Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling (ICAPS 2019)

An Exact Algorithm to Make a Trade-Off between Cost and Probability in SSPs

Valdinei Freire,[1] **Karina Valdivia Delgado,**[1] **Willy Arthur Silva Reis**[1]
[1]University of Sao Paulo
valdinei.freire@usp.br, kvd@usp.br, willy.reis@usp.br

Abstract

In stochastic sequential decision problems, such as Stochastic Shortest Path Problems, the GUBS (Goal with Utility-Based Semantics) criterion considers a trade-off between probability-to-goal and cost-to-goal using a goal semantics based on Expected Utility Theory (EUT); in such a semantics, goal paths have priority over non-goal paths, but it implies neither the MAXPROB criterion nor the dual optimization criterion that finds the cheapest policy among the policies that maximize goal probability. Whereas evaluation criteria based on a sound theory such as EUT are desirable, optimal policies under GUBS are non-markovian. Non-markovian solutions are undesirable because there is not always a finite representation and even if it can be represented in a finite way, the representation may be too large to be stored. Here we define a special case of GUBS criterion that allows a finite representation to the optimal policy, the eGUBS criterion, where the cost utility function is exponencial. Considering this special case, we contribute with: (*i*) the proof that the eGUBS-optimal policy has a finite representation; (*ii*) the first exact algorithm to obtain finite optimal policies for the eGUBS criterion, and (*iii*) four strategies to find sub-optimal policies. We conduct experiments on one synthetic problem to evaluate each strategy. Although optimal solutions have a high memory cost, sub-optimal policies can save memory space with a small decrease in performance.

1 Introduction

Stochastic Shortest Paths Problems (SSPs) (Bertsekas and Tsitsiklis 1991) have become the standard model for probabilistic planning. SSP models the interaction between an agent and its environment. The agent selects and executes an action (that has probabilistic outcomes) which has a cost and leads the agent to a next state. The agent's objective is to reach the goal while minimizing the expected cost through a sequence of actions. The solution for an SSP is a deterministic policy that maps states to actions.

While many works in the literature about SSPs assume that at least one policy exists that guarantees to reach the goal (Bertsekas and Tsitsiklis 1991; Bonet and Geffner 2005; 2003), some works deal with problems where there are dead ends and therefore a goal can not be reached with probability one.

For SSPs with dead ends, some research has focused only on finding policies that maximize the probability of reaching a goal (MAXPROB criterion) (Kolobov et al. 2011; Teichteil-Königsbuch, Kuter, and Infantes 2010; Camacho, Muise, and McIlraith 2016), while other approaches work with two criteria: maximizing the probability of reaching a goal and minimizing the average accumulated costs of reaching a goal (Teichteil-Königsbuch 2012; Kolobov, Mausam, and Weld 2012; Trevizan, Teichteil-Königsbuch, and Thiébaux 2017).

Two approaches that work with these two conflicting criteria are iSSPUDE (Kolobov, Mausam, and Weld 2012) and S^3P (Teichteil-Königsbuch 2012) that consider a dual optimization criterion, which finds the cheapest policy among the policies that maximize goal probability. This dual criterion prioritizes the probability of success and the performance is treated as secondary. Another criterion is the Min-Cost given Max-Prob (MCMP) that was proposed in (Trevizan, Teichteil-Königsbuch, and Thiébaux 2017) and which results in the minimum expected cost policy among those with maximum success probability. However, these criteria do not really establish a compromise between probability-to-goal and cost-to-goal.

To deal with this limitation, Freire and Delgado (2017) proposed the GUBS (Goals with Utility-Based Semantics) criterion to evaluate policies; the GUBS criterion defines a trade-off between cost-to-goal and probability-to-goal for SSPs with dead ends based on the expected utility theory (EUT). This criterion derives a parametric model from a set of axioms and describe how to set the parameters rationally. Freire and Delgado (2017) also proposed an approximate algorithm to solve discrete-cost SSPs based on the GUBS criterion. Evaluation criteria based on normative theories such as EUT are sound and desirable, but optimal policies under GUBS are non-markovian. Non-markovian solutions is a concern for planning algorithms; first, it is not always the case that a finite representation is possible; second, even if it can be represented in a finite way, the representation may be impractical.

In this paper, considering a special case of the GUBS criterion where exponential cost utility is used, we present the first exact algorithm to obtain finite optimal policies for SSPs with dead ends. Even if a finite optimal solution exists, it might be impractical storing such a solution; therefore, we also design four sub-optimal strategies to save memory space.

2 Stochastic Shortest Path Problem

In the discussion that follows, we consider Stochastic Shortest Path Problem (Bertsekas and Tsitsiklis 1991) described by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, c, \mathcal{G} \rangle$ where: $\mathcal{S}$ is a finite set of states; $\mathcal{A}$ is a finite set of actions that can be performed at each period of decision $t \in \{0, 1, 2, \ldots\}$; $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is a transition function that represents the probability of state $s' \in \mathcal{S}$ be reached after the agent executes an action $a \in \mathcal{A}$ in a state $s \in \mathcal{S}$, i.e., $\Pr(s_{t+1} = s' | s_t = s, a_t = a) = P(s, a, s')$; $c : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}^+$ is a positive cost function that represents the cost of executing an action $a \in \mathcal{A}$ in a state $s \in \mathcal{S}$, i.e., $c_t = c(s_t, a_t)$; and $\mathcal{G}$ is a non-empty set of goal states that are absorbing (states with no outgoing transitions).

An SSP defines a decision process, where the agent starts in state s_0 and at any time step t the agent: observes the current state s_t, chooses an action a_t, pays the cost $c_t = c(s_t, a_t)$, and the process transits to s_{t+1} by following conditional probability distribution $\Pr(s_{t+1} = s' | s_t = s, a_t = a) = P(s, a, s')$.

Under an SSP and the agent policy π to take decisions, we consider the following random variables: (*i*) history $H^\pi_T = \langle s_0, a_0, c_0, s_1, a_1, c_1, \ldots, s_T \rangle$; (*ii*) accumulate cost $C^\pi_T = \sum_{t=0}^{T} c_t$; and (*iii*) goal indicator $\beta^\pi_T = 1$ if $\{\exists t \leq T | s_t \in \mathcal{G}\}$ and $\beta^\pi_T = 0$ otherwise. The agent is supposed to act by minimizing accumulated cost and maximizing the probability to reach a goal state.

