Efficient Solutions for Stochastic Shortest Path Problems with Dead Ends

Felipe Trevizan
Data61, CSIRO
Australian National University
felipe.trevizan@anu.edu.au

Florent Teichteil-Königsbuch
Airbus
Central Research & Technology
florent.teichteil-koenigsbuch@airbus.com

Sylvie Thiébaux
Data61, CSIRO
Australian National University
sylvie.thiebaux@anu.edu.au

Abstract

Many planning problems require maximizing the probability of goal satisfaction as well as minimizing the expected cost to reach the goal. To model and solve such problems, there have been several attempts at extending Stochastic Shortest Path problems (SSPs) to deal with dead ends and optimize a dual optimization criterion. Unfortunately these extensions lack either theoretical robustness or practical efficiency. We study a new, perhaps more natural optimization criterion capturing these problems, the Min-Cost given Max-Prob (MCMP) criterion. This criterion leads to the minimum expected cost policy among those with maximum success probability, and accurately accounts for the cost and risk of reaching dead ends. Moreover, it lends itself to efficient solution methods that build on recent heuristic search algorithms for the dual representation of stochastic shortest paths problems. Our experiments show up to one order of magnitude speedup over the state of the art.

1 INTRODUCTION

Stochastic Shortest Path Problems (SSPs) [Bertsekas and Tsitsiklis, 1991] are the *de facto* model for planning in stochastic environments in which an agent need to reach a goal state while minimizing the cost of doing so. Most theoretical results regarding SSPs assume that there are no dead ends in the environment, yet this is hardly the case in practice. Unfortunately, if dead ends are unavoidable, i.e., the probability of reaching the goal is always less than 1, SSPs implicitly become a multi-objective optimization problem with two potentially conflicting objectives: the maximization of the probability to reach the goal and the minimization of the expected cost to reach the goal.

To illustrate the conflict between these two objectives, consider a factory that can use either a human employee or a robot to manufacture a fragile good. Let the cost of the actions represent the cost in dollars of the actions for both the human (e.g., fraction of their salary) and the robot (e.g., cost of energy to perform the action) and the dead ends represent states in which the good cannot be manufactured anymore and the source materials are wasted. For this example, the robot is not capable of perfectly grasping the fragile source materials but if it succeeds ($p_r = 0.9$), the good can be perfectly manufactured due to the robots precision at an expected cost of \$100 for the electric energy used. Alternatively, human employees perfectly handle the fragile source materials; however, they lack the precision of the robot and can only complete the manufacture of the good with probability $p_h = 0.95$ with an expected labour cost of \$500. The conflict appears when these two options are compared: is it worth paying the extra \$400 to increase the probability of producing the good from 0.9 to 0.95?

One approach, known as Finite-Penalty [Mausam and Kolobov, 2012], is to assign a finite and fixed penalty D for not reaching the goal. For instance, the expected cost of assigning a human employee and a robot in the factory example is $475 + 0.05D$ and $90 + 0.1D$, respectively. Although intuitive, Finite-Penalty has the drawback of labelling any state with expected cost greater than D as a dead end. In our example, if the source materials cost \$10 and we use this value as dead-end penalty D, then using either the human or the robot are considered solutions that never reach the goal since their expected costs is greater than $D = 10$. Although it might be easy to derive a good value for D in our factory example, this is not the case for domain-independent planning (i.e., for algorithms that do not use knowledge of the problem being solved) which is the scope of this paper. For instance, there is no domain-independent method to derive D automatically given a problem description. There is also no known method to learn a function that returns, for all planning problems, a value of D that is always large enough but never so large that it leads to numerical instability or increased CPU time.

In order to avoid the need for the fixed penalty D (and its drawbacks), one can ignore the expected cost and maximize only the probability to reach the goal. This approach, known as Max-Prob [Kolobov et al., 2011], would always choose the human employee in the factory example; however it would be unable to distinguish between different employees with the same probability of success but with different pay rates. To be able to distinguish such cases in addition to getting rid of finite penalties for dead ends, two similar approaches have been proposed: S^{3}P [Teichteil-Königsbuch, 2012] and iSSPUDE [Kolobov et al., 2012]. Both approaches minimize the expected cost to reach the goal in a transformed SSP in which dead ends are removed and whose feasible solutions are exactly the same as the solutions that maximizes the probability of reaching the goal in the original problem. Unfortunately, computing this dead-end-free SSP potentially requires enumerating all the possible Max-Prob solutions which, as we later show, can be extremely inefficient.

In this paper, we introduce a new optimization criterion for SSPs with dead ends, the Min-Cost given Max-Prob (MCMP) criterion, which leads to the minimum expected cost policy among those with maximum success probability. MCMP differs from S^{3}P and iSSPUDE on how paths leading to dead ends are accounted for. This difference results in no transformation from the original SSP to a dead-end-free SSP being required, whether a priori or on-the-fly. As we show in our experiments, the optimal MCMP solution for SSPs can be computed up to 1 order of magnitude faster than its respective S^{3}P and iSSPUDE solution.

This performance improvement is due to how search is performed in the space of Max-Prob solutions: S^{3}P and iSSPUDE potentially compute the probability of reaching the goal for all the states reachable under any Max-Prob policy from the initial state. In contrast, MCMP need only find the maximum probability of reaching the goal from the initial state, and then search in the original SSP search space for a solution that minimizes the expected cost and reaches the goal with that probability. This allows MCMP to explore the original SSP search space using heuristics for both the probability of reaching the goal and the expected cost of reaching the goal. Moreover, the MCMP criterion is naturally expressed using dual formulation of SSPs and can trivially accommodate cost constraints; therefore, it also trivially applies to Constrained SSPs [Altman, 1999], which is not possible using S^{3}P or iSSPUDE.

2 NOTATION AND BACKGROUND

A Stochastic Shortest Path problem (SSP) [Bertsekas and Tsitsiklis, 1991] is a tuple $\mathbb{S} = \langle \mathsf{S}, s_0, \mathsf{G}, \mathsf{A}, P, C\rangle$ in which: S is the finite set of states; $s_0 \in \mathsf{S}$ is the initial state; $\mathsf{G} \subseteq \mathsf{S}$ is the non-empty set of goal states; A is the finite set of actions; $P(s'|s,a)$ is the probability that s' is reached after applying action a in state s; and $C(s,a) \in \mathbb{R}^*_+$ is the cost of applying action a in state s. We represent by $\mathsf{A}(s)$ the actions applicable in state s.

A solution to an SSP is a policy π mapping from states to actions. We denote by Π the set of all (deterministic) policies for $\mathbb{S}$. A trace $T_{\pi,s}$ is a sequence of states $s\, s_1 s_2 \ldots$ visited when following $\pi \in \Pi$ from s. We denote the i-th state of $T_{\pi,s}$ by $T^i_{\pi,s}$ and require that $P(T^{i+1}_{\pi,s}|T^i_{\pi,s}, \pi(T^i_{\pi,s})) > 0$ for all pairs $T^i_{\pi,s}T^{i+1}_{\pi,s} \in T_{\pi,s}$. If s is omitted, then s_0 is implied. Also, π is omitted when clear from context.

A trace T can be finite or infinite. If finite, the last state s of T is either a goal state (i.e., $s \in \mathsf{G}$) or there is no applicable action in s (i.e., $\mathsf{A}(s) = \emptyset$). Moreover, for a finite trace T, its probability is $P(T) = \prod_{i=1}^{|T|} P(T^{i+1}|T^i, \pi(T^i))$ and its cost is $C(T) = \sum_{i=1}^{|T|} C(T^i, \pi(T^i))$. Infinite traces happen when the execution of π enters a cycle that never reaches the goal, getting trapped there forever. Since action costs are strictly positive, the cost of any infinite trace is infinity.

