Two Views on Multiple Mean-Payoff Objectives in Markov Decision Processes

Tomáš Brázdil
Faculty of Informatics
Masaryk University
Brno, Czech Republic
brazdil@fi.muni.cz

Václav Brožek
LFCS, School of Informatics
University of Edinburgh, UK
vaclav.brozek@gmail.com

Krishnendu Chatterjee
IST Austria
Klosterneuburg, Austria
krish.chat@gmail.com

Vojtěch Forejt
Computing Laboratory
University of Oxford, UK
vojfor@comlab.ox.ac.uk

Antonín Kučera
Faculty of Informatics
Masaryk University
Brno, Czech Republic
kucera@fi.muni.cz

Abstract—**We study Markov decision processes (MDPs) with multiple limit-average (or mean-payoff) functions. We consider two different objectives, namely, expectation and satisfaction objectives. Given an MDP with k reward functions, in the expectation objective the goal is to maximize the expected limit-average value, and in the satisfaction objective the goal is to maximize the probability of runs such that the limit-average value stays above a given vector. We show that under the expectation objective, in contrast to the single-objective case, both randomization and memory are necessary for strategies, and that finite-memory randomized strategies are sufficient. Under the satisfaction objective, in contrast to the single-objective case, infinite memory is necessary for strategies, and that randomized memoryless strategies are sufficient for ε-approximation, for all $\varepsilon > 0$. We further prove that the decision problems for both expectation and satisfaction objectives can be solved in polynomial time and the trade-off curve (Pareto curve) can be ε-approximated in time polynomial in the size of the MDP and $\frac{1}{\varepsilon}$, and exponential in the number of reward functions, for all $\varepsilon > 0$. Our results also reveal flaws in previous work for MDPs with multiple mean-payoff functions under the expectation objective, correct the flaws and obtain improved results.**

I. Introduction

Markov decision processes (MDPs) are the standard models for probabilistic dynamic systems that exhibit both probabilistic and nondeterministic behaviors [14], [8]. In each state of an MDP, a controller chooses one of several actions (the nondeterministic choices), and the system stochastically evolves to a new state based on the current state and the chosen action. A reward (or cost) is associated with each transition and the central question is to find a strategy of choosing the actions that optimizes the rewards obtained over the run of the system. One classical way to combine the rewards over the run of the system is the *limit-average (or mean-payoff)* function that assigns to every run the long-run average of the rewards over the run. MDPs with single mean-payoff functions have been widely studied in literature (see, e.g., [14], [8]). In many modeling domains, however, there is not a single goal to be optimized, but multiple, potentially dependent and conflicting goals. For example, in designing a computer system, the goal is to maximize average performance while minimizing average power consumption. Similarly, in an inventory management system, the goal is to optimize several potentially dependent costs for maintaining each kind of product. These motivate the study of MDPs with multiple mean-payoff functions.

Traditionally, MDPs with mean-payoff functions have been studied with only the *expectation* objective, where the goal is to maximize (or minimize) the expectation of the mean-payoff function. There are numerous applications of MDPs with expectation objectives in inventory control, planning, and performance evaluation [14], [8]. In this work we consider both the expectation objective and also the *satisfaction* objective for a given MDP. In both cases we are given an MDP with k reward functions, and the goal is to maximize (or minimize) either the k-tuple of expectations, or the probability of runs such that the mean-payoff value stays above a given vector.

To get some intuition about the difference between the expectation/satisfaction objectives and to show that in some scenarios the satisfaction objective is preferable, consider a filehosting system where the users can download files at various speed, depending on the current setup and the number of connected customers. For simplicity, let us assume that a user has 20% chance to get a 2000kB/sec connection, and 80% chance to get a slow 20kB/sec connection. Then, the overall performance of the server can be reasonably measured by the expected amount of transferred data per user and second (i.e., the expected mean payoff) which is 416kB/sec. However, a single user is more interested in her chance of downloading the files quickly, which can be measured by the probability of establishing and maintaining a reasonably fast connection (say, $\geq$ 1500kB/sec). Hence, the system administrator may want to maximize the expected mean payoff (by changing the internal setup of the system), while a single user aims at maximizing the probability of satisfying her preferences (she can achieve that, e.g., by buying a priority access, waiting till 3 a.m., or simply connecting to a different server; obviously, she might also wish to minimize other mean payoffs such as the price per transferred bit). In other words, the expectation objective is relevant in situations when we are interested in the "average" behaviour of many instances of a given system, while the satisfaction objective is useful for analyzing and optimizing particular executions.

In MDPs with multiple mean-payoff functions, various strategies may produce incomparable solutions, and consequently there is no "best" solution in general. Informally, the set of *achievable solutions*

(i) under the expectation objective is the set of all vectors $\vec{v}$

such that there is a strategy to ensure that the expected mean-payoff value vector under the strategy is at least $\vec{v}$;

(ii) under the satisfaction objective is the set of tuples $(\nu, \vec{v})$ where $\nu \in [0,1]$ and $\vec{v}$ is a vector such that there is a strategy under which with probability at least ν the mean-payoff value vector of a run is at least $\vec{v}$.

The "trade-offs" among the goals represented by the individual mean-payoff functions are formally captured by the *Pareto curve*, which consists of all minimal tuples (wrt. componentwise ordering) that are not strictly dominated by any achievable solution. Intuitively, the Pareto curve consists of "limits" of achievable solutions, and in principle it may contain tuples that are not achievable solutions (see Section III). Pareto optimality has been studied in cooperative game theory [12] and in multi-criterion optimization and decision making in both economics and engineering [11], [17], [16].

Our study of MDPs with multiple mean-payoff functions is motivated by the following fundamental questions, which concern both basic properties and algorithmic aspects of the expectation/satisfaction objectives:

Q.1 What type of strategies is sufficient (and necessary) for achievable solutions?
Q.2 Are the elements of the Pareto curve achievable solutions?
Q.3 Is it decidable whether a given vector represents an achievable solution?
Q.4 Given an achievable solution, is it possible to compute a strategy which achieves this solution?
Q.5 Is it decidable whether a given vector belongs to the Pareto curve?
Q.6 Is it possible to compute a finite representation/approximation of the Pareto curve?

We provide comprehensive answers to the above questions, both for the expectation and the satisfaction objective. We also analyze the complexity of the problems given in Q.3–Q.6. From a practical point of view, it is particularly encouraging that most of the considered problems turn out to be solvable *efficiently*, i.e., in polynomial time. More concretely, our answers to Q.1–Q.6 are the following:

1a. For the expectation objectives, finite-memory strategies are sufficient and necessary for all achievable solutions.
1b. For the satisfaction objectives, achievable solutions require infinite memory in general, but memoryless randomized strategies are sufficient to approximate any achievable solution up to an arbitrarily small $\varepsilon > 0$.
2. All elements of the Pareto curve are achievable solutions.
3. The problem whether a given vector represents an achievable solution is solvable in polynomial time.
4.a For the expectation objectives, a strategy which achieves a given solution is computable in polynomial time.
4.b For the satisfaction objectives, a strategy which ε-approximates a given solution is computable in polynomial time.
5. The problem whether a given vector belongs to the Pareto curve is solvable in polynomial time.
6. A finite description of the Pareto curve is computable in exponential time. Further, an ε-approximate Pareto curve is computable in time which is polynomial in $1/\varepsilon$ and the size of a given MDP, and exponential in the number of mean-payoff functions.

A more detailed and precise explanation of our results is postponed to Section III.

Let us note that MDPs with multiple mean-payoff functions under the expectation objective were also studied in [4], and it was claimed that randomized memoryless strategies are sufficient for ε-approximation of the Pareto curve, for all $\varepsilon > 0$, and an NP algorithm was presented to find a randomized memoryless strategy achieving a given vector. We show with an example that under the expectation objective there exists $\varepsilon > 0$ such that randomized strategies *do require* memory for ε-approximation, and thus reveal a flaw in the earlier paper (our results not only correct the flaws of [4], but also significantly improve the complexity of the algorithm for finding a strategy achieving a given vector).

Similarly to the related papers [5], [7], [9] (see Related Work), we obtain our results by a characterization of the set of achievable solutions by a set of linear constraints, and from the linear constraints we construct witness strategies for any achievable solution. However, our approach differs significantly from the previous works. In all the previous works, the linear constraints are used to encode a *memoryless* strategy either directly for the MDP [5], or (if memoryless strategies do not suffice in general) for a finite "product" of the MDP and the specification function expressed as automata, from which the memoryless strategy is then transfered to a finite-memory strategy for the original MDP [7], [9], [6]. In our setting new problems arise. Under the expectation objective with mean-payoff function, neither is there any immediate notion of "product" of MDP and mean-payoff function and nor do memoryless strategies suffice. Moreover, even for memoryless strategies the linear constraint characterization is not straightforward for mean-payoff functions, as in the case of discounted [5], reachability [7] and total reward functions [9]: for example, in [4] even for memoryless strategies there was no linear constraint characterization for mean-payoff function and only an NP algorithm was given. Our result, obtained by a characterization of linear constraints directly on the original MDP, requires involved and intricate construction of witness strategies. Moreover, our results are significant and non-trivial generalizations of the classical results for MDPs with a single mean-payoff function, where memoryless pure optimal strategies exist, while for multiple functions both randomization and memory is necessary. Under the satisfaction objective, any finite product on which a memoryless strategy would exist is not feasible as in general witness strategies for achievable solutions may need an infinite amount of memory. We establish a correspondence between the set of achievable solutions under both types of objectives for strongly connected MDPs. Finally, we use this correspondence to obtain our result for satisfaction objectives.

