

Variations on the Stochastic Shortest Path Problem^{*}

Mickael Randour¹, Jean-François Raskin², and Ocan Sankur²

¹ LSV, CNRS & ENS Cachan, France

² Département d'Informatique, Université Libre de Bruxelles (U.L.B.), Belgium

Abstract. In this invited contribution, we revisit the stochastic shortest path problem, and show how recent results allow one to improve over the classical solutions: we present algorithms to synthesize strategies with multiple guarantees on the distribution of the length of paths reaching a given target, rather than simply minimizing its expected value. The concepts and algorithms that we propose here are applications of more general results that have been obtained recently for Markov decision processes and that are described in a series of recent papers.

1 Introduction

Markov decision processes (MDP) [18] are natural models for systems that exhibit both non-deterministic and stochastic evolutions. An MDP is executed in rounds. In each round, the MDP is in a given state and an action is chosen by a controller (this is the resolution of non-determinism). Once this action has been fixed then the next state is determined following a probability distribution associated to the current state and the action that has been chosen by the controller. A controller can thus be considered as a *strategy* (a.k.a. *policy*) that determines which action to choose according to the history of the execution so far. MDPs have been studied intensively and there are algorithms to synthesize strategies that enforce a large variety of objectives like omega-regular objectives [9], PCTL objectives [1], or quantitative objectives [18].

One philosophy, three variants. The classical strategy synthesis setting often considers a single objective to be optimized such as the reachability probability, or the expected cost to target. Such simple objectives are not always sufficient to describe the properties required from an efficient controller. Indeed, on the one hand, one often has several measures of performance, and several objectives to satisfy, so the desired strategies have to settle for trade-offs between these. On the other hand, the strategies computed in the classical setting are tailored for the precise probabilities given in the MDP, which often correspond to the *average behavior* of the system in hand. This approach is not satisfactory if one is also interested in giving some formal guarantees under several scenarios, say,

^{*} Work partially supported by ERC starting grant inVEST (FP7-279499) and European project CASSTING (FP7-ICT-601148).

under normal conditions (i.e., average behavior), but also a minor failure, and a major failure. In this paper, we summarize recent results that we have obtained in this direction with the common goal of improving the strategies that can be synthesized for probabilistic systems. They were presented in three recent publications [5,20,19]. All three models that we studied share a common philosophy which is to provide a framework for the synthesis of strategies ensuring *richer performance guarantees* than the traditional models. The three problems we tackle can be summarized as follows.

First, in [5], we study a problem that is at the crossroad between the analysis of two-player zero-sum quantitative graph games and of quantitative MDPs. In the former, we want strategies for the controller that ensure a given minimal performance against *all* possible behaviors of its environment: we ask for strict guarantees on the *worst-case* performance. In the latter, the controller plays against a stochastic environment, and we want strategies that ensure a good expected performance, with no guarantee on individual outcomes. Both models have clear weaknesses: strategies that are good for the worst-case may exhibit suboptimal behaviors in probable situations while strategies that are good for the expectation may be terrible in some unlikely but possible situations. The *beyond worst-case synthesis problem* asks to construct strategies that provide both worst-case guarantees and guarantees on the expected value against a particular stochastic model of the environment given as input. We have considered both the mean-payoff value problem and the shortest path problem.

Second, in [19], we study multi-dimensional weighted MDPs, which are useful for modeling systems with multiple objectives. Those objectives may be conflicting, and so the analysis of trade-offs is important. To allow the analysis of those trade-offs, we study a general form of *percentile queries*. Percentile queries are as follows: given a multi-dimensional weighted MDP and a quantitative payoff function f (such as mean-payoff or truncated sum), quantitative thresholds v_i (one per dimension), and probability thresholds α_i , we show how to compute a *single* strategy that enforces that for all dimension i , the probability that an outcome ρ satisfies $f_i(\rho) \geq v_i$ is at least α_i . We have obtained several new complexity results on the associated decision problems and established efficient algorithms to solve these problems.

Third, in [20], we introduce *multi-environment* MDPs (MEMDPs) which are MDPs with a *set* of probabilistic transition functions. The goal in an MEMDP is to synthesize a single controller with guaranteed performances against *all* environments of this set even though the environment is unknown a priori. While MEMDPs can be seen as a special class of partially observable MDPs, several verification problems that are undecidable for partially observable MDPs, are decidable for MEMDPs and sometimes even allow for efficient solutions.

Stochastic shortest path. To illustrate those results in a uniform manner, we consider the *stochastic shortest path problem*, SSP problem, and study several variations. The *shortest path* problem is a classical optimization problem that asks, given a weighted graph, to find a path from a starting state to a target state such that the sum of weights along edges used in the path is minimized.

Stochastic variants consider edges with probabilistic distributions on destinations and/or on weights. We revisit here some of those variants at the light of the results that we have obtained in the contributions described above.

Structure of the paper. Our paper is organized as follows. In Sect. 2, we recall some elementary notions about MDPs. In Sect. 3, we define two classical stochastic variations on the SSP problem: the first one asks to minimize the expected length of paths to target, and the second one asks to force short paths with high probability. In Sect 4, we apply the *beyond worst-case analysis* to the shortest path problem and summarize our results presented in [5]. In Sect. 5, we consider a multi-dimension version of the shortest path problem where edges both have a length and a cost. We illustrate how *percentile queries*, that we have studied in [19], are natural objectives for the study of trade-offs in this setting. In Sect. 6, we study a version of the SSP where the stochastic information is given by several probabilistic transition relations instead of one, so we apply the *multi-environment* MDP analysis introduced in [20] on this variant. Throughout Sect. 4-6, we also give a summary of our general results on the corresponding models, as well as additional pointers to the literature.

2 Preliminaries

Markov decision processes. A (finite) *Markov decision process* (MDP) is a tuple $D = (S, s_{\text{init}}, A, \delta)$ where S is a finite set of *states*, $s_{\text{init}} \in S$ is the initial state, A is a finite set of *actions*, and $\delta: S \times A \rightarrow \mathcal{D}(S)$ 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 $\delta(s, a)(s')$. A *weighted* MDP $D = (S, s_{\text{init}}, A, \delta, w)$ is an MDP with a *d-dimension integer weight function* $w: A \rightarrow \mathbb{Z}^d$. For any dimension $i \in \{1, \dots, d\}$, we denote by $w_i: A \rightarrow \mathbb{Z}$ the projection of w to the i -th dimension, i is omitted when there is only one dimension.

We define a *run* ρ of D as a finite or infinite sequence $\rho = s_1 a_1 \dots a_{n-1} s_n \dots$ of states and actions such that $\delta(s_i, a_i, s_{i+1}) > 0$ for all $i \geq 1$. We denote the prefix of ρ up to state s_i by $\rho(i)$. A run is called *initial* if it starts in the initial state s_{init} . We denote the set of runs of D by $\mathcal{R}(D)$ and its set of *initial* runs by $\mathcal{R}_{s_{\text{init}}}(D)$. Finite runs that end in a state are also called *histories*, and denoted by $\mathcal{H}(D)$ and $\mathcal{H}_{s_{\text{init}}}(D)$, respectively.

