




Simple Strategies in Multi-Objective MDPs*

Technical Report

Florent Delgrange^{1,2,†} , Joost-Pieter Katoen¹ ,
Tim Quatmann¹ , and Mickael Randour²

¹ RWTH Aachen University, Aachen, Germany

² UMONS – Université de Mons, Mons, Belgium

Abstract We consider the verification of multiple expected reward objectives at once on Markov decision processes (MDPs). This enables a trade-off analysis among multiple objectives by obtaining a Pareto front. We focus on strategies that are easy to employ and implement. That is, strategies that are pure (no randomization) and have bounded memory. We show that checking whether a point is achievable by a pure stationary strategy is NP-complete, even for two objectives, and we provide an MILP encoding to solve the corresponding problem. The bounded memory case is treated by a product construction. Experimental results using STORM and GUROBI show the feasibility of our algorithms.

1 Introduction

MDPs. Markov decision processes (MDPs) [4,3] are a key model in stochastic decision making. The classical setting involves a system subject to a stochastic model of its environment, and the goal is to synthesize a system controller, represented as a *strategy* for the MDP, ensuring a given level of *expected performance*. Tools such as PRISM [30] and STORM [16] support MDP model checking.

Multi-objective MDPs. MDPs where the goal is to achieve a *combination* of objectives (rather than just one) are popular in e.g., AI [41] and verification [2]. This is driven by applications, where controllers have to fulfill multiple, potentially conflicting objectives, requiring a *trade-off* analysis. This includes multi-dimension MDPs [14,19,40,13] where weight vectors are aggregated at each step and MDPs where the specification mixes different views (e.g., average and worst case performance) of the same weight [11,8]. With multiple objectives, optimal strategies no longer exist in general: instead, *Pareto-optimal* strategies are considered. The Pareto front, i.e., the set of non-dominated achievable value vectors is usually non-trivial. Elaborate techniques are needed to explore it efficiently, e.g., [22,23].

Simple strategies. Another stumbling block in multi-objective MDPs is the complexity of strategies: Pareto-optimal strategies typically need both *memory* and

[†] currently affiliated with Vrije Universiteit Brussel.

* Research partially supported by F.R.S.-FNRS Grant n° F.4520.18 (*ManySynth*). Mickael Randour is an F.R.S.-FNRS Research Associate.

randomization. A simple conjunction of reachability objectives already requires randomization and exponential memory (in the number of reachability sets) [40]. Some complex objectives even need infinite memory, e.g., [11,8]. In controller synthesis, strategies requiring randomization and/or (much) memory may not be practical. Limited-memory strategies are required on devices with limited resources [7]. Randomization is elegant and powerful from a theoretical view, but has practical limitations, e.g., it limits reproducibility which complicates debugging. Randomized strategies are also often despised for medical applications [33] and product design – all products should have the same design, not a random one. This motivates to consider the analysis of *simple strategies*, i.e., strategies using no randomization and a limited amount of memory (given as a parameter). While most works study the Pareto front among *all* strategies, we establish ways to explore efficiently the Pareto front among *simple* strategies only.

Problem statement. We consider pure (i.e., no randomization) and bounded-memory strategies and study two problems: (a) *achievability* queries – is it possible to achieve a given value vector – and (b) *approximation of the Pareto front*. Considering pure, bounded-memory strategies is natural as randomization can be traded for memory [12]: without randomization, optimal strategies may require arbitrarily large memory, (see Ex. 4). We study mixtures of *expected (accumulated) reward objectives*, covering various studied settings like reachability [19,40], shortest path [39,40,28,9] and total reward objectives [22,23].

Contributions. We first consider the achievability problem for pure stationary (i.e., memoryless) strategies and show that finding optimal strategies for multi-objective MDPs is NP-complete, even for two objectives. This contrasts the case of general strategies, where the problem is polynomial-time if the number of objectives is fixed [40]. We provide a *mixed integer linear program (MILP)* encoding. The crux lies in dealing with end components. The MILP is polynomial in the input MDP and the number of objectives. Inspired by [21], we give an alternative MILP encoding which is better suited for total reward objectives. To approximate the Pareto front under pure stationary strategies, we solve multiple MILP queries. This iteratively divides the solution space into achievable and non-achievable regions. Bounded-memory strategies are treated via a product construction. Our approach works for finite *and* infinite expected rewards.

Practical evaluation. We successfully compute Pareto fronts for 13 benchmarks using our implementation in STORM, exploiting the MILP solver GUROBI. Despite the hard nature of the problem, our experiments show that Pareto fronts for models with tens of thousands of states can be successfully approximated.

Related work. NP completeness for discounted rewards under pure strategies was shown in [14]. [18] claims that this generalizes to PCTL objectives but no proof is given. [42] treats multi-objective bounded MDPs whose transition probabilities are intervals. A set of Pareto optimal policies is computed using policy iteration and an efficient heuristic is exploited to compute a set of mutually non-dominated policies that are likely to be Pareto optimal. Pure stationary Pareto optimal strategies for discounted rewards are obtained in [44] using value-iteration but is restricted to small MDPs where all probabilities are 0 or 1. In [34], Tchebycheff-

optimal strategies for discounted rewards are obtained via an LP approach; such strategies minimize the distance to a reference point and are not always pure.

2 Preliminaries

For a finite set Ω , let $Dist(\Omega) = \{\mu: \Omega \rightarrow [0, 1] \mid \sum_{\omega \in \Omega} \mu(\omega) = 1\}$ be the set of probability distributions over Ω with support $supp(\mu) = \{\omega \in \Omega \mid \mu(\omega) > 0\}$. We write $\mathbb{R}_{\geq 0} = \{|x| \mid x \in \mathbb{R}\}$ and $\mathbb{R}_{\infty} = \mathbb{R} \cup \{\infty\}$ for the non-negative and extended real numbers, respectively. $\mathbf{1}^{\ell} = \langle 1, \dots, 1 \rangle$ denotes the vector of size $\ell \in \mathbb{N}$ with all entries 1. We just write $\mathbf{1}$ if ℓ is clear. Let $\mathbf{p}[i]$ denote the i^{th} entry and $\mathbf{p} \cdot \mathbf{p}'$ the dot product of $\mathbf{p}, \mathbf{p}' \in (\mathbb{R}_{\infty})^{\ell}$. $\mathbf{p} \leq \mathbf{p}'$, $\mathbf{p} + \mathbf{p}'$, and $|\mathbf{p}|$ are entry-wise. For Boolean expression $cond$, let $[cond] = 1$ if $cond$ is true and $[cond] = 0$ otherwise.

2.1 Markov Decision Processes, Strategies, and End Components

Definition 1 (Markov decision process [36]). A Markov decision process (MDP) is a tuple $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$ with finite set of states S , initial state $s_I \in S$, finite set of actions Act , and transition function $\mathbf{P}: S \times Act \times S \rightarrow [0, 1]$ with $\sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{0, 1\}$ for all $s \in S$ and $\alpha \in Act$.

We fix an MDP $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$. Intuitively, $\mathbf{P}(s, \alpha, s')$ is the probability to take a transition from s to s' when choosing action α . An infinite path in \mathcal{M} is a sequence $\pi = s_0 \alpha_1 s_1 \alpha_2 \dots \in (S \times Act)^{\omega}$ with $\mathbf{P}(s_i, \alpha_{i+1}, s_{i+1}) > 0$ for all $i \in \mathbb{N}$. We write $\pi[i] = s_i$ for the $(i+1)$ th state visited by π and define the length of π as $|\pi| = \infty$. A finite path is a finite prefix $\hat{\pi} = s_0 \alpha_1 \dots \alpha_n s_n$ of infinite path π , where $last(\hat{\pi}) = s_n \in S$, $|\hat{\pi}| = n$ and $\hat{\pi}[i] = s_i$ for $i \leq n$. The set of finite (infinite) paths in \mathcal{M} is denoted by $Paths_{\text{fin}}^{\mathcal{M}}$ ($Paths_{\text{inf}}^{\mathcal{M}}$). The *enabled actions* at a state $s \in S$ are given by the set $Act(s) = \{\alpha \in Act \mid \exists s' \in S: \mathbf{P}(s, \alpha, s') > 0\}$. We assume $Act(s) \neq \emptyset$ for all s . If $|Act(s)| = 1$ for all $s \in S$, \mathcal{M} is called a *Markov Chain (MC)*. We write \mathcal{M}_s for the MDP obtained by replacing the initial state of \mathcal{M} by $s \in S$. For $s \in S$ and $\alpha \in Act$, we define the set of successor states $succ(s, \alpha) = \{s' \mid \mathbf{P}(s, \alpha, s') > 0\}$. For $s' \in S$, the set of predecessor state-action pairs is given by $pre(s') = \{\langle s, \alpha \rangle \mid \mathbf{P}(s, \alpha, s') > 0\}$. For a set $\mathcal{E} \subseteq S \times Act$, we define $S[\mathcal{E}] = \{s \in S \mid \exists \alpha: \langle s, \alpha \rangle \in \mathcal{E}\}$, $Act[\mathcal{E}] = \{\alpha \in Act \mid \exists s: \langle s, \alpha \rangle \in \mathcal{E}\}$, and $\mathbf{P}[\mathcal{E}](s, \alpha, s') = [\langle s, \alpha \rangle \in \mathcal{E}] \cdot [s' \in S[\mathcal{E}]] \cdot \mathbf{P}(s, \alpha, s')$. We say \mathcal{E} is *closed* for \mathcal{M} if $\forall \langle s, \alpha \rangle \in \mathcal{E}: \alpha \in Act(s)$ and $succ(s, \alpha) \subseteq S[\mathcal{E}]$.

Definition 2 (Sub-MDP). The sub-MDP of \mathcal{M} , closed $\mathcal{E} \subseteq S \times Act$, and $s \in S[\mathcal{E}]$ is given by $\mathcal{M}[\mathcal{E}, s] = \langle S[\mathcal{E}], Act[\mathcal{E}], \mathbf{P}[\mathcal{E}], s \rangle$. We also write $\mathcal{M}[\mathcal{E}]$ for the sub-MDP $\mathcal{M}[\mathcal{E}, s]$ and an arbitrary state $s \in S[\mathcal{E}]$.

Definition 3 (End Component). A non-empty set $\mathcal{E} \subseteq S \times Act$ is an end component (EC) of \mathcal{M} if \mathcal{E} is closed for \mathcal{M} and for each pair of states $s, s' \in S[\mathcal{E}]$ there is a finite path $\hat{\pi} \in Paths_{\text{fin}}^{\mathcal{M}[\mathcal{E}]}$ with $\hat{\pi}[0] = s$ and $last(\hat{\pi}) = s'$. An EC \mathcal{E} is maximal, if there is no other EC \mathcal{E}' with $\mathcal{E} \subsetneq \mathcal{E}'$. The set of all maximal end components of \mathcal{M} is $MECS(\mathcal{M})$.

The maximal ECs of a Markov chain are also called *bottom strongly connected components (BSCCs)*. A strategy resolves nondeterminism in MDPs:

Definition 4 (Strategy). A (general) strategy for MDP \mathcal{M} is a function $\sigma: Paths_{\text{fin}}^{\mathcal{M}} \rightarrow Dist(Act)$ with $supp(\sigma(\hat{\pi})) \subseteq Act(last(\hat{\pi}))$ for all $\hat{\pi} \in Paths_{\text{fin}}^{\mathcal{M}}$.

Let σ be a strategy for \mathcal{M} . Intuitively, $\sigma(\hat{\pi})(\alpha)$ is the probability to perform action α after observing history $\hat{\pi} \in Paths_{\text{fin}}^{\mathcal{M}}$. A strategy is *pure* if all histories are mapped to *Dirac distributions*, i.e., the support is a singleton. A strategy is *stationary* if its decisions only depend on the current state, i.e., $\forall \hat{\pi}, \hat{\pi}' \in Paths_{\text{fin}}^{\mathcal{M}}: last(\hat{\pi}) = last(\hat{\pi}') \implies \sigma(\hat{\pi}) = \sigma(\hat{\pi}')$. We often assume $\sigma: S \rightarrow Dist(Act)$ for stationary and $\sigma: S \rightarrow Act$ for pure stationary strategies σ . Let $\Sigma^{\mathcal{M}}$ and $\Sigma_{\text{PS}}^{\mathcal{M}}$ be the sets of general and pure stationary strategies, respectively. A set of paths $\Pi \subseteq Paths_{\text{inf}}^{\mathcal{M}}$ is *compliant* with $\sigma \in \Sigma^{\mathcal{M}}$ if for all $\pi = s_0\alpha_1s_1 \cdots \in \Pi$ and prefixes $\hat{\pi}$ of π satisfy $\sigma(\hat{\pi})(\alpha_{|\hat{\pi}|+1}) > 0$. The *induced Markov chain* of \mathcal{M} and $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$ is given by $\mathcal{M}^{\sigma} = \mathcal{M}[\mathcal{E}^{\sigma}, s_I]$ with $\mathcal{E}^{\sigma} = \{(s, \sigma(s)) \mid s \in S\}$.

MDP \mathcal{M} and strategy $\sigma \in \Sigma^{\mathcal{M}}$ induce a probability measure $\text{Pr}_{\sigma}^{\mathcal{M}}$ on subsets $\Pi \subseteq Paths_{\text{inf}}^{\mathcal{M}}$ given by a standard cylinder set construction [4,21]. The expected value of $X: Paths_{\text{inf}}^{\mathcal{M}} \rightarrow \mathbb{R}_{\infty}$ is $E_{\sigma}^{\mathcal{M}}(X) = \int_{\pi} X(\pi) d\text{Pr}_{\sigma}^{\mathcal{M}}(\{\pi\})$. For $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$, $\text{Pr}_{\sigma}^{\mathcal{M}}$ and $E_{\sigma}^{\mathcal{M}}$ coincide with the corresponding measures on MC \mathcal{M}^{σ} .

