Multi-objective Optimization of Long-run Average and Total Rewards

Tim Quatmann[1](✉) and Joost-Pieter Katoen[1]

RWTH Aachen University, Aachen, Germany
tim.quatmann@cs.rwth-aachen.de

Abstract This paper presents an efficient procedure for multi-objective model checking of long-run average reward (aka: mean pay-off) and total reward objectives as well as their combination. We consider this for Markov automata, a compositional model that captures both traditional Markov decision processes (MDPs) as well as a continuous-time variant thereof. The crux of our procedure is a generalization of Forejt *et al.*'s approach for total rewards on MDPs to arbitrary combinations of long-run and total reward objectives on Markov automata. Experiments with a prototypical implementation on top of the Storm model checker show encouraging results for both model types and indicate a substantial improved performance over existing multi-objective long-run MDP model checking based on linear programming.

1 Introduction

MDP model checking In various applications, multiple decision criteria and uncertainty frequently co-occur. Stochastic decision processes for which the objective is to achieve multiple—possibly partly conflicting—objectives occur in various fields. These include operations research, economics, planning in AI, and game theory, to mention a few. This has stimulated model checking of Markov decision processes (MDPs) [46], a prominent model in decision making under uncertainty, against multiple objectives. This development enlarges the rich plethora of automated MDP verification algorithms against single objectives [7].

Multi-objective MDP Various types of objectives known from conventional—single-objective—model checking have been lifted to the multi-objective case. These objectives range over ω-regular specifications including LTL [26,27], expected (discounted and non-discounted) total rewards [21,27,28,52,22], step-bounded and reward-bounded reachability probabilities [28,35], and—most relevant for this work—*expected long-run average (LRA) rewards* [18,11,20], also known as mean pay-offs. For the latter, all current approaches build upon linear programming (LP) which yields a theoretical time-complexity polynomial in the model size. However, in practice, LP-based methods are often outperformed by approaches based on value- or strategy iteration [28,1,42]. The LP-based approach of [27] and the iterative approach of [28] are both implemented in PRISM [45] and Storm [40]. The LP formulation of [11,20] is implemented in MultiGain [12], an extension of PRISM for multi-objective LRA rewards.

© The Author(s) 2021
J. F. Groote and K. G. Larsen (Eds.): TACAS 2021, LNCS 12651, pp. 230–249, 2021.
https://doi.org/10.1007/978-3-030-72016-2_13

Contributions of this paper We present a computationally efficient procedure for multi-objective model checking of LRA reward and total reward objectives as well as their mixture. The crux of our procedure is *a generalization* of Forejt *et al.*'s iterative approach [28] for total rewards on MDPs *to expected LRA reward objectives*. In fact, our approach supports the arbitrary *mixtures* of expected LRA and total reward objectives. To our knowledge, such mixtures have not been considered so far. Experiments on various benchmarks using a prototypical implementation in Storm indicate that this generalized iterative algorithm outperforms the LP approach implemented in MultiGain.

In addition, we extend this approach towards *Markov automata* (MA) [25,23], a continuous-time variant of MDP that is amenable to compositional modeling. This model is well-suited, among others, to provide a formal semantics for dynamic fault trees and generalized stochastic Petri nets [24]. Our multi-objective LRA approach for MA builds upon the value-iteration approach for single-objective expected LRA rewards on MA [17] which—on practical models—outperforms the LP-based approach of [30]. To the best of our knowledge, this is the *first multi-objective expected LRA reward approach for MA*. Experimental results on MA benchmarks show that the treatment of a continuous-time variant of LRA comes at almost no time penalty compared to the MDP setting.

Other related work Mixtures of various other objectives have been considered for MDPs. This includes conditional expectations or ratios of reward functions [5,4]. [31] considers LTL formulae with probability thresholds while maximizing an expected LRA reward. [35,41] address multi-objective quantiles on reachability properties while [50,20] consider multi-objective combinations of percentile queries on MDP and LRA objectives. [6] treats resilient systems ensuring constraints on the repair mechanism while maximizing the expected LRA reward when being operational. The trade-off between expected LRA rewards and their variance is analyzed in [13]. [33] studies multiple objectives on interval MDP, where transition probabilities can be specified as intervals in cases where the concrete probabilities are unknown. Multiple LRA reward objectives for *stochastic games* have been treated using LP [19] and value iteration over convex sets [8,9]; the latter is included in PRISM-games [44,43]. These approaches can also be applied to MDPs when viewed as one-player stochastic games. Algorithms for single-objective model checking of MA deal with objectives such as expected total rewards, time-bounded reachability probabilities, and expected long-run average rewards [38,29,30,15]. The only multi-objective approach for MA so far [47] shows that any method for multi-objective MDP can be applied on (a discretized version of) an MA for queries involving unbounded or time-bounded reachability probabilities and expected total rewards, but no long-run average rewards.

2 Preliminaries

The set of *probability distributions* over a finite set Ω is given by $Dist(\Omega) = \{\mu\colon \Omega \mapsto [0,1] \mid \sum_{\omega\in\Omega} \mu(\omega) = 1\}$. For a distribution $\mu \in Dist(\Omega)$ we let $supp(\mu) = \{\omega \in \Omega \mid \mu(\omega) > 0\}$ denote its support. μ is *Dirac* if $|supp(\mu)| = 1$.

Let $\mathbb{R}_{\geq 0} = \{x \in \mathbb{R} \mid x \geq 0\}$, $\mathbb{R}_{>0} = \{x \in \mathbb{R} \mid x > 0\}$, and $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ denote the non-negative, positive, and extended real numbers, respectively. For a point $\mathbf{p} = \langle p_1, \ldots, p_\ell \rangle \in \mathbb{R}^\ell$, $\ell \in \mathbb{N}$ and $i \in \{1, \ldots, \ell\}$ we write $\mathbf{p}[\![i]\!]$ for its i^{th} entry p_i. For $\mathbf{p}, \mathbf{q} \in \mathbb{R}^\ell$ let $\mathbf{p} \cdot \mathbf{q}$ denote the dot product. We further write $\mathbf{p} \leq \mathbf{q}$ iff $\forall\, i\colon\ \mathbf{p}[\![i]\!] \leq \mathbf{q}[\![i]\!]$ and $\mathbf{p} \lneq \mathbf{q}$ iff $\mathbf{p} \leq \mathbf{q} \wedge \mathbf{p} \neq \mathbf{q}$. The *closure* of a set $P \subseteq \mathbb{R}^\ell$ is the union of P and its boundary, denoted by $cl(P)$. The *convex hull* of P is given by $conv(P) = \left\{ \sum_{i=1}^{\ell} \mu(i) \cdot \mathbf{p}_i \mid \mu \in Dist(\{1, \ldots, \ell\}), \mathbf{p}_1, \ldots, \mathbf{p}_\ell \in P \right\}$. The *downward convex hull* of P is given by $dwconv(P) = \left\{ \mathbf{q} \in \mathbb{R}^\ell \mid \exists\, \mathbf{p} \in conv(P)\colon\ \mathbf{q} \leq \mathbf{p} \right\}$.

2.1 Markov Automata

Markov automata (MA) [25,23] provide an expressive formalism that allows one to model exponentially distributed delays, nondeterminism, probabilistic branching, and instantaneous (undelayed) transitions.

Definition 1. *A* Markov Automaton *is a tuple $\mathcal{M} = \langle S, Act, \Delta, \mathbf{P} \rangle$ where S is a finite set of states, Act is a finite set of actions, $\Delta\colon S \to \mathbb{R}_{>0} \cup 2^{Act}$ is a transition function assigning exit rates to Markovian states $MS^{\mathcal{M}} = \{s \in S \mid \Delta(s) \in \mathbb{R}_{>0}\}$ and sets of enabled actions to probabilistic states $PS^{\mathcal{M}} = \{s \in S \mid \Delta(s) \subseteq Act\}$, and $\mathbf{P}\colon MS^{\mathcal{M}} \cup SA^{\mathcal{M}} \to Dist(S)$ with $SA^{\mathcal{M}} = \{\langle s, \alpha \rangle \in PS \times Act \mid \alpha \in \Delta(s)\}$ is a probability function that assigns a distribution over possible successor states for each Markovian state and enabled state-action pair.*

Let $\mathcal{M} = \langle S, Act, \Delta, \mathbf{P} \rangle$ be an MA. If $\mathcal{M}$ is clear from the context, we may omit the superscript from $MS^{\mathcal{M}}$, $PS^{\mathcal{M}}$, $SA^{\mathcal{M}}$, and further notations introduced below. Intuitively, the time $\mathcal{M}$ stays in a Markovian state $s \in MS$ is governed by an *exponential distribution* with rate $\Delta(s) \in \mathbb{R}_{>0}$, i.e., the probability to take a transition from s within $t \in \mathbb{R}_{\geq 0}$ time units is $1 - e^{-\Delta(s) \cdot t}$. Upon taking a transition, a successor state $s' \in S$ is drawn from the distribution $\mathbf{P}(s)$, i.e., $\mathbf{P}(s)(s')$ is the probability that the transition leads to $s' \in S$. For probabilistic states $\hat{s} \in PS$, an enabled action $\alpha \in \Delta(\hat{s})$ has to be picked and a successor state is drawn from $\mathbf{P}(\langle \hat{s}, \alpha \rangle)$ (without any delay). Nondeterminism is thus only possible at probabilistic states. We assume deadlock free MA, i.e., $\forall\, s \in PS^{\mathcal{M}}\colon\ \Delta(s) \neq \emptyset$.

Remark 1. To enable more flexible modeling such as parallel compositions, the literature (e.g., [25,30]) often considers a more liberal variant of MA where (i) different successor distributions can be assigned to the same state-action pair and (ii) states can be both, Markovian *and* probabilistic. MAs as in Definition 1—also known as closed MA—are equally expressive: they can be constructed via action renaming and by applying the so-called *maximal progress assumption* [25].