Strategies. A *strategy* σ is a function $\mathcal{H}(D) \rightarrow \mathcal{D}(A)$ such that for all $h \in \mathcal{H}(D)$ ending in s , we have $\text{Supp}(\sigma(h)) \subseteq A(s)$, where Supp denotes the support of the probability distribution. The set of all possible strategies is denoted by Σ . A strategy is *pure* if all histories are mapped to *Dirac distributions*. A strategy σ can be encoded by a *stochastic Moore machine*, $(\mathcal{M}, \sigma_a, \sigma_u, \alpha)$ where \mathcal{M} is a finite or infinite set of memory elements; $\sigma_a: S \times \mathcal{M} \rightarrow \mathcal{D}(A)$ the *next action function* where $\text{Supp}(\sigma_a(s, m)) \subseteq A(s)$ for any $s \in S$ and $m \in \mathcal{M}$; $\sigma_u: A \times S \times \mathcal{M} \rightarrow \mathcal{D}(\mathcal{M})$ the *memory update function*; and α the *initial distribution on* \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 define such strategies as functions $s \mapsto \mathcal{D}(A(s))$ for all $s \in S$. Otherwise a strategy is *infinite-memory*.

Markov chains. A weighted *Markov chain* (MC) is a tuple $M = (S, d_{\text{init}}, \Delta, w)$ where S is a (non-necessarily finite) set of *states*, $d_{\text{init}} \in \mathcal{D}(S)$ is the initial distribution, $\Delta: S \rightarrow \mathcal{D}(S)$ is the *probabilistic transition function*, and $w: S \times S \rightarrow \mathbb{Z}^d$ is a *d-dimension weight function*. Markov chains are essentially MDPs where for all $s \in S$, we have that $|A(s)| = 1$.

We define a *run* of M as a finite or infinite sequence $s_1 s_2 \dots s_n \dots$ of states such that $\Delta(s_i, s_{i+1}) > 0$ for all $i \geq 1$. A run is called *initial* if it starts in the initial state s such that $d_{\text{init}}(s) > 0$. Runs of M are denoted by $\mathcal{R}(M)$, and its set of *initial* runs by $\mathcal{R}_{d_{\text{init}}}(M)$.

Markov chains induced by a strategy. An MDP $D = (S, s_{\text{init}}, A, \delta)$ and a strategy σ encoded by $(\mathcal{M}, \sigma_a, \sigma_u, \alpha)$ determine a Markov chain $M = D^\sigma$ 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_{init}, 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 $\sigma_a(s, m)(a) \cdot \delta(s, a, s') \cdot \sigma_u(s, m, a)(m')$. So, a *run* of D^σ is a finite or infinite sequence of the form $(s_1, m_1), a_1, (s_2, m_2), a_2, \dots$ where each $((s_i, m_i), a_i, (s_{i+1}, m_{i+1}))$ is a transition with non-zero probability in D^σ , and $s_1 = s_{\text{init}}$. In this case, the run $s_1 a_1 s_2 a_2 \dots$, obtained by projection to D , is said to be *compatible with σ* .

In an MC M , an *event* is a measurable set of runs $\mathcal{E} \subseteq \mathcal{R}_{d_{\text{init}}}(M)$. Every event has a uniquely defined probability [24] (Carathodory's extension theorem induces a unique probability measure on the Borel σ -algebra over $\mathcal{R}_{d_{\text{init}}}(M)$). We denote by $\mathbb{P}_M(\mathcal{E})$ the probability that a run belongs to \mathcal{E} when the initial state is chosen according to d_{init} , and M is executed for an infinite number of steps. Given a measurable function $f: \mathcal{R}(M) \rightarrow \mathbb{R} \cup \{\infty\}$, we denote by $\mathbb{E}_M(f)$ the *expected value* or *expectation* of f over initial runs in M . When considering probabilities of events in D^σ , for D an MDP and σ a strategy on D , we often consider runs defined by their projection on D . Thus, given $\mathcal{E} \subseteq \mathcal{R}(D)$, we denote by $\mathbb{P}_D^\sigma[\mathcal{E}]$ the probability of the runs of D^σ whose projection to D is in \mathcal{E} .

3 The stochastic shortest path problem

The *shortest path problem* in a weighted graph is a classical problem that asks, given a starting state s and a set of target states T , to find a path from s to a state $t \in T$ of minimal length (i.e., that minimizes the sum of the weights along the edges in the path). See for example [8]. There have been several stochastic variants of this classical graph problem defined and studied in the literature, see for example [18]. We recall here two main variants of this problem, other new variants are defined and studied in the subsequent sections.

Let $D = (S, s_{\text{init}}, A, \delta, w)$ be an MDP with a *single-dimensional* weight function $w: A \rightarrow \mathbb{N}_0$ that assigns to each action $a \in A$ a strictly positive integer.

Let $T \subseteq S$ be a set of target states. Given an initial run $\rho = s_1 s_2 \dots s_i \dots$ in the MDP, we define its *truncated sum* up to T to be $\text{TS}^T(\rho) = \sum_{j=1}^{n-1} w(a_j)$ if s_n is the first visit of a state in $T \subseteq S$ within ρ ; otherwise if T is never reached, then we set $\text{TS}^T(\rho) = \infty$. The function TS^T is measurable, and so this function has an expected value in a weighted MC and sets of runs defined from TS^T are measurable. The following two problems have been considered in the literature.

Minimizing the expected length of paths to target. Given a weighted MDP, we may be interested in strategies (choices of actions) that minimize the expected length of paths to target. This is called the *stochastic shortest path expectation* problem, SSP-E for short, and it is defined as follows.

