2-2017

Optimizing expectation with guarantees in POMDPs

Krishnendu CHATTERJEE

Guillermo A. PEREZ

Jean-François RASKIN

Dorde ZIKELIC
Singapore Management University, dzikelic@smu.edu.sg

Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Optimizing Expectation with Guarantees in POMDPs

Krishnendu Chatterjee, Petr Novotný
IST Austria, Klosterneuburg, Austria
krishnendu.chatterjee@ist.ac.at, pnovotny@ist.ac.at

Guillermo A. Pérez,* **Jean-François Raskin**
Université Libre de Bruxelles, Brussels, Belgium
jraskin@ulb.ac.be, gperezme@ulb.ac.be

Đorđe Žikelić
University of Cambridge, Cambridge, UK
dz277@cam.ac.uk

Abstract

A standard objective in partially-observable Markov decision processes (POMDPs) is to find a policy that maximizes the expected discounted-sum payoff. However, such policies may still permit unlikely but highly undesirable outcomes, which is problematic especially in safety-critical applications. Recently, there has been a surge of interest in POMDPs where the goal is to maximize the probability to ensure that the payoff is at least a given threshold, but these approaches do not consider any optimization beyond satisfying this threshold constraint. In this work we go beyond both the "expectation" and "threshold" approaches and consider a "guaranteed payoff optimization (GPO)" problem for POMDPs, where we are given a threshold t and the objective is to find a policy σ such that a) each possible outcome of σ yields a discounted-sum payoff of at least t, and b) the expected discounted-sum payoff of σ is optimal (or near-optimal) among all policies satisfying a). We present a practical approach to tackle the GPO problem and evaluate it on standard POMDP benchmarks.

1 Introduction

The *de facto* model for decision making under uncertainty are partially-observable Markov decision processes (POMDPs) (Littman 1996; Papadimitriou and Tsitsiklis 1987), and they have been applied in diverse applications ranging from planning (Russell and Norvig 2010), to reinforcement learning (Kaelbling, Littman, and Moore 1996), to robotics (Kress-Gazit, Fainekos, and Pappas 2009; Kaelbling, Littman, and Cassandra 1998). One of the classical and fundamental payoff function for POMDPs is the *discounted-sum payoff* that aggregates the rewards of the transitions as a discounted sum. The traditional objective in POMDPs has been to obtain policies that maximize the expected discounted-sum payoff.

One crucial drawback of the traditional objective (that asks for expectation maximization) is that it allows for undesirable events that can happen with low probability. For example, consider a policy σ_1 that with probability $1/2$ achieves payoff 100 and with probability $1/2$ achieves payoff 0, and a different policy σ_2 that achieves payoff 20 with probability 1. If payoff values below 10 are undesirable, then the first policy, though better for expected payoff, allows undesirable events with significant probability, and hence the second policy is preferable. Hence, there has been a recent interest to study objectives where, instead of maximizing the expected payoff (Hou, Yeoh, and Varakantham 2016), the goal is to maximize the probability that the payoff is above a threshold.

A drawback of the approach to maximize the probability that the payoff exceeds a threshold is that it ignores the optimization aspect of maximizing the expectation. In this work we consider an objective for POMDPs where both aspects are present. More precisely, we consider a "guaranteed payoff optimization (GPO)" problem for POMDPs, where given a threshold t, the goal is to maximize the expectation while ensuring that the payoff is at least t.

As a concrete motivation for the GPO problem, consider planning under uncertainty (e.g., self-driving cars) where certain events are catastrophic (e.g., crashes), and in the model they are assigned low payoffs. Such catastrophic events must be avoided even at the expense of expected payoff. That is, policies must maximize the expected payoff, ensuring the avoidance of catastrophic events. Hence, for planning in safety-critical applications the GPO problem is natural.

In this work, our main contributions are as follows:

1. We study the GPO problem for POMDPs, and present a practical solution approach for the problem. In particular, given a POMDP with the GPO problem, we present a transformation to a different POMDP where it suffices to solve the traditional expectation objective. Our solution approach first constructs a representation of all strategies that satisfies item a) of the GPO problem, and then we extend the partially-observable Monte Carlo planning (POMCP) approach to obtain optimal policies w.r.t. expectation among the above strategies.
2. We present experimental results on several classical POMDP examples from the literature to show how our approach can efficiently solve the GPO problem for POMDPs.

Due to lack of space, we omit formal proofs for some of our claims in this article. We refer the interested reader to the full technical report (Chatterjee et al. 2016).

Related Works. Works studying POMDPs with discounted sum range from theoretical results (see, e.g., (Pa-

*Author supported by an F.R.S.-FNRS Aspirant fellowship.

