# Expectation Optimization with Probabilistic Guarantees in POMDPs with Discounted-sum Objectives[*]

**Krishnendu Chatterjee[1], Adrián Elgyütt[1], Petr Novotný[1], Owen Rouillé[2]**
[1] Institute of Science and Technology Austria, Klosterneuburg, Austria
[2] École Normale Supérieure de Rennes, Rennes, France
{Krishnendu.Chatterjee, adrian.elgyuett, petr.novotny}@ist.ac.at, owen.rouille@ens-rennes.fr

## Abstract

Partially-observable Markov decision processes (POMDPs) with discounted-sum payoff are a standard framework to model a wide range of problems related to decision making under uncertainty. Traditionally, the goal has been to obtain policies that optimize the expectation of the discounted-sum payoff. A key drawback of the expectation measure is that even low probability events with extreme payoff can significantly affect the expectation, and thus the obtained policies are not necessarily risk-averse. An alternate approach is to optimize the probability that the payoff is above a certain threshold, which allows obtaining risk-averse policies, but ignores optimization of the expectation. We consider the expectation optimization with probabilistic guarantee (EOPG) problem, where the goal is to optimize the expectation ensuring that the payoff is above a given threshold with at least a specified probability. We present several results on the EOPG problem, including the first algorithm to solve it.

## 1 Introduction

*POMDPs and Discounted-Sum Objectives.* Decision making under uncertainty is a fundamental problem in artificial intelligence. Markov decision processes (MDPs) are the *de facto* model that allows both decision-making choices as well as stochastic behavior [Howard, 1960; Puterman, 2005]. The extension of MDPs with uncertainty about information gives rise to the model of partially-observable Markov decision processes (POMDPs) [Littman, 1996; Papadimitriou and Tsitsiklis, 1987]. POMDPs are used in a wide range of areas, such as planning [Russell and Norvig, 2010], reinforcement learning [Kaelbling *et al.*, 1996], robotics [Kress-Gazit *et al.*, 2009; Kaelbling *et al.*, 1998], to name a few. In decision making under uncertainty, the objective is to optimize a payoff function. A classical and basic payoff function is the *discounted-sum payoff*, where every transition of the POMDP is assigned a reward, and for an infinite path (that consists of an infinite sequence of transitions) the payoff is the discounted-sum of the rewards of the transitions.

*Expectation Optimization and Drawback.* Traditionally, POMDPs with discounted-sum payoff have been studied, where the goal is to obtain policies that optimize the expected payoff. A key drawback of the expectation optimization is that it is not robust with respect to risk measures. For example, a policy $\sigma_1$ that achieves with probability $1/100$ payoff $10^4$ and with the remaining probability payoff 0 has higher expectation than a policy $\sigma_2$ that achieves with probability $99/100$ payoff $100$ and with the remaining probability payoff 0. However the second policy is more robust and less risk-prone, and is desirable in many scenarios.

*Probability Optimization and Drawback.* Due to the drawback of expectation optimization, there has been recent interest to study the optimization of the probability to ensure that the payoff is above a given threshold [Hou *et al.*, 2016]. While this ensures risk-averse policies, it ignores the expectation optimization.

*Expectation Optimization with Probabilistic Guarantee.* A formulation that retains the advantages of both the above optimization criteria, yet removes the associated drawbacks, is as follows: given a payoff threshold $\tau$ and risk bound $\alpha$, the objective is the expectation maximization w.r.t. to all policies that ensure the payoff is at least $\tau$ with probability at least $1-\alpha$. We study this expectation optimization with probabilistic guarantee (EOPG) problem for discounted-sum POMDPs.

*Motivating Examples.* We present some motivating examples for the EOPG formulation.

- *Bad events avoidance.* Consider planning under uncertainty (e.g., self-driving cars) where certain events are dangerous (e.g., the distance between two cars less than a specified distance), and it must be ensured that such events happen with low probability. Thus, desirable policies aim to maximize the expected payoff, ensuring the avoidance of bad events with a specified high probability.
- *Gambling.* In gambling, while the goal is to maximize the expected profit, a desirable risk-averse policy would ensure that the probability loss is less than a given risk.

[*]This research was supported by the Vienna Science and Technology Fund (WWTF) grant ICT15-003; Austrian Science Fund (FWF): S11407-N23 (RiSE/SHiNE); and an ERC Start Grant (279307: Graph Games).

Thus, the EOPG problem for POMDPs with discounted-sum payoff is an important problem which we consider.

*Previous Results.* Several related problems have been considered, and two most relevant works are the following:

1. *Chance-constrained (CC) problem.* In the CC problem, certain bad states of the POMDP must not be reached with some probability threshold. That is, in the CC problem the probabilistic constraint is state-based (some states should be avoided) rather than the execution-based discounted-sum payoff. This problem was considered in [Santana *et al.*, 2016], but only with deterministic policies. As already noted in [Santana *et al.*, 2016], randomized (or mixed) policies are more powerful.
2. *Probability 1 bound.* The special case of the EOPG problem with $\alpha = 0$ has been considered in [Chatterjee *et al.*, 2017]. This formulation represents the case with no risk.

*Our Contributions.* Our main contributions are as follows:

1. *Algorithm.* We present a randomized algorithm for approximating (up to any given precision) the optimal solution to the EOPG problem. This is the first approach to solve the EOPG problem for discounted-sum POMDPs.
2. *Practical approach.* We present a practical approach where certain searches of our algorithm are only performed for a time bound. This gives an *anytime* algorithm which approximates the probabilistic guarantee and then optimizes the expectation.
3. *Experimental results.* We present experimental results of our algorithm on classical POMDPs.

Due to space constraints, details such as full proofs are deferred to the full version of the paper [Chatterjee *et al.*, 2018].

**Related Works.** POMDPs with discounted-sum payoff have been widely studied, both for theoretical results [Papadimitriou and Tsitsiklis, 1987; Littman, 1996] as well as practical tools [Kurniawati *et al.*, 2008; Silver and Veness, 2010; Ye *et al.*, 2017]. Traditionally, expectation optimization has been considered, and recent works consider policies that optimize probabilities to ensure discounted-sum payoff above a threshold [Hou *et al.*, 2016]. Several problems related to the EOPG problem have been considered before: (a) for probability threshold 1 for long-run average and stochastic shortest path problem in fully-observable MDPs [Bruyère *et al.*, 2014; Randour *et al.*, 2015]; (b) for risk bound 0 for discounted-sum payoff for POMDPs [Chatterjee *et al.*, 2017]; and (c) for general probability threshold for long-run average payoff in fully-observable MDPs [Chatterjee *et al.*, 2015b]. The *chance constrained*-optimization was studied in [Santana *et al.*, 2016]. The general EOPG problem for POMDPs with discounted-sum payoff has not been studied before, although development of similar objectives was proposed for perfectly observable MDPs [Defourny *et al.*, 2008]. A related approach for POMDPs is called *constrained POMDPs* [Undurti and How, 2010; Poupart *et al.*, 2015], where the aim is to maximize the expected payoff ensuring that the expectation of some other quantity is bounded. In contrast, in the EOPG problem the constraint is probabilistic rather than an expectation constraint, and as mentioned before, the probabilistic constraint ensures risk-averseness as compared to the expectation constraint. Thus, the constrained POMDPs and the EOPG problem, though related, consider different optimization criteria.