Related Work. In [5] MDPs with multiple discounted reward functions were studied. It was shown that memoryless strategies suffice for Pareto optimization, and a polynomial time algorithm was given to approximate (up to a given relative error) the Pareto curve by reduction to multi-objective linear-programming and using the results of [13]. MDPs with multiple qualitative ω-regular specifications were studied in [7]. It was shown that the Pareto curve can be approximated in polynomial time; the algorithm reduces the problem to MDPs with multiple reachability specifications, which can be solved by multi-objective linear-programming. In [9], the results of [7] were extended to combine ω-regular and expected total reward objectives. MDPs with multiple mean-payoff functions under expectation objectives were considered in [4], and our results reveal flaws in the earlier paper, correct the flaws, and present significantly improved results (a polynomial time algorithm for finding a strategy achieving a given vector as compared to the previously known NP algorithm). Moreover, the satisfaction objective has not been considered in multi-objective setting before, and even in single objective case it has been considered only in a very specific setting [2].

II. Preliminaries

We use $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ to denote the sets of positive integers, integers, rational numbers, and real numbers, respectively. Given two vectors $\vec{v}, \vec{u} \in \mathbb{R}^k$, where $k \in \mathbb{N}$, we write $\vec{v} \le \vec{u}$ iff $\vec{v}_i \le \vec{u}_i$ for all $1 \le i \le k$, and $\vec{v} < \vec{u}$ iff $\vec{v} \le \vec{u}$ and $\vec{v}_i < \vec{u}_i$ for some $1 \le i \le k$.

We assume familiarity with basic notions of probability theory, e.g., *probability space*, *random variable*, or *expected value*. As usual, a *probability distribution* over a finite or countably infinite set X is a function $f : X \to [0,1]$ such that $\sum_{x \in X} f(x) = 1$. We call f *positive* if $f(x) > 0$ for every $x \in X$, *rational* if $f(x) \in \mathbb{Q}$ for every $x \in X$, and *Dirac* if $f(x) = 1$ for some $x \in X$. The set of all distributions over X is denoted by $dist(X)$.

Markov chains. A *Markov chain* is a tuple $M = (L, \to, \mu)$ where L is a finite or countably infinite set of locations, $\to \subseteq L \times (0,1] \times L$ is a transition relation such that for each fixed $\ell \in L$, $\sum_{\ell \xrightarrow{x} \ell'} x = 1$, and μ is the initial probability distribution on L.

A *run* in M is an infinite sequence $\omega = \ell_1 \ell_2 \dots$ of locations such that $\ell_i \xrightarrow{x} \ell_{i+1}$ for every $i \in \mathbb{N}$. A *finite path* in M is a finite prefix of a run. Each finite path w in M determines the set $\mathsf{Cone}(w)$ consisting of all runs that start with w. To M we associate the probability space $(\mathsf{Runs}_M, \mathcal{F}, \mathbb{P})$, where Runs_M is the set of all runs in M, $\mathcal{F}$ is the σ-field generated by all $\mathsf{Cone}(w)$, and $\mathbb{P}$ is the unique probability measure such that $\mathbb{P}(\mathsf{Cone}(\ell_1, \dots, \ell_k)) = \mu(\ell_1) \cdot \prod_{i=1}^{k-1} x_i$, where $\ell_i \xrightarrow{x_i} \ell_{i+1}$ for all $1 \le i < k$ (the empty product is equal to 1).

Markov decision processes. A *Markov decision process* (MDP) is a tuple $G = (S, A, Act, \delta)$ where S is a *finite* set of states, A is a *finite* set of actions, $Act : S \to 2^A \setminus \emptyset$ is an action enabledness function that assigns to each state s the set $Act(s)$ of actions enabled at s, and $\delta : S \times A \to dist(S)$ is a probabilistic transition function that given a state s and an action $a \in Act(s)$ enabled at s gives a probability distribution over the successor states. For simplicity, we assume that every action is enabled in exactly one state, and we denote this state $Src(a)$. Thus, henceforth we will assume that $\delta : A \to dist(S)$.

A *run* in G is an infinite alternating sequence of states and actions $\omega = s_1 a_1 s_2 a_2 \dots$ such that for all $i \ge 1$, $Src(a_i) = s_i$ and $\delta(a_i)(s_{i+1}) > 0$. We denote by Runs_G the set of all runs in G. A *finite path* of length k in G is a finite prefix $w = s_1 a_1 \dots a_{k-1} s_k$ of a run in G. For a finite path w we denote by $last(w)$ the last state of w.

A pair (T, B) with $\emptyset \ne T \subseteq S$ and $B \subseteq \bigcup_{t \in T} Act(t)$ is an *end component* of G if (1) for all $a \in B$, whenever $\delta(a)(s') > 0$ then $s' \in T$; and (2) for all $s, t \in T$ there is a finite path $\omega = s_1 a_1 \dots a_{k-1} s_k$ such that $s_1 = s$, $s_k = t$, and all states and actions that appear in w belong to T and B, respectively. (T, B) is a *maximal end component (MEC)* if it is maximal wrt. pointwise subset ordering. Given an end component $C = (T, B)$, we sometimes abuse notation by using C instead of T or B, e.g., by writing $a \in C$ instead of $a \in B$ for $a \in A$.

Strategies and plays. Intuitively, a strategy in an MDP G is a "recipe" to choose actions. Usually, a strategy is formally defined as a function $\sigma : (SA)^* S \to dist(A)$ that given a finite path w, representing the history of a play, gives a probability distribution over the actions enabled in $last(w)$. In this paper, we adopt a somewhat different (though equivalent—see [1]) definition, which allows a more natural classification of various strategy types. Let M be a finite or countably infinite set of *memory elements*. A *strategy* is a triple $\sigma = (\sigma_u, \sigma_n, \alpha)$, where $\sigma_u : A \times S \times \mathsf{M} \to dist(\mathsf{M})$ and $\sigma_n : S \times \mathsf{M} \to dist(A)$ are *memory update* and *next move* functions, respectively, and α is an initial distribution on memory elements. We require that for all $(s, m) \in S \times M$, the distribution $\sigma_n(s, m)$ assigns a positive value only to actions enabled at s. The set of all strategies is denoted by Σ (the underlying MDP G will be always clear from the context).

Let $s \in S$ be an initial state. A *play* of G determined by s and a strategy σ is a Markov chain G_s^σ (or just G^σ if s is clear from the context) where the set of locations is $S \times M \times A$, the initial distribution μ is positive only on (some) elements of $\{s\} \times M \times A$ where $\mu(s, m, a) = \alpha(m) \cdot \sigma_n(s, m)(a)$, and $(t, m, a) \xrightarrow{x} (t', m', a')$ iff

$$x \quad = \quad \delta(a)(t') \cdot \sigma_u(a, t', m)(m') \cdot \sigma_n(t', m')(a') \quad > \quad 0\,.$$

Hence, G_s^σ starts in a location chosen randomly according to α and σ_n. In a current location (t, m, a), the next action to be performed is a, hence the probability of entering t' is $\delta(a)(t')$. The probability of updating the memory to m' is $\sigma_u(a, t', m)(m')$, and the probability of selecting a' as the next action is $\sigma_n(t', m')(a')$. We assume that these choices are independent, and thus obtain the product above.

In this paper, we consider various functions over Runs_G that become random variables over $\mathsf{Runs}_{G_s^\sigma}$ after fixing some σ and s. For example, for $F \subseteq S$ we denote by $Reach(F) \subseteq \mathsf{Runs}_G$ the set of all runs reaching F. Then $Reach(F)$ naturally determines $Reach_s^\sigma(F) \subseteq \mathsf{Runs}_{G_s^\sigma}$ by simply "ignoring" the visited memory elements. To simplify and unify our notation, we write, e.g., $\mathbb{P}_s^\sigma[Reach(F)]$ instead of $\mathbb{P}_s^\sigma[\mathit{Reach}_s^\sigma(F)]$, where $\mathbb{P}_s^\sigma$ is the probability measure of the probability space associated to G_s^σ. We also adopt this notation for other events and functions, such as $\mathrm{lr}_{\inf}(\vec{r})$ or $\mathrm{lr}_{\sup}(\vec{r})$ defined in the next section, and write, e.g., $\mathbb{E}_s^\sigma[\mathrm{lr}_{\inf}(\vec{r})]$ instead of $\mathbb{E}[\mathrm{lr}_{\inf}(\vec{r})_s^\sigma]$.