Definition 1 (SSP-E problem). *Given a single-dimensional weighted MDP $D = (S, s_{\text{init}}, A, \delta, w)$ and a threshold $\ell \in \mathbb{N}$, decide if there exists σ such that $\mathbb{E}_D^\sigma(\text{TS}^T) \leq \ell$.*

Theorem 1 ([2]). *The SSP-E problem can be decided in polynomial time. Optimal pure memoryless strategies always exist and can be constructed in polynomial time.*

There are several algorithms proposed in the literature to solve this problem. We recall a simple one based on linear programming (LP). For other solutions based on *value iteration* or *strategy iteration*, we refer the interested reader to, e.g., [2,10]. To apply the reduction to LP, we must make the hypothesis that, for each state $s \in S$ of the MDP, there is a path from s to the target set T . It is clear that the expectation of states that are not connected to the target set T by a path is infinite. So, we will assume that all such states have first been removed from the MDP. This can easily be done in linear time. Also, it is clear that for all states in T , the expected length of the shortest path is trivially equal to 0. So, we restrict our attention to states in $S \setminus T$. For each state $s \in S \setminus T$, we consider one variable x_s , and we define the following linear program:

$$\max \sum_{s \in S \setminus T} x_s$$

under the constraints

$$x_s \leq w(a) + \sum_{s' \in S \setminus T} \delta(s, a, s') \cdot x_{s'} \quad \text{for all } s \in S \setminus T, \text{ for all } a \in A(s).$$

It can be shown (e.g., in [2]) that the optimal solution \mathbf{v} for this LP is such that \mathbf{v}_s is the expectation of the length of the shortest path from s to a state in T under an optimal strategy. Such an optimal strategy can easily be constructed from the optimal solution \mathbf{v} . The following pure memoryless strategy $\sigma^{\mathbf{v}}$ is optimal:

$$\sigma^{\mathbf{v}}(s) = \arg \min_{a \in A(s)} \left[w(a) + \sum_{s' \in S \setminus T} \delta(s, a, s') \cdot \mathbf{v}_{s'} \right].$$

Forcing short paths with high probability. As an alternative to the expectation, given a weighted MDP, we may be interested in strategies that maximize the probability of short paths to target. This is called the *stochastic shortest path percentile* problem, SSP-P for short, and provides a preferable solution if we are risk-averse. The problem is defined as follows.

Definition 2 (SSP-P problem). *Given a single-dimensional weighted MDP $D = (S, s_{\text{init}}, A, \delta, w)$, value $\ell \in \mathbb{N}$, and probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$, decide if there exists a strategy σ such that $\mathbb{P}_D^\sigma[\{\rho \in \mathcal{R}_{s_{\text{init}}}(D) \mid \text{TS}^T(\rho) \leq \ell\}] \geq \alpha$.*

Theorem 2. *The SSP-P problem can be decided in pseudo-polynomial time, and it is PSPACE-hard. Optimal pure strategies with exponential memory always exist and can be constructed in exponential time.*

The PSPACE-hardness result was recently proved in [15]. An algorithm to solve this problem can be obtained by a (pseudo-polynomial-time) reduction to the *stochastic reachability problem*, SR for short.

Given an unweighted MDP $D = (S, s_{\text{init}}, A, \delta)$, a set of target states $T \subseteq S$, and a probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$, the SR problem asks to decide if there is a strategy σ that ensures, when played from s_{init} , to reach the set T with a probability that exceeds the threshold α . The SR problem can also be solved in polynomial time by a reduction to linear programming. Here is a description of the LP. For all states $s \in S$, we consider a variable x_s in the following LP:

$$\min \sum_{s \in S} x_s$$

under the constraints

$$\begin{aligned} x_s &= 1 && \forall s \in T, \\ x_s &= 0 && \forall s \in S \text{ which cannot reach } T, \\ x_s &\geq \sum_{s' \in S} \delta(s, a, s') \cdot x_{s'} && \forall a \in A(s). \end{aligned}$$

The optimal solution \mathbf{v} for this LP is such that \mathbf{v}_s is the maximal probability to reach the set of targets T that can be achieved from s . From the optimal solution \mathbf{v} , we can define a pure memoryless strategy $\sigma^{\mathbf{v}}$ which achieve \mathbf{v}_s when played from s , we define it for all states $s \notin T$ that can reach T :

$$\sigma^{\mathbf{v}}(s) = \arg \max_{a \in A(s)} \left[\sum_{s' \in S} \delta(s, a, s') \cdot x_{s'} \right].$$

We are now ready to define the reduction from the SSP-P problem to the SR problem. Given a weighted MDP $D = (S, s_{\text{init}}, A, \delta, w)$, a set of targets $T \subseteq S$, a value $\ell \in \mathbb{N}$, and a probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$, we construct an MDP D_ℓ . D_ℓ is constructed from D and contains an additional information in its state space: it records the sum of the weights encountered so far. Formally, $D_\ell = (S', s'_{\text{init}}, A', \delta', w')$ where:

- S' is the set of states, each one being a pair (s, v) , where $s \in S$ and $v \in \{0, 1, \dots, \ell\} \cup \{\perp\}$. Intuitively v records the running sum along an execution in D ($\perp > \ell$ by convention);
- its initial state s'_{init} is equal to $(s_{\text{init}}, 0)$;
- the set of actions is A and the weight function is unchanged, i.e., $A' = A$ and $w' = w$;
- the transition relation is as follows: for all pairs $(s, v), (s', v') \in S'$, and actions $a \in A$, we have that $\delta((s, v), a)(s', v') = \delta(s, a)(s')$ if $v' = v + w(a) \leq \ell$, $\delta((s, v), a)(s', v') = \delta(s, a)(s')$ if $v' = \perp$ and $v + w(a) > \ell$, and $\delta((s, v), a)(s', v') = 0$ otherwise.

The size of D_ℓ is proportional to the size of D and the value ℓ , i.e., it is thus *pseudo-polynomial* in the encoding of the SSP-P problem. The SR objective in D_ℓ is to reach $T' = \{(s, v) \mid s \in T \wedge v \leq \ell\}$ with a probability at least α .

Runs that satisfy the reachability objective in D_ℓ are in bijection with runs that reach T in D with a truncated sum at most ℓ . So if there is a strategy that enforces reaching T' in D_ℓ with probability $p \geq \alpha$, then there is a strategy in D to ensure that T is reached with a path of length at most ℓ with probability $p \geq \alpha$ (the strategy that is followed in D_ℓ can be followed in D if we remember what is the sum of weights so far). The converse also holds. As for a reachability objective, memoryless strategies are optimal, we deduce that pseudo-polynomial-size memory is sufficient (and is sometimes necessary) in the SSP-P problem, and the problem can be solved in pseudo-polynomial time. As the problem has been shown to be PSPACE-Hard, this pseudo-polynomial-time solution is essentially optimal, see [15] for details.

Illustration. We illustrate the concepts of this paper on a running example that we have introduced in [5] and that we extend in the subsequent sections. The MDP of Fig. 1 models the choices that an employee faces when he wants to reach work from home. He has the choice between taking the train, driving or biking. When he decides to bike, he reaches his office in 45 minutes. If he decides to take his car, then the journey depends on traffic conditions that are modeled by a probabilistic distribution between light, medium and heavy traffic. The employee can also try to catch a train, which takes 35 minutes to reach work. But trains can be delayed (potentially multiple times): in that case, the employee decides if he waits or if he goes back home (and then take his car or his bike). We consider two scenarios that correspond to the two problems defined above.

If the employee wants to minimize the expected duration of his journey from home to work, we need to solve a SSP-E problem. By Theorem 1, we know that pure memoryless strategies suffice to be optimal. It turns out that in our example, taking the car is the strategy that minimizes the expected time to work: this choice gives an expectation equal to 33 minutes.