The probability-to-goal of a policy π starting in state $s_0 = s$ is given by:

$$P^\pi_G(s) = \lim_{T \to \infty} \mathrm{E}[\beta^\pi_T | s_0 = s]. \tag{1}$$

The cost-to-goal of a policy π starting in state $s_0 = s$ is given by $\overline{C}^\pi_G(s) = \lim_{T \to \infty} \mathrm{E}[C^\pi_T | s_0 = s, \beta^\pi_T = 1]$.

The solution to an SSP consists in a policy π. A policy π can assume different descriptions and we list some important ones bellow:

- Stationary policies ($\pi : \mathcal{S} \rightarrow \mathcal{A}$): a stationary policy maps every state s_t into an action $a_t = \pi(s_t)$;
- Non-Markovian policies ($\pi : \mathcal{H} \rightarrow \mathcal{A}$): a non-Markovian policy maps every history H_t at step t into an action $a_t = \pi(H_t)$; and
- Accumulated-cost augmented policies ($\pi : \mathcal{S} \times \mathcal{R} \rightarrow \mathcal{A}$): a non-Markovian policy that maps every accumulated cost C_t at step t and state s_t into an action $a_t = \pi(s_t, C_t)$.

2.1 The GUBS Criterion

An agent should strive for reaching goal state while avoiding high cost, but it is not clear how to establish a trade-off between both objectives. The GUBS criterion (Freire and Delgado 2017) formalizes optimal policies by: (*i*) considering preference axioms that prioritize goal histories over non-goal histories; and (*ii*) considering EUT to average over the random variables C^π_T and β^π_T.

Definition 1 (**The GUBS criterion** (Freire and Delgado 2017))**.** *Let $u(\cdot)$ be a utility function over cost, K_g be a constant utility for reaching the goal. The* GUBS *criterion is defined by the following utility function:*

$$U(C_T, \beta_T) = u(C_T) + K_g \beta_T,$$

and a policy π using this criterion is evaluated by:

$$V^\pi(s) = E[U(C_T, \beta_T) | \pi, s_0 = s].$$

Theorem 1 (**GUBS conditions** (Freire and Delgado 2017))**.** *The model* GUBS *guarantees the decision-maker prioritizes goals histories over non-goal histories under the following conditions:*

1. *$u : \mathcal{R} \rightarrow [U_{min}, U_{max}]$;*
2. *$u(C)$ is strictly decreasing in C; and*
3. *$K_g > U_{max} - U_{min}$.*

where $U_{min}, U_{max} \in \mathcal{R}$.

Different from iSSPUDE (Kolobov, Mausam, and Weld 2012), S^3P (Teichteil-Königsbuch 2012) and MCMP (Trevizan, Teichteil-Königsbuch, and Thiébaux 2017), the GUBS criterion accepts a policy π' to be preferred to another policy π ($\pi' \succ \pi$), even if $P^{\pi'}_G < P^\pi_G$ provided that the cost-to-goal in π' make up for the loss in the probability-to-goal. Theorem 2 describes a necessary conditions such that $\pi' \succ \pi$.

Theorem 2 (**GUBS Mininum Trade-off** (Freire and Delgado 2017))**.** *Consider an SSP and two policies π and π' such that $P^\pi_G > P^{\pi'}_G$, then, under the* GUBS *model, $\pi' \succ \pi$ only if:*

$$\frac{P^{\pi'}_G}{P^\pi_G} > \frac{K_g}{U_{max} - U_{min} + K_g}.$$

Although the GUBS criterion allows an adequate trade-off between probability-to-goal and cost-to-goal, finding an optimal policy for GUBS has three main problems. First, an optimal solution is non-Markovian in the accumulated cost. Second, just like finite horizon MDP, optimal policy must be obtained backwards. Third, we may require too much memory space to store an optimal policy.

2.2 Risk Sensitive SSP and Exponential Utility

In this work, we consider an *exponential utility function* based on risk-sensitive SSPs (RS-SSP). A Risk Sensitive SSP (RS-SSP) (Patek 2001) is defined by the tuple $\mathcal{RSSSP} = \langle \mathcal{M}, \lambda \rangle$ where $\mathcal{M}$ is an SSP and λ is the risk factor that models the agent's risk-attitude. The exponential utility function u used by the RS-SSP is:

$$u(x) = -\operatorname{sgn}(\lambda) e^{\lambda x},$$

where $\operatorname{sgn}(\lambda)$ is a function that returns the sign of λ, and $x = \sum_{t=0}^{M} c(s_t, \pi(s_t))$, where $M \in \mathcal{N}$ is the time step when the goal is reached and $M = \infty$ if the goal is not reached. If $\lambda < 0$ the agent considers a risk-prone attitude, if $\lambda > 0$

the agent considers a risk-averse attitude and in the limit if $\lambda \to 0$ the agent considers a risk-neutral attitude.

If $\lambda < 0$, the value function of a policy π in RS-SSPs is well-defined and is given by:

$$V_\lambda^\pi(s) = \lim_{T\to\infty} \mathrm{E}\left[-\operatorname{sgn}(\lambda)e^{\lambda C_T^\pi}|s_0 = s\right],$$

and the value of a policy π can be computed by solving the following system of equations:

$$V_\lambda^\pi(s) = \begin{cases} -\operatorname{sgn}(\lambda), & \text{if } s \in \mathcal{G} \\ e^{\lambda c(s,\pi(s))} \sum_{s'\in\mathcal{S}} P(s,\pi(s),s')V^\pi(s'), & \\ & \text{otherwise} \end{cases} \tag{2}$$

3 Exponential GUBS

To use the GUBS criterion, we must define a bounded utility function $u(\cdot)$ and a constant K_g. Both must be elicited from a user whom the agent takes action in the name of. The exponential utility function satisfies conditions 1 and 2 of Theorem 1 and as we have seen in the previous section, presents good convergence properties when the risk factor is negative. Therefore, in this paper we consider Exponential GUBS (eGUBS), a restricted GUBS criterion where the utility function $u(\cdot)$ is exponential and the risk factor is negative. The eGUBS criterion has only two parameters to be defined: λ and K_g.

Definition 2 (The eGUBS criterion). *The eGUBS criterion considers the GUBS criterion where $u(x) = e^{\lambda x}$ and $\lambda < 0$.*

To solve the GUBS criterion, it is necessary to augment the state space with the accumulate cost C_t and consider a State-Cost Value Function. However, the accumulate cost may be infinite and, when using dynamic programming, optimality is obtained backwards.

Definition 3 (State-Cost Value Function). *Consider that the agent has already paid cost C and is in the state s, then follows policy π from now on. The State-Cost Value Function is defined by:*

$$V^\pi(s,C) = E[U(C + C_T, \beta_T)|\pi, s_0 = s],$$

and describes the expected utility obtained by the agent.