We denote by $\mathsf{T}_{\pi,s}$ the set of all traces of π from s. Notice that $\mathsf{T}_{\pi,s}$ may be an infinite set (e.g., if an action prescribed by π has a positive probability of looping). We also partition $\mathsf{T}_{\pi,s}$ into the set of traces that reach the goal ($\mathsf{T}^{\mathsf{G}}_{\pi,s}$) and its complement $\mathsf{T}^{\mathsf{DE}}_{\pi,s}$, i.e, the set of traces that do not reach the goal. While every trace in $\mathsf{T}^{\mathsf{G}}_{\pi,s}$ is finite (because its last state must be a goal state), $\mathsf{T}^{\mathsf{DE}}_{\pi,s}$ can have infinite traces, that is, traces that loop indefinitely without reaching a goal state. We use the same notation simplifications for $\mathsf{T}_{\pi,s}$ and its partitions as for traces.

Most results and algorithms for SSPs assume that there are no unavoidable dead ends, i.e., there exists a policy for which the goal can be reached from the initial state with probability 1. Using our trace notation, this is equivalent to assuming that there exists a policy π such that $\mathsf{T}^{\mathsf{DE}}_{\pi,s_0} = \emptyset$. When this assumption holds, the optimal solution of an SSP is characterized by a unique fixed-point solution for the following set of equations, known as Bellman equations:

$$V^*(s) = \min_{a \in A(s)} C(s,a) + \textstyle\sum_{s' \in \mathsf{S}} P(s'|s,a)V^*(s')$$

for all $s \in \mathsf{S} \setminus \mathsf{G}$ and $V^*(s_g) = 0$ for all $s_g \in \mathsf{G}$. Any greedy policy w.r.t. to the optimal value function V^* (i.e., by applying argmin instead of $\min$) is an optimal policy.

In the presence of dead ends, the Bellman equations might diverge resulting in the optimal solution to be ill-defined. This is always the case when the problem has unavoidable dead ends. Several criteria have been proposed to characterize the optimal solution of SSPs with dead ends, such as the Finite-Penalty criterion:

Definition 1 (Finite-Penalty)**.** *Given a dead-end penalty $D \in \mathbb{R}^*_+$, find a greedy policy for the unique fixed-point solution of:*

$$V^*_{FP}(s) = \min\{D, \min_{a \in \mathsf{A}(s)} C(s,a) + \sum_{s' \in \mathsf{S}} P(s'|s,a)V^*_{FP}(s')\}$$

for all $s \in \mathsf{S} \setminus \mathsf{G}$ and $V^*_{FP}(s_g) = 0$ for all $s_g \in \mathsf{G}$.

The Finite-Penalty criterion is equivalent to solving the SSP $\mathbb{S}' = \langle \mathsf{S}, s_0, \mathsf{G}, \mathsf{A}', P', C'\rangle$ where, for all $s \in \mathsf{S}$, $\mathsf{A}'(s) = \mathsf{A}(s) \cup \{\text{give-up}\}$, $C'(s, \text{give-up}) = D$, $C'(s,a) = C(s,a)$ for $a \in \mathsf{A}(s)$, and $P(s_g|s, \text{give-up}) = 1$ for any $s_g \in \mathsf{G}$. Notice that $\mathbb{S}'$ has no dead ends and can be solved using any algorithm for SSPs.

A drawback of the Finite-Penalty criterion is that every state with expected cost of reaching the goal greater than the dead-end penalty D is considered a dead end, regardless of its maximum probability of reaching the goal. E.g.,

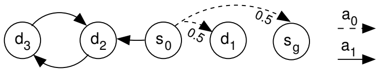


Figure 1: Example of SSP with multiple fixed-point solutions for P^*. The goal set is $\mathsf{G} = \{s_g\}$ and $P^*(s_0) = 0.5$.

consider a simple SSP with two states s_0 and s_g ($G = \{s_g\}$) and one action a s.t. $P(s_g|s_0, a) = p > 0$ and $P(s_0|s_0, a) = 1 - p$. If $C(s, a)/p$ is greater than D then s_0 is considered a dead end under the Finite-Penalty criterion and its optimal policy prescribes the "give-up" action for s_0 even though the probability of reaching s_g from s_0 is 1. To overcome this, an alternative is to directly maximize the probability of reaching the goal, a criterion known as Max-Prob:

Definition 2 (Max-Prob). *Find a policy π that maximizes the probability of reaching any goal state $s_g \in \mathsf{G}$ when following π from s_0, i.e., $\operatorname{argmax}_{\pi \in \Pi} P(\mathsf{T}^{\mathsf{G}}_{\pi,s_0})$. We denote by p^{max} this maximum probability and by Π^{MP} the set of all Max-Prob policies, i.e., $\{\pi \in \Pi | P(\mathsf{T}^{\mathsf{G}}_{\pi,s_0}) = p^{max}\}$.*

The optimal solution under the Max-Prob criterion can be computed by solving a problem $\mathcal{M}$ in which everything stays the same as in the original SSP $\mathbb{S}$ except that we want to *maximize* the expected sum, over all traces $T \in \mathsf{T}_{\pi,s_0}$, of the function $R(s, a) = \sum_{s_g \in \mathsf{G}} P(s_g|s, a)$. We denote the solution of the Bellman equations for $\mathcal{M}$ as P^* and $P^*(s)$ represents the maximum probability of reaching the goal from $s \in \mathsf{S}$ (thus $p^{\max}$ equals $P^*(s_0)$). Also, $P^*(s) \in [0, 1]$ for all s even if $\mathcal{M}$ has unavoidable dead ends; however, the Bellman equations for $\mathcal{M}$ can have several non-optimal fixed-point solutions. For instance, the SSP in Fig. 1 has the following *non-optimal* fixed-point solution for $\mathcal{M}$: $P(d_1) = 0$ and $P(s_g) = P(s_0) = P(d_2) = P(d_3) = 1$.

One approach to compute P^* (and avoid being trapped in a non-optimal solution) is to apply value iteration on $\mathcal{M}$ after initializing the algorithm with 1 for all $s_g \in \mathsf{G}$ and 0 otherwise. Another approach is to solve $\mathcal{M}$ using heuristic search algorithms such as FRET [Kolobov et al., 2011] and FRET-π [Steinmetz et al., 2016]. Both these algorithms iterate between: (i) finding a fixed-point solution for P^* using any optimal heuristic search algorithm for SSPs; and (ii) post processing the candidate solution to remove cycles that have not converged to P^*. In our example from Fig. 1, step (ii) would identify the loop between d_2 and d_3 and update $P(d_2)$ and $P(d_3)$ to 0.

In addition to the non-optimal fixed-point solutions, the Max-Prob criterion has another drawback: all policies in Π^{MP} are equally good for Max-Prob, independently of their expected cost. The S^{3}P criterion addresses this isssue:

Definition 3 (S^{3}P [Teichteil-Königsbuch, 2012]). *Find a policy*

$$\pi = \operatorname*{argmin}_{\pi \in \Pi^{MP}} \mathbf{E}[C(T) \,|\, T \in \mathsf{T}^{\mathsf{G}}_{\pi,s_0}].$$

The S^{3}P criterion is inspired by the evaluation metric used at the International Probabilistic Planning Competition (IPPC) up to 2008 [Younes et al., 2005, Bryce and Buffet, 2008] that accounted both for the proportion of simulations of a policy that reached the goal and for the average cost accumulated by these policies only when they reached the goal. Thus, S^{3}P only considers policies in Π^{MP} and the traces of these policies that reach the goal. Moreover, given our assumption of $C(s, a) > 0$ for all $s \in \mathsf{S}$ and $a \in \mathsf{A}(s)$, the S^{3}P criterion is equivalent to the iSSPUDE criterion [Kolobov et al., 2012]. Given P^*, the S^{3}P optimal solution of an SSP is equivalent to the optimal solution of the following transformed SSP [Kolobov et al., 2012]:

Definition 4 (P^*-SSP). *Given an SSP $\langle \mathsf{S}, s_0, \mathsf{G}, \mathsf{A}, P, C\rangle$ and its P^*, the P^*-SSP is the tuple $\langle \mathsf{S}, s_0, \mathsf{G}, \mathsf{A}', P', C\rangle$ where: $\mathsf{A}'(s) = \mathsf{A}(s)$ if $P^*(s) > 0$ and empty otherwise; and $P'(s'|s, a) = P(s'|s, a)P^*(s')/P^*(s)$ if $P^*(s) > 0$ and 0 otherwise.*

Since P^*-SSP has no dead ends, any SSP algorithm can be used for solving it. In practice, P^* can be computed on-the-fly while solving the P^*-SSP, as we do in our experiments.