An *infinite path* in $\mathcal{M}$ is a sequence $\pi = s_0 \kappa_1 s_1 \kappa_2 \ldots$ where for each $i \geq 0$ either $s_i \in MS$, $\kappa_{i+1} \in \mathbb{R}_{\geq 0}$, and $\mathbf{P}(s_i)(s_{i+1}) > 0$ or $s_i \in PS$, $\kappa_{i+1} \in \Delta(s_i)$, and $\mathbf{P}(\langle s_i, \kappa_{i+1} \rangle)(s_{i+1}) > 0$. Intuitively, if s_i is Markovian, $\kappa_{i+1} \in \mathbb{R}_{\geq 0}$ reflects the time we have stayed in s_i until transitioning to s_{i+1}. If s_i is probabilistic, $\kappa_{i+1} \in Act$ is the performed action via which we transition to s_{i+1}. A finite path

$\hat{\pi} = s_0\kappa_1 s_1\kappa_2 \ldots \kappa_n s_n$ is a finite prefix of an infinite path π. We set $last(\hat{\pi}) = s_n$ and $|\hat{\pi}| = n$ for finite $\hat{\pi}$ and $|\pi| = \infty$ for infinite π. For (finite or infinite) path $\bar{\pi} = s_0\kappa_1 s_1\kappa_2 \ldots$ let $dur(\bar{\pi}) = \sum_{i=1}^{|\bar{\pi}|} dur(\kappa_i)$ be the total duration of $\bar{\pi}$ where $dur(\kappa) = \kappa$ if $\kappa \in \mathbb{R}_{\geq 0}$ and 0 otherwise. If $\bar{\pi}$ is infinite and $dur(\bar{\pi}) < \infty$, the path is called *Zeno*. For $k \in \mathbb{N}$ with $k \leq |\bar{\pi}|$ we let $prefix_{steps}(\bar{\pi}, k)$ denote the unique prefix π' of $\bar{\pi}$ with $|\pi'| = k$ and for $t \in \mathbb{R}_{\geq 0}$ we let $prefix_{time}(\bar{\pi}, t)$ denote the largest prefix of $\bar{\pi}$ with total duration at most t. The sets of infinite and finite paths of $\mathcal{M}$ are given by $Paths^{\mathcal{M}}_{\text{inf}}$ and $Paths^{\mathcal{M}}_{\text{fin}}$, respectively.

A *component* of $\mathcal{M}$ is a set $C \subseteq MS \cup SA$. We set $states(C) = (C \cap MS) \cup \{s \in PS \mid \exists\, \alpha\colon \langle s, \alpha\rangle \in C\}$. C is *closed* if $\forall\, c \in C\colon supp(\mathbf{P}(c)) \subseteq states(C)$ and *connected* if for all $s, s' \in states(C)$ there is $s_0\kappa_1 \ldots \kappa_n s_n \in Paths_{\text{fin}}$ with $s = s_0$, $s' = s_n$, and for each $i \geq 0$ either $s_i \in C \cap MS$ or $\langle s_i, \kappa_{i+1}\rangle \in C \cap SA$. An *end component (EC)* is a closed and connected component. An EC is *maximal* if it is not a proper subset of another EC. $MECS(\mathcal{M})$ denotes the maximal ECs of $\mathcal{M}$. For an EC C let $exits(C) = \big\{\langle s, \alpha\rangle \in SA^{\mathcal{M}} \mid s \in states(C) \text{ and } \langle s, \alpha\rangle \notin C\big\}$.

Definition 2. *The* sub-MA *of $\mathcal{M}$ induced by a closed component C is given by $\mathcal{M}[\![C]\!] = \langle states(C), Act, \Delta_C, \mathbf{P}_C\rangle$ where $\Delta_C(s) = \Delta(s)$ if $s \in C \cap MS^{\mathcal{M}}$ and otherwise $\Delta_C(s) = \{\alpha \in \Delta(s) \mid \langle s, \alpha\rangle \in C\}$, and $\mathbf{P}_C$ is the restriction of $\mathbf{P}$ to C.*

A *strategy* for $\mathcal{M}$ resolves the nondeterminism at probabilistic states by providing probability distributions over enabled actions based on the execution history.

Definition 3. *A (general)* strategy *for MA $\mathcal{M} = \langle S, Act, \Delta, \mathbf{P}\rangle$ is a function $\sigma\colon Paths_{\text{fin}} \to Dist(Act) \cup \{\tau\}$ such that for $\hat{\pi} \in Paths_{\text{fin}}$ we have $\sigma(\hat{\pi}) \in Dist(\Delta(last(\hat{\pi})))$ if $last(\hat{\pi}) \in PS$ and $\sigma(\hat{\pi}) = \tau$ otherwise.*

A strategy σ is called *memoryless* if the choice only depends on the current state, i.e., $\forall\, \hat{\pi}, \hat{\pi}' \in Paths_{\text{fin}}\colon last(\hat{\pi}) = last(\hat{\pi}')$ implies $\sigma(\hat{\pi}) = \sigma(\hat{\pi}')$. If all assigned distributions are Dirac, σ is called *deterministic*. Let $\Sigma^{\mathcal{M}}$ and $\Sigma^{\mathcal{M}}_{\text{md}}$ denote the set of general and memoryless deterministic strategies of $\mathcal{M}$, respectively. For simplicity, we often interpret $\sigma \in \Sigma^{\mathcal{M}}_{\text{md}}$ as a function $\sigma\colon S \to Act \cup \{\tau\}$. The *induced sub-MA* for $\sigma \in \Sigma^{\mathcal{M}}_{\text{md}}$ is given by $\mathcal{M}[\![\, MS \cup \{\langle s, \sigma(s)\rangle \mid s \in PS\}\,]\!]$. Strategy $\sigma \in \Sigma^{\mathcal{M}}$ and initial state $s_I \in S$ define a *probability measure* $\Pr^{\mathcal{M}, s_I}_{\sigma}$ that assigns probabilities to sets of infinite paths [38]. The expected value of $f\colon Paths_{\text{inf}} \to \bar{\mathbb{R}}$ is given by the Lebesque integral $\mathrm{Ex}^{\mathcal{M}, s_I}_{\sigma}(f) = \int_{\pi \in Paths_{\text{inf}}} f(\pi)\, \mathrm{d}\Pr^{\mathcal{M}, s_I}_{\sigma}(\pi)$.

2.2 Reward-based Objectives

MA can be equipped with *rewards* to model various quantities like, e.g., energy consumption or the number of produced units. We distinguish between *transition* rewards $\mathcal{R}_{\text{trans}}\colon MS \cup SA \times S \to \mathbb{R}$ that are collected when transitioning from one state to another and *state* rewards $\mathcal{R}_{\text{state}}\colon S \to \mathbb{R}$ that are collected over time, i.e., staying in state s for t time units yields a reward of $\mathcal{R}_{\text{state}}(s) \cdot t$. Since no time passes in probabilistic states, state rewards $\mathcal{R}_{\text{state}}(s)$ for $s \in PS$ are not relevant. A reward assignment combines the two notions.

Definition 4. *A* reward assignment *for MA* $\mathcal{M}$ *and* $\mathcal{R}_{\text{state}}, \mathcal{R}_{\text{trans}}$ *as above is a function* $\mathcal{R}\colon (MS \times \mathbb{R}_{\geq 0}) \cup SA \times S \to \mathbb{R}$ *with*

$$\mathcal{R}(\langle s, \kappa \rangle, s') = \begin{cases} \mathcal{R}_{\text{state}}(s) \cdot \kappa + \mathcal{R}_{\text{trans}}(s, s') & \text{if } s \in MS, \kappa \in \mathbb{R}_{\geq 0} \\ \mathcal{R}_{\text{trans}}(\langle s, \kappa \rangle, s') & \text{if } s \in PS, \kappa \in \Delta(s). \end{cases}$$

We fix a reward assignment $\mathcal{R}$ for $\mathcal{M}$. $\mathcal{R}$ can also be applied to any sub-MA $\mathcal{M}[\![C]\!]$ of $\mathcal{M}$ in a straightforward way. For a component $C \subseteq MS \cup SA$ we write $\mathcal{R}(C) \geq 0$ if all rewards assigned within C are non-negative, formally $\forall\, \langle s, \kappa \rangle \in (C \cap SA) \cup ((C \cap MS) \times \mathbb{R}_{\geq 0})\colon \forall\, s' \in \mathit{states}(C)\colon \mathcal{R}(\langle C, \kappa \rangle, s') \geq 0$. The shortcuts $\mathcal{R}(C) \leq 0$ and $\mathcal{R}(C) = 0$ are similar. The reward of a finite path $\hat{\pi} = s_0 \kappa_1 s_1 \kappa_2 \ldots \kappa_n s_n$ is denoted by $\mathcal{R}(\hat{\pi}) = \sum_{i=1}^{|\hat{\pi}|} \mathcal{R}(\langle s_{i-1}, \kappa_i \rangle, s_i)$.

Definition 5. *The* total reward objective *for reward assignment* $\mathcal{R}$ *is given by* $\mathit{tot}(\mathcal{R})\colon \mathit{Paths}_{\text{inf}} \to \bar{\mathbb{R}}$ *with* $\mathit{tot}(\mathcal{R})(\pi) = \limsup_{k \to \infty} \mathcal{R}(\mathit{prefix}_{\mathit{steps}}(\pi, k))$.

Definition 6. *The* long-run average (LRA) reward objective *for* $\mathcal{R}$ *is given by* $\mathit{lra}(\mathcal{R})\colon \mathit{Paths}_{\text{inf}} \to \bar{\mathbb{R}}$ *with* $\mathit{lra}(\mathcal{R})(\pi) = \limsup_{t \to \infty} \frac{1}{t} \cdot \mathcal{R}(\mathit{prefix}_{\mathit{time}}(\pi, t))$.

Sect. 4 considers assumptions under which the limit in both definitions can be attained, i.e., lim sup can be replaced by lim. The incorporation of other objectives such as *reachability probabilities* are discussed in Remark 3.

2.3 Markov Decision Processes

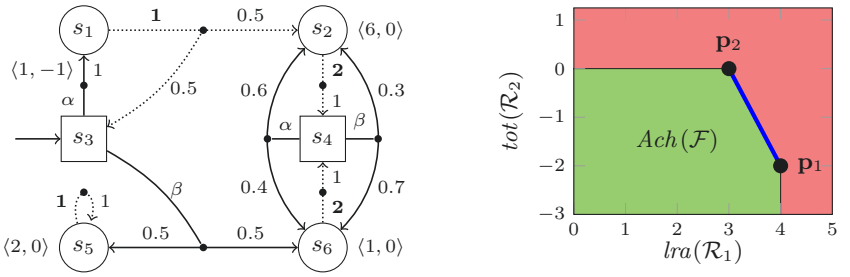
A *Markov Decision Process (MDP)* $\mathcal{M}$ is an MA with only probabilistic states, i.e., $MS^{\mathcal{M}} = \emptyset$. All notions above also apply to MDP. However, since all paths of an MDP have duration 0, there is no timing information available. For MDP, we therefore usually consider *steps* instead of time. In particular, for reward assignment $\mathcal{R}$ we consider $\mathit{lra}_{\mathit{steps}}(\mathcal{R})$ instead of $\mathit{lra}(\mathcal{R})$, where $\mathit{lra}_{\mathit{steps}}(\mathcal{R})(\pi) = \limsup_{k \to \infty} \frac{1}{k} \cdot \mathcal{R}(\mathit{prefix}_{\mathit{steps}}(\pi, k))$. Below, we focus on MA. Applying our results to step-based LRA rewards on MDPs is straightforward. Time-based LRA reward objectives for MA can not straightforwardly be reduced to step-based measures for MDP due to the interplay of delayed- and undelayed transitions.