Theorem 3 (State-Cost Value of Stationary Policies to eGUBS). *Consider a stationary policy π and the eGUBS criterion, then the State-Cost Value Function of π is:*

$$V^\pi(s,C) = e^{\lambda C}V_\lambda^\pi(s) + K_g P_G^\pi(s),$$

where $V_\lambda^\pi(s)$ and $P_G^\pi(s)$ were defined in Equations 2 and 1, respectively.

Proof.

$$\begin{aligned} V^\pi(s,C) &= \mathrm{E}[U(C + C_T, \beta_T)|\pi, s_0 = s] \\ &= \mathrm{E}[u(C + C_T^\pi) + K_g\beta_T^\pi|s_0 = s] \\ &= \mathrm{E}[e^{\lambda(C+C_T^\pi)} + K_g\beta_T^\pi|s_0 = s] \\ &= e^{\lambda C}V_\lambda^\pi(s) + K_g P_G^\pi(s) \end{aligned}$$

□

Since solutions to SSPs with the GUBS criterion are non-Markovian policies, it may be the case that a stationary policy can be improved by augmenting the state space with the accumulated cost. Definition 4 describes how to improve a stationary policy, while Theorem 4 describe sufficient conditions to a stationary policy being optimal.

Definition 4 (Look-Ahead Policy Improvement to eGUBS criterion). *Given a stationary policy π, such a policy can be improved into a non-markovian policy π', for all $s \in \mathcal{S}$, by:*

$$\pi'(s,C) = \arg\max_{a\in\mathcal{A}} \left\{ \sum_{s'\in S} P(s,a,s') \times \left(e^{\lambda(C+c(s,a))}V_\lambda^\pi(s') + K_g P_G^\pi(s')\right)\right\}.$$

Theorem 4 (C_{max} and Optimal Stationary Policy to eGUBS). *In the eGUBS criterion, there exists a value C_{max} such that if accumulated cost reach C_{max}, the optimal policy from that point is stationary. Such a stationary policy π^* maximizes a lexicographic dual criterion:*

- *π^* maximizes probability to goal, i.e., $P_G^{\pi^*}(s) \geq P_G^\pi(s) \forall s \in \mathcal{S}, \pi \in \Pi$, where Π is the set of stationary policies; and*
- *π^* maximizes the expected exponential utility, i.e.,*

$$V_\lambda^{\pi^*}(s) \geq V_\lambda^\pi(s) \forall s \in \mathcal{S}, \pi \in \Pi^*,$$

where Π^* is the set of stationary policies that maximizes probability to goal.

Such a value C_{max} is given by:

$$C_{max} = \max_{(s,a)\in\mathcal{X}} \{W\}$$

where:

$$W = -\frac{1}{\lambda} \log \frac{V_\lambda^{\pi^*}(s) - \sum_{s'\in S} P(s,a,s')e^{\lambda c(s,a)}V_\lambda^{\pi^*}(s')}{K_g\left(\sum_{s'\in S} P(s,a,s')P_G^{\pi^*}(s') - P_G^{\pi^*}(s)\right)}$$

and

$$\mathcal{X} = \left\{(s \in \mathcal{S}, a \in \mathcal{A}) \middle| (V_\lambda^{\pi^*}(s) - \sum_{s'\in S} P(s,a,s')e^{\lambda c(s,a)}V_\lambda^{\pi^*}(s')) < 0\right\}.$$

Proof. A policy π^* is optimal if it cannot be improved through look-ahead, i.e., it does not exist any state s that can be improved by using an action $a \neq \pi^*(s)$. Since π^* optimizes probability-to-goal, only pairs in $(s,a) \in \mathcal{X}$, that improve on expected exponential utility are candidate to improve policy π^*. Therefore, for policy π^* to be optimal, the following constraint must be observed for all pair $(s, a) \in \mathcal{X}$:

$$\begin{aligned}
&e^{\lambda C} V_\lambda^\pi(s) + K_g P_G^\pi(s) \geq \\
&\quad \sum_{s' \in \mathcal{S}} P(s, a, s') \left(e^{\lambda(C + c(s,a))} V_\lambda^\pi(s') + K_g P_G^\pi(s') \right) \\
&e^{\lambda C} \left(V_\lambda^\pi(s) - \sum_{s' \in \mathcal{S}} P(s, a, s') e^{\lambda c(s,a)} V_\lambda^\pi(s') \right) \geq \\
&\quad \sum_{s' \in \mathcal{S}} P(s, a, s') K_g P_G^\pi(s') - K_g P_G^\pi(s) \\
&\frac{V_\lambda^\pi(s) - \sum_{s' \in \mathcal{S}} P(s, a, s') e^{\lambda c(s,a)} V_\lambda^\pi(s')}{\sum_{s' \in \mathcal{S}} P(s, a, s') K_g P_G^\pi(s') - K_g P_G^\pi(s)} \leq e^{-\lambda C}.
\end{aligned}$$

While the first passage is only simple mathematical manipulations, in the second passage we take advantage of the inequality $\sum_{s' \in \mathcal{S}} P(s, a, s') K_g P_G^\pi(s') - K_g P_G^\pi(s) < 0$. Applying log in both sides of the inequality, we obtain:

$$\begin{aligned}
&\log \frac{V_\lambda^\pi(s) - \sum_{s' \in \mathcal{S}} P(s, a, s') e^{\lambda c(s,a)} V_\lambda^\pi(s')}{\sum_{s' \in \mathcal{S}} P(s, a, s') K_g P_G^\pi(s') - K_g P_G^\pi(s)} \leq -\lambda C \\
&C \geq -\frac{1}{\lambda} \log \frac{V_\lambda^\pi(s) - \sum_{s' \in S} P(s, a, s') e^{\lambda c(s,a)} V_\lambda^\pi(s')}{K_g \left(\sum_{s' \in S} P(s, a, s') P_G^\pi(s') - P_G^{\pi^*}(s) \right)}
\end{aligned}$$

□

In the next section we present an exact algorithm to solve SSPs that use the eGUBS criterion. Definition 4 motivates to find a stationary policy (Algorithm 1) and lift it to a piecewise-stationary policy (Algorithm 2), and to do this we define an accumulated cost schedule. The results of Theorem 4 then allow us to discuss under what conditions the obtained policy is eGUBS-optimal (Theorem 6).

4 An Exact Algorithm to the eGUBS Criterion

In this section we contribute with the first exact algorithm to solve SSPs that use the GUBS criterion by considering the special case when utility function is exponential. Such algorithm consists of three phases:

1. Finding an optimal *Risk Sensitive Dual Criterion policy* π;
2. Defining an *accumulated-cost schedule* that is a non-negative strictly monotonic increasing sequence, i.e., an ordered set that determines points in the accumulated cost where policy (value) can be altered; and
3. Finding a finite horizon optimal policy π^* for SSPs with the eGUBS criterion.

