Multi-objective discounted reward verification in graphs and MDPs

Krishnendu Chatterjee[1], Vojtěch Forejt[2], and Dominik Wojtczak[3]

[1] IST Austria
[2] Department of Computer Science, University of Oxford, UK
[3] Department of Computer Science, University of Liverpool, UK

Abstract. We study the problem of achieving a given value in Markov decision processes (MDPs) with several independent discounted reward objectives. We consider a generalised version of discounted reward objectives, in which the amount of discounting depends on the states visited and on the objective. This definition extends the usual definition of discounted reward, and allows to capture the systems in which the value of different commodities diminish at different and variable rates.
We establish results for two prominent subclasses of the problem, namely *state-discount models* where the discount factors are only dependent on the state of the MDP (and independent of the objective), and *reward-discount models* where they are only dependent on the objective (but not on the state of the MDP). For the state-discount models we use a straightforward reduction to expected total reward and show that the problem whether a value is achievable can be solved in polynomial time. For the reward-discount model we show that memory and randomisation of the strategies are required, but nevertheless that the problem is decidable and it is sufficient to consider strategies which after a certain number of steps behave in a memoryless way.
For the general case, we show that when restricted to graphs (i.e. MDPs with no randomisation), pure strategies and discount factors of the form $1/n$ where n is an integer, the problem is in PSPACE and finite memory suffices for achieving a given value. We also show that when the discount factors are not of the form $1/n$, the memory required by a strategy can be infinite.

1 Introduction

Dynamic systems with multiple objectives. Graphs are a classical model for dynamical systems with non-deterministic behaviors. Markov decision processes (MDPs) extend the model of graphs by allowing both non-deterministic as well as probabilistic behavior. An MDP is given by a finite number of states and actions, together with a transition function which to a state and an action assigns a probabilistic distribution on (successor) states. Initially, a token is placed in a distinguished initial state, and an action is chosen, possibly in a probabilistic way. The token is then moved to a state determined by the transition function, and the process continues from the beginning, starting from the successor state.

The infinite sequence of states that is produced is called a *run*, and the aim usually is to ensure that the produced runs have certain properties.

Discounted objectives. One of the most fundamental optimization objective for dynamical systems is the discounted reward objective. Given a reward function that assigns a reward to every state, and a discount factor λ which to every state assigns a number strictly smaller than 1, the discounted reward of a run is the sum of the rewards accumulated along the run, where every single reward accumulated is weighted by the product of discounts of states previously visited. The goal is to maximize the expected discounted reward.

Traditionally graphs and MDPs have been studied with the aim to optimize a unique objective. However, in most modeling domains for dynamical systems, there is not a single objective to optimize, but multiple, potentially dependent and conflicting objectives. Hence recently the problem of multi-objective optimization for dynamical systems has become an active area of research. We consider graphs and MDPs with multiple discounted reward objectives (n-dimensional discounted sum objectives). In the simplest case (which we call *uniform-discount* model) the discount factor λ is independent of the states as well as the dimension of the objectives. More generally the discount factor can depend on the states of the system (called *state-discount* model, where the discount factor at state s is $\lambda(s)$); or it can depend on the objective (*reward-discount* model, where the discount factor for dimension i is λ_i). In the most general case (*unrestricted model*) the discount factors can depend on the dimension as well as on the state of the system (discount factor for dimension i in state s is $\lambda_i(s)$).

Discounted objectives are intended to capture the fact that the reward gained soon is more valuable than a reward gained in the distant future. For example, in financial applications one often has the opportunity of putting money on a risk-free account with a small interest, and so any reward form (risky) investment needs to be discounted by the value the invested money could gain on the risk-free account. Our notion of discount allow to capture the fact that for different currencies or commodities the interest on the risk-free account can vary (reward-discount model), or that it can change depending on the circumstances (state-discount model). The unrestricted model captures both these phenomena.