## 2 Preliminaries

Throughout this work, we mostly follow standard (PO)MDP notations from [Puterman, 2005; Littman, 1996]. We denote by $\mathcal{D}(X)$ the set of all probability distributions on a finite set $X$, i.e. all functions $f : X \to [0,1]$ s.t. $\sum_{x\in X} f(x) = 1$.

**Definition 1** ***(POMDPs.)*** *A* POMDP *is a tuple* $P = (S, \mathcal{A}, \delta, r, \mathcal{Z}, \mathcal{O}, \lambda)$ *where* $S$ *is a finite set of* states, $\mathcal{A}$ *is a finite alphabet of* actions, $\delta : S \times \mathcal{A} \to \mathcal{D}(S)$ *is a* probabilistic transition function *that given a state* $s$ *and an action* $a \in \mathcal{A}$ *gives the probability distribution over the successor states,* $r : S \times \mathcal{A} \to \mathbb{R}$ *is a reward function,* $\mathcal{Z}$ *is a finite set of* observations, $\mathcal{O} : S \to \mathcal{D}(\mathcal{Z})$ *is a probabilistic* observation function *that maps every state to a distribution over observations, and* $\lambda \in \mathcal{D}(S)$ *is the* initial belief. *We abbreviate* $\delta(s,a)(s')$ *and* $\mathcal{O}(s)(o)$ *by* $\delta(s'|s,a)$ *and* $\mathcal{O}(o|s)$*, respectively.*

**Plays & Histories.** A *play* (or an infinite path) in a POMDP is an infinite sequence $\rho = s_0 a_1 s_1 a_1 s_2 a_2 \dots$ of states and actions s.t. $s_0 \in \mathrm{Supp}(\lambda)$ and for all $i \geq 0$ we have $\delta(s_{i+1} \mid s_{i-1}, a_i) > 0$. A *finite path* (or just *path*) is a finite prefix of a play ending with a state, i.e. a sequence from $(S \cdot \mathcal{A})^* \cdot S$. A *history* is a finite sequence of actions and observations $h = a_1 o_1 \dots a_{i-1} o_i \in (\mathcal{A} \cdot \mathcal{Z})^*$ s.t. there is a path $w = s_0 a_1 s_1 \dots a_i s_i$ with $\mathcal{O}(o_j \mid s_j) > 0$ for each $1 \leq j \leq i$. We write $h = H(w)$ to indicate that history $h$ corresponds to a path $w$. The *length* of a path (or history) $w$, denoted by $len(w)$, is the number of actions in $w$, and the length of a play $\rho$ is $len(\rho) = \infty$.

**Discounted Payoff.** Given a play $\rho = s_0 a_1 s_1 a_2 s_2 a_3 \dots$ and a discount factor $0 \leq \gamma < 1$, the *finite-horizon discounted payoff* of $\rho$ for horizon $N$ is $\mathsf{Disc}_{\gamma,N}(\rho) = \sum_{i=0}^{N} \gamma^i \cdot r(s_i, a_{i+1})$. The *infinite-horizon discounted payoff* $\mathsf{Disc}_\gamma$ of $\rho$ is $\mathsf{Disc}_\gamma(\rho) = \sum_{i=0}^{\infty} \gamma^i \cdot r(s_i, a_{i+1})$.

**Policies.** A *policy* (or *strategy*) is a blueprint for selecting actions based on the past history. Formally, it is a function $\sigma$ which assigns to a history a probability distribution over the actions, i.e. $\sigma(h)(a)$ is the probability of selecting action $a$ after observing history $h$ (we abbreviate $\sigma(h)(a)$ to $\sigma(a \mid h)$). A policy is *deterministic* if for each history $h$ the distribution $\sigma(\cdot \mid h)$ selects a single action with probability 1. For $\sigma$ deterministic we write $\sigma(h) = a$ to indicate that $\sigma(a \mid h) = 1$.

**Beliefs.** A *belief* is a distribution on states (i.e. an element of $\mathcal{D}(S)$) indicating the probability of being in each particular state given the current history. The initial belief $\lambda$ is given as a part of the POMDP. Then, in each step, when the history observed so far is $h$, the current belief is $b_h$, an action $a \in \mathcal{A}$ is played, and an observation $o \in \mathcal{Z}$ is received, the updated belief $b_{h'}$ for history $h' = hao$ can be computed by a standard Bayesian formula [Cassandra, 1998].

**Expected Value of a Policy.** Given a POMDP $P$, a policy $\sigma$, a horizon $N$, and a discount factor $\gamma$, the *expected value* of $\sigma$ from $\lambda$ is the expected value of the infinite-horizon discounted sum under policy $\sigma$ when starting in a state sampled from the initial belief of $\lambda$ of $P$: $eVal(\sigma) = \mathbb{E}^{\sigma}_{\lambda}[\mathsf{Disc}_{\gamma,N}]$.

**Risk.** A *risk level* $rl(\sigma, \tau, \mathsf{Disc}_{\gamma,N})$ of a policy $\sigma$ at threshold $\tau \in \mathbb{R}$ w.r.t. payoff function $\mathsf{Disc}_{\gamma,N}$ is the probability that the payoff of a play generated by $\sigma$ is below $\tau$, i.e.

$$rl(\sigma, \tau, \mathsf{Disc}_{\gamma,N}) = \mathbb{P}^{\sigma}_{\lambda}(\mathsf{Disc}_{\gamma,N} < \tau).$$

**EOPG Problem.** We now define the problem of *expectation optimization with probabilistic guarantees* (the *EOPG* problem for short). We first define a finite-horizon variant, and then discuss the infinite-horizon version in Section 3. In EOPG problem, we are given a threshold $\tau \in \mathbb{R}$, a risk bound $\alpha$, and a horizon $N$. A policy $\sigma$ is a *feasible solution* of the problem if $rl(\sigma, \tau, \mathsf{Disc}_{\gamma,N}) \leq \alpha$. The goal of the EOPG problem is to find a feasible solution $\sigma$ maximizing $eVal(\sigma)$ among all feasible solutions, provided that feasible solutions exist.

