

Bi-objective Lexicographic Optimization in Markov Decision Processes with Related Objectives

Damien Busatto-Gaston, Debraj Chakraborty, Anirban Majumdar, Sayan Mukherjee, Guillermo Pérez, Jean-François Raskin

▶ **To cite this version:**

Damien Busatto-Gaston, Debraj Chakraborty, Anirban Majumdar, Sayan Mukherjee, Guillermo Pérez, et al.. Bi-objective Lexicographic Optimization in Markov Decision Processes with Related Objectives. 21st International Symposium on Automated Technology for Verification and Analysis (ATVA 2023), Oct 2023, Singapore, Singapore. pp.203-223, 10.1007/978-3-031-45329-8_10 . hal-04442545

HAL Id: hal-04442545
https://hal.science/hal-04442545v1

Submitted on 13 May 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization

arXiv:2305.09634v2 [cs.GT] 16 Aug 2023

Bi-Objective Lexicographic Optimization in Markov Decision Processes with Related Objectives*

Damien Busatto-Gaston[1], Debraj Chakraborty[2], Anirban Majumdar[3], Sayan Mukherjee[3], Guillermo A. Pérez[4], and Jean-François Raskin[3]

[1] Université Paris Est Créteil, LACL, F-94010, France
damien.busatto-gaston@u-pec.fr
[2] Masaryk University, Brno, Czech Republic chakraborty@fi.muni.cz
[3] Université Libre de Bruxelles, Brussels, Belgium
{anirban.majumdar,sayan.mukherjee,jean-francois.raskin}@ulb.be
[4] University of Antwerp – Flanders Make, Antwerp, Belgium
guillermo.perez@uantwerpen.be

Abstract. We consider lexicographic bi-objective problems on Markov Decision Processes (MDPs), where we optimize one objective while guaranteeing optimality of another. We propose a two-stage technique for solving such problems when the objectives are related (in a way that we formalize). We instantiate our technique for two natural pairs of objectives: minimizing the (conditional) expected number of steps to a target while guaranteeing the optimal probability of reaching it; and maximizing the (conditional) expected average reward while guaranteeing an optimal probability of staying safe (w.r.t. some safe set of states). For the first combination of objectives, which covers the classical frozen lake environment from reinforcement learning, we also report on experiments performed using a prototype implementation of our algorithm and compare it with what can be obtained from state-of-the-art probabilistic model checkers solving optimal reachability.

Keywords: Markov decision processes · Multi-objective · Synthesis.

1 Introduction

Probabilistic model-checkers, such as Storm [16] or Prism [18], have been developed to solve the model-checking problem for logics like PCTL and models like Markov decision processes. These tools can be used to compute strategies (or schedulers) that maximize the probability of, for instance, reaching a set of

* G. A. Pérez was supported by the iBOF "DESCARTES" and FWO "SAILor" projects. Debraj Chakraborty, Anirban Majumdar, Sayan Mukherjee and Jean-François Raskin were supported by the EOS project *Verifying Learning Artificial Intelligent Systems* (F.R.S.-FNRS and FWO). Debraj Chakraborty was also supported by MASH (MUNI/I/1757/2021) of Masaryk University.

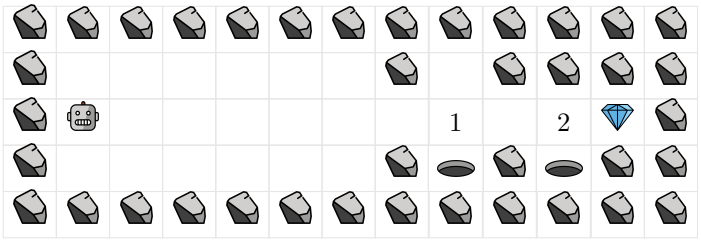


Fig. 1. In the game of Frozen Lake, a robot moves in a slippery grid. It has to reach the target (the gem) while avoiding holes in the grid. The robot can no longer move once in a hole. Part of the grid contains walls and the robot cannot move into them. The frozen surface of the lake being slippery, when the robot tries to move by picking a cardinal direction, the next state is determined stochastically over adjacent directions. For example, trying to move right would result on the robot going to the cell on the right with probability 0.8 but going up or down with probability 0.1 for each.

states. As a concrete example, they can be used to solve the Frozen Lake problem shown in Figure 1, where a robot must navigate from an initial point to a target while avoiding holes in the ice. The ground is frozen and so the movements of the robot are subject to stochastic dynamics. While model-checkers provide optimal strategies for the probability of reaching the target, those strategies may not be efficient in terms of the expected number of steps required to reach it. For instance, the strategy returned by Storm for the grid given in Fig. 1 requires on average 345 steps to reach the target, while there are other strategies that are optimal for reachability that can reach the target in just 34 steps on average. Indeed, a strategy can be optimal in terms of the probability to reach the target while (seemingly) behaving like a random walk on the grid (on portions without holes in particular). In the worst case, one could expect to reach the target after large number of steps (even on grids where there is a short and direct path to target), which can be considered useless for practical purposes.[5] Therefore, in this context, we aim to not only *maximize the probability* of reaching the target, but also *minimize the expected number of steps* required to reach it, which is thus a multi-objective problem. Unfortunately, multi-objective optimization is not yet standard for probabilistic model checkers and most of them support it only for specific combinations of some objectives. For instance, Storm can solve the optimal reachability problem and compute the minimal expected cost to target, but only for target sets that can be reached with probability one. The latter is not usually the case in the Frozen Lake problem: the robot may need to

[5] In particular, the strategy could be used as a component of some larger approach dealing with a more challenging problem too difficult for exact methods. In these cases, such as [7], one frequently relies on machine-learning techniques (*e.g.* Monte-Carlo methods or reinforcement learning) that run simulations for a fixed number of steps. Thus, a strategy that takes needlessly too many steps to reach a target will not help with learning practical and relevant strategies.

walk next to a hole, and risk falling into it, in order to reach the target. In this paper, we demonstrate how to address the problems we have identified with the Frozen Lake example by leveraging the algorithms implemented in Storm.

We identify a family of bi-objective optimization problems that can be solved in two steps using readily available model-checking tools. This family of problems is formalized as follows. Let $\mathcal{M}$ be an MDP and $\Sigma(\mathcal{M})$ the set of all strategies for it. We study reward functions that map strategies $\sigma \in \Sigma(\mathcal{M})$ to real numbers via the induced MC $\mathcal{M}_\sigma$. Concretely, let $f, g : \Sigma(\mathcal{M}) \to \mathbb{R}$. We say a strategy $\sigma \in \Sigma(\mathcal{M})$ is f-optimal if $f(\sigma) = \sup_{\tau \in \Sigma(\mathcal{M})} f(\tau)$ and write Σ_f for the set of all f-optimal strategies.

There are multiple ways in which one can approach the problem of finding optimal strategies with respect to both f and g (see, e.g., [10] and references therein). In this work, we fix a lexicographic order on the functions. Formally, we want to compute a strategy σ such that the following holds:

$$\sigma \in \Sigma_f \text{ and } g(\sigma) = \sup_{\tau \in \Sigma_f} g(\tau) \tag{1}$$

Our contribution. In this paper, we discuss the problem described above for two concrete cases of f and g. First, we tackle the motivating example from Frozen Lake and detail how to find strategies that maximize f, the probability of reaching a set of target states, while minimizing the conditional expected number of steps to reach them, encoded as g. It is not clear how to obtain an exact finite representation of the set Σ_f of all optimal strategies for f. To solve this problem, we first compute an over-approximation Σ_f^{over} of Σ_f. We then prune the original MDP in such a way that the set of all strategies in the pruned MDP is exactly Σ_f^{over}. In this context, Σ_f^{over} will be the set of strategies that only play actions used by at least one optimal strategy for reachability. We then optimize a modified objective g' in the pruned MDP, that, in turn, optimizes both f and g in the lexicographic order in the original MDP. The pruned MDP may contain actions from states that are part of some strategy maximizing the probability of reaching a target but which (taken together) do not make any progress towards the target (for example, a self-loop). These actions, however, are not part of the strategies that optimize g' in the pruned MDP and hence they are also not part of the strategies that are returned by our algorithm. Secondly, we also consider the problem of maximizing the probability of remaining in a safe set of states, encoded as f, while maximizing the expected mean-payoff along safe paths, encoded as g. Unlike the case for reachability, in this problem, we can in fact construct an exact finite representation of Σ_f in the form of an MDP (Theorem 2), which we again construct by pruning the original one. Similar to the reachability case, we then optimize a modified objective g' in the pruned MDP. In both of these cases, we prove (in Theorems 1 and 3) that the strategies optimizing g' in the pruned MDP, are solutions to Eq. 1.

Note that, the solution to the second problem is related to the *shielding* [2] framework and similar works [11,17], where one computes an exact representation of the set of all optimal strategies for the first objective and then solves

for the second objective within that space. However, as remarked earlier, it is unclear how to get an exact representation of Σ_f in the first problem.

In both cases, our solution to these (lexicographic) bi-objective problems can be implemented by using two calls to off-the-shelf tools like Storm or PRISM, thus resulting in a polynomial-time solution. We report on experimental results for the Frozen Lake example that validate the need and practicality of our approach. Finally, we discuss other instances of multi-objective problems where our approach naturally generalizes.

Related works. The strategy synthesis problem in MDPs (or stochastic games, their 2.5-players extension) can be defined for a wide variety of temporal objectives and quantitative rewards. Multi-objective problems are particularly challenging, as they need to optimize for multiple, potentially conflicting, goals. [9] detailed a strategy synthesis algorithm for lexicographic combinations of ω-regular objectives. This problem has also been studied with model-free, reinforcement learning approaches [15]. However, these approaches do not consider objectives that maximize quantitative rewards, and cannot optimize for properties such as the time to reach a target. In [6], one can mix LTL objectives with mean-payoff rewards and in [20] a lexicographic combination of discounted-sum rewards is considered. Moreover, a discounted semantics of LTL[6] is studied in [1], and can be used as a way to optimize for the time until a target is reached. Combinations of LTL and total-reward objectives have been considered in works such as [14] and [12], under assumptions that exclude our problem. Indeed, while minimizing the time to reach a target can be encoded as optimizing the total-reward of a slightly modified structure (where costs are 1 at every move before the target is reached then 0 forever), these works are not directly applicable to our problem: applying [14] requires the assumption that the optimal probability to reach a target is 1 in order to minimize the expected time to target, and [12] searches for a strategy on the Pareto frontier instead of optimizing for a lexicographic combination of objectives.

Note that minimizing the time to reach a target (a variant of the *stochastic shortest path problem* [5]) is only well-defined under the condition that the target is reached, so that our example requires studying *conditional* probabilities. This notion has been studied in single-objective settings, so that for example probabilistic model-checkers can optimize for the (conditional) probability of satisfying an ω-regular event under the condition that another ω-regular event holds [3]. In particular, [4] details how to maximize the expected total-reward until a target is reached, under the assumption that the target is indeed reached with positive probability. This does not solve our motivating example however, as it may yield a strategy that is suboptimal for the probability of reaching the target. Finally, we note that tools such as [8] can handle settings similar to our second example (optimizing for safety and mean-payoff), but they do not consider conditional mean-payoff.

[6] This allows one to express constraints on the number of steps needed to satisfy an Until operator.

Overall, our general two-stage technique covers combinations of objectives that are subcases of problems previously studied (e.g. in [9]) but it is also applicable to combinations not previously considered. Interestingly, and to the best of our knowledge, optimizing for a reachability objective while minimizing the conditional time to satisfy is not formally covered by previous work on multi-objective strategy synthesis, and is not an available feature of probabilistic model-checkers. It may be possible that this problem can be reduced to finding *bias-optimal strategies* [13] in a slightly modified MDP. However, this does not generalize to other objectives.