Classes of strategies. In case of graphs as well as MDPs strategies are recipes that resolve the non-determinism of the system, i.e. say how actions should be selected. A strategy looks into the current execution of the system, and specifies how to resolve the non-deterministic behavior. The class of strategies can be broadly classified into *randomised* strategies that can specify a probability distribution over the non-deterministic choices, and the special case of *pure* strategies, where for every execution of the system one of the non-deterministic choices is executed (i.e., pure strategies only use Dirac distribution over the choices).

The achievability question. Given a graph or an MDP with n discounted reward objectives, and a vector $(v_1, v_2, \ldots, v_n)$, the achievability question is whether there exists a randomised strategy (resp. pure strategy if restricted to pure strategies) such that for each dimension $1 \leq i \leq n$ the expected discounted reward with respect to ith objective is at least v_i.

Our contributions. While the general problem for graphs and MDPs with multiple discounted sum objectives is challenging, we provide several partial answers. Below we present our contributions and then list some of the key open problems.

1. The decidability of MDPs with multiple discounted reward objectives under randomised strategies under uniform-discount model was established in [6]. We first observe that the problem for state-discount model can be solved by a simple reduction to total reward objectives (solved in [11]). In both the above cases randomised memoryless strategies (that do not depend on the past) are sufficient. We then consider MDPs with randomised strategies under reward-discount model. We show that in contrast to uniform-discount and state-discount model, randomised memoryless strategies are not sufficient, but it is sufficient to consider "eventually memoryless" strategies, i.e. the strategies which behave memorylessly after a fixed-length history-dependent prefix; this helps us in establishing the decidability of the achievability question.
2. For pure strategies, we consider the problem for graphs and establish decidability for the unrestricted model when the discount factors are of the form $1/n$ for natural numbers n. In the above case we show that finite-memory strategies are sufficient, whereas we also show that this is not the case when we lift the restriction on discount factors.

Key open questions: We now list some interesting open questions related to graphs and MDPs with multiple discounted sum objectives: (1) The decidability of MDPs with randomised strategies under the unrestricted model remains open. (2) The decidability of MDPs with pure strategies under the uniform-discount model with discount factors of the form $1/n$ also remains open. (3) Given a graph and a single rational discount factor (independent of the states), the decidability of existence of a pure strategy (i.e. a path) such that the discounted sum is *exactly* zero is another important open question. In fact, (3) can be reformulated in terms of achievability question for two dimensions, with one reward being the negative of the other, such that both are required to be at least zero.

Related work. The study of Markov decision processes with multiple objectives has been an area of research in applied probability theory, where it is known as *constrained MDPs* [14, 1]. The attention in the study of constrained MDPs has been focused mainly to restricted classes of MDPs, such as unichain MDPs where all states are visited infinitely often under any strategy. For general finite-state MDPs, [6] studied MDPs with multiple discounted reward functions under the uniform-discount model. It was shown that memoryless randomised strategies suffice, 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 [10]. It was shown that the Pareto curve can be approximated in polynomial time in the size of the model; the algorithm reduces the problem to MDPs with multiple reachability specifications, which can be solved by multi-objective linear programming. In [11, 12], the results of [10] were extended to combine ω-regular and expected total reward objectives. MDPs

with multiple mean-payoff functions objectives were considered in [4]. Finally, [7, 8] study multi-objective verification problem for stochastic games, which is a model extending MDPs with a second kind of nondeterminism [7, 8].

2 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 $\boldsymbol{v}, \boldsymbol{u} \in \mathbb{R}^k$, where $k \in \mathbb{N}$, we write $\boldsymbol{v} \leq \boldsymbol{u}$ iff $\boldsymbol{v}_i \leq \boldsymbol{u}_i$ for all $1 \leq i \leq k$, and $\boldsymbol{v} < \boldsymbol{u}$ iff $\boldsymbol{v} \leq \boldsymbol{u}$ and $\boldsymbol{v}_i < \boldsymbol{u}_i$ for some $1 \leq i \leq k$. Given a vector $\boldsymbol{v}$ and a number $t \in \mathbb{R}$ we use $\boldsymbol{v} + t$ for the vector $(\boldsymbol{v}_1 + t, \ldots, \boldsymbol{v}_k + t)$

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 *Dirac* if $f(x) = 1$ for some $x \in X$. The set of all distributions over X is denoted by $dist(X)$, and given two distributions d and d', we define $|d - d'| := \max_{x \in X} |d(x) - d'(x)|$.