**Observable Rewards.** We solve the EOPG problem under the assumption that rewards in the POMDP are *observable*. This means that $r(s, a) = r(s', a)$ whenever $\mathcal{O}(s) = \mathcal{O}(s')$ or if both $s$ and $s'$ have a positive probability under the initial belief. This is a natural assumption satisfied by many standard benchmarks [Hou *et al.*, 2016; Chatterjee *et al.*, 2015a]. At the end of Section 4 we discuss how could be our results extended to unobservable rewards.

**Efficient Algorithms.** A standard way of making POMDP planning more efficient is to design an algorithm that is *online* (i.e., it computes a local approximation of the $\varepsilon$-optimal policy, selecting the best action for the current belief [Ross *et al.*, 2008]) and *anytime,* i.e. computing better and better approximation of the $\varepsilon$-optimal policy over its runtime, returning a solution together with some guarantee on its quality if forced to terminate early.

## 3 Relationship to CC-POMDPs

We present our first result showing that an approximate infinite-horizon (IH) EOPG problem can be reduced to a finite-horizon variant. While similar reductions are natural when dealing with discounted payoff, for the EOPG problem the reduction is somewhat subtle due to the presence of the risk constraint. We then show that the EOPG problem can be reduced to chance-constrained POMDPs and solved using the RAO* algorithm [Santana *et al.*, 2016], but we also present several drawbacks of this approach.

Formally, we define the IH-EOPG problem as follows: we are given $\tau$ and $\alpha$ as before and in addition, an error term $\varepsilon$. We say that an algorithm $\varepsilon$-solves the IH-EOPG problem if, whenever the problem has a feasible solution (feasibility is defined as before, with $\mathsf{Disc}_{\gamma,N}$ replaced by $\mathsf{Disc}_{\gamma}$), the algorithm finds a policy $\sigma$ s.t. $rl(\sigma, \tau - \varepsilon, \mathsf{Disc}_{\gamma}) \leq \alpha$ and $\mathbb{E}^{\sigma}_{\lambda}[\mathsf{Disc}_{\gamma}] \geq rVal(\tau, \alpha) - \varepsilon$, where

$$rVal(\tau, \alpha) = \sup\{\mathbb{E}^{\pi}_{\lambda}[\mathsf{Disc}_{\gamma}] \mid rl(\pi, \tau, \mathsf{Disc}_{\gamma}) \leq \alpha\}.$$

**Infinite to Finite Horizon.** Let $P$ be a $\gamma$-discounted POMDP, $\tau$ a payoff threshold, $\alpha \in [0, 1]$ a risk bound, and $\varepsilon > 0$ an error term. Let $N(\varepsilon)$ be a horizon such that the following holds: $\gamma^{N(\varepsilon)} \cdot |\max\{0, r_{\max}\} - \min\{0, r_{\min}\}| \leq (1 - \gamma) \cdot \frac{\varepsilon}{2}$, where $r_{\max}$ and $r_{\min}$ are the maximal and minimal rewards appearing in $P$, respectively.

**Lemma 1** *If there exists a feasible solution of the IH-EOPG problem with threshold $\tau$ and risk bound $\alpha$. Then there exists a policy $\sigma$ satisfying $rl(\sigma, \tau - \frac{\varepsilon}{2}, \mathsf{Disc}_{\gamma,N(\varepsilon)}) \leq \alpha$. Moreover, let $\sigma$ be an optimal solution to the EOPG problem with the risk bound $\alpha$, horizon $N(\varepsilon)$, and with threshold $\tau' = \tau - \frac{\varepsilon}{2}$. Then $\sigma$ is an $\varepsilon$-optimal solution to the IH-EOPG problem.*

The previous lemma effectively shows that to solve an approximate version of the EOPG problem, it suffices to solve its finite horizon version.

**Chance-Constrained POMDPs.** In the chance constrained (CC) optimization problem [Santana *et al.*, 2016], we are given a POMDP $P$, a finite-horizon bound $N \in \mathbb{N}$, and a set of constraint-violating states $X$, which is a subset of the set of states of $P$. We are also given a *risk bound* $\Delta$. The goal is to optimize the expected finite-horizon payoff, i.e. the expectation of the following random variable:

$$\mathsf{Payoff}_N(s_0 a_1 s_1 a_2 s_2 \ldots) = \textstyle\sum_{i=0}^{N} r(s_i, a_{i+1}).$$

The optimization is subject to a constraint that the probability of entering a state from $C$ (so-called *execution risk*) stays below the risk bound $\Delta$.

**From EOPGs to CC-POMDPs.** We sketch how the FH-EOPG relates to CC-POMDP optimization. In the EOPG problem, the constraint violation occurs when the finite-horizon discounted payoff in step $N$ is smaller than a threshold $\tau$. To formulate this in a CC-POMDP setting, we need to make the constraint violation a property of a *state* of the POMDP. Hence, we construct a new POMDP $P'$ with an extended state space: the states of $P'$ are triples of the form $\tilde{t} = (s, i, x)$, where $s$ is a state of the original POMDP $P$, $0 \leq i \leq N(\varepsilon)$ is a time index, and $x \in \mathbb{R}$ is a number representing the discounted reward accumulated before reaching the state $\tilde{t}$. The remaining components of $P'$ are then extended in a natural way from $P$. By solving the CC-POMDP problem for $P'$, where the set $X$ contains extended states $(s, N, x)$ where $x < \tau$, we obtain a policy in $P'$ which can be carried back to $P$ where it forms an optimal solution of the FH-EOPG problem.

**Discussion of the CC-POMDP Approach.** It follows that we could, in principle, reduce the EOPG problem to CC-POMDP optimization and then solve the latter using the known RAO* algorithm [Santana *et al.*, 2016]. However, there are several issues with this approach.

First, RAO* aims to find an optimal *deterministic* policy in CC-POMDPs. But as already mentioned in [Santana *et al.*, 2016], the optimal solution to the CC-POMDP (and thus also to EOPG) problem might require randomization, and deterministic policies may have arbitrarily worse expected payoff than randomized ones (it is well-known that randomization might be necessary for optimality in constrained (PO)MDPs, see [Feinberg and Shwartz, 1995; Kim *et al.*, 2011; Sprauel *et al.*, 2014]).