padimitriou and Tsitsiklis 1987; Littman 1996)) to practical tools (e.g. (Kurniawati, Hsu, and Lee 2008; Silver and Veness 2010)). Recent works focus on extracting policies which ensure that, with a given probability bound, the obtained discounted-sum payoff is above a threshold (see, e.g., (Hou, Yeoh, and Varakantham 2016)). The problem of ensuring the payoff is above a given threshold while optimizing the expectation has been considered for fully-observable MDPs and the long-run average and stochastic shortest path objectives (Bruyère et al. 2014; Randour, Raskin, and Sankur 2015); and also with probabilistic thresholds for long-run average payoff (Chatterjee, Komárková, and Kretínský 2015). As for POMDPs, we mention *constrained POMDPs* (Undurti and How 2010; Poupart et al. 2015), where the aim is to maximize the expected payoff while ensuring that the expectation of some other quantity is bounded. In contrast, our constraints are *hard*, i.e. they must hold always, not just on average. The work probably closest to ours is (Santana, Thiébaux, and Williams 2016) that also considers maximizing expected payoff among all policies satisfying a given constraint, but there are two key differences from our work: they consider finite horizon POMDPs, while we consider infinite horizon ones, and more importantly, their constraints are *state-based*, i.e. their policy must ensure that the execution of the POMDP does not go through certain "violating" states. In contrast, our "threshold constraint" is *execution-based*: whether a execution yields payoff at least t cannot be determined solely by looking at the set of states appearing in the execution, but the whole infinite execution has to be considered. This requires very different techniques. To our best knowledge, the GPO problem has never been considered for POMDPs with discounted sum.

2 Preliminaries

Throughout this work, we follow standard (PO)MDP notations from (Puterman 2005; Littman 1996).

2.1 POMDPs

We denote by $\mathcal{D}(X)$ the set of all probability distributions on a finite set X, i.e. all functions $f : X \to [0,1]$ such that $\sum_{x \in X} f(x) = 1$. For $f \in \mathcal{D}(X)$ we denote by $\mathrm{Supp}(f)$ the *support* of f, i.e. the set $\{x \in X \mid f(x) > 0\}$.

Definition 1 ***POMDPs.*** *A* POMDP *is defined as 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* $\delta(s,a)(s')$ *by* $\delta(s'|s,a)$*,*

Remark 1 (Deterministic observation function) *Deterministic observation functions of type* $\mathcal{O} : S \to \mathcal{Z}$ *are sufficient in POMDPs (see Remark* 1 *in (Chatterjee et al. 2014)). Informally, the probabilistic aspect of the observation function can be encoded into the transition function and, by letting the product of the states and observations be the new state-space, we obtain a deterministic observation function. Thus, without loss of generality, we will always consider observation functions of type* $\mathcal{O} : S \to \mathcal{Z}$*, which greatly simplifies the notation.*

Plays & Histories. A *play* (or an infinite path) in a POMDP is an infinite sequence $\rho = s_0 a_0 s_1 a_1 s_2 a_2 \ldots$ of states and actions such that $s_0 \in \mathrm{Supp}(\lambda)$ and for all $i \geq 0$ we have $\delta(s_i, a_i)(s_{i+1}) > 0$. We write Ω for the set of all plays. 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 $h = a_0 o_1 \ldots a_{i-1} o_i \in (\mathcal{A} \cdot \mathcal{Z})^*$ such that there is a path $w = s_0 a_0 s_1 \ldots a_{i-1} s_i$ with $o_j = \mathcal{O}(s_j)$ 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 ρ is $len(\rho) = \infty$.

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 λ is given as 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 $z \in \mathcal{Z}$ is received, the updated belief $b_{h'}$ for history $h' = hao$ can be computed by a standard formula (Cassandra 1998).

Infinite-horizon Discounted Payoff. Given a play $\rho = s_0 a_0 s_1 a_1 s_2 a_2 \ldots$ and a discount factor $0 \leq \gamma < 1$, the *infinite-horizon discounted payoff* Disc_γ of ρ is:

$$\mathsf{Disc}_\gamma(\rho) = \sum_{i=0}^{\infty} \gamma^i r(s_i, a_i).$$

We also define a discounted payoff of a finite path w as $\mathsf{Disc}_\gamma(w) = \sum_{i=0}^{len(w)-1} \gamma^i r(s_i, a_i)$.

Policies. A *policy* is a blueprint for selecting actions based on the past history of observations and actions. Formally, it is a function σ 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 often abbreviate $\sigma(h)(a)$ to $\sigma(a \mid h)$).

Consistent Plays. A play or a path w is *consistent* with a policy σ if it can be obtained by extending its finite prefixes using σ. Formally, $w = s_0 a_0 s_1 a_1 \ldots$ is consistent with σ if for each $0 \leq i \leq len(w)$ there is action a such that $\sigma(a \mid H(s_0 a_0 \ldots a_{i-1} s_i)) > 0$ and $\delta(s_{i+1} \mid s_i, a) > 0$. A history h is consistent with σ if there is a path w consistent with σ such that $h = H(w)$.

Expected Value $eVal^P$ of Policies. Given a POMDP P, a policy σ, a discount factor γ, and an initial belief λ, the *expected value* of σ from λ is the expected value of the

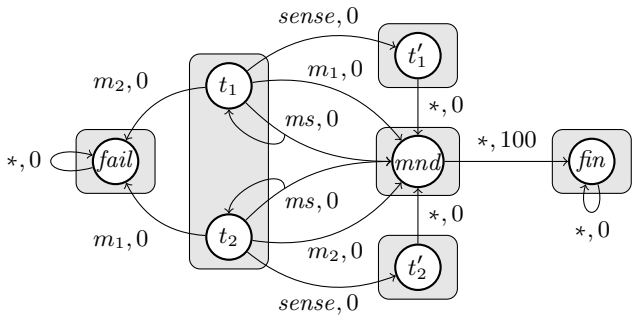

Figure 1: Illustrative POMDP. We assume a discount factor $\gamma = \frac{1}{2}$. Gray rectangles represent observations. The only probabilistic branching occurs when ms is played in t_1 or t_2, and for both $i \in \{1, 2\}$ we have $\delta(mnd \mid t_i, ms) = \frac{3}{5}$ and $\delta(t_i \mid t_i, ms) = \frac{2}{5}$. The initial belief λ assigns $\frac{9}{10}$ to state t_1 and $\frac{1}{10}$ to t_2. Asterisks denote that a transition is performed under any action.

infinite-horizon discounted sum under policy σ when starting in a state sampled from λ: $eVal^P(\sigma) = \mathbb{E}^\sigma_\lambda[\mathsf{Disc}_\gamma]$. This definition can be formalized by a standard construction of a probability measure induced by σ over the set of all plays, which also gives rise to the expectation operator $\mathbb{E}^\sigma_\lambda$ (see, e.g., (Puterman 2005)).