Markov decision processes. A *Markov decision process* (MDP) is a tuple $M = (S, A, Act, \delta)$ 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 *graph* is an MDP in which $\delta(a)$ is Dirac for all $a \in A$.

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

Strategies and probabilities. Intuitively, a strategy in an MDP M is a "recipe" to choose actions. It 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 general, a strategy may use infinite memory. According to the use of randomisation, a strategy σ, can be classified as *pure* (or *deterministic*) if $\sigma(w)$ is always Dirac, and *finite-memory* if it can be defined using a finite automaton that reads a history and the choice made by σ is based solely on the state in which the automaton ends. A strategy σ is *eventually memoryless* if there is ℓ such that for all ws and $w's$ where $|ws|, |w's| \geq \ell$ we have $\sigma(ws) = \sigma(w's)$.

Each finite path w in M determines the set $\mathsf{Cone}(w)$ consisting of all runs that start with w. To M, an initial state s and σ we associate the probability space $(\mathsf{Runs}_M, \mathcal{F}, \mathbb{P}^\sigma_{M,s})$, where Runs_M is the set of all runs in M, $\mathcal{F}$ is the σ-field generated by all $\mathsf{Cone}(w)$, and $\mathbb{P}^\sigma_{M,s}$ is the unique probability measure such that $\mathbb{P}^\sigma_{M,s}(\mathsf{Cone}(s_1 a_1 \ldots s_k)) = \mu(s_1) \cdot \prod_{i=1}^{k-1} \sigma(s_1 a_1 \ldots s_i)(a_i) \cdot \delta(a_i)(s_{i+1})$, where $\mu(s_1)$ is 1 if $s_1 = s$ and 0 otherwise. We often omit the M from the subscript.

Rewards. A *(discounted) reward structure* is a tuple (r, λ) where $r : S \to \mathbb{R}$ is a *reward function* and $\lambda : S \to (0, 1)$ is a *discount factor*. A discounted reward of a run $\omega = s_1 a_1 s_2 a_2 \ldots$ is defined to be $r(\omega) = \sum_{i=1}^{\infty} r(s_i) \cdot \prod_{j=1}^{i-1} \lambda(s_j)$.

The expected discounted reward under a strategy σ is $\mathbb{E}_s^{\sigma}[r, \lambda] := \int_{\omega \in \mathsf{Runs}_M} r(\omega)\, d\mathbb{P}_s^{\sigma}$, and the optimal discounted reward is $\mathbb{E}_s^{opt}[r, \lambda] = \sup_{\sigma} \mathbb{E}_s^{\sigma}[r, \lambda]$. We also use $\mathbb{E}_{s,a}^{opt}[r, \lambda]$ to denote the optimal value after taking the action a in s, i.e. $\delta(a)(s') \cdot \mathbb{E}_{s'}^{\sigma'}[r, \lambda]$ where σ' is defined by $\sigma'(w) = \sigma(saw)$ for all w.

In this paper we deal with multi-objective rewards, i.e. we assume that we are given n objectives, each as a tuple (r_i, λ_i), together with a vector $\boldsymbol{v} \in \mathbb{R}^n$. The problem we aim to solve is to decide whether there is a strategy σ such that $\mathbb{E}_s^{\sigma}[r_i, \lambda_i] \geq \boldsymbol{v}_i$ for every $1 \leq i \leq n$. If the answer is positive, we call the vector $\boldsymbol{v}$ *achievable*. The vectors $\boldsymbol{v}$ such that $\boldsymbol{v} - \tau$ are achievable for all $\tau > 0$, but no $\boldsymbol{u} > \boldsymbol{v}$ is achievable, are called *Pareto-optimal vectors*.

As we have already mentioned, in general this problem appears to be difficult, and hence we study several interesting and useful sub-classes of the general model (which we call the *unrestricted model*).