3 Efficient Multi-objective Model Checking

We formalize common tasks in multi-objective model checking and sketch our solution method based on [28]. We fix an MA $\mathcal{M} = \langle S, \mathit{Act}, \Delta, \mathbf{P} \rangle$ with initial state $s_I \in S$ and $\ell > 0$ objectives $f_1, \ldots, f_\ell\colon \mathit{Paths}_{\text{inf}} \to \mathbb{R}$ with $\mathcal{F} = \langle f_1, \ldots, f_\ell \rangle$. The notation for expected values is lifted to tuples: $\mathrm{Ex}_\sigma(\mathcal{F}) = \langle \mathrm{Ex}_\sigma(f_1), \ldots, \mathrm{Ex}_\sigma(f_\ell) \rangle$.

3.1 Multi-objective Model Checking Queries

Our aim is to maximize the expected value for each (potentially conflicting) objective f_j. We impose the following assumption which can be asserted using single-objective model checking. We further discuss the assumption in Remark 2.



(a) MA $\mathcal{M}$ with rewards $\langle \mathcal{R}_1, \mathcal{R}_2 \rangle$ (b) $Ach(\mathcal{F})$ (green) and $Pareto(\mathcal{F})$ (blue)

Figure 1: MA with achievable points and Pareto front for $\mathcal{F} = \langle lra(\mathcal{R}_1), tot(\mathcal{R}_2) \rangle$

Assumption 1 (Objective Finiteness) $\forall j\colon \sup\{\mathrm{Ex}_\sigma(f_j) \mid \sigma \in \Sigma\} < \infty$.

Definition 7. *For $\mathcal{F}$ as above, $Ach(\mathcal{F}) = \{\mathbf{p} \in \mathbb{R}^\ell \mid \exists \sigma \in \Sigma\colon \mathbf{p} \le \mathrm{Ex}_\sigma(\mathcal{F})\}$ is the set of* achievable points. *The* Pareto front *is given by $Pareto(\mathcal{F}) = \{\mathbf{p} \in cl(Ach(\mathcal{F})) \mid \forall \mathbf{p}' \gtrsim \mathbf{p}\colon \mathbf{p}' \notin cl(Ach(\mathcal{F}))\}$.*

A point $\mathbf{p} \in Ach(\mathcal{F})$ is called *achievable* and there is a single strategy σ that for each objective f_j *achieves* an expected value of at least $\mathbf{p}[\![j]\!]$. Due to Assumption 1, the Pareto front is the *frontier* of the set of achievable points, meaning that it is the smallest set $P \subseteq \mathbb{R}^\ell$ with $dwconv(P) = cl(Ach(\mathcal{F}))$. We can thus interpret $Pareto(\mathcal{F})$ as a representation for $cl(Ach(\mathcal{F}))$ and vice versa. The set of achievable points is closed iff all points on the Pareto front are achievable.

Example 1. Fig. 1a shows an MA with initial state s_3. Transitions are annotated with actions, rates (boldfaced), and successor probabilities. We also depict two reward assignments $\mathcal{R}_1$ and $\mathcal{R}_2$ by labeling states and transitions with tuples $\langle r_1, r_2 \rangle$ where, e.g., $\mathcal{R}_2(s_3, \alpha, s_1) = -1$ and for $t \in \mathbb{R}_{\ge 0}$: $\mathcal{R}_1(s_2, t, s_4) = 6 \cdot t$.

For $\sigma_1 \in \Sigma_{\mathrm{md}}$ with $\sigma_1\colon s_3, s_4 \mapsto \alpha$, the EC $\{s_2, \langle s_4, \alpha\rangle, \langle s_4, \beta\rangle, s_6\}$ is reached almost surely (with probability 1), yielding $\mathrm{Ex}_{\sigma_1}(lra(\mathcal{R}_1)) = 0.6 \cdot 6 + 0.4 \cdot 1 = 4$ and $\mathrm{Ex}_{\sigma_1}(tot(\mathcal{R}_2)) = \sum_{i=0}^{\infty} -1 \cdot (0.5)^i = -2$. It follows that the point $\mathbf{p}_1 = \langle 4, -2\rangle$ as indicated in Fig. 1b is achievable. Similarly, $\sigma_2 \in \Sigma_{\mathrm{md}}$ with $\sigma_2\colon s_3 \mapsto \beta, s_4 \mapsto \alpha$ achieves the point $\mathbf{p}_2 = \langle 3, 0\rangle$. With strategies that randomly pick an action at s_3, we can also achieve any point on the blue line in Fig. 1b that connects $\mathbf{p}_1$ and $\mathbf{p}_2$. This line coincides with the Pareto front $Pareto(\mathcal{F})$ for $\mathcal{F} = \langle lra(\mathcal{R}_1), tot(\mathcal{R}_2) \rangle$. The set of achievable points $Ach(\mathcal{F})$ (indicated in green) coincides with the downward convex hull of the Pareto front.

For multi-objective model checking we are concerned with the following queries:

Multi-objective Model Checking Queries	
Qualitative Achievability:	Given point $\mathbf{p} \in \mathbb{R}^\ell$, decide if $\mathbf{p} \in Ach(\mathcal{F})$.
Quantitative Achievability:	Given $p_2, p_3, \ldots, p_\ell \in \mathbb{R}$, compute or approximate $\sup\{p \in \mathbb{R} \mid \langle p, p_2, p_3, \ldots, p_\ell\rangle \in Ach(\mathcal{F})\}$.
Pareto:	Compute or approximate $Pareto(\mathcal{F})$.

Input : MA $\mathcal{M}$ with initial state s_I, objectives $\mathcal{F} = \langle f_1, \ldots, f_\ell \rangle$
Output : An approximation of $Ach(\mathcal{F})$
1 $P \leftarrow \emptyset$ *// Collects achievable points found so far.*
2 $Q \leftarrow \mathbb{R}^\ell$ *// Excludes points that are known to be unachievable.*
3 **repeat**
4 Select weights $\mathbf{w} \in \{\mathbf{w}' \in (\mathbb{R}_{\geq 0})^\ell \mid \sum_{j=1}^{\ell} \mathbf{w}'[\![j]\!] = 1\}$ and $\varepsilon > 0$
5 Find $v_{\mathbf{w}} \geq \sup\{\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) \mid \sigma \in \Sigma\}$, $\sigma_{\mathbf{w}} \in \Sigma$ s.t. $|v_{\mathbf{w}} - \mathbf{w} \cdot \mathrm{Ex}_{\sigma_{\mathbf{w}}}(\mathcal{F})| \leq \varepsilon$
6 Compute $\mathbf{p}_{\mathbf{w}} \in \mathbb{R}^\ell$ with $\forall j\colon\ \mathbf{p}_{\mathbf{w}}[\![j]\!] = \mathrm{Ex}_{\sigma_{\mathbf{w}}}(f_j)$
7 $P \leftarrow P \cup \{\mathbf{p}_{\mathbf{w}}\}$; $Q \leftarrow Q \cap \{\mathbf{p} \in \mathbb{R}^\ell \mid \mathbf{w} \cdot \mathbf{p} \leq v_{\mathbf{w}}\}$
8 **until** Approximation $dwconv(P) \subseteq Ach(\mathcal{F}) \subseteq Q$ answers multi-obj. query

Algorithm 1: Approximating the set of achievable points

3.2 Approximation of Achievable Points

A practically efficient approach that tackles the above queries for expected total rewards in MDP was given in [28]. It is based on so-called *sandwich algorithms* known from convex multi-objective optimization [53,51]. We extend the algorithm to arbitrary combinations of objectives f_j on MA, including—and this is the main algorithmic novelty—mixtures of total- and LRA reward objectives.

The idea is to iteratively refine an approximation of the set of achievable points $Ach(\mathcal{F})$. The refinement loop is outlined in Algorithm 1. At the start of each iteration, the algorithm chooses a weight vector $\mathbf{w}$ and a precision parameter ε after some heuristic (details below). Then, Line 5, considers the weighted sum of the expected values of the objectives f_j. More precisely, an upper bound $v_{\mathbf{w}}$ for $\sup\{\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) \mid \sigma \in \Sigma\}$ as well as a "near optimal" strategy $\sigma_{\mathbf{w}}$ need to be found such that the difference between the bound $v_{\mathbf{w}}$ and the weighted sum induced by $\sigma_{\mathbf{w}}$ is at most ε. In Sect. 4, we outline the computation of $v_{\mathbf{w}}$ and $\sigma_{\mathbf{w}}$ for the case where $\mathcal{F}$ consists of total-and LRA reward objectives. Next, in Line 6 the algorithm computes a point $\mathbf{p}_{\mathbf{w}}$ that contains the expected values for each individual objective f_j under strategy $\sigma_{\mathbf{w}}$. These values can be computed using off-the-shelf single-objective model checking algorithms on the model induced by $\sigma_{\mathbf{w}}$. By definition, $\mathbf{p}_{\mathbf{w}}$ is achievable. Finally, Line 7 inserts the found point into the initially empty set P and excludes points from the set Q (which initially contains all points) that are known to be unachievable. The following theorem establishes the correctness of the approach. We prove it using Lemmas 1 and 2.

Theorem 1. *Algorithm 1 maintains the invariant* $dwconv(P) \subseteq Ach(\mathcal{F}) \subseteq Q$.

Lemma 1. $\forall\, \mathbf{p} \in Ach(\mathcal{F}), \mathbf{w} \in (\mathbb{R}_{\geq 0})^\ell\colon\ \mathbf{w} \cdot \mathbf{p} \leq \sup\{\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) \mid \sigma \in \Sigma\}$.

Proof. Let $\mathbf{p} \in Ach(\mathcal{F})$ be achieved by strategy $\sigma_{\mathbf{p}} \in \Sigma$. The claim follows from

$$\mathbf{w} \cdot \mathbf{p} = \sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathbf{p}[\![j]\!] \leq \sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathrm{Ex}_{\sigma_{\mathbf{p}}}(f_j) \leq \sup\left\{ \sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathrm{Ex}_\sigma(f_j) \,\middle|\, \sigma \in \Sigma \right\}.$$

Lemma 2. *$Ach(\mathcal{F})$ is convex, i.e., $Ach(\mathcal{F}) = conv(Ach(\mathcal{F}))$.*

Proof. We need to show that for two points $\mathbf{p}_1, \mathbf{p}_2 \in Ach(\mathcal{F})$ with achieving strategies $\sigma_1, \sigma_2 \in \Sigma$, any point $\mathbf{p}$ on the line connecting $\mathbf{p}_1$ and $\mathbf{p}_2$ is also achievable. Formally, for $w \in [0,1]$ show that $\mathbf{p}_w = w \cdot \mathbf{p}_1 + (1-w) \cdot \mathbf{p}_2 \in Ach(\mathcal{F})$. Consider the strategy σ_w that initially makes a coin flip[1]: With probability w it mimics σ_1 and otherwise it mimics σ_2. We can show for all objectives f_j:

$$\mathbf{p_w}[\![j]\!] = w \cdot \mathbf{p}_1[\![j]\!] + (1-w) \cdot \mathbf{p}_2[\![j]\!] \leq w \cdot \mathrm{Ex}_{\sigma_1}(f_j) + (1-w) \cdot \mathrm{Ex}_{\sigma_2}(f_j) = \mathrm{Ex}_{\sigma_w}(f_j).$$

We now show Theorem 1. A similar proof was given in [28].