Next we contribute with algorithms to phase 1 (Risk-Sensitive Dual Criterion algorithm), and phase 3 (eGUBS Value Iteration algorithm, eGUBS-VI). Then, regarding accumulated-cost schedule, we discuss in Section 4.3 conditions for the optimality of eGUBS-VI algorithm.

4.1 Finding an Optimal Risk-Sensitive Dual Criterion Policy

In this section we define the Risk-Sensitive Dual Criterion and describe the algorithm to compute the optimal policy for this criterion.

Definition 5 (**The Risk-Sensitive Dual Criterion**). *The Risk-Sensitive Dual Criterion considers a lexicographic criterion, where for all $s \in \mathcal{S}$:*

(i) *if $P_G^\pi(s) > P_G^{\pi'}(s)$, then $\pi \succ \pi'$; or*

(ii) *if $P_G^\pi(s) = P_G^{\pi'}(s)$ and $V_\lambda^\pi > V_\lambda^{\pi'}$, then $\pi \succ \pi'$.*

Given an SSP, a risk factor λ and a minimum error ϵ, Algorithm 1 computes $V_\lambda(\cdot)$, $P_G(\cdot)$ and the optimal policy $\pi(\cdot)$ for the Risk-Sensitive Dual Criterion. In line 7 the algorithm computes the maximum probability, in line 8 computes the set of actions that are MAXPROB for each state, and in line 9 computes the value function for the risk-sensitive criterion considering these actions. The stop criterion of this algorithm depends on δ_1 and δ_2. The algorithm continues updating $V_\lambda(\cdot)$ and $P_G(\cdot)$ while δ_1, the maximum sum of the difference of the values and the difference of the probabilities, is greater than ϵ, or the difference of using another action that is not MAXPROB ($a \in \mathcal{A} \setminus A(s)$) is better than using the MAXPROB action ($\delta_2 \leq 0$). This last condition guarantees that even if ϵ is inappropriate, the result of Algorithm 1 can still be used on Theorem 4.

Theorem 5 (**RS-Dual Algorithm Convergence**). *The Risk-Sensitive Dual Criterion algorithm converges.*

Proof sketch. Note that, both operators (lines 7 and 9) are contractions. Since $P_G(s)$ is updated with null cost, $P_G(s)$ are bounded by initial values 1 of goal states. Because $\lambda < 0$ and $c(s, a) > 0$, $e^{\lambda c(s,a)}$ works as discount, which guarantees convergence (Minami and Silva 2012); since every history that does not reach goals has infinite cost and value 0. □

4.2 eGUBS-VI Algorithm

The eGUBS-VI algorithm is presented in Algorithm 2. eGUBS-VI algorithm consider two accumulated-cost schedule: $\mathcal{C}$, the policy schedule, and $\mathcal{D}$, the value schedule. eGUBS-VI algorithm constructs a piecewise-stationary policy $\pi^* : \mathcal{S} \times \mathcal{C} \to \mathcal{A}$ based on an accumulated-cost schedule $\mathcal{C}$ such that:

$$a_t = \pi^*(s_t, C_{next}(C_t, \mathcal{C})),$$

where $C_{next}(C_t, \mathcal{C}) = \min\{D | D \in \mathcal{C} \cup \{C_{max}\}$ and $D \geq C_t\}$. Because $\mathcal{C}$ is used to define a policy, we call it *policy schedule*.

We make use of Theorem 4 to calculate a finite horizon $C_{max}(\mathcal{M}, K_g, \lambda, V_\lambda, P_G, \pi)$. eGUBS-VI algorithm keeps track of three value functions based on an accumulated-cost schedule $\mathcal{D}$ (we call it *value schedule*):

- $V_\lambda^*(s, C)$ is the value function of the risk-sensitive criterion starting at state s and following policy π^* from the accumulate cost C;

Algorithm 1 Risk-Sensitive Dual Criterion Algorithm.

```
1: Input: SSP $\mathcal{M}$, risk factor $\lambda$, minimum error $\epsilon$
2: Initialize: $\delta_1 \leftarrow +\infty$, $\delta_2 \leftarrow 0$, $V_\lambda(s) \leftarrow 0$ for all $s \in \mathcal{S} \setminus \mathcal{G}$, $V_\lambda(s) \leftarrow -\operatorname{sgn}(\lambda)$ for all $s \in \mathcal{G}$, $P_G(s) \leftarrow 0$ for all $s \in \mathcal{S} \setminus \mathcal{G}$, and $P_G(s) \leftarrow 1$ for all $s \in \mathcal{G}$
3: while $\delta_1 \geq \epsilon$ or $\delta_2 \leq 0$ do
4:     $V' \leftarrow V_\lambda$
5:     $P' \leftarrow P_G$
6:     for all $s \in \mathcal{S} \setminus \mathcal{G}$ do
7:         $P_G(s) \leftarrow \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} P(s,a,s')P'(s')$
8:         $A(s) \leftarrow \left\{ a \,\middle|\, P_G(s) = \sum_{s' \in \mathcal{S}} P(s,a,s')P'(s') \right\}$
9:         $V_\lambda(s) \leftarrow \max_{a \in A(s)} \left\{ e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s')V'(s') \right\}$
10:    end for
11:    $\delta_1 \leftarrow \max_{s \in \mathcal{S}} \left\{ |V_\lambda(s) - V'(s)| + |P_G(s) - P'(s)| \right\}$
12:    $\delta_2 \leftarrow \min_{s \in \mathcal{S}, a \in \mathcal{A} \setminus A(s)} \{P_G(s) - \sum_{s' \in \mathcal{S}} P(s,a,s')P_G(s')\}$
13: end while
14: for all $s \in \mathcal{S}$ do
15:    $\pi(s) \leftarrow \arg\max_{a \in A(s)} \left\{ e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s')V'(s') \right\}$
16: end for
17: return $V_\lambda(\cdot), P_G(\cdot), \pi(\cdot)$
```

- $P_G^*(s, C)$ is the probability to goal starting at state s and following policy π^* from the accumulate cost C; and
- $V^*(s, C)$ is the value function for the eGUBS criterion.

eGUBS-VI is a dynamic programming algorithm; it starts from the risk-sensitive dual criterion policy π (line 2) and do backup updates from D_N until D_1 (line 3) on the three value functions V_λ^* (line 17), P_G^* (line 18), and V^* (line 19). If $C \in \mathcal{C}$, then C is a point of policy improvement and a stationary policy is generated (lines 11 and 12, Definition 4); otherwise the current stationary policy is considered (lines 14 and 15).

