Solving Risk-Sensitive POMDPs With and Without Cost Observations

Ping Hou
Department of Computer Science
New Mexico State University
Las Cruces, NM 88003, USA
phou@cs.nmsu.edu

William Yeoh
Department of Computer Science
New Mexico State University
Las Cruces, NM 88003, USA
wyeoh@cs.nmsu.edu

Pradeep Varakantham
School of Information Systems
Singapore Management University
Singapore 188065
pradeepv@smu.edu.sg

Abstract

Partially Observable Markov Decision Processes (POMDPs) are often used to model planning problems under uncertainty. The goal in *Risk-Sensitive POMDPs* (RS-POMDPs) is to find a policy that maximizes the probability that the cumulative cost is within some user-defined cost threshold. In this paper, unlike existing POMDP literature, we distinguish between the two cases of whether costs can or cannot be observed and show the empirical impact of cost observations. We also introduce a new search-based algorithm to solve RS-POMDPs and show that it is faster and more scalable than existing approaches in two synthetic domains and a taxi domain generated with real-world data.

Introduction

In planning their daily movement strategies to get customers, most taxi drivers (Ziebart et al. 2008; Varakantham et al. 2012) are focused on reducing the risk associated with achieving a certain goal in terms of revenue. Similarly, electric car drivers (Eisner, Funke, and Storandt 2011) are more focused on reducing the risk of being stranded on a highway than trying to reach a destination several minutes earlier. Trucks deployed to clear snow in cities (Salazar-Aguilar, Langevin, and Laporte 2013) are motivated by the need to reduce the risk of not clearing snow on "important" highways and roads. Motivated by such problems, this paper focuses on risk-sensitive planning under uncertainty.

Markov Decision Processes (MDPs) and *Partially Observable MDPs* (POMDPs) have been shown to be effective models for planning under uncertainty. There is a large body of work by researchers who incorporated notions of risk in MDPs and POMDPs (Liu and Koenig 2005; 2006; Osogami 2011; 2012; Bäuerle and Rieder 2014; Marecki and Varakantham 2010). Among them, Yu, Lin, and Yan (1998) introduced *Risk-Sensitive MDPs* (RS-MDPs), which seek a policy that maximizes the probability that the cumulative cost is within some user-defined cost threshold. In this model, the *Risk-Sensitive criterion* (RS-criterion) is equivalent to having a utility function that is a step function, where an execution trajectory has no utility if its cost exceeds the threshold and a non-zero constant utility otherwise.

In this paper, we investigate goal-directed *Risk-Sensitive POMDPs* (RS-POMDPs), a generalization of RS-MDPs to the partially observable case. In the motivating domains earlier, partial observability may correspond to taxi drivers not completely observing other taxis in the same zone, electric car drivers not knowing about traffic conditions, and lack of accurate knowledge of snow conditions on the ground. Additionally, it is often not explicitly stated in the POMDP literature if agents can or cannot observe actual costs incurred during execution (Kaelbling, Littman, and Cassandra 1998). When costs can be observed, one can use them to update and get more accurate belief states, which can result in improved policies. Unfortunately, most algorithms do not explicitly use them to do so even when costs can be observed.

Towards solving goal-directed RS-POMDPs, we make the following key contributions: (*i*) We generalize Functional Value Iteration (FVI) (Marecki and Varakantham 2010), which was developed to solve finite-horizon RS-POMDPs, to now solve goal-directed RS-POMDPs with our RS-criterion; (*ii*) We propose a new exact search-based algorithm that is tailored specifically to RS-POMDPs with our RS-criterion; (*iii*) In domains where agents can observe actual costs of actions and they can provide more information than the observations, we provide a mechanism to exploit the extra information provided by observed costs; and (*iv*) Finally, we experimentally show that the new algorithm performs better than FVI in two synthetic domains and a taxi domain (Ziebart et al. 2008; Varakantham et al. 2012) generated with real-world data.

Background

A *Goal-Directed POMDP* (GD-POMDP) (Geffner and Bonet 2013) is represented as a tuple $\mathbf{P} = \langle \mathbf{S}, b_0, \mathbf{G}, \mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{\Omega}, \mathbf{O} \rangle$. It consists of a set of states $\mathbf{S}$; an initial belief state b_0; a subset $\mathbf{G} \subseteq \mathbf{S}$ of *observable* goal states; a set of actions $\mathbf{A}$; a transition function $\mathbf{T} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \to [0, 1]$ that gives the probability $T(s, a, s')$ of transitioning from state s to state s' when action a is executed; a cost function $\mathbf{C} : \mathbf{S} \times \mathbf{A} \to \mathbb{R}^+$ that gives the cost $C(s, a)$ of executing action a in state s; a set of observations $\mathbf{\Omega}$; and an observation function $\mathbf{O} : \mathbf{A} \times \mathbf{S} \times \mathbf{\Omega} \to [0, 1]$ that gives the probability $O(a, s', o)$ of receiving observation o when the action a is executed and state s' is reached. In this paper, we will focus on GD-POMDPs and will thus use the term POMDPs to refer to GD-POMDPs.