3 MCMP CRITERION

As shown in the previous section, the S^{3}P criterion extends the Max-Prob criterion by ranking Max-Prob policies according to their expected cost when the traces reaching dead ends are ignored. One drawback of this approach is that, when executing an S^{3}P optimal policy $\pi^*_{\text{S}^3\text{P}}$, unavoidable dead ends will still be reached and the cost of doing so will also be incurred; therefore the optimal value computed by the S^{3}P algorithms might be different from the expected cost obtained by an execution $\pi^*_{\text{S}^3\text{P}}$. In this section we address this issue, but first let us define the function ψ that truncates a trace at the first encountered dead end state:

Definition 5 (ψ). *Let $\psi\colon \mathsf{T} \mapsto \mathsf{T}$ be the function:*

$$\psi(s_i s_{i+1} \cdots) = \begin{cases} s_i & \textit{if } |T| = 1 \textit{ or } P(\mathsf{T}^{\mathsf{G}}_{s_i}) = 0 \\ s_i\, \psi(s_{i+1} \cdots) & \textit{otherwise} \end{cases}$$

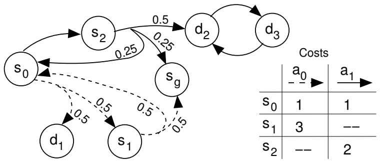
Notice that the $\psi(T)$ is always a finite trace since either a goal is reached or a state s_i with probability 0 of reaching the goal ($P(\mathsf{T}^{\mathsf{G}}_{s_i}) = 0$) is reached and $s_{i+1}s_{i+2}\cdots$ are ignored. Moreover, $\psi(T) = T$ for all $T \in \mathsf{T}^{\mathsf{G}}$.

Our optimal criterion for SSPs with dead ends, the Min-Cost given Max-Prob (MCMP) is defined in Definition 6 and, although it does not distinguish the different traces reaching the same dead end, it differs from S^{3}P because MCMP considers the complete set of traces T_{π,s_0}. Example 1 illustrates the differences between S^{3}P and MCMP and show that their optimal policies can be different.

Definition 6 (MCMP). *The Min-Cost Max-Prob (MCMP) problem is to find a policy*

$$\pi = \operatorname*{argmin}_{\pi \in \Pi^{MP}} \mathbf{E}[C(\psi(T))].$$

Example 1. *Consider the SSP in Fig. 2. There are two possible policies in this SSP: π_0 that always applies a_0 and π_1 that always applies a_1. The probability of reaching s_g for*



	a_0	a_1
s_0	1	1
s_1	3	--
s_2	--	2

Figure 2: Example of SSP. The goal is $\mathsf{G} = \{s_g\}$. The initial state is s_0 and states d_1, d_2 and d_3 are dead ends. See Example 1 for more details.

each one of them is: $P(\mathsf{T}^{\mathsf{G}}_{s_0,\pi_0}) = .5(.5{+}.5P(\mathsf{T}^{\mathsf{G}}_{s_0,\pi_0})) = 1/3$ *and* $P(\mathsf{T}^{\mathsf{G}}_{s_0,\pi_1}) = .25 + .25P(\mathsf{T}^{\mathsf{G}}_{s_0,\pi_1}) = 1/3$, *thus* $\Pi^{MP} = \{\pi_0,\pi_1\}$ *and* $p^{max} = 1/3$. *For the* S^{3}P *criterion, we have:*

π_0: *Only traces of the form* $(s_0 s_1)^+ s_g$ *are considered, where* $(s_0 s_1)^+$ *represent 1 or more repetitions of the sequence* $s_0 s_1$. *The expected cost of* π_0 *is* $V^{\pi_0}_{S^3P}(s_0) = (\sum_{i=1}^{\infty}(0.5 \times 0.5)^i (1+3)i)/p^{max} = 16/3$.

π_1: *Only* $(s_0 s_2)^+ s_g$ *traces are considered. The expected cost of* π_1 *is* $V^{\pi_1}_{S^3P}(s_0) = (\sum_{i=1}^{\infty}(1{\times}0.25)^i(1{+}2)i)/p^{max} = 4$.

For MCMP, we have:

π_0: *All traces in* T_{π_0,s_0} *are finite since* $A(d_1) = \emptyset$. *Thus* $\psi(T) = T$ *for all* $T \in \mathsf{T}_{\pi_0,s_0}$. *Since* s_g *is reached with probability* $p^{max} = 1/3$, *then* s_1 *is visited in expectation* $p^{max}/0.5 = 2/3$ *and* s_0 *visited* $(2/3)/0.5 = 4/3$. *Thus,* $V^{\pi_0}_{MCMP}(s_0) = 4/3 + 3 \times 2/3 = 10/3$.

π_1: *Since* d_2 *is the first state in the infinite loop* $d_2 d_3 d_2 \cdots$, $\psi(s_0 \cdots s_2 d_2 d_3 d_2 \cdots) = s_0 \cdots s_2 d_2$. *Thus, we only need to consider the expected number of times the costs* $C(s_0, a_1) = 1$ *and* $C(s_2, a_1) = 2$ *are incurred. Since* s_g *is reached with probability* $p^{max} = 1/3$, *then* s_0 *and* s_1 *are visited in expectation* $p^{max}/0.25 = 4/3$. *Therefore* $V^{\pi_1}_{MCMP}(s_0) = 4/3(1+2) = 4$.

Therefore, $V^{\pi_0}_{MCMP}(s_0) < V^{\pi_1}_{MCMP}(s_0) = V^{\pi_1}_{S^3P}(s_0) < V^{\pi_0}_{S^3P}(s_0)$, *and the optimal policy according to the MCMP and* S^{3}P *criteria is* π_0 *and* π_1, *respectively.*

As illustrated above, ignoring the traces that reaches dead ends can lead to unexpected optimal policies. In the case of our factory scenario from the introduction, the S^{3}P criterion would consider that we do not have to pay for electricity or the employee wages if a failure happened when producing the good. In contrast, the MCMP criterion accounts for all the cost until a dead end is reached, therefore, the electricity and employee wages are considered.

4 ALGORITHM FOR MCMP

Although it might seem costly to detect the infinite traces and prune them, this can be efficiently done by solving LP 1 using the heuristic search algorithm i-dual [Trevizan et al., 2016] (more below). This LP is a modified version of the dual LP for SSPs with no dead ends [Altman, 1999]. Its variables are the policy's occupation measures $x_{s,a}$ representing the expected number of times action $a \in A(s)$ will be executed in state s.