4.3 Theoretical Results about eGUBS-VI Optimality

In this section we present the sufficient conditions for eGUBS-VI algorithm returning the optimal policy. First, we consider the case when the cost is rational.

Theorem 6 (**Rational cost** - Optimal policy). *Consider that the cost function has an image in the rational numbers, i.e., $c : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{I} \subset \mathcal{Q}$ and that all values $r_i \in \mathcal{I}$ has a quotient $r_i = \frac{p_i}{q}$ representation with the same denominator q (if it is not the case, it can be obtained by calculating the least common multiplier). Finally, let p be the greatest common divisor of the numerator of $\mathcal{I}$.*

Let

- *π be the optimal policy for the risk-sensitive dual criterion on the SSP $\mathcal{M}$;*
- *V_λ be the accumulated exponential cost of π,*

Algorithm 2 eGUBS-VI Algorithm.

```
1: Input: SSP $\mathcal{M}$, risk factor $\lambda$, goal terminal reward $K_g$, accumulate exponential cost $V_\lambda$, probability to goal $P_G$, Risk-Sensitive dual-criterion policy $\pi$, value schedule $\mathcal{D} = \{D_1 = 0, D_2, \cdots, D_N, \infty\}$ where $D_i < D_j \Leftrightarrow i < j$, policy schedule $\mathcal{C} = \{C_1, C_2, \cdots, C_M, \infty\}$ where $C_i < C_j \Leftrightarrow i < j$ and $\mathcal{C} \subseteq \mathcal{D}$
2: Initialize: $V_\lambda^*(s, \infty) = V_\lambda(s)$, $P_G^*(s, \infty) = P_G(s)$ and $\pi^*(s, \infty) = \pi(s)$ for all $s \in \mathcal{S}$
3: for $C = D_N$ down to $D_1$ do
4:     for $s \in \mathcal{S}$ do
5:         for $a \in \mathcal{A}$ do
6:             $C' = \min\{D | D \in \mathcal{D}$ and $D \geq C + c(s,a)\}$
7:             $V(a) \leftarrow e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s')V_\lambda^*(s', C')$
8:             $P(a) \leftarrow \sum_{s' \in \mathcal{S}} P(s,a,s')P_G^*(s', C')$
9:         end for
10:        if $C \in \mathcal{C}$ then
11:            $\pi^*(s, C) = \arg\max_{a \in \mathcal{A}} e^{\lambda C} V(a) + K_g P(a)$
12:            $a^* \leftarrow \pi^*(s, C)$
13:        else
14:            $C' = \min\{D | D \in \mathcal{C}$ and $D \geq C\}$
15:            $a^* \leftarrow \pi^*(s, C')$
16:        end if
17:        $V_\lambda^*(s, C) = V(a^*)$
18:        $P_G^*(s, C) = P(a^*)$
19:        $V^*(s, C) = V_\lambda^*(s, C) + K_g P_G^*(s, C)$
20:    end for
21: end for
22: return $\pi^*(\cdot), V^*(\cdot)$
```

- *P_G be the probability to goal of π;*
- *C_{max} be like in Theorem 4; and*
- *$\mathcal{C} = \mathcal{D} = \left\{ 0, \frac{p}{q}, \frac{2p}{q}, \frac{3p}{q}, \ldots, C_{max}, \infty \right\}$;*

then, the input $(\mathcal{M}, \lambda, K_g, V_\lambda, P_G, \pi, \mathcal{D}, \mathcal{C})$ is sufficient for eGUBS-VI algorithm returning the optimal policy.

Proof sketch. First, from Theorem 4 and initialization of $\pi^*(s, \infty)$, it is true that π^* is optimal for $C > C_{max}$. Second, from Theorem 3, it is also true that the three value functions V_λ^*, P_G^*, and V^* are correct for $C > C_{max}$. Third, the value schedule $\mathcal{D}$ and policy schedule $\mathcal{C}$ have all the reachable accumulated cost. Therefore, because cost is always positive and eGUBS-VI algorithm does backup backwards, all the values used in the backup are correct, and the result follows. □

When the cost is integer, we can use $\mathcal{C} = \mathcal{D} = \{0, 1, 2, \ldots, C_{max}, \infty\}$ to guarantee that eGUBS returns the optimal policy.

Corollary 1 (**Integer cost** - Optimal policy). *Consider that the cost function has an image in the natural numbers, i.e., $c : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{I} \subset \mathcal{N}$. Let $\mathcal{C} = \mathcal{D} = \{0, 1, 2, \ldots, C_{max}, \infty\}$ and V_λ, P_G, and π be as in the Theorem 6, then the input*

$(\mathcal{M}, \lambda, K_g, V_\lambda, P_G, \pi, \mathcal{D}, \mathcal{C})$ is sufficient for eGUBS-VI algorithm returning the optimal policy.

Corollary 2 (Correct value function). *Consider cost function $c(\cdot)$ as in the Theorem 6 and let V_λ, P_G, π and $\mathcal{D}$ as in the Theorem 6, and let $\mathcal{C} \subset \mathcal{D}$, then, the input $(\mathcal{M}, \lambda, K_g, V_\lambda, P_G, \pi, \mathcal{D}, \mathcal{C})$ is sufficient for eGUBS-VI algorithm returning a value function V^* that is correct regarding policy π^*.*

In other words, if the accumulated-cost schedule respects Theorem 6, eGUBS-VI algorithm retuns an optimal policy, if it is not the case eGUBS-VI algorithm returns a sub-optimal one.

4.4 Sub-Optimal Strategies

In this section we describe four strategies to find sub-optimal policies to eGUBS.

Definition 6 (Combinatorial Search). *Consider cost function $c(\cdot)$ as in the Theorem 6 and let V_λ, P_G, π and $\mathcal{D}$ as in the Theorem 6. Consider an initial state distribution $\theta : \mathcal{S} \to [0, 1]$ and find $\mathcal{C}$ such that:*

- *$\mathcal{C} \subset \mathcal{D}$;*
- *$|\mathcal{C}| = M \leq N$; and*
- *maximizes the average value $\sum_{s \in \mathcal{S}} \theta(s) V^*(s, 0)$.*

Because combinatorial search has a high complexity, we experiment with a forward greedy search, in addition to solving the problem exactly using exhaustive search. **Exhaustive search** evaluates every subset of $\mathcal{D}$ with M elements. Thus, exhaustive search demands $\frac{N!}{(N-M)!\,M!}$ evaluations. **Forward greedy search** chooses one element on each iteration and does not remove it from $\mathcal{C}$. Thus, forward greed search demands $\sum_{i=N-M+1}^{N} i = \frac{2NM - M^2 + M}{2} \leq NM$ evaluations.