Worst-Case Value $wVal^P$ of Policies. The *worst-case* value of a policy σ from belief λ is $wVal^P(\sigma) = \inf_\rho \mathsf{Disc}_\gamma(\rho)$, where the infimum is taken over the set of *all plays that are consistent with* σ and start in a state sampled from λ.

Example 1 *Figure 1 shows a toy POMDP: A mining robot has to mine ore, which can be of two types (states t_1 and t_2). The exact type is unknown, but t_1 is more likely to occur (initial belief λ). The goal is to reach the "ore mined" (mnd) state, in which a lump-sum reward is received. The robot can use several mining modes: safe mode (action ms), which succeeds with probability 0.6 and does not do anything if it fails, or type-specific mining modes (m_1 and m_2) which succeed if applied on the correct type but result in a catastrophic failure if used on a wrong type. It can also use a sensor to accurately determine the type (after which a type-specific action can be safely used), at a cost of a one-step delay.*

An exhaustive analysis of possible policies reveals that the expected value is maximized by any policy σ which selects m_1 in the first step (we then have $eVal^P(\sigma) = 45$). However, the worst-case value of such a policy is 0, as it can result in entering $fail$ after the first step. On the other hand, a policy σ' which plays $sense$ in the first step has $eVal^P(\sigma) = wVal^P(\sigma) = 25$.

Main Computational Questions. The standard *POMDP planning* problem asks to compute (or approximate) the policy maximizing the expected value. In *online POMDP planning*, instead of computing the whole policy we have to compute, in each time step, the best action in the current situation. In other words, we must compute a good local approximation of a (near-)optimal policy. (Ross et al. 2008). In contrast, in the *threshold* planning problem we are asked to compute a policy maximizing the worst-case value and thus provide strict guarantees on the performance of the system (Zwick and Paterson 1996). In this paper, we combine these two approaches and study the *guaranteed payoff optimization (GPO)* problem, where we are given a POMDP P and a threshold $t \in \mathbb{R}$ and we have to compute a policy σ such that

a) σ satisfies a *threshold constraint*: $wVal^P(\sigma)$ is at least t.
b) Let $gVal^P(t)$ denote the best expected value obtainable while ensuring a worst-case payoff of at least t, i.e. $gVal^P(t) := \sup\{eVal(\pi) \mid wVal^P(\pi) \geq t\}$. Among all policies that satisfy item a), σ has ε-maximal expected value, i.e. $eVal^P(\sigma) \geq gVal^P(t) - \varepsilon$.

To efficiently tackle the GPO problem we aim to compute, in an online fashion, a local approximation of policy σ above. However, we *do not* relax requirement a). Approximations notwithstanding, the online planning algorithm we seek is such that given t, the discounted payoff of every single play that can be produced by the algorithm is at least t.

Example 2 *Take the POMDP in Figure 1 and a threshold $t = 5$. As shown in Example 1, a policy σ' playing $sense$ in the first step satisfies $wVal^P(\sigma') \geq t$. However, there are better (w.r.t. the expected value) policies satisfying this constraint. The best such policy is a policy σ'' which twice plays ms and then plays $sense$. This policy satisfies $eVal^P(\sigma'') = 37$ and $wVal^P(\sigma'') = 6.25$. (Also note that the optimal policy to maximize the expected payoff plays m_1 at the very start. However, with non-zero probability, this strategy violates the worst-case threshold $t = 5$.)*

3 Policies for GPO Problem

We first show the GPO problem is different from the classical expectation maximization.

Example 3 (Beliefs are not sufficient for GPO.) *It is known that beliefs form a sufficient statistic of history for achieving the optimal expected value, i.e. there is always a deterministic belief-based policy σ — that is, a policy such that for each history h the distribution $\sigma(h)$ is Dirac and determined solely by the belief after observing h — with optimal expected value (Sondik 1971). However, beliefs are not a sufficient statistic for the GPO problem, as witnessed by Example 2: suppose that we use policy σ'' and consider histories $h = (ms, o, ms, o)$ and $\bar{h} = (ms, o)$, where o is the observation received in t_1 and t_2. The beliefs b_h and $b_{\bar{h}}$ are identical, and yet $\sigma''(h) \neq \sigma''(\bar{h})$, i.e. σ'' is not belief-based.*

Overview of Policy Representation. We show (in Corollary 1) that a sufficient statistic for solving the GPO problem is a tuple $(b_h, rem^t_\gamma(h))$, where b_h is the belief after history h and $rem^t_\gamma(h)$ is the "remaining" distance to the threshold

which we need to accumulate in the future. Formally,

$$rem^t_\gamma(h) = (t - \min\{\mathsf{Disc}_\gamma(w) \mid H(w) = h\}) / \gamma^{len(h)}.$$