2 Preliminaries

A *probability distribution* on a countable set S is a function $d : S \to [0, 1]$ such that $\sum_{s \in S} d(s) = 1$. We denote the set of all probability distributions on set S by $\mathcal{D}(S)$. The support of a distribution $d \in \mathcal{D}(S)$ is $\mathsf{Supp}(d) = \{s \in S \mid d(s) > 0\}$.

2.1 Markov Chain

Definition 1 (Markov chain). *A (discrete-time) Markov chain or an MC is a tuple $M = (S, P)$, where S is a countable set of states and P is a mapping from S to $\mathcal{D}(S)$.*

For states $s, s' \in S$, $P(s)(s')$ denotes the probability of moving from state s to state s' in a single transition and we denote this probability $P(s)(s')$ as $P(s, s')$.

For a Markov chain M, a *finite path* $\rho = s_0 s_1 \dots s_i$ of length $i > 0$ is a sequence of $i+1$ consecutive states such that for all $t \in [0, i-1]$, $s_{t+1} \in \mathsf{Supp}(P(s_t))$. We also consider states to be paths of length 0. Similarly, An *infinite path* is an infinite sequence $\rho = s_0 s_1 s_2 \dots$ of states such that for all $t \in \mathbb{N}$, $s_{t+1} \in \mathsf{Supp}(P(s_t))$. For a finite or infinite path $\rho = s_0 s_1 \dots$, we denote its $(i+1)^{th}$ state by $\rho[i] = s_i$. We denote the last state of a finite path $\rho = s_0 s_1 \dots s_n$ by $\mathsf{last}(\rho) = s_n$. Let $\rho = s_0 s_1 \dots s_i$ and $\rho' = s'_0 s'_1 \dots s'_j$ be two paths such that $s_i = s'_0$. Then, $\rho \cdot \rho'$ denotes the path $s_0 s_1 \dots s_i s'_1 \dots s'_j$. For a finite or infinite path $\rho = s_0 s_1 \dots$, we denote its *i-length prefix* as $\rho_{|_i} = s_0 s_1 \dots s_i$.

For a finite path $\rho \in \mathsf{Paths}_M$, we use $\mathsf{Paths}^\omega_M(\rho)$ to denote the set of all paths $\rho' \in \mathsf{Paths}^\omega_M$ such that there exists $\rho'' \in \mathsf{Paths}^\omega_M$ with $\rho' = \rho \cdot \rho''$. $\mathsf{Paths}^\omega_M(\rho)$ is called the *cylinder set* of ρ.

The σ-algebra associated with the MC M is the smallest σ-algebra that contains the cylinder sets $\mathsf{Paths}^\omega_M(\rho)$ for all $\rho \in \mathsf{Paths}_M$. For a state s in S, a measure is defined for the cylinder sets as –

$$\mathbb{P}_{M,s}(\mathsf{Paths}^\omega_M(s_0 s_1 \dots s_i)) = \begin{cases} \prod_{t=0}^{i-1} P(s_t)(s_{t+1}) & \text{if } s_0 = s \\ 0 & \text{otherwise.} \end{cases}$$

We also have $\mathbb{P}_{M,s}(\mathsf{Paths}^\omega_M(s)) = 1$ and $\mathbb{P}_{M,s}(\mathsf{Paths}^\omega_M(s')) = 0$ for $s' \neq s$. This can be extended to a unique probability measure $\mathbb{P}_{M,s}$ on the aforementioned

σ-algebra. In particular, if $\mathcal{C} \subseteq \mathsf{Paths}_M$ is a set of finite paths forming pairwise disjoint cylinder sets, then $\mathbb{P}_{M,s}(\cup_{\rho\in\mathcal{C}}\mathsf{Paths}^{\omega}_M(\rho)) = \sum_{\rho\in\mathcal{C}}\mathbb{P}_{M,s}(\mathsf{Paths}^{\omega}_M(\rho))$. Moreover, if $\Pi \in \mathsf{Paths}^{\omega}_M$ is the complement of a measurable set Π', then $\mathbb{P}_{M,s}(\Pi) = 1 - \mathbb{P}_{M,s}(\Pi')$.

2.2 Markov Decision Process

Definition 2 (Markov decision process). *A Markov decision process or an MDP is a tuple $\mathcal{M} = (S, A, P)$, where S is a finite set of states, A is a finite set of actions, and P is a (partial) mapping from $S \times A$ to $\mathcal{D}(S)$.*

$P(s,a)(s')$ denotes the probability that action a in state s leads to state s' and we denote this probability $P(s,a)(s')$ as $P(s,a,s')$. Note that not all actions may be *legal* from a state. Therefore, if an action a is legal from a state s, we will have $\sum_{s'\in S} P(s,a,s') = 1$. Otherwise, we will have $P(s,a,s')$ is undefined (denoted by $\perp$) for all $s' \in S$.

The definitions and notations used for paths in Markov chain can be extended in the case of MDPs. In an MDP, a *path* is a sequence of states and actions.

For an MDP $\mathcal{M}$, a (probabilistic) *strategy* is a function $\sigma : \mathsf{Paths}_{\mathcal{M}} \to \mathcal{D}(A)$ that maps a finite path ρ to a probability distribution in $\mathcal{D}(A)$. For a path $\rho \in \mathsf{Paths}_{\mathcal{M}}$ and a strategy σ, we will write $\sigma(\rho, a)$ in place of $\sigma(\rho)(a)$. A strategy σ is *deterministic* if the support of the probability distributions $\sigma(\rho)$ has size 1. A strategy σ is *memoryless* if $\sigma(\rho)$ depends only on $\mathsf{last}(\rho)$, i.e. if σ satisfies that for all $\rho, \rho' \in \mathsf{Paths}_{\mathcal{M}}$, $\mathsf{last}(\rho) = \mathsf{last}(\rho') \Rightarrow \sigma(\rho) = \sigma(\rho')$. We denote the set of all finite paths in $\mathcal{M}$ starting from s following σ by $\mathsf{Paths}_{\mathcal{M}}(s,\sigma)$.

An MDP $\mathcal{M}$ induced by a strategy σ defines an MC $\mathcal{M}_\sigma$. Intuitively, this is obtained by unfolding $\mathcal{M}$ using the strategy σ and using the probabilities in $\mathcal{M}$ to define the transition probabilities. Formally, $\mathcal{M}_\sigma = (\mathsf{Paths}_{\mathcal{M}}, P_\sigma)$ where for all paths $\rho \in \mathsf{Paths}_{\mathcal{M}}$, $P_\sigma(\rho)(\rho \cdot as) = \sigma(\rho)(a) \cdot P(\mathsf{last}(\rho), a)(s)$. Thus, a state ρ in $\mathsf{Paths}_{\mathcal{M}}$ uniquely *matches* a finite path ρ' in $\mathcal{M}_\sigma$ where $\mathsf{last}(\rho') = \rho$. This way when a strategy σ and a state s is fixed, the probability measure $\mathbb{P}_{\mathcal{M}_\sigma,s}$ defined in $\mathcal{M}_\sigma$ is also extended for paths in $\mathsf{Paths}_{\mathcal{M}}$. We write the expected value of a random variable X with respect to the probability distribution $\mathbb{P}_{\mathcal{M}_\sigma,s}$ as $\mathbb{E}_{\mathcal{M}_\sigma,s}(X)$. For the ease of notation, we write $\mathbb{P}_{\mathcal{M}_\sigma,s}$ and $\mathbb{E}_{\mathcal{M}_\sigma,s}$ as $\mathbb{P}_{\sigma,s}$ and $\mathbb{E}_{\sigma,s}$ respectively, if the MDP $\mathcal{M}$ is clear from the context. Also, we write $\mathsf{Paths}^{\omega}_{\mathcal{M}_\sigma}(\rho)$ as $\mathsf{Cyl}_\sigma(\rho)$, if the MDP $\mathcal{M}$ is clear from the context.

In the sequel, we make use of (technical) lemmas that follow from the extensive literature on Markov chains and MDPs. However, for completeness, and to give the reader intuition regarding the presented objectives, we also give proofs for some of them.

3 Length-Optimal Strategy for Reachability

We begin by considering the multi-objective problem motivated by the game of frozen lake – the robot tries to reach a target with as few steps as possible while

not compromising on the probability of reaching a target. More formally, in this section, we find a strategy in an MDP that minimizes the expected number of steps to reach some goal states among those strategies that maximize the probability of reaching the goal states.

We consider a set of target states $T \subseteq S$ in $\mathcal{M}$, and assume that every state in T is a sink state, that is, it has only one outgoing action to itself with probability 1. Given a path ρ in an MC $M = (S, P)$, we use $\text{len}_T(\rho)$ to denote the length of the shortest prefix of ρ that reaches one of the states of T, that is, $\text{len}_T(\rho) = i$ if $\rho[i] \in T$ and for all $j < i$, $\rho[j] \notin T$.

For an MDP $\mathcal{M} = (S, A, P)$, let $\mathbb{P}_{\sigma,s}(\Diamond T)$ be the probability of reaching a state in T, starting from $s \in S$, following the strategy σ in $\mathcal{M}$. Then, let $\mathsf{Val}_{\mathcal{M}}(s) = \max_\sigma \mathbb{P}_{\sigma,s}(\Diamond T)$ be the maximum probability to reach T from s, and $\Sigma_{\mathcal{M},s}(\Diamond T) = \arg\max_\sigma \mathbb{P}_{\sigma,s}(\Diamond T)$ be the set of all optimal strategies.

Problem statement.

Given an MDP $\mathcal{M}$, an initial state s_0 and a set of goal states T, our objective is to find a strategy that minimizes $\mathbb{E}_{\sigma,s_0}(\text{len}_T \mid \Diamond T)$ among the strategies in $\Sigma_{\mathcal{M},s_0}(\Diamond T)$, that is, the strategies which maximize $\mathbb{P}_{\sigma,s_0}(\Diamond T)$.

For the rest of this section, we fix the MDP $\mathcal{M} = (S, A, P)$ and a set of target states $T \subseteq S$. Note that, in this case, the functions $\sigma \mapsto \mathbb{P}_{\sigma,s_0}(\Diamond T)$ and $\sigma \mapsto -\mathbb{E}_{\sigma,s_0}(\text{len}_T \mid \Diamond T)$ correspond to the two functions f and g, respectively, and the set $\Sigma_{\mathcal{M},s_0}(\Diamond T)$ corresponds to Σ_f, described in the introduction (Eq. 1).

3.1 Maximizing Probability to Reach a Target

We denote the set $\{(s, a) \in S \times A \mid \mathsf{Val}_{\mathcal{M}}(s) = \sum_{s'} P(s, a, s') \cdot \mathsf{Val}_{\mathcal{M}}(s')\}$ by $\mathsf{Opt}_{\mathcal{M}}$. For $s \in S$, let $\mathsf{Opt}_{\mathcal{M}}(s)$ be the set $\{a \mid (s, a) \in \mathsf{Opt}_{\mathcal{M}}\}$. Finally, we use $\Sigma^{\mathsf{Opt}}_{\mathcal{M}}$ to represent the set of strategies that takes actions according to $\mathsf{Opt}_{\mathcal{M}}$, that is, $\Sigma^{\mathsf{Opt}}_{\mathcal{M}} = \{\sigma \mid \forall \rho, \forall a \in \mathsf{Supp}(\sigma(\rho)); (\mathsf{last}(\rho), a) \in \mathsf{Opt}_{\mathcal{M}}\}$.