$$\min_x \sum_{s \in \mathsf{S}, a \in \mathsf{A}} x_{s,a} C(s,a) \quad \text{s.t.} \quad \text{(C1) -- (C6)} \qquad \text{(LP 1)}$$

$$x_{s,a} \geq 0 \qquad \forall s \in \mathsf{S}, a \in \mathsf{A}(s) \quad \text{(C1)}$$

$$in(s) = \sum_{s' \in \mathsf{S}, a \in \mathsf{A}(s')} x_{s',a} P(s|s',a) \qquad \forall s \in \mathsf{S} \quad \text{(C2)}$$

$$out(s) = \sum_{a \in \mathsf{A}(s)} x_{s,a} \qquad \forall s \in \mathsf{S} \setminus \mathsf{G} \quad \text{(C3)}$$

$$out(s) - in(s) \leq 0 \qquad s \in \mathsf{S} \setminus (\mathsf{G} \cup \{s_0\}) \quad \text{(C4)}$$

$$out(s_0) - in(s_0) \leq 1 \qquad \text{(C5)}$$

$$\sum_{s_g \in \mathsf{G}} in(s_g) = p^{\text{max}} \qquad \text{(C6)}$$

The modifications in LP 1 from the regular dual LP are on constraint C4 – C6. These modifications implicitly allow states to choose a give-up action as in the Finite-Penalty criterion; however, this implicit give-up action *has cost zero*. The implicit give-up actions are encoded in C4 and C5 by upper bounding the conservation of flow instead of using an equality constraint. Formally, C4 is equivalent to $out(s) - in(s) + x_{s,\text{give-up}} = 0$ for $x_{s,\text{give-up}} \geq 0$ and $C(s, \text{give-up}) = 0$, thus not affecting the objective value. Moreover, C6 constrains the solution found by LP 1 to be in Π^{MP} because it enforces that, from the one unit of flow injected in the system (C5), p^{max} units of flow reach the goal (p^{max} is a constant for LP 1). Therefore, the implicit give-up actions are only used when strictly necessary, as otherwise not enough flow would reach the goal. They are used in a state s that either: has no applicable action, in which case $out(s)$ can only be zero while $in(s)$ might be greater than 0; or is an infinite loop that never reaches the goal, in which case the give-up action avoids an infinite cost.

Computing the MCMP solution using LP 1 has the advantage that it only requires p^{max}, which can be computed by finding any Max-Prob policy $\pi \in \Pi^{\text{MP}}$. In contrast, the S^{3}P criterion requires the P*-SSP, thus all Max-Prob policies will be considered in the worst case. Another advantage is that we can solve LP 2 to compute p^{max} and, since the optimal solution x^* for LP 2 (below) is a feasible solution for LP 1, we can use x^* to *warm start* the search in LP 1.

$$\max_x \sum_{s_g \in \mathsf{G}} in(s_g) \quad \text{s.t.} \quad \text{(C1) -- (C3), (C7) -- (C8)} \qquad \text{(LP 2)}$$

$$out(s) - in(s) = 0 \qquad \forall s \in \mathsf{S} \setminus (\mathsf{G} \cup \{s_0\}) \quad \text{(C7)}$$

$$out(s_0) - in(s_0) = 1 \qquad \text{(C8)}$$

LP 2 uses the original dual SSP flow constraints (C7 and C8) and it represents the dual formulation for the Max-Prob problem: we inject one unit of flow in s_0 and maximize the flow reaching the sink, i.e., the goal. Its optimal value is p^{max} and its optimal solution x^* can be converted into an optimal Max-Prob policy $\pi^*_{\text{MP}}(s) = a$ where $a \in A(s)$ is the only action such that $x_{s,a} > 0$ [Altman, 1999]. Differently from the space explored by Value Iteration, FRET, and FRET-π, the dual space explored by LP 2 does not contain non-optimal fixed-point solutions. For instance, in the SSP shown in Fig. 1, the dual formulation prevents any flow from reaching d_2 because, by C7, we have: $out(d_2) - in(d_2) = x_{d_2,a_1} - (x_{s_0,a_1} + x_{d_3,a_1}) = x_{s_0,a_1} = 0$.

Lastly, i-dual [Trevizan et al., 2016] can be used to efficiently solve LP 1 and LP 2 using heuristic search. Given an SSP, i-dual generates and solves increasingly large LPs (in the dual representation) to solve the current SSP. A parameter of i-dual is a cost heuristic for SSPs and this heuristic is used for choosing what states should be expanded to generate the next LP. I-dual can be trivially adapted for the MCMP criterion as follows. For solving LP 1 (MCMP second phase), LP 1 is used as template in line 12 of the original i-dual pseudo-code. Notice that, the heuristic for this stage is any cost heuristic for SSPs, e.g., $h^{\max}$ [Teichteil-Königsbuch et al., 2011], h^{roc} and h^{pom} [Trevizan et al., 2017]. Similarly, the first stage of MCMP (Max-Prob) can be solved by using LP 2 as template for the generation of LPs in line 12 of i-dual. It is important to notice that, for the first stage of MCMP, the cost of actions is ignored (LP 2 does not uses the cost function of the SSP) and the heuristic passed as a parameter to i-dual is a (admissible) Max-Prob heuristic, i.e., a (upper) bound on P^*. For both stages, the dead-end penalties required by i-dual are ignored since both LP 1 and LP 2 do not use them. The parameter $p^{\max}$ of LP 2 is obtained as the objective function of the last LP generated by i-dual when solving the first stage of MCMP.

5 RELATIONS BETWEEN CRITERIA

A property of the MCMP criterion, independently of the approach used to solve it, is that a dead-end penalty can be easily incorporated to the MCMP solution without the need to re-solve the problem. Formally, given $p^{\max}$ and $V^*_{\text{MCMP}}(s_0)$, the optimal solution cost for s_0 with dead-end penalty D is $V^*_{\text{MCMP}}(s_0) + (1 - p^{\max})D$. This is equivalent to use $C(\psi(T)) + D$ as the cost for all traces $T \in \mathsf{T}^{\text{DE}}$ since $\mathbf{E}[C(\psi(T)) + D | T \in \mathsf{T}^{\text{DE}}] = \mathbf{E}[C(\psi(T)) | T \in \mathsf{T}^{\text{DE}}] + D$.

The advantage of this approach over Finite-Penalty is that states s s.t. $P(\mathsf{T}^{\mathsf{G}}_s) > 0$ can have expected value greater than D, that is, the planner cannot avoid a large expected cost solution from s by using the give-up action. For instance, consider Example 1 and let the dead-end penalty be $D = 0.5$. For Finite-Penalty, we have $V^{\pi_0}_{\text{FP}}(s_0) = V^{\pi_1}_{\text{FP}}(s_0) = 0.5$ since $C(s_0, a_0) = C(s_0, a_1) = 1 > D$; moreover, $\pi^*_{\text{FP}}(s_0) = \text{give-up}$. Alternatively, for MCMP, the optimal policy is π_0 and its expected cost with the dead-end penalty is $10/3 + 0.5 \times 2/3 = 11/3$.

This intuition that the MCMP criterion captures the expected cost from s_0 more accurately than Finite-Penalty is formalized in the following theorem:

Theorem 1. *Let V^*_{FP} and V^*_{MCMP} be the optimal solution of Finite-Penalty and MCMP, respectively, for the same SSP and $D \in \mathbb{R}^*_+$ be the dead-end penalty used by Finite-Penalty. Then*

$$V^*_{FP}(s_0) \le V^*_{MCMP}(s_0) + (1 - p^{max})D$$

Proof Sketch. Notice that $V^*_{\text{FP}}(d) = D$ for all $d \in \mathsf{S}$ such that $P(\mathsf{T}^{\mathsf{G}}_d) = 0$ since d either (i) has no applicable action and thus it is assigned a cost of D; or (ii) is in a loop that never reaches the goal, thus it has infinite expected cost and the $\min$ operator of Finite-Penalty limits its expected cost to D. Therefore, $(1 - p^{\max})D$ is equivalent to the expected cost incurred in the Finite-Penalty for all such states d. Moreover, by the definition of Finite-Penalty, $V^*_{\text{FP}}(s) \le D$ for all $s \in \mathsf{S}$ even if $P(\mathsf{T}^{\mathsf{G}}_s) = 1$. Thus Finite-Penalty considers states with expected cost greater than D as dead ends. However, this is not the case for MCMP and $V^*_{\text{MCMP}}(s)$ can be arbitrarily large since it accounts for the real costs of transitions independently of D. Thus, $V^*_{\text{FP}}(s_0) - (1 - p^{\max})D \le V^*_{\text{MCMP}}(s_0)$. □