We also consider two heuristics that demands only one evaluation. **Uniform distribution** that chooses elements to be put in $\mathcal{C}$ in such way that: $0 \in \mathcal{C}$ and that difference between two consecutive accumulated cost are close to $\frac{N-1}{M}$, more precisely D_i is in $\mathcal{C}$ if and only if there exists scalar k such that $\left| D_i - k\frac{C_{\max}}{N} \right| \leq 0.5$. The uniform distribution demands only one evaluation. **Initial-dense distribution** considers simply $\mathcal{C} = \{D_0, D_1, \ldots, D_{M-1}, \infty\}$. Just like uniform distribution, initial-dense distribution demands only one evaluation, however, if $\mathcal{D}$ is chosen just like $\mathcal{C}$, the value function is still correct, but the evaluation is cheaper.

5 Experiments

We run experiments in the river problem (Freire and Delgado 2017) to evaluate the eGUBS-VI algorithm and the sub-optimal strategies.

5.1 River problem

In the river problem an agent is in one side of the river and wants to go to the other side. The agent has two options: (*i*) swimming from any point of the river bank, or (*ii*) going along the river bank until a bridge. This problem is modeled as a grid $N_x \times N_y$, where: (i) $x = 1$ and $x = N_x$ represent the river banks; (ii) $y = N_y$ represents the bridge; and (iii) $y = 1$ represents a waterfall where the agent can get trapped or die. The goal is to get the other side of the river bank far from the bridge, $x_g = N_x$ and $y_g = 1$.

The agent can move in the cardinal directions and actions succeed with different probability depending on the state. To avoid proper policies, if actions are taken on the river bank, we add a probability of 0.01 of falling in the river. If actions are taken in the bridge, transitions are deterministic. If actions are taken in the river then transitions are probabilistic and follows the chosen cardinal directions with probability $(1 - Prob_{river})^2$, follows down the river with probability $(Prob_{river})^2$ or stays in the same place with probability $2Prob_{river}(1 - Prob_{river})$. The waterfall is modeled as dead-end states. The immediate cost is 1.

5.2 eGUBS-VI algorithm and C_{max}

One of the main drawback of eGUBS-VI algorithm is its memory cost, which is proportional to C_{max}. We conduct a set of experiments to see the influence of problem and eGUBS criterion parameters. In all of them we consider a base configuration with: $K_g = 1$, $\lambda = -0.1$, $N_y = 50$, $N_x = 5$, and $Prob_{river} = 0.8$. Then, we vary systematically K_g, λ, N_y and $Prob_{river}$.

Figure 1 shows the influence of the problem in C_{max}. Remember that C_{max} is the sufficient accumulated-cost to the RS-Dual criterion being optimal under the eGUBS criterion. River probability $Prob_{river}$ controls the difficulty of crossing the river, the bigger $Prob_{river}$, the smaller the probability to goal when crossing the river; therefore when $Prob_{river}$ is large, crossing the river only pays off if accumulated cost is small.

Regarding size of the state space, N_y controls the cost of the safest path, the largest probability to goal; therefore crossing the river pays off if N_y is too large.

Figure 2 shows the influence of the GUBS criterion parameters in C_{max}. Theorem 4 shows that C_{max} decreases with K_g and increases with the absolute value of λ. Figure 2 confirms such a theoretic result. In fact, if $\lambda \to 0$, then the accumulated cost decreases utility linearly, and RS-Dual policy is not optimal for any accumulated cost; on the other side, if $|\lambda|$ is large, utility decays fast to its inferior bound, and future cost are irrelevant, when the RS-Dual criterion is optimal. Regarding K_g, if $K_g \to 0$, probability to goal is not considered and the RS-Dual criterion is hardly optimal, on the other side, if K_g is large, the RS-Dual criterion becomes optimal with less accumulated cost. Note that as defined in Theorem 4, C_{max} decay logarithmic with K_g.

5.3 eGUBS-VI and sub-optimal strategies

As we show in previous section, C_{max} may get too large depending on the problem or eGUBS criterion parameters. In

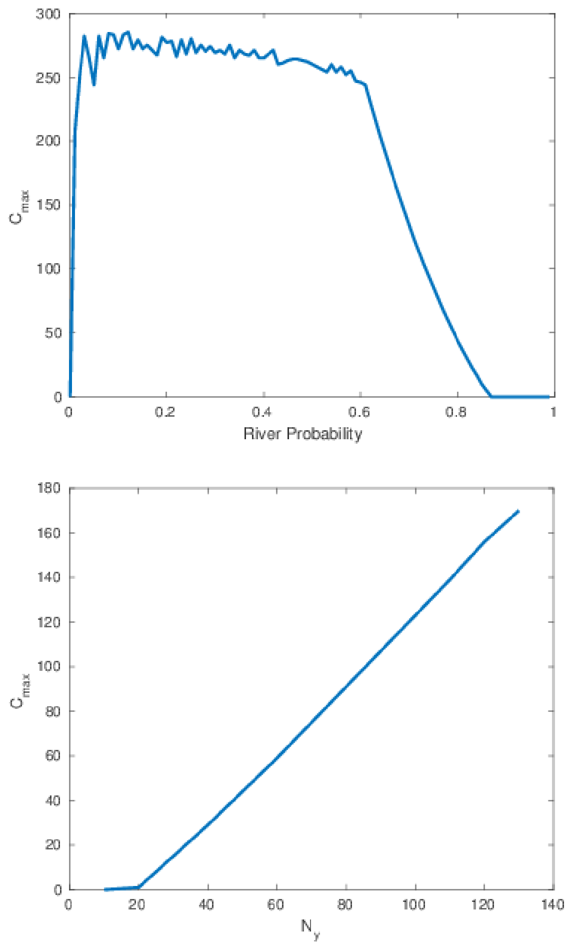

Figure 1: Influence of problem configuration in C_{max}. (Top) Influence of River Probability in C_{max}. (Bottom) Influence of N_y in C_{max}.