Strategy types. In general, a strategy may use infinite memory, and both σ_u and σ_n may randomize. According to the use of randomization, a strategy, σ, can be classified as

- *pure* (or *deterministic*), if α is Dirac and both the memory update and the next move function give a Dirac distribution for every argument;
- *deterministic-update*, if α is Dirac and the memory update function gives a Dirac distribution for every argument;
- *stochastic-update*, if α, σ_u, and σ_n are unrestricted.

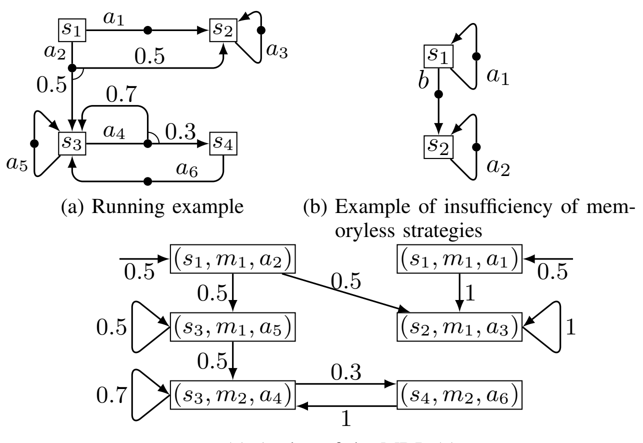
Note that every pure strategy is deterministic-update, and every deterministic-update strategy is stochastic-update. A *randomized* strategy is a strategy which is not necessarily pure. We also classify the strategies according to the size of memory they use. Important subclasses are *memoryless* strategies, in which M is a singleton, *n-memory* strategies, in which M has exactly n elements, and *finite-memory* strategies, in which M is finite. By Σ^M we denote the set of all memoryless strategies. Memoryless strategies can be specified as $\sigma : S \to \mathit{dist}(A)$. *Memoryless pure* strategies, i.e., those which are both pure and memoryless, can be specified as $\sigma : S \to A$.

For a finite-memory strategy σ, a *bottom strongly connected component* (BSCC) of G_s^σ is a subset of locations $W \subseteq S \times M \times A$ such that for all $\ell_1 \in W$ and $\ell_2 \in S \times M \times A$ we have that (i) if ℓ_2 is reachable from ℓ_1, then $\ell_2 \in W$, and (ii) for all $\ell_1, \ell_2 \in W$ we have that ℓ_2 is reachable from ℓ_1. Every BSCC W determines a unique end component $(\{s \mid (s,m,a) \in W\}, \{a \mid (s,m,a) \in W\})$ of G, and we sometimes do not strictly distinguish between W and its associated end component.

As we already noted, stochastic-update strategies can be easily translated into "ordinary" strategies of the form $\sigma : (SA)^*S \to \mathit{dist}(A)$, and vice versa (see [1]). Note that a finite-memory stochastic-update strategy σ can be easily implemented by a *stochastic finite-state automaton* that scans the history of a play "on the fly" (in fact, G_s^σ simulates this automaton). Hence, finite-memory stochastic-update strategies can be seen as natural extensions of ordinary (i.e., deterministic-update) finite-memory strategies that are implemented by deterministic finite-state automata.

A running example (I). As an example, consider the MDP $G = (S, A, \mathit{Act}, \delta)$ of Fig. 1a. Here, $S = \{s_1, \ldots, s_4\}$, $A = \{a_1, \ldots, a_6\}$, Act is denoted using the labels on lines going from actions, e.g., $\mathit{Act}(s_1) = \{a_1, a_2\}$, and δ is given by the arrows, e.g., $\delta(a_4)(s_4) = 0.3$. Note that G has four end components (two different on $\{s_3, s_4\}$) and two MECs.

Let s_1 be the initial state and $\mathsf{M} = \{m_1, m_2\}$. Consider a stochastic-update finite-memory strategy $\sigma = (\sigma_u, \sigma_n, \alpha)$ where α chooses m_1 deterministically, and $\sigma_n(m_1, s_1) = [a_1 \mapsto 0.5, a_2 \mapsto 0.5]$, $\sigma_n(m_2, s_3) = [a_4 \mapsto 1]$ and otherwise σ_n chooses self-loops. The memory update function σ_u leaves the memory intact except for the case $\sigma_u(m_1, s_3)$ where both



(a) Running example

(b) Example of insufficiency of memoryless strategies

(c) A play of the MDP (a)

Fig. 1: Example MDPs

m_1 and m_2 are chosen with probability 0.5. The play $G_{s_1}^\sigma$ is depicted in Fig. 1c.

III. Main Results

In this paper we establish basic results about Markov decision processes with *expectation* and *satisfaction* objectives specified by multiple *limit average* (or *mean payoff*) functions. We adopt the variant where rewards are assigned to edges (i.e., actions) rather than states of a given MDP.

Let $G = (S, A, \mathit{Act}, \delta)$ be a MDP, and $r : A \to \mathbb{Q}$ a *reward function*. Note that r may also take negative values. For every $j \in \mathbb{N}$, let $A_j : \mathsf{Runs}_G \to A$ be a function which to every run $\omega \in \mathsf{Runs}_G$ assigns the j-th action of ω. Since the limit average function $\mathrm{lr}(r) : \mathsf{Runs}_G \to \mathbb{R}$ given by

$$\mathrm{lr}(r)(\omega) = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} r(A_t(\omega))$$

may be undefined for some runs, we consider its lower and upper approximation $\mathrm{lr}_{\inf}(r)$ and $\mathrm{lr}_{\sup}(r)$ that are defined for *all* $\omega \in \mathsf{Runs}$ as follows:

$$\begin{aligned} \mathrm{lr}_{\inf}(r)(\omega) &= \liminf_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} r(A_t(\omega)), \\ \mathrm{lr}_{\sup}(r)(\omega) &= \limsup_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} r(A_t(\omega)). \end{aligned}$$

For a vector $\vec{r} = (r_1, \ldots, r_k)$ of reward functions, we similarly define the $\mathbb{R}^k$-valued functions

$$\begin{aligned} \mathrm{lr}(\vec{r}) &= (\mathrm{lr}(r_1), \ldots, \mathrm{lr}(r_k)), \\ \mathrm{lr}_{\inf}(\vec{r}) &= (\mathrm{lr}_{\inf}(r_1), \ldots, \mathrm{lr}_{\inf}(r_k)), \\ \mathrm{lr}_{\sup}(\vec{r}) &= (\mathrm{lr}_{\sup}(r_1), \ldots, \mathrm{lr}_{\sup}(r_k)). \end{aligned}$$

Now we introduce the expectation and satisfaction objectives determined by $\vec{r}$.

- The *expectation* objective amounts to maximizing or minimizing the expected value of $\mathrm{lr}(\vec{r})$. Since $\mathrm{lr}(\vec{r})$ may be undefined for some runs, we actually aim at maximizing the expected value of $\mathrm{lr}_{\inf}(\vec{r})$ or minimizing the expected value of $\mathrm{lr}_{\sup}(\vec{r})$ (wrt. componentwise ordering $\leq$).

- The *satisfaction* objective means maximizing the probability of all runs where $\mathrm{lr}(\vec{r})$ stays above or below a given vector $\vec{v}$. Technically, we aim at maximizing the probability of all runs where $\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}$ or $\mathrm{lr}_{\sup}(\vec{r}) \leq \vec{v}$.

The expectation objective is relevant in situtaions when we are interested in the average or aggregate behaviour of many instances of a system, and in contrast, the satisfaction objective is relevant when we are interested in particular executions of a system and wish to optimize the probability of generating the desired executions. Since $\mathrm{lr}_{\inf}(\vec{r}) = -\mathrm{lr}_{\sup}(-\vec{r})$, the problems of maximizing and minimizing the expected value of $\mathrm{lr}_{\inf}(\vec{r})$ and $\mathrm{lr}_{\sup}(\vec{r})$ are dual. Therefore, we consider just the problem of maximizing the expected value of $\mathrm{lr}_{\inf}(\vec{r})$. For the same reason, we consider only the problem of maximizing the probability of all runs where $\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}$.

If k (the dimension of $\vec{r}$) is at least two, there might be several incomparable solutions to the expectation objective; and if $\vec{v}$ is slightly changed, the achievable probability of all runs satisfying $\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}$ may change considerably. Therefore, we aim not only at constructing a particular solution, but on characterizing and approximating the whole space of *achievable solutions* for the expectation/satisfaction objective. Let $s \in S$ be some (initial) state of G. We define the sets $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ and $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ of *achievable vectors* for the expectation and satisfaction objectives as follows:

$$\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r})) = \{\vec{v} \mid \exists \sigma \in \Sigma : \ \mathbb{E}^{\sigma}_{s}[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}\},$$
$$\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r})) = \{(\nu, \vec{v}) \mid \exists \sigma \in \Sigma : \ \mathbb{P}^{\sigma}_{s}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] \ \geq \ \nu\}.$$

Intuitively, if $\vec{v}, \vec{u}$ are achievable vectors such that $\vec{v} > \vec{u}$, then $\vec{v}$ represents a "strictly better" solution than $\vec{u}$. The set of "optimal" solutions defines the *Pareto curve* for $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ and $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$. In general, the Pareto curve for a given set $Q \subseteq \mathbb{R}^k$ is the set P of all minimal vectors $\vec{v} \in \mathbb{R}^k$ such $\vec{v} \not< \vec{u}$ for all $\vec{u} \in Q$. Note that P may contain vectors that are not in Q (for example, if $Q = \{x \in \mathbb{R} \mid x < 2\}$, then $P = \{2\}$). However, every vector $\vec{v} \in P$ is "almost" in Q in the sense that for every $\varepsilon > 0$ there is $\vec{u} \in Q$ with $\vec{v} \leq \vec{u} + \vec{\varepsilon}$, where $\vec{\varepsilon} = (\varepsilon, \ldots, \varepsilon)$. This naturally leads to the notion of an *ε-approximate Pareto curve*, P_ε, which is a subset of Q such that for all vectors $\vec{v} \in P$ of the Pareto curve there is a vector $\vec{u} \in P_\varepsilon$ such that $\vec{v} \leq \vec{u} + \vec{\varepsilon}$. Note that P_ε is not unique.