Observe that taking the car presents some risk: if the traffic is heavy, then work is only reached after 71 minutes. This can be unacceptable for the employee's boss if it happens too frequently. So if the employee is *risk-averse*, optimizing the expectation may not be the best choice. For example, the employee may want to reach work within 40 minutes with high probability, say 95%. In

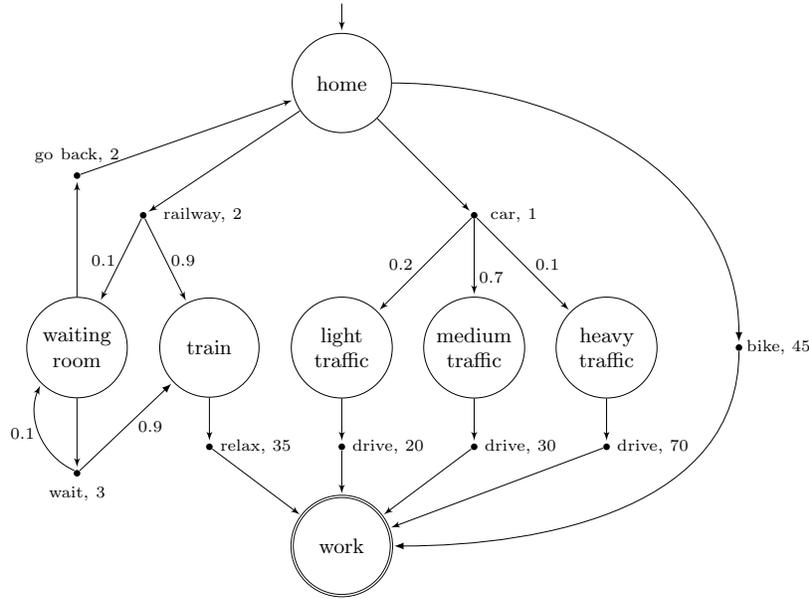


Fig. 1. An everyday life application of stochastic shortest path problems: choosing a mean of transport to go from home to work. Actions (black dots) are labeled with durations in minutes, and stochastic transitions are labeled with their probability.

this case, we need to solve a SSP-P problem. First, observe that taking the train ensures to reach work within 40 minutes in 99% of the runs. Indeed, if the train is not delayed, we reach work with 37 minutes, and this happens with probability $9/10$. Now, if the train is late and the employee decides to wait, the train arrives in the next 3 minutes with probability $9/10$: in that case, the employee arrives at work within 40 minutes. So, the strategy consisting in going to the railway station and waiting for the train (as long as needed) gives us a probability $99/100$ to reach work within 40 minutes, fulfilling our objective. Second, it is easy to see that both bicycle and car are excluded in order to satisfy the SSP-P problem. With bicycle we reach work in 45 minutes with probability one, and with the car we reach work in 71 minutes with probability $1/10$, hence we miss the constraint of 40 minutes too often.

Related work. The SSP-P problem was studied in MDPs with either all non-negative or all non-positive weights in [17,22]. The related notion of *quantile queries* was studied in [23]: such queries are essentially equivalent to minimizing the value ℓ inside the constraint of an SSP-P problem such that there still exists a satisfying strategy for some fixed α . It has been recently extended to *cost problems* [15], which can handle arbitrary Boolean combinations of inequalities over the truncated sum instead of only $\text{TS}^T(\rho) \leq \ell$. All those works only study single-dimensional MDPs. For the SPP-E problem, extensions to multi-dimensional MDPs have been considered in [14].

4 Good expectation under acceptable worst-case

Worst-case guarantees. Assume now that the employee wants a strategy to go from home to work such that work is *guaranteed* to be reached within 60 minutes (e.g., to avoid missing an important meeting with his boss). It is clear that both optimal (w.r.t. problems SSP-E and SSP-P respectively) strategies of Sect. 3 are excluded: there is the possibility of heavy traffic with the car (and a journey of 71 minutes), and trains can be delayed indefinitely in the *worst case*.

To ensure a strict upper bound on the length of the path, an adequate model is the *shortest path game problem*, SP-G for short. In a shortest path game, the uncertainty becomes adversarial: when there is some uncertainty about the outcome of an action, we do not consider a probabilistic model but we let an *adversary* decide the outcome of the action. So, to model a shortest path game based on an MDP $D = (S, s_{\text{init}}, A, \delta, w)$, we modify the interpretation of the transition relation as follows: after some history h that ends up in state s , if the strategy chooses action $a \in A(s)$, then the adversary chooses the successor state within $\text{Supp}(\delta(s, a))$ without taking into account the actual values of the probabilities. With this intuition in mind, if we fix a strategy σ (for the controller), then the set of possible *outcomes* in D , noted Out_D^σ , is the set of initial runs that are compatible with σ , i.e., $\text{Out}_D^\sigma = \{\rho \in \mathcal{R}_{s_{\text{init}}}(D) \mid \forall i \geq 0: a_i \in \text{Supp}(\sigma(\rho(i)))\}$. Now, we can define the SP-G problem as follows.

Definition 3 (SP-G problem). *Given single-dimensional weighted MDP $D = (S, s_{\text{init}}, A, \delta, w)$, set of target states $T \subseteq S$, and value $\ell \in \mathbb{N}$, decide if there exists a strategy σ such that for all $\rho \in \text{Out}_D^\sigma$, we have that $\text{TS}^T(\rho) \leq \ell$.*

Theorem 3 ([16]). *The SP-G problem can be decided in polynomial time. Optimal pure memoryless strategies always exist and can be constructed in polynomial time.*

Under the hypothesis that actions in D have strictly positive weight, the controller has no interest in forming cycles, and if he cannot avoid to close cycles (before reaching T), then there will be outcomes that will never reach T , yielding an infinite truncated sum. As a consequence, the only option for the controller is to win within $|S| = n$ steps. So, to solve the SP-G problem, we compute for each state s and for each i , $0 \leq i \leq n$, the value $\mathbb{C}(s, i)$, representing the lowest bound on the length to the target T from s that the controller can ensure, if the game is played for i steps. Those values can be computed using *dynamic programming* as follows: for all $s \in T$, $\mathbb{C}(s, 0) = 0$, and for all $s \in S \setminus T$, $\mathbb{C}(s, 0) = +\infty$. Now, assume that $0 < i < n$ and that we have already computed $\mathbb{C}(s, i - 1)$ for all $s \in S$. Then for i steps, we have that

$$\mathbb{C}(s, i) = \min \left[\mathbb{C}(s, i - 1), \min_{a \in A(s)} \max_{s' \in \text{Supp}(\delta(s, a))} w(a) + \mathbb{C}(s', i - 1) \right].$$

So, $\mathbb{C}(s_{\text{init}}, n)$ can be computed in polynomial time, and we have that the controller can force to reach T from s_{init} with a path of length at most ℓ if and only if $\mathbb{C}(s_{\text{init}}, n) \leq \ell$.