2.2 Objectives

A reward structure $\mathbf{R}: S \times Act \times S \rightarrow \mathbb{R}_{\geq 0}$ assigns non-negative rewards to transitions. We accumulate rewards on (in)finite paths $\pi = s_0\alpha_1s_1\alpha_2 \dots: \mathbf{R}(\pi) = \sum_{i=1}^{|\pi|} \mathbf{R}(s_{i-1}, \alpha_i, s_i)$. For a set of goal states $G \subseteq S$, let $\mathbf{R} \diamond G(\pi) = \mathbf{R}(\hat{\pi})$, where $\hat{\pi}$ is the smallest prefix of π with $last(\hat{\pi}) \in G$ (or $\hat{\pi} = \pi$ if no such prefix exists). Intuitively, $\mathbf{R} \diamond G(\pi)$ is the reward accumulated on π until a state in G is reached. A (reward) *objective* has the form $\mathbb{E}_{\sim}(\mathbf{R} \diamond G)$ for $\sim \in \{\geq, \leq\}$. We write $\langle \mathcal{M}, \sigma, p \rangle \models \mathbb{E}_{\sim}(\mathbf{R} \diamond G)$ iff $E_{\sigma}^{\mathcal{M}}(\mathbf{R} \diamond G) \sim p$, i.e., for \mathcal{M} and σ , the expected accumulated reward until reaching G is at least (or at most) $p \in \mathbb{R}_{\infty}$. We call the objective *maximizing* if $\sim = \geq$ and *minimizing* otherwise. If $G = \emptyset$ (i.e., $\mathbf{R} \diamond G(\pi) = \mathbf{R}(\pi)$ for all paths π), we call the objective a *total reward objective*. Let the reward structure \mathbf{R}^G be given by $\mathbf{R}(s, \alpha, s') = [s' \in G]$. Then, $\text{Pr}_{\sigma}^{\mathcal{M}}(\diamond G) = E_{\sigma}^{\mathcal{M}}(\mathbf{R}^G \diamond G)$ for every $\sigma \in \Sigma^{\mathcal{M}}$, where $\diamond G \subseteq Paths_{\text{inf}}^{\mathcal{M}}$ denotes the set of paths that visit a state in G . We use $\mathbb{P}_{\sim}(\diamond G)$ as a shortened for $\mathbb{E}_{\sim}(\mathbf{R}^G \diamond G)$ and call such an objective a *reachability objective*.

Definition 5 (Multi-objective query). For MDP \mathcal{M} , an ℓ -dimensional multi-objective query is a tuple $\mathcal{Q} = \langle \psi_1, \dots, \psi_{\ell} \rangle$ of ℓ objectives $\psi_j = \mathbb{E}_{\sim_j}(\mathbf{R}_j \diamond G_j)$.

Each objective ψ_j considers a different reward structure \mathbf{R}_j . The MDP \mathcal{M} , strategy σ , and point $\mathbf{p} \in (\mathbb{R}_{\infty})^{\ell}$ satisfy a multi-objective query $\mathcal{Q} = \langle \psi_1, \dots, \psi_{\ell} \rangle$ (written $\langle \mathcal{M}, \sigma, \mathbf{p} \rangle \models \mathcal{Q}$) iff $\forall j: \langle \mathcal{M}, \sigma, \mathbf{p} \llbracket j \rrbracket \rangle \models \psi_j$. Then, we also say σ *achieves* \mathbf{p} and call \mathbf{p} *achievable*. Let $Ach^{\mathcal{M}}(\mathcal{Q})$ ($Ach_{\text{PS}}^{\mathcal{M}}(\mathcal{Q})$) denote the set of points achieved by a general (pure stationary) strategy. The *closure* of a set $P \subseteq (\mathbb{R}_{\infty})^{\ell}$ with respect to query \mathcal{Q} is $cl^{\mathcal{Q}}(P) = \{\mathbf{p} \in (\mathbb{R}_{\infty})^{\ell} \mid \exists \mathbf{p}' \in P: \forall j: \mathbf{p}' \llbracket j \rrbracket \sim_j \mathbf{p} \llbracket j \rrbracket\}$.

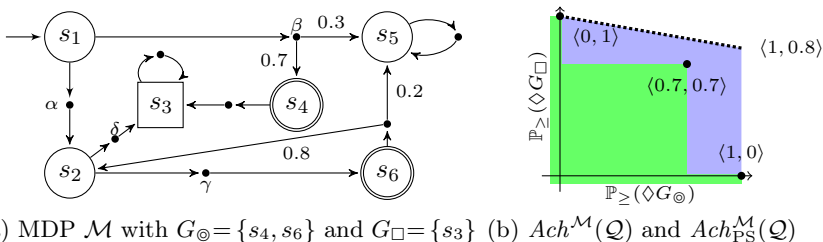


Figure 1: An MDP and a plot of the pure stationary and general Pareto fronts.

For $\mathbf{p}, \mathbf{p}' \in (\mathbb{R}_{\infty})^{\ell}$, we say that \mathbf{p} dominates \mathbf{p}' if $\mathbf{p}' \in cl^{\mathcal{Q}}(\{\mathbf{p}\})$. In this case, $\langle \mathcal{M}, \sigma, \mathbf{p} \rangle \models \mathcal{Q}$ implies $\langle \mathcal{M}, \sigma, \mathbf{p}' \rangle \models \mathcal{Q}$ for any $\sigma \in \Sigma^{\mathcal{M}}$. We are interested in the Pareto front, which is the set of non-dominated achievable points.

Definition 6 (Pareto front). The (general) Pareto front for \mathcal{M} and \mathcal{Q} is $Pareto^{\mathcal{M}}(\mathcal{Q}) = \left\{ \mathbf{p} \in Ach^{\mathcal{M}}(\mathcal{Q}) \mid \forall \mathbf{p}' \in Ach^{\mathcal{M}}(\mathcal{Q}): \mathbf{p} \in cl^{\mathcal{Q}}(\{\mathbf{p}'\}) \implies \mathbf{p} = \mathbf{p}' \right\}$.

The Pareto front is the smallest set $P \subseteq (\mathbb{R}_{\infty})^{\ell}$ with $cl^{\mathcal{Q}}(P) = Ach^{\mathcal{M}}(\mathcal{Q})$. In a similar way, we define the pure stationary Pareto front $Pareto_{PS}^{\mathcal{M}}(\mathcal{Q})$ which only consider points in $Ach_{PS}^{\mathcal{M}}(\mathcal{Q})$.

Example 1. Let \mathcal{M} be the MDP in Fig. 1a and $\mathcal{Q} = \langle \mathbb{P}_{\geq}(\diamond G_{\odot}), \mathbb{P}_{\geq}(\diamond G_{\square}) \rangle$. A pure stationary strategy choosing β at s_1 reaches both, $s_4 \in G_{\odot}$ and $s_3 \in G_{\square}$ with probability 0.7 and thus achieves $\langle 0.7, 0.7 \rangle$. Similarly, $\langle 0, 1 \rangle$ and $\langle 1, 0 \rangle$ are achievable by a pure stationary strategy. Point $\langle 1, 0.8 \rangle$ is achievable by a non-stationary pure strategy that chooses α at s_1 , γ at the first visit of s_2 , and δ in all other cases. Changing this strategy by picking γ only with probability 0.5 achieves $\langle 0.5, 0.9 \rangle$. Fig. 1b illustrates $Pareto_{PS}^{\mathcal{M}}(\mathcal{Q})$ (dots), $Ach_{PS}^{\mathcal{M}}(\mathcal{Q})$ (green area), $Pareto^{\mathcal{M}}(\mathcal{Q})$ (dotted line), and $Ach^{\mathcal{M}}(\mathcal{Q})$ (blue and green area).

3 Deciding Achievability

The achievability problem asks whether a given point is achievable.

GENERAL MULTI-OBJECTIVE ACHIEVABILITY PROBLEM (GMA)
Input: MDP \mathcal{M} , ℓ -dimensional multi-objective query \mathcal{Q} , point $\mathbf{p} \in (\mathbb{R}_{\infty})^{\ell}$
Output: Yes iff $\mathbf{p} \in Ach^{\mathcal{M}}(\mathcal{Q})$ holds

For GMA, the point can be achieved by a general strategy that can potentially make use of memory and randomization. As discussed earlier, this class of strategies is not suitable for various applications. In this work, we focus on a variant of the achievability problem that only considers pure stationary strategies. Sect. 5 also addresses pure strategies that can store more information from the history, e.g., whether a goal state set has been reached already.

<p style="text-align: center; margin: 0;">PURE STATIONARY MULTI-OBJECTIVE ACHIEVABILITY PROBLEM (PSMA)</p> <p>Input: MDP \mathcal{M}, ℓ-dimensional multi-objective query \mathcal{Q}, point $\mathbf{p} \in (\mathbb{R}_\infty)^\ell$</p> <p>Output: Yes iff $\mathbf{p} \in \text{Ach}_{\text{PS}}^{\mathcal{M}}(\mathcal{Q})$ holds</p>
--

3.1 Complexity Results

GMA is PSPACE hard (already with only reachability objectives) [40] and solvable within exponential runtime [19,22]. To the best of our knowledge, a PSPACE upper bound on the complexity of GMA is unknown. This complexity is rooted in the dimension ℓ of the query \mathcal{Q} : for fixed ℓ , the algorithms of [19,22] have polynomial runtime. In contrast, PSMA is NP-complete, even if restricted to 2 objectives.

Lemma 1. *PSMA with only reachability objectives is NP-hard.*

Proof. The result follows by a reduction from the subset sum problem. Given $n \in \mathbb{N}$, $\mathbf{a} \in \mathbb{N}^n$ and $z \in \mathbb{N}$, the subset sum problem is to decide the existence of $\mathbf{v} \in \{0,1\}^n$ such that $\mathbf{v} \cdot \mathbf{a} = z$. This problem is NP-complete [24]. For a given instance of the subset sum problem, we construct the MDP $\mathcal{M}^* = \langle S, \text{Act}, \mathbf{P}, s_I \rangle$ with state space $S = \{s_I, s_1, \dots, s_n, g_1, g_2\}$, actions $\text{Act} = \{\alpha, Y, N\}$, and for all $i \in \{1, \dots, n\}$, $\mathbf{P}(s_I, \alpha, s_i) = \frac{\mathbf{a}[i]}{\mathbf{1} \cdot \mathbf{a}}$ and $\mathbf{P}(s_i, Y, g_1) = \mathbf{P}(s_i, N, g_2) = 1$. States g_1 and g_2 are made absorbing, i.e., $\mathbf{P}(g_1, \alpha, g_1) = \mathbf{P}(g_2, \alpha, g_2) = 1$.

We claim that the PSMA problem for \mathcal{M}^* , $\mathcal{Q} = \langle \mathbb{P}_{\geq}(\diamond \{g_1\}), \mathbb{P}_{\geq}(\diamond \{g_2\}) \rangle$, and $\mathbf{p} = (\frac{z}{\mathbf{1} \cdot \mathbf{a}}, 1 - \frac{z}{\mathbf{1} \cdot \mathbf{a}})$ answers “yes” iff there is a vector \mathbf{v} satisfying the subset sum problem for n , \mathbf{a} and z . Consider the bijection $f: \Sigma_{\text{PS}}^{\mathcal{M}^*} \rightarrow \{0,1\}^n$ with $f(\sigma)[i] = [\sigma(s_i)=Y]$ for all $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}^*}$ and $i \in \{1, \dots, n\}$. We get $\Pr_{\sigma}^{\mathcal{M}^*}(\diamond \{g_1\}) = \sum_{i=1}^n \frac{\mathbf{a}[i]}{\mathbf{1} \cdot \mathbf{a}} [\sigma(s_i)=Y] = \frac{f(\sigma) \cdot \mathbf{a}}{\mathbf{1} \cdot \mathbf{a}}$. Moreover, $\Pr_{\sigma}^{\mathcal{M}^*}(\diamond \{g_2\}) = 1 - \Pr_{\sigma}^{\mathcal{M}^*}(\diamond \{g_1\}) = 1 - \frac{f(\sigma) \cdot \mathbf{a}}{\mathbf{1} \cdot \mathbf{a}}$. It follows that σ achieves \mathbf{p} iff $f(\sigma)$ is a solution to the instance of the subset sum problem. Our construction is inspired by similar ideas from [14,40].

Lemma 2 ([14]). *PSMA with only total reward objectives is NP-hard.*

Theorem 1. *PSMA is NP-complete.*

Proof. Containment follows by guessing a pure stationary strategy and evaluating it on the individual objectives. This can be done in polynomial time [4]. Hardness follows by either Lemma 1 or 2.

Proofs of Lemmas 1 and 2 only consider 2-dimensional multi-objective queries. Hence, in contrast to GMA, the hardness of PSMA is not due to the size of the query.

Corollary 1. *PSMA with only two objectives is NP-complete.*

3.2 A Mixed Integer Linear Programming Approach

An MDP $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$ has exactly $|\Sigma_{\text{PS}}^{\mathcal{M}}| = \prod_{s \in S} |Act(s)|$ many pure stationary strategies. A simple algorithm for PSMA enumerates all $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$ and checks whether $\langle \mathcal{M}, \sigma, \mathbf{p} \rangle \models \mathcal{Q}$ holds. In practice, however, such a brute-force approach is not feasible. For the MDPs that we consider in our experiments in Sect. 6, the number of pure stationary strategies often exceeds 10^{10000} . Instead, our approach is to encode an instance for PSMA as an MILP problem.

