Multi-cost Bounded Reachability in MDP

Arnd Hartmanns[1], Sebastian Junges[2], Joost-Pieter Katoen[1,2], and Tim Quatmann[2](✉)

[1] University of Twente, Enschede, The Netherlands
[2] RWTH Aachen University, Aachen, Germany
tim.quatmann@cs.rwth-aachen.de

Abstract. We provide an efficient algorithm for multi-objective model-checking problems on Markov decision processes (MDPs) with multiple cost structures. The key problem at hand is to check whether there exists a scheduler for a given MDP such that all objectives over cost vectors are fulfilled. Reachability and expected cost objectives are covered and can be mixed. Empirical evaluation shows the algorithm's scalability. We discuss the need for output beyond Pareto curves and exploit the available information from the algorithm to support decision makers.

1 Introduction

Markov decision processes [41] (MDPs) with *rewards* or *costs* are a popular model to describe planning problems under uncertainty. Planning algorithms aim to find strategies which perform well (or even optimally) for a given objective. These algorithms typically assume *that a goal is reached eventually* [41,45]. This however is unrealistic in many scenarios, e.g. due to insufficient resources or the possibility of failing actions. Furthermore, these policies often admit single runs which perform far below the user's expectation, which is unsuitable in many scenarios with high stakes. Examples range from deliveries reaching an airport after the plane's departure to more serious scenarios in e.g. wildfire management [1]. In particular, many scenarios call for minimising the probability to run out of resources before reaching the goal: while it is *beneficial* for a plane to reach its destination with low *expected* fuel consumption, it is *essential* to reach its destination with the *fixed* available amount of fuel.

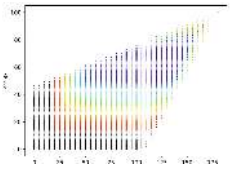
Policies that optimise solely for the probability to reach a goal are mostly very expensive. Even in the presence of just a single cost structure, decision makers have to trade the success probability against the costs. This makes many planning problems inherently multi-objective [12,17]. In particular, safety properties cannot be averaged out by good performance [21]. Planning scenarios in various application areas [44] have different resource constraints. Typical examples are energy consumption and time [11], or optimal expected revenue and time [38] in robotics, and monetary cost and available capacity in logistics [17].

This work is supported by the 3TU project "Big Software on the Run", CDZ project CAP, and DFG RTG 2236 "UnRAVeL".

© The Author(s) 2018
D. Beyer and M. Huisman (Eds.): TACAS 2018, LNCS 10806, pp. 320–339, 2018.
https://doi.org/10.1007/978-3-319-89963-3_19

task	time	energy consumption	scientific value	prob.
1	high	{low, medium}	medium	$1/2$
2	low	{medium, high}	medium	$3/5$
3	low	low	low	$4/5$
4	high	high	high	$1/10$

(a) Different possible tasks for a Mars rover



(b) Tradeoffs in costs

Fig. 1. Science on Mars: planning under several resource-constraints

Illustrative Example. Consider a simplified (discretised) version of the Mars rover task scheduling problem [11]. The task is to plan a variety of experiments for a day on Mars. The experiments vary in their success probability, time, energy consumption and their scientific value upon success. The time, energy consumption, and scientific value are uncertain and modelled by probability distributions, cf. Fig. 1(a). The objective is to achieve a minimum of daily scientific progress while limiting the risk of running out of time or out of energy. As the rover is expected to work for a longer period, we prefer a high expected scientific value.

Contributions and approach. This paper focuses on multi-objective cost-bounded reachability queries on MDPs, a natural setting for the aforementioned planning problems. The input is an MDP with multiple cost structures (e.g. energy, utility or time) and multiple objectives of the form "maximise/minimise the probability to reach a state in G_i such that the cumulative cost for the i-th cost structure is below/above a threshold b_i". This multi-objective variant of cost-bounded reachability is PSPACE-hard [43]. The focus of this paper is on the practical side: we aim at finding a practically efficient algorithm to obtain (an approximation of) the Pareto-optimal points. To accomplish this, we adapt and generalise recent approaches for the single-objective case [27,34] towards the multi-objective setting. The basic idea of [27,34] is to *implicitly* unfold the MDP along cost epochs, and exploit the regularities of the epoch-MDPs. Prism [37] and the Modest Toolset [29] have been updated with such methods for the single-objective case and significantly outperform the explicit unfolding approach of [2,40]. This paper presents an algorithm that lifts this principle to multiple cost objectives and determines approximation errors when using value iteration. Extensions towards quantiles and expected costs are considered too. Evaluation using a prototypical implementation in Storm [20] shows promising results. In addition, we equip our algorithm with means to visualise (inspired by the recent techniques in [39]) the trade-offs between various objectives that go beyond Pareto curves; we believe that this is key to obtain better insights into multi-objective decision making. An example is given in Fig. 1(b): it depicts the probability to satisfy an objective based on the remaining energy (y-axis) and time (x-axis).

Related work. The analysis of single-objective (cost-bounded) reachability in MDPs is an active area of research in both AI and formal method communities,

and referred to in, e.g., [18,35,48]. Various model checking approaches for single objectives exist. In [32], the topology of the unfolded MDP is exploited to speed up the value iteration. In [27], three different model checking approaches are explored and compared. A survey for heuristic approaches is given in [45]. A Q-learning based approach is described in [13]. An extension of this problem in the partially observable setting was considered in [14], and for probabilistic timed automata in [27]. The method from [4] computes optimal expected values under e.g. the *condition* that the goal is reached, and is thus applicable in settings where a goal is not *necessarily* reached. A similar problem is considered in [46]. For multi-objective analysis, the model checking community typically focuses on probabilities and expected costs as in the seminal works [15,22]. Implementations are typically based on a value iteration approach in [24], and have been extended to stochastic games [16], Markov automata [42], and interval MDPs [28]. Other considered cases include e.g. multi-objective mean-payoff objectives [8], objectives over instantaneous costs [10], and parity objectives [7]. Multi-objective problems for MDPs with an unknown cost-function are considered in [33]. Surveys on multi-objective decision making in AI and machine learning can be found in [44] and [47], respectively.

2 Preliminaries

We write 2^S for the powerset of S. The i-th component of a tuple $\boldsymbol{t} = \langle v_1, \ldots, v_n \rangle$ is $\boldsymbol{t}[i] \stackrel{\text{def}}{=} v_i$. A (discrete) *probability distribution* over a set Ω is a function $\mu \in \Omega \to [0,1]$ such that $\text{support}(\mu) \stackrel{\text{def}}{=} \{\, \omega \in \Omega \mid \mu(\omega) > 0 \,\}$ is countable and $\sum_{\omega \in \text{support}(\mu)} \mu(\omega) = 1$. $\text{Dist}(\Omega)$ is the set of all probability distributions over Ω. $\mathcal{D}(s)$ is the *Dirac distribution* for s, defined by $\mathcal{D}(s)(s) = 1$.

Definition 1. *A* Markov decision process *(MDP) with m cost structures is a triple $M = \langle S, T, s_{init} \rangle$ where S is a finite set of states, $T \in S \to 2^{\text{Dist}(\mathbb{N}^m \times S)}$ is the transition function, and $s_{init} \in S$ is the initial state. For all $s \in S$, we require that $T(s)$ is finite and non-empty.*

We write $s \to_T \mu$ for $\exists\, \mu \in T(s)$ and call it a *transition*. We write $s \xrightarrow{\boldsymbol{c}}_T s'$ if additionally $\langle \boldsymbol{c}, s' \rangle \in \text{support}(\mu)$. $\langle \boldsymbol{c}, s' \rangle$ is a *branch* with cost vector $\boldsymbol{c}$. If T is clear from the context, we just write $\to$. Graphically, transitions are lines to a node from which branches labelled with their probability and costs lead to successor states. We may omit the node and probability for transitions into Dirac distributions.

Example 1. Figure 2 shows an MDP M_{ex}. From the initial state s_0, the choice of going towards s_1 or s_2 is nondeterministic. Either way, the probability to return to s_0 is 0.5, otherwise we move to s_1 (or s_2). M_{ex} has two cost structures: Failing to move to s_1 has a cost of 1 for the first, and 2 for the second structure. Moving to s_2 yields cost 2 for the first and no cost for the second structure.

In the remainder of this paper, we fix a given MDP $M = \langle S, T, s_{init} \rangle$. Its semantics is captured by the notion of paths. A *path* in M represents the

infinite concrete resolution of both nondeterministic and probabilistic choices: $\pi = s_0\, \mu_0\, \boldsymbol{c}_0\, s_1\, \mu_1\, \boldsymbol{c}_1 \ldots$ where $s_i \in S$, $s_i \to \mu_i$, and $\langle \boldsymbol{c}_i, s_{i+1} \rangle \in \mathrm{support}(\mu_i)$ for all $i \in \mathbb{N}$. A *finite path* $\pi_{\mathrm{fin}} = s_0\, \mu_0\, \boldsymbol{c}_0\, s_1\, \mu_1\, \boldsymbol{c}_1\, s_2 \ldots \mu_{n-1}\, \boldsymbol{c}_{n-1}\, s_n$ is a finite prefix of a path with $\mathrm{last}(\pi_{\mathrm{fin}}) \stackrel{\mathrm{def}}{=} s_n \in S$. Let $\mathrm{cost}_i(\pi_{\mathrm{fin}}) \stackrel{\mathrm{def}}{=} \sum_{j=0}^{n-1} \boldsymbol{c}_j[i]$. $\mathrm{Paths}_{\mathrm{fin}}(M)$ ($\mathrm{Paths}(M)$) are the set of all (in)finite finite paths starting in s_{init}. A scheduler (*adversary*, *policy* or *strategy*) resolves nondeterministic choices:

Definition 2. $\mathfrak{S} \in \mathrm{Paths}_{\mathrm{fin}}(M) \to \mathrm{Dist}(\mathrm{Dist}(\mathbb{N}^m \times S))$ *is a* scheduler *for* M *if* $\forall\, \pi_{\mathrm{fin}}\colon \mu \in \mathrm{support}(\mathfrak{S}(\pi_{\mathrm{fin}})) \Rightarrow \mathrm{last}(\pi_{\mathrm{fin}}) \to_T \mu$. *The set of all schedulers of* M *is* $\mathrm{Sched}(M)$. $\mathfrak{S}$ *is* deterministic *if* $|\mathrm{support}(\mathfrak{S}(\pi))| = 1$ *for all finite paths* π.