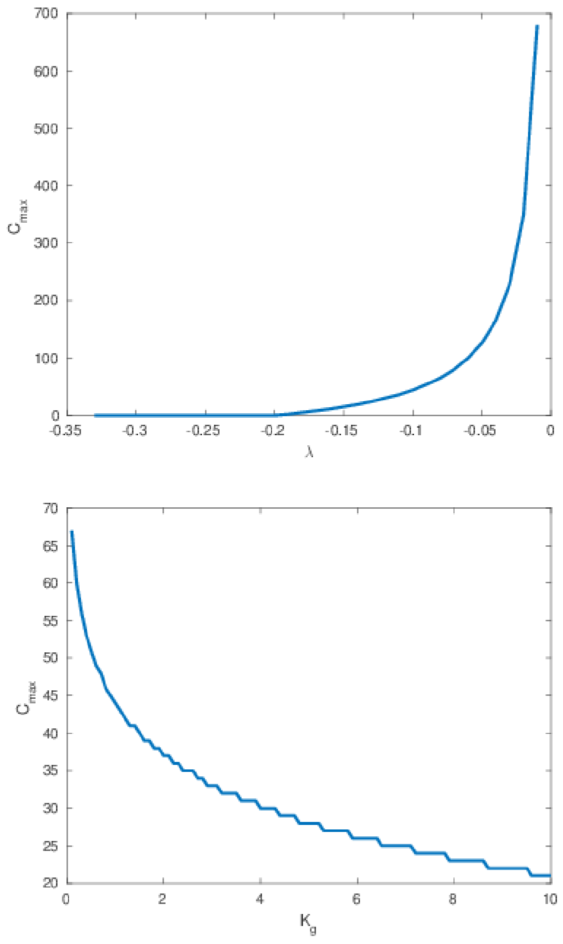

Figure 2: Influence of problem eGUBS configuration in C_{max}. (Top) Influence of risk factor λ in C_{max}. (Bottom) Influence of K_g in C_{max}.

such scenarios, considering sub-optimal strategies is mandatory. We evaluated the four strategies described in Section 4.4 in two scenarios; we fixed $K_g = 1$, $\lambda = -0.1$, $N_x = 5$, $Prob_{river} = 0.8$, and test for $N_y = 50$ (scenario 1), where $C_{max} = 44$, and $N_y = 100$ (scenario 2), where $C_{max} = 123$.

The value schedule $\mathcal{D}$ is considered always full; since cost is unitary, $\mathcal{D} = \{0, 1, \ldots, C_{max} - 1\}$. The policy schedule $\mathcal{C}$ is constructed following each strategy constrained to a given number of schedule points; we study how performance varies according to the number of schedule points.

Figure 3 and 4 show results for scenarios 1 and 2, respectively. In scenario 1, the number of schedule points were varied from 0 (RS-Dual criterion) to 20, and in scenario 2, the number of schedule points where varied from 0 to 40. Because Exhaustive search presents a high time cost, exhaustive search was tested only for schedule points between 0 and 4 in scenario 1, and it was not tested in scenario 2.

As expected, all four strategies converges to the same level as the number of schedule points increases (top figures). While Exhaustive search is proved to found the best schedule constrained to a given number of schedule points, greedy search presents a performance very close to exhaustive search. Clearly, greedy search strategy is better in performance than uniform and initial-dense strategies. Uniform strategy is slightly better than Initial-Dense one in scenario 1, but it is clearly worse than Initial-Dense strategy in the scenario 2. In both scenarios with only 5 (five) schedule points, Greedy search strategy is very close to optimality.

Regarding the cost-to-goal and probability-to-goal, all the sub-optimal strategies present different patterns of trade-off (bottom figures). The eGUBS criterion makes a trade-off between probability-to-goal and cost-to-goal based on parameters K_g and λ, what is different of simply maximizing cost-to-goal for a fixed probability-to-goal. On the other side, adequate trade-offs can be obtained by simply choosing $K_g = 1$ and $|\lambda|$ small.

First, note the difference between RS-Dual and eGUBS trade-off; in scenario 1, the RS-Dual criterion presents probability to goal 0.921 and average cost to goal 47.8 (right-top

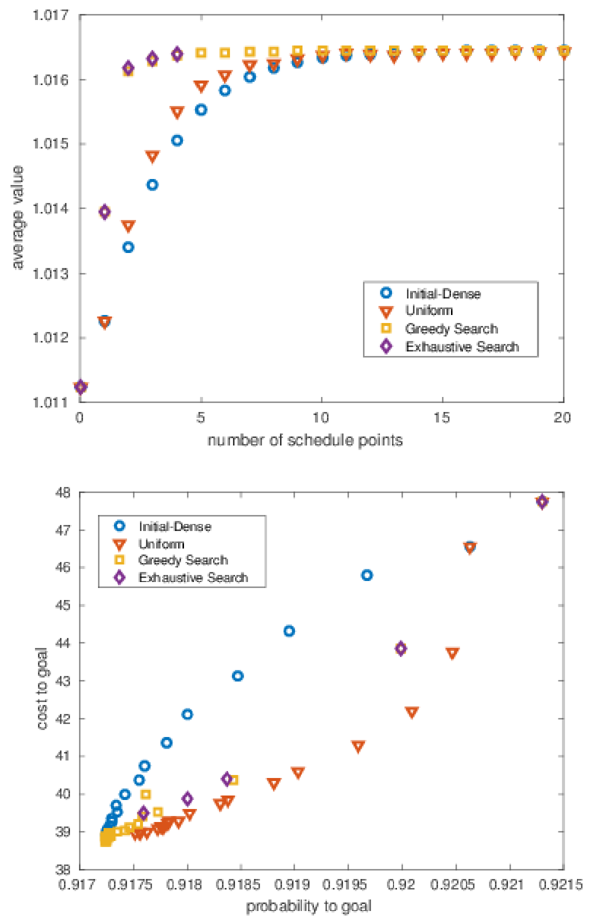

Figure 3: Comparison between sub-optimal strategies in a problem with $N_y = 50$. (Top) Influence of number of schedules points. (Bottom) Trade-off between probability-to-goal and cost-to-goal.