A running example (II). Consider again the MDP G of Fig. 1a, and the strategy σ constructed in our running example (I). Let $\vec{r} = (r_1, r_2)$, where $r_1(a_6) = 1$, $r_2(a_3) = 2$, $r_2(a_4) = 1$, and otherwise the rewards are zero. Let

$$\omega = (s1, m1, a_2)(s_3, m_1, a_5)\big((s_3, m_2, a_4)(s_4, m_2, a_6)\big)^{\omega}$$

Then $\mathrm{lr}(\vec{r})(\omega) = (0.5, 0.5)$. Considering the expectation objective, we have that $\mathbb{E}^{\sigma}_{s_1}[\mathrm{lr}_{\inf}(\vec{r})] = (\frac{3}{52}, \frac{22}{13})$. Considering the satisfaction objective, we have that $(0.5, 0, 2) \in \mathsf{AcSt}(\vec{r})$ because $\mathbb{P}^{\sigma}_{s_1}[\mathrm{lr}_{\inf}(\vec{r}) \geq (0, 2)] = 0.5$. The Pareto curve for $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ consists of the points $\{(\frac{3}{13}x, \frac{10}{13}x + 2(1-x)) \mid 0 \leq x \leq 0.5\}$, and the Pareto curve for $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ is $\{(1, 0, 2)\} \cup \{(0.5, x, 1-x) \mid 0 < x_1 \leq \frac{10}{13}\}$.

Now we are equipped with all the notions needed for understanding the main results of this paper. Our work is motivated by the six fundamental questions given in Section I. In the next subsections we give detailed answers to these questions.

A. Expectation objectives

The answers to Q.1-Q.6 for the expectation objectives are the following:

A.1 2-memory stochastic-update strategies are sufficient for all achievable solutions, i.e., for all $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ there is a 2-memory stochastic-update strategy σ satisfying $\mathbb{E}^{\sigma}_{s}[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$.

A.2 The Pareto curve P for $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ is a subset of $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, i.e., all optimal solutions are achievable.

A.3 There is a polynomial time algorithm which, given $\vec{v} \in \mathbb{Q}^k$, decides whether $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$.

A.4 If $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, then there is a 2-memory stochastic-update strategy σ constructible in polynomial time satisfying $\mathbb{E}^{\sigma}_{s}[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$.

A.5 There is a polynomial time algorithm which, given $\vec{v} \in \mathbb{R}^k$, decides whether $\vec{v}$ belongs to the Pareto curve for $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$.

A.6 $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ is a convex hull of finitely many vectors that can be computed in exponential time. The Pareto curve for $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ is a union of all facets of $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ whose vectors are not strictly dominated by vectors of $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$. Further, an ε-approximate Pareto curve for $\mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ is computable in time polynomial in $\frac{1}{\varepsilon}$, $|G|$, and $\max_{a \in A} \max_{1 \leq i \leq k} |\vec{r}_i(a)|$, and exponential in k.

Let us note that A.1 is tight in the sense that neither memoryless randomized nor pure strategies are sufficient for achievable solutions. This is witnessed by the MDP of Fig. 1b with reward functions r_1, r_2 such that $r_i(a_i) = 1$ and $r_i(a_j) = 0$ for $i \neq j$. Consider a strategy σ which initially selects between the actions a_1 and b randomly (with probability 0.5) and then keeps selecting a_1 or a_2, whichever is available. Hence, $\mathbb{E}^{\sigma}_{s_1}[\mathrm{lr}_{\inf}((r_1, r_2))] = (0.5, 0.5)$. However, the vector $(0.5, 0.5)$ is not achievable by a strategy σ' which is memoryless or pure, because then we inevitably have that $\mathbb{E}^{\sigma'}_{s_1}[\mathrm{lr}_{\inf}((r_1, r_2))]$ is equal either to $(0, 1)$ or $(1, 0)$.

On the other hand, the 2-memory stochastic-update strategy constructed in the proof of Theorem 1 can be efficiently transformed into a finite-memory deterministic-update randomized strategy, and hence the answers A.1 and A.4 are also valid for *finite-memory deterministic-update randomized* strategies (see [1]). Observe that A.2 can be seen as a generalization of the well-known result for single payoff functions which says that finite-state MDPs with mean-payoff objectives have optimal strategies (in this case, the Pareto curve consists of a single number known as the "value"). Also observe that A.2 does *not* hold for infinite-state MDPs (a counterexample is trivial to construct).

Finally, note that if σ is a finite-memory stochastic-update strategy, then G^{σ}_{s} is a *finite-state* Markov chain. Hence, for almost all runs ω in G_s^σ we have that $\mathrm{lr}(\vec{r})(\omega)$ exists and it is equal to $\mathrm{lr}_{\inf}(\vec{r})(\omega)$. This means that there is actually no difference between maximizing the expected value of $\mathrm{lr}_{\inf}(\vec{r})$ and the expected value of $\mathrm{lr}(\vec{r})$.

B. Satisfaction objectives

The answers to Q.1-Q.6 for the satisfaction objectives are presented below.

B.1 Achievable vectors require strategies with infinite memory in general. However, memoryless randomized strategies are sufficient for ε-approximate achievable vectors, i.e., for every $\varepsilon > 0$ and $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$, there is a memoryless randomized strategy σ with

$$\mathbb{P}_s^\sigma[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}] \;\geq\; \nu - \varepsilon.$$

Here $\vec{\varepsilon} = (\varepsilon, \dots, \varepsilon)$.

B.2 The Pareto curve P for $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ is a subset of $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$, i.e., all optimal solutions are achievable.

B.3 There is a polynomial time algorithm which, given $\nu \in [0,1]$ and $\vec{v} \in \mathbb{Q}^k$, decides whether $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$.

B.4 If $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$, then for every $\varepsilon > 0$ there is a memoryless randomized strategy σ constructible in polynomial time such that $\mathbb{P}_s^\sigma[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}] \;\geq\; \nu - \varepsilon$.

B.5 There is a polynomial time algorithm which, given $\nu \in [0,1]$ and $\vec{v} \in \mathbb{R}^k$, decides whether $(\nu, \vec{v})$ belongs to the Pareto curve for $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$.

B.6 The Pareto curve P for $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ may be neither connected, nor closed. However, P is a union of finitely many sets whose closures are convex polytopes, and, perhaps surprisingly, the set $\{\nu \mid (\nu, \vec{v}) \in P\}$ is always finite. The sets in the union that gives P (resp. the inequalities that define them) can be computed. Further, an ε-approximate Pareto curve for $\mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ is computable in time polynomial in $\frac{1}{\varepsilon}$, $|G|$, and $\max_{a \in A} \max_{1 \leq i \leq k} |\vec{r}_i(a)|$, and exponential in k.

The algorithms of B.3 and B.4 are polynomial in the size of G and the size of binary representations of $\vec{v}$ and $\frac{1}{\varepsilon}$.

The result B.1 is again tight. One can show (see [1]) that memoryless pure strategies are insufficient for ε-approximate achievable vectors, i.e., there are $\varepsilon > 0$ and $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ such that for every memoryless pure strategy σ we have that $\mathbb{P}_s^\sigma[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}] \;<\; \nu - \varepsilon$.

As noted in B.1, a strategy σ achieving a given vector $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ may require infinite memory. Still, our proof of B.1 reveals a "recipe" for constructing such a σ by simulating the memoryless randomized strategies σ_ε which ε-approximate $(\nu, \vec{v})$ (intuitively, for smaller and smaller ε, the strategy σ simulates σ_ε longer and longer; the details are discussed in Section V). Hence, for almost all runs ω in G_s^σ we again have that $\mathrm{lr}(\vec{r})(\omega)$ exists and it is equal to $\mathrm{lr}_{\inf}(\vec{r})(\omega)$.