Second, although RAO* converges to an optimal constrained deterministic policy, it *does not* provide *anytime* guarantees about the risk of the policy it constructs. RAO* is an AO*-like algorithm that iteratively searches the belief space and in each step computes a *greedy policy* that is optimal on the already explored fragment of the belief space. During its execution, RAO* works with an *under-approximation* of a

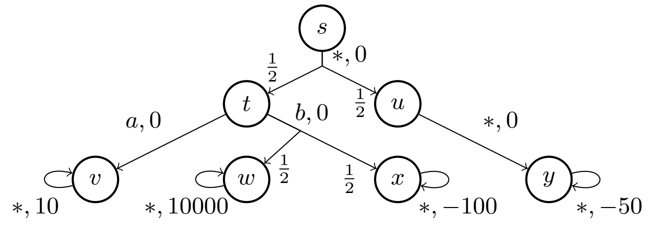

Figure 1: A perfectly observable MDP with actions $a$, $b$ and with a discount factor $\gamma = \frac{1}{2}$. The star denotes arbitrary action, and immediate reward of an action in a state is given next to the action label.

risk taken by the greedy policy: this is because an optimal risk to be taken in belief states that were not yet explored is under-approximated by an *admissible heuristic*. So if the algorithm is stopped prematurely, the actual risk taken by the current greedy policy can be much larger than indicated by the algorithm. This is illustrated in the following example.

**Example 1** *Consider the MDP in Figure 1, and consider $\tau = 1$ and $\alpha = \frac{2}{3}$; that is, in the chance-constrained reformulation, we seek an optimal policy for which the probability of hitting $x$ and $y$ is at most $\frac{2}{3}$. Consider the execution of RAO* which explores states $s$, $t$, $v$, $w$, and $x$ (since the MDP is perfectly observable, we work directly with states) and then is prematurely terminated. (The order in which unexplored nodes are visited is determined by a heuristic, and in general we cannot guarantee that $u$ is explored earlier in RAO*'s execution). At this moment, the risk taken by an optimal policy from $u$ is under-approximated using an admissible heuristic. In case of using a* myopic *heuristic, as suggested in [Santana* et al.*, 2016], the risk from $u$ is under-approximated by $0$. Hence, at this moment the best greedy policy satisfying the risk constraint is the one which plays action $b$ in state $t$: the risk-estimate of this policy is $\frac{1}{4} < \frac{2}{3}$ (the probability of reaching $x$). However, any deterministic policy that selects $b$ in state $t$ takes an overall risk $\frac{3}{4} > \frac{2}{3}$.*

## 4 Risk-Aware POMCP

The previous example illustrates the main challenge in designing an online and anytime algorithm for the (finite horizon) EOPG problem: we need to keep upper bounds on the minimal risk achievable in the POMDP. Initially, the upper bound is 1, and to decrease it, we need to discover a sufficiently large probability mass of paths that yield payoff above $\tau$.

We propose an algorithm for the EOPG problem based on the popular POMCP [Silver and Veness, 2010] planning algorithm: the *risk-aware POMCP* (or RAMCP for short). RAMCP solves the aforementioned challenge by performing, in each decision step, a large number of simulations using the POMCP heuristic to explore promising histories first. The key feature of RAMCP is that it extends POMCP with a new data structure, so called *explicit tree*, which contains those histories explored during simulations that have payoff above the required threshold $\tau$. The explicit tree allows us to keep track of the upper bound on the risk that needs to be taken

**Algorithm 1:** RAMCP.

```
  global $thr, rbound, n, \mathcal{T}_{srch}, \mathcal{T}_{exp}$
1 $thr \leftarrow \tau$; $rbound \leftarrow \alpha$; $n \leftarrow N$;
2 $\mathcal{T}_{srch} \leftarrow \mathcal{T}_{exp} \leftarrow$ empty history $\epsilon$
3 initialize $\epsilon.U = 1$
4 while $n > 0$ do
5     Explore($n$)
6     SelectAction()
7     $n \leftarrow n - 1$
  procedure Explore($n$)
8     $h \leftarrow$ the root of $\mathcal{T}_{srch}$; $s \leftarrow$ sample from $b_h$
9     while not timeout do Simulate($s, h, n, 0$)
```

from the initial belief. After the simulation phase concludes, RAMCP uses the explicit tree to construct a perfectly observable tree-shaped *constrained MDP* [Altman, 1999] encoding the EOPG problem on the explored fragment of the history tree of the input POMDP. The optimal distribution on actions is then computed using a linear program for constrained MDP optimization [Altman, 1999]. In the rest of this section, we present details of the algorithm and formally state its properties. In the following, we fix a POMDP $P$, a horizon $N$, a threshold $\tau$ and a risk bound $\alpha$.

**RAMCP.** The main loop of RAMCP is pictured in Algorithm 1. In each decision step, RAMCP performs a *search* phase followed by *action selection* followed by *playing* the selected action (the latter two performed within SelectAction procedure). We describe the three phases separately.

**RAMCP: Search Phase.** The search phase is shown in Algorithms 1 and 2. We first introduce the data structures the algorithm works with, then the elements of these structures, and finally we sketch how the search phase executes.

*Data Structures.* In the search phase, RAMCP explores, by performing simulations, the *history tree* $\mathcal{T}_P$ of the input POMDP $P$. Nodes of the tree are the histories of $P$ of length $\leq N$. The tree is rooted in the empty history, and for each history $h$ of length at most $N-1$, each action $a$, and observation $o$, the node $h$ has a child $hao$. RAMCP works with two data structures, that are both sub-trees of $\mathcal{T}_P$: a search tree $\mathcal{T}_{srch}$ and explicit tree $\mathcal{T}_{exp}$. Intuitively, $\mathcal{T}_{srch}$ corresponds to the standard POMCP search tree while $\mathcal{T}_{exp}$ is a sub-tree of $\mathcal{T}_{srch}$ containing histories leading to payoff above $\tau$. The term "explicit" stems from the fact that we explicitly compute beliefs and transition probabilities for the nodes in $\mathcal{T}_{exp}$. Initially (before the first search phase), both $\mathcal{T}_{srch}$ and $\mathcal{T}_{exp}$ contain a single node: empty history.