Similarly to MCMP, we can add a dead-end penalty in S^{3}P criterion, i.e., define $C(T) = D$ for $T \in \mathsf{T}^{\text{DE}}$. Formally,

$$\begin{aligned}\mathbf{E}[C(T)]& \\ &= p^{\max}\,\mathbf{E}[C(T) | T \in \mathsf{T}^{\mathsf{G}}] + (1 - p^{\max})\,\mathbf{E}[C(T) | T \in \mathsf{T}^{\text{DE}}] \\ &= p^{\max} V^*_{\text{S}^3\text{P}}(s_0) + (1 - p^{\max})\,\mathbf{E}[C(T) | T \in \mathsf{T}^{\text{DE}}] \\ &\ge p^{\max} V^*_{\text{S}^3\text{P}}(s_0) + (1 - p^{\max})D\end{aligned}$$

Thus, we are effectively approximating the cost of all traces reaching a dead end by D. Unfortunately, it is not always the case that this extension of the S^{3}P criterion captures the expected cost from s_0 more accurately than Finite-Penalty as shown in the following theorem:

Theorem 2. *Let V^*_{FP} and $V^*_{S^3P}$ be the optimal solution of Finite-Penalty and S^{3}P, respectively, for the same SSP and $D \in \mathbb{R}^*_+$ be the dead-end penalty used by Finite-Penalty. Then $p^{max} V^*_{S^3P}(s_0) + (1 - p^{max})D$ is neither a lower nor an upper bound for V^*_{FP} as a function of D.*

Proof. Consider the SSP in Fig. 2 with the following modifications: (i) there is only action a_0; and (ii) $C(s_0, a_0) = C(s_1, a_0) = c \in \mathbb{R}^*_+$. As shown in Example 1, $p^{\max} = 1/3$ and $V^*_{\text{S}^3\text{P}}(s_0) = 8c/3$. For Finite-Penalty, we have: $V^*_{\text{FP}}(s_0) = \min\{D, c + .5D + .5V^*_{\text{FP}}(s_1)\} = \min\{D, c + .5D + .5(\min\{D, c + .5V^*_{\text{FP}}(s_0)\})\} = \min\{D, 2c + 2D/3\}$. Thus, if $0 < c < 3D/8$ then $V^*_{\text{S}^3\text{P}}(s_0)/p^{\max} + (1 - p^{\max})D = 8c/3 + 2D/3 < V^*_{\text{FP}}(s_0) \le D$ and if $c > 3D/8$, then $V^*_{\text{FP}}(s_0) = D < 8c/3 + 2D/3$. □

Lastly, the following theorem shows that, by considering the traces leading to dead ends, the MCMP criterion captures the expected cost from s_0 more accurately than S^{3}P:

Theorem 3. *Let $V^*_{S^3P}$ and V^*_{MCMP} be the optimal solution of S^{3}P and MCMP, respectively, for the same SSP. Then $p^{max} V^*_{S^3P}(s_0) \le V^*_{MCMP}(s_0)$.*

Proof. This follows from their respective definitions: $V^*_{\text{MCMP}}(s_0) = \mathbf{E}[C(\psi(T)] = p^{\max}\,\mathbf{E}[C(\psi(T)) | T \in \mathsf{T}^{\mathsf{G}}] + (1 - p^{\max})\,\mathbf{E}[C(\psi(T)) | T \in \mathsf{T}^{\text{DE}}] \ge p^{\max}\,\mathbf{E}[C(\psi(T)) | T \in \mathsf{T}^{\mathsf{G}}] = p^{\max} V^*_{\text{S}^3\text{P}}(s_0)$. □

6 MCMP FOR CONSTRAINED SSPs

Another advantage of implementing the MCMP criterion via LP 1 and LP 2 is that we can add cost constraints to both LPs, and thus easily apply the MCMP criterion to Constrained SSPs (C-SSPs) [Altman, 1999]. In a C-SSP, multiple cost functions capture potentially competing objectives (e.g., time, money). One of these objectives is optimized whilst constraining the others.

Formally, a C-SSP $\langle \mathsf{S}, s_0, \mathsf{G}, \mathsf{A}, P, \vec{C}, \vec{u} \rangle$ is an SSP whose cost function is replaced by a vector of $n + 1$ cost functions $\vec{C} = [C_0, \ldots, C_n]$ and a vector of n bounds $\vec{u} = [u_1, \ldots, u_n]$ ($u_j > 0$ for all j). We call C_0 the *primary cost* and the other cost functions the *secondary costs*. An optimal solution for a C-SSP is a *stochastic* policy $\pi : \mathsf{S} \mapsto \mathsf{A} \times [0, 1]$, which minimizes the expected primary cost C_0 to reach the goal G from the s_0, subject to the expected values of the secondary cost C_j being upper bounded by u_j for $j \in \{1, \ldots, n\}$. These stochastic policies are needed to optimally account for trade-offs between the various cost functions, but do not alter the complexity of optimally solving C-SSPs which remains polynomial in the size of the C-SSP [Dolgov and Durfee, 2005].

Currently, the only approach to solve C-SSP with dead ends is Finite-Penalty [Trevizan et al., 2016] and, as shown in Section 3, the MCMP criterion has several advantages over Finite-Penalty. We can apply MCMP to C-SSPs by replacing $C(s, a)$ with $C_0(s, a)$ in LP 1 and augmenting both LP 2 and LP 1 with constraints of the form:

$$\sum_{s \in \mathsf{S}, a \in \mathsf{A}(s)} x_{s,a} C_j(s, a) \leq u_j \qquad \forall j \in \{1, \ldots, n\} \qquad \text{(C9)}$$

As before, i-dual can be used to optimally solve these LPs using heuristic search, provided admissible heuristics $H_j(s) \leq V_j^*(s)$ are available for each of the cost functions C_j. The only difference with the SSP case is that the policies in Π^{MP} are now potentially stochastic.

7 EMPIRICAL EVALUATION

In this section, we empirically evaluate different algorithms for solving SSPs using the S^{3}P and MCMP criteria and C-SSPs using the Finite-Penalty and MCMP criteria. All our experiments were conducted on a cluster of Intel E5-2660 run at 2.60GHz and, unless noted otherwise, we enforced a cputime and memory cutoff of 30 minutes and 10Gb, respectively. We used Gurobi 6.5 as LP solver and our results are averaged over 30 runs using different random seeds. The coverage for each planner is the number of runs in which a given planner successfully finds the optimal solution for the same problem under different random seeds.

7.1 Experiments on SSPs

We implemented two planners for S^{3}P. The first planner computes P^* using FRET [Kolobov et al., 2012] (LRTDP [Bonet and Geffner, 2003] is used as Find-and-Revise algorithm) and then solves P^*-SSP using LRTDP. The second planner is the same as the first one but FRET-π is used instead of FRET. We implemented the P^*-SSP using a lazy evaluation of P^*, i.e., FRET and FRET-π are called on-demand to give values for P^*. Every time we computed $P^*(s)$, we also cached the values of any other state s' in which $P^*(s')$ has also converged. This lazy approach is particularly beneficial for FRET-π since its focused search can find $P^*(s)$ for a single optimal policy as opposed to FRET that converges over the union of the greedy policies; therefore, the heuristic search performed by LRTDP in the second stage can avoid the query of P^* for states that are too expensive to visit. We refer to these planners for S^{3}P as FRET and FRET-π.

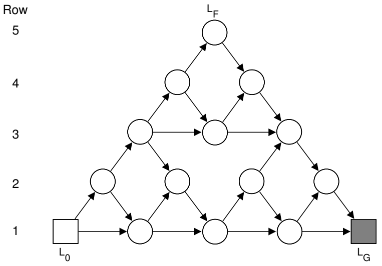


Figure 3: Map for the triangle tireworld problem #2. The goal is to reach location L_G from location L_0. Location L_F represents the opposite corner from L_0 and L_G.