In POMDPs, a belief state b is a probability distribution over the states such that $b(s)$ is the probability of s being the actual state. We use $\mathbf{B}$ to denote the set of belief states. Let $b_a(s) = \sum_{s'} b(s')T(s', a, s)$ denote the belief state after performing action a in belief state b and $Pr(o \,|\, b, a) = \sum_s b_a(s)O(a, s, o)$ denote the probability of observing o after performing action a in belief state b. Then, $b_a^o(s) = \frac{b_a(s)O(a,s,o)}{Pr(o|b,a)}$ denotes the belief state after performing action a in belief state b and observing o.

A *POMDP policy* $\pi : \mathbf{B} \rightarrow \mathbf{A}$ is a mapping from belief states to actions. A common objective is to find a policy π^* with the minimum expected cost $\mathcal{C}^*(b_0)$, defined by

$$\mathcal{C}^*(b) = c(b, a^*) + \sum_o Pr(o \,|\, b, a^*)\mathcal{C}^*(b_{a^*}^o) \tag{1}$$

for all belief states $b \in \mathbf{B}$, where $a^* = \pi^*(b)$ is the action prescribed by policy π^* in belief state b and $c(b, a^*) = \sum_s b(s)C(s, a^*)$ is the expected cost of applying action a^* in belief state b.

Exact Algorithms: Exact POMDP algorithms (Kaelbling, Littman, and Cassandra 1998) use a finite set Γ of $|\mathbf{S}|$-dimensional real vectors. Each vector corresponds to a particular policy and each element in a vector corresponds to the expected cost $\alpha(s)$ of starting at a particular state s. Then, the expected cost of a belief state b is:

$$\mathcal{C}(b) = \min_{\alpha \in \Gamma} \sum_s b(s)\alpha(s) \tag{2}$$

The number of vectors in Γ corresponds to the number of undominated policies.[1] The algorithms iteratively perform full *Dynamic Programming* (DP) updates to update the vector set Γ, where, in each iteration k, each vector in the set represents the expected cost of policies up to time step k.

Since DP updates are done backwards in time (i.e., starting from the horizon), each policy in iteration $k+1$ will have policies in iteration k as subpolicies. If we define a function $v : \mathbf{\Omega} \rightarrow \Gamma_k$ to map each observation to the $|\mathbf{S}|$-dimensional real vector of a possible subpolicy in iteration k, and $\mathcal{V}_k$ as the set of all such functions, then the full set of possible vectors after the update in iteration $k+1$ is:

$$\Gamma_{k+1} = \{\alpha_{a,v} \,|\, a \in \mathbf{A}, v \in \mathcal{V}_k\} \tag{3}$$

where $\alpha_{a,v}(s) = C(s, a) + \sum_{s',o} T(s,a,s')O(a,s',o)v(o)(s')$. Some of these vectors may be dominated by other vectors and can be pruned by using linear program.

Point-Based Algorithms: Since the number of vectors in Γ grows exponentially in each iteration, researchers have introduced point-based algorithms (Pineau, Gordon, and Thrun 2003; Shani, Pineau, and Kaplow 2013) that update the vectors by considering only a restricted subset of belief points. In each iteration, these algorithms keep only a vector with the smallest cost for each belief point, thereby restricting the number of vectors in Γ to the number of belief points.

[1]A policy is an undominated policy if it has the smallest cost for at least one belief state.

Risk-Sensitive POMDP (RS-POMDP) Model

An RS-POMDP is defined by the tuple $\langle \mathbf{P}, \mathbf{\Theta}, \theta_0 \rangle$, where $\mathbf{P}$ is a POMDP, $\mathbf{\Theta}$ is a set of possible cost thresholds, and $\theta_0 \in \mathbf{\Theta}$ is the user-defined initial cost threshold. The objective is to find a policy π that maximizes the probability that the expected cumulative cost $c^{\mathcal{T}(s,\pi)}$ over all states s with a non-zero initial belief $b_0(s)$ is no greater than the initial cost threshold θ_0:

$$\operatorname*{argmax}_{\pi} \sum_s b_0(s) \cdot Pr(c^{\mathcal{T}(s,\pi)} \leq \theta_0) \tag{4}$$

The cumulative cost $c^{\mathcal{T}(s,\pi)}$ of a trajectory $\mathcal{T}(s, \pi) = \langle s_0 = s, s_1, s_2, \ldots \rangle$, formed by executing policy π from state s, is defined by $c^{\mathcal{T}(s,\pi)} = \sum_{t=0}^{\infty} c_t$, where c_t is the cost incurred when transitioning from s_t to s_{t+1} in time step t.

Cost Observation: It is often not explicitly stated if agents can or cannot observe the actual cost incurred during execution (Kaelbling, Littman, and Cassandra 1998). While one can imagine incorporating the observed cost in the observation function, as the observed cost depends on the predecessor state s, action a, and successor state s', the new observation function $O'(s, a, s', o')$ must include all these elements. Let $O(a, s', o)$ denote the observation function without costs, where o is the regular POMDP observation without costs. Also, let c denote the observed cost. Then, the new observation $o' = (c, o)$ and the new observation function $O'(s, a, s', o')$ should satisfy the constraint $O'(s, a, s', o') = O(a, s', o)$ if $c = C(s, a)$ and $O'(s, a, s', o') = 0$ otherwise. While it is possible to represent cost observations this way, we provide a more compact representation in this paper that separates the observed costs from the observation function.