This is similar to other (PO)MDP planning problems that work with thresholds (White 1993; Hou, Yeoh, and Varakantham 2016). However, we prove more: we obtain a precise local characterization of policies that satisfy the threshold constraint. More precisely, we show that for each history h, there is a set of *allowed* actions $Allow^t_\gamma(h)$ such that a policy σ satisfies $wVal^P(\sigma) \geq t$ if and only if for each history h it holds $\mathrm{Supp}(\sigma(h)) \subseteq Allow^t_\gamma(h)$. We show that the function $Allow^t_\gamma$ can be finitely represented and, for any history h, its value can be computed algorithmically. This permits us to split the solution of the GPO problem into two separate parts: 1.) We compute the function $Allow^t_\gamma$, and 2.) we use it to restrict a standard online planning algorithm so that it always returns an action allowed for the current history.

Allowed Actions $Allow^t_\gamma$. Intuitively, an action a should be allowed after some history h only if the payoff we are guaranteed to accumulate using a in the current step (i.e. $\min_{s \in \mathrm{Supp}(b_h)} r(s,a)$) plus the best payoff which we can guarantee from the next step onward is at least $rem^t_\gamma(h)$. To formalize the "best payoff guaranteed from the next step on" we define the *future value* of any history h as

$$fVal(h) = \sup_\sigma wVal^{P[b_h]}(\sigma),$$

where $P[b_h]$ is a POMDP identical to P except for having initial belief b_h and the supremum is taken over all policies in $P[b_h]$.

Belief Supports Suffice for the Worst Case. The crucial observation is that the future value of a history h is determined only by the support of b_h.

Lemma 1 *If histories h, h' in a POMDP P are such that $\mathrm{Supp}(b_h) = \mathrm{Supp}(b_{h'})$, then $fVal(h) = fVal(h')$.*

Intuitively, this is because the worst-case value of a policy (and thus also a future value of a history) does not depend on any transition probabilities. In a slight abuse of notation, we sometimes treat $fVal$ as a function from 2^S to $\mathbb{R}$, i.e. $fVal(B)$, for $B \subseteq S$, is equal to $fVal(h)$ for all histories h such that $\mathrm{Supp}(b_h) = B$.

Ψ as an Approximation of $fVal$. Since computing $fVal(B)$ exactly can be inefficient in practice, we often need to work with approximations of $fVal(B)$, without relaxing the threshold constraint. We thus introduce a notion of a Ψ-allowed action. Let $\Psi \colon 2^S \to \mathbb{R}$ be a function assigning numbers to belief supports. We say that an action a is *Ψ-allowed* for $t \in \mathbb{R}$ after history h, and write it $a \in \Psi\text{-}Allow^t_\gamma(h)$, if for all states $s \in \mathrm{Supp}(b_h)$ and all observations $o \in \mathcal{Z}$ such that hao is a history it holds that

$$r(s,a) + \gamma \cdot \Psi(\mathrm{Supp}(b_{hao})) \geq rem^t_\gamma(h). \qquad (1)$$

If Ψ is the function $fVal$, we write simply $a \in Allow^t_\gamma(h)$. We typically aim at computing a lower bound on $fVal$, i.e. a function Ψ such that $\Psi(B) \leq fVal(B)$ for each $B \in 2^S$. Then, as shown below, playing Ψ-allowed actions still guarantees that the threshold t is eventually surpassed.

Correctness of the Approximation. The correctness of the definition is summarized in the following proposition. We say that a policy σ is Ψ-safe for $t \in \mathbb{R}$ if for each history h consistent with σ it holds that $\mathrm{Supp}(\sigma(h)) \subseteq \Psi\text{-}Allow^t_\gamma(h)$.

Proposition 1 *Let $\Psi \colon 2^S \to \mathbb{R}$ be a function such that $\Psi(B) \leq fVal(B)$ for each $B \in 2^S$. Then any policy σ that is Ψ-safe for t satisfies $wVal^P(\sigma) \geq t$. Moreover a policy π is $fVal$-safe for t if and only if $wVal^P(\pi) \geq t$.*

Corollary 1 *Assume that there is a policy σ with $wVal^P(\sigma) \geq t$. Then there is also a policy π such that $wVal^P(\pi) \geq t$ and $eVal^P(\pi) = gVal^P(t)$, and moreover, π is belief-and-payoff, based, i.e. for all histories h, h' such that $(b_h, rem^t_\gamma(h)) = (b_{h'}, rem^t_\gamma(h'))$ it holds $\pi(h) = \pi(h')$.*

From (1) we see that to compute $Allow^t_\gamma(h)$ we have to keep track of $rem^t_\gamma(h)$ (which can be easily done online) and to compute $fVal(\mathrm{Supp}(b_h))$ (or a suitable under-approximation thereof). In the next section we show how to do the latter.