Lemma 1. *For every state $s \in S$ and for every $a \in A$,*

$$\mathsf{Val}_{\mathcal{M}}(s) \geq \sum_{s'} P(s, a, s') \cdot \mathsf{Val}_{\mathcal{M}}(s').$$

Proof. Suppose, there is a state $s \in S$ and an action $a \in A$ such that $\mathsf{Val}_{\mathcal{M}}(s) < \sum_{s'} P(s, a, s') \cdot \mathsf{Val}_{\mathcal{M}}(s')$. Now, consider the strategy σ' that takes action a from s and then from paths $s \cdot as'$ follows a strategy $\sigma_{s'} \in \Sigma_{\mathcal{M},s'}(\Diamond T)$ that maximizes the probability to reach states in T from s'. Formally,

$$\sigma'(\rho) = \begin{cases} a & \text{if } \rho = s \\ \sigma_{s'}(\rho') & \text{if } \rho = s \cdot as' \cdot \rho' \end{cases}$$

Then, $\mathbb{P}_{\sigma',s}(\Diamond T) = \sum_{s'} P(s, a, s') \cdot \mathbb{P}_{\sigma_{s'},s'}(\Diamond T) = \sum_{s'} P(s, a, s') \cdot \mathsf{Val}_{\mathcal{M}}(s') > \mathsf{Val}_{\mathcal{M}}(s)$, which is a contradiction as $\mathsf{Val}_{\mathcal{M}}(s) \geq \mathbb{P}_{\sigma,s}(\Diamond T)$ for any σ. □

Lemma 2. *For every state $s \in S$, $\Sigma_{\mathcal{M},s}(\Diamond T) \subseteq \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$.*

Proof. Towards a contradiction, suppose that, there is a strategy $\sigma^* \in \Sigma_{\mathcal{M},s}(\Diamond T)$ such that $\sigma^* \notin \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$. Then there exists a path ρ and an action $a \in \mathsf{Supp}(\sigma^*(\rho))$ such that $(\mathsf{last}(\rho), a) \notin \mathsf{Opt}_{\mathcal{M}}$. Let $\mathsf{last}(\rho) = t$. Then, from Lemma 1 and the fact that $(t, a) \notin \mathsf{Opt}_{\mathcal{M}}$, we get:

$$\mathsf{Val}_{\mathcal{M}}(t) > \sum_{s'} P(t, a, s') \cdot \mathsf{Val}_{\mathcal{M}}(s')\,, \tag{2}$$

and for every other action $a' \neq a$,

$$\mathsf{Val}_{\mathcal{M}}(t) \geq \sum_{s'} P(t, a', s') \cdot \mathsf{Val}_{\mathcal{M}}(s')\,. \tag{3}$$

Consider the strategy $\overline{\sigma}^*$ that differs from σ^* only on paths with ρ as prefix: on every path having ρ as a prefix, $\overline{\sigma}^*$ takes the next action according to a strategy $\sigma_t \in \Sigma_{\mathcal{M},t}(\Diamond T)$ that maximizes the probability to reach a state in T from t, whereas, it takes action according to σ^* on every other path. Formally,

$$\overline{\sigma}^*(\rho') = \begin{cases} \sigma_t(\rho'') & \text{if } \rho' = \rho \cdot \rho'' \\ \sigma^*(\rho') & \text{otherwise.} \end{cases}$$

Note that, for every strategy σ, and for all $a' \in A$, $\mathbb{P}_{\mathcal{M}_{\sigma^*}, \rho \cdot a' s'}(\Diamond T) \leq \mathsf{Val}_{\mathcal{M}}(s')\,.$

$$\begin{aligned} \text{Therefore,} \quad \mathbb{P}_{\sigma^*,\rho}(\Diamond T) &= \sum_{a'} \left(\sigma^*(\rho, a') \cdot \sum_{s'} \left(P(t, a', s') \cdot \mathbb{P}_{\sigma^*, \rho \cdot a' s'}(\Diamond T) \right) \right) \\ &\leq \sum_{a'} \left(\sigma^*(\rho, a') \cdot \sum_{s'} \left(P(t, a', s') \cdot \mathsf{Val}_{\mathcal{M}}(s') \right) \right) \\ &< \sum_{a'} \sigma^*(\rho, a') \cdot \mathsf{Val}_{\mathcal{M}}(t) \quad \text{[from Eq. 2 and 3]} \\ &= \mathsf{Val}_{\mathcal{M}}(t) \end{aligned}$$

So, $\mathbb{P}_{\overline{\sigma}^*,\rho}(\Diamond T) = \mathbb{P}_{\sigma_t,t}(\Diamond T) = \mathsf{Val}_{\mathcal{M}}(t) > \mathbb{P}_{\sigma^*,\rho}(\Diamond T)$. For a finite path ρ and an infinite path ρ', we write $\rho \sqsubseteq \rho'$ if there exists an infinite path ρ'' such that $\rho' = \rho \cdot \rho''$. Now note that, for every strategy σ,

$$\begin{aligned} \mathbb{P}_{\sigma,s}(\Diamond T) &= \mathbb{P}_{\sigma,s}(\rho' \models \Diamond T \wedge \rho \sqsubseteq \rho') + \mathbb{P}_{\sigma,s}(\rho' \models \Diamond T \wedge \rho \not\sqsubseteq \rho') \\ &= \mathbb{P}_{\sigma,s}(\mathsf{Cyl}_{\sigma}(\rho)) \cdot \mathbb{P}_{\sigma,\rho}(\Diamond T) + \mathbb{P}_{\sigma,s}(\rho' \models \Diamond T \wedge \rho \not\sqsubseteq \rho') \end{aligned} \tag{4}$$

Since for any ρ' such that $\rho \not\sqsubseteq \rho'$, $\sigma^*(\rho') = \overline{\sigma}^*(\rho')$, we have $\mathbb{P}_{\sigma^*,s}(\mathsf{Cyl}_{\sigma}(\rho))$ is equal to $\mathbb{P}_{\overline{\sigma}^*,s}(\mathsf{Cyl}_{\overline{\sigma}^*}(\rho))$, and furthermore, $\mathbb{P}_{\sigma^*,s}(\rho' \models \Diamond T \wedge \rho \not\sqsubseteq \rho')$ is equal to $\mathbb{P}_{\overline{\sigma}^*,s}(\rho' \models \Diamond T \wedge \rho \not\sqsubseteq \rho')$. Plugging this into Eq. 4 for σ^* and $\overline{\sigma}^*$, and the fact that $\mathbb{P}_{\sigma^*,\rho}(\Diamond T) < \mathbb{P}_{\overline{\sigma}^*,\rho}(\Diamond T)$, we conclude $\mathbb{P}_{\sigma^*,s}(\Diamond T) < \mathbb{P}_{\overline{\sigma}^*,s}(\Diamond T)$, which contradicts the fact that σ^* is an optimal strategy. □

3.2 Minimizing Expected Conditional Length to Target

In the following, we propose a simple two-step pruning algorithm to solve the multi-objective problem defined earlier in this section. Towards that direction, we first modify the given MDP $\mathcal{M}$ in the following manner.

Definition 3. *We define the* pruned MDP $\mathcal{M}' = (S', A, P')$ *with* $S' = \{s \in S \mid \mathsf{Val}_{\mathcal{M}}(s) > 0\}$ *and* P' *constructed from* P *in the following way:*

$$P'(s,a,s') = \begin{cases} P(s,a,s') \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s')}{\mathsf{Val}_{\mathcal{M}}(s)} & \text{if } (s,a) \in \mathsf{Opt}_{\mathcal{M}} \text{ and } s, s' \in S' \\ \bot & \text{otherwise.} \end{cases}$$

Note that $\mathcal{M}' = (S', A, P')$ is well-defined, since P' is a probability distribution. Indeed, $\sum_{s'} P'(s,a,s') = \sum_{s'} P(s,a,s') \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s')}{\mathsf{Val}_{\mathcal{M}}(s)} = \frac{\mathsf{Val}_{\mathcal{M}}(s)}{\mathsf{Val}_{\mathcal{M}}(s)} = 1$.

From the construction of $\mathcal{M}'$, we get that the set $\Sigma(\mathcal{M}')$ of all strategies in $\mathcal{M}'$ is, in fact, $\Sigma^{\mathsf{Opt}}_{\mathcal{M}}$. Following similar notation as introduced earlier, for a strategy $\sigma \in \Sigma(\mathcal{M}')$, we write $\mathbb{P}_{\mathcal{M}'_\sigma, s}$ and $\mathbb{E}_{\mathcal{M}'_\sigma, s}$ as $\mathbb{P}'_{\sigma,s}$ and $\mathbb{E}'_{\sigma,s}$, respectively. Also, we write $\mathsf{Paths}^{\omega}_{\mathcal{M}'_\sigma}(\rho)$ as $\mathsf{Cyl}'_\sigma(\rho)$.

We now have all the ingredients to present the algorithm:

Algorithm 1

Input: $\mathcal{M} = (S, A, P)$, $s_0 \in S$ and $T \subseteq S$.

1: Create MDP $\mathcal{M}' = (S', A, P')$ according to Definition 3.
2: Find a strategy σ^* that minimizes the expected length in $\mathcal{M}'$:

$$\sigma^* \in \arg\min_\sigma \mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T)\,.$$

3: **return** σ^*.

Note that, the strategies present in the pruned MDP $\mathcal{M}'$ contain every strategy of $\mathcal{M}$ that optimizes the probability of reaching a target (Lemma 2). To show that Algorithm 1 indeed returns a length-optimal strategy maximizing the probability of reachability in $\mathcal{M}$, we need to show the following:

- the strategy given by Algorithm 1 is indeed a strategy that optimizes the probability to reach a target, and
- for every strategy $\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$, the conditional expected length to a target state $\mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T)$ in $\mathcal{M}$ is the same as $\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T)$ in $\mathcal{M}'$. Therefore, it is enough to minimize the expected length in $\mathcal{M}'$.

We first show a relation between the measures of cylinder sets in $\mathcal{M}$ and $\mathcal{M}'$.

Lemma 3. *For every strategy* $\sigma \in \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$ *and for every path* $\rho = s_0 a_0 s_1 \dots s_n \in ((S' \setminus T) \cdot A)^* T \cap \mathsf{Paths}_{\mathcal{M}'}(s_0, \sigma)$, $\mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) = \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}$.

Proof. As $s_n \in T$, $\mathsf{Val}_{\mathcal{M}}(s_n) = 1$. So,

$$\begin{aligned}\mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) &= \prod_{i=0}^{n-1} \sigma(\rho_{|i}, a_i) \cdot P'(s_i, a_i, s_{i+1}) \\ &= \prod_{i=0}^{n-1} \sigma(\rho_{|i}, a_i) \cdot P(s_i, a_i, s_{i+1}) \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s_{i+1})}{\mathsf{Val}_{\mathcal{M}}(s_i)} \\ &= \mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho)) \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s_n)}{\mathsf{Val}_{\mathcal{M}}(s_0)} = \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}\,. \end{aligned}$$ □

Using Lemma 3 we will prove that (cf. Corollary 1) every strategy that maximizes the probability of reaching a target state in $\mathcal{M}$, reaches a target state in $\mathcal{M}'$ with probability 1, and vice versa.

Lemma 4. *For every strategy $\sigma \in \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$, $\mathbb{P}'_{\sigma,s_0}(\Diamond T) = \frac{\mathbb{P}_{\sigma,s_0}(\Diamond T)}{\mathsf{Val}_{\mathcal{M}}(s_0)}$.*

Proof. Note that, $\mathsf{Paths}_{\mathcal{M}'}(s_0) \cap ((S' \setminus T) \cdot A)^* T = \mathsf{Paths}_{\mathcal{M}}(s_0) \cap ((S \setminus T) \cdot A)^* T$, since in the construction of $\mathcal{M}'$ we only remove states of $\mathcal{M}$ from which no state in T is reachable. Therefore, using Lemma 3, we get:

$$\begin{aligned}\mathbb{P}'_{\sigma,s_0}(\Diamond T) &= \sum_{\rho \in \mathsf{Paths}_{\mathcal{M}'}(s_0) \cap ((S' \setminus T)A)^* T} \mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) \\ &= \sum_{\rho \in \mathsf{Paths}_{\mathcal{M}}(s_0) \cap ((S \setminus T)A)^* T} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)} \\ &= \frac{\mathbb{P}_{\sigma,s_0}(\Diamond T)}{\mathsf{Val}_{\mathcal{M}}(s_0)}\,. \end{aligned}$$ □

Corollary 1. *For every $\sigma \in \Sigma(\mathcal{M})$, $\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$ iff $\mathbb{P}'_{\sigma,s_0}(\Diamond T) = 1$.*

Since for every $\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$, $\mathbb{P}_{\sigma,s_0}(\Diamond T \mid \Diamond T) = 1$, we can write:

$$\begin{aligned}\mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T) &= \sum_{r=0}^{\infty} r \cdot \frac{\mathbb{P}_{\sigma,s_0}(\{\rho \mid \rho \models \Diamond T \wedge \mathrm{len}_T(\rho) = r\})}{\mathbb{P}_{\sigma,s_0}(\Diamond T)} \\ &= \sum_{r=0}^{\infty} r \cdot \sum_{\rho \in \mathsf{Paths}_{\mathcal{M}}(s_0) \cap ((S \setminus T)A)^* T : \mathrm{len}_T(\rho) = r} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}\,. \end{aligned}$$

We now relate the expected length of reaching a target state in $\mathcal{M}'$ with the expected conditional length of reaching a target state in $\mathcal{M}$.