In cases where actual costs can be observed, most POMDP algorithms do not explicitly use them to update the belief state. In some real-world applications, the actual costs can indeed be observed. For example, in our electric car example in the introduction, costs can correspond to the amount of battery power used, and the driver can observe the drop in its internal battery power. Therefore, in this paper, we distinguish between the two cases and describe solution approaches for both cases. Additionally, we assume that in both cases, the agent can accurately detect if the actual cumulative cost of its trajectory is greater than its initial cost threshold (i.e., when its resource is depleted or its deadline has passed) and it will then stop executing actions.

RS-POMDP Policy: The optimal policy for a POMDP often does not depend on the accumulated cost thus far. In contrast, an optimal policy for an RS-POMDP does depend on the accumulated cost thus far. Therefore, RS-POMDP policies need to take accumulated costs into account.

One way to do so is to incorporate costs in the belief states, which we define for the following two cases:

- If actual costs *cannot* be observed, then a belief state b is now a probability distribution over pairs (s, θ) of states $s \in \mathbf{S}$ and cost thresholds $\theta \in \mathbf{\Theta}$, which is the initial cost threshold θ_0 minus the accumulated cost thus far. Then, $b(s, \theta)$ is the probability of (s, θ) being the pair of actual state and cost threshold. We use $\mathbf{B}_{\neg c}$ to denote this set of belief states.

- If actual costs *can* be observed, then a belief state b is also a probability distribution as defined for the previous case, except that *all* pairs (s, θ) with non-zero probability $b(s, \theta) > 0$ have exactly the same cost threshold θ since costs can be observed. We use $\mathbf{B}_c$ to denote this set of belief states.

Finally, an RS-POMDP policy π is a mapping of belief states to actions, where belief states are defined as above. More specifically, it is $\pi : \mathbf{B}_{\neg c} \rightarrow \mathbf{A}$ if actual costs cannot be observed and $\pi : \mathbf{B}_c \rightarrow \mathbf{A}$ if costs can be observed.

We use $P^\pi(b)$ to denote the *reachable probability*:

$$P^\pi(b) = \sum_{s,\theta} b(s, \theta) \cdot Pr(c^{\mathcal{T}(s,\pi)} \leq \theta) \tag{5}$$

In other words, it is the probability that the accumulated cost of starting from belief state b is no larger than the corresponding cost threshold θ with policy π. Thus, in solving an RS-POMDP, the goal is to find a policy π^* such that:

$$\pi^* = \operatorname*{argmax}_\pi P^\pi(b_0) \tag{6}$$

One can represent reachable probabilities as piecewise-constant utility functions of cost thresholds θ. For example, consider a simple POMDP with $\mathbf{S} = \{s_1, s_2, s_3, s_g\}$; $\mathbf{G} = \{s_g\}$; $b_0(s_i) = P^i$ for each $s_i \notin \mathbf{G}$; and an action a that can be executed from all states $s_i \notin \mathbf{G}$ and transitions to the goal state s_g with probability 1 and cost θ^i. Assume that cost $\theta^3 > \theta_0$ exceeds the initial cost threshold, and $\theta^1 < \theta^2 < \theta_0$. Then, Figures 1(a) and 1(b) represent the utility functions of the belief b_0 at states s_1 and s_2, respectively. For example, if the agent is at state s_1 with probability $b_0(s_1) = P^1$, and it has a current cost threshold $\theta \in [\theta^1, \theta_0]$, then it receives a utility equal to the probability P^1 of reaching a goal since its action a can reach a goal with complete certainty. The utility function of the initial belief state is thus the sum of the two utility functions and is shown in Figure 1(c).

Functional Value Iteration (FVI)

FVI is an exact algorithm that can solve finite-horizon RS-POMDPs without cost observations and with arbitrary piecewise-linear non-decreasing utility functions (Marecki and Varakantham 2010). In this section, we describe how to generalize FVI to solve goal-directed RS-POMDPs with or without cost observations. Additionally, we describe how to optimize FVI for our piecewise-constant utility functions.

Recall that, in POMDPs, Γ is a set of $|\mathbf{S}|$-dimensional vectors, where each element of these vectors is the expected cost of starting at a particular state. In RS-POMDPs, Γ is also a set of $|\mathbf{S}|$-dimensional vectors, but each element of these vectors is now a piecewise-constant function that describes the reachable probability as a function of cost thresholds. In other words, each vector in Γ is a mapping $\mathbf{S} \rightarrow (\mathbf{\Theta} \rightarrow [0, 1])$, that is, a particular state $s \in \mathbf{S}$ maps to a piecewise-constant utility function, and a particular cost threshold $\theta \in \mathbf{\Theta}$ maps to a reachable probability in that utility function. Then, the reachable probability of a belief state b is:

$$P(b) = \max_{\alpha \in \Gamma} \sum_{s,\theta} b(s, \theta) \alpha(s)(\theta) \tag{7}$$

Similar to exact POMDP algorithms, FVI iteratively updates the vector set Γ until convergence. Then, the full set of possible vectors after the update in iteration $k + 1$ is still:

$$\Gamma_{k+1} = \{\alpha_{a,v} \,|\, a \in \mathbf{A}, v \in \mathcal{V}_k\} \tag{8}$$

where $\mathcal{V}_k$ is the set of functions v, and the definition of v differs in the following two cases:

- If actual costs cannot be observed, then $v : \mathbf{\Omega} \rightarrow \Gamma_k$ is similar to the one defined for POMDPs.
- If actual costs can be observed, then $v : \mathbf{C} \times \mathbf{\Omega} \rightarrow \Gamma_k$.