Example 4 *Consider the POMDP from Figure 1 with a threshold $t = 6.25$. Then $fVal(\{fin\}) = fVal(\{fail\}) = 0$, $fVal(\{t_1, t_2\}) = 25$, $fVal(\{t'_1\}) = fVal(\{t'_2\}) = 50$, and $fVal(\{mnd\}) = 100$. Initially, for the empty history, we have $rem^{6.25}_{0.5}(\cdot) = 6.25$ and therefore the only allowed actions are ms and $sense$ because for all $i \in \{1,2\}$ we have $r(t_i, m_{3-i}) + \gamma fVal(\{fail\}) < rem^{6.25}_{0.5}(\cdot)$. Suppose that ms is played and that the next observation witnessed is $\mathcal{O}(t_1) = \mathcal{O}(t_2)$ (thus, the belief is the same as before). We have $rem^{6.25}_{0.5}(ms\mathcal{O}(t_1)) = 12.5$. In this case, the only allowed action is $sense$ because for all $i \in \{1,2\}$ $r(t_i, ms) + \gamma fVal(\{t_1, t_2\}) < rem^{6.25}_{0.5}(ms\mathcal{O}(t_1))$ and m_1 and m_2 are still not allowed (since we have not accumulated any payoff and have the same belief as before). Hence, $sense$ is played and consequently we obtain a payoff of 25 (because of discounting). We remark that 25 is, as required, above the threshold $t = 6.25$.*

4 Computing Future Values

The threshold constraint in the GPO problem is *global*, i.e. it talks about *all* runs compatible with a policy. Hence, solving the GPO problem is unlikely to be amenable to *purely* online methods, which compute only local approximations of policies. In this section we show how to compute future values in an offline pre-processing step. Although this requires a global analysis of a POMDP, the pre-processing step can be done efficiently since computation of future values only requires working with belief supports rather than beliefs.

Belief Supports & Valid Belief Supports $VBelSup$**.** A belief support $B \subseteq 2^S$ is *valid* if either $B = \mathrm{Supp}(\lambda)$ or there is a history h such that $B = \mathrm{Supp}(b_h)$. Only valid supports can be encountered during the planning process and thus we only need to compute future values thereof. We denote by $VBelSup(P)$ the set of valid belief supports of POMDP P; the set can be computed by a simple iterative procedure.

Observable Rewards. We present efficient computation of future values under the assumption that *rewards are observable*. This holds for many real-world applications, see, e.g. examples in (Hou, Yeoh, and Varakantham 2016; Chatterjee et al. 2015). Formally, POMDP P has observable rewards if $r(s,a) = r(s',a)$ whenever $\mathcal{O}(s) = \mathcal{O}(s')$. From a theoretical point of view, observability of rewards is necessary since without it, the computation of future values is at least as hard as solving a long-standing open problem in algebraic number theory. More precisely, if the rewards of a given POMDP are not observable, the computation of future values is at least as hard as solving the *target discounted sum problem*, a long-standing open problem in automata theory related to other open problems in algebra (Boker, Henzinger, and Otop 2015). However, for POMDPs with unobservable rewards we can at least obtain an under-approximation Ψ of $fVal$, and hence our framework is also applicable to them.

Lemma 2 *If rewards in P are observable, then for each $B \in VBelSup(P)$ and each $s, s' \in B, a \in \mathcal{A}$ it holds $r(s,a) = r(s',a)$.*

We thus define $r(B,a)$ as $r(s,a)$ for some $s \in B$.

Future Value Characterization. We start by providing a characterization of future values. A *successor* of a belief support B under action a and observation o is a belief support $\Delta(B,a,o) = o \cap \bigcup_{s \in B} \mathrm{Supp}(\delta(s,a))$. Consider the following system of max-min equations with variables x_B, $B \in VBelSup(P)$:

$$x_B = \max_{a \in \mathcal{A}} \min_{\substack{o \in \mathcal{Z} \\ \Delta(B,a,o) \neq \emptyset}} r(B,a) + \gamma \cdot x_{\Delta(B,a,o)}. \qquad (2)$$

(Each $B \in VBelSup(P)$ appears on the LHS of exactly one equation in the system.)

Proposition 2 *The system* (2) *has a unique solution $\{\tilde{x}_B\}_{B \in VBelSup(P)}$, and it satisfies $\tilde{x}_B = fVal(B)$.*

Game Perspective for the Worst Case. Hence, it suffices to find a solution to system (2). But the form of the system is identical to the one characterizing optimal values in 2-player zero-sum discounted games (Zwick and Paterson 1996). These games can be imagined as fully-observable MDPs in which the outcomes of actions are not resolved by a random choice but by a malicious adversary. The system (2) *per se* corresponds to a game where elements of $VBelSup(P)$ are the states, actions are the same as in P, and possible effects of actions are given by the function Δ.

Algorithms to Compute Future Values. Hence, to compute future values in practice we can employ one of several efficient algorithms for solving discounted-sum games (e.g. (Brenguier 2016)). A simple yet efficient approach is to use the standard *value iteration* for games: we compute a sequence $f^{(0)} f^{(1)} f^{(2)} \dots$ of functions of type $VBelSup(P) \to \mathbb{R}$ such that $f^{(0)}(B) = 0$ for each B, and for $i \geq 1$ we inductively define

$$f^{(i)}(B) = \max_{a \in \mathcal{A}} \min_{\substack{o \in \mathcal{Z} \\ \Delta(B,a,o) \neq \emptyset}} r(B,a) + \gamma \cdot f^{(i-1)}(\Delta(B,a,o)).$$

From (Zwick and Paterson 1996) it follows there is always j such that for all $B \in VBelSup(P)$ we have $f^j(B) = f^{j-1}(B)$, i.e. $f^j(B)$ is the solution to (2), and moreover $j \leq 3 + \log_2(\max_{(s,a) \in S \times \mathcal{A}} |r(s,a)|) + \frac{1}{2} \cdot (|S|+3)^2 \cdot \frac{\log_2(den(\gamma))}{1-\gamma}$, where $den(\gamma)$ is a denominator of γ in its reduced form. Hence, the value iteration converges in at most exponentially many steps.[1]