Proof (of Theorem 1). All $\mathbf{p_w} \in P$ are achievable, i.e., $P \subseteq Ach(\mathcal{F})$. By Definition 7 and Lemma 2 we get $dwconv(P) \subseteq dwconv(Ach(\mathcal{F})) = conv(Ach(\mathcal{F})) = Ach(\mathcal{F})$. Now let $\mathbf{p} \in Ach(\mathcal{F})$ and let $\mathbf{w}$ be an arbitrary weight vector considered in some iteration of Algorithm 1 with corresponding value $v_{\mathbf{w}}$ computed in Line 5. Lemma 1 yields $\mathbf{w} \cdot \mathbf{p} \leq \sup\{\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) \mid \sigma \in \Sigma\} \leq v_{\mathbf{w}}$ and thus $\mathbf{p} \in Q$.

Algorithm 1 can be stopped at any time and the current approximation of $Ach(\mathcal{F})$ can be used to (i) decide qualitative achievability, (ii) provide a lower and an upper bound for quantitative achievability, and (iii) obtain an approximative representation of the Pareto front.

The *precision parameter* ε can be decreased dynamically to obtain a gradually finer approximation. If $Ach(\mathcal{F})$ is closed, the supremum $\sup\{\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) \mid \sigma \in \Sigma\}$ can be attained by some strategy $\sigma_{\mathbf{w}}$, allowing us to set $\varepsilon = 0$.

We briefly sketch the *selection of weight vectors* as proposed in [28]. In the first ℓ iterations of Algorithm 1, we optimize each objective f_j individually, i.e., we consider for all j the weight vector $\mathbf{w}$ with $\mathbf{w}[\![i]\!] = 0$ for $i \neq j$ and $\mathbf{w}[\![j]\!] = 1$. After that, we consider weight vectors that are orthogonal to a facet of the downward convex hull of the current set of points P. To approximate the Pareto front, facets with a large distance to $\mathbb{R}^\ell \setminus Q$ are considered first. To answer a qualitative or quantitative achievability query, the selection can be guided further based on the input point $\mathbf{p} \in \mathbb{R}^\ell$ or the input values $p_2, p_3, \ldots, p_\ell \in \mathbb{R}$. More details and further discussions on these heuristics can be found in [28].

Remark 2. Assumption 1 does not exclude $\mathrm{Ex}_\sigma(f_j) = -\infty$ which occurs, e.g., when objectives reflect resource consumption and some (bad) strategies require infinite resources. Moreover, if Assumption 1 is violated for an objective f_j we observe that for this objective, any (arbitrarily high) value $p \in \mathbb{R}$ can be achieved with some strategy $\sigma \in \Sigma$ such that $p \leq \mathrm{Ex}_\sigma(f_j)$. Similar to the proof of Lemma 2, a strategy can be constructed that—with a small probability—mimics a strategy inducing a very high expected value for f_j and—with the remaining (high) probability—optimizes for the other objectives. Let $\mathcal{F}_{-j}$ be the tuple $\mathcal{F}$ without f_j and similarly for $\mathbf{p} \in \mathbb{R}^\ell$ let $\mathbf{p}_{-j} \in \mathbb{R}^{\ell-1}$ be the point $\mathbf{p}$ without the j^{th} entry. Assuming $\inf\{\mathrm{Ex}_\sigma(f_j) \mid \sigma \in \Sigma\} > -\infty$, we can show that $cl(Ach(\mathcal{F})) = \{\mathbf{p} \in \mathbb{R}^\ell \mid \mathbf{p}_{-j} \in cl(Ach(\mathcal{F}_{-j}))\}$. Put differently, $cl(Ach(\mathcal{F}))$ can be constructed from the achievable points obtained without the objective f_j.

[1] Strategies as in Definition 3 can not "store" the outcome of the initial coin flip. Thus, given $\hat{\pi} \in Paths_{\text{fin}}$, strategy σ_w actually has to consider the *conditional* probability for the outcome of the coin flip, given that $\hat{\pi}$ has been observed. Alternatively, we could have also introduced strategies with memory.

4 Optimizing Weighted Combinations of Objectives

We now analyze weighted sums of expected values as in Line 5 of Algorithm 1.

WEIGHTED SUM OPTIMIZATION PROBLEM
Input: MA $\mathcal{M}$ with initial state s_I, objectives $\mathcal{F} = \langle f_1, \ldots, f_\ell \rangle$,
weight vector $\mathbf{w} \in \{\mathbf{w}' \in (\mathbb{R}_{\geq 0})^\ell \mid \sum_{j=1}^{\ell} \mathbf{w}'[\![j]\!] = 1\}$, precision $\varepsilon > 0$
Output: Value $v_{\mathbf{w}} \in \mathbb{R}$, with $v_{\mathbf{w}} \geq \sup\{\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) \mid \sigma \in \Sigma\}$ and
strategy $\sigma_{\mathbf{w}} \in \Sigma$ such that $|v_{\mathbf{w}} - \mathbf{w} \cdot \mathrm{Ex}_{\sigma_{\mathbf{w}}}(\mathcal{F})| \leq \varepsilon$.

We only consider total- and LRA reward objectives. Remark 3 discusses other objectives. We show that instead of a weighted sum of the expected values we can consider weighted sums of the rewards. This allows us to combine all objectives into a single reward assignment and then apply single-objective model checking.

4.1 Pure Long-run Average Queries

Initially, we restrict ourselves to LRA objectives and show a reduction of the weighted sum optimization problem to a single-objective long-run average reward computation. As usual for MA [38,29,17] we forbid so-called Zeno behavior.

Assumption 2 (Non-Zenoness) $\forall \sigma \in \Sigma^{\mathcal{M}} \colon \Pr_\sigma^{\mathcal{M}}(\{\pi \mid dur(\pi) < \infty\}) = 0$.

The assumption is equivalent to assuming that every EC of $\mathcal{M}$ contains at least one Markovian state. If the assumption holds, the limit in Definition 6 can be attained almost surely (with probability 1) and corresponds to a value $v \in \mathbb{R}$. Thus, Assumption 1 for LRA objectives is already implied by Assumption 2. Let $\mathcal{F}_{lra} = \langle lra(\mathcal{R}_1), \ldots, lra(\mathcal{R}_\ell) \rangle$ with reward assignments $\mathcal{R}_j$. Moreover, for weight vector $\mathbf{w}$ let $\mathcal{R}_{\mathbf{w}}$ be the reward assignment with $\mathcal{R}_{\mathbf{w}}(\langle s, \kappa \rangle, s') = \sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathcal{R}_j(\langle s, \kappa \rangle, s')$.

Theorem 2. $\forall \sigma \in \Sigma \colon \mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}_{lra}) = \mathrm{Ex}_\sigma(lra(\mathcal{R}_{\mathbf{w}}))$.

Proof. Due to Assumption 2 we have for almost all paths $\pi \in Paths_{\mathrm{inf}}$ that for all $j \in \{1, \ldots, \ell\}$ the limit $\lim_{t \to \infty} \frac{1}{t} \cdot \mathcal{R}_j(prefix_{time}(\pi, t))$ exists and

$$\sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot lra(\mathcal{R}_j)(\pi) = \lim_{t \to \infty} \frac{1}{t} \cdot \sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathcal{R}_j(prefix_{time}(\pi, t)) = lra(\mathcal{R}_{\mathbf{w}})(\pi).$$

The theorem follows with

$$\sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathrm{Ex}_\sigma(lra(\mathcal{R}_j)) = \int_\pi \sum_{j=1}^{\ell} \mathbf{w}[\![j]\!] \cdot lra(\mathcal{R}_j) \, \mathrm{dPr}_\sigma(\pi) = \mathrm{Ex}_\sigma(lra(\mathcal{R}_{\mathbf{w}})).$$

Due to Theorem 2, it suffices to consider the expected LRA reward for the *single* reward assignment $\mathcal{R}_{\mathbf{w}}$. The supremum $\sup\{\mathrm{Ex}_\sigma(lra(\mathcal{R}_{\mathbf{w}})) \mid \sigma \in \Sigma\}$ is attained by some memoryless deterministic strategy $\sigma_{\mathbf{w}} \in \Sigma_{\mathrm{md}}$ [30]. Such a strategy and the induced value $v_{\mathbf{w}} = \mathrm{Ex}_{\sigma_{\mathbf{w}}}(lra(\mathcal{R}_{\mathbf{w}}))$ can be computed (or approximated) with *linear programming* [30], *strategy iteration* [42] or *value iteration* [17,1].

4.2 A Two-phase Approach for Single-objective LRA

The computation of single-objective expected LRA rewards for reward assignment $\mathcal{R}_{\mathbf{w}}$ can be divided in two phases [29,17,1]. First, each maximal end component $C \in \mathit{MECS}(\mathcal{M})$ is analyzed individually by computing for sub-MA $\mathcal{M}[\![C]\!]$ and some[2] $s \in \mathit{states}(C)$ the value $v_C = \max\{\mathrm{Ex}_\sigma^{\mathcal{M}[\![C]\!],s}(\mathit{lra}(\mathcal{R}_{\mathbf{w}})) \mid \sigma \in \Sigma_{\mathrm{md}}^{\mathcal{M}[\![C]\!]}\}$.

Secondly, we consider a quotient model $\mathcal{M}' = \mathcal{M}_{\setminus \mathit{MECS}(\mathcal{M})}$ of $\mathcal{M}$ that replaces the states of each $C \in \mathit{MECS}(\mathcal{M})$ by a single state.