Via the standard cylinder set construction [25], a scheduler $\mathfrak{S}$ induces a probability measure $\mathcal{P}_M^{\mathfrak{S}}$ on measurable sets of paths starting from s_{init}. We define the *extremal* values $\mathcal{P}_M^{\max}(\Pi) = \sup_{\mathfrak{S} \in \mathrm{Sched}(M)} \mathcal{P}_M^{\mathfrak{S}}(\Pi)$ and $\mathcal{P}_M^{\min}(\Pi) = \inf_{\mathfrak{S} \in \mathrm{Sched}(M)} \mathcal{P}_M^{\mathfrak{S}}(\Pi)$ for measurable $\Pi \subseteq \mathrm{Paths}(M)$. For clarity, we focus on probabilities in this paper, but note that expected accumulated costs can be defined analogously [25] and our methods apply to them with only minor changes.

Cost-Bounded Reachability. We are interested in the probabilities of sets of paths that reach certain goal states within multiple cost bounds:

Definition 3. *A* cost bound *is given by* $\langle \mathrm{C}_j \rangle_{\sim b}\, G$ *where* $j \in \{1, \ldots, m\}$ *identifies a cost structure,* $\sim \in \{<, \leq, >, \geq\}$, $b \in \mathbb{N}$ *is a bound value, and* $G \subseteq S$ *is a set of goal states. A* cost-bounded reachability formula *is a conjunction* $\bigwedge_{i=1}^{n \in \mathbb{N}} (\langle \mathrm{C}_{j_i} \rangle_{\sim_i b_i}\, G_i)$ *of cost bounds. It characterises the measurable set of paths* Π *where, for every* i, *every* $\pi \in \Pi$ *has a prefix* π_{fin}^i *with* $\mathrm{last}(\pi_{\mathrm{fin}}^i) \in G_i$ *and* $\mathrm{cost}_{j_i}(\pi_{\mathrm{fin}}^i) \sim_i b_i$.

A (single-objective) multi-cost bounded reachability query asks for $\mathcal{P}_M^{opt}(e)$ where $opt \in \{\max, \min\}$ and e is a cost-bounded reachability formula. Unbounded and step-bounded reachability are special cases of cost-bounded reachability. A single-objective query may contain multiple bounds, but asks for a *single* scheduler that optimises the probability of satisfying them all.

We also consider multi-objective *tradeoffs*, i.e. sets of single-objective queries written as $\Phi = \mathit{multi}\big(\mathcal{P}_M^{opt_1}(e_1), \ldots, \mathcal{P}_M^{opt_\ell}(e_\ell)\big)$. We call the e_k *objectives*. For tradeoffs, we are interested in the *Pareto curve* $\mathit{Pareto}(M, \Phi)$ which consists of all achievable probability vectors $\boldsymbol{p}_{\mathfrak{S}} = \langle \mathcal{P}_M^{\mathfrak{S}}(e_1), \ldots, \mathcal{P}_M^{\mathfrak{S}}(e_\ell) \rangle$ for $\mathfrak{S} \in \mathrm{Sched}(M)$ that are not *dominated* by another achievable vector $\boldsymbol{p}_{\mathfrak{S}'}$. More precisely, $\boldsymbol{p}_{\mathfrak{S}} \in \mathit{Pareto}(M, \Phi)$ iff for all $\mathfrak{S}' \in \mathrm{Sched}(M)$ either $\boldsymbol{p}_{\mathfrak{S}} = \boldsymbol{p}_{\mathfrak{S}'}$ or for some $i \in \{1, \ldots, \ell\}$ we have $(opt_i = \max \wedge \boldsymbol{p}_{\mathfrak{S}}[i] > \boldsymbol{p}_{\mathfrak{S}'}[i]) \vee (opt_i = \min \wedge \boldsymbol{p}_{\mathfrak{S}}[i] < \boldsymbol{p}_{\mathfrak{S}'}[i])$.

Example 2. We consider $\Phi = \mathit{multi}\big(\mathcal{P}_{M_{ex}}^{\max}(\langle \mathrm{C}_1 \rangle_{\leq 1}\, \{s_1\}), \mathcal{P}_{M_{ex}}^{\max}(\langle \mathrm{C}_2 \rangle_{\leq 3}\, \{s_2\})\big)$ for M_{ex} of Fig. 2. Let $\mathfrak{S}_j$ be the scheduler that tries to move to s_1 for at most j attempts and afterwards moves to s_2. The induced probability vectors $\boldsymbol{p}_{\mathfrak{S}_1} = \langle 0.5, 1 \rangle$ and $\boldsymbol{p}_{\mathfrak{S}_2} = \langle 0.75, 0.75 \rangle$ both lie on the Pareto curve since no

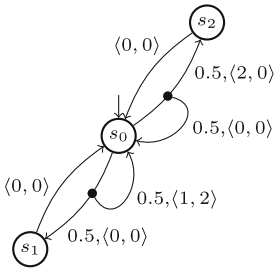
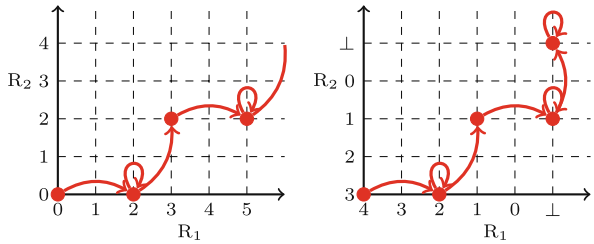


Fig. 2. Example MDP M_{ex}



(a) The naive approach (b) Cost epochs

Fig. 3. An illustration of epochs

$\mathfrak{S} \in \mathrm{Sched}(M_{ex})$ induces (strictly) larger probabilities $\boldsymbol{p}_{\mathfrak{S}}$. By also considering schedulers that randomise between the choices of $\mathfrak{S}_1$ and $\mathfrak{S}_2$ we obtain $\mathit{Pareto}(M_{ex}, \Phi) = \{w \cdot \boldsymbol{p}_{\mathfrak{S}_1} + (1-w) \cdot \boldsymbol{p}_{\mathfrak{S}_2} \mid w \in [0,1]\}$.

For clarity of presentation, we restrict to tradeoffs Φ where every cost structure occurs exactly once, i.e., the number m of cost structures of M matches the number of cost bounds occurring in Φ. Furthermore, we require that none of the sets of goal states contains the initial state. Both assumptions are w.l.o.g. by copying cost structures as needed and adding a new initial state with zero-cost transition to the old initial state.

3 Multi-dimensional Sequential Value Iteration

We present a practically efficient approach to compute (an approximation of) the Pareto curve for MDP M with m cost structures and tradeoff Φ. We merge the ideas of [24] to approximate a Pareto curve for an (unbounded) multi-objective tradeoff with those of [27,34] to efficiently compute (single-objective) cost-bounded reachability probabilities. For clarity of presentation we start with the upper-bounded maximum case and assume a tradeoff of the form $\Phi = \mathit{multi}\big(\mathcal{P}_M^{\max}(e_1), \ldots, \mathcal{P}_M^{\max}(e_\ell)\big)$ with $e_k = \bigwedge_{i=n_{k-1}}^{n_k-1} (\langle \mathrm{C}_i \rangle_{\leq b_i} G_i)$ and $0 = n_0 < n_1 < \cdots < n_\ell = m$. Other variants are discussed in Sect. 3.3.

