounds for selection in X + Y and other multisets. J ACM 25(4):556–570
25. Minsky ML (1961) Recursive unsolvability of Post's problem of "tag" and other topics in theory of Turing machines. Ann Math 74(3):437–455
26. Ohtsubo Y (2004) Optimal threshold probability in undiscounted Markov decision processes with a target set. Appl Math Comput 149(2):519–532
27. Puterman ML (1994) Markov decision processes: discrete stochastic dynamic programming, 1st edn. Wiley, New York
28. Randour M, Raskin J-F, Sankur O (2015) Percentile queries in multi-dimensional Markov decision processes. In: Proceedings of CAV, LNCS 9206, Springer, pp 123–139

29. Randour M, Raskin J-F, Sankur O (2015) Variations on the stochastic shortest path problem. In: Proceedings of VMCAI, LNCS 8931, Springer, pp 1–18
30. Sakaguchi M, Ohtsubo Y (2013) Markov decision processes associated with two threshold probability criteria. J Contro Theory Appl 11(4):548–557
31. Toda S (1991) PP is as hard as the polynomial-time hierarchy. SIAM J Comput 20(5):865–877
32. Tracol M (2009) Fast convergence to state-action frequency polytopes for MDPs. Oper Res Lett 37(2):123–126
33. Travers SD (2006) The complexity of membership problems for circuits over sets of integers. Theor Comput Sci 369(1–3):211–229
34. Ummels M, Baier C (2013) Computing quantiles in Markov reward models. In: Proceedings of FOSSACS, LNCS 7794, Springer, pp 353–368
35. Vardi MY (1985) Automatic verification of probabilistic concurrent finite-state programs. In: IEEE Proceedings of FOCS, pp 327–338
36. White DJ (1993) Minimizing a threshold probability in discounted Markov decision processes. J Math Anal Appl 173(2):634–646
37. Congbin W, Lin Y (1999) Minimizing risk models in Markov decision processes with policies depending on target values. J Math Anal Appl 231(1):47–67
38. Xu H, Mannor S (2011) Probabilistic goal Markov decision processes. In: Proceedings of IJCAI, pp 2046–2052