IV. Proofs for Expectation Objectives

The technical core of our results for expectation objectives is the following:

Theorem 1: Let $G = (S, A, Act, \delta)$ be a MDP, $\vec{r} = (r_1, \dots, r_k)$ a tuple of reward functions, and $\vec{v} \in \mathbb{R}^k$. Then there exists a system of linear inequalities L constructible in polynomial time such that

- every nonnegative solution of L induces a 2-memory stochastic-update strategy σ satisfying $\mathbb{E}_{s_0}^\sigma[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$;
- if $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, then L has a nonnegative solution.

$$\mathbf{1}_{s_0}(s) + \sum_{a \in A} y_a \cdot \delta(a)(s) = \sum_{a \in Act(s)} y_a + y_s \quad \text{for all } s \in S \tag{1}$$

$$\sum_{s \in S} y_s = 1 \tag{2}$$

$$\sum_{s \in C} y_s = \sum_{a \in A \cap C} x_a \quad \text{for all MEC } C \text{ of } G \tag{3}$$

$$\sum_{a \in A} x_a \cdot \delta(a)(s) = \sum_{a \in Act(s)} x_a \quad \text{for all } s \in S \tag{4}$$

$$\sum_{a \in A} x_a \cdot \vec{r}_i(a) \geq \vec{v}_i \quad \text{for all } 1 \leq i \leq n \tag{5}$$

Fig. 2: System L of linear inequalities. (We define $\mathbf{1}_{s_0}(s) = 1$ if $s = s_0$, and $\mathbf{1}_{s_0}(s) = 0$ otherwise.)

As we already noted in Section I, the proof of Theorem 1 is non-trivial and it is based on novel techniques and observations. Our results about expectation objectives are corollaries to Theorem 1 and the arguments developed in its proof. For the rest of this section, we fix an MDP G, a vector of rewards, $\vec{r} = (r_1, \dots, r_k)$, and an initial state s_0 (in the considered plays of G, the initial state is not written explicitly, unless it is different from s_0).

Consider the system L of Fig. 2 (parametrized by $\vec{v}$). Obviously, L is constructible in polynomial time. Probably most demanding are Eqns. (1) and Eqns. (4). The equations of (1) are analogous to similar equalities in [7], and their purpose is clarified at the end of the proof of Proposition 2. The meaning of Eqns. (4) is explained in Lemma 1.

As both directions of Theorem 1 are technically involved, we prove them separately as Propositions 1 and 2.

Proposition 1: Every nonnegative solution of the system L induces a 2-memory stochastic-update strategy σ satisfying $\mathbb{E}_{s_0}^\sigma[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$.

Proof of Proposition 1: First, let us consider Eqn. (4) of L. Intuitively, this equation is solved by an "invariant" distribution on actions, i.e., each solution gives frequencies of actions (up to a multiplicative constant) defined for all $a \in A$, $s \in S$, and $\sigma \in \Sigma$ by

$$\mathrm{freq}(\sigma, s, a) := \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{P}_s^\sigma[A_t = a]\,,$$

assuming that the defining limit exists (which might not be the case—cf. the proof of Proposition 2). We prove the following:

Lemma 1: Assume that assigning (nonnegative) values $\bar{x}_a$ to x_a solves Eqn. (4). Then there is a memoryless strategy ξ such that for every BSCCs D of G^ξ, every $s \in D \cap S$, and every $a \in D \cap A$, we have that $\mathrm{freq}(\xi, s, a)$ equals a common value $\mathrm{freq}(\xi, D, a) := \bar{x}_a / \sum_{a' \in D \cap A} \bar{x}_{a'}$.

A proof of Lemma 1 is given in [1]. Assume that the system L is solved by assigning nonnegative values $\bar{x}_a$ to x_a and $\bar{y}_\chi$

to y_χ where $\chi \in A \cup S$. Let ξ be the strategy of Lemma 1. Using Eqns. (1), (2), and (3), we will define a 2-memory stochastic update strategy σ as follows. The strategy σ has two memory elements, m_1 and m_2. A run of G^σ starts in s_0 with a given distribution on memory elements (see below). Then σ plays according to a suitable memoryless strategy (constructed below) until the memory changes to m_2, and then it starts behaving as ξ forever. Given a BSCC D of G^ξ, we denote by $\mathbb{P}^\sigma_{s_0}[\text{switch to } \xi \text{ in } D]$ the probability that σ switches from m_1 to m_2 while in D. We construct σ so that

$$\mathbb{P}^\sigma_{s_0}[\text{switch to } \xi \text{ in } D] \quad = \quad \sum_{a \in D \cap A} \bar{x}_a \,. \tag{6}$$

Then $\mathrm{freq}(\sigma, s_0, a) = \mathbb{P}^\sigma_{s_0}[\text{switch to } \xi \text{ in } D] \cdot \mathrm{freq}(\xi, D, a) = \bar{x}_a$. Finally, we obtain the following:

$$\mathbb{E}^\sigma_{s_0}[\mathrm{lr}_{\inf}(\vec{r}_i)] = \sum_{a \in A} \vec{r}_i(a) \cdot \bar{x}_a \,. \tag{7}$$

A complete derivation of Eqn. (7) is given in [1]. Note that the right-hand side of Eqn. (7) is greater than or equal to $\vec{v}_i$ by Inequality (5) of L.

So, it remains to construct the strategy σ with the desired "switching" property expressed by Eqn. (6). Roughly speaking, we proceed in two steps.

1. We construct a *finite-memory* stochastic update strategy $\bar{\sigma}$ satisfying Eqn. (6). The strategy $\bar{\sigma}$ is constructed so that it initially behaves as a certain finite-memory stochastic update strategy, but eventually this mode is "switched" to the strategy ξ which is followed forever.
2. The only problem with $\bar{\sigma}$ is that it may use more than two memory elements in general. This is solved by applying the results of [7] and reducing the "initial part" of $\bar{\sigma}$ (i.e., the part before the switch) into a *memoryless* strategy. Thus, we transform $\bar{\sigma}$ into an "equivalent" strategy σ which is 2-memory stochastic update.

Now we elaborate the two steps.

Step 1. For every MEC C of G, we denote by y_C the number $\sum_{s \in C} \bar{y}_s = \sum_{a \in A \cap C} \bar{x}_a$. By combining the solution of L with the results of Sections 3 and 5 of [7] (the details are given in [1]), one can construct a finite-memory stochastic-update strategy ζ which stays eventually in each MEC C with probability y_C.

The strategy $\bar{\sigma}$ works as follows. For a run initiated in s_0, the strategy $\bar{\sigma}$ plays according to ζ until a BSCC of G^ζ is reached. This means that every possible continuation of the path stays in the current MEC C of G. Assume that C has states $s_1, \ldots, s_k$. We denote by $\bar{x}_s$ the sum $\sum_{a \in Act(s)} \bar{x}_a$. At this point, the strategy $\bar{\sigma}$ changes its behavior as follows: First, the strategy $\bar{\sigma}$ strives to reach s_1 with probability one. Upon reaching s_1, it chooses (randomly, with probability $\frac{\bar{x}_{s_1}}{y_C}$) either to behave as ξ forever, or to follow on to s_2. If the strategy $\bar{\sigma}$ chooses to go on to s_2, it strives to reach s_2 with probability one. Upon reaching s_2, the strategy $\bar{\sigma}$ chooses (randomly, with probability $\frac{\bar{x}_{s_2}}{y_C - \bar{x}_{s_1}}$) either to behave as ξ forever, or to follow on to s_3, and so, till s_k. That is, the probability of switching to ξ in s_i is $\frac{\bar{x}_{s_i}}{y_C - \sum_{j=1}^{i-1} \bar{x}_{s_j}}$.

Since ζ stays in a MEC C with probability y_C, the probability that the strategy $\bar{\sigma}$ switches to ξ in s_i is equal to $\bar{x}_{s_i}$. However, then for every BSCC D of G^ξ satisfying $D \cap C \neq \emptyset$ (and thus $D \subseteq C$) we have that the strategy $\bar{\sigma}$ switches to ξ in a state of D with probability $\sum_{s \in D \cap S} \bar{x}_s = \sum_{a \in D \cap A} \bar{x}_a$. Hence, $\bar{\sigma}$ satisfies Eqn. (6).

Step 2. Now we show how to reduce the first phase of $\bar{\sigma}$ (before the switch to ξ) into a memoryless strategy, using the results of [7, Section 3]. Unfortunately, these results are not applicable directly. We need to modify the MDP G into a new MDP G' as follows: For each state s we add a new absorbing state, d_s. The only available action for d_s leads to a loop transition back to d_s with probability 1. We also add a new action, a^d_s, to every $s \in S$. The distribution associated with a^d_s assigns probability 1 to d_s.