Definition 8. *For $\mathcal{M} = \langle S, \mathit{Act}, \Delta, \mathbf{P}\rangle$ and a set of ECs $\mathcal{C}$, the* quotient *is the MA $\mathcal{M}_{\setminus\mathcal{C}} = \langle S_{\setminus\mathcal{C}}, \mathit{Act}_{\setminus\mathcal{C}}, \Delta_{\setminus\mathcal{C}}, \mathbf{P}_{\setminus\mathcal{C}}\rangle$ where*

- $S_{\setminus\mathcal{C}} = \big(S \setminus \bigcup_{C\in\mathcal{C}} \mathit{states}(C)\big) \uplus \mathcal{C} \uplus \{s_\perp\}$, $\mathit{Act}_{\setminus\mathcal{C}} = \mathit{Act} \uplus \big(\bigcup_{C\in\mathcal{C}} \mathit{exits}(C)\big) \uplus \{\perp\}$,
- $\Delta_{\setminus\mathcal{C}}(\hat{s}) = \begin{cases} \Delta(\hat{s}) & \text{if } \hat{s} \in S \\ \mathit{exits}(\hat{s}) \cup \{\perp\} & \text{if } \hat{s} \in \mathcal{C} \\ 1 & \text{if } \hat{s} = s_\perp, \text{ and} \end{cases}$
- $\mathbf{P}_{\setminus\mathcal{C}}(c) = \begin{cases} \mathbf{P}(c) & \text{if } c \in \mathit{MS}^{\mathcal{M}} \cup \mathit{SA}^{\mathcal{M}} \\ \mathbf{P}(\langle s,\alpha\rangle) & \text{if } c = \langle C, \langle s,\alpha\rangle\rangle \text{ for } C \in \mathcal{C} \text{ and } \langle s,\alpha\rangle \in \mathit{exits}(C) \\ \{s_\perp \mapsto 1\} & \text{if } c \in \mathcal{C} \times \{\perp\} \cup \{s_\perp\} \end{cases}$

Intuitively, selecting action $\perp$ at a state $C \in \mathit{MECS}(\mathcal{M})$ in $\mathcal{M}'$ reflects any strategy of $\mathcal{M}$ that upon visiting the EC C will stay in this EC forever. We can thus mimic any strategy of the sub-MA $\mathcal{M}[\![C]\!]$, in particular a memoryless deterministic strategy that maximizes the expected value of $\mathit{lra}(\mathcal{R}_{\mathbf{w}})$ in $\mathcal{M}[\![C]\!]$. Contrarily, selecting an action $\langle s,\alpha\rangle$ at a state C of $\mathcal{M}'$ reflects a strategy of $\mathcal{M}$ that upon visiting the EC C enforces that the states of C will be left via the exiting state-action pair $\langle s,\alpha\rangle$. Let $\mathcal{R}^*$ be the reward assignment for $\mathcal{M}'$ that yields $\mathcal{R}^*(\langle C, \perp\rangle, s_\perp) = v_C$ and 0 in all other cases. It can be shown that $\max\{\mathrm{Ex}_\sigma^{\mathcal{M},s_I}(\mathit{lra}(\mathcal{R}_{\mathbf{w}})) \mid \sigma \in \Sigma^{\mathcal{M}}\} = \max\{\mathrm{Ex}_\sigma^{\mathcal{M}',s_I'}(\mathit{tot}(\mathcal{R}^*)) \mid \sigma \in \Sigma^{\mathcal{M}'}\}$, where $s_I' = C_I$ if s_I is contained in some $C_I \in \mathit{MECS}(\mathcal{M})$ and $s_I' = s_I$ otherwise.

The maximal total reward in $\mathcal{M}'$ can be computed using standard techniques such as *value iteration* and *policy iteration* [46] as well as the more recent *sound value iteration* and *optimistic value iteration* [48,36]. The latter two provide sound precision guarantees for the output value v, i.e., $|v - \max\{\mathrm{Ex}_\sigma^{\mathcal{M}',s_I'}(\mathit{tot}(\mathcal{R}^*)) \mid \sigma \in \Sigma^{\mathcal{M}'}\}| \le \varepsilon$ for a given $\varepsilon > 0$.

4.3 Combining Long-run Average and Total Rewards

We now consider arbitrary combinations of total- and long-run average reward objectives $\mathcal{F} = \langle \mathit{tot}(\mathcal{R}_1), \ldots, \mathit{tot}(\mathcal{R}_k), \mathit{lra}(\mathcal{R}_{k+1}), \ldots, \mathit{lra}(\mathcal{R}_\ell)\rangle$ with $0 < k < \ell$.

The above-mentioned procedure for LRA reduces the analysis to an expected total reward computation on the quotient model $\mathcal{M}_{\setminus\mathit{MECS}(\mathcal{M})}$. This approach suggests to also incorporate other total-reward objectives for $\mathcal{M}$ in the quotient

[2] The value v_C does not depend on the selected state s. Intuitively, this is because any other state $s' \in \mathit{states}(C)$ can be reached from s almost surely.

model. However, special care has to be taken concerning total rewards collected within ECs of $\mathcal{M}$ that would no longer be present in the quotient $\mathcal{M}_{\setminus MECS(\mathcal{M})}$. We discuss how to deal with this issue by considering the quotient only for ECs in which no (total) reward is collected. We start with restricting the (total) rewards that might be assigned to transitions within EC.

Assumption 3 (Sign-Consistency) *For all total reward objectives $tot(\mathcal{R}_j)$ either $\forall\, C \in MECS(\mathcal{M})\colon\ \mathcal{R}_j(C) \geq 0$ or $\forall\, C \in MECS(\mathcal{M})\colon\ \mathcal{R}_j(C) \leq 0$.*

The assumption implies that paths on which infinitely many positive *and* infinitely many negative reward is collected have probability 0. One consequence is that the limit in Definition 5 exists for almost all paths [3]. A discussion on objectives $tot(\mathcal{R}_j)$ that violate Assumption 3 for single-objective MDP is given in [3]. Their multi-objective treatment is left for future work.

When Assumptions 1 and 3 hold, we get $\mathcal{R}_j(C) \leq 0$ for all objectives $tot(\mathcal{R}_i)$ and EC C. Put differently, all non-zero total rewards collected in an EC have to be negative. Strategies that induce a total reward of $-\infty$ for some objective $tot(\mathcal{R}_i)$ will not be taken into account for the set of achievable points. Therefore, transitions within ECs that yield negative reward should only be taken finitely often. These transitions can be disregarded when computing the expected LRA rewards, i.e., only the 0-ECs [3] are relevant for the LRA computation.

Definition 9. *A 0-EC of $\mathcal{M}$ and $\mathcal{R}_1, \ldots, \mathcal{R}_k$ is an EC C of $\mathcal{M}$ with $\mathcal{R}_i(C) = 0$ for all $\mathcal{R}_i$. The set of maximal 0-ECs is given by $MECS_0(\mathcal{M}, \langle \mathcal{R}_1, \ldots, \mathcal{R}_i \rangle)$.*

$MECS_0(\mathcal{M}, \langle \mathcal{R}_1, \ldots, \mathcal{R}_k \rangle)$ can be computed by constructing the maximal ECs of the sub-MA of $\mathcal{M}$ where transitions with a non-zero reward are erased.

We are ready to describe our approach that combines LRA rewards of 0-ECs and the remaining total rewards into a single total-reward objective. Let $\mathcal{R}^{tot}_{\mathbf{w}}$ and $\mathcal{R}^{lra}_{\mathbf{w}}$ be reward assignments with $\mathcal{R}^{tot}_{\mathbf{w}}(\langle s, \kappa \rangle, s') = \sum_{i=1}^{k} \mathbf{w}[\![i]\!] \cdot \mathcal{R}_i(\langle s, \kappa \rangle, s')$ and $\mathcal{R}^{lra}_{\mathbf{w}}(\langle s, \kappa \rangle, s') = \sum_{j=k}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathcal{R}_j(\langle s, \kappa \rangle, s')$. Moreover, for $\pi \in Paths_{\text{inf}}$ we set $(tot(\mathcal{R}^{tot}_{\mathbf{w}}) + lra(\mathcal{R}^{lra}_{\mathbf{w}}))(\pi) = tot(\mathcal{R}^{tot}_{\mathbf{w}})(\pi) + lra(\mathcal{R}^{lra}_{\mathbf{w}})(\pi)$.

Theorem 3. $\forall\, \sigma \in \Sigma\colon\ \mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) = \mathrm{Ex}_\sigma(tot(\mathcal{R}^{tot}_{\mathbf{w}}) + lra(\mathcal{R}^{lra}_{\mathbf{w}}))$.

Proof. Using a similar reasoning as in the proof of Theorem 2, we get:

$$\begin{aligned}\mathbf{w} \cdot \mathrm{Ex}_\sigma(\mathcal{F}) &= \Big(\sum_{i=1}^{k} \mathbf{w}[\![i]\!] \cdot \mathrm{Ex}_\sigma(tot(\mathcal{R}_i))\Big) + \Big(\sum_{j=k+1}^{\ell} \mathbf{w}[\![j]\!] \cdot \mathrm{Ex}_\sigma(lra(\mathcal{R}_j))\Big) \\ &= \mathrm{Ex}_\sigma(tot(\mathcal{R}^{tot}_{\mathbf{w}})) + \mathrm{Ex}_\sigma(lra(\mathcal{R}^{lra}_{\mathbf{w}})) = \mathrm{Ex}_\sigma(tot(\mathcal{R}^{tot}_{\mathbf{w}}) + lra(\mathcal{R}^{lra}_{\mathbf{w}})).\end{aligned}$$

Algorithm 2 outlines the procedure for solving the weighted sum optimization problem. It first computes optimal LRA rewards and inducing strategies for each maximal 0-EC (Lines 1 to 3). Then, a quotient model $\mathcal{M}^*$ and a reward assignment $\mathcal{R}^*$ incorporating all total- and LRA rewards is build and analyzed (Lines 4 to 6). $\mathcal{M}^*$ might still contain ECs other than $\{s_\perp\}$. Those ECs shall be left eventually to avoid collecting infinite negative reward for a total reward objective $tot(\mathcal{R}_i)$. Note that the weight $\mathbf{w}[\![i]\!]$ for such an objective might be zero,

Input : MA $\mathcal{M}$ with initial state s_I, objectives $\mathcal{F} = \langle tot(\mathcal{R}_1), \ldots, tot(\mathcal{R}_k), lra(\mathcal{R}_{k+1}), \ldots, lra(\mathcal{R}_\ell)\rangle$, weight vector $\mathbf{w}$
Output : Value $v_{\mathbf{w}}$, strategy $\sigma_{\mathbf{w}}$ as in the weighted sum optimization problem

1 $\mathcal{C} \leftarrow MECS_0(\mathcal{M}, \langle \mathcal{R}_1, \ldots, \mathcal{R}_i\rangle)$ *// Compute maximal 0-ECs and their LRA.*
2 **foreach** $C \in \mathcal{C}$ **do**
3 Compute $v_C = \max\left\{\mathrm{Ex}_\sigma^{\mathcal{M}[\![C]\!]}(lra(\mathcal{R}_{\mathbf{w}}^{lra})) \mid \sigma \in \Sigma_{\mathrm{md}}^{\mathcal{M}[\![C]\!]}\right\}$ and inducing strategy $\sigma_C \in \Sigma_{\mathrm{md}}^{\mathcal{M}[\![C]\!]}$
4 $\mathcal{M}^* \leftarrow \mathcal{M}_{\setminus\mathcal{C}}$ *// Build and analyze quotient model.*
5 Build reward assignment $\mathcal{R}^*$ with

$$\mathcal{R}^*(\langle s,\kappa\rangle, s') = \begin{cases} v_C & \text{if } s = C, \kappa = \bot, \text{ and } s' = s_\bot \\ \mathcal{R}_{\mathbf{w}}^{tot}(\langle \hat{s},\alpha\rangle, s') & \text{if } s = C, \kappa = \langle \hat{s},\alpha\rangle \in exits(C) \\ \mathcal{R}_{\mathbf{w}}^{tot}(\langle s,\alpha\rangle, s') & \text{otherwise} \end{cases}$$