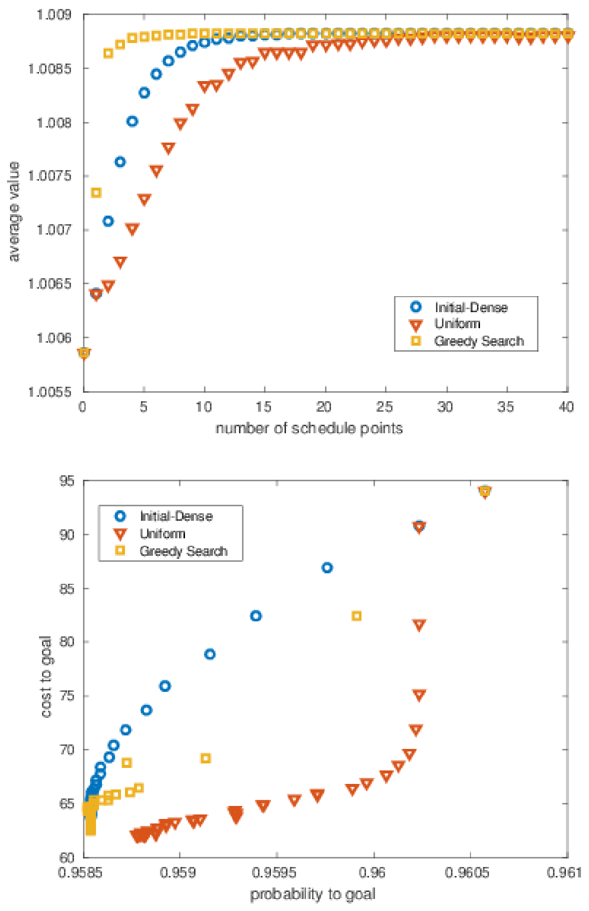

Figure 4: Comparison between sub-optimal strategies in a problem with $N_y = 100$. (Top) Influence of number of schedules points. (Bottom) Trade-off between probability-to-goal and cost-to-goal.

points), while eGUBS presents probability to goal 0.917 and average cost to goal 38.7 (left-bottom points), and in scenario 2, the RS-Dual criterion presents probability to goal 0.961 and average cost to goal 94.0, while eGUBS presents probability to goal 0.959 and average cost to goal 61.6. In both scenario, a small amount of probability is payed for a significant decrease in average cost-to-goal.

Regarding sub-optimal strategies, initial-dense and uniform strategies present a complete different behaviour, the first one presenting a better trade-off between cost-to-goal and probability-to-goal, i.e., a smaller decrease in probability produces a greater decrease in cost-to-goal. Greedy search presents a trade-off between initial-dense and uniform strategies; greedy search strategy also approaches the eGUBS solution faster (fewer points far from optimality).

The greedy search strategy presented a good trade-off between value and number of schedule points, but, for small M, we have a complexity of MN evaluations of eGUBS-VI algorithm. Since Initial-dense and uniform strategies are heuristics, they demand only 1 (one) evaluation.

Figure 5 shows which accumulated-cost point is chosen in each iteration of greedy search. We can see that the first schedule points to be chosen are limited to small accumulated cost; therefore, search may be heuristically limited to the initial region.

6 Conclusion

The GUBS criterion makes a trade-off between probability-to-goal and cost-to-goal based on the expected utility theory. We contribute with the first exact algorithm to solve SSPs that use the GUBS criterion when exponential utility function is considered with a negative risk factor, the eGUBS-VI algorithm. Since optimal policy is non-markovian, the state space must be extended, however space augmentation are memory and time demanding. We make experiments with four sub-optimal strategies, and we conclude that good policies can be found with an order of magnitude increase in the memory space of stationary policies.

The exact algorithm developed here can be adapted in many directions. We do not consider an initial state and

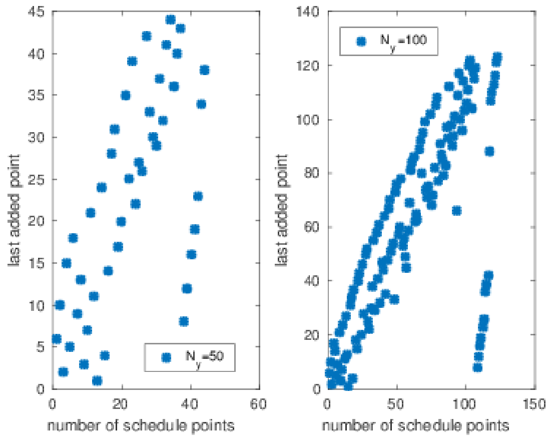

Figure 5: Order of entrance of each accumulated cost in the schedule points.

C_{max} based on an initial state may be smaller than C_{max} for every state. However, since we make use of dynamic programming, the states reached from an initial state and consequently C_{max} will depend on the policy that is being constructed. Additionally, the eGUBS criterion may be used in problems where a reward must be maximized; in this case, because null reward is allowed, an adapted version of eGUBS-VI algorithm might be time consuming.

We also pointed directions to improve on greedy search strategies with small losses in performance. Two possible directions are: (*i*) making use of heuristics to accelerate search, while keeping small number of schedule points; and (*ii*) making use of initial-dense strategy to include new schedule points while enough improvements are obtained, in this case, the number of schedule points may be larger than necessary.

Acknowledgment

The authors would like to thank the São Paulo Research Foundation (FAPESP) for the financial support (grant #2015/01587-0).

References

Bertsekas, D. P., and Tsitsiklis, J. N. 1991. An analysis of stochastic shortest path problems. *Mathematics of Operations Research* 16(3):580–595.

Bonet, B., and Geffner, H. 2003. Labeled RTDP: Improving the convergence of real-time dynamic programming. *Proceedings of International Conference on Automated Planning and Scheduling* 12–21.

Bonet, B., and Geffner, H. 2005. mGPT: A probabilistic planner based on heuristic search. In *Journal of Artificial Intelligence Research*, volume 24, 933–944.

Camacho, A.; Muise, C.; and McIlraith, S. A. 2016. From FOND to robust probabilistic planning: Computing compact policies that bypass avoidable deadends. In *The 26th International Conference on Automated Planning and Scheduling*, 65–69.

Freire, V., and Delgado, K. V. 2017. Gubs: a utility-based semantic for goal-directed markov decision processes. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, 741–749.

Kolobov, A.; Mausam; Weld, D.; and Geffner, H. 2011. Heuristic search for generalized stochastic shortest path MDPs. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling (ICAPS)*, 130–137.

Kolobov, A.; Mausam; and Weld, D. S. 2012. A theory of goal-oriented MDPs with dead ends. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 438–447.

Minami, R., and Silva, V. F. d. 2012. Shortest stochastic path with risk sensitive evaluation. In *11th Mexican International Conference on Artificial Intelligence (MICAI 2012)*, volume 7629 of *Lecture Notes in Artificial Intelligence*, 370–381.

Patek, S. D. 2001. On terminating Markov decision processes with a risk averse objective function. *Automatica* 37:1379–1386.

Teichteil-Königsbuch, F.; Kuter, U.; and Infantes, G. 2010. Incremental plan aggregation for generating policies in MDPs. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 1231–1238.

Teichteil-Königsbuch, F. 2012. Stochastic safest and shortest path problems. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI 12)*, 1825–1831.

Trevizan, F.; Teichteil-Königsbuch, F.; and Thiébaux, S. 2017. Efficient solutions for stochastic shortest path problems with dead ends. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*.