- *State-discount model*: Here we assume that the discount factor is fully determined by the state, i.e. $\lambda_i(s) = \lambda_j(s)$ for every $1 \leq i, j \leq n$ and $s \in S$.
- *Reward-discount model*: In reward-discount model the discount factor depends only on the objective, i.e. $\lambda_i(s) = \lambda_i(s')$ for all $1 \leq i \leq n$ and $s, s' \in S$. In such case we can write just λ instead of $\lambda(s)$.

Both the above models subsume the restriction studied in [6], where the discount factor is given as one number, independent of the state or the reward. We refer to the model of [6] as the *uniform-discount* model.

3 Results for MDPs and Randomised Strategies

3.1 State-discount model

We start with presenting the most direct of our results, which concerns the solution for state-discount model. This result can be obtained by straightforwardly extending a well-known reduction from discounted rewards to total reward. Given a reward function r, the expected total reward of a run $w = s_1 a_1 s_2 \ldots$ is defined to be $\sum_{i=1}^{\infty} r(s_i)$, and the expected value $\mathbb{E}_s^{\sigma}[r]$ under a strategy σ is defined accordingly.

Theorem 1. *Let us have a state-discount model given by an MDP $M = (S, A, Act, \delta)$ and objectives $(r_1, \lambda), \ldots (r_n, \lambda)$. The problem of deciding whether a point $\boldsymbol{u}$ is achievable can be solved in polynomial time.*

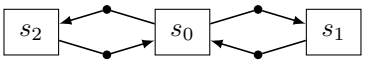
Proof. We create an MDP $M' = (S \cup \{s_\perp\}, A, Act, \delta')$ from M by adding a mandatory transition to dead state $s_\perp$ from $s \in S$ with probability $1 - \lambda(s)$. Formally, for all $s \in S$ we define δ' by $\delta'(s, a)(s') = \lambda(s) \cdot \delta(s, a)(s')$ for all

$s' \neq s_\perp$ and $\delta'(s,a)(s_\perp) = 1 - \lambda(s)$. The state $s_\perp$ has only self-loops available. We also define a reward function r' by $r'(s) = r(s)$ for all $s \in S$, and $r'(s_\perp) = 0$.

It is then easy to show that for any strategy σ we have $\mathbb{E}^\sigma_s[r, \lambda] = \mathbb{E}^\sigma_s[r']$. Thus we can use the results of [11] for multi-objective total reward to obtain the desired result. □

3.2 Reward-discount model

We now present the results related to the reward-discount model. We will show that when looking for strategies that achieve a given vector $\boldsymbol{v}$, it is sufficient to consider randomised history-dependent strategies which are eventually memoryless. To motivate this result, we first give an example where neither memoryless nor deterministic history-dependent strategies suffice. Let us have an MDP (which is in fact a graph) from the following picture:



The initial state is s_0, and there are two reward functions, r_1 and r_2, where $r_1(s_1) = r_2(s_2) = 1$ and $r_i(s_j) = 0$ for $i \neq j$. The discount factors are given by $\lambda_1 = 0.25$ and $\lambda_2 = 0.5$.

When the initial state is fixed to s_0, every strategy σ is completely determined by the probabilities $p^\sigma(i)$ of being in the state s_1 after $2 \cdot i + 1$ steps. In addition, we have

$$\mathbb{E}^\sigma_{s_0}[r_1, \lambda_1] = \sum_{i=0}^{\infty} 0.25^{2 \cdot i + 1} \cdot p^\sigma(i)$$

$$\mathbb{E}^\sigma_{s_0}[r_2, \lambda_2] = \sum_{i=0}^{\infty} 0.5^{2 \cdot i + 1} \cdot (1 - p^\sigma(i))$$

Consider the strategy σ such that $p^\sigma(0) = 1$, $p^\sigma(1) = 0.5$ and $p^\sigma(i) = 0$ for all $i > 1$. Under such strategy we get $\mathbb{E}^\sigma_{s_0}[r_1, \lambda_1] \approx 0.258$ and $\mathbb{E}^\sigma_{s_0}[r_2, \lambda_2] \approx 0.1$. Obviously, σ must be a history-dependent randomised strategy.