*Elements of Data Structures.* Each node $h$ of $\mathcal{T}_{srch}$ has these attributes: for each action $a$ there is $h.V_a$, the average expected payoff obtained from the node $h$ after playing $a$ during past simulations, and $h.N_a$, the number of times action $a$ was selected in node $h$ in past simulations. Next, we have $h.N$, the number of times the node was visited during past simulations. Each node $h$ of $\mathcal{T}_{srch}$ also contains a particle-filter approximation of the corresponding belief $b_h$. A node $h$ of the explicit tree has an attribute $h.U$, the upper bound on the risk from belief $b_h$, and, for each action $a$, attribute $h.U_a$, the upper bound on the risk when playing $a$ from belief $b_h$. Also, each node

**Algorithm 2:** RAMCP simulations.

```
procedure Simulate ($s, h, depth, pay$)
    if $depth = 0$ then
        if $pay \geq thr$ then UpdateTrees ($h$)
        return $0$
    if $h \notin \mathcal{T}_{srch}$ then
        add $h$ to $\mathcal{T}_{srch}$
        return Rollout ($s, h, depth, pay$)
    $a \leftarrow \arg\max_a h.V_a + K \cdot \sqrt{\frac{\log h.N}{h.N_a}}$
    sample $(s', o, r)$ from $(\delta(\cdot \mid s, a), \mathcal{O}(\cdot \mid s'), r(s, a))$
    $R \leftarrow r + \gamma \cdot$Simulate ($s', hao, depth - 1, pay + r$)
    $(h.N, h.N_a) \leftarrow (h.N + 1, h.N_a + 1)$
    $h.V_a \leftarrow h.V_a + \frac{R - h.V_a}{h.N_a}$
    return $R$

procedure Rollout ($s, h, depth, pay$)
    if $depth = 0$ then
        if $pay \geq thr$ then UpdateTrees ($h$)
        return $0$
    choose $a \in \mathcal{A}$ uniformly at random
    sample $(s', o, r)$ from $(\delta(\cdot \mid s, a), \mathcal{O}(\cdot \mid s'), r(s, a))$
    return
      $r + \gamma \cdot$Rollout ($s', hao, depth - 1, pay + r$)

procedure UpdateTrees ($h$)
    $h_{old} \leftarrow$ shortest prefix of $h$ not in $\mathcal{T}_{exp}$
    for $g$ prefix of $h$ and strict extension of $h_{old}$ do
        add $g$ to $\mathcal{T}_{exp}$
        let $g = \hat{g}ao$; add edge $(\hat{g}, g)$ to $\mathcal{T}_{exp}$
        compute $p(\hat{g}, g)$ and $rew(\hat{g}, g)$
    $g \leftarrow h$; $R \leftarrow 0$
    $h.U \leftarrow 0$
    repeat
        let $g = \hat{g}ao$
        foreach $a \in \mathcal{A}$ do
            $\hat{g}.U_a \leftarrow 1 - \sum_{o \in \mathcal{Z}} p(\hat{g}, \hat{g}ao) \cdot (1 - \hat{g}ao.U)$
        $\hat{g}.U \leftarrow \min_{a \in \mathcal{A}} \hat{g}.U_a$
        if $g \notin \mathcal{T}_{srch}$ then
            add $g$ to $\mathcal{T}_{srch}$
            $g.N \leftarrow 1$; $g.N_a \leftarrow 1$; $R \leftarrow R + rew(\hat{g}, g)$
            $g.V_a \leftarrow R$
        $g \leftarrow \hat{g}$
    until $g =$ empty history
```

of $\mathcal{T}_{exp}$ contains an exact representation of the corresponding belief, and each edge $(h, hao)$ of the explicit tree is labelled by numbers $p(h, hao) = \sum_{s,s' \in S} b_h(s) \cdot \delta(s' \mid s, a) \cdot \mathcal{O}(o \mid s')$, i.e. by the probability of observing $o$ when playing action $a$ after history $h$, and $rew(h, hao) = rew(h, a)$, where $rew(h, a)$ is equal to $r(s, a)$ for any state $s$ with $b_h(s) > 0$ (here we use the facts that rewards are observable).

*Execution of Search Phase.* Procedures Simulate and Rollout are basically the same as in POMCP—within the search tree we choose actions heuristically (in line 7 of Simulate, the number $K$ is POMCP's exploration constant), outside of it we choose actions uniformly at random. However, whenever a simulation succeeds in surpassing the threshold $\tau$, we add the observed history and all its prefixes to both the explicit and search trees (procedure UpdateTrees). Note that computing $p(\hat{g}, g)$ on line 5 entails computing full Bayesian updates on the path from $h_{old}$ to $h$ so as to compute exact beliefs of the corresponding nodes. Risk bounds for the nodes corresponding to prefixes of $h$ are updated accordingly using a standard dynamic programming update (lines 9–12), starting from the newly added leaf whose risk is 0 (as it corresponds to a history after which $\tau$ is surpassed). We have the following:

**Lemma 2** *1. At any point there exists a policy $\sigma$ such that $\mathbb{P}^{\sigma}_{b_{h_{root}}}(\mathsf{Disc}_{\gamma, N - len(h_{root})} < thr) \leq h_{root}.U$.*

*2. As timeout $\to \infty$, the probability that $h_{root}.U$ becomes equal to $\inf_\sigma \mathbb{P}^{\sigma}_{b_{h_{root}}}(\mathsf{Disc}_{\gamma, N - len(h_{root})} < thr)$ before timeout expires converges to* 1.

*Proof (sketch).* For part (1.) we prove the following stronger statement: Fix any point of algorithm's execution, and let $L$ be the length of $h_{root}$ (the history at the root of $\mathcal{T}_{srch}$ and $\mathcal{T}_{exp}$) at this point. Then for any node $f$ of $\mathcal{T}_{exp}$ there exists a policy $\sigma$ s.t. $\mathbb{P}^{\sigma}_{b_f}(\mathsf{Disc}_{\gamma, len(f) - N} < (thr - (\mathsf{Disc}_{\gamma,N}(f) - \mathsf{Disc}_{\gamma,N}(h_{root})) \cdot \gamma^{-len(f)}) / \gamma^{len(f) - L}) \leq f.U$. The proof proceeds by a rather straightforward induction. The statement of the lemma then follows by plugging the root of $\mathcal{T}_{exp}$ into $f$.

For part (2.), the crucial observation is that as *timeout* $\to \infty$, with probability converging to 1 the tree $\mathcal{T}_{exp}$ will at some point contain all histories of length $N$ (that have $h_{root}$ as a prefix) whose payoff is above the required threshold. It can be easily shown that at such a point $h_{root}.U = \inf_\sigma \mathbb{P}^{\sigma}_{b_{h_{root}}}(\mathsf{Disc}_{\gamma, N - len(h_{root})} < thr)$, and $h_{root}.U$ will not change any further. □

**RAMCP: Action Selection.** The action selection phase is sketched in Algorithm 3. If the current risk bound is 1, there is no real constraint and we select an action maximizing the expected payoff. Otherwise, to compute a distribution on actions to select, we construct and solve a certain constrained MDP.