6 Compute $v_{\mathbf{w}} = \max\left\{\mathrm{Ex}_\sigma^{\mathcal{M}^*}(tot(\mathcal{R}^*)) \,\middle|\, \sigma \in \Sigma_{\mathrm{md}}^{\mathcal{M}^*},\ \Pr_\sigma^{\mathcal{M}^*}(\Diamond\{s_\bot\}) = 1\right\}$ and inducing strategy $\sigma^* \in \Sigma_{\mathrm{md}}^{\mathcal{M}^*}$
7 Build strategy $\sigma_{\mathbf{w}} \in \Sigma_{\mathrm{md}}^{\mathcal{M}}$ with

$$\sigma_{\mathbf{w}}(s) = \begin{cases} \sigma_C(s) & \text{if } \exists C \in \mathcal{C}\colon\ s \in states(C) \text{ and } \sigma^*(C \in \mathcal{C}) = \bot \\ \alpha & \text{if } \exists C \in \mathcal{C}\colon\ s \in states(C) \text{ and } \sigma^*(C) = \langle s,\alpha\rangle \\ \sigma_{C\Diamond\{s'\}}(s) & \text{if } \exists C \in \mathcal{C}\colon\ s \in states(C) \text{ and } \sigma^*(C) = \langle s',\alpha\rangle \text{ for } s' \neq s \\ \sigma^*(s) & \text{otherwise} \end{cases}$$

Algorithm 2: Optimizing the weighted sum for total and LRA objectives

i.e., the rewards of $\mathcal{R}_i$ are not present in $\mathcal{R}^*$. It is therefore necessary to explicitly restrict the analysis to strategies that almost surely (i.e., with probability 1) reach $s_\bot$. To compute the maximal expected total reward in Line 6 with, e.g., standard value iteration, we can consider another quotient model for $\mathcal{M}^*$ and the 0-ECs of $\mathcal{M}^*$ and $\mathcal{R}^*$. In contrast to Definition 8, this quotient should not introduce the $\bot$ action since it shall not be possible to remain in an EC forever. In Line 7, the strategies for the 0-ECs and for the quotient $\mathcal{M}^*$ are combined into one strategy $\sigma_{\mathbf{w}}$ for $\mathcal{M}$. Here, $\sigma_{C\Diamond s'}$ refers to a strategy of $\mathcal{M}[\![C]\!]$ for which every state $s \in states(C)$ eventually reaches $s' \in states(C)$ almost surely.

Since Algorithm 2 produces a memoryless deterministic strategy $\sigma_{\mathbf{w}}$, the point $\mathbf{p}_{\mathbf{w}} \in \mathbb{R}^\ell$ in Line 6 of Algorithm 1 can be computed on the induced sub-MA for $\sigma_{\mathbf{w}}$. Assuming exact single-objective solution methods, the resulting value $v_{\mathbf{w}}$ and strategy $\sigma_{\mathbf{w}} \in \Sigma_{\mathrm{md}}^{\mathcal{M}}$ of Algorithm 2 satisfy $v_{\mathbf{w}} = \mathbf{w} \cdot \mathrm{Ex}_{\sigma_{\mathbf{w}}}(\mathcal{F})$, yielding an exact solution to the weighted sum optimization problem. As the number of memoryless deterministic strategies is bounded, we conclude the following, extending results for pure LRA queries [11] to mixtures with total rewards.

Corollary 1. *For total- and LRA reward objectives $\mathcal{F}$, $Ach(\mathcal{F})$ is closed and is the downward convex hull of at most $|\Sigma_{\mathrm{md}}^{\mathcal{M}}| = \prod_{s\in PS} |\Delta(s)|$ points.*

Remark 3. Our framework can be extended to support objectives beyond total- and LRA rewards. *Minimizing objectives* where one is interested in a strategy σ

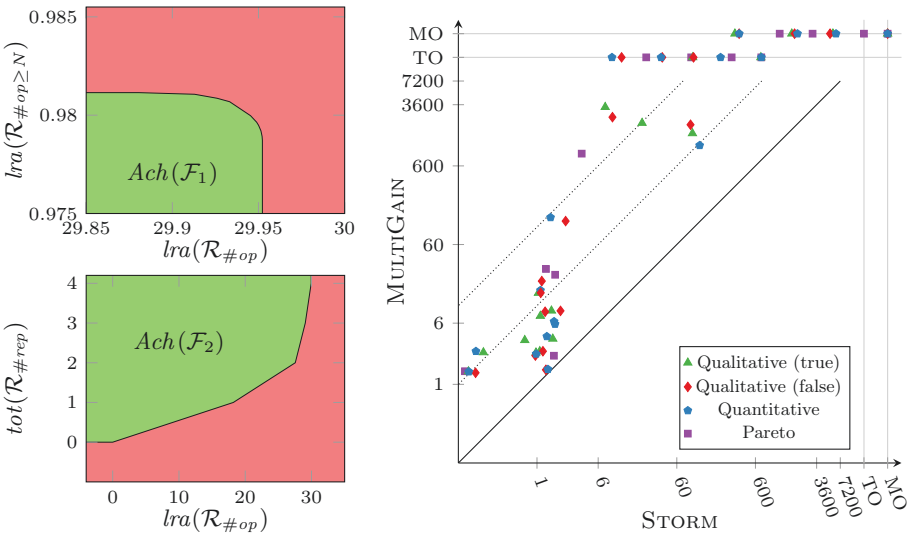
that induces a *small* expected value can be considered by multiplying all rewards with -1. Since we already allow negative values in reward assignments, no further adaptions are necessary. We emphasize that our framework lifts a restriction imposed in [28] that disabled a simultaneous analysis of maximizing *and* minimizing total reward objectives. *Reachability probabilities* can be transformed to expected total rewards on a modified model in which the information whether a goal state has already been visited is stored in the state-space. *Goal-bounded* total rewards as in [30], where no further rewards are collected as soon as one of the goal states is reached can be transformed similarly. For MDP, *step- and reward-bounded* reachability probabilities can be converted to total reward objectives by unfolding the current amount of steps (or rewards) into the state-space of the model. Approaches that avoid such an expensive unfolding have been presented in [28] for objectives with step-bounds and in [34,35] for objectives with one or multiple reward-bounds. *Time-bounded* reachability probabilities for MA have been considered in [47]. Finally, ω-regular specifications such as *linear temporal logic (LTL)* formulae have been transformed to total reward objectives in [27]. However, the optimization of LRA rewards within the ECs of the model might interfere with the satisfaction of one or more ω-regular specifications [31].

5 Experimental Evaluation

Implementation details Our approach has been implemented in the model checker Storm [40]. Given an MA or MDP (specified using the PRISM language or JANI [14]), the tool answers qualitative- and quantitative achievability as well as Pareto queries. Beside of mixtures of total- and LRA reward objectives, Storm also supports most of the extensions in Remark 3—with the notable exception of LTL. We use LRA value iteration [17,1] and sound value iteration [48] for calls to single-objective model checking. Both provide sound precision guarantees, i.e., the relative error of these computations is at most ε, where we set $\varepsilon = 10^{-6}$.

Workstation cluster To showcase the capabilities of our implementation, we present a workstation cluster—originally considered in [39] as a CTMC—now modeled as an MA. The cluster considers two sub-clusters each consisting of one *switch* and N *workstations*. Within each sub-cluster the workstations are connected to the switch in a star topology and the two switches are connected with a *backbone*. Each of the components may fail with a certain rate. A controller can (i) acquire additional repair units (up to M) and (ii) control the movements of the repair units. In Fig. 2a we depict the resulting sets of achievable points—as computed by our implementation—for $N = 16$ and $M = 4$. As objectives, we considered the long-run average number of operating workstations $lra(\mathcal{R}_{\#op})$, the long-run average probability that at least N workstations are operational $lra(\mathcal{R}_{\#op \geq N})$, and the total number of acquired repair units $tot(\mathcal{R}_{\#rep})$.

Related tools MultiGain [12] is an extension of PRISM [45] that implements the LP-based approach of [11] for multiple LRA objectives on MDP to answer



(a) Results for workstation cluster (b) Comparison of Storm and MultiGain

Figure 2: Exemplary results and runtime comparison with MultiGain

qualitative and quantitative achievability as well as Pareto queries. For the latter, it is briefly mentioned in [12] that ideas of [28] were used similar to our approach but no further details are provided. MultiGain does not support MA, *mixtures* with total reward objectives, and Pareto queries with $\ell > 2$ objectives. However, it does support more general quantitative achievability queries.

PRISM-games [44,43] implements value iteration over convex sets [8,9] to analyze multiple LRA reward objectives on stochastic games (SGs). By converting MDPs to 1-player SGs, PRISM-games could also be applied in our setting. However, some experiments on 1-player SGs indicated that this approach is not competitive compared to the dedicated MDP implementations in MultiGain and Storm. We therefore do not consider PRISM-games in our evaluation.

Benchmarks We consider 10 different case studies including the workstation cluster (clu) as well as benchmarks from QVBS [37] (dpm, rqs, res), from MultiGain [12] (mut, phi, vir), from [42] (csn, sen), and from [47] (pol). For each case study we consider 3 concrete instances resulting in 12 MAs and 18 MDPs. The analyzed objectives range over LRA rewards, (goal-bounded) total rewards, and time-, step- and unbounded reachability probabilities.

Set-up We evaluated the performance of Storm and MultiGain Version 1.0.2[3]. All experiments were run on 4 cores[4] of an Intel Xeon Platinum 8160 CPU with

[3] Obtained from http://qav.cs.ox.ac.uk/multigain and invoked with Gurobi [32].

[4] Storm uses one core, MultiGain uses multiple cores due to Java's garbage collection and Gurobi's parallel solving techniques.