MIXED INTEGER LINEAR PROGRAMMING PROBLEM (MILP)	
Input:	$\ell, m, n \in \mathbb{N}, \mathbf{A} \in \mathbb{Q}^{n \times (\ell+m)}, \mathbf{b} \in \mathbb{Q}^n, \mathbf{c} \in \mathbb{Q}^{\ell+m}$
Output:	$\begin{cases} \mathbf{x} \in \arg \max_{\mathbf{x} \in \mathcal{X}} \mathbf{c}^T \mathbf{x} & \text{if } \mathcal{X} \neq \emptyset \\ \text{infeasible} & \text{if } \mathcal{X} = \emptyset \end{cases}$ with $\mathcal{X} = \{\mathbf{x} \in \mathbb{Z}^{\ell} \times \mathbb{R}^m \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$

For an MILP instance as above, each of the n rows of the inequation system $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ represent a *constraint* that is linear over the ℓ integral and m real-valued *variables* given by \mathbf{x} . We call the constraints *feasible* if there is a solution to the inequation system. The task is to decide whether the constraints are feasible and if so, find a solution that maximizes a linear optimization function $\mathbf{c}^T \mathbf{x}$. The optimization function can be omitted if we are only interested in feasibility. MILP is NP-complete [35]. However, tools such as GUROBI [26] and SCIP [25] implement practically efficient algorithms that can solve large instances.

For the rest of this section, let $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$, $\mathcal{Q} = \langle \psi_1, \dots, \psi_{\ell} \rangle$ with $\psi_j = \mathbb{E}_{\sim_j}(\mathbf{R}_j \diamond G_j)$, and $\mathbf{p} \in (\mathbb{R}_{\infty})^{\ell}$ be an instance for PSMA. We provide a translation of the PSMA instance to an instance for MILP that has a feasible solution iff $\mathbf{p} \in \text{Ach}_{\text{PS}}^{\mathcal{M}}(\mathcal{Q})$. The MILP encoding considers integer variables to encode a pure stationary strategy $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$. The other variables and constraints encode the expected reward for each objective on the induced MC \mathcal{M}^{σ} .

3.3 Unichain MDP and Finite Rewards

Restriction 1 (Unichain MDP). *MDP \mathcal{M} has exactly one end component.*

Restriction 2 (Reward Finiteness). *$E_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j) < \infty$ holds for each objective $\psi_j = \mathbb{E}_{\sim_j}(\mathbf{R}_j \diamond G_j)$, state s , and pure stationary strategy σ .*

For simplicity, we first explain our encoding for unichain MDP with finite reward. Sect. 3.5 lifts Restriction 1 and Sect. 3.6 lifts Restriction 2 with more details given in Appendix B. Sect. 3.4 presents an alternative to the encoding of this section, which is smaller but restricted to *total* reward objectives.

Fig. 2 shows the MILP encoding in case Restrictions 1 and 2 hold. We assume $\forall j: \mathbf{p} \llbracket j \rrbracket \neq \infty$ for the point \mathbf{p} since (i) $E_{\sigma}^{\mathcal{M}}(\mathbf{R}_j \diamond G_j) \leq \infty$ holds trivially and (ii) $E_{\sigma}^{\mathcal{M}}(\mathbf{R}_j \diamond G_j) \geq \infty$ will never hold due to Restriction 2. For $j \in \{1, \dots, \ell\}$, let $S_0^j = \{s \in S \mid \forall \sigma \in \Sigma^{\mathcal{M}}: E_{\sigma}^{\mathcal{M}}(\mathbf{R}_j \diamond G_j) = 0\}$ and $S_?^j = \{s \in S \setminus S_0^j \mid s \text{ can be reached from } s_I \text{ without visiting a state in } S_0^j\}$. These sets can be obtained a priori by analyzing the graph structure of \mathcal{M} [4]. Moreover, we consider upper bounds $U_s^j \in \mathbb{Q}$ for the expected reward at state $s \in S_?^j$ such that

$$\begin{array}{l}
\forall s \in S: \qquad \qquad \qquad \triangleright \text{Select an action at each state} \\
\left| \begin{array}{l}
\forall \alpha \in \text{Act}(s): \qquad a_{s,\alpha} \in \{0,1\} \qquad (1) \\
\qquad \qquad \qquad \sum_{\alpha \in \text{Act}(s)} a_{s,\alpha} = 1 \qquad (2)
\end{array} \right. \\
\forall j \in \{1, \dots, \ell\}: \qquad \qquad \qquad \triangleright \text{Compute expected reward values} \\
\left| \begin{array}{l}
\forall s \in S_0^j: \qquad \qquad x_s^j = 0 \qquad (3) \\
\text{If } \psi_j \text{ is maximising, } \pm = + \text{ and } [\text{min}] = 0. \text{ Otherwise, } \pm = - \text{ and } [\text{min}] = 1. \\
\forall s \in S_?^j: \qquad \qquad \pm x_s^j \in [0, U_s^j] \qquad (4) \\
\left| \begin{array}{l}
\forall \alpha \in \text{Act}(s): \qquad \pm x_{s,\alpha}^j \in [0, U_s^j] \qquad (5) \\
\qquad \qquad \qquad x_{s,\alpha}^j \leq \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot (x_{s'}^j \pm \mathbf{R}_j(s, \alpha, s')) \qquad (6) \\
\qquad \qquad \qquad x_{s,\alpha}^j \leq U_s^j \cdot (a_{s,\alpha} - [\text{min}]) \qquad (7) \\
\qquad \qquad \qquad x_s^j \leq \sum_{\alpha \in \text{Act}(s)} x_{s,\alpha}^j + [\text{min}] \cdot (|\text{Act}(s)| - 1) \cdot U_s^j \qquad (8)
\end{array} \right. \\
\left. \right| \qquad \qquad \qquad \pm x_{s_I}^j \sim_j \mathbf{p}[[j]] \qquad \qquad \qquad \triangleright \text{Assert value at initial state} \qquad (9)
\end{array}
\end{array}$$

Figure 2: MILP encoding for unichain MDP and finite rewards.

$U_s^j \geq \max_{\sigma \in \Sigma^{\mathcal{M}}} E_{\sigma}^{\mathcal{M}^s}(\mathbf{R}_j \diamond G_j)$. We compute such upper bounds using single-objective model checking techniques [4,5]. The MILP encoding applies the characterization of expected rewards for MCs as a *linear equation system* [4].

Lemma 3. *For every $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$, the following equation system has a unique solution $\Phi: \{x_s \mid s \in S\} \rightarrow \mathbb{R}^{|\mathcal{S}|}$ satisfying $\Phi(x_s) = E_{\sigma}^{\mathcal{M}^s}(\mathbf{R}_j \diamond G_j)$:*

$$\forall s \in S_0^j: x_s = 0 \qquad \forall s \in S_?^j: x_s = \sum_{s' \in S} \mathbf{P}(s, \sigma(s), s') \cdot (x_{s'} + \mathbf{R}(s, \sigma(s), s'))$$

Proof. Since \mathcal{M} is unichain and we do not collect infinite reward, the only EC of \mathcal{M} (i.e., the only BSCC of \mathcal{M}^{σ} for any σ) either contains a goal state or only contains transitions with zero reward. It follows that $\forall \sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}: \Pr_{\sigma}^{\mathcal{M}}(\diamond S_0^j) = 1$. Lemma 3 follows by standard arguments for MCs with rewards [4, Section 10.5.1].

We discuss the intuition of each constraint in Fig. 2. Let $\Phi: \text{Var} \rightarrow \mathbb{R}$ be an assignment of the occurring variables Var to values. Φ is a solution of the constraints if all (in)equations are satisfied upon replacing all variables v by $\Phi(v)$.

Lines 1 and 2 encode a strategy $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$ by considering a binary variable $a_{s,\alpha}$ for each state s and enabled action α such that $\sigma(s)(\alpha) = 1$ iff $\Phi(a_{s,\alpha}) = 1$ for a solution Φ . Due to Line 2, exactly one action has to be chosen at each state.

Lines 3 to 8 encode for each objective ψ_j the expected rewards obtained for the encoded strategy σ . For every $s \in S$, the variable x_s^j represents a (lower or upper) bound on the expected reward at s . Line 3 sets this value for all $s \in S_0^j$, reflecting the analogous case from Lemma 3. For $s \in S_?^j$, we distinguish maximizing ($\sim_j = \geq$) and minimizing ($\sim_j = \leq$) objectives ψ_j .

For maximizing ψ_j , we have $\Phi(x_s^j) \leq E_{\sigma}^{\mathcal{M}^s}(\mathbf{R}_j \diamond G_j)$ for every solution Φ . This is achieved by considering a variable $x_{s,\alpha}^j$ for each enabled action $\alpha \in \text{Act}(s)$. In

Line 6, we use the equation system characterization from Lemma 3 to assert that the value of $x_{s,\alpha}^j$ can not be greater than the expected reward at s , given that the encoded strategy σ selects α . If σ does not select α (i.e., $\Phi(a_{s,\alpha}) = 0$), Line 7 implies $\Phi(x_{s,\alpha}^j) = 0$. Otherwise, this constraint has no effect. Line 8 ensures that every solution satisfies $\Phi(x_s^j) \leq \Phi(x_{s,\alpha}^j) \leq E_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j)$ for α with $\Phi(a_{s,\alpha}) = 1$.

For minimizing ψ_j , we have $-\Phi(x_s^j) \geq E_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j)$ for every solution Φ , i.e., we consider the negated reward values. The encoding is as for maximizing objectives. However, Line 7 yields $\Phi(x_{s,\alpha}^j) = -U_s^j$ if α is not selected. Thus, in Line 8 we add U_s^j for each of the $(|Act(s)| - 1)$ non-selected actions.

Line 9 and our observations above yield $E_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j) \geq \Phi(x_s^j) \geq \mathbf{p} \llbracket j \rrbracket$ for maximizing and $E_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j) \leq -\Phi(x_s^j) \leq \mathbf{p} \llbracket j \rrbracket$ for minimizing objectives. Therefore, \mathbf{p} is achievable if a solution Φ exists. On the other hand, if \mathbf{p} is achievable by some $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$, the solution Φ exists with $\Phi(a_{s,\alpha}) = \sigma(s)(\alpha)$, $\Phi(x_s^j) = \Phi(x_{s,\alpha}^j) = \pm E_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j)$ if $\alpha = \sigma(s)$, and $\Phi(v) = 0$ for other $v \in Var$.

Theorem 2. *For unichain \mathcal{M} and finite rewards, the constraints in Fig. 2 are feasible iff $\mathbf{p} \in \text{Ach}_{\text{PS}}^{\mathcal{M}}(\mathcal{Q})$.*

Proposition 1. *The MILP encoding above considers $\mathcal{O}(|S| \cdot |Act| \cdot \ell)$ variables.*

3.4 Alternative Encoding for Total Rewards

We now consider PSMA instances where all objectives $\psi_j = \mathbb{E}_{\sim_j}(\mathbf{R}_j \diamond G_j)$ are expected *total* reward objectives, i.e., $G_j = \emptyset$. For such instances, we can employ an encoding from [22] (restated in Lemma 4) for GMA. In fact, we can often translate reachability reward objectives to total reward objectives, e.g., if the set of goal states can not be left or if all objectives consider the same goal states.

Lemma 4 ([22]). *For $S_0 \subseteq S$, let $\Phi: Var \rightarrow \mathbb{R}_{\geq 0}$ be an assignment of variables $Var = \{y_{s,\alpha} \mid s \in S \setminus S_0, \alpha \in Act(s)\}$ and let σ_{Φ} be a stationary strategy satisfying $\sigma_{\Phi}(s)(\alpha) = \Phi(y_{s,\alpha}) / \sum_{\beta \in Act(s)} \Phi(y_{s,\beta})$ for all $s \in S \setminus S_0$ and $\alpha \in Act(s)$ for which the denominator is non-zero. Then, Φ is a solution to the equation system*

$$\begin{aligned} \forall s \in S \setminus S_0: \quad & \sum_{\alpha \in Act(s)} y_{s,\alpha} = [s = s_I] + \sum_{(s', \alpha') \in \text{pre}(s)} \mathbf{P}(s', \alpha', s) \cdot y_{s', \alpha'} \\ & 1 = \sum_{y_{s,\alpha} \in Var} y_{s,\alpha} \cdot \sum_{s' \in S_0} \mathbf{P}(s, \alpha, s') \end{aligned}$$

iff $\text{Pr}_{\sigma_{\Phi}}^{\mathcal{M}}(\diamond S_0) = 1$ and $\forall y_{s,\alpha} \in Var: \Phi(y_{s,\alpha}) = E_{\sigma_{\Phi}}^{\mathcal{M}}(\mathbf{R}_{s,\alpha} \diamond S_0)$ with reward structure $\mathbf{R}_{s,\alpha}$ given by $\mathbf{R}_{s,\alpha}(\hat{s}, \hat{\alpha}, s') = [\hat{s} = s \text{ and } \hat{\alpha} = \alpha]$.