Lemma 5. *For any strategy $\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$, $\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T) = \mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T)$.*

Proof. Using $\mathsf{Paths}_{\mathcal{M}'}(s_0) \cap ((S' \setminus T) \cdot A)^* T = \mathsf{Paths}_{\mathcal{M}}(s_0) \cap ((S \setminus T) \cdot A)^* T$, Lemma 3 and Corollary 1, we get:

$$\begin{aligned}\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T) &= \sum_{r=0}^{\infty} r \cdot \mathbb{P}'_{\sigma,s_0}(\{\rho \mid \rho \models \Diamond T \wedge \mathrm{len}_T(\rho) = r\}) \\ &= \sum_{r=0}^{\infty} r \cdot \sum_{\rho \in \mathsf{Paths}_{\mathcal{M}'}(s_0) \cap ((S' \setminus T)A)^* T : \mathrm{len}_T(\rho) = r} \mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) \end{aligned}$$

$$= \sum_{r=0}^{\infty} r \cdot \sum_{\rho \in \mathsf{Paths}_{\mathcal{M}}(s_0) \cap ((S \setminus T)A)^* T : \mathrm{len}_T(\rho) = r} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_{\sigma}(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}$$
$$= \mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T)\,. \qquad \square$$

Finally, we prove the correctness of Algorithm 1:

Theorem 1. *Given an MDP $\mathcal{M} = (S, A, P)$, a state $s_0 \in S$ and $T \subseteq S$, let σ^* be the strategy returned by Algorithm 1. Then,*

1. $\mathbb{P}_{\sigma^*,s_0}(\Diamond T) = \mathsf{Val}_{\mathcal{M}}(s_0)$.
2. $\mathbb{E}_{\sigma^*,s_0}(\mathrm{len}_T \mid \Diamond T) = \min\limits_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)} \mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T)$

Proof. From Corollary 1, we get that $\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T) \neq \infty$ iff $\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$. So if $\sigma^* \notin \Sigma_{\mathcal{M},s_0}(\Diamond T)$, then $\mathbb{E}'_{\sigma^*,s_0}(\mathrm{len}_T) = \infty$. But since for any strategy σ in $\Sigma_{\mathcal{M},s_0}(\Diamond T)$, $\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T) < \infty$, it contradicts the fact that σ^* minimizes $\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T)$. Therefore, $\sigma^* \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$, and hence $\mathbb{P}_{\sigma^*,s_0}(\Diamond T) = \mathsf{Val}_{\mathcal{M}}(s_0)$.

From Lemma 5, we get for any $\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)$,

$$\mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T) = \mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T)$$
$$\implies \operatorname*{arg\,min}_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)} \mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T) = \operatorname*{arg\,min}_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)} \mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T)$$

Hence, $\sigma^* \in \arg\min_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)} \mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T)$ and therefore, we conclude, $\mathbb{E}_{\sigma^*,s_0}(\mathrm{len}_T \mid \Diamond T) = \min\limits_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Diamond T)} \mathbb{E}_{\sigma,s_0}(\mathrm{len}_T \mid \Diamond T)$. $\square$

Note that, constructing the MDP $\mathcal{M}'$ (Line 1 of Algorithm 1) takes polynomial time. Finding a strategy that optimizes $\mathbb{E}'_{\sigma,s_0}(\mathrm{len}_T)$ also takes polynomial time [5]. Therefore, the overall algorithm terminates in polynomial time.

4 Experimental Results

We have made a prototype implementation of the pruning-based algorithm (Algorithm 1) described in Section 3. In this section, we compare the performance (expected number of steps to reach the goal states) of our algorithm with the strategies generated by Storm that (only) maximize the probability of reaching the goal states.

In our MDP, when the robot tries to move by picking a direction, the next state is determined randomly over the neighbouring positions of the robot, according to the following distribution weights: the intended direction gets a weight of 10, and other directions that are not a wall and not the reverse direction of the intended one get a weight of 1, the distribution is then normalized so that the weights sum up to 1.

We generated 100 layouts of size 10×10 where we placed walls in (i) each cell in the border of the grid and (ii) with probability 0.1, at each of other cells. We then placed holes in the remaining empty cells with the same probability.

layouts ($\mathcal{M}$)	$\mathsf{Val}_{\mathcal{M}}(s_0, \Diamond T)$	Shortest distance	$v_{DistOpt}$	v_{Storm}
1	0.66	9	76.48	76.48
2	0.52	18	299.75	629.16
3	1.00	2	2.40	12.12
4	1.00	3	3.44	34.47
5	1.00	6	7.71	137.56
6	0.68	10	264.04	9598.81
7	1.00	5	112.69	9367.02
8	0.91	10	11.49	5879.63
9	1.00	3	3.66	5711.76
10	0.91	5	12.89	149357.57

Table 1. Comparison of the expected conditional length to reach the target for the strategies given by Algorithm 1 ($v_{DistOpt}$) and STORM (v_{Storm}) on some of the randomly generated layouts, sorted by their ratio. 'Shortest distance' refer to the length of the shortest path to the target (without considering the stochastic dynamics of Frozen Lake) and '$\mathsf{Val}_{\mathcal{M}}(s_0, \Diamond T)$' represents the maximum probability of reaching the target from the initial position of the robot (s_0).

Finally, we chose the position of the target and the starting position from the remaining empty cells uniformly at random.

From these layouts, we constructed MDPs described in the PRISM language, a format supported by STORM. For each MDP, we extracted two strategies: (i) a strategy $\sigma_{\text{Storm}} \in \Sigma_{\mathcal{M},s}(\Diamond T)$ that is produced by STORM that optimizes the probability to reach the target, and (ii) σ_{DistOpt}, a strategy that is derived from Algorithm 1. Note that, both of these strategies are optimal for the probability to reach the target. However, the first strategy does not focus on optimizing the length to reach the target. For both of these strategies, we calculate the expected conditional distance to the target in their induced Markov chains. Table 1 reports on our experimental results for a representative subset of the 100 layouts we generated, one of each decile (one layout from the 10 best percents, one from the $10 - 20\%$ range, etc).

Observe that the strategy given by Algorithm 1 does not necessarily suggest following the shortest path, as this may not optimize the first objective (reaching the target with maximum probability). For example, in the layout in Figure 1, the 'shortest' path to the target has length 10. But if we need to maximize the probability to reach the target, from the cell in the grid marked with 1, instead of going right, a better strategy would be to keep going to the cell above and then coming back. This way, the agent will avoid the hole below with certainty, and will eventually go to the right. This is the strategy that Algorithm 1 provides, which has expected conditional length to the target 33.85. On the other hand, the expected conditional length to the target while following the optimal strategy produced by STORM is much larger (345.34). This is because it asks the robot to loop in the 6×3 area in the left. Because of the stochastic dynamics it eventually leaves this area and reaches the target, but it may take a long time, increasing the expected conditional length.

While performing the experiments on the 100 randomly generated layouts, we observed that in 9 layouts out of 10, the expected conditional length (v_{Storm}) for the strategy σ_{Storm} is at least twice the expected conditional length (v_{DistOpt}) for the strategy σ_{DistOpt}. In 69% of the layouts, v_{Storm} values are 10 times worse than the v_{DistOpt} values. In the worst cases (23% of the layouts), v_{Storm} values are at least a 1000 times worse than the v_{DistOpt} values.

5 Safety and Expected Mean Payoff

In this section, we consider another multi-objective problem – as a first objective, we maximize the probability of avoiding a set of states in an MDP, and as a second objective, we maximize the expected conditional Mean Payoff. We propose a pruning-based algorithm, similar to Algorithm 1, to solve this problem.

For this section, we augment the definition of an MDP $\mathcal{M}$ with a *reward* function $R : S \times A \to \mathbb{R}$, where S, A and P are the same as in the previous sections. Furthermore, we consider a set of states $\mathsf{Bad} \subset S$ in $\mathcal{M}$ and assume that every state in Bad is a sink state.

For an MDP $\mathcal{M} = (S, A, P, R)$, let $\mathbb{P}_{\mathcal{M}_\sigma,s}(\Box\neg\mathsf{Bad})$ be the probability of avoiding all states in Bad, starting from $s \in S$, following the strategy σ in $\mathcal{M}$. Then, let $\mathsf{Val}_{\mathcal{M}}(s) = \max_\sigma \mathbb{P}_{\mathcal{M}_\sigma,s}(\Box\neg\mathsf{Bad})$ be the maximum probability to avoid Bad from s, and $\Sigma_{\mathcal{M},s}(\Box\neg\mathsf{Bad}) = \arg\max_\sigma \mathbb{P}_{\mathcal{M}_\sigma,s}(\Box\neg\mathsf{Bad})$ be the set of all optimal strategies for safety.

To formally define the second objective, we first define the *total reward* of horizon n for a path $\rho = s_0 a_0 \dots$ as $\mathsf{Rew}_n(\rho) = \sum_{i=0}^{n-1} R(s_i, a_i)$. Then, for a strategy σ and a state s, the *expected mean-payoff* is defined as

$$\mathbb{E}_{\sigma,s}(\mathsf{MP}) = \liminf_{n\to\infty} \frac{1}{n}\mathbb{E}_{\sigma,s}(\mathsf{Rew}_n)\,.$$

The optimal *expected average reward* starting from a state s in an MDP $\mathcal{M}$ is defined over all strategies σ in $\mathcal{M}$ as $\sup_\sigma \mathbb{E}_{\sigma,s}(\mathsf{MP})$. One can restrict the supremum to the deterministic memoryless strategies [19, section 9.1.4].

We use $\mathbb{E}_{\sigma,s}(\mathsf{Rew}_n \mid \Box\neg\mathsf{Bad})$ to denote the *expected conditional finite horizon reward*. Then the *expected conditional mean-payoff* is defined as

$$\mathbb{E}_{\sigma,s}(\mathsf{MP} \mid \Box\neg\mathsf{Bad}) = \liminf_{n\to\infty} \frac{1}{n}\mathbb{E}_{\sigma,s}(\mathsf{Rew}_n \mid \Box\neg\mathsf{Bad})\,.$$

Intuitively, it represents the expected mean-payoff one would obtain by following the strategy σ and staying safe.

Problem statement.

Given an MDP $\mathcal{M}$, an initial state s_0 and a set of states Bad where $\mathsf{Val}_{\mathcal{M}}(s_0) > 0$, our objective is to find a strategy that maximizes $\mathbb{E}_{\mathcal{M}_\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$ among the strategies in $\Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})$, i.e., the strategies maximizing $\mathbb{P}_{\mathcal{M}_\sigma,s_0}(\Box\neg\mathsf{Bad})$.

For the rest of this section, we fix the MDP $\mathcal{M} = (S, A, P, R)$ and a set of bad states $\mathsf{Bad} \subset S$. Note that, in this case, the functions $\sigma \mapsto \mathbb{P}_{\sigma,s_0}(\Box\neg\mathsf{Bad})$ and $\sigma \mapsto \mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$ correspond to the two functions f and g, respectively, and $\Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})$ corresponds to Σ_f, described in the introduction (Eq. 1).