Theorem 1 *Future values of all valid belief supports in P can be computed in time exponential in the size of P.*

Although the theoretical bound is exponential, there are several reasons for the method to work well in practice: (1.) In a concrete instance, the number of valid supports can be significantly smaller than exponential. (2.) Reaching the fixed-point of the value iteration may also require significantly smaller number of steps than the theoretical upper bound suggests. (3.) One can show that for each $i \geq 0$, $f^{(i)} \leq fVal$. Hence, even if reaching the fixed point takes too much time, we can set up a suitable timeout after which the value iteration is stopped, say at iteration i. Then, by Proposition 1 any policy that is $f^{(i)}$-safe for t has worst-case value $\geq t$. (4.) Value iteration is a simple and standard algorithm for which efficient implementations exist (see, e.g., (Littman, Dean, and Kaelbling 1995; Spaan and Vlassis 2005)).

Important note on Ψ: generally, $\Psi \leq fVal$ does not guarantee that a Ψ-safe policy exists, which is necessary to apply Proposition 1. The following lemma resolves this.

Lemma 3 *For any $i \geq 0$ the following holds for the functions $f^{(i)}$ produced by game value iteration: if $f^{(i)}(\mathrm{Supp}(\lambda)) \geq t$, then there exists a policy σ which is $f^{(i)}$-safe for t.*

In particular, if $fVal(\mathrm{Supp}(\lambda)) \geq t$ then a $fVal$-safe policy for t exists, irrespective of the way in which $fVal$ is computed.

5 Solving the GPO problem

We solve the GPO problem by modifying the partially-observable Monte Carlo planning (POMCP) algorithm (Silver and Veness 2010).

[1] Since the number $\frac{1}{1-\gamma}$ can be exponential in the bitsize of γ.

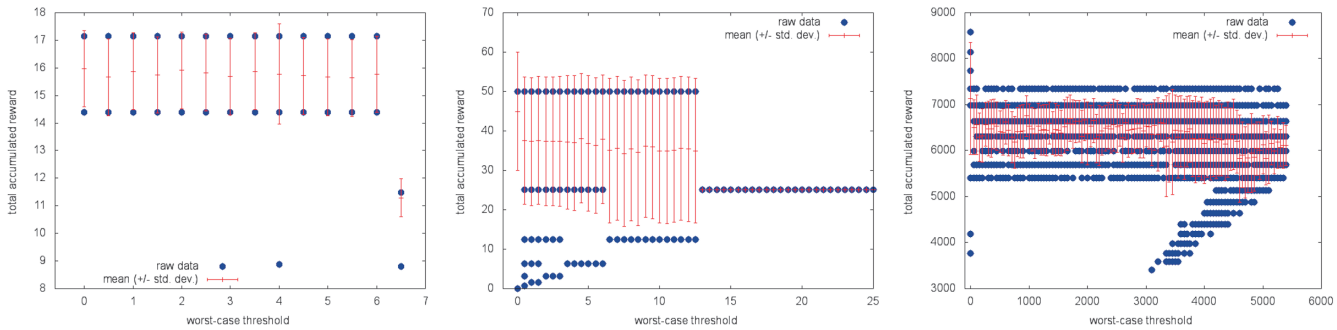

Figure 2: Plots of results obtained from simulating (1.) a RockSample benchmark, (2.) the POMDP from Example 1, and (3.) a hallway benchmark with probabilistic spinning (a.k.a. traps), all with increasing worst-case thresholds (until $fVal(\mathrm{Supp}(\lambda))$). Each circle with coordinates (x, y) corresponds to a simulation of G-POMCP, ran with worst-case threshold x, that obtained y as accumulated payoff. The vertical bars show the mean and standard deviation per worst-case threshold. (We have plotted at least 100 data-points per worst-case threshold for the RockSample benchmark; 1000 for Example 1; 20 for the hallway benchmark.)

POMCP. POMCP is an online planning method which in each decision epoch aims to select the best action given the current history h. In each epoch, POMCP performs a number of finite-horizon simulations starting from belief b_h in order to compute a local approximation of the optimal expected value function: each simulation extends history h by selecting actions according to certain rules until the horizon is reached. The payoff of the produced path is then evaluated, and the result is used to update the optimal value approximation. After all the simulations proceed, the best action according to the estimated values is played, a new observation is received, and the process continues as above.

POMCP data-structure. POMCP stores the information gained in past simulations in a *search tree*, in which each node corresponds to some history h' and contains belief $b_{h'}$, the number $N_{h'}$ of times the history has been observed in previous simulations, and an approximation of the optimal expected value from $b_{h'}$. The search tree is used to guide simulations: each step in which the current history corresponds to an internal node of the tree is treated as a multi-armed bandit with parameters determined by numbers stored in children of this node, which balances exploration of new branches and exploitation of previous simulations (akin to the UCT algorithm for MDPs (Kocsis and Szepesvári 2006)). Once the simulation runs out of the scope of the search tree, it enters a *rollout* phase, where a fixed policy (e.g. selecting actions at random) is used to extend paths.