To solve SSPs using the MCMP criterion, we implemented i-dual to solve both LP 2 and LP 1 and use the solution of the former to warm start the latter. For all planners considered in this section, we used $h^{\max}$ over the all-outcomes determinization as heuristic for both stages [Teichteil-Königsbuch et al., 2011]. The Max-Prob version of $h^{\max}$ returns 0 if no solution is found, i.e., a queried state is a dead end, and 1 otherwise. We consider problems in the following domains:

Exploding Blocks World. From the IPPC'08 [Bryce and Buffet, 2008], this domain is a probabilistic extension of the deterministic blocks world in which blocks can explode and destroy other blocks or the table. All actions have the same effects as in their deterministic blocks world counterpart and put-down and put-on-block have the probabilistic side effect of detonating the block being held and destroying the table or the block below with probability 0.4 and 0.1, respectively. Once a block or the table is destroyed, nothing can be placed on them, and destroyed blocks cannot be moved; therefore, problems in this domain can have unavoidable dead ends. Also, each block detonates only once and, after its detonation, the block can be safely moved. We use the corrected version of this domain that forbids a block to be placed on top of itself.

Triangle Tireworld. From [Little et al., 2007] and used in the IPPCs, this domain represents a car that has to travel between locations in triangular map (Fig. 3) to reach a goal

			Avg. Time (s)		Avg # States Expl.	
#	planner	cov.	Total	MaxPr	1st stage	2nd stage
1	MCMP	30	0.1	0.1	746	197
	FRET	30	0.5	0.4	3676	92
	FRET-π	30	0.2	0.1	676	95
2	MCMP	30	0.7	0.4	1639	1205
	FRET	30	2.7	2.7	5631	188
	FRET-π	30	9.6	1.6	2112	202
3	MCMP	30	0.2	0.1	730	447
	FRET	30	8.6	8.4	63160	452
	FRET-π	30	54.9	0.6	4871	493
4	MCMP	30	0.7	0.4	1987	1868
	FRET	30	20.5	20.4	32358	1227
	FRET-π	30	48.4	7.2	11826	1337
5	MCMP	30	0.3	0.3	1107	210
	FRET	30	89.0	88.9	60040	249
	FRET-π	30	19.7	14.3	6525	272
7	MCMP	30	114.3	1.3	2675	50519
	FRET-π	7	1568.2	764.8	2316519	77928
9	MCMP	30	108.2	31.2	19209	40553

Table 1: Results for the Exploding Blocks World problems. # is the problem number in IPPC'08.

location from its initial location. When the car moves, a flat tire happens with probability 0.5 and the car becomes unable to move if neither the car nor the location have a spare tire. Problems in this domain are such that only the longest path (i.e., to move from L_0 to L_F then to L_G) has enough spare tires to reach the goal with probability 1; therefore, these problems have avoidable dead ends.

Reversed Triangle Tireworld. This is a variation of the triangle tireworld that we designed to illustrate the tradeoffs between the S^{3}P and MCMP algorithms. The difference between the original and the reverse triangle tireworld is only the location of the spare tires: for the reversed tireworld, the spare tires are available in every locations in rows 1 to r (Fig. 3) where r is a parameter of the problem. Therefore, for all r, the policy representing the shortest path from L_0 to L_G has probability 1 of reaching the goal in the reversed tire world; moreover, this policy is always the optimal policy. As with the original triangle tireworld, the problems in this domain have avoidable dead ends. The reversed tireworld problems highlight the tradeoffs between the different algorithms because the parameter r controls the size of Π^{MP} while maintaining the same optimal policy under all SSP criteria. Moreover, the cost heuristic h^{max} used for solving these problems guides the cost optimization of the different criteria directly to the optimal policy; therefore, narrowing down the differences in performance to the differences in the search approaches over Π^{MP}.

Results. For the exploding blocks world (Tab. 1), i-dual is the only planner to obtain 100% coverage in all the problems and to scale up to problem #9. Moreover, i-dual is always the fastest planner and its speed-up ranges from 2x to 40x compared to the second best planner. The performance gains from i-dual come mainly from its first stage (columns 5 and 6 in Tab. 1) that efficiently computes p^{max} without exploring too many states. This efficiency is due to the search in the dual space which does not contain local maxima, as opposed to the search in the primal space performed by FRET and FRET-π that need to escape local maxima (i.e., fixed-point solutions for P^* that are not optimal). For instance, both FRET and FRET-π need to escape loopy policies in which a detonated block is picked up and then placed in its previous location similarly to loop between d_2 and d_3 in Fig. 1. Notice that these loopy policies can happen in any problem with reversible actions, i.e., loops between doing and undoing an action.

			Avg. Time (s)		Avg # States Expl.	
#	planner	cov.	Total	MaxPr	1st stage	2nd stage
3	MCMP	30	1.4	0.2	2454	6459
	FRET	30	4.3	3.5	41944	17706
	FRET-π	30	3.9	3.1	32151	17750
4	MCMP	30	125.4	1.0	6973	108097
	FRET	30	115.9	98.8	777722	299921
	FRET-π	30	95.8	79.0	555260	299734

Table 2: Results for the Triangle Tireworld problems. # is the problem number in IPPC'08.

For the triangle tireworld (Tab. 2), all planners scaled up to instance #4 and both FRET and FRET-π are faster than i-dual in the largest instance. Although this domain is acyclic and represents the best case scenario for FRET and FRET-π (there is only one fixed-point solution for P^*), i-dual still spends less time computing p^{max}. Moreover, i-dual explores considerably fewer states in both stages: up to 80x less for the first stage and 3x less for the second stage. The underperformance of i-dual is because the triangle tireworld was designed to mislead planners and heuristics based on the all-outcomes determinization: the second stage of i-dual is constantly misdirected towards dead ends by h^{max} and has to infer that those states do not reach the goal with probability 1 (this can be verified by the average time spent in the second stage, i.e., column 5 minus column 6 in Tab. 2). In other words, the main advantage of i-dual, namely the usage of the cost heuristics to prune the search in the Max-Prob space, is lost and becomes an overhead since we still need to compute the heuristic cost. Notice that both FRET and FRET-π do not suffer for the misdirection of h^{max} in the second stage because the P^*-SSP is a trivial SSP since it has only one policy. As we demonstrate in the reversed triangle tireworld experiments, this behavior is an exception caused by the fact that $|\Pi^{\text{MP}}| = 1$.

For the reversed triangle tireworld (Tab. 3), i-dual scales up to all problems while: FRET solves only the instances with small r since $|\Pi^{\text{MP}}|$ is small for them; and FRET-π scales up to all values of r for tire04 but fails to obtain 100% coverage for the larger values of r for tire05. The total time speed up obtained by i-dual is up to 293x and 17x w.r.t. FRET and FRET-π. This large difference in performance is due to how i-dual combines the search for the policy with minimal cost and a policy with maximum probability to reach the goal in its second phase, enabling it to prune large areas of the state space when these are proven too costly, irrespective of their P^* value. Notice that our implementation of the P^*-SSP performs a lazy evaluation of P^*, which also allows both FRET and FRET-π to avoid