5.1 Maximizing Probability of Staying Safe

We denote the set $\{(s,a) \in S \times A \mid \mathsf{Val}_{\mathcal{M}}(s) = \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s')\}$ using $\mathsf{Opt}_{\mathcal{M}}$. For $s \in S$, let $\mathsf{Opt}_{\mathcal{M}}(s)$ be the set $\{a \mid (s,a) \in \mathsf{Opt}_{\mathcal{M}}\}$. Finally, we use $\Sigma^{\mathsf{Opt}}_{\mathcal{M}}$ to represent the set of strategies that takes actions according to $\mathsf{Opt}_{\mathcal{M}}$, that is, $\Sigma^{\mathsf{Opt}}_{\mathcal{M}} = \{\sigma \mid \forall \rho, \forall a \in \mathsf{Supp}(\sigma(\rho)); (\mathsf{last}(\rho), a) \in \mathsf{Opt}_{\mathcal{M}}\}$.

We first state the following results, analogous to Lemma 1 and 2 respectively, which can be proved similarly as in the case of reachability.

Lemma 6. *For every state $s \in S \setminus \mathsf{Bad}$ and for every action a,*

$$\mathsf{Val}_{\mathcal{M}}(s) \geq \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s').$$

Lemma 7. *For every state $s \in S$, $\Sigma_{\mathcal{M},s}(\Box\neg\mathsf{Bad}) \subseteq \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$.*

Furthermore, we will show that, unlike reachability, in this case, the other direction of the containment also holds:

Lemma 8. *For every state $s \in S$, $\Sigma_{\mathcal{M},s}(\Box\neg\mathsf{Bad}) \supseteq \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$.*

In order to prove Lemma 8, we first develop a few intermediate results. We start with defining the following notations:

$$\mathsf{UPre}^0(\mathsf{Bad}) = \mathsf{Bad}, \quad \mathsf{UPre}^{i+1}(\mathsf{Bad}) = \{s \mid \forall a, \exists s' \in \mathsf{UPre}^i(\mathsf{Bad}), P(s,a,s') > 0\},$$

$$\mathsf{UPre}^*(\mathsf{Bad}) = \bigcup_{i=0}^{\infty} \mathsf{UPre}^i(\mathsf{Bad}).$$

Furthermore, we define $\mathsf{Good} = S \setminus \mathsf{UPre}^*(\mathsf{Bad})$, $V = S \setminus (\mathsf{Good} \cup \mathsf{Bad})$.

Lemma 9. *For every state $s \in S$, $\mathsf{Val}_{\mathcal{M}}(s) = 1$ iff $s \in \mathsf{Good}$.*

Proof. For $s \in \mathsf{Good}$, $\exists a$ such that $\mathsf{Supp}(P(s,a)) \subseteq \mathsf{Good}$. This gives a strategy to surely avoid Bad, and hence $\mathsf{Val}_{\mathcal{M}}(s) = 1$.

If $s \notin \mathsf{Good}$, then either (i) $s \in \mathsf{Bad}$, in which case $\mathsf{Val}_{\mathcal{M}}(s) = 0$, or (ii) $s \in \mathsf{UPre}^*(\mathsf{Bad})$, and hence $s \in \mathsf{UPre}^i(\mathsf{Bad}) \setminus \mathsf{UPre}^{i-1}(\mathsf{Bad})$ for some i. Then, for every action a, $\mathsf{Supp}(P(s,a)) \cap \mathsf{UPre}^{i-1}(\mathsf{Bad}) \neq \emptyset$. This implies, for any strategy σ, there is a path from s of length at most i reaching Bad following σ. Since this path has a non-zero probability, we therefore get that $\mathsf{Val}_{\mathcal{M}}(s) < 1$. □

For a strategy σ and a finite path ρ, we define the strategy σ_ρ as follows: for any finite path ρ' starting from $\mathsf{last}(\rho)$, $\sigma_\rho(\rho') = \sigma(\rho \cdot \rho')$.

Lemma 10. *For every strategy $\sigma \in \Sigma_{\mathcal{M}}^{\mathsf{Opt}}$, and every finite path ρ in $\mathcal{M}$ following σ, $\mathbb{P}_{\sigma_\rho,\mathsf{last}(\rho)}(\Box\neg\mathsf{Bad}) = 1$ iff $\mathsf{last}(\rho) \in \mathsf{Good}$.*

Proof. We denote $\mathsf{last}(\rho)$ by s. First, let $s \in \mathsf{Good}$. Then we can show that $\forall a \in \mathsf{Opt}_{\mathcal{M}}(s), \mathsf{Supp}(P(s,a)) \subseteq \mathsf{Good}$. Indeed, if there exists an action $a \in \mathsf{Opt}_{\mathcal{M}}(s)$ and a state $s' \in \mathsf{Supp}(P(s,a))$ such that $s' \notin \mathsf{Good}$, then from Lemma 9, $\mathsf{Val}(s') < 1$, which would further imply that

$$\mathsf{Val}_{\mathcal{M}}(s) = \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s') < \sum_{s'} P(s,a,s') = 1\,,$$

which contradicts the fact that $s \in \mathsf{Good}$ (using Lemma 9). So for every strategy σ in $\Sigma_{\mathcal{M}}^{\mathsf{Opt}}$, every path from s following σ_ρ only visits states from Good. Therefore, $\mathbb{P}_{\sigma_\rho,s}(\Box\neg\mathsf{Bad}) = 1$.

To conclude, observe that if $s \in S \setminus \mathsf{Good}$, $\mathbb{P}_{\sigma_\rho,s}(\Box\neg\mathsf{Bad}) \leq \mathsf{Val}_{\mathcal{M}}(s) < 1$. □

In the following, for the ease of notation, for any state $s \in S$ and a strategy σ, we denote $\mathbb{P}_{\sigma,s}(\mathsf{Cyl}_\sigma(\rho))$ by $\mathbf{P}_{\sigma,s}(\rho)$. Recall that, for every action $a \in \mathsf{Opt}_{\mathcal{M}}(s)$, $\mathsf{Val}_{\mathcal{M}}(s) = \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s')$. We can then expand $\mathsf{Val}_{\mathcal{M}}(s)$ as:

$$\mathsf{Val}_{\mathcal{M}}(s) = \sum_{a} \sigma(s,a)\mathsf{Val}_{\mathcal{M}}(s) = \sum_{a} \sigma(s,a) \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s')\,.$$

We can generalize the above statement by unfolding $\mathsf{Val}_{\mathcal{M}}(\cdot)$ for n steps:

Lemma 11. *For every state $s \in S$ and for every strategy $\sigma \in \Sigma_{\mathcal{M}}^{\mathsf{Opt}}$,*

$$\mathsf{Val}_{\mathcal{M}}(s) = \sum_{\rho \in (VA)^n V} \mathbf{P}_{\sigma,s}(\rho) \cdot \mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho)) + \sum_{\rho \in (VA)^{<n}\mathsf{Good}} \mathbf{P}_{\sigma,s}(\rho)$$

The summation in the first term of the above expression is taken over all paths that reach neither Good nor Bad within n steps, whereas the summation in the second term is over all paths that reach some state in Good within n steps. The result in Lemma 11 follows from the following result:

Lemma 12. *For every finite path $\rho = s_0 a_0 s_1 \dots s_n$ of length n and for every strategy σ in $\Sigma_{\mathcal{M}}^{\mathsf{Opt}}$, for all $k < n$:*

$$\mathsf{Val}_{\mathcal{M}}(s_k) = \sum_{\rho' \in (VA)^{n-k}V} \mathbf{P}_{\sigma_{\rho_{|k}},s_k}(\rho') \cdot \mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho')) + \sum_{\rho' \in (VA)^{<n-k}\mathsf{Good}} \mathbf{P}_{\sigma_{\rho_{|k}},s_k}(\rho')\,.$$

Using Lemma 12, we can now prove Lemma 11:

Proof of Lemma 11. Putting $k = 0$ in Lemma 12, we get:

$$\mathsf{Val}_{\mathcal{M}}(s_0) = \sum_{\rho' \in (VA)^n V} \mathbf{P}_{\sigma,s_0}(\rho') \cdot \mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho')) + \sum_{\rho' \in (VA)^{<n}\mathsf{Good}} \mathbf{P}_{\sigma,s_0}(\rho')\,. \qquad \square$$

We now characterize $\mathbb{P}(\cdot)$ in the same way as we did for $\mathsf{Val}(\cdot)$. Note that, for any state s and any strategy σ, we can expand $\mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad})$ as

$$\mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad}) = \sum_a \sigma(s,a) \sum_{s'} P(s,a,s') \cdot \mathbb{P}_{\sigma_{sas'},s'}(\square\neg\mathsf{Bad})\,.$$

Analogous to Lemma 11, we can generalize this statement by unfolding $\mathbb{P}(\cdot)$ for n steps:

Lemma 13. *For every state $s \in S$ and for every strategy $\sigma \in \Sigma_{\mathcal{M}}^{\mathsf{Opt}}$,*

$$\mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad}) = \sum_{\rho\in(VA)^nV} \mathbf{P}_{\sigma,s}(\rho) \cdot \mathbb{P}_{\sigma_\rho,\mathsf{last}(\rho)}(\square\neg\mathsf{Bad}) + \sum_{\rho\in(VA)^{<n}\mathsf{Good}} \mathbf{P}_{\sigma,s}(\rho)\,.$$

Lemma 14. *For every $s \in S$, and every $\sigma \in \Sigma_{\mathcal{M}}^{\mathsf{Opt}}$, $\mathsf{Val}_{\mathcal{M}}(s) = \mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad})$.*

Proof. If $s \in \mathsf{Good}$, $\mathsf{Val}_{\mathcal{M}}(s) = \mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad}) = 1$. If $s \in \mathsf{Bad}$, $\mathsf{Val}_{\mathcal{M}}(s) = \mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad}) = 0$. Finally, if $s \in V$, from Lemma 11 and Lemma 13,

$$\begin{aligned}\mathsf{Val}_{\mathcal{M}}(s) - \mathbb{P}_{\sigma,s}(\square\neg\mathsf{Bad}) &= \sum_{\rho\in(VA)^nV} \mathbf{P}_{\sigma,s}(\rho) \cdot (\mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho)) - \mathbb{P}_{\sigma_\rho,\mathsf{last}(\rho)}(\square\neg\mathsf{Bad})) \\ &< \sum_{\rho\in(VA)^nV} \mathbf{P}_{\sigma,s}(\rho) \text{ [Using Lemma 10]}\end{aligned}$$

For $s \in \mathsf{UPre}^*(\mathsf{Bad})$, there is a path of length at most $|V|$ reaching Bad in $\mathcal{M}_\sigma$. So $\lim_{n\to\infty} \sum_{\rho\in(VA)^nV} \mathbf{P}_{\sigma,s}(\rho) = 0$. □

Lemma 8 follows directly from Lemma 14. Then, using Lemma 7 and 8, we conclude the following theorem:

Theorem 2. *For every state $s \in S$, $\Sigma_{\mathcal{M},s}(\square\neg\mathsf{Bad}) = \Sigma_{\mathcal{M}}^{\mathsf{Opt}}$.*

5.2 Maximizing Expected Conditional Mean Payoff

We propose a simple two-step pruning algorithm, similar to Algorithm 1, to solve the multi-objective problem defined by safety and mean-payoff. We first modify the given MDP $\mathcal{M}$ in the following manner.