We show that no other strategy performs same as or better than σ. Note that any strategy σ' which satisfies $p^{\sigma'}(i) < 1$ and $p^{\sigma'}(j) > 0$ for $i < j$ can be improved as follows. Let $q = \min\{p^{\sigma'}(j), (1 - p^{\sigma'}(i))/\lambda_1^{2 \cdot (j-i)}\}$. We change the strategy σ' to the strategy σ'' defined by

$$p^{\sigma''}(\ell) = \begin{cases} p^{\sigma'}(\ell) + q \cdot \lambda_1^{2 \cdot (j-i)} & \text{for } \ell = i \\ p^{\sigma'}(\ell) - q & \text{for } \ell = j \\ p^{\sigma'}(\ell) & \text{otherwise} \end{cases}$$

We then get

$$\mathbb{E}^{\sigma''}_{s_0}[r_1, \lambda_1] = q \cdot \lambda_1^{2 \cdot (j-i)} \cdot \lambda_1^{2 \cdot i + 1} - q \cdot \lambda_1^{2 \cdot j + 1} + \mathbb{E}^{\sigma'}_{s_0}[r_1, \lambda_1] = \mathbb{E}^{\sigma'}_{s_0}[r_1, \lambda_1]$$

$$\mathbb{E}^{\sigma''}_{s_0}[r_2, \lambda_2] = -q \cdot \lambda_1^{2 \cdot (j-i)} \cdot \lambda_2^{2 \cdot i + 1} + q \cdot \lambda_2^{2 \cdot j + 1} + \mathbb{E}^{\sigma'}_{s_0}[r_2, \lambda_2] > \mathbb{E}^{\sigma'}_{s_0}[r_2, \lambda_2]$$

so σ'' performs better than σ'. Since this step can be repeated as long as there are $p^{\sigma'}(i) < 1$ and $p^{\sigma'}(j) > 0$ for some $i < j$, we get that any strategy is outperformed by some strategy $\bar{\sigma}$ for which there is ℓ and x satisfying $p^{\bar{\sigma}}(k) = 1$ for $k < \ell$, $p^{\bar{\sigma}}(\ell) = x$ and $p^{\bar{\sigma}}(k) = 0$ for $k > \ell$. However, one can easily see that any such $\bar{\sigma}$, except for σ itself, gives a worse reward than σ in the first or in the second objective, and so in particular no memoryless randomised or history-dependent deterministic strategy can outperform σ.

Now we prove that for the reward-discount model the problem whether a vector $\boldsymbol{v}$ is achievable is decidable. The intuition for the proof is the following. Given discount structures $(r_1, \lambda_1), \ldots, (r_n, \lambda_n)$ where the discount factors are pairwise different, after m steps any future contribution of ith reward will be discounted by λ_i^m. For m being sufficiently large and for $\lambda_i > \lambda_j$ we get that $\lambda_i^m \gg \lambda_j^m$, and so any reward accumulated w.r.t. jth objective becomes comparably negligible and cannot be meaningfully "traded off" for reward accumulated w.r.t. ith objective. This leads to the notion of eventually memoryless strategies, which after certain number of steps don't make any tradeoffs, but instead greedily give the highest priority to the rewards with the higher discount factors.

For the rest of this subsection we fix a reward-discount model given by an MDP $M = (S, A, Act, \delta)$ and reward structures $(r_1, \lambda_1), \ldots, (r_n, \lambda_n)$ such that $\lambda_i \geq \lambda_{i+1}$ for all $1 \leq i \leq n-1$. By U we denote a bound on maximal/minimal value of discounted rewards; for example, we can set $U := \sum_{j=0}^{\infty} \lambda_1^j \cdot r_{\max}$ where $r_{\max} = \max_{1 \leq i \leq n} \max_{s \in S} |r_i(s)|$.

The following two lemmas state some basic properties of the set of achievable points.