Table 1: Results for pure LRA Pareto queries

Model	Par.	#lra	$\|S\|$	$\|MS\|$	$\|\Delta\|$	$\|\mathcal{C}\|$	$\|S_{\mathcal{C}}\|$	#iter	Storm	MultiGain
csn	3	3	177		427	38	158	9	1.23	
csn	4	4	945		2753	176	880	30	109	
csn	5	5	4833		$2{\cdot}10^4$	782	4622		TO	
mut	3	2	$3{\cdot}10^4$		$5{\cdot}10^4$	1	$3{\cdot}10^4$	15	3.7	859
mut	4	2	$7{\cdot}10^5$		$1{\cdot}10^6$	1	$7{\cdot}10^5$	14	91.4	TO
mut	5	2	$1{\cdot}10^7$		$3{\cdot}10^7$	1	$1{\cdot}10^7$	12	3197	MO
phi	4	2	9440		$4{\cdot}10^4$	1	9440	6	1.7	24.7
phi	5	2	$9{\cdot}10^4$		$4{\cdot}10^5$	1	$9{\cdot}10^4$	18	24.5	TO
phi	6	2	$2{\cdot}10^6$		$1{\cdot}10^7$	1	$2{\cdot}10^6$	12	1221	MO
res	5-5	2	2618		8577	1	2618	16	1.64	2.31
res	15-15	2	$2{\cdot}10^5$		$7{\cdot}10^5$	1	$2{\cdot}10^5$	3	712	TO
res	20-20	2	$8{\cdot}10^5$		$2{\cdot}10^6$	1	$8{\cdot}10^5$	7	299	TO
sen	2	3	7855		$2{\cdot}10^4$	3996	6105	13	3.41	
sen	3	3	$8{\cdot}10^4$		$3{\cdot}10^5$	$5{\cdot}10^4$	$7{\cdot}10^4$	14	274	
sen	4	3	$6{\cdot}10^5$		$3{\cdot}10^6$	$4{\cdot}10^5$	$5{\cdot}10^5$		TO	
vir	2	2	80		393	2	66	4	<1	1.47
vir	3	2	$2{\cdot}10^4$		$2{\cdot}10^5$	2	$2{\cdot}10^4$	2	1.3	29.3
vir	4	2	$4{\cdot}10^7$		$7{\cdot}10^8$	?	?		MO	MO
clu	8-3	2	$2{\cdot}10^5$	$1{\cdot}10^5$	$4{\cdot}10^5$	4	$2{\cdot}10^5$	11	287	
clu	16-4	2	$2{\cdot}10^6$	$9{\cdot}10^5$	$4{\cdot}10^6$	5	$2{\cdot}10^6$	10	4199	
clu	32-3	2	$2{\cdot}10^6$	$1{\cdot}10^6$	$5{\cdot}10^6$	4	$2{\cdot}10^6$		TO	
dpm	3-3	2	2640	1008	3240	1	2640	32	19.5	
dpm	4-4	2	$3{\cdot}10^4$	$1{\cdot}10^4$	$4{\cdot}10^4$	1	$3{\cdot}10^4$	33	1179	
dpm	5-5	2	$6{\cdot}10^5$	$2{\cdot}10^5$	$7{\cdot}10^5$	1	$6{\cdot}10^5$		TO	
pol	3-3	2	9522	4801	$2{\cdot}10^4$	1	9522	17	3.44	
pol	4-3	2	$5{\cdot}10^4$	$3{\cdot}10^4$	$1{\cdot}10^5$	1	$5{\cdot}10^4$	19	19.2	
pol	4-4	2	$8{\cdot}10^5$	$5{\cdot}10^5$	$2{\cdot}10^6$	1	$8{\cdot}10^5$	29	3350	
rqs	2-2	2	1619	628	2296	1	1618	63	4.52	
rqs	3-3	2	$9{\cdot}10^4$	$4{\cdot}10^4$	$1{\cdot}10^5$	1	$9{\cdot}10^4$	106	162	
rqs	5-3	2	$2{\cdot}10^6$	$1{\cdot}10^6$	$4{\cdot}10^6$	1	$2{\cdot}10^6$	97	4345	

a time limit of 2 hours and 32 GB RAM. For each experiment we measured the total runtime (including model building) to solve one query. For qualitative and quantitative achievability we consider thresholds close to the Pareto front. For Pareto queries, the approximation precision 10^{-4} was set to both tools.

Results Fig. 2b visualizes the runtime comparison with MultiGain. A point $\langle x, y\rangle$ in the plot corresponds to a query that has been solved by Storm in x seconds and by MultiGain in y seconds. Points on the solid diagonal mean that both tools were equally fast. The two dotted lines indicate experiments where Storm only required $\frac{1}{10}$ resp. $\frac{1}{100}$ of the time of MultiGain. TO and MO indicate a time- or memory out. Tables 1 and 2 provide further data for Pareto queries. The columns indicate model name and parameters, the number of LRA reward, total reward, and bounded reachability objectives, the number of states ($|S|$), Markovian states ($|MS|$), successor distributions ($|\Delta|$), 0-ECs ($|\mathcal{C}|$), and states within 0-ECs ($|S_{\mathcal{C}}|$) of the MA or MDP, the number of iterations (#iters) of Algorithm 1 performed by Storm, and the total runtime of Storm and MultiGain in seconds. Runtimes are omitted if the tool does not support the query. MDP (MA) benchmarks are at the top (bottom) of each table. Table 1 considers pure LRA queries, whereas Table 2 considers mixtures.

Table 2: Results for Pareto queries with other objective types

Model	Par.	#lra/tot/bnd	$\|S\|$	$\|MS\|$	$\|\Delta\|$	$\|\mathcal{C}\|$	$\|S_{\mathcal{C}}\|$	#iter	Storm
res	5-5	2-0-1	2618		8577	1	2618	17	4.27
res	5-5	2-1-0	2618		8577	1	1705	6	1.43
res	15-15	2-0-1	$2{\cdot}10^5$		$7{\cdot}10^5$	1	$2{\cdot}10^5$	4	792
res	15-15	2-1-0	$2{\cdot}10^5$		$7{\cdot}10^5$	1	$1{\cdot}10^5$	8	1061
res	20-20	2-0-1	$8{\cdot}10^5$		$2{\cdot}10^6$	1	$8{\cdot}10^5$	8	641
res	20-20	2-1-0	$8{\cdot}10^5$		$2{\cdot}10^6$	1	$4{\cdot}10^5$	4	101
clu	8-3	1-1-0	$2{\cdot}10^5$	$1{\cdot}10^5$	$4{\cdot}10^5$	4	$2{\cdot}10^5$	7	163
clu	16-4	1-1-0	$2{\cdot}10^6$	$9{\cdot}10^5$	$4{\cdot}10^6$	5	$2{\cdot}10^6$	9	3432
clu	32-3	1-1-0	$2{\cdot}10^6$	$1{\cdot}10^6$	$5{\cdot}10^6$	4	$2{\cdot}10^6$	7	3328
dpm	3-3	1-0-1	5232	1980	6408	46	3045	2	11.2
dpm	3-3	1-1-0	4584	1656	5562	25	2856	4	<1
dpm	4-4	1-0-1	$7{\cdot}10^4$	$2{\cdot}10^4$	$8{\cdot}10^4$	497	$4{\cdot}10^4$	2	214
dpm	4-4	1-1-0	$6{\cdot}10^4$	$2{\cdot}10^4$	$7{\cdot}10^4$	301	$4{\cdot}10^4$	4	3.32
dpm	5-5	1-0-1	$1{\cdot}10^6$	$3{\cdot}10^5$	$1{\cdot}10^6$	6476	$6{\cdot}10^5$		TO
dpm	5-5	1-1-0	$1{\cdot}10^6$	$3{\cdot}10^5$	$1{\cdot}10^6$	4321	$6{\cdot}10^5$	4	329
pol	3-3	1-1-0	$1{\cdot}10^4$	5309	$2{\cdot}10^4$	1	9522	3	1.37
pol	4-3	1-1-0	$6{\cdot}10^4$	$3{\cdot}10^4$	$1{\cdot}10^5$	1	$5{\cdot}10^4$	3	2.52
pol	4-4	1-1-0	$9{\cdot}10^5$	$5{\cdot}10^5$	$2{\cdot}10^6$	1	$8{\cdot}10^5$	3	237
rqs	2-2	1-1-0	2805	1039	4159	1	1618	3	<1
rqs	3-3	1-1-0	$1{\cdot}10^5$	$6{\cdot}10^4$	$3{\cdot}10^5$	1	$9{\cdot}10^4$	3	4.51
rqs	5-3	1-1-0	$3{\cdot}10^6$	$2{\cdot}10^6$	$7{\cdot}10^6$	1	$2{\cdot}10^6$	3	182

Discussion As indicated in Fig. 2b, our implementation outperforms MultiGain on almost all benchmarks and for all types of queries and is often orders of magnitude faster. According to MultiGain's log files, the majority of its runtime is spend for solving LPs, suggesting that the better performance of Storm is likely due to the iterative approach presented in this work.

Table 1 shows that *pure LRA queries on models with millions of states can be handled*. There were no significant runtime gaps between MA and MDP models. For csn, the increased number of objectives drastically increases the overall runtime. This is partly due to our naive implementation of the geometric set representations used in Algorithm 1. Table 2 indicates that the performance and scalability for mixtures of LRA and other types of objectives is similar. One exception are queries involving time-bounded reachability on MA (e.g., dpm). Here, our implementation is based on the single-objective approach of [29] that is known to be slower than more recent methods [16,15].

Data availability The implementation, models, and log files are available at [49].

6 Conclusion

The analysis of multi-objective model checking queries involving multiple long-run average rewards can be incorporated into the framework of [28] enabling (i) the use of off-the-shelf single-objective algorithms for LRA and (ii) the combination with other kinds of objectives such as total rewards. Our experiments indicate that this approach clearly outperforms existing algorithms based on linear programming. Future work includes lifting the approach to *partially observable MDP* and *stochastic games*, potentially using ideas of [10] and [2], respectively.