In [22], the lemma is applied to decide achievability of multiple total reward objectives under strategies that are stationary, but not necessarily pure. Intuitively, $E_{\sigma_{\Phi}}^{\mathcal{M}}(\mathbf{R}_{s,\alpha} \diamond S_0)$ coincides with the expected number of times action α is taken at state s until S_0 is reached. Since this value can be infinite if $\text{Pr}_{\sigma_{\Phi}}^{\mathcal{M}}(\diamond S_0) < 1$, a solution Φ can only exist if it induces a strategy that almost surely reaches S_0 .

$$\begin{array}{l}
\forall s \in S_?, \alpha \in Act(s): \quad y_{s,\alpha} \in [0, V_s \cdot a_{s,\alpha}] \quad (10) \\
\left| \quad \sum_{\alpha \in Act(s)} y_{s,\alpha} = [s = s_I] + \sum_{\langle s', \alpha' \rangle \in pre(s)} \mathbf{P}(s', \alpha', s) \cdot y_{s', \alpha'} \quad (11) \right. \\
\quad \quad \quad 1 = \sum_{s \in S_?} \sum_{\alpha \in Act(s)} y_{s,\alpha} \cdot \sum_{s' \in S_0} \mathbf{P}(s, \alpha, s') \quad (12) \\
\left. \forall j \in \{1, \dots, \ell\} : \quad x_{s_I}^j = \sum_{s \in S_?} \sum_{\alpha \in Act(s)} y_{s,\alpha} \cdot \sum_{s' \in S} (\mathbf{P}(s, \alpha, s') \cdot \mathbf{R}_j(s, \alpha, s')) \quad (13) \right. \\
\quad \quad \quad x_{s_I}^j \sim_j \mathbf{p}[[j]] \quad (14)
\end{array}$$

Figure 3: MILP encoding for total reward objectives.

The encoding for unichain MDP with finite rewards and total reward objectives is shown in Fig. 3, where $S_0 = \bigcap_j S_0^j$ and $S_? = S \setminus S_0$. We consider the constraints in conjunction with Lines 1 and 2 from Fig. 2. Let Φ be a solution and let σ be the strategy encoded by such a solution, i.e., $\sigma(s)(\alpha) = \Phi(a_{s,\alpha})$.

Lines 10 to 12 reflect the equations of Lemma 4. Since \mathcal{M} is unichain and we assume finite rewards, there is just one end component in which no reward can be collected. Hence, S_0 is almost surely reached. Line 10 ensures that the strategy in Lemma 4 coincides with the encoded pure strategy σ . We write V_s for an upper bound of the value a solution can possibly assign to $y_{s,\alpha}$, i.e., $\forall \sigma \in \Sigma_{PS}^M: V_s \geq E_\sigma^M(\mathbf{R}_{s,\alpha} \diamond S_0)$. Such an upper bound can be computed based on ideas of [5]. More details are given in Appendix A.

With Lemma 4 we get that $\Phi(y_{s,\sigma(s)})$ is the expected number of times state s is visited under strategy σ . Therefore, in Line 13 we sum up for each state $s \in S_?$ the expected amount of reward collected at s . This yields $\Phi(x_{s_I}^j) = E_\sigma^M(\mathbf{R}_j \diamond G_j)$. Finally, Line 14 asserts that the resulting values exceed the thresholds given by \mathbf{p} .

Theorem 3. *For unichain \mathcal{M} , finite rewards, and total reward objectives, the constraints in Fig. 3 and Lines 1 and 2 of Fig. 2 are feasible iff $\mathbf{p} \in \text{Ach}_{PS}^M(\mathcal{Q})$.*

Proposition 2. *The MILP encoding above considers $\mathcal{O}(|S| \cdot |Act| + \ell)$ variables.*

The encoding for total reward objectives considers fewer variables compared to the encoding of Sect. 3.3 (cf. Proposition 1). In practice, this often leads to faster solving times as we will see in Sect. 6.

3.5 Extension to Multichain MDP

We now lift the restriction to unichain MDP, i.e., we consider multichain MDP with finite rewards. We focus on the encoding of Sect. 3.3. Details for the approach of Sect. 3.4 are in Appendix C. The key challenge is that the equation system in Lemma 3 does not yield a unique solution for multichain MDP.

Example 2. For the multichain MDP in Fig. 5a with $G = \{s_1\}$ we have $S_0 = \{s_1\}$ and $S_? = \{s_0\}$ (the superscript j is omitted as there is only one objective). For σ with $\sigma(s_0) = \alpha$ we get $E_\sigma^M(\mathbf{R} \diamond G) = 0$, but every $\Phi: \{x_{s_0}, x_{s_1}\} \rightarrow \mathbb{R} \times \{0\}$ is a solution for the equation system in Lemma 3.

$$\begin{array}{l}
\forall j \in \{1, \dots, \ell\}, \mathcal{E} \in \text{MECS}(\mathcal{M}[\mathcal{E}_\gamma^j]): \quad \triangleright \text{Detect states with zero reward} \\
\left| \begin{array}{l}
\forall s \in S[\mathcal{E}]: \quad \pm x_s^j \leq U_s^j \cdot (1 - e_s^j) \quad (15) \\
\forall \langle s, \alpha \rangle \in \mathcal{E}: \quad e_{s,\alpha}^j \in \{0, a_{s,\alpha}\} \quad (16) \\
\left| \begin{array}{l}
\forall s' \in \text{succ}(s, \alpha): \quad e_{s,\alpha}^j \leq e_{s'}^j \quad (17) \\
\forall s \in S[\mathcal{E}]: \quad e_s^j = \sum_{\alpha \in \text{Act}(s)} [\langle s, \alpha \rangle \in \mathcal{E}] \cdot e_{s,\alpha}^j \quad (18) \\
\left| \begin{array}{l}
\forall \alpha \in \text{Act}(s): \quad z_{s,\alpha}^j \in [0, V_s \cdot a_{s,\alpha}] \quad (19) \\
z_{s,\perp}^j \in [0, V_s \cdot e_s^j] \quad (20) \\
z_{s,\perp}^j + \sum_{\alpha \in \text{Act}(s)} z_{s,\alpha}^j = \frac{1}{|S[\mathcal{E}]|} + \sum_{\langle s', \alpha' \rangle \in \text{pre}(s) \cap \mathcal{E}} \mathbf{P}(s', \alpha', s) \cdot z_{s',\alpha'}^j \quad (21) \\
1 = \sum_{s \in S[\mathcal{E}]} \left(z_{s,\perp}^{\psi_j} + \sum_{\alpha \in \text{Act}(s)} [\langle s, \alpha \rangle \notin \mathcal{E}] \cdot z_{s,\alpha}^j \right) \quad (22)
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}$$

Figure 4: MILP encoding for detection of end components.

For multichain MDP it can be the case that for some strategy σ the set S_0^j is not reached with probability 1, i.e., there is a positive probability to stay in the set S_γ^j forever. For the induced Markov chain \mathcal{M}^σ , this means that there is a reachable BSCC consisting only of states in S_γ^j . Since BSCCs of \mathcal{M}^σ coincide with end components of \mathcal{M} , we need to inspect the ECs of \mathcal{M} that only consist of S_γ^j -states. These ECs correspond to the ECs of the sub-MDP $\mathcal{M}[\mathcal{E}_\gamma^j]$, where \mathcal{E}_γ^j is the largest subset of $S_\gamma^j \times \text{Act}$ that is closed for \mathcal{M} . For each $\mathcal{E} \in \text{MECS}(\mathcal{M}[\mathcal{E}_\gamma^j])$, we need to detect whether the encoded strategy induces a BSCC $\mathcal{E}' \subseteq \mathcal{E}$.

To cope with multiple ECs, we consider the constraints from Fig. 2 in conjunction with the constraints from Fig. 4. Let Φ be a solution to these constraints and let σ be the encoded strategy σ with $\sigma(s)(\alpha) = \Phi(a_{s,\alpha})$. For each objective ψ_j and state s , a binary variable e_s^j is set to 1 if s lies on a BSCC of the induced MC \mathcal{M}^σ . We only need to consider states $s \in S[\mathcal{E}]$ for $\mathcal{E} \in \text{MECS}(\mathcal{M}[\mathcal{E}_\gamma^j])$.

Line 15 ensures that the value of x_s^j is set to 0 if s lies on a BSCC of \mathcal{M}^σ . Lines 16 to 18 introduce binary variables $e_{s,\alpha}^j$ for each state-action pair in the EC such that any solution Φ satisfies $\Phi(e_{s,\alpha}^j) = 1$ iff $\Phi(e_s^j) = \Phi(a_{s,\alpha}) = 1$. Line 17 yields that $\Phi(e_{s,\alpha}^j) = 1$ implies $\Phi(e_{s'}^j) = 1$ for all successors s' of s and the selected action α . Hence, for all s with $\Phi(e_s^j) = 1$ and for all s' reachable from s in \mathcal{M}^σ , we have $\Phi(e_{s'}^j) = 1$ and $\langle s', \sigma(s') \rangle \in \mathcal{E}$. Therefore, we can only set e_s^j to 1 if there is a BSCC $\mathcal{E}' \subseteq \mathcal{E}$ that either contains s or that is almost surely reached from s without leaving \mathcal{E} . As finite rewards are assumed, \mathcal{E} can not contain a transition with positive reward, yielding $E_\sigma^{\mathcal{M}}(\mathbf{R}_j \diamond G_j) = 0$ if $\Phi(e_s^j) = 1$.

An assignment that sets all variables e_s^j and $e_{s,\alpha}^j$ to 0 trivially satisfies the constraints in Lines 15 to 18. In Lines 19 to 22 we therefore ensure that if a BSCC $\mathcal{E}' \subseteq \mathcal{E}$ exists in \mathcal{M}^σ , $\Phi(e_s^j) = 1$ holds for at least one $s \in S[\mathcal{E}']$. The idea is based on the observation that if a BSCC $\mathcal{E}' \subseteq \mathcal{E}$ exists, there is a state $s \in S[\mathcal{E}']$ that does not reach the set $S \setminus S[\mathcal{E}']$ almost surely. We consider the MDP $\mathcal{M}^\mathcal{E}$, a mild

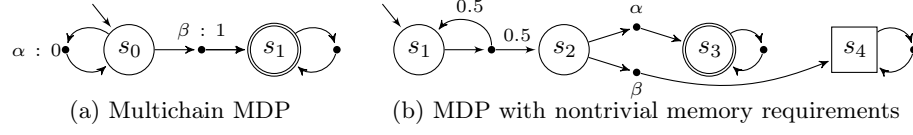


Figure 5: MDPs referenced in Examples 2 and 4.

extension of $\mathcal{M}[\mathcal{E}]$ given by $\mathcal{M}^\mathcal{E} = (S[\mathcal{E}] \uplus \{s_I^\mathcal{E}, s_\perp^\mathcal{E}\}, Act \uplus \{\alpha_I, \perp\}, \mathbf{P}^\mathcal{E}, s_I^\mathcal{E})$, where $\mathbf{P}^\mathcal{E}$ extends $\mathbf{P}[\mathcal{E}]$ such that $\mathbf{P}^\mathcal{E}(s_\perp^\mathcal{E}, \perp, s_\perp^\mathcal{E}) = 1$ and $\forall s \in S[\mathcal{E}]$:

- $\mathbf{P}^\mathcal{E}(s_I^\mathcal{E}, \alpha_I, s) = 1/|S[\mathcal{E}]|$, $\mathbf{P}^\mathcal{E}(s, \perp, s_\perp^\mathcal{E}) = 1$, and
- $\forall \alpha \in \{\hat{\alpha} \in Act(s) \mid \langle s, \hat{\alpha} \rangle \notin \mathcal{E}\} : \mathbf{P}^\mathcal{E}(s, \alpha, s_\perp^\mathcal{E}) = 1$.

Lines 21 and 22 reflect the equation system from Lemma 4 for MDP $\mathcal{M}^\mathcal{E}$ and $S_0 = \{s_\perp\}$. Additionally, Lines 19 and 20 exclude negative solutions and assert $\Phi(z_{s,\alpha}^j) = 0$ if $\Phi(a_{s,\alpha}) = 0$ and $\Phi(z_{s,\perp}^j) = 0$ if $\Phi(e_s^j) = 0$ for any solution Φ . Hence, for states $s \in S[\mathcal{E}]$ where $\Phi(e_s^j) = 0$, the strategy σ encoded by the variables $a_{s,\alpha}$ coincides with the strategy considered in Lemma 4. Assume that solution Φ yields a BSCC within the states of \mathcal{E} in \mathcal{M}^σ and therefore also a BSCC in $(\mathcal{M}^\mathcal{E})^\sigma$. Since $s_\perp^\mathcal{E}$ has to be reached almost surely in $\mathcal{M}^\mathcal{E}$ (cf. Lemma 4), the BSCC has to contain at least one state s with $\Phi(e_s^j) = 1$.

In summary, Lines 19 to 22 imply that every BSCC $\mathcal{E}' \subseteq \mathcal{E}$ of \mathcal{M}^σ contains at least one state s with $\Phi(e_s^j) = 1$. Then, with Lines 16 to 18 we get that $\Phi(e_{s'}^j) = 1$ has to hold for all $s' \in S[\mathcal{E}']$. In \mathcal{M}^σ , the set $S_0^j \cup \{s \mid \Phi(e_s^j) = 1\}$ is therefore reached almost surely and all the states in this set get assigned value 0. In this case, the solution of the equation system from Lemma 3 becomes unique again.

Theorem 4. *For finite rewards, the constraints in Figs. 2 and 4 are feasible iff $p \in Ach_{\text{PS}}^{\mathcal{M}}(\mathcal{Q})$.*

3.6 Extension to Infinite Rewards

Our approach can be modified to allow PSMA instances where infinite expected reward can be collected, i.e., where Restriction 2 does not hold. Infinite reward can be collected if we cycle through an EC of \mathcal{M} that contains a transition with positive reward. Such instances are of practical interest as this often corresponds to strategies that do not accomplish a certain goal (e.g., a robot that stands still and therefore requires infinite time to finish its task).

We sketch the necessary modifications. More details are in Appendix B. Let S_∞ be the set of states where every pure strategy induces infinite reward for at least one minimizing objective. To ensure that the MILP instance has a (real-valued) solution, we consider the sub-MDP of \mathcal{M} obtained by removing S_∞ .