Lemma 1. *The set of achievable points for a reward-discount model is convex.*

Proof. When we have two achievable points $\boldsymbol{u}$ and $\boldsymbol{v}$ together with $0 \leq c \leq 1$, the point $c \cdot \boldsymbol{u} + (1-c) \cdot \boldsymbol{v}$ can be achieved by a strategy that in the first step randomly (with probability c and $1-c$) decides whether to mimic the strategy for $\boldsymbol{u}$ or $\boldsymbol{v}$, and sticks to the decision forever. □

Lemma 2. *Pareto-optimal strategies exist for a reward-discount model, i.e. limit of a sequence of achievable points is achievable.*

Proof. Let $\boldsymbol{u}$ be the limit of a sequence of achievable points, and let us have an infinite sequence of strategies $\sigma_0, \sigma_1, \ldots$ such that σ_i achieves the point $\boldsymbol{u} - \frac{1}{i}$. Let Θ_0 denote the set of the strategies in this sequence, and let $w_1, w_2, \ldots$ be the enumeration of all finite paths in the MDP. We construct infinite sets $\Theta_0, \Theta_1 \ldots$ and distributions $d_1, d_2 \ldots$ such that for every $\varepsilon > 0$ the set Θ_i contains a strategy σ that achieves value $\boldsymbol{u} - \varepsilon$, which satisfies $|\sigma(w_j) - d_j| \leq \varepsilon$ for all $j \leq i$.

We then define a strategy σ by $\sigma(w_i) = d_i$ for all $i \geq 1$. We claim that σ achieves the point $\boldsymbol{u}$. Suppose this is not the case, then there must be some ε such that σ does not achieve $\boldsymbol{u} - \varepsilon$, and let k be such that $\lambda_i^k \cdot U \geq \varepsilon/4$ for all $1 \leq i \leq n$. Note that for any strategy $\bar{\sigma}$ we have

$$\mathbb{E}^{\bar{\sigma}}_{s,a}[r_i, \lambda_i] \geq \Big(\sum_{w = s_1 a_0 s_1 \ldots s_k} \mathbb{P}^{\bar{\sigma}}_{s,a}[w] \cdot \sum_{\ell=1}^{k} \lambda_i^{\ell-1} \cdot r_i(s_\ell) \Big) - \varepsilon/4$$

Let m be an index such that all paths of length at most m are in the sequence $w_1, \ldots, w_m$. There must be a strategy $\sigma' \in \Theta_m$ such that

- $\mathbb{E}^{\sigma'}_{s,a}[r_i, \lambda_i] \geq \boldsymbol{u}_i - \varepsilon/4$
- We have $\prod_{j=0}^{m} |\sigma'(w_j) - d_j| \leq \frac{\varepsilon}{(U+\varepsilon)\cdot 4}$

Fix one such strategy σ', then we get

$$\begin{aligned}
\mathbb{E}^{\sigma}_{s}[r_i, \lambda_i] &\geq \sum_{w=s_0a_0s_1\ldots s_k} \mathbb{P}^{\sigma}_{s}[w] \cdot \sum_{\ell=0}^{k} \lambda_i^{\ell} \cdot r_i(s_\ell) - \frac{\varepsilon}{4} \\
&\geq \sum_{w=s_0a_0s_1\ldots s_k} \Big(\mathbb{P}^{\sigma'}_{s}[w] - \frac{\varepsilon}{(U+\varepsilon)\cdot 4}\Big) \cdot \sum_{\ell=0}^{k} \lambda_i^{\ell} \cdot r_i(s_\ell) - \frac{\varepsilon}{4} \\
&\geq \sum_{w=s_0a_0s_1\ldots s_k} \Big(\mathbb{P}^{\sigma'}_{s}[w] \cdot \sum_{\ell=0}^{k} \lambda_i^{\ell} \cdot r_i(s_\ell) - \frac{\varepsilon \cdot \sum_{\ell=0}^{k} \lambda_i^{\ell} \cdot r_i(s_\ell)}{(U+\varepsilon)\cdot 4}\Big) - \frac{\varepsilon}{4} \\
&\geq \sum_{w=s_0a_0s_1\ldots s_k} \mathbb{P}^{\sigma'}_{s}[w] \cdot \sum_{\ell=0}^{k} \lambda_i^{\ell} \cdot r_i(s_\ell) - \frac{\varepsilon}{2} \\
&\geq (\boldsymbol{u} - \frac{\varepsilon}{4}) - \frac{\varepsilon}{2} \quad \geq \quad \boldsymbol{u} - \frac{3\cdot\varepsilon}{4}
\end{aligned}$$

which contradicts that σ does not achieve the value $\boldsymbol{u} - \varepsilon$. □