				Avg. Time (s)		Avg # States Expl.	
#	r	planner	cov.	Total	MaxPr	1st stage	2nd stage
4	1	MCMP	30	0.8	0.2	1014	3766
		FRET	30	1.3	1.0	6186	3011
		FRET-π	30	1.2	1.0	5877	3006
	2	MCMP	30	3.1	0.7	4949	7500
		FRET	30	92.1	89.8	808958	27396
		FRET-π	30	30.6	29.6	227976	27520
	3	MCMP	30	3.7	0.6	3937	8573
		FRET	30	481.7	470.9	4394767	40018
		FRET-π	30	53.7	52.2	418872	39978
	4	MCMP	30	4.0	0.8	4790	8699
		FRET	30	1174.6	1148.3	10695787	43191
		FRET-π	30	84.3	82.5	654374	43214
	5	MCMP	30	4.2	1.0	5716	8692
		FRET-π	30	83.6	81.7	662169	43260
	6	MCMP	30	4.2	1.0	5491	8695
		FRET-π	30	86.3	84.5	682331	43549
	7	MCMP	30	4.1	0.9	5218	8698
		FRET-π	30	86.2	84.4	681039	43280
	8	MCMP	30	4.2	1.0	5553	8695
		FRET-π	30	87.1	85.3	682876	43280
	9	MCMP	30	4.1	0.9	5211	8688
		FRET-π	30	85.9	84.1	682290	43280
5	1	MCMP	30	6.4	0.4	1948	15780
		FRET	30	8.1	6.9	25192	12223
		FRET-π	30	7.3	6.1	23911	12225
	2	MCMP	30	43.7	2.2	10067	41243
		FRET-π	30	386.0	375.4	1999963	190251
	3	MCMP	30	75.5	2.2	9018	52572
		FRET-π	29	789.9	771.5	4180058	313663
	4	MCMP	30	80.9	2.4	9392	54931
		FRET-π	28	1217.6	1193.7	6599042	359568
	5	MCMP	30	83.3	2.5	10568	55315
		FRET-π	26	1268.4	1244.3	6987555	366708
	6	MCMP	30	83.3	3.6	13869	55300
		FRET-π	25	1384.1	1358.9	7517924	368086
	7	MCMP	30	82.6	3.2	12211	55189
		FRET-π	26	1395.6	1369.6	7635691	366690
	8	MCMP	30	82.4	3.2	12684	55282
		FRET-π	24	1381.5	1356.5	7481011	366047
	9	MCMP	30	79.8	3.6	13588	55261
		FRET-π	25	1374.6	1349.3	7612778	364596
	10	MCMP	30	81.4	3.2	12174	55303
		FRET-π	24	1380.3	1354.9	7489997	366921
	11	MCMP	30	81.0	2.9	11705	55246
		FRET-π	26	1406.8	1381.5	7707530	367452

Table 3: Results for the Reversed Triangle Tireworld problems. # is of the problem number of the original Triangle Tireworld and r is number of rows with spare tires.

querying P^* for regions of the state space that have a high expected cost. Nonetheless, both FRET and FRET-π still compute P^* for a large number of states due to their primal space search procedure: they explore millions of states to answer the P^* queries while i-dual explores less than 15000 states, resulting in up to 1435x and 476x speed-up w.r.t. FRET and FRET-π in the first stage.

The reversed tireworld experiment also shows that the worst-case scenario for FRET and FRET-π is more than simply problems without dead ends, as claimed in [Kolobov et al., 2012, Sec. 8]: rather, their worst-case scenarios are problems with a large number of Max-Prob solutions (i.e., large $|\Pi^{\mathrm{MP}}|$) with small intersection between the reachable states of the policies $\pi \in \Pi^{\mathrm{MP}}$. This is because S^{3}P minimizes the expected cost in the P^*-SSP thus, in the worst case, the complete space of the P^*-SSP is explored and this space is exactly the union of all states reachable from s_0 using any Max-Prob policy.

			Avg Time (s)		Avg # States Expl.		Pr.
r	b	criteria	Total	MaxPr	1st stage	2nd stage	Goal
4	10	$D=25$	19.19	–	–	31769	0.45
		$D=50$	32.50	–	–	42085	0.64
		MCMP	74.23	25.20	38020	48742	0.66
	12	$D=25$	18.72	–	–	32501	0.54
		$D=50$	95.08	–	–	55039	0.76
		MCMP	255.00	96.28	60338	60217	0.76
	14	$D=25$	21.78	–	–	33455	0.62
		$D=50$	66.99	–	–	56489	0.84
		MCMP	145.59	61.71	63504	60256	0.84
	16	$D=25$	21.31	–	–	34954	0.70
		$D=50$	61.47	–	–	59496	0.91
		MCMP	136.12	52.65	68378	60607	0.91
	18	$D=25$	17.95	–	–	30920	0.77
		$D=50$	83.94	–	–	63682	0.99
		MCMP	183.88	77.96	85720	62185	0.99
	20	$D=25$	19.09	–	–	31092	0.83
		$D=50$	68.10	–	–	53574	1.00
		MCMP	174.48	90.05	88807	53581	1.00
5	10	$D=25$	44.84	–	–	55658	0.45
		$D=50$	219.54	–	–	87220	1.00
		MCMP	298.50	11.95	30801	87177	1.00
	12	$D=25$	47.12	–	–	57209	0.54
		$D=50$	146.11	–	–	88210	1.00
		MCMP	213.32	19.30	41866	88276	1.00
	14	$D=25$	51.43	–	–	58558	0.62
		$D=50$	158.94	–	–	90003	1.00
		MCMP	230.69	24.74	47667	89812	1.00
	16	$D=25$	57.42	–	–	58811	0.70
		$D=50$	160.72	–	–	90074	1.00
		MCMP	262.99	58.33	69568	90127	1.00
	18	$D=25$	31.54	–	–	47196	0.77
		$D=50$	163.79	–	–	90329	1.00
		MCMP	257.68	46.60	64807	90352	1.00
	20	$D=25$	31.80	–	–	47295	0.83
		$D=50$	161.53	–	–	90484	1.00
		MCMP	321.12	109.91	92912	90465	1.00

Table 4: Results for the constrained reversed tireworld problems. All problems considered are based on the problem #4 of the original tireworld domain. r and b represent the number of rows with available spare tires and the budget to buy spare tires, respectively. For Finite-Penalty, the dead-end penalty D is displayed on the criteria column. All planners obtained 100% coverage for these problems.

7.2 Experiments on Constrained SSPs

To illustrate the feasibility of solving C-SSPs using the MCMP criterion, we compare i-dual using the Finite-Penalty and MCMP criteria in an extension of the reversed tireworld domain. In this extension, the main cost function is the same as before and represents travel time, and we added a secondary cost function to represent the money spent buying spare tires. For these problems, if a location l has an available spare tire, the planner has to pay \$1, \$5, or \$10 to buy that spare if l is, respectively, a cheap, regular, or expensive location. We set all the outside locations (i.e., in the direct path from L_0 to L_F and L_F to L_G) as cheap locations. All internal locations from rows 1 to $\lceil n/3 \rceil$, $\lceil n/3 \rceil$+1 to 2$\lceil n/3 \rceil$, and 2$\lceil n/3 \rceil$+1 to n (where n is the total number of rows) are, respectively, expensive, regular and cheap. Thus, the more direct a path from L_0 to L_G is, more money is necessary in expectation to buy spare tires.

For the constrained reversed tireworld, we use a cost constraint over the secondary cost function, that is, we enforce a maximum expected amount of money that can be used for buying spare tires. Therefore, if spare tire budget b is too small, no policy will be able to reach the goal with probability 1 even if the unconstrained problem has no dead end. For this experiment, we used the tire04 problem and let the parameters r and b vary, and we also used a different values of dead-end penalty D. Tab. 4 shows the results for selected combinations of r, b, and d. As the last column shows, $D = 25$ is too small as a dead-end penalty and solutions that can reach the goal with higher probability are unduely pruned, while $D = 50$ is large enough for all parametrizations, except $r = 4$ and $b = 10$ in which the MCMP reaches the goal with a larger probability. Regarding runtime, i-dual using the MCMP criterion takes close to the double of the time of the finite-penalty $D = 50$, showing that it is feasible to use MCMP as a criterion for C-SSPs. Moreover, the solution for the Max-Prob stage (i.e., solution of LP 2) is computed quite fast (which can be as small as 1/10 of the total time of the finite-penalty approaches) and already satisfies the cost constraints by design; therefore, the Min-Cost stage can be seen as an iterative refinement procedure that can be stopped at anytime.