If infinite reward can be collected in an EC, it should not be considered in Fig. 4. We therefore let \mathcal{E} range over maximal ECs that only consist of (a) states in S_0^j and (b) transitions with reward 0.

The upper bounds U_s^j for the maximal expected rewards at each state can not be set to ∞ . However, for the encoding it suffices to compute values that

are sufficiently large. However, we remark that in practice our approach from Appendix B can lead to very large values, yielding numerical instabilities.

For maximizing objectives, we introduce one additional objective which, in a nutshell, checks that the probability to reach a 0-reward BSCC is below 1. If this is the case, there is a positive probability to reach a BSCC in which infinitely many reward can be collected.

4 Computing the Pareto Front

Our next goal is to compute the *pure stationary Pareto front* $Pareto_{PS}^M(\mathcal{Q})$ for MDP \mathcal{M} and multi-objective query \mathcal{Q} . This set can be very large, in particular if the objectives are strongly conflicting with many different tradeoffs. In the worst case, every pure stationary strategy induces a point $\mathbf{p} \in Pareto_{PS}^M(\mathcal{Q})$ (e.g., for $\mathcal{Q} = \langle \mathbb{E}_{\leq}(\mathbf{R} \diamond G), \mathbb{E}_{\geq}(\mathbf{R} \diamond G) \rangle$). We try to find an approximation of $Pareto_{PS}^M(\mathcal{Q})$.

Definition 7. Let $\epsilon \in (\mathbb{R}_{>0})^\ell$. An ϵ -approximation of $P \subseteq (\mathbb{R}_{\infty})^\ell$ is a pair $\langle L, U \rangle$ with $L \subseteq P \subseteq U$ and $\forall \mathbf{p} \in P: \exists \mathbf{p}' \in L \cup ((\mathbb{R}_{\infty})^\ell \setminus U): |\mathbf{p} - \mathbf{p}'| \leq \epsilon$.

PURE STATIONARY PARETO APPROXIMATION PROBLEM (PSP \approx)

Input: MDP \mathcal{M} , ℓ -dimensional multi-objective query \mathcal{Q} , precision $\epsilon \in (\mathbb{R}_{>0})^\ell$ such that $Pareto_{PS}^M(\mathcal{Q}) \subseteq \mathbb{R}^\ell$

Output: An ϵ -approximation of $cl^{\mathcal{Q}}(Pareto_{PS}^M(\mathcal{Q}))$

For simplicity, we only consider inputs that satisfy restriction Restriction 2, i.e., for $\psi_j = \mathbb{E}_{\sim_j}(\mathbf{R}_j \diamond G_j)$ there is $U^j \neq \infty$ such that $\forall \sigma \in \Sigma_{PS}^M: U^j \geq E_{\sigma}^M(\mathbf{R}_j \diamond G_j)$. Ideas of Sect. 3.6 can be used for some other inputs. An all-embracing treatment of infinite rewards, in particular for maximizing ψ_j , is subject to future work.

Our approach for PSP \approx successively divides the solution space into candidate regions. For each region \mathcal{R} (initially, let $\mathcal{R} = [0, U^1] \times \dots \times [0, U^\ell]$), we use the MILP encoding from Sect. 3 with an optimization function to find a point $\mathbf{p} \in \mathcal{R} \cap Pareto_{PS}^M(\mathcal{Q})$ (or find out that no such point exists). The region \mathcal{R} is divided into (i) an achievable region $\mathcal{R}_A \subseteq Ach_{PS}^M(\mathcal{Q})$, (ii) an unachievable region $\mathcal{R}_U \subseteq \mathbb{R}^\ell \setminus Ach_{PS}^M(\mathcal{Q})$, (iii) further candidate regions $\mathcal{R}_1, \dots, \mathcal{R}_n$ that are analyzed subsequently, and (iv) the remaining area $\mathcal{R} \setminus (\mathcal{R}_A \cup \mathcal{R}_U \cup \mathcal{R}_1 \cup \dots \cup \mathcal{R}_n)$ which does not require further analysis as we are only interested in an ϵ -approximation. The procedure stops as soon as no more candidate regions are found.

Example 3. Fig. 6 sketches the approach for an MDP \mathcal{M} and a query \mathcal{Q} with two maximizing objectives. We maintain a set of achievable points (light green) and a set of unachievable points (red). Initially, our candidate region corresponds to $\mathcal{R}_1 = [0, U^1] \times [0, U^2]$ given by the white area in Fig. 6a. We consider the direction vector \mathbf{w}_1 which is orthogonal to the line connecting $\langle U^1, 0 \rangle$ and $\langle 0, U^2 \rangle$. To find some point $\mathbf{p} \in Pareto_{PS}^M(\mathcal{Q}) \cap \mathcal{R}_1$, we solve the MILP resulting from the constraints as in Sect. 3, the constraint $\langle x_{s_l}^1, x_{s_l}^2 \rangle \in \mathcal{R}_1$, and the optimization function $\mathbf{w}_1 \cdot \langle x_{s_l}^1, x_{s_l}^2 \rangle$. Fig. 6b shows the obtained point $\mathbf{p}_1 \in \mathcal{R}_1$. Since \mathbf{p}_1 is

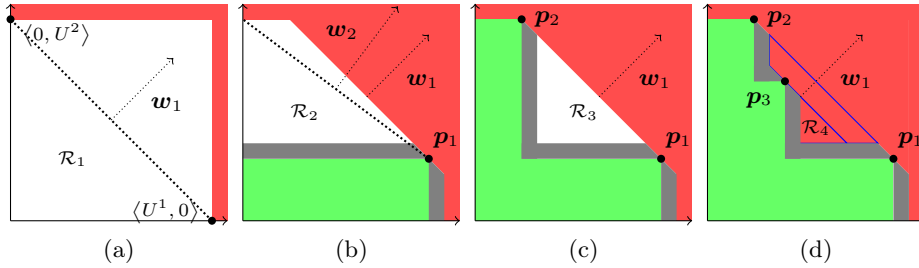


Figure 6: Example exploration of achievable points.

achievable, we know that any point in $cl^{\mathcal{Q}}(\{\mathbf{p}_1\})$ has to be achievable as well. Moreover, the set $\{\mathbf{p} \in \mathcal{R}_1 \mid \mathbf{w}_1 \cdot \mathbf{p} > \mathbf{w}_1 \cdot \mathbf{p}_1\}$ indicated by the area above the diagonal line in Fig. 6b can not contain an achievable point. The gray areas do not have to be checked in order to obtain an ϵ -approximation. We continue with \mathcal{R}_2 indicated by the white area and the direction vector \mathbf{w}_2 , orthogonal to the line connecting $\langle 0, U^2 \rangle$ and \mathbf{p}_1 . As before, we solve an MILP now yielding the point \mathbf{p}_2 in Fig. 6c. We find achievable points $cl^{\mathcal{Q}}(\{\mathbf{p}_2\})$ but no further unachievable points. The next iteration considers candidate region \mathcal{R}_3 and direction vector \mathbf{w}_1 , yielding point \mathbf{p}_3 shown in Fig. 6d. The trapezoidal area is added to the unachievable points whereas $cl^{\mathcal{Q}}(\{\mathbf{p}_3\})$ is achievable. Finally, we check \mathcal{R}_4 for which the corresponding MILP instance is infeasible, i.e., \mathcal{R}_4 is unachievable.

The ideas sketched above can be lifted to $\ell > 2$ objectives. Inspired by [23, Alg. 4], we choose direction vectors that are orthogonal to the convex hull of the achievable points found so far. In fact, for total reward objectives we can apply the approach of [23] to compute the points in $Pareto_{PS}^{\mathcal{M}}(\mathcal{Q}) \cap Pareto^{\mathcal{M}}(\mathcal{Q})$ first and only perform MILP-solving for the remaining regions. As the distance between two found points \mathbf{p}, \mathbf{p}' is at least $\|\mathbf{p} - \mathbf{p}'\| \geq \epsilon$, we can show that our approach terminates after finding at most $\prod_j U^j / \epsilon \llbracket j \rrbracket$ points. Other strategies for choosing direction vectors are possible and can strongly impact performance.

5 Bounded Memory

For GMA, it is necessary and sufficient to consider strategies that require memory exponential in the number of objectives [19,23,40] by storing which goal state set has been reached already. In contrast, restricting to pure (but not necessarily stationary) strategies imposes nontrivial memory requirements that do not only depend on the number of objectives, but also on the point that is to be achieved.

Example 4. Let \mathcal{M} be the MDP in Fig. 5b and $\mathcal{Q} = \langle \mathbb{P}_{\geq}(\diamond G_{\otimes}), \mathbb{P}_{\geq}(\diamond G_{\square}) \rangle$. The point $\mathbf{p}_k = \langle 0.5^k, 1 - 0.5^k \rangle$ for $k \in \mathbb{N}$ is achievable by taking α with probability 0.5^k . \mathbf{p}_k is also achievable with the *pure* strategy σ_k where $\sigma_k(\hat{\pi}) = \alpha$ iff $|\hat{\pi}| \geq k$. σ_k uses k memory states. Pure strategies with fewer memory states do not suffice.

We search for pure strategies with *bounded memory*. For an MDP \mathcal{M} and $K > 0$, let $\Sigma_{\mathbf{P},K}^{\mathcal{M}}$ denote the set of pure K -memory strategies, i.e., any $\sigma \in \Sigma_{\mathbf{P},K}^{\mathcal{M}}$ can be represented by a *Mealy machine* using up to K states (c.f. Appendix D). For a query \mathcal{Q} , let $Ach_{\mathbf{P},K}^{\mathcal{M}}(\mathcal{Q})$ be the set of points achievable by some $\sigma \in \Sigma_{\mathbf{P},K}^{\mathcal{M}}$, and let $Pareto_{\mathbf{P},K}^{\mathcal{M}}(\mathcal{Q})$ be the *pure K -memory Pareto front*.

PURE BOUNDED MULTI-OBJECTIVE ACHIEVABILITY PROBLEM (PBMA)

Input: MDP \mathcal{M} , multi-objective query \mathcal{Q} , memory bound K , point $\mathbf{p} \in (\mathbb{R}_{\infty})^{\ell}$

Output: Yes iff $\mathbf{p} \in Ach_{\mathbf{P},K}^{\mathcal{M}}(\mathcal{Q})$

The pure bounded Pareto approximation problem is defined similarly. We reduce a PBMA instance to an instance for PSMA. The idea is to incorporate a memory structure of size K into \mathcal{M} and then construct a pure stationary strategy in this product MDP (see, e.g., [29] for a similar construction).

Definition 8 (Memory structure). A memory structure of size $K > 0$ is a tuple $\mathcal{N}_K = \langle M, \delta, m_I \rangle$ with $|M| = K$, initial memory state $m_I \in M$, and nondeterministic memory update function $\delta: M \rightarrow 2^M \setminus \emptyset$.

Definition 9 (Memory product). The product of MDP $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$ and memory structure $\mathcal{N}_K = \langle M, \delta, m_I \rangle$ is given by the MDP $\mathcal{M} \otimes \mathcal{N}_K = \langle S \times M, Act \times M, \mathbf{P}', \langle s_I, m_I \rangle \rangle$, where for $s, s' \in S$, $m, m' \in M$, and $\alpha \in Act$: $\mathbf{P}'(\langle s, m \rangle, \langle \alpha, m' \rangle, \langle s', m' \rangle) = \mathbf{P}(s, \alpha, s') \cdot [m' \in \delta(m)]$.

Intuitively, $\delta(m)$ gives the possible successors of memory state $m \in M$. The memory product enriches \mathcal{M} with a new level of nondeterminism corresponding to the choice of the next memory state in \mathcal{N}_K . We set up an equivalence between K -memory strategies for \mathcal{M} and stationary strategies for $\mathcal{M} \otimes \mathcal{N}_K$ by considering *complete* memory structures, i.e., memory structures with $\forall m \in M: \delta(m) = M$.

Lemma 5. Let $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$ be an MDP, $\mathcal{N}_K = \langle M, \delta, m_I \rangle$ be a complete nondeterministic memory structure, and $\mathcal{M} \otimes \mathcal{N}_K = \langle S', Act', \mathbf{P}', s'_I \rangle$ be their product. There are equivalence relations between (i) strategies $\sigma \in \Sigma_{\mathbf{P},K}^{\mathcal{M}}$ and $\sigma' \in \Sigma_{\mathbf{P}'}^{\mathcal{M} \otimes \mathcal{N}_K}$, and (ii) paths compliant with σ and σ' such that equivalence (i) preserves the probability of paths in equivalence relation (ii).

Corollary 2. Let $\mathcal{Q} = \langle \psi_1, \dots, \psi_{\ell} \rangle$ and $\mathcal{Q}' = \langle \psi'_1, \dots, \psi'_{\ell} \rangle$ be a multi-objective queries for \mathcal{M} and $\mathcal{M} \otimes \mathcal{N}_K$ respectively such that $\forall j: \psi'_j = \mathbb{E}_{\sim}(\mathbf{R}' \diamond G')$ iff $\psi_j = \mathbb{E}_{\sim}(\mathbf{R} \diamond G)$, with $G' = G \times M$ if $G \neq \emptyset$ and $G' = \emptyset$ otherwise, and $\mathbf{R}'((s, m), (a, m'), (s', m')) = \mathbf{R}(s, a, s')$ for $s, s' \in S$, $\alpha \in Act$, and $m, m' \in M$. Then, $Ach_{\mathbf{P},K}^{\mathcal{M}}(\mathcal{Q}) = Ach_{\mathbf{P}'}^{\mathcal{M} \otimes \mathcal{N}_K}(\mathcal{Q}')$.