Definition 4. *Let $S' = \{s \in S \mid \mathsf{Val}_{\mathcal{M}}(s) > 0\}$. We define $\mathcal{M}' = (S', A, P', R)$ where P' is defined as follows:*

$$P'(s,a,s') = \begin{cases} P(s,a,s') \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s')}{\mathsf{Val}_{\mathcal{M}}(s)} & \text{if } (s,a) \in \mathsf{Opt}_{\mathcal{M}} \text{ and } s \in S' \\ \bot & \text{otherwise.} \end{cases}$$

Note that $\mathcal{M}'$ is again well-defined. We now present the two-step algorithm:

For a state s_0, a strategy σ, and a finite path $\rho = s_0a_0s_1\ldots s_n \in (S'A)^*S' \cap \mathsf{Paths}_{\mathcal{M}'}(s_0,\sigma)$, we define, $\mathsf{GoodCyl}_\sigma(\rho) = \mathsf{Cyl}_\sigma(\rho) \cap \{\rho' \mid \rho' \models \square\neg\mathsf{Bad}\}$. Then, using Lemma 14, we get that if $\sigma \in \Sigma_{\mathcal{M}}^{\mathsf{Opt}}$, then

$$\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho)) = \mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho)) \cdot \mathbb{P}_{\sigma_\rho,s_n}(\square\neg\mathsf{Bad}) = \mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho)) \cdot \mathsf{Val}_{\mathcal{M}}(s_n). \quad (5)$$

Algorithm 2

Input: $\mathcal{M} = (S, A, P, R)$, $s_0 \in S$, $\mathsf{Bad} \subseteq S$

1: Create the MDP $\mathcal{M}' = (S', A, P', R')$ according to Definition 4.
2: Find a strategy σ^* that maximizes the expected mean payoff in $\mathcal{M}'$:

$$\sigma^* \in \arg\max_\sigma \mathbb{E}'_{\sigma,s_0}(\mathsf{MP})\,.$$

3: **return** σ^*.

Lemma 15. *For every strategy $\sigma \in \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$, $s_0 \in S'$ and every finite path $\rho = s_0 a_0 s_1 \dots s_n \in (S'A)^* S' \cap \mathsf{Paths}_{\mathcal{M}'}(s_0, \sigma)$, $\mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) = \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}$.*

Proof.

$$\begin{aligned}
\mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) &= \prod_{i=0}^{n-1} \sigma(\rho_{|i}, a_i) \cdot P'(s_i, a_i, s_{i+1}) \\
&= \prod_{i=0}^{n-1} \sigma(\rho_{|i}, a_i) \cdot P(s_i, a_i, s_{i+1}) \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s_{i+1})}{\mathsf{Val}_{\mathcal{M}}(s_i)} \\
&= \mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho)) \cdot \frac{\mathsf{Val}_{\mathcal{M}}(s_n)}{\mathsf{Val}_{\mathcal{M}}(s_0)} = \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)} \quad \text{[from Eq. 5]} \ \square
\end{aligned}$$

We now show the following correlation between the expected mean-payoff in $\mathcal{M}'$ and the expected conditional mean-payoff in $\mathcal{M}$:

Lemma 16. *For every strategy σ, $\mathbb{E}'_\sigma(\mathsf{MP}) = \mathbb{E}_\sigma(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$*

Proof. For $r \in \mathbb{R}$, we define $\xi_r = \{\rho \in \mathsf{Paths}_{\mathcal{M}}(s_0) \cap (SA)^n S \mid \mathsf{Rew}_n(\rho) = r\}$ and $\xi'_r = \{\rho \in \mathsf{Paths}_{\mathcal{M}'}(s_0) \cap (S'A)^n S' \mid \mathsf{Rew}_n(\rho) = r\}$. Note that for a fixed n, there are finitely many such non-empty ξ_r. From the definition of the conditional expected reward in $\mathcal{M}$, we get:

$$\begin{aligned}
\mathbb{E}_{\sigma,s_0}(\mathsf{Rew}_n \mid \Box\neg\mathsf{Bad}) &= \sum_r r \cdot \frac{\mathbb{P}_{\sigma,s_0}(\{\rho \mid \mathsf{Rew}_n(\rho) = r\} \cap \Box\neg\mathsf{Bad})}{\mathbb{P}_{\sigma,s_0}(\Box\neg\mathsf{Bad})} \\
&= \sum_r r \cdot \sum_{\rho\in\xi_r} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{Cyl}_\sigma(\rho) \cap \Box\neg\mathsf{Bad})}{\mathsf{Val}_{\mathcal{M}}(s_0)} \\
&= \sum_r r \cdot \sum_{\rho\in\xi_r} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}\,.
\end{aligned}$$

$$\begin{aligned}
\text{Then, } \mathbb{E}'_{\sigma,s_0}(\mathsf{Rew}_n) &= \sum_r r \cdot \mathbb{P}'_{\sigma,s_0}(\{\rho \mid \mathsf{Rew}_n(\rho) = r\}) = \sum_r r \cdot \sum_{\rho\in\xi'_r} \mathbb{P}'_{\sigma,s_0}(\mathsf{Cyl}'_\sigma(\rho)) \\
&= \sum_r r \cdot \sum_{\rho\in\xi'_r} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)}
\end{aligned}$$

$$= \sum_{r} r \cdot \sum_{\rho \in \xi_r} \frac{\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho))}{\mathsf{Val}_{\mathcal{M}}(s_0)} \tag{6}$$

$$= \mathbb{E}_{\sigma,s_0}(\mathsf{Rew}_n \mid \Box\neg\mathsf{Bad}) \tag{7}$$

The equality in Eq. 6 is due to the fact that for any finite path $\rho = s_0 \dots s_n \in (SA)^n S \setminus (S'A)^n S'$, $\exists i$ s.t. $\mathsf{Val}_{\mathcal{M}}(s_i) = 0$, which implies, $\mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(\rho)) \leq \mathbb{P}_{\sigma,s_0}(\mathsf{GoodCyl}_\sigma(s_0 \dots s_i)) = \mathbb{P}_{\sigma,s_0\dots s_i}(\Box\neg\mathsf{Bad}) \leq \mathsf{Val}_{\mathcal{M}}(s_i) = 0$.

Finally, dividing by n and taking limit on the both sides of Eq. 7, we get $\mathbb{E}'_{\sigma,s_0}(\mathsf{MP}) = \mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$. □

Now we prove the correctness of Algorithm 2:

Theorem 3. *Given an MDP $\mathcal{M} = (S, A, P, R)$, a state $s_0 \in S$ and $\mathsf{Bad} \subset S$, let σ^* be the strategy returned by Algorithm 2. Then,*

1. $\mathbb{P}_{\sigma^*,s_0}(\Box\neg\mathsf{Bad}) = \mathsf{Val}_{\mathcal{M}}(s_0)$.
2. $\mathbb{E}_{\sigma^*,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad}) = \max_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})} \mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$

Proof. From Theorem 2, for any σ in $\Sigma^{\mathsf{Opt}}_{\mathcal{M}}$, $\mathbb{P}_{\sigma,s_0}(\Box\neg\mathsf{Bad}) = \mathsf{Val}_{\mathcal{M}}(s_0)$. Note that a strategy in $\mathcal{M}'$ would be in $\Sigma^{\mathsf{Opt}}_{\mathcal{M}}$. Therefore, $\mathbb{P}_{\sigma^*,s_0}(\Box\neg\mathsf{Bad}) = \mathsf{Val}_{\mathcal{M}}(s_0)$.

From Lemma 16, we get for any σ,

$$\mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad}) = \mathbb{E}'_{\sigma,s_0}(\mathsf{MP})$$

$$\Rightarrow \operatorname*{arg\,max}_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})} \mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad}) = \operatorname*{arg\,max}_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})} \mathbb{E}'_{\sigma,s_0}(\mathsf{MP})$$

Hence, $\sigma^* \in \arg\max_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})} \mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$ and therefore, we conclude, $\mathbb{E}_{\sigma^*,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad}) = \max_{\sigma \in \Sigma_{\mathcal{M},s_0}(\Box\neg\mathsf{Bad})} \mathbb{E}_{\sigma,s_0}(\mathsf{MP} \mid \Box\neg\mathsf{Bad})$. □

Note that, constructing the MDP $\mathcal{M}'$ (Line 1 of Algorithm 2) takes polynomial time. Finding a strategy that optimizes $\mathbb{E}'_{\sigma,s_0}(\mathsf{MP})$ also takes polynomial time [19, Chapter 9]. Therefore, the overall algorithm takes polynomial time.

6 Discussion

The work presented in this article proposes a pruning-based approach (Algorithms 1, 2) that can be used to solve certain multi-objective problems in MDPs. The algorithms work by first pruning the given MDP based on the first objective, and then solving the (possibly simplified) second objective on the pruned MDP. Note that, optimizing the second objective, in turn, optimizes both of the objectives in the lexicographic order.

The case where the first objective is to maximize the probability of reaching a set of (target) states in an MDP and the second objective is to minimize the conditional expected time to reach the same set of states, has been discussed in Section 3. Note that one can consider more general (positive) cost functions

and try to minimize the conditional expected cost to reach the target states as a secondary objective, keeping the first objective unchanged.

Based on a suggestion by Jakob Piribauer, we conjecture that the second objective considered in this paper can, in fact, be replaced by any measurable function. More precisely, when the first objective is to remain safe, our technique can be applied to solve the bi-objective problem where the second objective is to optimize the expected value of a measurable function g, conditioned on the event that safety is satisfied. To this end, we can prove the following result: every strategy in the original MDP $\mathcal{M}$ that maximizes $\mathbb{E}(g \mid \Box\neg\mathsf{Bad})$ while maximizing the probability of staying safe, also maximizes the expected value of g in the pruned MDP $\mathcal{M}'$, that is,

$$\sup_{\sigma \in \Sigma^{\mathcal{M}}_{\mathsf{Safe}}} \mathbb{E}_\sigma(g \mid \Box\neg\mathsf{Bad}) = \sup_{\sigma \in \Sigma(\mathcal{M}')} \mathbb{E}'_\sigma(g)$$

where $\Sigma^{\mathcal{M}}_{\mathsf{Safe}}$ denotes the set of all strategies that maximize the probability of staying safe in $\mathcal{M}$. We believe this result can be proved by generalizing the proof of Lemma 16.

Similarly, when the primary objective is to reach a set of target states with as high probability as possible, we believe our technique will be able to compute the optimal strategy when the secondary objective is given by any measurable function g. We conjecture that the following result will hold: any strategy in $\mathcal{M}$ that first maximizes the probability to reach a target and further maximizes the expected value of a measurable function g conditioned on reaching a target state, will also maximize the (unconditional) expected value of g in the pruned MDP $\mathcal{M}'$, among the strategies that reach a target almost surely, that is, with probability 1. More formally, we can obtain the following result:

$$\sup_{\sigma \in \Sigma^{\mathcal{M}}_{\mathsf{Reach}}} \mathbb{E}_\sigma(g \mid \Diamond T) = \sup_{\sigma \in \Sigma^{\mathcal{M}'}_{\mathsf{a.s.Reach}}} \mathbb{E}'_\sigma(g)$$

where $\Sigma^{\mathcal{M}'}_{\mathsf{a.s.Reach}}$ is the set of all strategies in $\mathcal{M}'$ that, when followed, forces $\mathcal{M}'$ to reach a target state with probability 1. Further, if it is the case that every strategy in $\mathcal{M}'$ maximizing the (unconditional) expected value of g reaches a target with probability 1 (which was the case in the pair of objectives considered in Section 3), then the problem reduces to finding a strategy in $\mathcal{M}'$ that maximizes the (unconditional) expected value of g among all strategies, that is,

$$\sup_{\sigma \in \Sigma^{\mathcal{M}}_{\mathsf{Reach}}} \mathbb{E}_\sigma(g \mid \Diamond T) = \sup_{\sigma \in \Sigma(\mathcal{M}')} \mathbb{E}'_\sigma(g)\,.$$

While we studied only two-dimensional lexicographic objectives for the sake of clarity and simplicity, we note that our work can be straight-forwardly extended to more than two reward structures. For example, one may want to optimize for safety first, reachability second, and minimal expected time to reach a target as a third objective. In this case, we would proceed in three steps: a first pruning of the MDP that solves the safety problem, a second pruning that over-approximate the winning strategies for reachability, and finally we would minimize the expected distance.