Finally, the update of the vectors $\alpha_{a,v}$ are different than those in POMDP algorithms as they now contain functions instead of real numbers:

- If actual costs cannot be observed, then

$$\alpha_{a,v}(s)(\theta) = \sum_{s',\theta',o} T(s, a, s')O(a, s', o)v(o)(s')(\theta') \tag{9}$$

for all $\theta \in \mathbf{\Theta}$, where $\theta' = \theta - C(s, a)$.

- If actual costs can be observed, then

$$\alpha_{a,v}(s)(\theta) = \tag{10}$$
$$\begin{cases} 0 & \text{if } c \neq C(s, a) \\ \sum_{s',\theta',o} T(s, a, s')O(a, s', o)v(c, o)(s')(\theta') & \text{otherwise} \end{cases}$$

for all $\theta \in \mathbf{\Theta}$, where $\theta' = \theta - C(s, a)$ and c is the observed cost.

We now describe how to prune dominated vectors from Γ to scale up FVI. Note that vector α_i is not dominated by other vectors if the following holds for all vectors $\alpha_j \in \Gamma$:

$$\exists b : \sum_{s,\theta} b(s, \theta) \left[\alpha_i(s)(\theta) - \alpha_j(s)(\theta)\right] \geq 0 \tag{11}$$

In order to compute $\alpha_i(s)(\theta) - \alpha_j(s)(\theta)$ efficiently for our piecewise-constant utility functions, we observe the following:

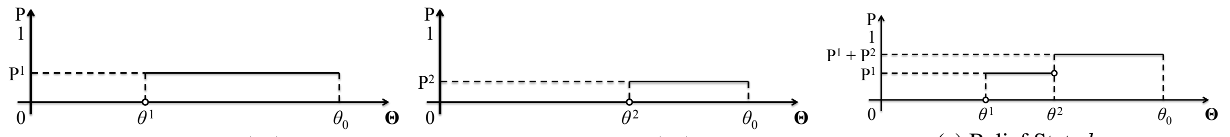
Observation 1 *Each piecewise-constant utility function can be represented by an ordered list of pairs* $\{(\theta^1, P^1), (\theta^2, P^2), \ldots, (\theta^n, P^n)\}$ *that describe the 'pieces' of the function.*

Observation 2 *For each piecewise-constant utility function, the reachable probability for all cost thresholds* $\theta \in [\theta^i, \theta^{i+1})$ *are identical.*

Therefore, instead of considering all cost thresholds θ in Equation 11, one can divide the utility functions into segments of cost thresholds $[\theta^1, \theta^2), [\theta^2, \theta^3), \ldots, [\theta^{n-1}, \theta^n)$, where, for each utility function $\alpha_i(s)$, the reachable probabilities $\alpha_i(s)(\theta) = \alpha_i(s)(\theta')$ are identical for all cost thresholds $\theta, \theta' \in [\theta^j, \theta^{j+1})$ within a segment. One can identify these segments by taking the union of all cost thresholds in the ordered list of pairs over all utility functions $\alpha_i(s)$ in all vectors $\alpha_i \in \Gamma$.

We now describe the optimizations and the pruning condition to replace Equation 11 for the following two cases:

- In the case where actual costs cannot be observed, since the computation of the difference $\alpha_i(s)(\theta) - \alpha_j(s)(\theta)$ is for the same state s, one can optimize the process above



(a) Belief $b_0(s_1)$ (b) Belief $b_0(s_2)$ (c) Belief State b_0

Figure 1: Example Piecewise-Constant Utility Functions

by computing segments for each state s in order to minimize the number of segments. Then, one can use the following condition to check for dominance:

$$\exists b : \sum_{s,k} \sum_{\theta \in [\theta_s^k, \theta_s^{k+1})} b(s,\theta) \left[\alpha_i(s)(\theta_s^k) - \alpha_j(s)(\theta_s^k)\right] \geq 0 \quad (12)$$

where θ_s^k is the start of the k-th cost threshold segment for state s. This dominance check can be implemented with a single linear program.

- In the case where actual costs can be observed, recall that for a particular belief state b, *all* pairs (s, θ) with non-zero probability $b(s, \theta) > 0$ have exactly the same cost threshold θ. Therefore, one needs to only check the following condition for that particular cost threshold θ:

$$\exists b : \sum_{s} b(s,\theta) \left[\alpha_i(s)(\theta^k) - \alpha_j(s)(\theta^k)\right] \geq 0 \quad (13)$$

where $\theta \in [\theta^k, \theta^{k+1})$ and θ^k is the start of the k-th cost threshold segment in the union of all cost threshold segments over all states. This dominance check can be implemented with n (= number of cost threshold segments) linear programs, where $\theta = \theta^k$ in the k-th linear program.

Point-Based FVI (PB-FVI): Similar to point-based POMDP algorithms, one can extend FVI to PB-FVI: In each iteration of the DP update, PB-FVI updates the vectors in Γ by considering only a restricted subset of belief points. While one can select the set of belief points arbitrarily, we select them in the same way as PBVI (Pineau, Gordon, and Thrun 2003) in our experiments.

Depth-First Search (DFS)

We now describe how to use DFS to search the reachable belief state space for an optimal policy.