Let us consider a finite-memory stochastic-update strategy, σ', for G' defined as follows. The strategy σ' behaves as $\bar{\sigma}$ before the switch to ξ. Once $\bar{\sigma}$ switches to ξ, say in a state s of G with probability p_s, the strategy σ' chooses the action a^d_s with probability p_s. It follows that the probability of $\bar{\sigma}$ switching in s is equal to the probability of reaching d_s in G' under σ'. By [7, Theorem 3.2], there is a memoryless strategy, σ'', for G' that reaches d_s with probability p_s. We define σ in G to behave as σ'' with the exception that, in every state s, instead of choosing an action a^d_s with probability p_s it switches to behave as ξ with probability p_s (which also means that the initial distribution on memory elements assigns p_{s_0} to m_2). Then, clearly, σ satisfies Eqn. (6) because

$$\begin{aligned}\mathbb{P}^\sigma_{s_0}[\text{switch in } D] \;&=\; \sum_{s \in D} \mathbb{P}^{\sigma''}_{s_0}\big[\text{fire } a^d_s\big] \;=\; \sum_{s \in D} \mathbb{P}^{\sigma'}_{s_0}\big[\text{fire } a^d_s\big] \\ &=\; \mathbb{P}^{\bar{\sigma}}_{s_0}[\text{switch in } D] \;=\; \sum_{a \in D \cap A} \bar{x}_a.\end{aligned}$$

This concludes the proof of Proposition 1. ∎

Proposition 2: If $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, then L has a nonnegative solution.

Proof of Proposition 2: Let $\varrho \in \Sigma$ be a strategy such that $\mathbb{E}^\varrho_{s_0}[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$. In general, the frequencies $\mathrm{freq}(\varrho, s_0, a)$ of the actions may not be well defined, because the defining limits may not exist. A crucial trick to overcome this difficulty is to pick suitable "related" values, $f(a)$, lying between $\liminf_{T\to\infty} \frac{1}{T} \sum_{t=1}^T \mathbb{P}^\varrho_{s_0}[A_t = a]$ and $\limsup_{T\to\infty} \frac{1}{T} \sum_{t=1}^T \mathbb{P}^\varrho_{s_0}[A_t = a]$, which can be safely substituted for x_a in L. Since every infinite sequence contains an infinite convergent subsequence, there is an increasing sequence of indices, $T_0, T_1, \ldots$, such that the following limit exists for each action $a \in A$

$$f(a) := \lim_{\ell\to\infty} \frac{1}{T_\ell} \sum_{t=1}^{T_\ell} \mathbb{P}^\varrho_{s_0}[A_t = a] \;\; .$$

Setting $x_a := f(a)$ for all $a \in A$ satisfies Inqs. (5) and Eqns. (4) of L. Indeed, the former follows from $\mathbb{E}^\varrho_{s_0}[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$ and the following inequality, which holds for all $1 \leq i \leq k$:

$$\sum_{a \in A} \vec{r}_i(a) \cdot f(a) \;\geq\; \mathbb{E}^\varrho_{s_0}[\mathrm{lr}_{\inf}(\vec{r}_i)] \,. \tag{8}$$

A proof of Inequality (8) is given in [1]. To prove that Eqns. (4) are satisfied, it suffices to show that for all $s \in S$ we have

$$\sum_{a\in A} f(a)\cdot\delta(a)(s) = \sum_{a\in Act(s)} f(a). \tag{9}$$

A proof of Eqn. (9) is given in [1].

Now we have to set the values for y_χ, $\chi \in A \cup S$, and prove that they satisfy the rest of L when the values $f(a)$ are assigned to x_a. Note that every run of G^ϱ eventually stays in some MEC of G (cf., e.g., [6, Proposition 3.1]). For every MEC C of G, let y_C be the probability of all runs in G^ϱ that eventually stay in C. Note that

$$\begin{aligned}\sum_{a\in A\cap C} f(a) &= \sum_{a\in A\cap C} \lim_{\ell\to\infty}\frac{1}{T_\ell}\sum_{t=1}^{T_\ell}\mathbb{P}^{\varrho}_{s_0}[A_t = a]\\ &= \lim_{\ell\to\infty}\frac{1}{T_\ell}\sum_{t=1}^{T_\ell}\sum_{a\in A\cap C}\mathbb{P}^{\varrho}_{s_0}[A_t = a] \qquad (10)\\ &= \lim_{\ell\to\infty}\frac{1}{T_\ell}\sum_{t=1}^{T_\ell}\mathbb{P}^{\varrho}_{s_0}[A_t \in C] = y_C.\end{aligned}$$

Here the last equality follows from the fact that $\lim_{\ell\to\infty}\mathbb{P}^{\varrho}_{s_0}[A_{T_\ell} \in C]$ is equal to the probability of all runs in G^ϱ that eventually stay in C (recall that almost every run stays eventually in a MEC of G) and the fact that the Cesàro sum of a convergent sequence is equal to the limit of the sequence.

To obtain y_a and y_s, we need to simplify the behavior of ϱ before reaching a MEC for which we use the results of [7]. As in the proof of Proposition 1, we first need to modify the MDP G into another MDP G' as follows: For each state s we add a new absorbing state, d_s. The only available action for d_s leads to a loop transition back to d_s with probability 1. We also add a new action, a^d_s, to every $s \in S$. The distribution associated with a^d_s assigns probability 1 to d_s. By [7, Theorem 3.2], the existence of ϱ implies the existence of a memoryless pure strategy ζ for G' such that

$$\sum_{s\in C}\mathbb{P}^{\zeta}_{s_0}[Reach(d_s)] = y_C. \tag{11}$$

Let U_a be a function over the runs in G' returning the (possibly infinite) number of times the action a is used. We are now ready to define the assignment for the variables y_χ of L.

$$\begin{aligned} y_a &:= \mathbb{E}^{\zeta}_{s_0}[U_a] && \text{for all } a \in A\\ y_s &:= \mathbb{E}^{\zeta}_{s_0}\left[U_{a^d_s}\right] = \mathbb{P}^{\zeta}_{s_0}[Reach(d_s)] && \text{for all } s \in S.\end{aligned}$$

Note that [7, Lemma 3.3] ensures that all y_a and y_s are indeed well-defined finite values, and satisfy Eqns. (1) of L. Eqns. (3) of L are satisfied due to Eqns. (11) and (10). Eqn. (11) together with $\sum_{a\in A} f(a)=1$ imply Eqn. (2) of L. This completes the proof of Proposition 2. ■

The item A.1 in Section III-A follows directly from Theorem 1. Let us analyze A.2. Suppose $\vec{v}$ is a point of the Pareto curve. Consider the system L' of linear inequalities obtained from L by replacing constants $\vec{v}_i$ in Inqs. (5) with new variables z_i. Let $Q \subseteq \mathbb{R}^n$ be the projection of the set of solutions of L' to $z_1,\ldots,z_n$. From Theorem 1 and the definition of Pareto curve, the (Euclid) distance of $\vec{v}$ to Q is 0. Because the set of solutions of L' is a closed set, Q is also closed and thus $\vec{v} \in Q$. This gives us a solution to L with variables z_i having values $\vec{v}_i$, and we can use Theorem 1 to get a strategy witnessing that $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$.

Now consider the items A.3 and A.4. The system L is linear, and hence the problem whether $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ is decidable in polynomial time by employing polynomial time algorithms for linear programming. A 2-memory stochastic-update strategy σ satisfying $\mathbb{E}^{\sigma}_{s}[\mathrm{lr}_{\inf}(\vec{r})] \geq \vec{v}$ can be computed as follows (note that the proof of Proposition 1 is *not* fully constructive, so we cannot apply this proposition immediately). First, we find a solution of the system L, and we denote by $\bar{x}_a$ the value assigned to x_a. Let $(T_1,B_1),\ldots,(T_n,B_n)$ be the end components such that $a \in \bigcup_{i=1}^{n} B_i$ iff $\bar{x}_a > 0$, and $T_1,\ldots,T_n$ are pairwise disjoint. We construct another system of linear inequalities consisting of Eqns. (1) of L and the equations $\sum_{s\in T_i} y_s = \sum_{s\in T_i}\sum_{a\in Act(s)}\bar{x}_a$ for all $1 \leq i \leq n$. Due to [7], there is a solution to this system iff in the MDP G' from the proof of Proposition 1 there is a strategy that for every i reaches d_s for $s \in T_i$ with probability $\sum_{s\in T_i}\sum_{a\in Act(s)}\bar{x}_a$. Such a strategy indeed exists (consider, e.g., the strategy σ' from the proof of Proposition 1). Thus, there is a solution to the above system and we can denote by $\hat{y}_s$ and $\hat{y}_a$ the values assigned to y_s and y_a. We define σ by

$$\begin{aligned}\sigma_n(s,m_1)(a) &= \bar{y}_a / \textstyle\sum_{a'\in Act(s)}\bar{y}_{a'}\\ \sigma_n(s,m_2)(a) &= \bar{x}_a / \textstyle\sum_{a'\in Act(s)}\bar{x}_{a'}\end{aligned}$$

and further $\sigma_u(a,s,m_1)(m_2)=y_s$, $\sigma_u(a,s,m_2)(m_2)=1$, and the initial memory distribution assigns $(1-y_{s_0})$ and y_{s_0} to m_1 and m_2, respectively. Due to [7] we have $\mathbb{P}^{\sigma}_{s_0}[\text{change memory to } m_2 \text{ in } s] = \hat{y}_s$, and the rest follows similarly as in the proof of Proposition 1.

The item A.5 can be proved as follows: To test that $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$ lies in the Pareto curve we turn the system L into a linear program LP by adding the objective to maximize $\sum_{1\leq i\leq n}\sum_{a\in A} x_a\cdot\vec{r}_i(a)$. Then we check that there is no better solution than $\sum_{1\leq i\leq n}\vec{v}_i$.