References

1. Almagor, S., Boker, U., Kupferman, O.: Discounting in LTL. In: Ábrahám, E., Havelund, K. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 424–439. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
2. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, (AAAI 2018). pp. 2669–2678. AAAI Press (2018)
3. Baier, C., Klein, J., Klüppelholz, S., Märcker, S.: Computing conditional probabilities in Markovian models efficiently. In: Ábrahám, E., Havelund, K. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 515–530. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
4. Baier, C., Klein, J., Klüppelholz, S., Wunderlich, S.: Maximizing the conditional expected reward for reaching the goal. In: Legay, A., Margaria, T. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 269–285. Springer Berlin Heidelberg, Berlin, Heidelberg (2017)
5. Bertsekas, D.P., Tsitsiklis, J.N.: An analysis of stochastic shortest path problems. Math. Oper. Res. **16**(3), 580–595 (1991), https://doi.org/10.1287/moor.16.3.580
6. Bohy, A., Bruyère, V., Filiot, E., Raskin, J.F.: Synthesis from LTL specifications with mean-payoff objectives. In: Piterman, N., Smolka, S.A. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 169–184. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
7. Chakraborty, D., Busatto-Gaston, D., Raskin, J., Pérez, G.A.: Formally-sharp dagger for MCTS: lower-latency monte carlo tree search using data aggregation with formal methods. In: Agmon, N., An, B., Ricci, A., Yeoh, W. (eds.) Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023. pp. 1354–1362. ACM (2023), https://dl.acm.org/doi/10.5555/3545946.3598783
8. Chatterjee, K., Henzinger, T.A., Jobstmann, B., Singh, R.: QUASY: Quantitative synthesis tool. In: Abdulla, P.A., Leino, K.R.M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 267–271. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
9. Chatterjee, K., Katoen, J.P., Mohr, S., Weininger, M., Winkler, T.: Stochastic games with lexicographic objectives. Formal Methods in System Design (Mar 2023), https://doi.org/10.1007/s10703-023-00411-4
10. Chatterjee, K., Majumdar, R., Henzinger, T.A.: Markov decision processes with multiple objectives. In: Durand, B., Thomas, W. (eds.) STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings. Lecture Notes in Computer Science, vol. 3884, pp. 325–336. Springer (2006), https://doi.org/10.1007/11672142_26
11. Chatterjee, K., Novotný, P., Pérez, G.A., Raskin, J., Zikelic, D.: Optimizing expectation with guarantees in POMDPs. In: Singh, S., Markovitch, S. (eds.) Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. pp. 3725–3732. AAAI Press (2017), http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14354
12. Chen, T., Kwiatkowska, M., Simaitis, A., Wiltsche, C.: Synthesis for multi-objective stochastic games: An application to autonomous urban driving. In: Joshi, K., Siegle, M., Stoelinga, M., D'Argenio, P.R. (eds.) Quantitative Evaluation of Systems. pp. 322–337. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)

13. Denardo, E.V.: Computing a bias-optimal policy in a discrete-time Markov decision problem. Operations Research **18**(2), 279–289 (1970), http://www.jstor.org/stable/168684
14. Forejt, V., Kwiatkowska, M., Norman, G., Parker, D., Qu, H.: Quantitative multi-objective verification for probabilistic systems. In: Abdulla, P.A., Leino, K.R.M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 112–127. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
15. Hahn, E.M., Perez, M., Schewe, S., Somenzi, F., Trivedi, A., Wojtczak, D.: Model-free reinforcement learning for lexicographic omega-regular objectives. In: Huisman, M., Păsăreanu, C., Zhan, N. (eds.) Formal Methods. pp. 142–159. Springer International Publishing, Cham (2021)
16. Hensel, C., Junges, S., Katoen, J., Quatmann, T., Volk, M.: The probabilistic model checker Storm. Int. J. Softw. Tools Technol. Transf. **24**(4), 589–610 (2022), https://doi.org/10.1007/s10009-021-00633-z
17. Junges, S., Jansen, N., Dehnert, C., Topcu, U., Katoen, J.: Safety-constrained reinforcement learning for MDPs. In: Chechik, M., Raskin, J. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings. Lecture Notes in Computer Science, vol. 9636, pp. 130–146. Springer (2016), https://doi.org/10.1007/978-3-662-49674-9_8
18. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) Proc. 23rd International Conference on Computer Aided Verification (CAV'11). LNCS, vol. 6806, pp. 585–591. Springer (2011)
19. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley Series in Probability and Statistics, Wiley (1994). https://doi.org/10.1002/9780470316887
20. Skalse, J., Hammond, L., Griffin, C., Abate, A.: Lexicographic multi-objective reinforcement learning. In: Raedt, L.D. (ed.) Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22. pp. 3430–3436. International Joint Conferences on Artificial Intelligence Organization (7 2022), https://doi.org/10.24963/ijcai.2022/476, main Track

A Missing proofs of Section 5

A.1 Proof of Lemma 6

Suppose, there is a state $s \in S \setminus \mathsf{Bad}$ and an action $a \in A$ such that $\mathsf{Val}_{\mathcal{M}}(s) < \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s')$. Now, consider the strategy σ' that takes action a from s and then from paths $s \cdot as'$ follows a strategy $\sigma_{s'} \in \Sigma_{\mathcal{M},s'}(\Box\neg\mathsf{Bad})$ that maximizes the probability to avoid states in Bad. Formally,

$$\sigma'(\rho) = \begin{cases} a & \text{if } \rho = s \\ \sigma_{s'}(\rho') & \text{if } \rho = s \cdot as' \cdot \rho' \end{cases}$$

Then, we get the following: $\mathbb{P}_{\mathcal{M}_{\sigma'},s}(\Box\neg\mathsf{Bad}) = \sum_{s'} P(s,a,s') \cdot \mathbb{P}_{\sigma_{s'},s'}(\Box\neg\mathsf{Bad}) = \sum_{s'} P(s,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s') > \mathsf{Val}_{\mathcal{M}}(s)$, which is a contradiction.

A.2 Proof of Lemma 7

Suppose that there is a strategy $\sigma^* \in \Sigma_{\mathcal{M},s}(\Box\neg\mathsf{Bad})$ where there exists a path ρ and an action $a \in \mathsf{Supp}(\sigma^*(\rho))$ such that $(\mathsf{last}(\rho), a) \notin \mathsf{Opt}_{\mathcal{M}}$. Let $\mathsf{last}(\rho) = t$. Assume that $\rho \models \Box\neg\mathsf{Bad}$. Then, from Lemma 6 and the fact that $(t,a) \notin \mathsf{Opt}_{\mathcal{M}}$,

$$\mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho)) > \sum_{s'} P(t,a,s') \cdot \mathsf{Val}_{\mathcal{M}}(s')$$

and for every other action $a' \neq a$,

$$\mathsf{Val}_{\mathcal{M}}(t) \geq \sum_{s'} P(t,a',s') \cdot \mathsf{Val}_{\mathcal{M}}(s')\,.$$

Consider the strategy σ'' which differs from σ^* only on paths with ρ as prefix: on every path having ρ as a prefix, σ'' takes the next action according to a strategy $\sigma_t \in \Sigma_{\mathcal{M},t}(\Box\neg\mathsf{Bad})$ that maximizes the probability to avoid states in Bad from $\mathsf{last}(\rho)$, whereas, it takes action according to σ^* on every other path. Formally,

$$\sigma''(\rho') = \begin{cases} \sigma_t(\rho'') & \text{if } \rho' = \rho \cdot \rho'' \\ \sigma^*(\rho') & \text{otherwise.} \end{cases}$$

Note that, for every strategy σ,

$$\mathbb{P}_{\mathcal{M}_\sigma,\rho}(\Box\neg\mathsf{Bad}) = \sum_{a'} \left(\sigma(\rho,a') \cdot \sum_{s'} \left(P(t,a',s') \cdot \mathbb{P}_{\mathcal{M}_\sigma,\rho\cdot a's'}(\Box\neg\mathsf{Bad}) \right) \right).$$

Also, $\mathbb{P}_{\mathcal{M}_{\sigma^*},\rho\cdot a's'}(\Box\neg\mathsf{Bad}) \leq \mathsf{Val}_{\mathcal{M}}(s')$ for all $a' \in A$. Therefore,

$$\mathbb{P}_{\mathcal{M}_{\sigma^*},\rho}(\Box\neg\mathsf{Bad}) = \sum_{a'} \left(\sigma^*(\rho,a') \cdot \sum_{s'} \left(P(t,a',s') \cdot \mathbb{P}_{\mathcal{M}_{\sigma^*},\rho\cdot a's'}(\Box\neg\mathsf{Bad}) \right) \right)$$

$$\begin{aligned}
&\leq \sum_{a'} \left(\sigma^*(\rho, a') \cdot \sum_{s'} \left(P(t, a', s') \cdot \mathsf{Val}_{\mathcal{M}}(s') \right) \right) \\
&< \sum_{a'} \sigma^*(\rho, a') \cdot \mathsf{Val}_{\mathcal{M}}(t) \\
&= \mathsf{Val}_{\mathcal{M}}(t)
\end{aligned}$$

So, $\mathbb{P}_{\mathcal{M}_{\sigma''},\rho}(\Box\neg\mathsf{Bad}) = \mathbb{P}_{\mathcal{M}_{\sigma_t},t}(\Box\neg\mathsf{Bad}) = \mathsf{Val}_{\mathcal{M}}(t,T) > \mathbb{P}_{\mathcal{M}_{\sigma^*},\rho}(\Box\neg\mathsf{Bad})$. Now note that, for any strategy σ,

$$\begin{aligned}
\mathbb{P}_{\mathcal{M}_\sigma,s}(\Box\neg\mathsf{Bad}) &= \mathbb{P}_{\mathcal{M}_\sigma,s}(\rho' \models \Box\neg\mathsf{Bad} \wedge \rho \sqsubseteq \rho') + \mathbb{P}_{\mathcal{M}_\sigma,s}(\rho' \models \Box\neg\mathsf{Bad} \wedge \rho \not\sqsubseteq \rho') \\
&= \mathbb{P}_{\mathcal{M}_\sigma,s}(\mathsf{Cyl}_\sigma(\rho)) \cdot \mathbb{P}_{\mathcal{M}_\sigma,\rho}(\Box\neg\mathsf{Bad}) + \mathbb{P}_{\mathcal{M}_\sigma,s}(\rho' \models \Box\neg\mathsf{Bad} \wedge \rho \not\sqsubseteq \rho')
\end{aligned}$$

As $\sigma^*(\rho') = \sigma''(\rho')$ for any ρ' such that $\rho \sqsubseteq \rho'$,

$$\mathbb{P}_{\mathcal{M}_{\sigma^*},s}((\mathsf{Cyl}_{\sigma^*}(\rho))) = \mathbb{P}_{\mathcal{M}_{\sigma''},s}((\mathsf{Cyl}_{\sigma''}(\rho)))\,.$$

and

$$\mathbb{P}_{\mathcal{M}_{\sigma^*},s}(\rho' \models \Box\neg\mathsf{Bad} \wedge \rho \not\sqsubseteq \rho') = \mathbb{P}_{\mathcal{M}_{\sigma''},s}(\rho' \models \Box\neg\mathsf{Bad} \wedge \rho \not\sqsubseteq \rho')\,.$$

Then $\mathbb{P}_{\mathcal{M}_{\sigma^*},s}(\Box\neg\mathsf{Bad}) < \mathbb{P}_{\mathcal{M}_{\sigma''},s}(\Box\neg\mathsf{Bad})$, which cannot be true as σ^* is an optimal strategy. Therefore, $\Sigma_{\mathcal{M},s}(\Box\neg\mathsf{Bad}) \subseteq \Sigma^{\mathsf{Opt}}_{\mathcal{M}}$.