Belief State Updates: In RS-POMDPs, the belief state updates are slightly different than in POMDPs because belief states now include cost thresholds:

- If actual costs cannot be observed, then, let

$$b_a(s', \theta') = \sum_{s,\theta} b(s,\theta) T(s,a,s') \quad (14)$$

denote the belief state after performing action a in belief state b, where $\theta' = \theta - C(s', a)$, and

$$b_a^o(s', \theta') = \frac{1}{Z} b_a(s', \theta') O(a, s', o) \quad (15)$$

denote the belief state after performing action a in belief state b and observing o; Z is the normalizing factor.

- If actual costs can be observed, then, let

$$b_{a,c}(s', \theta') = \sum_{s,\theta} b_{a,c}(s,\theta,s') \quad (16)$$

$$b_{a,c}(s,\theta,s') = \begin{cases} b(s,\theta)T(s,a,s') & \text{if } c = C(s,a) \\ 0 & \text{otherwise} \end{cases}$$

denote the unnormalized belief state after performing action a in belief state b and observing cost c, where $\theta' = \theta - c$, and

$$b_{a,c}^o(s', \theta') = \frac{1}{Z} b_{a,c}(s', \theta') O(a, s', o) \quad (17)$$

denote the belief state after performing action a in belief state b and observing c and o; Z is the normalizing factor.

Reachable Probability Backups: Instead of using vectors in Γ to compute reachable probabilities, one can also compute them using the system of linear equations below:

- If actual costs cannot be observed, then

$$P(b) = \max_a \sum_{s', \theta'} P(a, s', \theta') \quad (18)$$

$$P(a, s', \theta') = \begin{cases} 0 & \text{if } \theta' < 0 \\ b_a(s', \theta') & \text{if } s' \in \mathbf{G}, \theta' \geq 0 \\ \sum_o b_a(s', \theta') O(a, s', o) P(b_a^o) & \text{if } s' \notin \mathbf{G}, \theta' \geq 0 \end{cases}$$

where $P(a, s', \theta')$ is the reachable probability of an agent taking action a and transitioning to successor state s' with cost threshold θ'. For each action-successor-threshold tuple (a, s', θ'), there are the following three cases:

- If the resulting cost threshold θ' is negative, then the successor cannot be reached. Thus, the reachable probability is 0.
- If the successor s' is a goal state and the resulting cost threshold θ' is non-negative, then the goal state can be reached. Thus, the reachable probability is the belief probability $b_a(s', \theta')$. Note that there is no need for the observation function here since agents can accurately identify if they have reached goal states.
- If the successor is not a goal state and the resulting cost threshold is non-negative, then the successor can be reached. Thus, the reachable probability can be recursively computed as the belief probability $b_a(s', \theta')$ multiplied by the product of the observation probability $O(a, s', o)$ and the reachable probability of the resulting belief state $P(b_a^o)$ summed over all observations o.

- If actual costs can be observed, then

$$P(b) = \max_a \sum_{c, s', \theta'} P(a, c, s', \theta') \quad (19)$$

$$P(a, c, s', \theta') = \begin{cases} 0 & \text{if } \theta' < 0 \\ b_{a,c}(s', \theta') & \text{if } s' \in \mathbf{G}, \theta' \geq 0 \\ \sum_o b_{a,c}(s', \theta') O(a, s', o) P(b_{a,c}^o) & \text{if } s' \notin \mathbf{G}, \theta' \geq 0 \end{cases}$$

Algorithm 1: DFS(b)

```
1  $P_{a^*} \leftarrow 0$
2  for actions $a \in \mathbf{A}$ do
3      $P_a \leftarrow 0$
4      $P_a^G \leftarrow 0$
5      $b_a \leftarrow$ BELIEF_UPDATE$(b, a)$
6      for states $s' \in \mathbf{S}$ and thresholds $\theta' \in \mathbf{\Theta}$ do
7          if $s' \in \mathbf{G}$ and $\theta' \geq 0$ then
8              $P_a^G \leftarrow P_a^G + b_a(s', \theta')$
9      $P_a \leftarrow P_a + P_a^G$
10     for observations $o \in \mathbf{\Omega}$ do
11         $P_{a,o}^{\neg G} \leftarrow 0$
12         for states $s' \in \mathbf{S}$ and thresholds $\theta' \in \mathbf{\Theta}$ do
13             if $s' \notin \mathbf{G}$ and $\theta' \geq 0$ then
14                 $P_{a,o}^{\neg G} \leftarrow P_{a,o}^{\neg G} + b_a(s', \theta') \cdot O(a, s', o)$
15         if $P_{a,o}^{\neg G} > 0$ then
16             $b_a^o \leftarrow$ BELIEF_UPDATE$(b, a, o)$
17             $P_a \leftarrow P_a + P_{a,o}^{\neg G} \cdot$ DFS$(b_a^o)$
18     if $P_a > P_{a^*}$ then
19         $P_{a^*} \leftarrow P_a$
20         record action $a$ in the policy tree
21 return $P_{a^*}$
```

where $P(a, c, s', \theta')$ is the reachable probability of an agent taking action a, observing cost c, and transitioning to successor state s' with cost threshold θ'. For each action-cost-successor-threshold tuple (a, c, s', θ'), there are three cases similar to the three cases in the situation where actual costs cannot be observed.

One can extract the optimal policy by taking the action that is returned by the maximization operator in Equations 18 and 19 for each belief state b.