*Constructing Constrained MDP.* RAMCP first computes a *closure* $\hat{\mathcal{T}}_{exp}$ of $\mathcal{T}_{exp}$. That is, first we set $\hat{\mathcal{T}}_{exp} \leftarrow \mathcal{T}_{exp}$ and then for each node $h \in \mathcal{T}_{exp}$ and each action $a$ such that $h$ has a successor of the form $hao \in \mathcal{T}_{exp}$ (in such a case, we say that $a$ is *allowed* in $h$), the algorithm checks if there exists a successor of the form $hao'$ that is not in $\mathcal{T}_{exp}$; all such "missing" successors of $h$ under $a$ are added to $\hat{\mathcal{T}}_{exp}$. Such a tree $\hat{\mathcal{T}}_{exp}$ defines a perfectly observable *constrained MDP* $\mathcal{M}$:

- the states of $\mathcal{M}$ are the nodes of $\hat{\mathcal{T}}_{exp}$;
- for each internal node $h$ of $\hat{\mathcal{T}}_{exp}$ and each action $a$ allowed in $h$ there is probability $p(h, hao)$ of transitioning from $h$ to $hao$ under $a$ (these probabilities sum up to 1 for each $h$ and $a$ thanks to computing the closure). If $h$ is a leaf of $\hat{\mathcal{T}}_{exp}$ and $len(h) < N$, playing any action $a$ in $h$ leads with probability 1 to a new sink state *sink* (*sink* has self-loops under all actions).

- Rewards in $\mathcal{M}$ are given by the function $rew$; self-loop on the sink and state-action pairs of the form $(h, a)$ with $len(h) = N$ have reward 0. Transitions from the other leaf nodes $h$ to the sink state have reward $\max_a h.V_a$. That is, from nodes that were never explored explicitly (and thus have $U$-attribute equal to 1) we estimate the optimal payoff by previous POMCP simulations.
- We also have a constraint function $C$ assigning penalties to state-action pairs: $C$ assigns $1/\gamma^{N-len(h_{root})}$ to pairs $(h, a)$ such that $h$ is a leaf of $\mathcal{T}_{exp}$ of length $N$, and 0 to all other state-action pairs.

*Solving MDP $\mathcal{M}$.* Using a linear programming formulation of constrained MDPs [Altman, 1999], RAMCP computes a randomized policy $\pi$ in $\mathcal{M}$ maximizing $\mathbb{E}^\pi[\mathsf{Disc}_\gamma]$ under the constraint $\mathbb{E}^\pi[\mathsf{Disc}_\gamma^C] \geq 1 - rbound$, where $\mathsf{Disc}_\gamma^C$ is a discounted sum of incurred penalties. The distribution $\pi$ is then the distribution on actions used by $\pi$ in the first step. An examination of the LP in [Altman, 1999] shows that each solution of the LP yields not only the policy $\pi$, but also for each action $a$ allowed in the root, a *risk vector* $d^a$, i.e. an $|\mathcal{Z}|$-dimensional vector such that $d^a(o) = \mathbb{P}^\pi_{\mathcal{M}}(\mathsf{Disc}_\gamma^C > 0 \mid a \text{ is played and } o \text{ is received})$.

**Remark 1 (Conservative risk minimization.)** *Note that $\mathcal{M}$ might have no policy satisfying the penalty constraint. This happens when the $U$-attribute of the root is greater than rbound. In such a case, the algorithm falls back to a policy that minimizes the risk, which means choosing action $a$ minimizing $root.U_a$ (line 6 of `SelectAction`). In such a case, all $d^a(o)$ are set to zero, to enforce that in the following phases the algorithm behaves conservatively (i.e., keep minimizing the risk).*

**Remark 2 (No feasible solution.)** *When our algorithm fails to obtain a feasible solution, it "silently" falls back to a risk-minimizing policy. This might not be the preferred option for safety-critical applications. However, the algorithm recognizes when it cannot guarantee meeting the original risk-constraint—this happens exactly when at the entry to the `SelectAction` procedure, the $U$-attribute in the root of $\mathcal{T}_{exp}$ is $>$ rbound. Thus, our algorithm has two desirable properties: (a) it can report that it has not obtained a feasible solution; (b) along with that, it presents a risk-minimizing policy.*

**Lemma 3** *Assume that the original EOPG problem has a feasible solution. For a suitable exploration constant $K$, as timeout $\to \infty$, the distribution $d_\pi$ converges, with probability 1, to a distribution on actions used in the first step by some optimal solution to the EOPG problem.*

*Proof (sketch).* Assuming the existence of a feasible solution, we show that at the time point in which the condition in Lemma 2 (2.) holds (such an event happens with probability converging to 1), the constrained MDP $\mathcal{M}$ has a feasible solution. It then remains to prove that the optimal constrained payoff achievable in $\mathcal{M}$ converges to the optimal risk-constrained payoff achievable in $P$. Since rewards in $\mathcal{M}$ are in correspondence with rewards in $\mathcal{T}_{exp}$, it suffices to show that for each leaf $h$ of $\mathcal{T}_{exp}$ with $len(h) < N$ and for each action $a$, the attribute $h.V_a$ converges with probability 1 to the optimal expected payoff for horizon $N - len(h)$ achievable in $P$ after playing action $a$ from belief $b_h$. But since the $V_a$ attributes are updated by POMCP simulations, this follows (for a suitable exploration constant) from the properties of POMCP (Theorem 1 in [Silver and Veness, 2010], see also [Kocsis and Szepesvári, 2006]). □

**Algorithm 3:** RAMCP: action selection and play.

```
procedure SelectAction()
1   if $rbound < 1 \wedge root.U < 1$ then
2       $\mathcal{M} \leftarrow$ constrained MDP determined by $\mathcal{T}_{exp}$
3       if $root.U \leq rbound$ then
4           $d_\pi, \{d^a\}_{a\in\mathcal{A}} \leftarrow$ solve LP formulation of $\mathcal{M}$
5       else
6           $d_\pi, \{d^a\}_{a\in\mathcal{A}} \leftarrow$ solve risk-minimizing
              variant of $\mathcal{M}$
7       $a \leftarrow$ sample from $d_\pi$
8   else
9       $a \leftarrow \arg\max_a root.V_a$
10      $d^a(o) = 1$ for each $o$
11  PlayAction($a, d^a$)

procedure PlayAction($a, d^a$)
1   play action $a$ and receive observation $o$ and reward $R$
2   $thr \leftarrow (thr - R)/\gamma$; $rbound \leftarrow d^a(o)$
3   $h \leftarrow$ root of $\mathcal{T}_{srch}$ (and $\mathcal{T}_{exp}$)
4   $\mathcal{T}_{srch} \leftarrow$ subtree of $\mathcal{T}_{srch}$ rooted in $hao$
5   $\mathcal{T}_{exp} \leftarrow$ subtree of $\mathcal{T}_{exp}$ rooted in $hao$
```

**RAMCP: Playing an Action.** The action-playing phase is shown in Algorithm 3. An action is played in the actual POMDP $P$ and a new observation $o$ and reward $R$ are obtained, and $thr$ and $rbound$ are updated. Then, both the tree data structures are pruned so that the node corresponding to the previous history extended by $a, o$ becomes the new root of the tree. After this, we proceed to the next decision step.