A.3 Proof of Lemma 12

We prove this by backward induction on k.
Base case: $k = n-1$. If $s_{n-1} \in \mathsf{Good}$, then $\mathsf{Val}_{\mathcal{M}}(s_{n-1}) = 1$. If $s_{n-1} \in \mathsf{Bad}$, then $\mathsf{Val}_{\mathcal{M}}(s_{n-1}) = 0$. In both cases, the statement is trivially true. If $s_{n-1} \in V$,

$$\begin{aligned}
\mathsf{Val}_{\mathcal{M}}(s_{n-1}) &= \sum_{a_{n-1}} \sigma(\rho_{|_{n-1}}, a_{n-1}) \sum_{s'_n} P(s_{n-1}, a_{n-1}, s'_n) \cdot \mathsf{Val}_{\mathcal{M}}(s'_n) \\
&= \sum_{\rho' \in \{s_{n-1}\}AS} \mathbf{P}_{\sigma_{\rho_{|_{n-1}}},s_{n-1}}(\rho') \cdot \mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho')) \\
&= \sum_{\rho' \in \{s_{n-1}\}AV} \mathbf{P}_{\sigma_{\rho_{|_{n-1}}},s_{n-1}}(\rho') \cdot \mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho')) + \\
&\quad \sum_{\rho' \in \{s_{n-1}\}A\mathsf{Good}} \mathbf{P}_{\sigma_{\rho_{|_{n-1}}},s_{n-1}}(\rho') \times 1 + \sum_{\rho' \in \{s_{n-1}\}A\mathsf{Bad}} \mathbf{P}_{\sigma_{\rho_{|_{n-1}}},s_{n-1}}(\rho') \times 0 \\
&= \sum_{\rho' \in VAV} \mathbf{P}_{\sigma_{\rho_{|_{n-1}}},s_{n-1}}(\rho') \cdot \mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho')) + \sum_{\rho' \in VA\mathsf{Good}} \mathbf{P}_{\sigma_{\rho_{|_{n-1}}},s_{n-1}}(\rho')
\end{aligned}$$

Suppose the statement is true for $k+1$. Then,

$$\begin{aligned}
\mathsf{Val}_{\mathcal{M}}(s_k) &= \sum_{a_k} \sigma(\rho_{|_k}, a_k) \sum_{s'_{k+1}} P(s_k, a_k, s'_{k+1}) \mathsf{Val}_{\mathcal{M}}(s'_{k+1}) \\
&= \sum_{a_k} \sigma(\rho_{|_k}, a_k) \left(\sum_{s'_{k+1} \in V} P(s_k, a_k, s'_{k+1}) \mathsf{Val}_{\mathcal{M}}(s'_{k+1}) + \right.
\end{aligned}$$

$$\left.\sum_{s'_{k+1}\in\mathsf{Good}} P(s_k,a_k,s'_{k+1})\mathsf{Val}_{\mathcal{M}}(s'_{k+1}) + \sum_{s'_{k+1}\in\mathsf{Bad}} P(s_k,a_k,s'_{k+1})\mathsf{Val}_{\mathcal{M}}(s'_{k+1})\right)$$

$$= \sum_{a_k}\sigma(\rho_{|_k},a_k)\left(\sum_{s'_{k+1}\in V} P(s_k,a_k,s'_{k+1})\mathsf{Val}_{\mathcal{M}}(s'_{k+1}) + \sum_{s'_{k+1}\in\mathsf{Good}} P(s_k,a_k,s'_{k+1})\right)$$

$$= \sum_{a_k}\sigma(\rho_{|_k},a_k)\sum_{s'_{k+1}\in V} P(s_k,a_k,s'_{k+1}) \sum_{\rho'\in(\{s'_{k+1}\}\cdot A)^{n-k-1}\cdot V} \mathbf{P}_{\sigma_{\rho_{|_{k+1}}},s'_{k+1}}(\rho')\cdot\mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho'))$$

$$+ \sum_{a_k}\sigma(\rho_{|_k},a_k)\sum_{s'_{k+1}\in V} P(s_k,a_k,s'_{k+1}) \sum_{\rho'\in(\{s'_{k+1}\}\cdot A)^{\leq n-k-1}\cdot\mathsf{Good}} \mathbf{P}_{\sigma_{\rho_{|_{k+1}}},s'_{k+1}}(\rho')$$

$$+ \sum_{a_k}\sigma(\rho_{|_k},a_k)\sum_{s'_{k+1}\in\mathsf{Good}} P(s_k,a_k,s'_{k+1})$$

$$= \sum_{\rho'\in(VA)^{n-k}V} \mathbf{P}_{\sigma_{\rho_{|_k}},s_k}(\rho')\cdot\mathsf{Val}_{\mathcal{M}}(\mathsf{last}(\rho')) + \sum_{\rho'\in(VA)^{<n-k}\mathsf{Good}} \mathbf{P}_{\sigma_{\rho_{|_k}},s_k}(\rho')\,.$$

A.4 Proof of Lemma 13

Here, for the ease of notation, we will use $\mathsf{Val}_\sigma(s)$ to denote $\mathbb{P}_{\sigma,s}(\Box\neg\mathsf{Bad})$.

Lemma 17. *Let $\rho = s_0a_0s_1\dots s_n$ be a finite path of length n. Let σ be a strategy in $\Sigma^{\mathsf{Opt}}_{\mathcal{M}}$. Then for all $k<n$:*

$$\mathsf{Val}_{\sigma_{p_{|_k}}}(s_k) = \sum_{\rho'\in(VA)^{n-k}V} \mathbf{P}_{\sigma_{p_{|_k}},s_k}(\rho')\cdot\mathsf{Val}_{\sigma_{p_{|_k}\cdot\rho'}}(\mathsf{last}(\rho')) + \sum_{\rho'\in(VA)^{<n-k}\mathsf{Good}} \mathbf{P}_{\sigma_{p_{|_k}},s_k}(\rho')$$

Proof. We prove this by backward induction on k. For $k=n-1$. If $s_{n-1}\in\mathsf{Good}$, then $\mathsf{Val}_{\mathcal{M}}(s_{n-1})=1$. If $s_{n-1}\in\mathsf{Bad}$, then $\mathsf{Val}_{\mathcal{M}}(s_{n-1})=0$. In both cases, the statement is trivially true. If $s_{n-1}\in V$,

$$\begin{aligned}
\mathsf{Val}_{\sigma_{p_{|_{n-1}}}}(s_{n-1}) &= \sum_{a'_{n-1}}\sigma(p_{|_{n-1}},a'_{n-1})\sum_{s'_n} P(s_{n-1},a'_{n-1},s'_n)\mathsf{Val}_{\sigma_{p_{|_{n-1}}\cdot a'_{n-1}s'_{n-1}}}(s'_n)\\
&= \sum_{\rho'\in\{s_{n-1}\}AS} \mathbf{P}_{\sigma_{p_{|_{n-1}}},s_{n-1}}(\rho')\cdot\mathsf{Val}_{\sigma_{p_{|_{n-1}}\cdot\rho'}}(\mathsf{last}(\rho'))\\
&= \sum_{\rho'\in\{s_{n-1}\}AV} \mathbf{P}_{\sigma_{p_{|_{n-1}}},s_{n-1}}(\rho')\cdot\mathsf{Val}_{\sigma_{p_{|_{n-1}}\cdot\rho'}}(\mathsf{last}(\rho'))\\
&\qquad + \sum_{\rho'\in\{s_{n-1}\}A\mathsf{Good}} \mathbf{P}_{\sigma_{p_{|_{n-1}}},s_{n-1}}(\rho')\times 1\\
&\qquad + \sum_{\rho'\in\{s_{n-1}\}A\mathsf{Bad}} \mathbf{P}_{\sigma_{p_{|_{n-1}}},s_{n-1}}(\rho')\times 0\\
&= \sum_{\rho'\in VAV} \mathbf{P}_{\sigma_{p_{|_{n-1}}},s_{n-1}}(\rho')\cdot\mathsf{Val}_{\sigma_{p_{|_{n-1}}\cdot\rho'}}(\mathsf{last}(\rho')) + \sum_{\rho'\in VA\mathsf{Good}} \mathbf{P}_{\sigma_{p_{|_{n-1}}},s_{n-1}}(\rho')
\end{aligned}$$

Suppose the statement is true for $k+1$. Then, using Lemma 10,

$$\mathsf{Val}_{\sigma_{p_{|_k}}}(s_k) = \sum_{a'_k}\sigma(p_{|_k},a'_k)\sum_{s'_{k+1}} P(s_k,a'_k,s'_{k+1})\mathsf{Val}_{\sigma_{p_{|_k}\cdot a'_k s'_{k+1}}}(s'_{k+1})$$

$$
\begin{aligned}
&= \sum_{a_k} \sigma(p_{|_k}, a_k) \left(\sum_{s'_{k+1} \in V} P(s_k, a'_k, s'_{k+1}) \mathsf{Val}_{\sigma_{p_{|_k} \cdot a'_k s'_{k+1}}}(s'_{k+1}) \right. \\
&\qquad + \sum_{s'_{k+1} \in \mathsf{Good}} P(s_k, a'_k, s'_{k+1}) \mathsf{Val}_{\sigma_{p_{|_k} \cdot a'_k s'_{k+1}}}(s'_{k+1}) \\
&\qquad \left. + \sum_{s'_{k+1} \in \mathsf{Bad}} P(s_k, a'_k, s'_{k+1}) \mathsf{Val}_{\sigma_{p_{|_k} \cdot a'_k s'_{k+1}}}(s'_{k+1}) \right) \\
&= \sum_{a_k} \sigma(p_{|_k}, a_k) \left(\sum_{s'_{k+1} \in V} P(s_k, a'_k, s'_{k+1}) \mathsf{Val}_{\sigma_{p_{|_k} \cdot a'_k s'_{k+1}}}(s'_{k+1}) \right. \\
&\quad \left. + \sum_{s'_{k+1} \in \mathsf{Good}} P(s_k, a'_k, s'_{k+1}) \right) \\
&= \sum_{a_k} \sigma(p_{|_k}, a_k) \sum_{s'_{k+1} \in V} P(s_k, a'_k, s'_{k+1}) \times \\
&\quad \left(\sum_{\rho' \in (VA)^{n-k-1}V} \mathbf{P}_{\sigma_{p_{|_k} \cdot a'_k s'_{k+1}, s_{k+1}}}(\rho') \cdot \mathsf{Val}_{\sigma_{p_{|_k} \cdot a'_k s'_{k+1} \cdot \rho'}}(\mathsf{last}(\rho')) \right. \\
&\qquad \left. + \sum_{\rho' \in (VA)^{<n-k} \mathsf{Good}} \mathbf{P}_{\sigma_{p_{|_k}, s_k}}(\rho') \right) \\
&\quad + \sum_{a_k} \sigma(p_{|_k}, a_k) \sum_{s'_{k+1} \in \mathsf{Good}} P(s_k, a'_k, s'_{k+1}) \\
&= \sum_{\rho' \in (VA)^{n-k}V} \mathbf{P}_{\sigma_{p_{|_k}, s_k}}(\rho') \cdot \mathsf{Val}_{\sigma_{p_{|_k} \cdot \rho'}}(\mathsf{last}(\rho')) \\
&\quad + \sum_{\rho' \in (VA)^{<n-k} \mathsf{Good}} \mathbf{P}_{\sigma_{p_{|_k}, s_k}}(\rho') \qquad \square
\end{aligned}
$$

Proof of Lemma 13. For $k = 0$ in Lemma 17, we get:

$$
\mathsf{Val}_\sigma(s_0) = \sum_{p' \in (VA)^n V} \mathbf{P}_{\sigma, s_0}(p') \cdot \mathsf{Val}_\sigma(\mathsf{last}(p')) + \sum_{p' \in (VA)^{<n} \mathsf{Good}} \mathbf{P}_{\sigma, s_0}(p') \,. \quad \square
$$