Algorithm Description: Algorithm 1 shows the pseudocode of the DFS algorithm for the case where actual costs cannot be observed. It uses the following variables:

- P_{a^*} stores the reachable probability for the best action a^*.
- P_a stores the reachable probability for action a.
- P_a^G stores the reachable probability of reaching a goal state with action a.
- $P_{a,o}^{\neg G}$ stores the reachable probability of reaching a non-goal state with action a and observation o.

The algorithm is implemented recursively corresponding to Equation 18, where it computes the reachable probability for all possible actions a and stores the action with the largest probability (lines 18-20). The function BELIEF_UPDATE is overloaded, where the function in lines 5 and 16 implements Equations 14 and 15, respectively.

For the case where actual costs can be observed, the algorithm needs to also loop over all possible costs c. This loop starts between lines 3 and 4 and ends between lines 17 and 18. Additionally, the BELIEF_UPDATE function includes c in its argument, and the function in lines 5 and 16 implements Equations 16 and 17, respectively. Therefore, b_a and b_a^o are actually $b_{a,c}$ and $b_{a,c}^o$, respectively, and the variables P_a^G and $P_{a,o}^{\neg G}$ should be called $P_{a,c}^G$ and $P_{a,c,o}^{\neg G}$, respectively, which are reachable probabilities for a particular cost c.

Theoretical Results

Theorem 1 *Solving RS-POMDPs optimally is PSPACE-hard in the original state space.*

PROOF SKETCH: Similar to the proof by Papadimitriou and Tsitsiklis (1987), one can reduce a *Quantified SAT* (QSAT) to an RS-POMDP. ■

Theorem 2 *Solving RS-POMDPs with negative costs is undecidable.*

PROOF SKETCH: Chatterjee et al. (2015) showed that checking for the existence of a policy that guarantees reaching a goal with an arbitrarily small expected cumulative cost is undecidable. Our RS-POMDP objective subsumes their objective and, thus, it is undecidable as well. ■

Theorem 3 *There is at least one optimal RS-POMDP policy that is both stationary and deterministic.*

PROOF SKETCH: One can create POMDPs with augmented states (s, θ) that are equivalent to RS-POMDPs. There is at least one optimal policy that is both stationary and deterministic for these POMDPs (Kaelbling, Littman, and Cassandra 1998) and, thus, it applies to RS-POMDPs as well. ■

Related Work

We now describe the related work in two different areas. The first area relates to the representation of risk through a utility function. Yu, Lin, and Yan (1998) introduced risk-sensitive MDPs (RS-MDPs), which optimizes our risk-sensitive criterion in MDPs, and a VI-based approach to solve RS-MDPs in their seminal work. Hou, Yeoh, and Varakantham (2014) followed up with improved search and dynamic programming based RS-MDP algorithms. Liu and Koenig (2006) generalized the piecewise-constant utility function to arbitrary piecewise-linear utility functions and introduced FVI to solve MDPs with these utility functions. Marecki and Varakantham (2010) extended FVI to solve POMDPs with these utility functions. Our work is the first that introduce RS-POMDP solvers that are optimized for piecewise-constant utility functions.

The second related area includes the body of work on the inclusion of reachability in the optimization criteria of MDPs and POMDPs. In this area, researchers have introduced the MAXPROB MDP, which is a goal-directed MDP with the objective of maximizing the probability of getting to a goal independent of cost (Kolobov et al. 2011; Kolobov, Mausam, and Weld 2012). Finally, Chatterjee et al. (2015) introduced a different optimization criterion for POMDPs, where they are interested in finding a policy that minimizes the expected cost but guarantees that the reachable probability of the initial belief state with this policy is 1. These criteria are different compared to the one we optimize for in RS-POMDPs.

(a) Randomly Generated POMDPs: Actual costs *cannot* be observed

	DFS			FVI w/o Pruning			FVI w/ Pruning			PB-FVI(10)			PB-FVI(100)			PB-FVI(1000)		
	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P
$\theta_0 = 1.25 \cdot \mathcal{C}^*_{det}$	100	225	0.160	10	540005	0.142	16	504695	0.126	100	2	0.136	100	26	0.160	100	1139	0.160
$\theta_0 = 1.50 \cdot \mathcal{C}^*_{det}$	100	3919	0.205	6	564001	0.169	10	540020	0.134	100	6	0.185	100	105	0.204	100	2768	0.205
$\theta_0 = 1.75 \cdot \mathcal{C}^*_{det}$	98	35241	0.227	4	576001	0.173	10	540490	0.135	100	26	0.208	100	560	0.228	100	18255	0.228
$\theta_0 = 2.00 \cdot \mathcal{C}^*_{det}$	86	132378	0.214	2	588000	0.178	4	576014	0.136	100	45	0.226	100	4088	0.247	80	171146	0.248
$\lvert\mathbf{S}\rvert = 50$	100	2627	0.208	12	528028	0.184	16	504394	0.154	100	1	0.184	100	50	0.208	100	1407	0.208
$\lvert\mathbf{S}\rvert = 100$	100	3919	0.205	6	564001	0.169	10	540020	0.134	100	6	0.185	100	105	0.204	100	2768	0.205
$\lvert\mathbf{S}\rvert = 200$	96	30390	0.113	6	564002	0.079	6	564004	0.051	100	9	0.089	100	295	0.115	100	10362	0.117
$\lvert\mathbf{S}\rvert = 400$	92	90179	0.081	2	588392	0.043	2	588024	0.021	100	31	0.055	100	2699	0.085	94	52983	0.085