Cost epochs and goal satisfaction. Central to our approach is the concept of *cost epochs*. Consider the path $\pi = (s_0\langle 2,0\rangle s_2\langle 0,0\rangle s_0\langle 1,2\rangle)^\omega$ through M_{ex} of Fig. 2. We plot the accumulated cost in both dimensions along this path in Fig. 3(a). Starting from $\langle 0,0\rangle$, the first transition yields cost 2 for the first cost structure: we jump to coordinate $\langle 2,0\rangle$. The next transition, back to s_0, has no cost, so we stay at $\langle 2,0\rangle$. Finally, the failed attempt to move to s_1 incurs costs $\langle 1,2\rangle$. Consequently, for an infinite path, infinitely many points in this grid may be reached. However, a tradeoff specifies bound values for the costs, e.g., for $\Phi_{ex} = \mathit{multi}\big(\mathcal{P}_{M_{ex}}^{\max}(\langle \mathrm{C}_1 \rangle_{\leq 4} \{s_1\}), \mathcal{P}_{M_{ex}}^{\max}(\langle \mathrm{C}_2 \rangle_{\leq 3} \{s_2\})\big)$ we get bound values 4 and 3. Once the bound value for a bound is reached, accumulating further costs in this dimension does not impact the satisfaction of its formula. It thus suffices

to keep track, for each bound, of the *remaining* costs before reaching the bound value. This leads to a finite grid as depicted in Fig. 3(b). We refer to each of its coordinates as a cost epoch:

Definition 4. *An m-dimensional* cost epoch *is a tuple in $\boldsymbol{E}_m \stackrel{\text{def}}{=} (\mathbb{N} \cup \{\bot\})^m$. For $\boldsymbol{e} \in \boldsymbol{E}_m$, $\boldsymbol{c} \in \mathbb{N}^m$, the* successor epoch *is $succ(\boldsymbol{e}, \boldsymbol{c})[i] \stackrel{\text{def}}{=} \boldsymbol{e}[i] - \boldsymbol{c}[i]$ if that value is non-negative and $\bot$ otherwise.*

If the entry for a bound is $\bot$, it cannot be satisfied any more: too much costs have already been incurred. To check whether an objective $e_k = \bigwedge_{i=n_{k-1}}^{n_k-1} (\langle \mathrm{C}_i \rangle_{\leq b_i} G_i)$ is satisfied, we memorise whether each individual bound already holds. This is also used to ensure that satisfying a bound more than once has no effect.

Definition 5. *A* goal satisfaction *$\boldsymbol{g} \in \boldsymbol{G}_m \stackrel{\text{def}}{=} \{0,1\}^m$ represents the cost structure indices i for which bound $\langle \mathrm{C}_i \rangle_{\leq b_i} G_i$ already holds, i.e. G_i was reached before the bound value b_i. For $\boldsymbol{g} \in \boldsymbol{G}_m$, $\boldsymbol{e} \in \boldsymbol{E}_m$ and $s \in S$, let $succ(\boldsymbol{g}, s, \boldsymbol{e}) \in \boldsymbol{G}_m$ define the update upon reaching s: $succ(\boldsymbol{g}, s, \boldsymbol{e})[i] = 1$ if $s \in G_i \wedge \boldsymbol{e}[i] \neq \bot$ and $succ(\boldsymbol{g}, s, \boldsymbol{e})[i] = \boldsymbol{g}[i]$ otherwise.*

3.1 The Unfolding Approach

$Pareto(M, \Phi)$ can be computed by reducing Φ to a multi-objective *unbounded* reachability problem on the *unfolded* MDP. Its states are the Cartesian product of the original MDP's states, the epochs, and the goal satisfactions:

Definition 6. *The* unfolding *for M as in Definition 1 and upper-bounded maximum tradeoff Φ is the MDP $M_{unf} = \langle S' \stackrel{\text{def}}{=} S \times \boldsymbol{E}_m \times \boldsymbol{G}_m, T', \langle s_{init}, \langle b_1, \ldots, b_m \rangle, \boldsymbol{0} \rangle \rangle$ with no cost structures, $T'(\langle s, \boldsymbol{e}, \boldsymbol{g} \rangle) \stackrel{\text{def}}{=} \{\, unf(\mu) \in \mathrm{Dist}(\mathbb{N}^0 \times S') \mid \mu \in T(s) \,\}$ and the unfolding of probability distribution μ defined by $unf(\mu)(\langle \langle s', \boldsymbol{e}', \boldsymbol{g}' \rangle \rangle) = \mu(\langle \boldsymbol{c}, s' \rangle)$ if $\boldsymbol{e}' = succ(\boldsymbol{e}, \boldsymbol{c}) \wedge \boldsymbol{g}' = succ(\boldsymbol{g}, s', \boldsymbol{e}')$ and 0 otherwise.*

Costs are now encoded in the state space, so it suffices to consider the unbounded tradeoff $\Phi' = multi\big(\mathcal{P}^{\max}_{M_{unf}}(e'_1), \ldots, \mathcal{P}^{\max}_{M_{unf}}(e'_\ell)\big)$ with $e'_k = \langle \cdot \rangle_{\geq 0} G'_k$ and $G'_k = \{\langle s, \boldsymbol{e}, \boldsymbol{g} \rangle \mid \bigwedge_{i=n_{k-1}}^{n_k-1} \boldsymbol{g}[i] = 1\}$.

Lemma 1. *There is a bijection $f \colon \mathrm{Sched}(M) \to \mathrm{Sched}(M_{unf})$ with $\mathcal{P}^{\mathfrak{S}}_M(e_k) = \mathcal{P}^{f(\mathfrak{S})}_{M_{unf}}(e'_k)$ for all $\mathfrak{S} \in \mathrm{Sched}(M)$ and $k \in \{\, 1, \ldots, \ell \,\}$. Consequently, we have that $Pareto(M, \Phi) = Pareto(M_{unf}, \Phi')$.*

$Pareto(M_{unf}, \Phi')$ can be computed with existing multi-objective model checking algorithms for unbounded reachability. We build on the one of [24]. It iteratively chooses weight vectors $\boldsymbol{w} = \langle w_1, \ldots, w_\ell \rangle \in [0,1]^\ell \setminus \{\boldsymbol{0}\}$ and computes points

$$\boldsymbol{p_w} = \langle \mathcal{P}^{\mathfrak{S}}_{M_{unf}}(e'_1), \ldots, \mathcal{P}^{\mathfrak{S}}_{M_{unf}}(e'_\ell) \rangle \text{ with } \mathfrak{S} \in \arg\max_{\mathfrak{S}'} \left(\sum_{k=1}^{\ell} w_k \cdot \mathcal{P}^{\mathfrak{S}'}_{M_{unf}}(e'_k) \right). \tag{1}$$

The Pareto curve $\boldsymbol{P}$ is convex, $\boldsymbol{p_w} \in \boldsymbol{P}$ for all $\boldsymbol{w}$, and $\boldsymbol{q} \in \boldsymbol{P}$ implies $\boldsymbol{q} \cdot \boldsymbol{w} \leq \boldsymbol{p_w} \cdot \boldsymbol{w}$. These observations allow us to approximate the Pareto curve with arbitrary precision; see [24] for details. [24] characterises $\boldsymbol{p_w}$ via weighted expected costs: M_{unf} is equipped with ℓ cost structures used to calculate the probability of each of the ℓ objectives. This is achieved by setting the value of the k-th cost structure on each branch to 1 iff the objective e'_k is satisfied in the target state of the branch but was *not* satisfied in the transition's source state. On a path π through the resulting model M^+_{unf}, we collect exactly one cost w.r.t. cost structure k iff π satisfies objective e_k.

Definition 7. *For* $\mathfrak{S} \in \mathrm{Sched}(M^+_{unf})$ *and* $\boldsymbol{w} \in [0,1]^\ell$*, the* weighted expected cost *is* $\mathcal{E}^{\mathfrak{S}}_{M^+_{unf}}(\boldsymbol{w}) = \sum_{k=1}^{\ell} \boldsymbol{w}[k] \cdot \int_{\pi \in \mathrm{Paths}(M)} \mathrm{cost}_k(\pi) \mathrm{d}\mathcal{P}^{\mathfrak{S}}_{M^+_{unf}}(\pi)$*, i.e. the expected value of the weighted sum of the costs accumulated on paths in* M^+_{unf}*.*

The following characterisation of $\boldsymbol{p_w}$ is equivalent to Eq. 1:

$$\boldsymbol{p_w} = \langle \mathcal{E}^{\mathfrak{S}}_{M^+_{unf}}(\mathbf{1}_1), \ldots, \mathcal{E}^{\mathfrak{S}}_{M^+_{unf}}(\mathbf{1}_\ell) \rangle \quad \text{where} \quad \mathfrak{S} \in \arg\max_{\mathfrak{S}'} \mathcal{E}^{\mathfrak{S}'}_{M^+_{unf}}(\boldsymbol{w}) \tag{2}$$

and $\mathbf{1}_k \in \{0,1\}^\ell$ is the weight vector defined by $\mathbf{1}_k[j] = 1$ iff $j = k$. Standard MDP model checking algorithms [41] can be applied to compute an optimal (deterministic and memoryless) scheduler $\mathfrak{S}$ and the induced costs $\mathcal{E}^{\mathfrak{S}}_{M^+_{unf}}(\mathbf{1}_k)$.

3.2 An Epoch Model Approach Without Unfolding

The unfolding approach does not scale well: If the original MDP has n states, the unfolding will have on the order of $n \cdot \prod_{i=1}^{m}(b_i + 2)$ states. This makes it infeasible for larger bound values b_i over multiple bounds. The bottleneck lies in computing the points $\boldsymbol{p_w}$ as in Eqs. 1 and 2. We now show how to do so efficiently, i.e. given a weight vector $\boldsymbol{w} = \langle w_1, \ldots, w_\ell \rangle \in [0,1]^\ell \setminus \{\mathbf{0}\}$, compute

$$\boldsymbol{p_w} = \langle \mathcal{P}^{\mathfrak{S}}_M(e_1), \ldots, \mathcal{P}^{\mathfrak{S}}_M(e_\ell) \rangle \text{ with } \mathfrak{S} \in \arg\max_{\mathfrak{S}'} \Big(\sum_{k=1}^{\ell} w_i \cdot \mathcal{P}^{\mathfrak{S}'}_M(\langle \cdot \rangle_{\geq 0}\, e_k) \Big) \tag{3}$$

without unfolding. The characterisations of $\boldsymbol{p_w}$ given in Eqs. 1 and 3 are equivalent due to Lemma 1.