Finally, the item A.6 is obtained by considering the system L' above and computing all exponentially many vertices of the polytope of all solutions. Then we compute projections of these vertices onto the dimensions $z_1,\ldots,z_n$ and retrieve all the maximal vertices. Moreover, if for every $\vec{v} \in \{\ell\cdot\varepsilon \mid \ell \in \mathbb{Z} \wedge -M_r \leq \ell\cdot\varepsilon \leq M_r\}^k$ where $M_r = \max_{a\in A}\max_{1\leq i\leq k}|\vec{r}_i(a)|$ we decide whether $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, we can easily construct an ε-approximate Pareto curve.

V. Proofs for Satisfaction Objectives

In this section we prove the items B.1–B.6 of Section III-B. Let us fix a MDP G, a vector of rewards, $\vec{r} = (r_1,\ldots,r_k)$, and an initial state s_0. We start by assuming that the MDP G is strongly connected (i.e., (S,A) is an end component).

Proposition 3: Assume that G is strongly connected and that there is a strategy π such that $\mathbb{P}^{\pi}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] > 0$. Then the following is true.

1. There is a strategy ξ satisfying $\mathbb{P}^{\xi}_{s}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] = 1$ for all $s \in S$.

2. For each $\varepsilon>0$ there is a memoryless randomized strategy ξ_ε satisfying $\mathbb{P}^{\xi_\varepsilon}_s[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}] = 1$ for all $s \in S$.

Moreover, the problem whether there is some π such that $\mathbb{P}^{\pi}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] > 0$ is decidable in polynomial time. Strategies ξ_ε are computable in time polynomial in the size of G, the size of the binary representation of $\vec{r}$, and $\frac{1}{\varepsilon}$.

Proof: By [3], [10], $\mathbb{P}^{\pi}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] > 0$ implies that there is a strategy ξ such that $\mathbb{P}^{\xi}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] = 1$ (the details are given in [1]). This gives us item 1. of Proposition 3 and also immediately implies $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$. It follows that there are nonnegative values $\bar{x}_a$ for all $a \in A$ such that assigning $\bar{x}_a$ to x_a solves Eqns. (4) and (5) of the system L (see Fig. 2). Let us assume, w.l.o.g., that $\sum_{a\in A} \bar{x}_a = 1$.

Lemma 1 gives us a memoryless randomized strategy ζ such that for all BSCCs D of G^ζ, all $s \in D \cap S$ and all $a \in D \cap A$ we have that $\mathrm{freq}(\zeta, s, a) = \frac{\bar{x}_a}{\sum_{a\in D\cap A} \bar{x}_a}$. We denote by $\mathrm{freq}(\zeta, D, a)$ the value $\frac{\bar{x}_a}{\sum_{a\in D\cap A} \bar{x}_a}$.

Now we are ready to prove the item 2 of Proposition 3. Let us fix $\varepsilon > 0$. We obtain ξ_ε by a suitable perturbation of the strategy ζ in such a way that all actions get positive probabilities and the frequencies of actions change only slightly. There exists an arbitrarily small (strictly) positive solution x'_a of Eqns. (4) of the system L (it suffices to consider a strategy τ which always takes the uniform distribution over the actions in every state and then assign $\mathrm{freq}(\tau, s_0, a)/N$ to x_a for sufficiently large N). As the system of Eqns. (4) is linear and homogeneous, assigning $\bar{x}_a + x'_a$ to x_a also solves this system and Lemma 1 gives us a strategy ξ_ε satisfying $\mathrm{freq}(\xi_\varepsilon, s_0, a) = (\bar{x}_a + x'_a)/X$. Here $X = \sum_{a'\in A} \bar{x}_{a'} + x'_{a'} = 1 + \sum_{a'\in A} x'_{a'}$. We may safely assume that $\sum_{a'\in A} x'_{a'} \leq \frac{\varepsilon}{2\cdot M_r}$ where $M_r = \max_{a\in A} \max_{1\leq i\leq k} |\vec{r}_i(a)|$. Thus, we obtain

$$\sum_{a\in A} \mathrm{freq}(\xi_\varepsilon, s_0, a) \cdot \vec{r}_i(a) \geq \vec{v}_i - \varepsilon. \tag{12}$$

A proof of Inequality (12) is given in [1]. As G^{ξ_ε} is strongly connected, almost all runs ω of G^{ξ_ε} initiated in s_0 satisfy

$$\mathrm{lr}_{\inf}(\vec{r})(\omega) \quad = \quad \sum_{a\in A} \mathrm{freq}(\xi_\varepsilon, s_0, a) \cdot \vec{r}(a) \quad \geq \quad \vec{v} - \vec{\varepsilon}.$$

This finishes the proof of item 2.

Concerning the complexity of computing ξ_ε, note that the binary representation of every coefficient in L has only polynomial length. As $\bar{x}_a$'s are obtained as a solution of (a part of) L, standard results from linear programming imply that each $\bar{x}_a$ has a binary representation computable in polynomial time. The numbers x'_a are also obtained by solving a part of L and restricted by $\left|\sum_{a'\in A} x'_{a'}\right| \leq \frac{\varepsilon}{2\cdot M_r}$ which allows to compute a binary representation of x'_a in polynomial time. The strategy ξ_ε, defined in the proof of Proposition 3, assigns to each action only small arithmetic expressions over $\bar{x}_a$ and x'_a. Hence, ξ_ε is computable in polynomial time.

To prove that the problem whether there is some ξ such that $\mathbb{P}^{\xi}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] > 0$ is decidable in polynomial time, we show that whenever $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, then $(1, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$. This gives us a polynomial time algorithm by applying Theorem 1. Let $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$. We show that there is a strategy ξ such that $\mathbb{P}^{\xi}_{s}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] = 1$. The strategy σ needs infinite memory (an example demonstrating that infinite memory is required is given in [1]).

Since $\vec{v} \in \mathsf{AcEx}(\mathrm{lr}_{\inf}(\vec{r}))$, there are nonnegative rational values $\bar{x}_a$ for all $a \in A$ such that assigning $\bar{x}_a$ to x_a solves Eqns. (4) and (5) of the system L. Assume, without loss of generality, that $\sum_{a\in A} \bar{x}_a = 1$.

Given $a \in A$, let $I_a : A \to \{0, 1\}$ be a function given by $I_a(a) = 1$ and $I_a(b) = 0$ for all $b \neq a$. For every $i \in \mathbb{N}$, we denote by ξ_i a memoryless randomized strategy satisfying $\mathbb{P}^{\xi_i}_s\left[\mathrm{lr}_{\inf}(I_a) \geq \bar{x}_a - 2^{-i-1}\right] = 1$. Note that for every $i \in \mathbb{N}$ there is $\kappa_i \in \mathbb{N}$ such that for all $a \in A$ and $s \in S$ we get

$$\mathbb{P}^{\xi_i}_s\left[\inf_{T\geq\kappa_i} \frac{1}{T}\sum_{t=0}^{T} I_a(A_t) \geq \bar{x}_a - 2^{-i}\right] \geq 1 - 2^{-i}.$$

Now let us consider a sequence $n_0, n_1, \ldots$ of numbers where $n_i \geq \kappa_i$ and $\frac{\sum_{j<i} n_j}{n_i} \leq 2^{-i}$ and $\frac{\kappa_{i+1}}{n_i} \leq 2^{-i}$. We define ξ to behave as ξ_1 for the first n_1 steps, then as ξ_2 for the next n_2 steps, then as ξ_3 for the next n_3 steps, etc. In general, denoting by N_i the sum $\sum_{j<i} n_j$, the strategy ξ behaves as ξ_i between the N_i'th step (inclusive) and N_{i+1}'th step (non-inclusive).

Let us give some intuition behind ξ. The numbers in the sequence $n_0, n_1, \ldots$ grow rapidly so that after ξ_i is simulated for n_i steps, the part of the history when ξ_j for $j < i$ were simulated becomes relatively small and has only minor impact on the current average reward (this is ensured by the condition $\frac{\sum_{j<i} n_j}{n_i} \leq 2^{-i}$). This gives us that almost every run has infinitely many prefixes on which the average reward w.r.t. I_a is arbitrarily close to $\bar{x}_a$ infinitely often. To get that $\bar{x}_a$ is also the limit average reward, one only needs to be careful when the strategy ξ ends behaving as ξ_i and starts behaving as ξ_{i+1}, because then up to the κ_{i+1} steps we have no guarantee that the average reward is close to $\bar{x}_a$. This part is taken care of by picking n_i so large that the contribution (to the average reward) of the n_i steps according to ξ_i prevails over fluctuations introduced by the first κ_{i+1} steps according to ξ_{i+1} (this is ensured by the condition $\frac{\kappa_{i+1}}{n_i} \leq 2^{-i}$).