8 CONCLUSION AND FUTURE WORK

In this paper, we presented MCMP, a natural optimization criterion for stochastic shortest paths problems with dead ends, which results in the minimum expected cost policy among those with maximum success probability. We also proposed an efficient solution method for MCMP, which leverages recent work on heuristic search in the dual space. Unlike the Finite-Penalty method [Kolobov et al., 2012], our approach does not suffer from having to guess an appropriate dead-end penalty. In comparison to other robust criteria such as S^{3}P and iSSPUDE [Teichteil-Königsbuch, 2012, Kolobov et al., 2012], MCMP more accurately reflects the expected cost of reaching dead ends, and as shown by our experiments, often provides a substantial computational advantage.

This computational advantage comes from applying heuristic search on the dual representation of SSPs: the Max-Prob problem is directly optimized by the flow objective function of the dual space which prevents heuristic search algorithms to be trapped in non-optimal fixed-point solutions. Alternatively, heuristic search algorithms in the primal representation optimize a reward function that indirectly represents the Max-Prob problem and are susceptible to converge to non-optimal fixed-point solutions. Thus, they need to detect if the current fixed-point solution is optimal or not and post-process the solution in the latter case. Moreover, our approach to solve MCMP is able to prune the search space using simultaneously heuristics for both the probability of reaching the goal and the expected cost of reaching the goal. Observe that this computational advantage of our approach does not transfer to S^{3}P or iSSPUDE, as they require (a priori or on-the-fly) the Max-Prob solution.

Another optimization criterion that can avoid the infinite cost associated with being trapped in a dead end is the Average Reward criterion [Puterman, 1994]. For SSPs, this criterion minimizes the average cost of the traces $T_{\pi,s}$ generated by π of size 1 to N for $N \to \infty$. This limiting sum allows policies to always have a finite value regardless of their probability of reaching dead ends. The drawback of the Average Reward criterion is that neither heuristic search algorithms nor heuristics for them have been proposed; therefore the full state space reachable from the initial state must be considered to solve any given problem. As a result, any algorithm for solving SSPs using the Average Reward criterion will not scale up to the large problems considered in this paper.

The related work for explicitly handling the conflict of optimizing goal probability reachability and expected cost includes three classes of approaches. The first, to which MCMP belongs, is ordered objective optimization where goal probability reachability is the primary objective. This includes S^{3}P and iSSPUDE, as well as earlier work studying the merits of Max-Prob, SSP, and discounted MDP models to handle robotics environments with dead ends [Koenig and Liu, 2002].

The second class of approaches are constrained MDPs, where one of the objective is viewed as a constraint while the other is optimized. Work in this area ranges from the seminal work of Altman [1999] to the recent work by Trevizan et al [2016] on i-dual which our approach builds on. Steinmetz et al. [2016] also consider the *At-Least-Prob* criterion for which an extra parameter $\underline{p} \in (0, 1]$ is given and the solution is to find a policy that reaches the goal with probability greater or equal to $\underline{p}$. LP 1 can be trivially adapted by replacing C6 with $\sum_{s_g \in \mathsf{G}} in(s_g) \geq \underline{p}$ to efficiently generate policies that minimize the expected cost whilst satisfying *At-Least-Prob* criterion.

A third class of relevant approaches are multi-objective MDPs viewing goal probability reachability and cost as competing objectives. However, multi-objective approaches are often more appropriate in a setting where a set of alternative policies (e.g. pareto sets of non-dominated policies) needs to be computed because the way of prioritising these objectives is unknown or too difficult to elicit a priori [Roijers et al., 2013]. Also, multi-objective MDPs are rather expensive to solve [Chatterjee et al., 2006], and existing work has focused on discounted MDPs.

As future work, we plan to improve our algorithm for MCMP even further. We suspect that i-dual is somewhat overkill as it can also solve problems requiring stochastic policies whereas MCMP only requires deterministic policies for SSPs. It is known that in the general case, it is harder to compute deterministic than stochastic policies for C-SSPs (NP-hard vs polynomial) [Dolgov and Durfee, 2005]. However, it should be possible to avoid this complexity increase in the case of MCMP and take advantage of the deterministic nature of optimal policies for SSPs.

Acknowledgements

This research was funded by AFOSR grant FA2386-15-1-4015.

References

[Altman, 1999] Altman, E. (1999). *Constrained Markov Decision Processes*, volume 7. CRC Press.

[Bertsekas and Tsitsiklis, 1991] Bertsekas, D. and Tsitsiklis, J. (1991). An Analysis of Stochastic Shortest Path Problems. *Mathematics of Operations Research*, 16(3):580–595.

[Bonet and Geffner, 2003] Bonet, B. and Geffner, H. (2003). Labeled RTDP: improving the convergence of real-time dynamic programming. In *Proc. Int. Conf. on Automated Planning and Scheduling*.

[Bryce and Buffet, 2008] Bryce, D. and Buffet, O. (2008). 6th Int. Planning Competition: Uncertainty Track. In *3rd Int. Probabilistic Planning Competition (IPPC-ICAPS'08)*.

[Chatterjee et al., 2006] Chatterjee, K., Majumdar, R., and Henzinger, T. A. (2006). Markov decision processes with multiple objectives. In *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*, pages 325–336.

[Dolgov and Durfee, 2005] Dolgov, D. A. and Durfee, E. H. (2005). Stationary deterministic policies for constrained mdps with multiple rewards, costs, and discount factors. In *Proc. Int. Joint Conf. on Artificial Intelligence*.

[Koenig and Liu, 2002] Koenig, S. and Liu, Y. (2002). The interaction of representations and planning objectives for decision-theoretic planning tasks. *J. Exp. Theor. Artif. Intell.*, 14(4):303–326.

[Kolobov et al., 2012] Kolobov, A., Mausam, and Weld, D. S. (2012). A theory of goal-oriented mdps with dead ends. In *Proc. Conf. on Uncertainty in Artificial Intelligence (UAI)*.

[Kolobov et al., 2011] Kolobov, A., Mausam, Weld, D. S., and Geffner, H. (2011). Heuristic search for generalized stochastic shortest path mdps. In *Proc. International Conference on Automated Planning and Scheduling (ICAPS)*.

[Little et al., 2007] Little, I., Thiebaux, S., et al. (2007). Probabilistic planning vs. replanning. In *ICAPS Workshop on IPC: Past, Present and Future*.

[Mausam and Kolobov, 2012] Mausam and Kolobov, A. (2012). *Planning with Markov Decision Processes*. Morgan & Claypool.

[Puterman, 1994] Puterman, M. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.

[Roijers et al., 2013] Roijers, D. M., Vamplew, P., Whiteson, S., and Dazeley, R. (2013). A survey of multi-objective sequential decision-making. *J. Artif. Intell. Res. (JAIR)*, 48:67–113.

[Steinmetz et al., 2016] Steinmetz, M., Hoffmann, J., and Buffet, O. (2016). Revisiting foal probability analysis in probabilistic planning. In *Proc. Int. Conf. on Automated Planning and Scheduling*.

[Teichteil-Königsbuch, 2012] Teichteil-Königsbuch, F. (2012). Stochastic safest and shortest path problems. In *Proc. AAAI Conf. on Artificial Intelligence*.

[Teichteil-Königsbuch et al., 2011] Teichteil-Königsbuch, F., Vidal, V., and Infantes, G. (2011). Extending Classical Planning Heuristics to Probabilistic Planning with Dead-Ends. In *Proc. AAAI Conf. on Artificial Intelligence*.

[Trevizan et al., 2017] Trevizan, F., Thiébaux, S., , and Haslum, P. (2017). Occupation Measure Heuristics for Probabilistic Planning. In *Proc. of 27th Int. Conf. on Automated Planning and Scheduling (ICAPS)*.

[Trevizan et al., 2016] Trevizan, F., Thiébaux, S., Santana, P., and Williams, B. (2016). Heuristic Search in Dual Space for Constrained Stochastic Shortest Path Problems. In *Proc. Int. Conf. on Automated Planning and Scheduling (ICAPS)*.

[Younes et al., 2005] Younes, H. L. S., Littman, M. L., Weissman, D., and Asmuth, J. (2005). The first probabilistic track of the international planning competition. *J. Artif. Int. Res.*, 24(1):851–887.