**Theorem 1** *Assume that an EOPG problem instance with a risk bound $\alpha$ and threshold $\tau$ has a feasible solution. As timeout $\to \infty$, the probability that RAMCP returns a payoff smaller than $\tau$ converges to a number $\leq \alpha$. For a suitable exploration constant $K$, the expected return of a RAMCP execution converges to $rVal(\tau, \alpha)$.*

*Proof (sketch).* The proof proceeds by an induction on the length of the horizon $N$, using Lemma 2 (2.) and Lemma 3. □

RAMCP also provides the following anytime guarantee.

**Theorem 2** *Let $u$ be the value of the $U$-attribute of the root of $\mathcal{T}_{exp}$ after the end of the first search phase of RAMCP execution. Then the probability that the remaining execution of RAMCP returns a payoff smaller than $\tau$ is at most $\max\{u, \alpha\}$.*

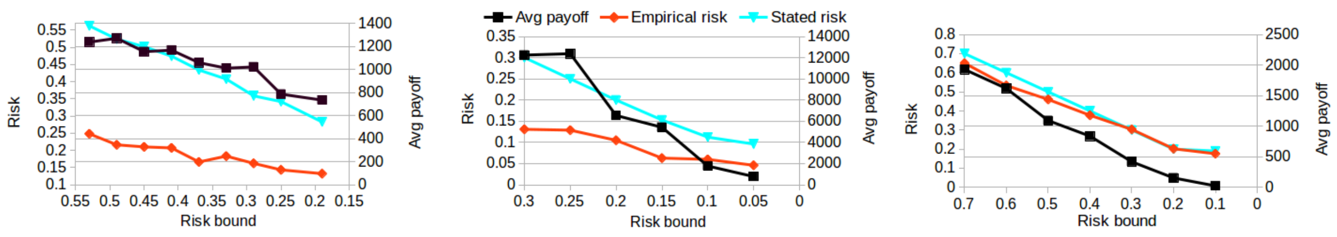

Figure 2: Plots of results obtained from simulating (1.) the larger hallway POMDP benchmark (left), (2.) the MDP hallway benchmark (middle), and (3.) the smaller hallway POMDP benchmark (right). The horizontal axis represents a risk bound $\alpha$.

**Unobservable Rewards.** RAMCP could be adapted to work with unobservable rewards, at the cost of more computations. The difference is in the construction of the constraint function $C$: for unobservable rewards, the same history of observations and actions might encompass both paths that have payoff above the threshold and paths that do not. Hence, we would need to compute the probability of paths corresponding to a given branch that satisfy the threshold condition. This could be achieved by maintaining beliefs over accumulated payoffs.

## 5 Experiments

We implemented RAMCP on top of the POMCP implementation in AI-Toolbox [AI-Toolbox, 2017] and tested on three sets of benchmarks. The first two are the classical Tiger [Kaelbling *et al.*, 1998] and Hallway [Smith and Simmons, 2004] benchmarks naturally modified to contain a risk taking aspect. In our variant of the Hallway benchmark, we again have a robot navigating a grid maze, oblivious to the exact heading and coordinates but able to sense presence of walls on neighbouring cells. Some cells of the maze are *tasks*. Whenever a task cell is entered, the robot attempts to perform the task. When doing this, there is a certain probability of a good outcome, after which a positive reward is gained, as well as a chance of a bad outcome, after which a negative penalty is incurred. There are different types of tasks in the maze with various expected rewards and risks of bad outcomes. Once a task is completed, it disappears from the maze. There are also "traps" that probabilistically spin the robot around. As a third benchmark we consider an MDP variant of the Hallway benchmark. Since the Tiger benchmark is small, we present results for the larger benchmarks. Our implementation and the benchmarks are available on-line.[1]

We ran the benchmarks with different risk thresholds, starting with risk achieved by unconstrained POMCP and progressively decreasing risk until RAMCP no longer finds a feasible solution. For each $\alpha$ we average outcomes of 1000 executions. In each execution, we used a timeout of 5 seconds in the first decision step and 0.1 seconds for the remaining steps. Intuitively, in the first step the agent is allowed a "pre-processing" phase before it starts its operation, trying to explore as much as possible. Once the agent performs the first action, it aims to select actions as fast as possible. We set the exploration constant to $\approx 2 \cdot X$, where $X$ is the difference between largest and smallest undiscounted payoffs achievable in a given instance.

[1]https://git.ist.ac.at/petr.novotny/RAMCP-public

The test configuration was CPU: Intel-i5-3470, 3.20GHz, 4 cores; 8GB RAM; OS: Linux Mint 18 64-bit.

**Discussion.** In Figure 2, we present the results for: (1.) The Hallway POMDP benchmark ($|S| = 67584, |\mathcal{A}| = 3, |\mathcal{Z}| = 43, \gamma = 0.95, N = 30, \tau = 10$); (2.) the perfectly observable version of our Hallway benchmark ($|S| = 4512, |\mathcal{A}| = 3, |\mathcal{Z}| = 1, \gamma = 0.98, N = 120, \tau = 10$); and (3.) smaller POMDP instance of the Hallway benchmark ($|S| = 7680, |\mathcal{A}| = 3, |\mathcal{Z}| = 33, \gamma = 0.8, N = 35, \tau = 10$). In each figure, the $x$ axis represents the risk bound $\alpha$. For each $\alpha$ considered we plot the following quantities: average payoff (secondary $y$ axis), empirical risk (the fraction of trials in which RAMCP returned payoff $< \tau$, primary $y$ axis) and stated risk (the average of $\max\{\alpha, U$-value of the root of $\mathcal{T}_{exp}$ after first search phase$\}$, primary $y$ axis). As expected, the stated risk approximates a lower bound on the empirical risk. Also, as risk bound decreases, the average payoff tends to decrease as well, since the risk bound constraints the agent's behaviour. This trend is somewhat violated in some datapoints: this is because in particular for larger benchmarks, the timeout does not allow for enough exploration so as to converge to a tight approximation of the optimal policy. The main bottleneck is the usage of exact belief updates within the explicit tree. An interesting direction for the future is to replace these updates with a particle-filter approximation (in line with POMCP) and thus improve the speed in exchange for weaker theoretical guarantees. Still, already the current version of RAMCP demonstrates the ability to perform risk vs. expectation trade-off in POMDP planning.