The efficient analysis of single-objective queries with a single bound $\Phi_1 = \mathcal{P}^{\max}_M(\langle \mathrm{C} \rangle_{\leq b}\, G)$ has recently been addressed in e.g. [27,34]. The key observation is that the unfolding M_{unf} can be decomposed into $b+2$ *epoch model* MDPs $M^b, \ldots, M^0, M^\perp$ corresponding to the cost epochs. The epoch models are copies of M with only slight adaptations. Reachability probabilities in copies corresponding to epoch i only depend on the copies $\{\, M^j \mid j \leq i \vee j = \perp \,\}$. It is thus possible to analyse $M^\perp, \ldots, M^b$ sequentially instead of considering all copies at once. In particular, it is not necessary to construct the full unfolding.

We lift this idea to multi-objective tradeoffs. The single-objective case is notably simpler in that reaching a goal state for the first time or exceeding the cost bound immediately suffices to determine whether the one property is

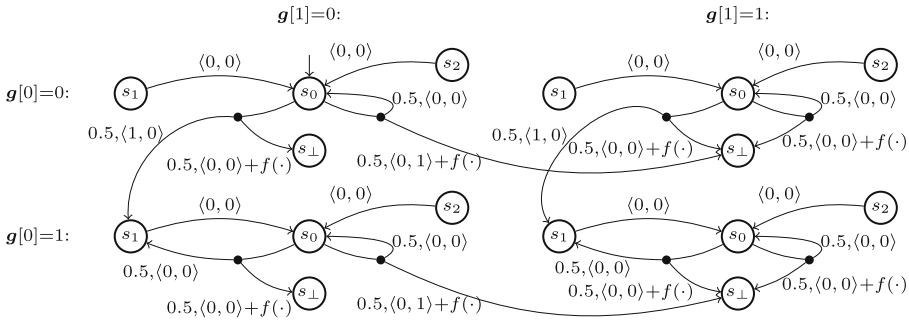


Fig. 4. An epoch model of M_{ex}

satisfied. In particular, while $M^{\perp}$ is just one sink state in the single-objective case, its structure is more involved here.

We first formalise the notion of *epoch models* for multiple bounds. The aim is to build an MDP for each epoch $\boldsymbol{e} \in \boldsymbol{E}_m$ that can be analysed via standard model checking techniques using the weighted expected cost encoding of objective probabilities. The state space of an epoch model consists of up to one copy of each original state for each goal satisfaction vector $\boldsymbol{g} \in \boldsymbol{G}_m$. Additional sink states $\langle s_{\perp}, \boldsymbol{g}\rangle$ encode the target for a jump to *any* other cost epoch $\boldsymbol{e}' \neq \boldsymbol{e}$. We consider ℓ cost structures to encode the objective probabilities. Let function $satObj_{\Phi} \colon \boldsymbol{G}_m \times \boldsymbol{G}_m \to \{0,1\}^{\ell}$ assign value 1 in entry k iff a reachability property e_k is satisfied according to the second goal vector but was not satisfied in the first. For the transitions' branches, we distinguish two cases: (1) If the successor epoch $\boldsymbol{e}' = succ(\boldsymbol{e}, \boldsymbol{c})$ with respect to the *original* cost $\boldsymbol{c} \in \mathbb{N}^m$ is the same as the current epoch $\boldsymbol{e}$, we jump to the successor state as before, and update the goal satisfaction. We collect the *new* costs for the *objectives* if updating the goal satisfaction newly satisfies an objective as given by $satObj_{\Phi}$ (2). If the successor epoch $\boldsymbol{e}' = succ(\boldsymbol{e}, \boldsymbol{c})$ is different from the current epoch $\boldsymbol{e}$, the probability is rerouted to the sink state with the corresponding goal state satisfaction vector. The collected costs contains the part of the goal satisfaction as in (1), but also the results obtained by analysing the reached epoch $\boldsymbol{e}'$, given by a function f.

Definition 8. *The* epoch model *of MDP M as in Definition 1 for $\boldsymbol{e} \in \boldsymbol{E}_m$ and a function $f \colon \boldsymbol{G}_m \times \mathrm{Dist}(\mathbb{N}^m \times S) \to [0,1]^{\ell}$ is the MDP $M_f^{\boldsymbol{e}} = \langle S^{\boldsymbol{e}}, T_f^{\boldsymbol{e}}, \langle s_{init}, \boldsymbol{0}\rangle\rangle$ with ℓ cost structures, $S^{\boldsymbol{e}} \stackrel{\text{def}}{=} (S \uplus s_{\perp}) \times \boldsymbol{G}_m$, $T_f^{\boldsymbol{e}}(\langle s_{\perp}, \boldsymbol{g}\rangle) = \{\, \mathcal{D}(\langle \boldsymbol{0}, \langle s_{\perp}, \boldsymbol{g}\rangle\rangle) \,\}$, and for every $\tilde{s} = \langle s, \boldsymbol{g}\rangle \in S^{\boldsymbol{e}}$ and $\mu \in T(s)$, there is some $\nu \in T_f^{\boldsymbol{e}}(\tilde{s})$ defined by:*

1. *$\nu(\langle satObj_{\Phi}(\boldsymbol{g}, \boldsymbol{g}'), \langle s', \boldsymbol{g}'\rangle\rangle) = \mu(\boldsymbol{c}, s')$ if $succ(\boldsymbol{e}, \boldsymbol{c}) = \boldsymbol{e} \wedge \boldsymbol{g}' = succ(\boldsymbol{g}, s', \boldsymbol{e})$, and*
2. *$\nu(\langle f(\boldsymbol{g}, \mu) + satObj_{\Phi}(\boldsymbol{g}, \boldsymbol{g}'), \langle s_{\perp}, \boldsymbol{g}'\rangle\rangle) = \sum_{\boldsymbol{c} \in \boldsymbol{C}} \sum_{s' \in S'_{\boldsymbol{c}}} \mu(\boldsymbol{c}, s')$ where $\boldsymbol{C} = \{\boldsymbol{c} \mid succ(\boldsymbol{e}, \boldsymbol{c}) \neq \boldsymbol{e}\}$ and $S'_{\boldsymbol{c}} = \{s' \mid succ(\boldsymbol{g}, s', succ(\boldsymbol{e}, \boldsymbol{c})) = \boldsymbol{g}'\}$.*

Figure 4 shows an epoch model $M_f^{\boldsymbol{e}}$ of the MDP M_{ex} in Fig. 2 with respect to tradeoff Φ as in Example 2 and any epoch $\boldsymbol{e} \in \boldsymbol{E}_2$ with $\boldsymbol{e}[1] \neq \perp$ and $\boldsymbol{e}[2] \neq \perp$.

Input : MDP $M = \langle S, T, s_{init} \rangle$, tradeoff $\Phi = \mathit{multi}\big(\mathcal{P}_M^{\max}(e_1), \ldots, \mathcal{P}_M^{\max}(e_\ell)\big)$ with bound values $b_1, \ldots, b_m$, weight vector $\boldsymbol{w} \in [0,1]^\ell$ and proper epoch sequence $\mathbb{E}$ ending with $\mathrm{last}(\mathbb{E}) = \langle b_1, \ldots, b_m \rangle$

Output : Point $\boldsymbol{p_w} \in \mathbb{R}^\ell$ satisfying Eq. 3

1 **foreach** $\boldsymbol{e} \in \mathbb{E}$ in ascending order **do**
2   **foreach** $\boldsymbol{g} \in \boldsymbol{G}_m$, $\mu \in \{\nu \mid \exists s\colon\ \nu \in T(s)\}$ **do**
3         $\boldsymbol{z} \leftarrow \boldsymbol{0}$
4   **foreach** $\langle \boldsymbol{c}, s' \rangle \in \mathrm{support}(\mu)$ **do**
5             $\boldsymbol{e}' \leftarrow \mathit{succ}(\boldsymbol{e}, \boldsymbol{c})$; $\boldsymbol{g}' \leftarrow \mathit{succ}(\boldsymbol{g}, s', \boldsymbol{e}')$
6   **if** $\boldsymbol{e}' \neq \boldsymbol{e}$ **then**
7                 $\boldsymbol{z} \leftarrow \boldsymbol{z} + \mu(\boldsymbol{c}, s') \cdot x^{\boldsymbol{e}'}[\langle s', \boldsymbol{g}' \rangle]$
8         $f(\boldsymbol{g}, \mu) \leftarrow \boldsymbol{z}$
9   build epoch model $M_f^{\boldsymbol{e}} = \langle S^{\boldsymbol{e}}, T_f^{\boldsymbol{e}}, s_{init}^{\boldsymbol{e}} \rangle$
10     $\mathfrak{S} \leftarrow \arg\max_{\mathfrak{S}'} \mathcal{E}_{M_f^{\boldsymbol{e}}}^{\mathfrak{S}'}(\boldsymbol{w})$
11   **foreach** $k \in \{1, \ldots, \ell\}$, $\tilde{s} \in S^{\boldsymbol{e}}$ **do**
12         $x^{\boldsymbol{e}}[\tilde{s}][k] \leftarrow \mathcal{E}_{M_f^{\boldsymbol{e}}}^{\mathfrak{S}}(\boldsymbol{1}_k)[\tilde{s}]$
13 **return** $x^{\mathrm{last}(\mathbb{E})}[s_{init}^{\mathrm{last}(\mathbb{E})}]$

Algorithm 1. Sequential multi-cost bounded analysis

Remark 1. The structure of $M_f^{\boldsymbol{e}}$ differs only slightly between epochs. In particular consider epochs $\boldsymbol{e}, \boldsymbol{e}'$ with $\boldsymbol{e}[i] = \bot$ iff $\boldsymbol{e}'[i] = \bot$. To construct epoch model $M_f^{\boldsymbol{e}'}$ from $M_f^{\boldsymbol{e}}$, only transitions to the bottom states $\langle s_\bot, \boldsymbol{g} \rangle$ need to be adapted.