Related work. For results about the SP-G problem when weights can also be negative, we refer the interested reader to [3] where a pseudo-polynomial-time algorithm has been designed and to [11] where complexity issues are discussed (see Theorem 8 in that reference). In multi-dimensional MDPs with both positive and negative weights, it follows from results on total-payoff games that the SP-G problem is undecidable [7].

Illustration. If we apply this technique on the example of Fig. 1, it shows that taking bicycle is a safe option to ensure the strict 60 minutes upper bound. However, the expected time to reach work when following this strategy is 45 minutes, which is far from the optimum of 33 minutes that can be obtained when we neglect the worst-case constraint.

In answer to this, we may be interested in synthesizing a strategy that minimizes the expected time to work under the constraint that work is reached within 60 minutes in the worst case. We claim that the optimal strategy in this case is the following: try to take the train, if the train is delayed three times consecutively, then go back home and take the bicycle. This strategy is safe as it always reaches work within 58 minutes and its expectation is $\approx 37,34$ minutes (so better than taking directly the bicycle). Observe that it is pure but requires finite memory, in contrast to the case of problems SSP-E and SSP-G.

Beyond worst-case synthesis. In [5,4], we study the synthesis of strategies that ensure, *simultaneously*, a worst-case threshold (when probabilities are replaced by adversarial choices), and a good expectation (when probabilities are taken into account). We can now recall the precise definition of the problem.

Definition 4 (SSP-WE problem). *Given a single-dimensional weighted MDP $D = (S, s_{\text{init}}, A, \delta, w)$, a set of target states $T \subseteq S$, and two values $\ell_1, \ell_2 \in \mathbb{N}$, decide if there exists a strategy σ such that:*

1. $\forall \rho \in \text{Out}_D^\sigma: \text{TS}^T(\rho) \leq \ell_1,$
2. $\mathbb{E}_D^\sigma(\text{TS}^T) \leq \ell_2.$

While the SP-G problem and the SSP-E problem are both solvable in polynomial time and pure memoryless strategies suffice in both cases, the SSP-WE problem proves to be inherently harder.

Theorem 4 ([5]). *The SSP-WE problem can be decided in pseudo-polynomial time and is NP-hard. Pseudo-polynomial memory is always sufficient and in general necessary, and satisfying strategies can be constructed in pseudo-polynomial time.*

The algorithm proposed in [5] to solve the SSP-WE problem can be summarized as follows. First, construct the MDP D_ℓ as for solving the SSP-P problem. States of D_ℓ are pairs (s, v) where $s \in S$ is a state of D and v is the sum of weights of edges traversed so far. Consider the target $T' = \{(s, v) \mid s \in T \wedge v \leq \ell\}$. Second, compute for each state (s, v) what are the *safe actions*, noted $\mathbb{A}(s, v)$, that ensure to reach T' in D_ℓ no matter how the adversary resolves non-determinism. $\mathbb{A}(s, v)$ can be computed inductively as follows: we start with $\mathbb{A}_0(s, v) = A(s)$ if $v \leq \ell$

and $\mathbb{A}_0(s, v) = \emptyset$ if $v = \perp$, i.e., a priori, all the actions are good in states that have not yet exceeded the sum ℓ while states that have exceeded ℓ are hopeless and none of the actions are good. Assume that we have computed $\mathbb{A}_i(s, v)$, for $i \geq 0$, then $\mathbb{A}_{i+1}(s, v) = \{a \in \mathbb{A}_i(s, v) \mid \forall (s', v') \in \text{Supp}(\delta((s, v), a)): \mathbb{A}_i(s', v') \neq \emptyset\}$. As the set of good actions is finite and is decreasing, it is easy to see that this process ends after a finite number of steps that is polynomial in the size of D_ℓ . We note $D_\ell^\mathbb{A}$, the MDP D_ℓ limited to the safe actions. Then, it remains to solve the SSP-E on $D_\ell^\mathbb{A}$. The overall complexity of the algorithm is pseudo-polynomial, and the NP-hardness result established in [5] implies that we cannot hope to obtain a truly-polynomial-time algorithm unless $\text{P} = \text{NP}$.

Additional results. In [5,4], we also study the so-called beyond worst-case synthesis for models with the *mean-payoff* function instead of the truncated sum. Mean-payoff games [12] are infinite-duration, two-player zero-sum games played on weighted graphs. In those games, the controller wants to maximize the long-run average of the weights of the edges traversed during the game while the adversary aims to minimize this long-run average. Given a mean-payoff game and a stochastic model of the adversary, their product defines an MDP on which we study the problem MP-WE, the mean-payoff analogue of problem SSP-WE. We have shown that it is in $\text{NP} \cap \text{coNP}$ for *finite-memory* strategies, essentially matching the complexity of the simpler problem MP-G of solving mean-payoff games without considering the expected value. We have also established that pure strategies with pseudo-polynomial-memory are sufficient. Our synthesis algorithm is much more complex than for SSP-WE, and requires to overcome several technical difficulties to prove $\text{NP} \cap \text{coNP}$ -membership.

5 Percentile queries in multi-dimensional MDPs

Illustration. Consider the MDP D depicted in Fig. 2. It gives a simplified choice model for commuting from home to work, but introduces two-dimensional weights: each action is labeled with a duration, in minutes, and a cost, in dollars. 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 a choice of transportation that gives us high probability to reach work in due time but also limits the risk of an expensive journey. Since faster options are often more expensive, trade-offs have to be considered.

Recall the SSP-P problem presented in Def. 2: it asks to decide the existence of strategies satisfying a *single percentile constraint*. This problem can only be applied to single-dimensional MDPs. For example, one may look for a strategy that ensures that 80% of compatible initial runs take at most 40 minutes (constraint C1), *or* that 50% of them cost at most 10 dollars (C2). A good strategy for C1 would be to take the taxi, which guarantees that work is reached within 10 minutes with probability $0.99 > 0.8$. For C2, taking the bus is a good option, because already 70% of the runs will reach work for only 3 dollars. Note that taking the taxi does not satisfy C2, nor does taking the bus satisfy C1.

In practice, a desirable strategy should be able to satisfy *both* C1 and C2. This is the goal of our model of *multi-constraint percentile queries*, introduced in [19]. For example, an appropriate strategy for the conjunction $(C1 \wedge C2)$ is to try the bus once, and then take the taxi if the bus does not depart. Indeed, this strategy ensures that work is reached within 40 minutes with probability larger than 0.99 thanks to runs home-bus-work (probability 0.7 and duration 30) and home-bus-home-taxi-work (probability 0.297 and duration 40). Furthermore, it also ensures that more than half the time, the total cost to target is at most 10 dollars, thanks

to run home-bus-work which has probability 0.7 and cost 3. Observe that this strategy requires *memory*. In this particular example, it is possible to build another acceptable strategy which is memoryless but requires *randomness*. Consider the strategy that flips an unfair coin in home to decide if we take the bus or the taxi, with probabilities $3/5$ and $2/5$ respectively. Constraint C1 is ensured thanks to runs home-bus-work (probability 0.42) and home-taxi-work (probability 0.396). Constraint C2 is ensured thanks to runs $(\text{home-bus})^n \cdot \text{work}$ with $n = 1, 2, 3$: they have probabilities $0.42, \geq 0.07$ and ≥ 0.01 respectively, totaling to ≥ 0.5 , while they all have cost at most $3 \cdot 3 = 9 < 10$. As we will see, percentile queries in general require strategies that both use memory *and* randomness, in contrast to the previous problems which could forgo randomness.