(b) Randomly Generated POMDPs: Actual costs *can* be observed

	DFS			FVI w/o Pruning			FVI w/ Pruning			PB-FVI(10)			PB-FVI(100)			PB-FVI(1000)		
	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P
$\theta_0 = 1.25 \cdot \mathcal{C}^*_{det}$	100	448	0.163	6	565055	0.049	8	552567	0.061	100	24	0.124	100	264	0.161	100	2751	0.163
$\theta_0 = 1.50 \cdot \mathcal{C}^*_{det}$	100	8735	0.214	4	576023	0.037	4	576051	0.059	100	59	0.168	100	814	0.210	100	11540	0.213
$\theta_0 = 1.75 \cdot \mathcal{C}^*_{det}$	92	73447	0.231	2	588000	0.037	4	576638	0.059	100	100	0.184	100	2558	0.241	100	43451	0.248
$\theta_0 = 2.00 \cdot \mathcal{C}^*_{det}$	80	182883	0.226	2	588001	0.037	2	588036	0.059	100	206	0.201	100	7049	0.268	92	155484	0.279
$\lvert\mathbf{S}\rvert = 50$	100	7116	0.212	4	576000	0.074	12	533890	0.096	100	8	0.165	100	145	0.208	100	2014	0.212
$\lvert\mathbf{S}\rvert = 100$	100	8735	0.214	4	576023	0.037	4	576051	0.059	100	59	0.168	100	814	0.210	100	11540	0.213
$\lvert\mathbf{S}\rvert = 200$	96	41439	0.121	2	588007	0.008	4	582148	0.022	100	153	0.071	100	2076	0.111	100	43319	0.126
$\lvert\mathbf{S}\rvert = 400$	84	141332	0.081	0	600000	0.000	0	600000	0.000	100	562	0.047	100	8528	0.073	86	231325	0.092

(c) Navigation and Taxi Domains: Actual costs *cannot* be observed

	DFS			FVI w/o Pruning			FVI w/ Pruning			PB-FVI(10)			PB-FVI(100)			PB-FVI(1000)		
	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P
Navigation	80	132629	0.057	10	540000	0.018	30	421011	0.057	100	0	0.000	100	7	0.006	100	935	0.045
Taxi	50	377887	0.317	0	600000	0.000	0	600000	0.000	100	239	0.000	100	322	0.000	50	331017	0.189

(d) Navigation and Taxi Domains: Actual costs *can* be observed

	DFS			FVI w/o Pruning			FVI w/ Pruning			PB-FVI(10)			PB-FVI(100)			PB-FVI(1000)		
	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P	%	time	P
Navigation	90	111525	0.058	10	540001	0.004	20	535112	0.053	100	3	0.000	100	229	0.003	100	698	0.014
Taxi	50	349765	0.317	0	600000	0.000	0	600000	0.000	100	27	0.000	100	265	0.000	100	84257	0.041

Table 1: Experimental Results

Experimental Results

We evaluate DFS, FVI (with and without pruning) and its point-based version on three domains: (*i*) Randomly generated POMDPs; (*ii*) the *Navigation* domain from ICAPS IPPC 2011; and (*iii*) a taxi domain (Ziebart et al. 2008; Varakantham et al. 2012) generated with real-world data. We conducted our experiments on a 3.40 GHz machine with 16GB of RAM.

Randomly Generated POMDPs: We randomly generated POMDPs from 50 to 400 states, 2 actions per state, 2 successors per action, and 2 observations per action-successor pair. Each problem has exactly 1 starting state and 1 goal state. We randomly chose the costs from the range $[1, 10]$ and varied the initial cost thresholds θ_0 as a function of the accumulated cost $\mathcal{C}^*_{det}$ of the shortest *deterministic* path from any starting state in the initial belief state b_0.

Tables 1(a) and 1(b) show our results for the two cases where actual costs can and cannot be observed. There are two subtables for each case, where we set $|\mathbf{S}| = 100$ and vary θ_0 in the first subtables and set $\theta_0 = 1.50 \cdot \mathcal{C}^*_{det}$ and vary $|\mathbf{S}|$ in the second subtables. We report scalability in terms of the percentage of instances (out of 50 instances) solved; average runtime in milliseconds; and average reachable probability for each of the algorithms. If an algorithm fails to solve an instance within a time limit of 10 minutes, we take the time limit as its runtime and the reachable probability of the current solution as its reachable probability. We make the following observations:

- With increasing initial cost threshold or number of states, in general, scalability decreases and runtime increases for all algorithms. The reason is that with a larger cost threshold or a larger number of states, each algorithm has to search over a larger search space.
- When optimal solutions are found, they are better (with larger reachable probabilities) when costs can be observed. This observation can be found in the cases where DFS solves all the instances optimally. The reason for this observation is that the optimal algorithms can exploit the observed cost to more accurately update the belief states. However, this comes at the cost of larger runtimes because they need to consider more candidate policies. For example, in FVI, $|\mathcal{V}_k|$ in Equation 8 grows with the number of possible costs $|\mathbf{C}|$.
- DFS is faster and more scalable than FVI because DFS ignores non-reachable belief states while FVI does not.
- PB-FVI finds better solutions with increasing number of belief points but at the cost of increasing runtime and decreasing scalability. With 1000 belief points, PB-FVI finds close to optimal solutions for the verifiable cases where DFS also solves all instances. In some cases where DFS fails to solve all instances (e.g., $|S| = 200$), DFS finds worse solutions than PB-FVI(1000) even though DFS is an optimal algorithm and PB-FVI is not. When DFS fails

to solve an instance, its reachable probability is 0 as it does not store any full suboptimal policies. In contrast, FVI and PB-FVI may have positive reachable probabilities since they iteratively improve their policies.
- Finally, pruning improves the scalability of FVI.