References

1. Ashok, P., Chatterjee, K., Daca, P., Kretínský, J., Meggendorfer, T.: Value iteration for long-run average reward in Markov decision processes. In: CAV (1). LNCS, vol. 10426, pp. 201–221. Springer (2017). https://doi.org/10.1007/978-3-319-63387-9_10
2. Ashok, P., Chatterjee, K., Kretínský, J., Weininger, M., Winkler, T.: Approximating values of generalized-reachability stochastic games. In: LICS. pp. 102–115. ACM (2020). https://doi.org/10.1145/3373718.3394761
3. Baier, C., Bertrand, N., Dubslaff, C., Gburek, D., Sankur, O.: Stochastic shortest paths and weight-bounded properties in Markov decision processes. In: LICS. pp. 86–94. ACM (2018). https://doi.org/10.1145/3209108.3209184
4. Baier, C., Dubslaff, C., Klüppelholz, S.: Trade-off analysis meets probabilistic model checking. In: CSL-LICS. pp. 1:1–1:10. ACM (2014). https://doi.org/10.1145/2603088.2603089
5. Baier, C., Dubslaff, C., Klüppelholz, S., Daum, M., Klein, J., Märcker, S., Wunderlich, S.: Probabilistic model checking and non-standard multi-objective reasoning. In: Gnesi, S., Rensink, A. (eds.) FASE. LNCS, vol. 8411, pp. 1–16. Springer (2014). https://doi.org/10.1007/978-3-642-54804-8_1
6. Baier, C., Dubslaff, C., Korenciak, L., Kucera, A., Rehák, V.: Synthesis of optimal resilient control strategies. In: ATVA. LNCS, vol. 10482, pp. 417–434. Springer (2017). https://doi.org/10.1007/978-3-319-68167-2_27
7. 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). https://doi.org/10.1007/978-3-319-91908-9_21
8. Basset, N., Kwiatkowska, M.Z., Topcu, U., Wiltsche, C.: Strategy synthesis for stochastic games with multiple long-run objectives. In: TACAS. LNCS, vol. 9035, pp. 256–271. Springer (2015). https://doi.org/10.1007/978-3-662-46681-0_22
9. Basset, N., Kwiatkowska, M.Z., Wiltsche, C.: Compositional strategy synthesis for stochastic games with multiple objectives. Inf. Comput. **261**(Part), 536–587 (2018). https://doi.org/10.1016/j.ic.2017.09.010
10. Bork, A., Junges, S., Katoen, J., Quatmann, T.: Verification of indefinite-horizon POMDPs. In: ATVA. LNCS, vol. 12302, pp. 288–304. Springer (2020). https://doi.org/10.1007/978-3-030-59152-6_16
11. 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). https://doi.org/10.2168/LMCS-10(1:13)2014
12. Brázdil, T., Chatterjee, K., Forejt, V., Kucera, A.: MultiGain: A controller synthesis tool for MDPs with multiple mean-payoff objectives. In: TACAS. LNCS, vol. 9035, pp. 181–187. Springer (2015). https://doi.org/10.1007/978-3-662-46681-0_12
13. Brázdil, T., Chatterjee, K., Forejt, V., Kucera, A.: Trading performance for stability in Markov decision processes. J. Comput. Syst. Sci. **84**, 144–170 (2017). https://doi.org/10.1016/j.jcss.2016.09.009
14. Budde, C.E., Dehnert, C., Hahn, E.M., Hartmanns, A., Junges, S., Turrini, A.: JANI: quantitative model and tool interaction. In: TACAS (2). LNCS, vol. 10206, pp. 151–168 (2017). https://doi.org/10.1007/978-3-662-54580-5_9
15. Butkova, Y., Fox, G.: Optimal time-bounded reachability analysis for concurrent systems. In: TACAS (2). LNCS, vol. 11428, pp. 191–208. Springer (2019), https://doi.org/10.1007/978-3-030-17465-1_11

16. Butkova, Y., Hatefi, H., Hermanns, H., Krcál, J.: Optimal continuous time Markov decisions. In: ATVA. LNCS, vol. 9364, pp. 166–182. Springer (2015). https://doi.org/10.1007/978-3-319-24953-7_12
17. Butkova, Y., Wimmer, R., Hermanns, H.: Long-run rewards for Markov automata. In: TACAS (2). LNCS, vol. 10206, pp. 188–203 (2017). https://doi.org/10.1007/978-3-662-54580-5_11
18. Chatterjee, K.: Markov decision processes with multiple long-run average objectives. In: FSTTCS. LNCS, vol. 4855, pp. 473–484. Springer (2007). https://doi.org/10.1007/978-3-540-77050-3_39
19. Chatterjee, K., Doyen, L.: Perfect-information stochastic games with generalized mean-payoff objectives. In: LICS. pp. 247–256. ACM (2016). https://doi.org/10.1145/2933575.2934513
20. Chatterjee, K., Kretínská, Z., Kretínský, J.: Unifying two views on multiple mean-payoff objectives in Markov decision processes. LMCS **13**(2) (2017). https://doi.org/10.23638/LMCS-13(2:15)2017
21. Chatterjee, K., Majumdar, R., Henzinger, T.A.: Markov decision processes with multiple objectives. In: STACS. LNCS, vol. 3884, pp. 325–336. Springer (2006), https://doi.org/10.1007/11672142_26
22. Delgrange, F., Katoen, J., Quatmann, T., Randour, M.: Simple strategies in multi-objective MDPs. In: TACAS (1). LNCS, vol. 12078, pp. 346–364. Springer (2020). https://doi.org/10.1007/978-3-030-45190-5_19
23. Deng, Y., Hennessy, M.: On the semantics of Markov automata. Inf. Comput. **222**, 139–168 (2013). https://doi.org/10.1016/j.ic.2012.10.010
24. Eisentraut, C., Hermanns, H., Katoen, J., Zhang, L.: A semantics for every GSPN. In: Petri Nets. LNCS, vol. 7927, pp. 90–109. Springer (2013)
25. Eisentraut, C., Hermanns, H., Zhang, L.: On probabilistic automata in continuous time. In: LICS. pp. 342–351. IEEE Computer Society (2010). https://doi.org/10.1109/LICS.2010.41
26. Etessami, K., Kwiatkowska, M.Z., Vardi, M.Y., Yannakakis, M.: Multi-objective model checking of Markov decision processes. LMCS **4**(4) (2008). https://doi.org/10.2168/LMCS-4(4:8)2008
27. 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), https://doi.org/10.1007/978-3-642-19835-9_11
28. Forejt, V., Kwiatkowska, M.Z., Parker, D.: Pareto curves for probabilistic model checking. In: ATVA. LNCS, vol. 7561, pp. 317–332. Springer (2012). https://doi.org/10.1007/978-3-642-33386-6_25
29. Guck, D., Hatefi, H., Hermanns, H., Katoen, J., Timmer, M.: Analysis of timed and long-run objectives for Markov automata. LMCS **10**(3) (2014). https://doi.org/10.2168/LMCS-10(3:17)2014
30. Guck, D., Timmer, M., Hatefi, H., Ruijters, E., Stoelinga, M.: Modelling and analysis of Markov reward automata. In: ATVA. LNCS, vol. 8837, pp. 168–184. Springer (2014). https://doi.org/10.1007/978-3-319-11936-6_13
31. Guo, M., Zavlanos, M.M.: Probabilistic motion planning under temporal tasks and soft constraints. IEEE Trans. Autom. Control. **63**(12), 4051–4066 (2018). https://doi.org/10.1109/TAC.2018.2799561
32. Gurobi Optimization, L.: Gurobi optimizer reference manual (2020), http://www.gurobi.com
33. Hahn, E.M., Hashemi, V., Hermanns, H., Lahijanian, M., Turrini, A.: Interval Markov decision processes with multiple objectives: From robust strategies to

pareto curves. ACM Trans. Model. Comput. Simul. **29**(4), 27:1–27:31 (2019). https://doi.org/10.1145/3309683
34. 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). https://doi.org/10.1007/978-3-319-89963-3_19
35. Hartmanns, A., Junges, S., Katoen, J., Quatmann, T.: Multi-cost bounded tradeoff analysis in MDP. J. Autom. Reason. **64**(7), 1483–1522 (2020). https://doi.org/10.1007/s10817-020-09574-9
36. Hartmanns, A., Kaminski, B.L.: Optimistic value iteration. In: CAV (2). LNCS, vol. 12225, pp. 488–511. Springer (2020). https://doi.org/10.1007/978-3-030-53291-8_26
37. Hartmanns, A., Klauck, M., Parker, D., Quatmann, T., Ruijters, E.: The Quantitative Verification Benchmark Set. In: TACAS (1). LNCS, vol. 11427, pp. 344–350. Springer (2019). https://doi.org/10.1007/978-3-030-17462-0_20
38. Hatefi, H., Hermanns, H.: Model checking algorithms for Markov automata. Electron. Commun. Eur. Assoc. Softw. Sci. Technol. **53** (2012). https://doi.org/10.14279/tuj.eceasst.53.783
39. Haverkort, B.R., Hermanns, H., Katoen, J.: On the use of model checking techniques for dependability evaluation. In: SRDS. pp. 228–237. IEEE Computer Society (2000). https://doi.org/10.1109/RELDI.2000.885410
40. Hensel, C., Junges, S., Katoen, J., Quatmann, T., Volk, M.: The probabilistic model checker Storm. CoRR **abs/2002.07080** (2020)
41. 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. Int. J. Softw. Tools Technol. Transf. **20**(2), 179–194 (2018). https://doi.org/10.1007/s10009-017-0456-3
42. Kretínský, J., Meggendorfer, T.: Efficient strategy iteration for mean payoff in Markov decision processes. In: ATVA. LNCS, vol. 10482, pp. 380–399. Springer (2017). https://doi.org/10.1007/978-3-319-68167-2_25
43. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Prism-games 3.0: Stochastic game verification with concurrency, equilibria and time. In: CAV (2). LNCS, vol. 12225, pp. 475–487. Springer (2020). https://doi.org/10.1007/978-3-030-53291-8_25
44. Kwiatkowska, M., Parker, D., Wiltsche, C.: PRISM-games: verification and strategy synthesis for stochastic multi-player games with multiple objectives. STTT **20**(2), 195–210 (2018). https://doi.org/10.1007/s10009-017-0476-z
45. Kwiatkowska, M.Z., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: CAV. LNCS, vol. 6806, pp. 585–591. Springer (2011). https://doi.org/10.1007/978-3-642-22110-1_47
46. Puterman, M.L.: Markov Decision Processes. John Wiley and Sons (1994)
47. Quatmann, T., Junges, S., Katoen, J.: Markov automata with multiple objectives. In: CAV (1). LNCS, vol. 10426, pp. 140–159. Springer (2017). https://doi.org/10.1007/978-3-319-63387-9_7
48. Quatmann, T., Katoen, J.: Sound value iteration. In: CAV (1). LNCS, vol. 10981, pp. 643–661. Springer (2018). https://doi.org/10.1007/978-3-319-96145-3_37
49. Quatmann, T., Katoen, J.: Multi-objective optimization of long-run average and total rewards: Supplemental material. Zenodo (2020). https://doi.org/10.5281/zenodo.4094999

50. Randour, M., Raskin, J., Sankur, O.: Percentile queries in multi-dimensional Markov decision processes. FMSD **50**(2-3), 207–248 (2017). https://doi.org/10.1007/s10703-016-0262-7
51. Rennen, G., van Dam, E.R., den Hertog, D.: Enhancement of sandwich algorithms for approximating higher-dimensional convex Pareto sets. INFORMS J. Comput. **23**(4), 493–517 (2011). https://doi.org/10.1287/ijoc.1100.0419
52. Roijers, D.M., Scharpff, J., Spaan, M.T.J., Oliehoek, F.A., de Weerdt, M., Whiteson, S.: Bounded approximations for linear multi-objective planning under uncertainty. In: ICAPS. AAAI (2014), http://www.aaai.org/ocs/index.php/ICAPS/ICAPS14/paper/view/7929
53. Solanki, R.S., Appino, P.A., Cohon, J.L.: Approximating the noninferior set in multiobjective linear programming problems. European Journal of Operational Research **68**(3), 356 – 373 (1993). https://doi.org/10.1016/0377-2217(93)90192-P

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (https://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.