To analyse an epoch model $M_f^{\boldsymbol{e}}$, any successor epoch $\boldsymbol{e}'$ of $\boldsymbol{e}$ needs to be analysed before. Since costs are non-negative, we can ensure this by analysing the epochs in a specific order. In the single dimensional case the order is uniquely given by $\bot, 0, 1, \ldots, b$. For multiple cost bounds any linearisation of the partial order $\preceq \subseteq \boldsymbol{E}_m \times \boldsymbol{E}_m$ with $\boldsymbol{e}' \preceq \boldsymbol{e}$ iff $\boldsymbol{e}'[i] \leq \boldsymbol{e}[i] \vee \boldsymbol{e}'[i] = \bot$ for all i can be considered. We call such a linearisation a *proper epoch sequence*.

We compute the points $\boldsymbol{p_w}$ by analysing the different epoch models (i.e. the coordinates of Fig. 3(b)) sequentially. The main procedure is outlined in Algorithm 1. The costs of the model for the current epoch are computed in lines 2-8. These costs comprise the results from previously analysed epochs $\boldsymbol{e}'$. In lines 9-12, the current epoch model $M_f^{\boldsymbol{e}}$ is built and analysed: We compute weighted expected costs on $M_f^{\boldsymbol{e}}$ where $\mathcal{E}_{M_f^{\boldsymbol{e}}}^{\mathfrak{S}}(\boldsymbol{w})[s]$ denotes the expected costs for $M_f^{\boldsymbol{e}}$ when changing the initial state to s. In line 10 a (deterministic and memoryless) scheduler $\mathfrak{S}$ that induces the maximal weighted expected costs (i.e. $\mathcal{E}_{M_f^{\boldsymbol{e}}}^{\mathfrak{S}}(\boldsymbol{w})[s] = \max_{\mathfrak{S}'} \mathcal{E}_{M_f^{\boldsymbol{e}}}^{\mathfrak{S}'}(\boldsymbol{w})[s]$ for all states s) is computed. In line 12 we then compute the expected costs induced by $\mathfrak{S}$ for the individual objectives.

Theorem 1. *The output of Algorithm 1 satisfies Eq. 3.*

Proof (sketch). Let $\boldsymbol{e}$ be the currently analysed epoch. Since $\mathbb{E}$ is assumed to be a *proper* epoch sequence, we already processed any reachable successor epoch $\boldsymbol{e}'$

of $\boldsymbol{e}$, i.e., line 7 is only executed for epochs $\boldsymbol{e}'$ for which $x^{\boldsymbol{e}'}$ has already been computed. One can show that the values $x^{\boldsymbol{e}}[\langle s, \boldsymbol{g}\rangle][k]$ computed by the algorithm coincide with the probability to satisfy e'_k from state $\langle s, \boldsymbol{e}, \boldsymbol{g}\rangle$ in the unfolding M_{unf} under a scheduler $\mathfrak{S}$ that maximises the weighted sum.

Error propagation. So far, we assumed that (weighted) expected costs $\mathcal{E}_M^{\mathfrak{S}}(\boldsymbol{w})$ are computed exactly. Practical implementations, however, are often based on numerical methods that only approximate the correct solution. In fact, methods based on value iteration—the de-facto standard in MDP model checking—do not give any guarantee on the accuracy of the obtained result [26]. We therefore consider interval iteration [5,9] which for a predefined precision $\varepsilon > 0$ guarantees that the obtained result x_s is ε-precise, i.e. we have $|x_s - \mathcal{E}_M^{\mathfrak{S}}(\boldsymbol{w})[s]| \leq \varepsilon$.

For the single-cost bounded variant of Algorithm 1, [27] discusses that in order to compute $\mathcal{P}_M^{\max}(\langle \mathrm{C}\rangle_{\leq b}\, G)$ with precision ε, each epoch model needs to be analysed with precision $\frac{\varepsilon}{b+1}$. We generalise this result to multi-dimensional tradeoffs. Assume the results of previously analysed epochs (given by f) are ε-precise and that $M_f^{\boldsymbol{e}}$ is analysed with precision δ. As in the single-dimensional case, the total error for $M_f^{\boldsymbol{e}}$ can accumulate to $\delta + \varepsilon$. Since a path through the MDP M can visit at most $\sum_{i=1}^{m}(b_i + 1)$ cost epochs whose analysis introduces error δ, the overall error can be upper bounded by $\delta \cdot \sum_{i=1}^{m}(b_i + 1)$.

Theorem 2. *If the values $x^{\boldsymbol{e}}[\tilde{s}][k]$ at line 12 of Algorithm 1 are computed with precision $\varepsilon / \sum_{i=1}^{m}(b_i+1)$ for some $\varepsilon > 0$, the output $\boldsymbol{p}'_{\boldsymbol{w}}$ of the algorithm satisfies $|\boldsymbol{p}_{\boldsymbol{w}} - \boldsymbol{p}'_{\boldsymbol{w}}| \cdot \boldsymbol{w} \leq \varepsilon$ where $\boldsymbol{p}_{\boldsymbol{w}}$ is as in Eq. 3.*

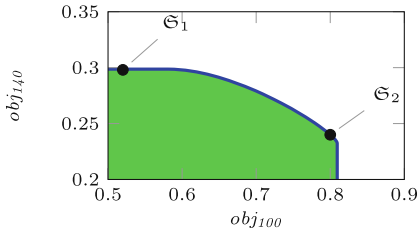
Remark 2. Alternatively, epochs can be analysed with the desired overall precision ε by lifting the results from topological interval iteration [5]. However, that requires to store the obtained bounds for the results of already analysed epochs.

3.3 Extensions

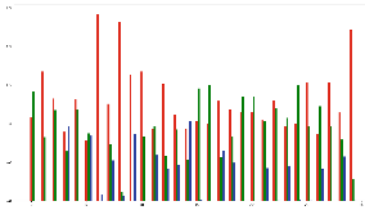
Minimising objectives. Objectives $\mathcal{P}_M^{\min}(e_k)$ can be handled by extending the function $satObj_{\Phi}$ in Definition 8 such that it assigns cost -1 to branches that lead to the satisfaction of e_k. To obtain the desired probabilities we then maximise negative costs and multiply the result by -1 afterwards. As interval iteration supports mixtures of positive and negative costs [5], arbitrary combinations of minimising and maximising objectives can be considered[1].

Beyond upper bounds. Our approach also supports bounds of the form $\langle \mathrm{C}_j\rangle_{\sim b}\, G$ for $\sim \in \{<, \leq, >, \geq\}$, i.e., we allow *combinations* of lower and upper cost-bounds. For strict upper bounds $< b$ and non-strict lower bounds $\geq b$ we consider $\leq b + 1$ and $> b - 1$ instead. For bound $\langle \mathrm{C}_i\rangle_{>b_i}\, G_i$ we adapt the update of goal satisfactions such that $succ(\boldsymbol{g}, s, \boldsymbol{e})[i] = 1$ if either $\boldsymbol{g}[i] = 1$ or $s \in G_i \wedge \boldsymbol{e}[i] = \bot$. Similarly, we support multi-bounded-single-goal queries of the form $\langle \mathrm{C}_{(j_1,\ldots,j_n)}\rangle_{(\sim_1 b_1,\ldots,\sim_n b_n)}\, G$ which characterises the paths π with a single prefix π_{fin} satisfying $\text{last}(\pi_{\text{fin}}) \in G$ and *all* cost bounds, i.e., $\text{cost}_{j_i}(\pi_{\text{fin}}) \sim_i b_i$.

[1] This supersedes a restriction of the algorithm of [24].



(a) Pareto curve for $multi(obj_{100}, obj_{140})$



(b) Optimal schedulers for 3 objectives

Fig. 5. Pareto curves

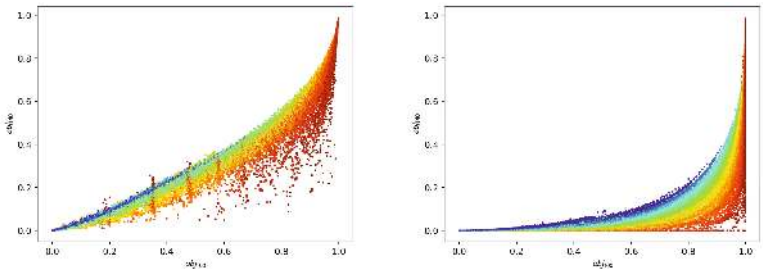
Example 3. The formula $e = \langle \mathrm{C}_{(1,1)} \rangle_{(\leq 1, \geq 1)}\, G$ expresses the paths that reach G while collecting exactly one cost w.r.t. the first cost structure. This formula is not equivalent to $e' = \langle \mathrm{C}_1 \rangle_{\leq 1}\, G \wedge \langle \mathrm{C}_1 \rangle_{\geq 1}\, G$ since, e.g., for $G = \{\, s_0 \,\}$ the path $\pi = s_0 \langle 2 \rangle s_0$ satisfies e' but not e.