G-POMCP: Adapting POMCP for GPO. We propose an augmentation of POMCP, which we call G-POMCP (*guaranteed* POMCP), specified as follows: First we enrich the nodes of the search tree so that a node corresponding to a history h additionally includes the set $B_h = \mathrm{Supp}(b_h)$ and the number $R_h = rem^{\gamma}_t(h)$. When adding a new node to a search tree by extending history h with action a and observation o, these attributes for the new node are updated as follows: $B_{hao} = \Delta(B_h, a, o)$ and $R_{hao} = (rem^{\gamma}_t(h) - r(B_h, a))/\gamma$. Note that updating B_h to B_{hao} requires just discrete set operations; as a matter of fact, the function Δ is computed already during the off-line computation of future values, after which it can be stored and used to efficiently update B_h during G-POMCP execution. In particular, updating B_h is independent of updating b_h, which is important so as not to compromise the threshold constraints with issues of belief precision and particle deprivation.

G-POMCP: playing safe. The execution of G-POMCP then proceeds in almost the same way as in POMCP, with a crucial exception: Whenever G-POMCP is to select a (real or simulated) action it selects only among those in $Allow^t_{\gamma}(h)$, where h is the current history. Note that checking whether an action is allowed is easy for histories within the search tree, since the necessary information (B_h and R_h) is stored in nodes of the tree. Out of the scope of the search tree, we need to update the current belief support and remaining payoff online, as the simulation proceeds. While this somewhat increases the complexity of rollouts, as current belief supports must be kept updated (POMCP only keeps track of the current state and of payoff won so far), as noted above, updating belief supports is easier than updating beliefs. Moreover, this increase in complexity is only an issue in the initial steps of the algorithm, where rollout steps dominate over tree traversal. Previous sections yield the following result:

Theorem 2 *For each threshold $t \leq fVal(\mathrm{Supp}(\lambda))$ the following holds: for each play $\rho = s_0 a_0 s_1 a_1 \ldots$ resulting from using G-POMCP on P ad infinitum it holds $\mathsf{Disc}_{\gamma}(\rho) \geq t$. This holds independently of how precisely the algorithm approximates beliefs.*

So unless it is impossible to satisfy the threshold constraint at all, it can be surely satisfied by using G-POMCP.

Convergence. Another question is the one of convergence. An algorithm is said to be convergent in the limit if, assuming precise belief representation, the local approximation of optimal value converges to true optimal value (in our case

No.	states	act.	obs.	pre. proc.	avg. lat.
tiger	7	4	6	< 0.001s	< 0.009s
r.sample	10207	7	168	184s	0.816s
hallway	2039	3	18	2.02s	1.308s

Table 1: Latency of G-POMCP with planning horizon of 1K

to $gVal^P(t)$) as the number of simulations and their depth increases. The limit convergence of G-POMCP can be proved by a straightforward adaptation of the limit convergence proof of POMCP (Silver and Veness 2010): we map executions of G-POMCP on POMDP P to the executions of UCT on a tree-shaped MDP P', whose states are histories of P (with the empty history as root) and where finite paths correspond to extending histories in P by playing allowed actions.

6 Experiments

We tested our algorithm on two classical sets of benchmarks. The first, Hallway, was introduced in (Littman, Cassandra, and Kaelbling 1995). In a hallway POMDP, a robot navigates a gridworld with walls and traps. We have considered variants in which traps cause non-recoverable damage and another in which they just "spin" the robot — making him more uncertain about his current location in the grid. Additionally, we have run our algorithm on RockSample POMDPs. The latter corresponds to the classical scenario described first in (Smith and Simmons 2004). (We use a slight adaptation with a single imprecise sensing action.) Our experimental results are summarized in Figure 2 and Table 1.

Test Environment Specifications: (1.) CPU: 6-Core Intel Zeon, 3.33 GHz, 6 cores; (2.) Memory: 256 KB of L2 Cache, 12 MB of L3 Cache, 32 GB; (3.) OS: Mac OS X 10.7.5.

Worst-Case vs. Expected Payoff. In Figure 2 we have plotted the results of running our G-POMCP algorithm on several benchmarks. In all three graphics, the trade-off between worst-case guarantees and expected payoff is clearly visible: In the left figure, the expected payoff stays around 15.7 for worst-case thresholds between 0 and 6; then drops to 11.3 for threshold values above 6.5. In the center figure, the expected payoff is ~44.7 when the worst-case threshold is 0; stays around 36 for thresholds between 1 and 12 (with a slightly negative slope); then drops to 25 for threshold values above 12.5. Finally, in the right figure, the expected payoff steadily decreases for increasing worst-case threshold values. In particular, for threshold 0 the expected payoff is ~7137 while for threshold 5150 it is ~6161.

Latency. In Table 1 we show the *latency* — the amount of time it takes to determine, at each epoch, which action to play next — of G-POMCP on three of the benchmarks we considered. (Though we have run the tool on several others, these are the biggest.) Observe that, even for relatively big POMDPs, the average latency is in the order of seconds. Also, note that the pre-processing step is not too costly.

Tool Availability. Our implementation of the G-POMCP algorithm can be fetched from `https://github.com/gaperez64/GPOMCP`.

7 Discussion

In this work we have given a practical solution for the GPO problem. Our algorithm, G-POMCP, allows to obtain a policy which ensures a worst-case discounted-sum payoff value while optimizing the expected payoff. We have implemented G-POMCP and evaluated its performance on classical families of benchmarks. Our experiments show that our approach is efficient despite the exact GPO problem being fundamentally more complicated.