Percentile queries. In [19], we study the synthesis of strategies that enforce percentile queries for the shortest path.

Definition 5 (SSP-PQ problem). *Given a d -dimensional weighted MDP $D = (S, s_{\text{init}}, A, \delta, w)$, and $q \in \mathbb{N}$ percentile constraints described by sets of target states $T_i \subseteq S$, dimensions $k_i \in \{1, \dots, d\}$, value thresholds $\ell_i \in \mathbb{N}$ and probability thresholds $\alpha_i \in [0, 1] \cap \mathbb{Q}$, where $i \in \{1, \dots, q\}$, decide if there exists a strategy σ such that*

$$\forall i \in \{1, \dots, q\}, \mathbb{P}_D^\sigma [\text{TS}_{k_i}^{T_i} \leq \ell_i] \geq \alpha_i,$$

where $\text{TS}_{k_i}^{T_i}$ denotes the truncated sum on dimension k_i and w.r.t. target set T_i .

Our algorithm is able to solve the problem for queries with multiple constraints, potentially related to different dimensions of the weight function and to different target sets: this offers great flexibility which is useful in modeling applications.

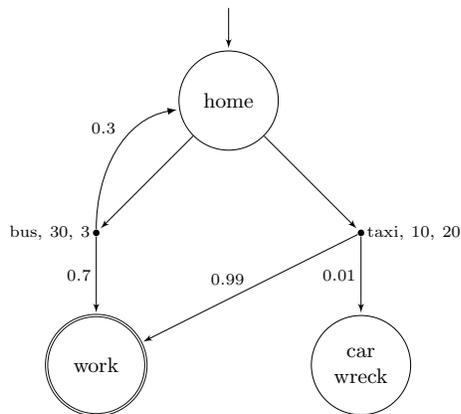


Fig. 2. Multi-percentile queries can help when actions both impact the duration of the journey (first dimension) and its cost (second dimension): trade-offs have to be considered.

Theorem 5 ([19]). *The SSP-PQ problem can be decided in exponential time in general, and pseudo-polynomial time for single-dimension single-target multi-constraint queries. The problem is PSPACE-hard even for single-constraint queries. Randomized exponential-memory strategies are always sufficient and in general necessary, and satisfying strategies can be constructed in exponential time.*

The first step to solve an SSP-PQ problem on MDP D is to build a new MDP D_ℓ similarly to what was defined for the SSP-P problem, but with $\ell = \max_i \ell_i$, and adapting the construction to multi-dimensional weights. In particular, we observe that a run can only be disregarded when the sum on each of its dimensions exceeds ℓ . Essentially, some runs may satisfy only a subset of constraints and still be interesting for the controller, as seen in the example above. Still, the size of D_ℓ can be maintained to a *single*-exponential by defining a suitable equivalence relation between states (pseudo-polynomial for single-dimensional MDPs and single-target queries). Precisely, the states of D_ℓ are in $S \times (\{0, \dots, \ell\} \cup \{\perp\})^d$. Now, for each constraint i , we compute a set of target states R_i in D_ℓ that exactly captures all runs satisfying the inequality of the constraint.

We are left with a *multiple reachability problem* on D_ℓ : we look for a strategy σ_ℓ that ensures that each of these sets R_i is reached with probability α_i . This is a generalization of the SR problem defined above. It follows from [13] that this multiple reachability problem can be answered in time polynomial in $|D_\ell|$ but exponential in the number of sets R_i , i.e., in q . The complexity can be reduced for single-dimensional MDPs and queries with a unique target T : in that case, sets R_i can be made absorbing, and the multiple reachability problem can be answered in time polynomial in $|D_\ell|$ through linear programming. Overall, our algorithm thus requires pseudo-polynomial time in that case. It is clear that σ_ℓ can be easily translated to a good strategy σ in D and conversely.

The PSPACE-hardness result already holds for the single-constraint case, i.e., the SSP-P problem (Theorem 2), following results of [15]. Hence the SSP-PQ framework offers a wide extension for basically no price in decision complexity.

Additional results. In [19], we establish that the SSP-PQ problem becomes undecidable if we allow for both negative and positive weights in multi-dimensional MDPs, even with a unique target set.

Furthermore, in [19], we study the concept of percentile queries for a large range of classical payoff functions, not limited to the truncated sum: *sup*, *inf*, *limsup*, *liminf*, *mean-payoff* and *discounted sum*. In all cases, the complexity for the most general setting - multi-dimensional MDPs, multiple constraints - is at most exponential, better in some cases. Interestingly, when the query size is fixed, all problems except for the discounted sum can be solved in polynomial time. Note that in most applications, the query size can be reasonably bounded while the model can be very large, so this framework is ideally suited. In many cases, we show how to reduce the complexity for single-dimensional queries, and for single-constraint queries. We also improve the knowledge of the multiple reachability problem sketched above by proving its PSPACE-hardness and identifying the subclass of queries with nested targets as solvable in polynomial time.

Related work. As mentioned in Sect. 3, there are several works that extend the SSP-P problem in different directions. In particular, *cost problems*, recently introduced in [15], can handle arbitrary Boolean combinations of inequalities φ over the truncated sum inside an SSP-P problem: it can be written as $\mathbb{P}_D^\sigma[\text{TS}^T \models \varphi] \geq \alpha$. Observe that this is orthogonal to our percentile queries. 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 whereas we analyze each event independently. Both settings are in general incomparable (see [19]), but they share the SSP-P problem as a common subclass.

6 Multiple environments

The probabilities in a stochastic process represent a model of the *environment*. For instance, in Fig. 1, the probability of a train coming when we wait in the train station is a simplified model of the behavior of the train network. Clearly, this behavior can be significantly different on some particular days, for instance, when there is a strike. In this section, we consider the problem of synthesizing strategies in probabilistic systems with guarantees against a finite number of different *environments*.

Illustration. Let us consider again the problem of commuting to work, and assume that some days there may be an unannounced strike (S) in the train service, and an accident (A) in the highway. Thus, four settings are possible: (\emptyset), (A), (S), (AS). When there is a strike, there is no train service; and when there is an accident, the highway is blocked. We assume that we are not informed of the strike or the accident in advance. Our goal is to synthesize a strategy with guarantees against these four environments with *no prior knowledge* of the situation we are in.