Expected cost objectives. We can consider cost-bounded expected cost objectives $\mathcal{E}_M^{opt}(\mathrm{R}_{j_1}, \langle \mathrm{C}_{j_2} \rangle_{\leq b})$ with $opt \in \{\, \max, \min \,\}$ which refer to the expected cost accumulated for cost structure j_1 within a given cost bound $\langle \mathrm{C}_{j_2} \rangle_{\leq b}$. Similar to cost-bounded reachability queries, we compute cost-bounded expected costs via computing (weighted) expected costs within epoch models.

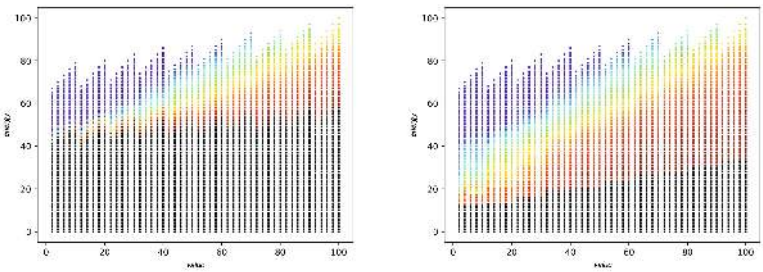
Quantiles. A (multi-dimensional) quantile has the form $Qu(\mathcal{P}_M^{opt}(e) \sim p)$ for $opt \in \{\, \min, \max \,\}$, $\sim \in \{<, \leq, >, \geq\}$, $e = \bigwedge_{i=1}^{n \in \mathbb{N}} (\langle \mathrm{C}_{j_i} \rangle_{\sim_i b_i}\, G_i)$ and a fixed probability threshold $p \in [0, 1]$. The quantile asks for the set of bound values $\mathcal{B}$ that satisfy the probability threshold, i.e., $\mathcal{B} = \{\langle b_1 \ldots, b_n \rangle \mid \mathcal{P}_M^{opt}(e) \sim p\}$. The computation of quantiles for single-cost bounded reachability has been discussed in [3,34], where multiple cost bounds are supported via unfolding. Unfolding requires to fix bound values $b_2, \ldots, b_n$ a priori, and one can only ask for all b_1 that satisfy the property. Our approach provides the basis for lifting the ideas of [3,34] to multi-bounded queries. Roughly, one extends the epoch sequence $\mathbb{E}$ in Algorithm 1 dynamically until the epochs in which the bounded reachability probability passes the threshold p are explored. Additional steps such as detecting the case where $\mathcal{B} = \emptyset$ are left for future work.

4 Visualisations

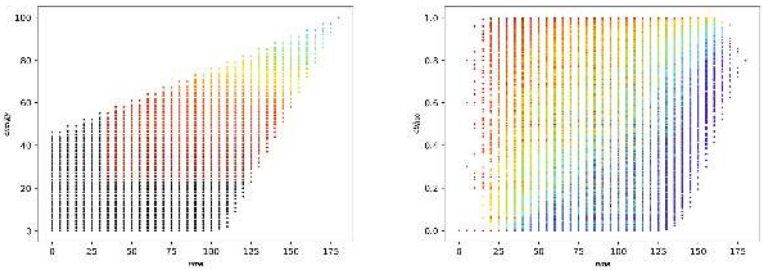
The results of a multi-objective model checking analysis are typically presented as a single (approximation of a) Pareto curve. For more than two objectives, the performance of the Pareto-optimal scheduler can be displayed in a bar chart as in Fig. 4, where the colours reflect different objectives and the groups different schedulers. The aim is to visualise the tradeoffs between the different objectives such that the user can make an informed decision about the system design or pick a scheduler for implementation. However, Pareto set visualisations alone



(a) Remaining scientific value requirement and the probabilities of the two objectives



(b) obj_{100} depending on value and energy, worst- (left)/best-case (right) time budget



(c) obj_{140} for time vs. energy (d) Value for time vs. probability

Fig. 6. Two-dimensional plots of Pareto-optimal schedulers for different quantities (Color figure online)

may not provide sufficient information, about, e.g., which objectives are aligned or conflicting (see e.g. [39] for a discussion in the non-probabilistic case). Cost bounds furthermore add an extra dimension for each cost structure. Consider the Mars rover MDP M_r and tradeoff $multi\big(obj_{100}, obj_{140}\big)$ with

$$obj_v = \mathcal{P}^{\max}_{M_r}(\langle \mathrm{C}_{time}\rangle_{\leq 175}\, B \wedge \langle \mathrm{C}_{energy}\rangle_{\leq 100}\, B \wedge \langle \mathrm{C}_{value}\rangle_{\geq v}\, B)$$

where B is the set of states where the rover has safely returned to its base. We ask for the tradeoff between performing experiments of scientific value at

least 100 before returning to base within 175 time units and maximum energy consumption of 100 units (obj_{100}) vs. achieving the same with scientific value at least 140 (obj_{140}). The Pareto curve (Fig. 5(a)) shows the tradeoff between achieving obj_{100} and obj_{140}. However, for each Pareto-optimal scheduler, our method has implicitly computed the probabilities of the two objectives for all reachable epochs as well, i.e. for all bounds on the three quantities below the ones required in the tradeoff. We visualise this information for deep insights into the behaviour of each scheduler, its robustness w.r.t. the bounds, and its preferences for certain objectives depending on the remaining budget for each quantity.

We use plots as shown in Fig. 6. They can be generated in no extra runtime or memory since all required data is already computed implicitly. We restrict to two-dimensional plots since they are easier to grasp than complex three-dimensional visualisations. In each plot, we can thus show the relationship between three different quantities: one on the x-axis (x), one on the y-axis (y), and one encoded as the colour of the points (z, where we use blue for high values, red for low values, black for probability zero, and white for unreachable epochs). Yet our example tradeoff already contains five quantities: the probability for obj_{100}, the probability for obj_{140}, the available time and energy to be spent, and the remaining scientific value to be accumulated. We thus need to project out some quantities. We do this by showing at every $\langle x, y\rangle$ coordinate the maximum or minimum value of the z quantity when ranging over *all* reachable values of the hidden *costs* at this coordinate. That is, we show a best- or worst-case situation, depending on the semantics of the respective quantities.

Out of the 30 possible combinations of quantities for our example, we showcase three to illustrate the added value of the obtained information. First, in Fig. 6(a), we plot the probabilities of the two objectives vs. the minimum scientific value that still needs to be accumulated for two different Pareto-optimal schedulers (left: $\mathfrak{S}_1$, right: $\mathfrak{S}_2$). White areas indicate that no epoch for the particular combination of probabilities is reachable from the tradeoff's bounds. These two and all other Pareto-optimal schedulers are white above the diagonal, which means that obj_{100} implies obj_{140}, i.e. the objectives are aligned. For the left scheduler, we further see that all blue-ish areas are associated to lower probabilities for both objectives. Since blue indicates higher values, this scheduler achieves only low probabilities when it still needs to make the rover accumulate a high amount of value. However, it overall achieves higher probabilities for obj_{140} at medium value requirements, whereas the right scheduler is "safer" and focuses on satisfying obj_{100}. The erratic spikes on the left occur because some probabilities are only reached after very unlikely paths.

In Fig. 6(b), we show for $\mathfrak{S}_1$ the probability to achieve obj_{100} depending on the remaining energy to be spent vs. the remaining scientific value to be accumulated. We see a white vertical line for every odd x-value; this is because, over all branches in the model, the gcd of all value costs is 2. The left plot shows the minimum probabilities over the hidden costs, i.e. we see the probability for the worst-case remaining time; the right plot shows the best-case scenario. Not surprisingly, when time is low, only a lot of energy makes it possible to reach the objective with non-zero probability.

Table 1. Runtime comparison for multi-cost single-objective queries

Benchmark instance							Interval It			Policy It.	
Case Study		$\|S\|$	$\|T\|$	r-m	$\|\mathbb{E}\|$	$\|S_{unf}\|$	UNF-dd	UNF-sp	SEQ	UNF-sp	SEQ
Service	[38]	$8{\cdot}10^4$	$2{\cdot}10^5$	1-1	162	$6{\cdot}10^6$	47	136	**10**	1945	**48**
JobSched2	[34]	349	660	2-2	503	$2{\cdot}10^4$	**<1**	**<1**	**<1**	1	**<1**
JobSched3		4584	$1{\cdot}10^5$	2-2	922	$3{\cdot}10^6$	**4**	10	**4**	26	**13**
JobSched5		$1{\cdot}10^6$	$4{\cdot}10^6$	2-2	2114	$4{\cdot}10^8$	**2944**	TO	3220	TO	TO
FireWire	[36]	776	1 411	2-2	6024	$7{\cdot}10^5$	7	8	**2**	274	**144**
FireWire		776	1 411	2-2	$1{\cdot}10^5$	$1{\cdot}10^7$	165	147	**45**	TO	**2803**
Resources	[6]	94	326	3-3	$2{\cdot}10^4$	$6{\cdot}10^5$	**<1**	18	5	46	**9**
Resources		94	326	3-3	$1{\cdot}10^7$	$6{\cdot}10^8$	TO	TO	**2693**	TO	TO
Rover		16	30	3-3	$9{\cdot}10^4$	$1{\cdot}10^6$	38	24	**4**	704	**106**
Rover		16	30	3-3	$1{\cdot}10^7$	$2{\cdot}10^8$	TO	6040	**713**	TO	TO
UAV	[23]	$1{\cdot}10^5$	$6{\cdot}10^4$	1-1	52	$4{\cdot}10^4$	**1**	**1**	**1**	**4**	27
UAV		$1{\cdot}10^5$	$6{\cdot}10^4$	1-1	102	$4{\cdot}10^5$	7	16	**2**	72	**46**
Wlan3	[36]	$1{\cdot}10^5$	$2{\cdot}10^5$	1-1	82	$3{\cdot}10^6$	9	63	**8**	**126**	800
Wlan3		$1{\cdot}10^5$	$2{\cdot}10^5$	1-1	202	$1{\cdot}10^7$	820	293	**14**	**848**	2155
Wlan6		$5{\cdot}10^6$	$1{\cdot}10^7$	1-1	82	$2{\cdot}10^7$	**12**	363	989	**643**	TO
Wlan6		$5{\cdot}10^6$	$1{\cdot}10^7$	1-1	202	$6{\cdot}10^8$	2292	TO	**1399**	TO	TO