PBMA can thus be decided by solving PSMA for $\mathcal{M} \otimes \mathcal{N}_K$. Proofs and further details are given in Appendix D. The strategies can be further simplified by considering non-full memory structures, e.g., a memory structure that only allows counting.

Table 1: Results for stationary strategies.

Bench- mark	ℓ	Instance 1				$\varepsilon=0.01$		$\varepsilon=0.001$		Instance 2				$\varepsilon=0.01$		$\varepsilon=0.001$	
		Par.	$ S $	$\%E$	\overline{Act}	Time	$ P $	Time	$ P $	Par.	$ S $	$\%E$	\overline{Act}	Time	$ P $	Time	$ P $
dpm	2*	2	1272	32	3.2	17	37	315	377	3	1696	30	3.2	82	30	TO	
eajs	2*	2-3	689	0	1.2	5	23	45	202	3-6	$2 \cdot 10^4$	0	1.2	201	52	3787	375
jobs	3*	3-2	17	0	1.1	3	3	2	3	5-2	117	0	1.5	2042	76	TO	
mutex	3*	1	1795	36	2.2	TO	TO	2	$1 \cdot 10^4$	33	2.3	TO	TO	TO	TO		
polling	2	2-2	233	86	1.5	6	5	23	6	3-2	990	84	1.8	299	5	TO	
rg	2*	2-1-20	2173	14	2.9	5	5	12	5	5-2-50	$3 \cdot 10^4$	5	3.1	496	27	TO	
rover	2*	2500	$2 \cdot 10^4$	0	1.2	110	47	417	251	5000	$4 \cdot 10^4$	0	1.2	258	47	3105	472
serv	2*		$5 \cdot 10^4$	93	1.9	1828	38	TO	TO								
str	2*	30	1426	0	1.3	11	21	822	218	500	$4 \cdot 10^5$	0	1.3	2428	17	TO	
team2	2*	2	1847	24	1.2	2	5	2	5	3	$1 \cdot 10^4$	21	1.2	18	43	MO	
team3	3*	2	1847	24	1.2	165	15	166	15	3	$1 \cdot 10^4$	21	1.2	TO	TO	TO	
uav	2*	750	$2 \cdot 10^5$	29	1.6	400	39	5799	332	1000	$4 \cdot 10^5$	31	1.8	3546	36	TO	
wlan	2*	0	2954	0	1.3	160	16	TO	TO	2	$3 \cdot 10^4$	0	1.3	6728	23	TO	

Remark 1 (Memory patterns). Complete memory structures $\mathcal{N}_K = \langle M, \delta, m_I \rangle$ are in general necessary for PBMA. However, according to the instance, it could be sufficient to consider other nondeterministic memory update functions δ , implying less transitions in $\mathcal{M} \otimes \mathcal{N}_K$ and considerably reducing the number of constraints of the MILP. For example, for the MDP of Fig. 5b, it is sufficient to consider a nondeterministic memory structure such that $\delta(m_i) = \{m_i, m_{i+1}\}$ if $i < K$ and $\delta(m_i) = \{m_i\}$ otherwise, to compute a pure K -memory strategy that achieves $\mathbf{p} = \left(\left(\frac{1}{2}\right)^{K-1}, 1 - \left(\frac{1}{2}\right)^{K-1} \right)$ to satisfy $\mathcal{Q} = \langle \mathbb{P}_{\geq}(G_{\odot}), \mathbb{P}_{\geq}(G_{\square}) \rangle$ (cf. Example 4). Note however that in general, with non-complete memory structure \mathcal{N}_K , we only have $Ach_{\text{PSP}}^{\mathcal{M}}(\mathcal{Q}) \subseteq Ach_{\text{PSP}}^{\mathcal{M} \otimes \mathcal{N}_K}(\mathcal{Q}') \subseteq Ach_{\text{P},K}^{\mathcal{M}}(\mathcal{Q})$.

6 Evaluation

We implemented our approach for $\text{PSP} \approx$ in the model checker STORM [16] using GUROBI [26] as back end for MILP-solving. The implementation takes an MDP (e.g., in PRISM syntax), a multi-objective query, and a precision $\varepsilon > 0$ as input and computes an ε -approximation of the Pareto front. Here, we set $\varepsilon[j] = \varepsilon \cdot \delta_j$, where δ_j is the difference between the maximal and minimal achievable value for objective ψ_j . We also support reward objectives for Markov automata via [38]. The computations within GUROBI might suffer from numerical instabilities. To diminish their impact, we use the *exact* engine of STORM to confirm for each MILP solution that the encoded strategy achieves the encoded point. However, sub-optimal solutions returned by GUROBI may still yield inaccurate results.

We evaluate our approach on 13 multi-objective benchmarks from [23,28,38], each considering one or two parameter instantiations. Application areas range over scheduling (dpm [37], eajs [1], jobs [10], polling [43]), planning (rg [6], rover [28], serv [32], uav [20]), and protocols (mutex [38], str [38], team [15], wlan [31]).

The results for pure stationary strategies are summarized in Table 1. For each benchmark we denote the number of objectives ℓ and whether the alternative encoding from Sect. 3.4 has been applied (*). For each parameter instantiation (Par.), the number of states ($|S|$), the percentage of the states that are contained in an end component ($\%E$), and the average number of available actions at each state (\overline{Act}) are given. For each precision $\varepsilon \in \{0.01, 0.001\}$, we then depict the runtime of STORM and the number of points on the computed approximation of the Pareto front. T0 denotes that the approach did not terminate within 2 hours, M0 denotes insufficient memory (16 GB). All experiments used 8 cores of an Intel[®] Xeon[®] Platinum 8160 Processor.

STORM is often able to compute pure stationary Pareto fronts, even for models with over 100 000 states (e.g., *uav*). However, the model structure strongly affects the performance. For example, the second instance of *jobs* is challenging although it only considers 117 states, a low degree of nondeterminism, and no (non-trivial) end components. Small increments in the model size can increase runtimes significantly (e.g., *dpm* or *uav*). If a higher precision is requested, much more points need to be found, which often leads to timeouts. Similarly, for more than 2 objectives the desired accuracy can often not be achieved within the time limit. The approach can be stopped at any time to report on the current approximation, e.g., after 2 hours STORM found 65 points for Instance 1 of *mutex*.

For almost all benchmarks, the objectives could be transformed to total reward objectives, making the more efficient encoding form Sect. 3.4 applicable. We plot the runtimes of the two encoding in Fig. 7a. The alternative encoding is superior for almost every benchmark. In fact, the original encoding timed out for many models as indicated at the horizontal line at the top of the figure.

In Fig. 7b we plot the Pareto front for the first *polling* instance under general strategies (Gen), pure 2-memory strategies that can change the memory state exactly once (PM_2), pure strategies that observe which goal state set G_j has been visited already (PM_G), and pure stationary strategies (PS). Adding simple memory structures already leads to noticeable improvements in the quality of strategies. In particular, PM_2 strategies perform quite well, and even outperform PM_G strategies (which would be optimal if randomization were allowed).

Data availability. The artifact [17] accompanying this paper contains source code, benchmark files, and replication scripts for our experiments.

7 Conclusion

Finding optimal pure strategies for multi-objective MDPs is NP-hard. Yet, such strategies are often desirable, e.g., when prescribing medication or designing a product. We presented an MILP encoding to find optimal pure and stationary strategies on an MDP in which a memory structure can be incorporated. The encoding is applied to approximate the set of Pareto optimal values. We successfully compute Pareto fronts for several case studies using our implementation in STORM. Despite the hard nature of the problem, our experiments show feasibility of the approach on practical models with tens of thousands of states.

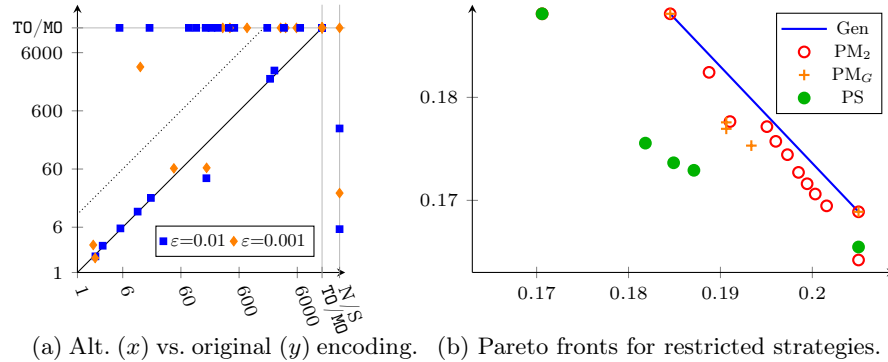


Figure 7: Comparison of the two encodings (left) and impact of memory (right).

Acknowledgments. The authors thank Sebastian Junges for his valuable contributions during early stages of this work.

References

1. Baier, C., Daum, M., Dubsloff, C., Klein, J., Klüppelholz, S.: Energy-utility quantiles. In: NASA Formal Methods, NFM. pp. 285–299 (2014). https://doi.org/10.1007/978-3-319-06200-6_24
2. Baier, C., Dubsloff, C., Klüppelholz, S.: Trade-off analysis meets probabilistic model checking. In: CSL-LICS. pp. 1:1–1:10. ACM (2014)
3. Baier, C., Hermanns, H., Katoen, J.: The 10, 000 facets of MDP model checking. In: Computing and Software Science, LNCS, vol. 10000, pp. 420–451. Springer (2019)
4. Baier, C., Katoen, J.P.: Principles of model checking. MIT Press (2008)
5. Baier, C., Klein, J., Leuschner, L., Parker, D., Wunderlich, S.: Ensuring the reliability of your model checker: Interval iteration for Markov decision processes. In: CAV (1). LNCS, vol. 10426, pp. 160–180. Springer (2017)
6. Barrett, L., Narayanan, S.: Learning all optimal policies with multiple criteria. In: (ICML). pp. 41–47 (2008)
7. Benini, L., Bogliolo, A., Paleologo, G.A., De Micheli, G.: Policy optimization for dynamic power management. Trans. Comp.-Aided Des. Integ. Cir. Sys. **18**(6), 813–833 (2006). <https://doi.org/10.1109/43.766730>
8. Berthon, R., Randour, M., Raskin, J.: Threshold constraints with guarantees for parity objectives in Markov decision processes. In: ICALP. LIPIcs, vol. 80, pp. 121:1–121:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2017)
9. Bouyer, P., González, M., Markey, N., Randour, M.: Multi-weighted Markov decision processes with reachability objectives. In: GandALF. EPTCS, vol. 277, pp. 250–264 (2018)
10. Bruno, J.L., Downey, P.J., Frederickson, G.N.: Sequencing tasks with exponential service times to minimize the expected flow time or makespan. J. ACM **28**(1), 100–113 (1981). <https://doi.org/10.1145/322234.322242>
11. Bruyère, V., Filiot, E., Randour, M., Raskin, J.: Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games. Inf. Comput. **254**, 259–295 (2017)

12. Chatterjee, K., de Alfaro, L., Henzinger, T.A.: Trading memory for randomness. In: QEST. pp. 206–217. IEEE Computer Society (2004)
13. Chatterjee, K., Kretínská, Z., Kretínský, J.: Unifying two views on multiple mean-payoff objectives in markov decision processes. LMCS **13**(2) (2017)
14. Chatterjee, K., Majumdar, R., Henzinger, T.A.: Markov decision processes with multiple objectives. In: STACS. LNCS, vol. 3884, pp. 325–336. Springer (2006)
15. Chen, T., Kwiatkowska, M.Z., Parker, D., Simaitis, A.: Verifying team formation protocols with probabilistic model checking. In: CLIMA. pp. 190–207 (2011)
16. Dehnert, C., Junges, S., Katoen, J.P., Volk, M.: A Storm is coming: A modern probabilistic model checker. In: CAV. LNCS, vol. 10427. Springer (2017)
17. Delgrange, F., Katoen, J.P., Quatmann, T., Randour, M.: Evaluated artifact for this paper. figshare (2020). <https://doi.org/10.6084/m9.figshare.11569485>
18. von Essen, C., Giannakopoulou, D.: Probabilistic verification and synthesis of the next generation airborne collision avoidance system. STTT **18**(2), 227–243 (2016)
19. Etessami, K., Kwiatkowska, M.Z., Vardi, M.Y., Yannakakis, M.: Multi-objective model checking of Markov decision processes. Logical Methods in Computer Science **4**(4) (2008). [https://doi.org/10.2168/LMCS-4\(4:8\)2008](https://doi.org/10.2168/LMCS-4(4:8)2008)
20. Feng, L., Wiltsche, C., Humphrey, L.R., Topcu, U.: Controller synthesis for autonomous systems interacting with human operators. In: ICCPS. pp. 70–79. ACM (2015)
21. Forejt, V., Kwiatkowska, M.Z., Norman, G., Parker, D.: Automated verification techniques for probabilistic systems. In: SFM. LNCS, vol. 6659, pp. 53–113. Springer (2011)
22. Forejt, V., Kwiatkowska, M.Z., Norman, G., Parker, D., Qu, H.: Quantitative multi-objective verification for probabilistic systems. In: TACAS. LNCS, vol. 6605, pp. 112–127. Springer (2011)
23. Forejt, V., Kwiatkowska, M.Z., Parker, D.: Pareto curves for probabilistic model checking. In: ATVA. LNCS, vol. 7561, pp. 317–332. Springer (2012)
24. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1979)
25. Gleixner, A., Bastubbe, M., Eifler, L., Gally, T., Gamrath, G., Gottwald, R.L., Hendel, G., Hojny, C., Koch, T., Lübbecke, M.E., Maher, S.J., Miltenberger, M., Müller, B., Pfetsch, M.E., Puchert, C., Rehfeldt, D., Schlösser, F., Schubert, C., Serrano, F., Shinano, Y., Viernickel, J.M., Walter, M., Wegscheider, F., Witt, J.T., Witzig, J.: The SCIP Optimization Suite 6.0. Technical report, Optimization Online (July 2018), http://www.optimization-online.org/DB_HTML/2018/07/6692.html
26. Gurobi Optimization, L.: Gurobi optimizer reference manual (2019), <http://www.gurobi.com>
27. Haddad, S., Monmege, B.: Reachability in MDPs: Refining convergence of value iteration. In: RP. LNCS, vol. 8762, pp. 125–137. Springer (2014)
28. Hartmanns, A., Junges, S., Katoen, J., Quatmann, T.: Multi-cost bounded reachability in MDP. In: TACAS (2). LNCS, vol. 10806, pp. 320–339. Springer (2018)
29. Junges, S., Jansen, N., Wimmer, R., Quatmann, T., Winterer, L., Katoen, J., Becker, B.: Finite-state controllers of POMDPs using parameter synthesis. In: UAI. pp. 519–529. AUAI Press (2018)
30. 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)