Navigation and Taxi Domains: For the navigation domain, we use all 10 IPPC instances, but we changed the costs to randomly vary from $[1, 10]$ to show the impact of cost observations. For the taxi domain, states are composed of the tuple $\langle$zone z, time interval t, hired rate level $p_{z,t}\rangle$, where there are 10 zones, each time interval is 30 minutes long, and the hired rate level is either high (= 0.75) or low (= 0.25). Each taxi has two types of actions: (a_1) move to a zone and (a_2) look for passengers in its zone. Taxis executing a_1 will move to their desired zone with probability 1 and 0 reward. Taxis executing a_2 have probability $p_{z,t}$ of successfully picking up a passenger and they can accurately observe $p_{z,t}$ with probability 0.8. If it fails to pick up a passenger, it ends up in the same zone with 0 reward. The probability $p_{z,t}$; the transition function, which determines which zone a hired taxi moves to; and the reward function, which determines the reward of the hired taxi, is generated with real-world data. We average our results over 10 instances. In both domains, we set $\theta_0 = 1.50 \cdot \mathcal{C}^*_{det}$.

Tables 1(c) and 1(d) show our results. In general, all the observations from the randomly generated POMDPs apply here as well except for the following: PB-FVI solves all instances but does not find good quality solutions. In fact, it finds infeasible solutions when the number of points is small. This observation highlights the fact that PB-FVI's behavior is highly dependent on domain structure.

Conclusions

We investigated RS-POMDPs, where we distinguishe between the two cases of whether costs can or cannot be observed. We also introduced a new RS-POMDP algorithm based on DFS and generalize FVI to solve RS-POMDPs with and without cost observations. Our experimental results show that DFS scales better than FVI on two synthetic domains and a taxi domain generated with real-world data.

Acknowledgments

This research is partially supported by NSF grant 1345232. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, or the U.S. government.

References

Bäuerle, N., and Rieder, U. 2014. More risk-sensitive Markov decision processes. *Mathematics of Operations Research* 39(1):105–120.

Chatterjee, K.; Chmelik, M.; Gupta, R.; and Kanodia, A. 2015. Optimal cost almost-sure reachability in POMDPs. In *Proc. of AAAI*, 3496–3502.

Eisner, J.; Funke, S.; and Storandt, S. 2011. Optimal route planning for electric vehicles in large networks. In *Proc. of AAAI*, 1108–1113.

Geffner, H., and Bonet, B. 2013. *A Concise Introduction to Models and Methods for Automated Planning*. Morgan & Claypool Publishers.

Hou, P.; Yeoh, W.; and Varakantham, P. 2014. Revisiting risk-sensitive MDPs: New algorithms and results. In *Proc. of ICAPS*, 136–144.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1–2):99–134.

Kolobov, A.; Mausam; Weld, D. S.; and Geffner, H. 2011. Heuristic search for generalized stochastic shortest path MDPs. In *Proc. of ICAPS*, 130–137.

Kolobov, A.; Mausam; and Weld, D. S. 2012. A theory of goal-oriented MDPs with dead ends. In *Proc. of UAI*, 438–447.

Liu, Y., and Koenig, S. 2005. Risk-sensitive planning with one-switch utility functions: Value iteration. In *Proc. of AAAI*, 993–999.

Liu, Y., and Koenig, S. 2006. Functional value iteration for decision-theoretic planning with general utility functions. In *Proc. of AAAI*, 1186–1193.

Marecki, J., and Varakantham, P. 2010. Risk-sensitive planning in partially observable environments. In *Proc. of AAMAS*, 1357–1368.

Osogami, T. 2011. Iterated risk measures for risk-sensitive Markov decision processes with discounted cost. In *Proc. of UAI*, 573–580.

Osogami, T. 2012. Robustness and risk-sensitivity in Markov decision processes. In *Proc. of NIPS*, 233–241.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3):441–450.

Pineau, J.; Gordon, G. J.; and Thrun, S. 2003. Point-based value iteration: An anytime algorithm for POMDPs. In *Proc. of IJCAI*, 1025–1032.

Salazar-Aguilar, A.; Langevin, A.; and Laporte, G. 2013. The synchronized arc and node routing problem: Application to road marking. *Computers & Operations Research* 40(7):1708–1715.

Shani, G.; Pineau, J.; and Kaplow, R. 2013. A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems* 27(1):1–51.

Varakantham, P.; Cheng, S.; Gordon, G. J.; and Ahmed, A. 2012. Decision support for agent populations in uncertain and congested environments. In *Proc. of AAAI*.

Yu, S.; Lin, Y.; and Yan, P. 1998. Optimization models for the first arrival target distribution function in discrete time. *Journal of Mathematical Analysis and Applications* 225:193–223.

Ziebart, B.; Maas, A.; Dey, A.; and Bagnell, J. A. 2008. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *Proc. of Ubicomp*, 322–331.