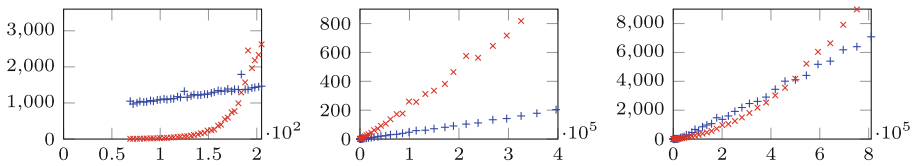
Table 2. Runtime comparison for multi-cost multi-objective queries

Benchmark instance							Interval It		Policy It.	
Case Study	$\|S\|$	$\|T\|$	ℓ-r-m	$\|\mathbb{E}\|$	$\#\boldsymbol{w}$	$\|S_{unf}\|$	UNF-sp	SEQ	UNF-sp	SEQ
Service	$8{\cdot}10^4$	$2{\cdot}10^5$	2-1-2	162	34	$6{\cdot}10^6$	1918	**543**	TO	**4679**
JobSched2	349	660	2-4-4	$4{\cdot}10^4$	2	$1{\cdot}10^5$	**3**	54	**15**	183
JobSched3	4584	$1{\cdot}10^5$	2-4-4	$1{\cdot}10^6$	35	$2{\cdot}10^6$	**96**	TO	**6239**	TO
JobSched5	$1{\cdot}10^6$	$4{\cdot}10^6$	2-4-4	$3{\cdot}10^5$	?	?	TO	TO	TO	TO
FireWire	776	1 411	2-2-2	6 024	3	$7{\cdot}10^5$	32	**17**	TO	**1159**
FireWire	776	1 411	2-2-2	$1{\cdot}10^5$	2	$1{\cdot}10^7$	863	**225**	TO	TO
Resources	94	326	2-3-4	$2{\cdot}10^5$	3	$6{\cdot}10^5$	25	**16**	2047	**52**
Resources	94	326	2-3-4	$1{\cdot}10^8$	?	?	TO	TO	TO	TO
Rover	16	30	2-3-3	$9{\cdot}10^5$	7	$1{\cdot}10^6$	177	**39**	5817	**3328**
Rover	16	30	2-3-3	$1{\cdot}10^8$	7	$2{\cdot}10^8$	TO	**5785**	TO	TO
UAV	$1{\cdot}10^5$	$6{\cdot}10^4$	2-1-2	52	18	$4{\cdot}10^4$	**2**	24	**102**	1098
UAV	$1{\cdot}10^5$	$6{\cdot}10^4$	2-1-2	102	22	$4{\cdot}10^5$	70	**39**	**2282**	3062
Wlan3	$1{\cdot}10^5$	$2{\cdot}10^5$	3-1-2	82	68	$3{\cdot}10^6$	5239	**2231**	TO	TO
Wlan3	$1{\cdot}10^5$	$2{\cdot}10^5$	3-1-2	202	4	$1{\cdot}10^7$	1769	**185**	TO	TO
Wlan6	$5{\cdot}10^6$	$1{\cdot}10^7$	3-1-2	82	?	$2{\cdot}10^7$	TO	TO	TO	TO

Finally, Fig. 6(c) shows the probability for obj_{140} depending on available time and energy for $\mathfrak{S}_2$. We plot the minimum probability over the hidden scientific value requirement, i.e. a worst-case view. The plot shows that time is of little use in case of low remaining energy, but it helps significantly when there is sufficient energy, too. In Fig. 6(d), we depict for the same scheduler the minimum remaining scientific value (z) under which a certain probability for obj_{100} can be achieved (y), given a certain remaining time budget (x). The upper left corner shows that a high probability in little time is only achievable if we need to collect little more value; the value requirement gradually relaxes as we aim for lower probabilities or have more time.

5 Experiments

Implementation. We implemented the presented approach into STORM [20] v1.2, and available via [19]. The implementation computes extremal probabilities for single-objective multi-cost bounded queries, as well as Pareto curves for the multi-objective case. We consider the *sparse* engine of STORM, i.e., explicit data structures such as sparse matrices. For single-cost bounded properties, this has already been addressed in [34]. For the computation of expected cost (Lines 10 to 12 of Algorithm 1) we employ interval iteration with finite precision floats as well as policy iteration with infinite precision rationals. The expected costs (lines 10 to 12 of Algorithm 1) are computed either numerically (via interval iteration over finite precision floats) or exactly (via policy iteration over infinite precision rationals). To reduce the memory consumption, the analysis result of an epoch model M_f^e is erased as soon as possible.



(a) Wlan6 (single-obj.) (b) Rover (multi-obj.) (c) Resources (multi-obj.)

Fig. 7. Runtime (y-axis) of SEQ (+) and UNF (×) for increasing cost bounds (x-axis)

Set-up & reproduction. We evaluate the approach on wide range of case studies, available in the artefact [30]. The models are given in PRISM's [37] guarded command language. For each case study we consider single- and multi-objective queries that yield non-trivial results, i.e., probabilities strictly between zero and one. We compare the naive unfolding approach (UNF) as in Sect. 3.1 with the sequential approach (SEQ) as in Sect. 3.2. The unfolding of the model is applied on the PRISM language level, by considering a parallel composition with cost counting structures. On the unfolded model we apply the algorithms for

unbounded reachability as available in Storm. We considered precision $\eta = 10^{-4}$ for the Pareto curve approximation and precision $\varepsilon = 10^{-6}$ for interval iteration. We increased the precision for single epoch models as in Theorem 2.

We ran our experiments on a single core (2 GHz) of a HP BL685C G7 system with 192 GB of memory. We stopped each experiment after a time limit of 2 hours. For experiments that completed within the time limit, we observed a memory consumption of up to 36 GB for UNF and up to 5 GB for SEQ.

A binary equivalent to the binary we used for the experiments is available in the artefact [30]. The binary has been tested in the artefact evaluation VM [31]. For other configurations, Storm should be recompiled using the sources [19].

Details on reproduction of the tables, as well as details on how to analyse multi-cost bounded properties using Storm in general can be found in the readme, enclosed in the artefact.

Experimental Results. Tables 1 and 2 show results for single- and multi-objective queries, respectively. The first columns yield the number of states and transitions of the original MDP, then for the query, the number of bounds m, the number of *different* cost structures r, and the number of reachable cost epochs (reflecting the magnitude of the bound values). $|S_{unf}|$ denotes the number of reachable states in the unfolding. For multi-objective queries, we additionally give the number of objectives and the number of analysed weight vectors $\boldsymbol{w}$. The remaining columns depict the runtimes of the different approaches in seconds. For UNF, we considered both the sparse (sp) and symbolic (dd) engine of Storm. The symbolic engine neither supports multi-objective model checking nor exact policy iteration.

On the majority of benchmarks, SEQ performs better than UNF. Typically, SEQ is less sensitive to increases in the magnitude of the cost bounds, as illustrated in Fig. 7. For three benchmark and query instances, we plot the runtime of both approaches against different numbers $|\mathbb{E}|$ of reachable epochs. While for small cost bounds, UNF is sometimes even faster compared to SEQ, SEQ scales better with increasing $|\mathbb{E}|$. It is not surprising that SEQ scales better, ultimately, the increased state space and the accompanying memory consumption in UNF is a bottleneck. The most important reason that UNF performs better for some (smaller) cost bounds is the induced overhead of checking the full epoch. In particular, the epoch contains (often many) states that are not reachable from the initial state (in the unfolding).

6 Conclusion

Many real-world planning problems consider several limited resources and contain tradeoffs. This paper present a practically efficient approach to analyse these problems. It has been implemented in the Storm model checker and shows significant performance benefits. The algorithm implicitly computes a large amount of information that is hidden in the standard plots of Pareto curves shown to visualise the results of a multi-objective analysis. We have developed a new set of

visualisations that exploit all the available data to provide new and clear insights to decision makers even for problems with many objectives and cost dimensions.

Data Availability Statement. The datasets analysed during the current study, and the binary used for the analysis, are available in the figshare repository [30]. Source code matching the binary is available in [19].