31. Kwiatkowska, M.Z., Norman, G., Parker, D.: The PRISM benchmark suite. In: QEST. pp. 203–204 (2012). <https://doi.org/10.1109/QEST.2012.14>
32. Lacerda, B., Parker, D., Hawes, N.: Multi-objective policy generation for mobile robots under probabilistic time-bounded guarantees. In: ICAPS. pp. 504–512. AAAI Press (2017)
33. Lizotte, D.J., Bowling, M., Murphy, S.A.: Linear fitted-Q iteration with multiple reward functions. *J. Mach. Learn. Res.* **13**, 3253–3295 (2012)
34. Perny, P., Weng, P.: On finding compromise solutions in multiobjective Markov decision processes. In: ECAI. FAIA, vol. 215, pp. 969–970. IOS Press (2010)
35. Pia, A.D., Dey, S.S., Molinaro, M.: Mixed-integer quadratic programming is in NP. *Math. Program.* **162**(1-2), 225–240 (2017)
36. Puterman, M.L.: *Markov Decision Processes*. John Wiley and Sons (1994)
37. Qiu, Q., Wu, Q., Pedram, M.: Stochastic modeling of a power-managed system: Construction and optimization. In: ISLPED. pp. 194–199. ACM (1999)
38. Quatmann, T., Junges, S., Katoen, J.: Markov automata with multiple objectives. In: CAV (1). LNCS, vol. 10426, pp. 140–159. Springer (2017)
39. Randour, M., Raskin, J., Sankur, O.: Variations on the stochastic shortest path problem. In: VMCAI. Lecture Notes in Computer Science, vol. 8931, pp. 1–18. Springer (2015)
40. Randour, M., Raskin, J., Sankur, O.: Percentile queries in multi-dimensional Markov decision processes. *FMSD* **50**(2-3), 207–248 (2017)
41. Roijers, D.M., Vamplew, P., Whiteson, S., Dazeley, R.: A survey of multi-objective sequential decision-making. *JAIR* **48**, 67–113 (2013)
42. Scheftelowitsch, D., Buchholz, P., Hashemi, V., Hermanns, H.: Multi-objective approaches to Markov decision processes with uncertain transition parameters. In: VALUETOOLS. pp. 44–51. ACM (2017)
43. Srinivasan, M.: Nondeterministic polling systems. *Management Science* **37**(6), 667–681 (1991). <https://doi.org/10.1287/mnsc.37.6.667>
44. Wiering, M.A., de Jong, E.D.: Computing optimal stationary policies for multi-objective Markov decision processes. In: ADPRL. pp. 158–165 (2007). <https://doi.org/10.1109/ADPRL.2007.368183>

A Upper Bounds for Expected Number of Visits

Let $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$ be an MDP and $\mathcal{Q} = \langle \psi_1, \dots, \psi_\ell \rangle$ be a multi-objective query over objectives $\psi_j = \mathbb{E}_{\sim_j}(\mathbf{R}_j \diamond G_j)$. We use notations as defined in Sect. 3.

For the encodings in Figs. 3 and 4 we have to compute values V_s for each s such that the expected number of times a path visits s from the initial state is at most V_s . For this, we only consider pure positional scheduler that reach a given set of sink states S_0 almost surely. Formally, we thus require

$$\forall \sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}: V_s \geq [\Pr_{\sigma}^{\mathcal{M}}(\diamond S_0) = 1] \cdot \mathbf{E}_{\sigma}^{\mathcal{M}}(\mathbf{R}_{s,\alpha} \diamond S_0).$$

For the case where S_0 is almost surely reached under any scheduler, [5] provide an efficient, graph based algorithm to compute these values. Since we have to deal with end components (in particular for the encoding in Sect. 3.5, we can not apply the approach of [5] directly.

The idea is to eliminate the end components as in, e.g., [27], but in a way that expected visiting times $\mathbf{E}_{\sigma}^{\mathcal{M}}(\mathbf{R}_{s,\alpha} \diamond S_0)$ are over-approximated. For each maximal EC $\mathcal{E} \in \text{MECS}(\mathcal{M}[\![\mathcal{E}_i^j]\!]])$ and for each $s \in S[\![\mathcal{E}]\!]]$ within this EC we perform the following steps:

1. Compute a lower bound $p > 0$ for the probability that starting in s , we leave the EC without visiting s again. For this lower bound, all pure stationary schedulers σ with $\Pr_{\sigma}^{\mathcal{M}_s}(\diamond S \setminus S[\![\mathcal{E}]\!]])$ have to be considered. To obtain such a lower bound, we can provide a lower bound on the probability of some finite path that leaves the EC. Since there has to be such a path that visits each $s' \in S[\![\mathcal{E}]\!]]$ at most once, we compute p as follows:

$$p = \prod_{s' \in S[\![\mathcal{E}]\!]]} \min_{\alpha \in Act(s')} \min_{s'' \in \text{supp}(\langle s', \alpha \rangle)} \mathbf{P}(s', \alpha, s'')$$

2. Set all transition probabilities $\mathbf{P}(s, \alpha, s')$ with $\langle s, \alpha \rangle \in \mathcal{E}$ to 0.
3. For each $s' \in S[\![\mathcal{E}]\!]]$ and $\alpha \in Act(s')$ such that $\langle s', \alpha \rangle \notin \mathcal{E}$, add a fresh action α' and set for each $s'' \in S$:

$$\mathbf{P}(s, \alpha', s'') = p \cdot \mathbf{P}(s', \alpha, s'') + (1 - p) \cdot [s = s''].$$

With these steps, we have eliminated all end components consisting of states in $S_{\mathcal{E}} = S \setminus S_0$. However, each state s within an end component gets an additional self loop probability of $(1 - p)$, which is an upper bound on the probability that we cycle through the end component and visit s again. Therefore, performing the approach of [5] yields the desired values.

We remark that computing the values V_s as above can lead to very large values which affect the numerical stability of the MILP solving. However, for the models in our experiments in Sect. 6, this was not a concern.

B Infinite Rewards

We now consider PSMA instances where infinite expected reward can be collected, i.e., for state s , objective ψ_j and $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$ we potentially have $\mathbf{E}_{\sigma}^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j) =$

∞ . We treat such instances by a combination of preprocessing steps and (slight) modifications of the constraints in Figs. 2 and 4. See Appendix C for the encoding of Sect. 3.4. As before, let

$$S_\infty = \{s \in S \mid \forall \sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}: \exists j: \psi_j \text{ is minimizing and } E_\sigma^{\mathcal{M}_s}(\mathbf{R}_j \diamond G_j) = \infty\}$$

be the set of states for which all schedulers induce infinite reward with respect to at least one minimizing objective. We can determine S_∞ by first finding the end components of \mathcal{M} in which no reward for any minimizing objective is collected. S_∞ corresponds to the set of states that can not reach such an end component.

We can show for all $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$ that $\text{Pr}_\sigma^{\mathcal{M}}(\diamond S_\infty) > 0$ implies $E_\sigma^{\mathcal{M}}(\mathbf{R}_j \diamond G_j) = \infty$ for at least one minimizing objective ψ_j . Hence, such a scheduler does not achieve the given point \mathbf{p} . We assume $s_I \notin S_\infty$ (otherwise $\text{Ach}_{\text{PS}}^{\mathcal{M}}(\mathcal{Q}) = \emptyset$). To exclude schedulers $\sigma \in \Sigma_{\text{PS}}^{\mathcal{M}}$ with $\text{Pr}_\sigma^{\mathcal{M}}(\diamond S_\infty) > 0$, we consider the sub-MDP $\mathcal{M}[\mathcal{E}_{\text{fin}}, s_I]$ instead of \mathcal{M} , where \mathcal{E}_{fin} is the largest subset of $(S \setminus S_\infty) \times \text{Act}$ that is closed for \mathcal{M} . The achievable points for $\mathcal{M}[\mathcal{E}_{\text{fin}}, s_I]$ coincide with the achievable points for \mathcal{M} . Moreover, there is a scheduler for $\mathcal{M}[\mathcal{E}_{\text{fin}}, s_I]$ that for each minimizing objective induces a finite expected reward at every state. This is a requirement for the existence of a (real-valued) solution of the constraints in Fig. 2.

$\mathcal{M}[\mathcal{E}_{\text{fin}}, s_I]$ can still contain ECs in which infinite reward is collected for either a minimizing or a maximizing objective. However, the constraints in Fig. 4 should only apply to ECs without any rewards. Hence, the constraints are considered for every maximal EC $\mathcal{E} \in \text{MECS}(\mathcal{M}[\mathcal{E}_\cap])$, where \mathcal{E}_\cap is the largest subset of $\mathcal{E}_{\text{fin}} \cap \mathcal{E}_?^j \cap \{(s, \alpha) \mid \forall s': \mathbf{R}_j(s, \alpha, s') = 0\}$ that is closed for \mathcal{M} .

Sect. 3.3 considers upper bounds $U_s^j \in \mathbb{Q}$ for the maximal expected rewards at state s with respect to ψ_j . We consider the case where this value is infinite by computing sufficiently large bounds as follows: Let \mathcal{M}' be the MDP obtained by eliminating the end components in which a positive reward can be collected using a very similar construction as in Appendix A. Note that we can not collect infinite reward in \mathcal{M}' . We can show that the expected rewards for \mathcal{M}' are an upper bound for the expected rewards for \mathcal{M} , assuming that only strategies yielding finite rewards are considered.

For maximizing objectives ψ_j , we also have to allow strategies that do collect infinite reward. We therefore add additional constraints that detect if infinite reward is collected. The idea is to compute the probability that a state in $S_0^j \cup \{s \mid \Phi(e_s) = 1\}$ is reached. If this probability is below 1, infinite reward is collected. An additional binary variable b^j is added, to ensure that either infinite reward is collected or the threshold given by $\mathbf{p}[j]$ is satisfied. The additional constraints are shown in Fig. 8. Observe that strict inequalities as in Line 25 are not allowed in MILP encodings. However, we can replace constraints of the form $a < b$ by $a + \epsilon \leq b$ and ask for a solution that maximizes ϵ .

Theorem 5. *With the modifications as above, the constraints in Figs. 2 and 4 applied to $\mathcal{M}[\mathcal{E}_{\text{fin}}, s_I]$ are feasible iff $\mathbf{p} \in \text{Ach}_{\text{PS}}^{\mathcal{M}}(\mathcal{Q})$.*

$$\begin{array}{l}
\forall \text{ maximizing } \psi_j \text{ with infinite reward possible:} \\
\left| \begin{array}{l}
b^j \in \{0, 1\} \\
x_{s_I}^j \geq \mathbf{p}[[j]] \cdot b^j \\
w_{s_I}^j < 1 + b^j \\
\forall s \in S_0^j: w_s^j = 1 \\
\forall s \in S_7^j: \\
\left| \begin{array}{l}
\forall \alpha \in Act(s): w_{s,\alpha}^j \geq 1 - a_{s,\alpha} \\
w_{s,\alpha}^j \geq e_{s,\alpha}^j \\
w_{s,\alpha}^j \geq \sum_{s' \in S_7^j} \mathbf{P}(s, \alpha, s') \cdot w_{s'}^j \\
w_s^j = \sum_{\alpha \in Act(s)} w_{s,\alpha}^j - (|Act(s)| - 1)
\end{array} \right.
\end{array} \right.
\end{array}
\tag{23}
\tag{24}
\tag{25}
\tag{26}
\tag{27}
\tag{28}
\tag{29}
\tag{30}$$

Figure 8: MILP encoding for maximizing objectives with possibly infinite rewards.

C Extensions for Alternative Encoding

The alternative encoding can be lifted to multichain MDP as well. For this, we use the constraints from Fig. 4 without Line 15 in conjunction with the constraints in Fig. 9. The latter is a slight extension of the encoding from Fig. 3. It considers additional variables $y_{s,\perp}$ which can only be non-zero, if s lies on an EC. This idea is similar to the variables $z_{s,\alpha}^j$ in Fig. 4.

After performing the preprocessing steps from Appendix B, the encoding also supports infinite rewards for minimizing objectives. In incorporation of infinite rewards for maximizing objectives is left for future work.

D Details for Bounded Memory Achievability

D.1 Pure strategies encoded by Mealy machines

A pure strategy σ for $\mathcal{M} = \langle S, Act, \mathbf{P}, s_I \rangle$ can be encoded by a *Mealy machine* $\langle M, \sigma_a, \sigma_u, m_I \rangle$ where M is a finite set of memory states, $m_I \in M$ the *initial memory state*, σ_a the *next action function* $\sigma_a: S \times M \rightarrow Act$ where $\sigma_a(s, m) \in Act(s)$ for any $s \in S$ and $m \in M$, and σ_u the *memory update function* $\sigma_u: S \times Act \rightarrow M$. A strategy is *K-memory* if $|M| = K$.