Let us now prove the correctness of the definition of ξ formally. We prove that almost all runs ω of G^ξ satisfy

$$\liminf_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T} I_a(A_t(\omega)) \geq \bar{x}_a.$$

Denote by E_i the set of all runs $\omega = s_0 a_0 s_1 a_1 \ldots$ of G^ξ such that for some $\kappa_i \leq d \leq n_i$ we have

$$\frac{1}{d}\sum_{j=N_i}^{N_i+d} I_a(a_j) \quad < \quad \bar{x}_a - 2^{-i}.$$

We have $\mathbb{P}^{\xi}_{s_0}[E_i] \leq 2^{-i}$ and thus $\sum_{i=1}^{\infty} \mathbb{P}^{\xi}_{s_0}[E_i] = \frac{1}{2} < \infty$. By Borel-Cantelli lemma [15], almost surely only finitely many of E_i take place. Thus, almost every run $\omega = s_0 a_0 s_1 a_1 \ldots$ of G^ξ satisfies the following: there is ℓ such that for all $i \geq \ell$ and all $\kappa_i \leq d \leq n_i$ we have that

$$\frac{1}{d}\sum_{j=N_i}^{N_i+d} I_a(a_j) \quad \geq \quad \bar{x}_a - 2^{-i}.$$

Consider $T \in \mathbb{N}$ such that $N_i \leq T < N_{i+1}$ where $i > \ell$. The following equation is proved in [1]:

$$\frac{1}{T}\sum_{t=0}^{T} I_a(a_t) \quad \geq \quad (\bar{x}_a - 2^{-i})(1 - 2^{1-i}). \qquad (13)$$

Since the above sum converges to $\bar{x}_a$ as i (and thus also T) goes to ∞, we obtain

$$\liminf_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T} I_a(a_t) \geq \bar{x}_a. \qquad \blacksquare$$

We are now ready to prove the items B.1, B.3 and B.4. Let $C_1, \ldots, C_\ell$ be all MECs of G. We say that a MEC C_i is *good for* $\vec{v}$ if there is a state s of C_i and a strategy π satisfying $\mathbb{P}^{\pi}_{s}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] > 0$ that never leaves C_i when starting in s. Using Proposition 3, we can decide in polynomial time whether a given MEC is good for a given $\vec{v}$. Let $\mathcal{C}$ be the union of all MECs good for $\vec{v}$. Then, by Proposition 3, there is a strategy ξ such that for all $s \in \mathcal{C}$ we have $\mathbb{P}^{\xi}_{s}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v}] = 1$ and for each $\varepsilon > 0$ there is a memoryless randomized strategy ξ_ε, computable in polynomial time, such that for all $s \in \mathcal{C}$ we have $\mathbb{P}^{\xi_\varepsilon}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}]$.

Consider a strategy τ, computable in polynomial time, which maximizes the probability of reaching $\mathcal{C}$. Denote by σ a strategy which behaves as τ before reaching $\mathcal{C}$ and as ξ afterwards. Similarly, denote by σ_ε a strategy which behaves as τ before reaching $\mathcal{C}$ and as ξ_ε afterwards. Note that σ_ε is computable in polynomial time.

Clearly, $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ iff $\mathbb{P}^{\tau}_{s_0}[\mathit{Reach}(\mathcal{C})] \geq \nu$ because σ achieves $\vec{v}$ with probability $\mathbb{P}^{\tau}_{s_0}[\mathit{Reach}(\mathcal{C})]$. Thus, we obtain that $\nu \leq \mathbb{P}^{\tau}_{s_0}[\mathit{Reach}(\mathcal{C})] \leq \mathbb{P}^{\xi_\varepsilon}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}]$.

Finally, to decide whether $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$, it suffices to decide whether $\mathbb{P}^{\tau}_{s_0}[\mathit{Reach}(\mathcal{C})] \geq \nu$ in polynomial time.

Now we prove item B.2. Suppose $(\nu, \vec{v})$ is a vector of the Pareto curve. We let $\mathcal{C}$ be the union of all MECs good for $\vec{v}$. Recall that the Pareto curve constructed for expectation objectives is achievable (item A.2). Due to the correspondence between AcSt and AcEx in strongly connected MDPs we obtain the following. There is $\lambda > 0$ such that for every MEC D not contained in $\mathcal{C}$, every $s \in D$, and every strategy σ that does not leave D, it is possible to have $\mathbb{P}^{\sigma}_{s}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{u}] > 0$ only if there is i such that $\vec{v}_i - \vec{u}_i \geq \lambda$, i.e., when $\vec{v}$ is greater than $\vec{u}$ by λ in some component. Thus, for every $\varepsilon < \lambda$ and every strategy σ such that $\mathbb{P}^{\sigma}_{s_0}[\mathrm{lr}_{\inf}(\vec{r}) \geq \vec{v} - \vec{\varepsilon}] \geq \nu - \varepsilon$ it must be the case that $\mathbb{P}^{\sigma}_{s_0}[\mathit{Reach}(\mathcal{C})] \geq \nu - \varepsilon$. Because for single objective reachability the optimal strategies exist, we get that there is a strategy τ satisfying $\mathbb{P}^{\tau}_{s_0}[\mathit{Reach}(\mathcal{C})] \geq \nu$, and by using methods similar to the ones of the previous paragraphs we obtain $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$.

The polynomial-time algorithm mentioned in item B.5 works as follows. First check whether $(\nu, \vec{v}) \in \mathsf{AcSt}(\mathrm{lr}_{\inf}(\vec{r}))$ and if not, return "no". Otherwise, find all MECs good for $\vec{v}$ and compute the maximal probability of reaching them from the initial state. If the probability is *strictly* greater than ν, return "no". Otherwise, continue by performing the following procedure for every $1 \leq i \leq k$, where k is the dimension of $\vec{v}$: Find all MECs C for which there is $\varepsilon > 0$ such that C is good for $\vec{u}$, where $\vec{u}$ is obtained from $\vec{v}$ by increasing the i-th component by ε (this can be done in polynomial time using linear programming). Compute the maximal probability of reaching these MECs. If for any i the probability is *at least* ν, return "no", otherwise return "yes".

The first claim of B.6 follows from Running example (II). The other claims of item B.6 require further observations and they are proved in [1].

Acknowledgements. The authors thank David Parker and Dominik Wojtczak for initial discussions on the topic. V. Brožek and V. Forejt are supported by the Newton International Fellowship. V. Forejt is also partially supported by the ESF project Games for Design and Verification and ERC Advanced Grant VERIWARE. T. Brázdil and A. Kučera are supported by the Czech Science Foundation, grant No P202/10/1469. K. Chatterjee is supported by the Austrian Science Fund (FWF) NFN Grant No S11407-N23 (RiSE) and a Microsoft Faculty Fellowship.

References

[1] T. Brázdil, V. Brožek, K. Chatterjee, V. Forejt, and A. Kučera. Two views on multiple mean-payoff objectives in Markov decision processes. Technical Report FIMU-RS-2011-02, FI MU, Brno, Czech Rep., 2011.

[2] T. Brázdil, V. Brožek, and K. Etessami. One-counter stochastic games. In K. Lodaya and M. Mahajan, editors, *FSTTCS*, volume 8 of *LIPIcs*, pages 108–119. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.

[3] K. Chatterjee. Concurrent games with tail objectives. *Theor. Comput. Sci.*, 388:181–198, December 2007.

[4] K. Chatterjee. Markov decision processes with multiple long-run average objectives. In *Proc. FSTTCS'07*, pages 473–484. Springer, 2007.

[5] K. Chatterjee, R. Majumdar, and T. Henzinger. Markov decision processes with multiple objectives. In *Proc. STACS'06*, pages 325–336. Springer, 2006.

[6] C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. *Automatic Control, IEEE Transactions on*, 43(10):1399–1418, Oct. 1998.

[7] K. Etessami, M. Kwiatkowska, M. Vardi, and M. Yannakakis. Multi-objective model checking of Markov decision processes. *LMCS*, 4(4):1–21, 2008.

[8] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer-Verlag, 1997.

[9] V. Forejt, M. Kwiatkowska, G. Norman, D. Parker, and H. Qu. Quantitative multi-objective verification for probabilistic systems. In *Proc. TACAS'11*, LNCS. Springer, 2011. To appear.

[10] H. Gimbert and F. Horn. Solving simple stochastic tail games. In M. Charikar, editor, *SODA*, pages 847–862. SIAM, 2010.

[11] J. Koski. Multicriteria truss optimization. In W. Stadler, editor, *Multicriteria Optimization in Engineering and in the Sciences*. Plenum Press, 1988.

[12] G. Owen. *Game Theory*. Academic Press, 1995.

[13] C. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *FOCS 00*, pages 86–92. IEEE Press, 2000.

[14] M. Puterman. *Markov Decision Processes*. John Wiley and Sons, 1994.

[15] H. Royden. *Real analysis*. Prentice Hall, 3rd edition, 12 Feb. 1988.

[16] R. Szymanek, F. Catthoor, and K. Kuchcinski. Time-energy design space exploration for multi-layer memory architectures. In *DATE 04*. IEEE, 2004.

[17] P. Yang and F. Catthoor. Pareto-optimization based run time task scheduling for embedded systems. In *CODES-ISSS 03*, pages 120–125. ACM, 2003.