References

1. The International Probabilistic Planning Competition, http://www.icaps-conference.org/index.php/Main/Competitions
2. Andova, S., Hermanns, H., Katoen, J.-P.: Discrete-time rewards model-checked. In: Larsen, K.G., Niebert, P. (eds.) FORMATS 2003. LNCS, vol. 2791, pp. 88–104. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-40903-8_8
3. Baier, C., Daum, M., Dubslaff, C., Klein, J., Klüppelholz, S.: Energy-utility quantiles. In: Badger, J.M., Rozier, K.Y. (eds.) NFM 2014. LNCS, vol. 8430, pp. 285–299. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06200-6_24
4. Baier, C., Klein, J., Klüppelholz, S., Wunderlich, S.: Maximizing the conditional expected reward for reaching the goal. In: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol. 10206, pp. 269–285. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54580-5_16
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: Majumdar, R., Kunčak, V. (eds.) CAV 2017, Part I. LNCS, vol. 10426, pp. 160–180. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63387-9_8
6. Barrett, L., Narayanan, S.: Learning all optimal policies with multiple criteria. In: ICML. AICPS, vol. 307, pp. 41–47. ACM (2008)
7. Berthon, R., Randour, M., Raskin, J.F.: 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)
8. Brázdil, T., Brozek, V., Chatterjee, K., Forejt, V., Kucera, A.: Two views on multiple mean-payoff objectives in Markov decision processes. LMCS **10**(1) (2014)
9. Brázdil, T., Chatterjee, K., Chmelík, M., Forejt, V., Křetínský, J., Kwiatkowska, M., Parker, D., Ujma, M.: Verification of Markov decision processes using learning algorithms. In: Cassez, F., Raskin, J.-F. (eds.) ATVA 2014. LNCS, vol. 8837, pp. 98–114. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11936-6_8
10. Brázdil, T., Chatterjee, K., Forejt, V., Kucera, A.: Trading performance for stability in Markov decision processes. J. Comput. Syst. Sci. **84**, 144–170 (2017)
11. Bresina, J.L., Jónsson, A.K., Morris, P.H., Rajan, K.: Activity planning for the mars exploration rovers. In: ICAPS, pp. 40–49. AAAI (2005)
12. Bryce, D., Cushing, W., Kambhampati, S.: Probabilistic planning is multi-objective. Technical report, Arizona State Univ., CSE (2007)
13. Cao, Z., Guo, H., Zhang, J., Oliehoek, F.A., Fastenrath, U.: Maximizing the probability of arriving on time: a practical q-learning method. In: AAAI, pp. 4481–4487. AAAI Press (2017)
14. Chatterjee, K., Chmelik, M., Gupta, R., Kanodia, A.: Optimal cost almost-sure reachability in POMDPs. Artif. Intell. **234**, 26–48 (2016)

15. Chatterjee, K., Majumdar, R., Henzinger, T.A.: Markov decision processes with multiple objectives. In: Durand, B., Thomas, W. (eds.) STACS 2006. LNCS, vol. 3884, pp. 325–336. Springer, Heidelberg (2006). https://doi.org/10.1007/11672142_26
16. Chen, T., Forejt, V., Kwiatkowska, M., Simaitis, A., Wiltsche, C.: On stochastic games with multiple objectives. In: Chatterjee, K., Sgall, J. (eds.) MFCS 2013. LNCS, vol. 8087, pp. 266–277. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40313-2_25
17. Cheng, L., Subrahmanian, E., Westerberg, A.W.: Multiobjective decision processes under uncertainty: applications, problem formulations, and solution strategies. Ind. Eng. Chem. Res. **44**(8), 2405–2415 (2005)
18. Christman, A., Cassamano, J.: Maximizing the probability of arriving on time. In: Dudin, A., De Turck, K. (eds.) ASMTA 2013. LNCS, vol. 7984, pp. 142–157. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39408-9_11
19. Dehnert, C., Junges, S., Katoen, J.P., Quatmann, T., Volk, M.: Storm source files. zenodo (2018), https://doi.org/10.5281/zenodo.1181896
20. Dehnert, C., Junges, S., Katoen, J.-P., Volk, M.: A Storm is coming: a modern probabilistic model checker. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017, Part II. LNCS, vol. 10427, pp. 592–600. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63390-9_31
21. Eastwood, R., Alexander, R., Kelly, T.: Safe multi-objective planning with a posteriori preferences. In: HASE, pp. 78–85. IEEE Computer Society (2016)
22. Etessami, K., Kwiatkowska, M., Vardi, M.Y., Yannakakis, M.: Multi-objective model checking of Markov decision processes. LMCS **4**(4) (2008)
23. Feng, L., Wiltsche, C., Humphrey, L., Topcu, U.: Controller synthesis for autonomous systems interacting with human operators. In: ICCPS, pp. 70–79. ACM (2015)
24. Forejt, V., Kwiatkowska, M., Parker, D.: Pareto curves for probabilistic model checking. In: Chakraborty, S., Mukund, M. (eds.) ATVA 2012. LNCS, pp. 317–332. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33386-6_25
25. Forejt, V., Kwiatkowska, M., Norman, G., Parker, D.: Automated verification techniques for probabilistic systems. In: Bernardo, M., Issarny, V. (eds.) SFM 2011. LNCS, vol. 6659, pp. 53–113. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21455-4_3
26. Haddad, S., Monmege, B.: Reachability in MDPs: refining convergence of value iteration. In: Ouaknine, J., Potapov, I., Worrell, J. (eds.) RP 2014. LNCS, vol. 8762, pp. 125–137. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11439-2_10
27. Hahn, E.M., Hartmanns, A.: A comparison of time- and reward-bounded probabilistic model checking techniques. In: Fränzle, M., Kapur, D., Zhan, N. (eds.) SETTA 2016. LNCS, vol. 9984, pp. 85–100. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47677-3_6
28. Hahn, E.M., Hashemi, V., Hermanns, H., Lahijanian, M., Turrini, A.: Multi-objective robust strategy synthesis for interval Markov decision processes. In: Bertrand, N., Bortolussi, L. (eds.) QEST 2017. LNCS, vol. 10503, pp. 207–223. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66335-7_13
29. Hartmanns, A., Hermanns, H.: The Modest Toolset: an integrated environment for quantitative modelling and verification. In: Ábrahám, E., Havelund, K. (eds.) TACAS 2014. LNCS, vol. 8413, pp. 593–598. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54862-8_51

30. Hartmanns, A., Junges, S., Katoen, J.P., Quatmann, T.: Evaluated artefact for this paper. figshare (2018), https://doi.org/10.6084/m9.figshare.5907349.v1
31. Hartmanns, A., Wendler, P.: Artefact vm. figshare (2018), https://doi.org/10.6084/m9.figshare.5896615
32. Hou, P., Yeoh, W., Varakantham, P.: Revisiting risk-sensitive MDPs: new algorithms and results. In: ICAPS. AAAI (2014)
33. Junges, S., Jansen, N., Dehnert, C., Topcu, U., Katoen, J.-P.: Safety-constrained reinforcement learning for MDPs. In: Chechik, M., Raskin, J.-F. (eds.) TACAS 2016. LNCS, vol. 9636, pp. 130–146. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49674-9_8
34. Klein, J., Baier, C., Chrszon, P., Daum, M., Dubslaff, C., Klüppelholz, S., Märcker, S., Müller, D.: Advances in probabilistic model checking with PRISM: variable reordering, quantiles and weak deterministic Büchi automata. In: STTT, pp. 1–16 (2017)
35. Kolobov, A., Mausam, Weld, D.S.: A theory of goal-oriented MDPs with dead ends. In: UAI, pp. 438–447. AUAI Press (2012)
36. Kwiatkowska, M., Norman, G., Parker, D.: The PRISM benchmark suite. In: QEST, pp. 203–204. IEEE CS Press (2012)
37. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 585–591. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_47
38. 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)
39. Lankaites Pinheiro, R., Landa-Silva, D., Atkin, J.: A technique based on trade-off maps to visualise and analyse relationships between objectives in optimisation problems. J. Multi-Criteria Decis. Anal. **24**(1–2), 37–56 (2017)
40. Laroussinie, F., Sproston, J.: Model checking durational probabilistic systems. In: Sassone, V. (ed.) FoSSaCS 2005. LNCS, vol. 3441, pp. 140–154. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31982-5_9
41. Puterman, M.L.: Markov Decision Processes. Wiley, New York (1994)
42. Quatmann, T., Junges, S., Katoen, J.-P.: Markov automata with multiple objectives. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017, Part I. LNCS, vol. 10426, pp. 140–159. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63387-9_7
43. Randour, M., Raskin, J.F., Sankur, O.: Percentile queries in multi-dimensional Markov decision processes. FMSD **50**(2–3), 207–248 (2017)
44. Roijers, D.M., Vamplew, P., Whiteson, S., Dazeley, R.: A survey of multi-objective sequential decision-making. J. Artif. Intell. Res. **48**, 67–113 (2013)
45. Steinmetz, M., Hoffmann, J., Buffet, O.: Goal probability analysis in probabilistic planning: exploring and enhancing the state of the art. J. Artif. Intell. Res. **57**, 229–271 (2016)
46. Teichteil-Königsbuch, F.: Stochastic safest and shortest path problems. In: AAAI. AAAI Press (2012)
47. Vamplew, P., Dazeley, R., Berry, A., Issabekov, R., Dekker, E.: Empirical evaluation methods for multiobjective reinforcement learning algorithms. Mach. Learn. **84**(1–2), 51–80 (2011)
48. Yu, S.X., Lin, Y., Yan, P.: Optimization models for the first arrival target distribution function in discrete time. J. Math. Anal. Appl. **225**(1), 193–223 (1998)

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.