The strategy σ induces an MC \mathcal{M}^σ defined on the state space $S \times M$ with initial state (s_I, m_I) such that, for any pair of states (s, m) and (s', m') , the probability of transition (s, m) to (s', m') when choosing action α is equal to $\mathbf{P}(s, \alpha, s') \cdot [\alpha = \sigma_a(s, m)] \cdot [m' = \sigma_u(m, s, \alpha)]$.

Remark 2. Let $K' \leq K$, from every $\sigma' \in \Sigma_{\mathbf{P}, K'}^{\mathcal{M}}$, we can trivially construct a strategy $\sigma \in \Sigma_{\mathbf{P}, K}^{\mathcal{M}}$ with $K - K'$ unused memory states. Thus, $Ach_{\mathbf{P}, K'}^{\mathcal{M}}(\mathcal{Q}) \subseteq Ach_{\mathbf{P}, K}^{\mathcal{M}}(\mathcal{Q})$. Therefore, to reason about points achieved by pure strategies with memory of maximal size K , it is sufficient to only consider pure K -memory strategies.

$$\begin{array}{l}
\forall s \in S: \quad \triangleright \text{Select an action at each state} \\
\left| \begin{array}{l}
\forall \alpha \in \text{Act}(s) \quad a_{s,\alpha} \in \{0,1\} \quad (31) \\
\sum_{\alpha \in \text{Act}(s)} a_{s,\alpha} = 1 \quad (32)
\end{array} \right. \\
\forall s \in S?: \quad y_{s,\perp} \in [0, V_s] \quad (33) \\
\left| \begin{array}{l}
y_{s,\perp} \leq [\exists \mathcal{E} \in \text{MECS}(\mathcal{M}[\mathcal{E}_?]): s \in S[\mathcal{E}]] \cdot e_s \quad (34) \\
\forall \alpha \in \text{Act}(s): \quad y_{s,\alpha} \in [0, V_s \cdot a_{s,\alpha}] \quad (35) \\
y_{s,\perp} + \sum_{\alpha \in \text{Act}(s)} y_{s,\alpha} = [s = s_I] + \sum_{\langle s', \alpha' \rangle \in \text{pre}(s)} \mathbf{P}(s', \alpha', s) \cdot y_{s', \alpha'} \quad (36) \\
1 = \sum_{s \in S?} \left(y_{s,\perp} + \sum_{\alpha \in \text{Act}(s)} y_{s,\alpha} \cdot \sum_{s' \in S_0} \mathbf{P}(s, \alpha, s') \right) \quad (37)
\end{array} \right. \\
\forall j \in \{1, \dots, \ell\}: \\
\left| \begin{array}{l}
x_{s_I}^j = \sum_{s \in S?} \sum_{\alpha \in \text{Act}(s)} y_{s,\alpha} \cdot \sum_{s' \in S} (\mathbf{P}(s, \alpha, s') \cdot \mathbf{R}_j(s, \alpha, s')) \quad (38) \\
x_{s_I}^j \sim_j \mathbf{p}[[j]] \quad (39)
\end{array} \right.
\end{array}$$

Figure 9: MILP encoding for total reward objectives on multichain MDP.

D.2 Proof of Lemma 5

We left out the proof of Lemma 5 from the main text: we present it here. Recall the statement:

Lemma 5. *Let $\mathcal{M} = \langle S, \text{Act}, \mathbf{P}, s_I \rangle$ be an MDP, $\mathcal{N}_K = \langle M, \delta, m_I \rangle$ be a complete nondeterministic memory structure, and $\mathcal{M} \otimes \mathcal{N}_K = \langle S', \text{Act}', \mathbf{P}', s'_I \rangle$ be their product. There are equivalence relations between (i) strategies $\sigma \in \Sigma_{\text{P},K}^{\mathcal{M}}$ and $\sigma' \in \Sigma_{\text{PS}}^{\mathcal{M} \otimes \mathcal{N}_K}$, and (ii) paths compliant with σ and σ' such that equivalence (i) preserves the probability of paths in equivalence relation (ii).*

Proof. Recall that $S' = S \times M$, $\text{Act}' = \text{Act} \times M$, and $s'_I = (s_I, m_I)$. First, from a strategy $\sigma \in \Sigma_{\text{P},K}^{\mathcal{M}}$ encoded as $\langle M', \sigma_a, \sigma_u, m'_I \rangle$, we can construct a strategy $\sigma' \in \Sigma_{\text{PS}}^{\mathcal{M} \otimes \mathcal{N}_K}$ as follows. Since σ is K -memory, we link each memory state of σ to a memory state of \mathcal{N}_K . So, consider w.l.o.g. that $M' = M$ and $m'_I = m_I$. For all $(s, m) \in S \times M$, let $\alpha = \sigma_a(s, m)$ and $m' = \sigma_u(m, s, \alpha)$, we define $\sigma'(s, m) = (\alpha, m')$. Note that σ' is well defined using completeness of \mathcal{N}_K : if $\alpha \in \text{Act}(s)$, then $(\alpha, m') \in \text{Act}'(s, m)$ since $\delta(m) = M$.

Second, we can also construct a strategy $\sigma = \langle M', \sigma_a, \sigma_u, m'_I \rangle$ from any strategy $\sigma' \in \Sigma_{\text{PS}}^{\mathcal{M} \otimes \mathcal{N}_K}$ as follows. Let $M' = M$, and $m'_I = m_I$, for all $s \in S$ and $m \in M$ such that $\sigma'(s, m) = (\alpha, m')$, we define $\sigma_a(s, m) = \alpha$, $\sigma_u(m, s, \alpha) = m'$, and $\sigma_u(m, s, \alpha') = m$ for $\alpha' \neq \alpha$.

These constructions form an equivalence between pure K -memory strategies for \mathcal{M} and pure stationary strategies for $\mathcal{M} \otimes \mathcal{N}_K$. Let $\sigma \in \Sigma_{\text{P},K}^{\mathcal{M}}$ and $\sigma' \in \Sigma_{\text{PS}}^{\mathcal{M} \otimes \mathcal{N}_K}$ be two such equivalent strategies, and let \mathcal{M}^σ , $(\mathcal{M} \otimes \mathcal{N}_K)^{\sigma'}$ be the MCs induced respectively by σ and σ' . Both MCs have the same state space defined on $S \times M$ and initial state (s_I, m_I) . Moreover, let \mathbf{P}^σ be the transition

function of \mathcal{M}^σ and $\mathbf{P}^{\sigma'}$ that of $(\mathcal{M} \otimes \mathcal{N}_K)^{\sigma'}$. We have $\mathbf{P}^\sigma((s, m), \alpha, (s', m')) = \mathbf{P}^{\sigma'}((s, m), (\alpha, m'), (s', m'))$ for $s, s' \in S$, $\alpha \in Act$, and $m, m' \in M$. Modulo a bijection consisting in a renaming of actions, both MCs are thus identical. Consequently, the probability of paths and events in \mathcal{M}^σ and $(\mathcal{M} \otimes \mathcal{N}_K)^{\sigma'}$ are equal.

D.3 Pure Bounded-memory Pareto Approximation Problem

PURE BOUNDED-MEMORY PARETO APPROXIMATION PROBLEM (PBP \approx)
Input: MDP \mathcal{M} , ℓ -dimensional multi-objective query \mathcal{Q} , memory bound $K \in \mathbb{N}$, precision $\epsilon > \mathbb{R}_{>0}^\ell$.
Output: An ϵ -approximation of $cl^\mathcal{Q}(Pareto_{\mathbb{P}, K}^{\mathcal{M}}(\mathcal{Q}))$.

We already have all the ingredients to solve this problem: let \mathcal{M} be an MDP and \mathcal{N}_K be a complete nondeterministic memory structure, recall Corollary 2:

Corollary 2. *Let $\mathcal{Q} = \langle \psi_1, \dots, \psi_\ell \rangle$ and $\mathcal{Q}' = \langle \psi'_1, \dots, \psi'_\ell \rangle$ be a multi-objective queries for \mathcal{M} and $\mathcal{M} \otimes \mathcal{N}_K$ respectively such that $\forall j: \psi'_j = \mathbb{E}_\sim(\mathbf{R}' \diamond G')$ iff $\psi_j = \mathbb{E}_\sim(\mathbf{R} \diamond G)$, with $G' = G \times M$ if $G \neq \emptyset$ and $G' = \emptyset$ otherwise, and $\mathbf{R}'((s, m), (a, m'), (s', m')) = \mathbf{R}(s, a, s')$ for $s, s' \in S$, $\alpha \in Act$, and $m, m' \in M$. Then, $Ach_{\mathbb{P}, K}^{\mathcal{M}}(\mathcal{Q}) = Ach_{\mathbb{P}, K}^{\mathcal{M} \otimes \mathcal{N}_K}(\mathcal{Q}')$.*

We can establish a reduction from PBP \approx to PSP \approx : for inputs \mathcal{M} , \mathcal{Q} , and $K \in \mathbb{N}$, we solve PBP \approx by computing the solution of the PSP \approx problem for $\mathcal{M} \otimes \mathcal{N}_K$ and \mathcal{Q}' with the approach of Section 4.

E Evaluation with GLPK

We repeated our experiments from Sect. 6 using the free MILP solver GLPK instead of GUROBI. Since GLPK does not have native support for multi-threaded MILP solving, the benchmarks were run on a single core. Apart from that, we have the same setup as described in Sect. 6.

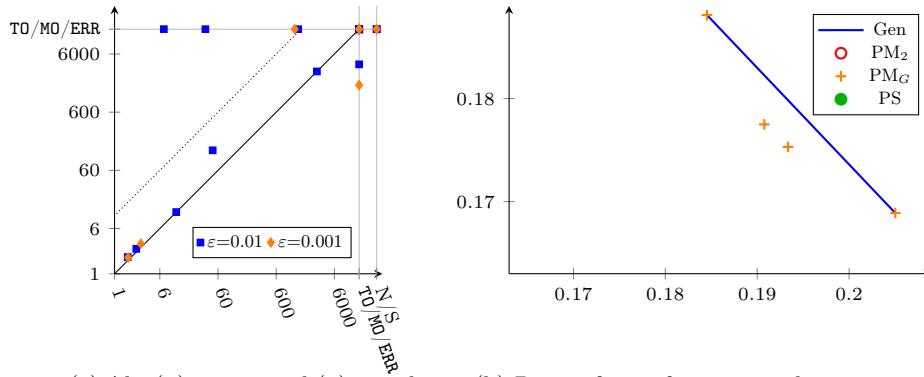
Table 2 shows the results for all benchmarks under positional stationary strategies using the alternative encoding whenever possible (cf. Table 1). For the second instance of `jobs` we observed an internal GLPK error. We conjecture that this is a consequence of numerical inaccuracies.

Fig. 10a compares the runtimes of both encodings (cf. Fig. 7a). Finally, Fig. 10b shows the resulting Pareto fronts for different kinds strategy classes (cf. Fig. 7b). Note that we have not been able to obtain values for PM₂ and PS within reasonable time.

Remark 3. When comparing the number of found Pareto optimal points ($|P|$ in Tables 1 and 2) we can observe slight differences between GUROBI and GLPK. This is because, in general, the ϵ -approximation that solves the pure stationary Pareto approximation problem is not unique and different MILP solvers might find different solutions.

Table 2: Results for stationary strategies.

Bench- mark	ℓ	Instance 1				$\varepsilon=0.01$		$\varepsilon=0.001$		Instance 2				$\varepsilon=0.01$		$\varepsilon=0.001$	
		Par.	$ S $	$\%E$	Act	Time	$ P $	Time	$ P $	Par.	$ S $	$\%E$	Act	Time	$ P $	Time	$ P $
dpm	2*	2	1272	32	3.2	1428	40	TO		3	1696	30	3.2	TO		TO	
eajs	2*	2-3	689	0	1.2	7	23	1249	201	3-6	$2 \cdot 10^4$	0	1.2	TO		TO	
jobs	3*	3-2	17	0	1.1	2	3	3	3	5-2	117	0	1.5	ERR		ERR	
mutex	3*	1	1795	36	2.2	TO		TO		2	$1 \cdot 10^4$	33	2.3	TO		TO	
polling	2	2-2	233	86	1.5	TO		TO		3-2	990	84	1.8	TO		TO	
rg	2*	2-1-20	2173	14	2.9	TO		TO		5-2-50	$3 \cdot 10^4$	5	3.1	TO		TO	
rover	2*	2500	$2 \cdot 10^4$	0	1.2	TO		TO		5000	$4 \cdot 10^4$	0	1.2	TO		TO	
serv	2*		$5 \cdot 10^4$	93	1.9	TO		TO									
str	2*	30	1426	0	1.3	11	21	TO		500	$4 \cdot 10^5$	0	1.3	3006	17	TO	
team2	2*	2	1847	24	1.2	2	5	2	5	3	$1 \cdot 10^4$	21	1.2	36	38	TO	
team3	3*	2	1847	24	1.2	49	15	TO		3	$1 \cdot 10^4$	21	1.2	TO		TO	
uav	2*	750	$2 \cdot 10^5$	29	1.6	TO		TO		1000	$4 \cdot 10^5$	31	1.8	TO		TO	
wlan	2*	0	2954	0	1.3	TO		TO		2	$3 \cdot 10^4$	0	1.3	TO		TO	



(a) Alt. (x) vs. original (y) encoding. (b) Pareto fronts for restricted strategies.

Figure 10: Comparison of the two encodings (left) and impact of memory (right).

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