Consider the MDP D of Fig. 3, which models the *normal conditions* without strike or accident. We will define three different MDPs from D on the same state space to model the three other environments, by modifying the probabilities of dotted edges. For each environment $E \in \{(A), (S), (AS)\}$, we define MDP D^E from D as follows.

1. For $D^{(S)}$, action `wait` from state `stat` deterministically leads back to `stat`.
2. For $D^{(A)}$, action `go` from state `h2` deterministically leads back to `h2`.
3. For $D^{(AS)}$, we apply both items 1 and 2.

Note that if strikes and accidents have small probabilities, instead of creating separate models, one could integrate their effect in a single model by adjusting the probabilities in D , for instance, by reducing the probability of moving forward in the highway. Such an approach may be useful (and simpler) for an *average analysis*. However, we are interested here in giving guarantees against each scenario rather than optimizing a global average. Our formulation can rather be

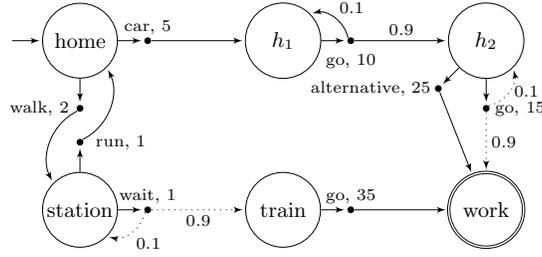


Fig. 3. Commuting to work. States h_1, h_2 represent sections of the highway. After h_1 , one may take an alternative road which is longer but not affected by traffic.

modeled by *partially observable* MDPs since the strategy is not aware of the state of the system. However, most problems are undecidable in this setting [6].

Our objective is to get to work with high probability within reasonable time. More precisely, we would like to make sure to be at work, with probability 0.95 in all cases: in 40 minutes if there is no strike, in 50 minutes if there is a strike but no accident, and 75 minutes if there is a strike and an accident. More formally, we would like to synthesize a *single* strategy σ such that:

$$\begin{aligned}
 & - \mathbb{P}_D^\sigma[\text{TS}^T \leq 40] \geq 0.95, & - \mathbb{P}_{D(A)}^\sigma[\text{TS}^T \leq 40] \geq 0.95, \\
 & - \mathbb{P}_{D(S)}^\sigma[\text{TS}^T \leq 50] \geq 0.95, & - \mathbb{P}_{D(SA)}^\sigma[\text{TS}^T \leq 75] \geq 0.95.
 \end{aligned}$$

Solution. We will describe a strategy that satisfies our objective. First, note that we shouldn't take the car right away since even if we take the alternative road, we will be at work in 40 minutes only with probability 0.90 (even if there is no accident, we may spend 20 minutes in h_1). Our strategy is the following. We first walk to the train station, and wait there at most twice. Clearly, if there is no strike, we get to work in less than 40 minutes with probability at least 0.99. Otherwise, we run back home, and take the car. Note that we already spent 5 minutes at this point. Our strategy on the highway is the following. We take the alternative road if, and only if we failed to make progress twice by taking action go (e.g., we observed $h_1 \cdot \text{go} \cdot h_1 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot h_2$).

We already saw that in the absence of strike, this strategy satisfies our objective. If there is a strike but no accident, we will surely take the car. Then the history ending with $h_1 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot \text{work}$ has probability 0.81 and takes 30 minutes. The histories ending with $h_1 \cdot \text{go} \cdot h_1 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot \text{work}$ and $h_1 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot \text{work}$ have each probability 0.081 and take 40 and 45 minutes respectively. Overall, with probability at least 0.97 we get to work in at most 50 minutes. If there is a strike and an accident, then the history $h_2 \cdot \text{go} \cdot \text{work}$ is never observed. In this case, the history ending with $h_1 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot h_2 \cdot \text{alternative}$ has probability 0.90 and takes 75 minutes, and history $h_1 \cdot \text{go} \cdot h_1 \cdot \text{go} \cdot h_2 \cdot \text{go} \cdot h_2 \cdot \text{alternative}$ has probability 0.09 and takes 75 minutes. Hence we ensure the constraint with probability 0.99.

Algorithms. Formally, we define a *multi-environment MDP* as a tuple $D = (S, s_{\text{init}}, A, (\delta_i)_{1 \leq i \leq k}, (w_i)_{1 \leq i \leq k})$, where each $(S, s_{\text{init}}, A, \delta_i, w_i)$ is an MDP, corresponding to a different environment.

Definition 6 (SSP-ME problem). *Given any single-dimensional multi-environment MDP $D = (S, s_{\text{init}}, A, (\delta_i)_{1 \leq i \leq k}, (w_i)_{1 \leq i \leq k})$, target states $T \subseteq S$, thresholds $\ell_1, \dots, \ell_k \in \mathbb{N}$, and probabilities $\alpha_1, \dots, \alpha_k \in [0, 1] \cap \mathbb{Q}$, decide if there exists a strategy σ satisfying*

$$\forall i \in \{1, \dots, k\}, \mathbb{P}_{D_i}^\sigma[\text{TS}^T \leq \ell_i] \geq \alpha_i.$$

For the particular case of $\alpha_1 = \dots = \alpha_k = 1$, the problem is called the *almost-sure SSP-ME problem*. The *limit-sure SSP-ME problem* asks whether the SSP-ME problem has a solution for *all* probability vectors $(\alpha_1, \dots, \alpha_k) \in]0, 1[^k$. If the limit-sure problem can be satisfied, the almost-sure case can be approximated arbitrarily closely. Note that in some multi-environment MDPs, the limit-sure SSP-ME problem has a solution although the almost-sure one does not.

Theorem 6 ([20]). *The almost-sure and limit-sure SSP-ME problems can be solved in pseudo-polynomial time for a fixed number of environments. Finite memory suffices for the almost-sure case, and a family of finite-memory strategies that witnesses the limit-sure problem can be computed.*

We analyze the structure of the MDPs to identify *learning components* in which one can almost-surely (resp. limit-surely) determine the current environment. Once these are identified, one can transform the MDPs into simpler forms on which known algorithms on (single-environment) MDPs are applied [21].

For an example of a *learning component*, consider two states s, t and action a , with $\delta_1(s, a, t) = 0.9, \delta_1(s, a, s) = 0.1$, and $\delta_1(t, a, s) = 1$ for the first environment, and $\delta_2(s, a, t) = 0.1, \delta_2(s, a, s) = 0.9$, and $\delta_2(t, a, s) = 1$ for the second environment. Now, at state s , repeating the action a a large number of times, and looking at the generated history, one can guess with arbitrarily high confidence the current environment. However, no strategy can guess the environment with certainty. If, we rather set $\delta_1(s, a, t) = 1$, and $\delta_2(s, a, s) = 1$, then an observed history uniquely determines the current environment.