**Comparison with Deterministic Policies.** We also ran experiments related to computation of deterministic policies (by computing, in action selection phase, an optimal deterministic policy in the constrained MDP $\mathcal{M}$, which entails solving a MILP problem). For instance, in benchmark (1.) for $\alpha = 0.41$ the deterministic policy yields expected payoff 645.017 compared to 1166.8 achieved by randomized policy. In benchmark (3.) with $\alpha = 0.3$ we have expected payoff 107.49 for deterministic vs. 695.81 for randomized policies.

## 6 Conclusion

In this work, we studied the expected payoff optimization with probabilistic guarantees in POMDPs. We introduced an online algorithm with anytime risk guarantees for the EOPG problem, implemented this algorithm, and tested it on variants of classical benchmarks. Our experiments show that our algorithm, RAMCP, is able to perform risk-averse planning in POMDPs.

## References

[AI-Toolbox, 2017] AI-Toolbox. AI-Toolbox [Computer Software]. https://github.com/Svalorzen/AI-Toolbox, 2017.

[Altman, 1999] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[Bruyère *et al.*, 2014] Véronique Bruyère, Emmanuel Filiot, Mickael Randour, and Jean-François Raskin. Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games. In *STACS*, pages 199–213. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014.

[Cassandra, 1998] A.R. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, 1998.

[Chatterjee *et al.*, 2015a] K. Chatterjee, M. Chmelik, R. Gupta, and A. Kanodia. Optimal cost almost-sure reachability in POMDPs. In *AAAI*. AAAI Press, 2015.

[Chatterjee *et al.*, 2015b] Krishnendu Chatterjee, Zuzana Komárková, and Jan Kretínský. Unifying two views on multiple mean-payoff objectives in Markov decision processes. In *LICS*, pages 244–256. IEEE Computer Society, 2015.

[Chatterjee *et al.*, 2017] Krishnendu Chatterjee, Petr Novotný, Guillermo A. Pérez, Jean-François Raskin, and Dorde Zikelic. Optimizing expectation with guarantees in POMDPs. In *AAAI*, pages 3725–3732. AAAI Press, 2017.

[Chatterjee *et al.*, 2018] K. Chatterjee, A. Elgyütt, P. Novotný, and O. Rouillé. Expectation optimization with probabilistic guarantees in POMDPs with discounted-sum objectives. *CoRR*, abs/1804.10601, 2018.

[Defourny *et al.*, 2008] Boris Defourny, Damien Ernst, and Louis Wehenkel. Risk-aware decision making and dynamic programming. Technical report, 2008.

[Feinberg and Shwartz, 1995] Eugene A Feinberg and Adam Shwartz. Constrained markov decision models with weighted discounted rewards. *Mathematics of Operations Research*, 20(2):302–320, 1995.

[Hou *et al.*, 2016] Ping Hou, William Yeoh, and Pradeep Varakantham. Solving risk-sensitive POMDPs with and without cost observations. In *AAAI*, pages 3138–3144. AAAI Press, 2016.

[Howard, 1960] H. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[Kaelbling *et al.*, 1996] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[Kaelbling *et al.*, 1998] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[Kim *et al.*, 2011] Dongho Kim, Jaesong Lee, Kee-Eung Kim, and Pascal Poupart. Point-based value iteration for constrained POMDPs. In *IJCAI*, pages 1968–1974, 2011.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit Based Monte-Carlo Planning. In *ECML*, volume 4212 of *LNCS*, pages 282–293. Springer, 2006.

[Kress-Gazit *et al.*, 2009] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics*, 25(6):1370–1381, 2009.

[Kurniawati *et al.*, 2008] H. Kurniawati, D. Hsu, and W.S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*, pages 65–72, 2008.

[Littman, 1996] M. L. Littman. *Algorithms for Sequential Decision Making*. PhD thesis, Brown University, 1996.

[Papadimitriou and Tsitsiklis, 1987] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of Markov decision processes. *Math. Oper. Res.*, 12:441–450, 1987.

[Poupart *et al.*, 2015] Pascal Poupart, Aarti Malhotra, Pei Pei, Kee-Eung Kim, Bongseok Goh, and Michael Bowling. Approximate linear programming for constrained partially observable Markov decision processes. In *AAAI*, pages 3342–3348. AAAI Press, 2015.

[Puterman, 2005] M L Puterman. *Markov Decision Processes*. Wiley-Interscience, 2005.

[Randour *et al.*, 2015] Mickael Randour, Jean-François Raskin, and Ocan Sankur. Variations on the stochastic shortest path problem. In *VMCAI*, volume 8931 of *LNCS*, pages 1–18. Springer, 2015.

[Ross *et al.*, 2008] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-draa. Online planning algorithms for POMDPs. *J. Artif. Intell. Res. (JAIR)*, 32:663–704, 2008.

[Russell and Norvig, 2010] Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.

[Santana *et al.*, 2016] Pedro Santana, Sylvie Thiébaux, and Brian C. Williams. RAO*: An algorithm for chance-constrained POMDP's. In *AAAI*, pages 3308–3314. AAAI Press, 2016.

[Silver and Veness, 2010] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *NIPS 23*, pages 2164–2172. Curran Associates, Inc., 2010.

[Smith and Simmons, 2004] T. Smith and R. Simmons. Heuristic search value iteration for POMDPs. In *UAI*, pages 520–527. AUAI Press, 2004.

[Sprauel *et al.*, 2014] Jonathan Sprauel, Andrey Kolobov, and Florent Teichteil-Königsbuch. Saturated path-constrained mdp: Planning under uncertainty and deterministic model-checking constraints. In *AAAI*, pages 2367–2373, 2014.

[Undurti and How, 2010] Aditya Undurti and Jonathan P How. An online algorithm for constrained POMDPs. In *ICRA*, pages 3966–3973. IEEE, 2010.

[Ye *et al.*, 2017] Nan Ye, Adhiraj Somani, David Hsu, and Wee Sun Lee. DESPOT: Online POMDP planning with regularization. *J. Artif. Intell. Res.*, 58:231–266, 2017.