Acknowledgements

The research leading to these results was supported by the Austrian Science Fund (FWF) NFN Grant no. S11407-N23 (RiSE/SHiNE); two ERC Starting grants (279307: Graph Games, 279499: inVEST); the Vienna Science and Technology Fund (WWTF) through project ICT15-003; and the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement no. [291734].

References

Boker, U.; Henzinger, T. A.; and Otop, J. 2015. The Target Discounted-Sum Problem. In *LICS*, 750–761.

Brenguier, R. 2016. A solver for Mean Payoff Games, based on gain and bias equations and the Z3 SMT solver. https://github.com/romainbrenguier/MeanPayoffSolver. Accessed date: 2016-08-07.

Bruyère, V.; Filiot, E.; Randour, M.; and Raskin, J.-F. 2014. Meet Your Expectations With Guarantees: Beyond Worst-Case Synthesis in Quantitative Games. In Mayr, E. W., and Portier, N., eds., *STACS*, volume 25 of *LIPIcs*, 199–213. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Cassandra, A. 1998. *Exact and approximate algorithms for partially observable Markov decision processes*. Brown University.

Chatterjee, K.; Chmelik, M.; Gupta, R.; and Kanodia, A. 2014. Optimal Cost Almost-sure Reachability in POMDPs. *CoRR* abs/1411.3880.

Chatterjee, K.; Chmelik, M.; Gupta, R.; and Kanodia, A. 2015. Optimal Cost Almost-sure Reachability in POMDPs. In *AAAI*. AAAI Press.

Chatterjee, K.; Novotný, P.; Pérez, G. A.; Raskin, J.-F.; and Đorđe Žikelić. 2016. Optimizing expectation with guarantees in pomdps (technical report). *CoRR* abs/1611.08696.

Chatterjee, K.; Komárková, Z.; and Kretínský, J. 2015. Unifying Two Views on Multiple Mean-Payoff Objectives in Markov Decision Processes. In *LICS*, 244–256. IEEE Computer Society.

Hou, P.; Yeoh, W.; and Varakantham, P. 2016. Solving Risk-Sensitive POMDPs With and Without Cost Observations. In Schuurmans, D., and Wellman, M. P., eds., *AAAI*, 3138–3144. AAAI Press.

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

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

Kocsis, L., and Szepesvári, C. 2006. Bandit Based Monte-Carlo Planning. In Fürnkranz, J.; Scheffer, T.; and Spiliopoulou, M., eds., *ECML*, volume 4212 of *LNCS*, 282–293. Springer.

Kress-Gazit, H.; Fainekos, G. E.; and Pappas, G. J. 2009. Temporal-Logic-Based Reactive Mission and Motion Planning. *IEEE Transactions on Robotics* 25(6):1370–1381.

Kurniawati, H.; Hsu, D.; and Lee, W. 2008. SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. In *Robotics: Science and Systems*, 65–72.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning Policies for Partially Observable Environments: Scaling Up. In *ICML*, 362–370.

Littman, M. L.; Dean, T. L.; and Kaelbling, L. P. 1995. On the Complexity of Solving Markov Decision Problems. In Besnard, P., and Hanks, S., eds., *UAI*, 394–402. Morgan Kaufmann.

Littman, M. L. 1996. *Algorithms for Sequential Decision Making*. Ph.D. Dissertation, Brown University.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov Decision Processes. *Mathematics of Operations Research* 12:441–450.

Poupart, P.; Malhotra, A.; Pei, P.; Kim, K.; Goh, B.; and Bowling, M. 2015. Approximate Linear Programming for Constrained Partially Observable Markov Decision Processes. In *AAAI*, 3342–3348. AAAI Press.

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

Randour, M.; Raskin, J.-F.; and Sankur, O. 2015. Variations on the Stochastic Shortest Path Problem. In D'Souza, D.; Lal, A.; and Larsen, K. G., eds., *VMCAI*, volume 8931 of *LNCS*, 1–18. Springer.

Ross, S.; Pineau, J.; Paquet, S.; and Chaib-draa, B. 2008. Online Planning Algorithms for POMDPs. *J. Artif. Intell. Res. (JAIR)* 32:663–704.

Russell, S. J., and Norvig, P. 2010. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education.

Santana, P. H. d. R. Q. e. A.; Thiébaux, S.; and Williams, B. C. 2016. RAO*: An Algorithm for Chance-Constrained POMDP's. In *AAAI*, 3308–3314. AAAI Press.

Silver, D., and Veness, J. 2010. Monte-Carlo Planning in Large POMDPs. In Lafferty, J. D.; Williams, C. K. I.; Shawe-Taylor, J.; Zemel, R. S.; and Culotta, A., eds., *Advances in Neural Information Processing Systems 23*. Curran Associates, Inc. 2164–2172.

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

Sondik, E. J. 1971. *The Optimal Control of Partially Observable Markov Processes.* Stanford University.

Spaan, M. T. J., and Vlassis, N. A. 2005. Perseus: Randomized Point-based Value Iteration for POMDPs. *J. Artif. Intell. Res. (JAIR)* 24:195–220.

Undurti, A., and How, J. P. 2010. An online algorithm for constrained POMDPs. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 3966–3973. IEEE.

White, D. 1993. Minimizing a Threshold Probability in Discounted Markov Decision Processes. *Journal of Mathematical Analysis and Applications* 173(2):634–646.

Zwick, U., and Paterson, M. 1996. The Complexity of Mean Payoff Games on Graphs. *Theoretical Computer Science* 158(1–2):343–359.