For the general SSP-ME problem, there is an algorithm for an *approximate* version of the above problem, namely the ε -gap problem. For any $\varepsilon > 0$, a procedure for the ε -gap SSP-ME problem answers Yes if the SSP-ME problem has a solution; it answers No if the SSP-ME problem has no solution when each α_i is replaced with $\alpha_i - \varepsilon$; and answers either Yes or No otherwise. Intuitively, such a procedure gives a correct answer on positive instances, and on instances that are clearly too far (by ε) to be satisfiable. However, there is an uncertainty zone of size ε on which the answer is not guaranteed to be correct. The algorithm is based on a reduction to the first order theory of the reals (see [20]).

Theorem 7. *The SSP-ME problem and the ε -gap SSP-ME are NP-hard. For any $\varepsilon > 0$, there is a procedure for the ε -gap SSP-ME problem.*

Additional results. In [21], we restricted our study to MDPs with *two* environments, and considered reachability, safety, and parity objectives. We proved these problems to be decidable in polynomial time for almost-sure and limit-sure

conditions. The general quantitative case, i.e., arbitrary satisfaction probabilities is shown to be NP-hard already for two environments and MDPs with no cycles other than self-loops. We gave a doubly exponential-space procedure to solve the ε -gap problem for reachability. We are currently studying the exact complexity of the case of arbitrary number of environments.

7 Conclusion

Through this paper, we gave an overview of classical approaches to the quantitative evaluation of strategies in MDPs, and presented three recent extensions that increase the modeling power of that framework. We chose to illustrate them through application to the stochastic shortest path problem. We hope this helps in understanding and comparing the different approaches. Let us sum up.

Given a weighted MDP modeling a stochastic shortest path problem, a first natural question is to find a strategy that minimizes the *expected* sum of weights to target. This is the SSP-E problem. Optimizing the average behavior of the controller is interesting if the process is to be executed a great number of times, but it gives no guarantee on individual runs, which may perform very badly. For a risk-averse controller, it may be interesting to look at the SSP-P problem, which asks to maximize the *probability* that runs exhibit an acceptable performance. When one really wants to ensure that no run will have an unacceptable performance, it is useful to resort to the SSP-G problem, which asks to optimize the *worst-case* performance of the controller.

In recent works, we introduced three related models that may be used to synthesize strategies with richer performance guarantees. First, if one reasons using the SSP-G problem, he may obtain a strategy which is sub-optimal on average while using the SSP-E problem gives no worst-case guarantee. With the framework of *beyond worst-case synthesis*, developed in [5,4], and presented here as the SSP-WE problem, we can build strategies that provide *both* worst-case guarantees and good expectation. Second, we are interested in describing rich constraints on the performance profile of strategies in multi-dimensional MDPs. To that end, we extended the SSP-P problem to the SSP-PQ problem, which handles multi-constraint *percentile queries* [19]. Those queries are particularly useful to characterize trade-offs between, for example, the length of a journey and its cost. Third and finally, we have discussed another extension of the SSP-P problem that models some uncertainty about the stochastic model of the environment which is defined in the MDP through the transition function. With the SSP-ME problem, we are able to analyze *multi-environment* MDPs and synthesize strategies with guarantees against all considered environments [21,20].

References

1. C. Baier and J.-P. Katoen. *Principles of model checking*. MIT Press, 2008.
2. D. P. Bertsekas and J. N. Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.

3. T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege. To reach or not to reach? efficient algorithms for total-payoff games. *CoRR*, abs/1407.5030, 2014.
4. V. Bruyère, E. Filiot, M. Randour, and J.-F. Raskin. Expectations or guarantees? I want it all! A crossroad between games and MDPs. In *Proc. of SR*, EPTCS 146, pages 1–8, 2014.
5. V. Bruyère, E. Filiot, M. Randour, and J.-F. Raskin. Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games. In *Proc. of STACS*, LIPIcs 25, pages 199–213. Schloss Dagstuhl - LZI, 2014.
6. K. Chatterjee, M. Chmelik, and M. Tracol. What is decidable about partially observable Markov decision processes with omega-regular objectives. In *Proc. of CSL*, LIPIcs 23. Schloss Dagstuhl - LZI, 2013.
7. K. Chatterjee, L. Doyen, M. Randour, and J.-F. Raskin. Looking at mean-payoff and total-payoff through windows. In *Proc. of ATVA*, LNCS 8172, pages 118–132. Springer, 2013.
8. B. V. Cherkassky, A. V. Goldberg, and T. Radzik. Shortest paths algorithms: Theory and experimental evaluation. *Math. programming*, 73(2):129–174, 1996.
9. C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, 1995.
10. L. de Alfaro. Computing minimum and maximum reachability times in probabilistic systems. In *Proc. of CONCUR*, LNCS 1664, pages 66–81. Springer, 1999.
11. F. E., R. Gentilini, and J.-F. Raskin. Quantitative languages defined by functional automata. In *Proc. of CONCUR*, LNCS 7454, pages 132–146. Springer, 2012.
12. A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8:109–113, 1979.
13. K. Etessami, M. Z. Kwiatkowska, M. Y. Vardi, and M. Yannakakis. Multi-objective model checking of Markov decision processes. *LMCS*, 4(4), 2008.
14. V. Forejt, M. Kwiatkowska, G. Norman, D. Parker, and H. Qu. Quantitative multi-objective verification for probabilistic systems. In *Proc. of TACAS*, LNCS 6605, pages 112–127. Springer, 2011.
15. C. Haase and S. Kiefer. The odds of staying on budget. *CoRR*, abs/1409.8228, 2014.
16. L. Khachiyan, E. Boros, K. Borys, K. M. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao. On short paths interdiction problems: Total and node-wise limited interdiction. pages 204–233, 2008.
17. Y. Ohtsubo. Optimal threshold probability in undiscounted markov decision processes with a target set. *Applied Math. and Computation*, 149(2):519 – 532, 2004.
18. M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.
19. M. Randour, J.-F. Raskin, and O. Sankur. Percentile queries in multi-dimensional Markov decision processes. *CoRR*, abs/1410.4801, 2014.
20. J.-F. Raskin and O. Sankur. Multiple-environment Markov decision processes. *CoRR*, abs/1405.4733, 2014.
21. J.-F. Raskin and O. Sankur. Multiple-environment Markov decision processes. In *Proc. of FSTTCS*, LIPIcs. Schloss Dagstuhl - LZI, 2014.
22. M. Sakaguchi and Y. Ohtsubo. Markov decision processes associated with two threshold probability criteria. *Journal of Control Theory and Applications*, 11(4):548–557, 2013.
23. M. Ummels and C. Baier. Computing quantiles in Markov reward models. In *Proc. of FOSSACS*, LNCS 7794, pages 353–368. Springer, 2013.
24. M. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *Proc. of FOCS*, pages 327–338. IEEE Computer Society, 1985.