Theorem 2. *Let let $\boldsymbol{u}$ be a Pareto point for a reward-discount model such that $\boldsymbol{u}$ is not a convex combination of any other Pareto points. Then there is a deterministic eventually memoryless strategy achieving $\boldsymbol{u}$.*

Proof. Since $\boldsymbol{u}$ is not a convex combination of other achievable points and because the set of achievable points is convex by Lemma 1 and downwards closed by definition, by the separating hyperplane theorem [3] there must be a nonnegative vector $\boldsymbol{w}$ and a number d such that for $\boldsymbol{u} \cdot \boldsymbol{w} = d$, but for all achievable points $\boldsymbol{v} \neq \boldsymbol{u}$ we have $\boldsymbol{v} \cdot \boldsymbol{w} < d$.

Note that any strategy σ that satisfies $\sum_{1\leq i\leq n} w_i \cdot \mathbb{E}^{\sigma}_{s}[r_i, \lambda_i] \geq d$ also satisfies that $\mathbb{E}^{\sigma}_{s}[r_i, \lambda_i] \geq \boldsymbol{u}_i$ for all $1 \leq i \leq n$, since otherwise it achieves the point $\boldsymbol{v} := (\mathbb{E}^{\sigma}_{s}[r_i, \lambda_i])_{1\leq i\leq n}$ with $\boldsymbol{v} \neq \boldsymbol{u}$ and $\boldsymbol{v} \cdot \boldsymbol{w} \geq d$, which is a contradiction.

Hence, it suffices to show that deterministic eventually memoryless optimal strategies σ suffice for optimising the value of $\sum_{1\leq i\leq n} w_i \cdot \mathbb{E}^{\sigma}_{s}[r_i, \lambda_i]$. Before we proceed, we show that we can simplify the problem in several respects. The first observation is that we can restrict to deterministic strategies σ (this can be proved by methods similar to the ones used in by [5], and using properties of single-objective discounted rewards). Further, we can easily preprocess the input to work with the sequence of the discount factors which is strictly decreasing: whenever $\lambda_i = \lambda_{i+1}$, then we can create an objective (r', λ_i) where $r'(s,a) = w_i \cdot r_i(s,a) + w_{i+1} \cdot r_{i+1}(s,a)$, and look for a strategy σ which satisfies $1 \cdot \mathbb{E}^{\sigma}_{s}[r', \lambda_i] + \sum_{j\neq i} w_j \cdot \mathbb{E}^{\sigma}_{s}[r_j, \lambda_j] \geq d$. At the same time, whenever some w_i is equal to 0, we can omit the ith reward since it does not affect the value $\sum_{1\leq i\leq n} w_i \cdot \mathbb{E}^{\sigma}_{s}[r_i, \lambda_i]$.

Thus, from now on we assume that $\lambda_i > \lambda_{i+1}$ for all i and that all w_i are nonzero.

In what follows we will use the notion of an optimal action subject to taking an action optimal w.r.t. different objectives. For a set $B \subseteq A$ we use $\mathbb{E}^{opt}_{s',B}[r_i, \lambda_i] := \sup_{\sigma \in \Sigma_B} \mathbb{E}^{\sigma}_{s}[r, \lambda]$ where Σ_B contains the strategies which only assign nonzero probabilities to actions from B. We put $A_0 = A$, and for all $1 \leq i \leq n$ we define A_i to contain the actions of A_{i-1} which give the best value w.r.t. (r_i, λ_i), i.e. $a \in A_i$ iff $a \in A_{i-1}$ and

$$\mathbb{E}^{opt}_{s,A_{i-1}}[r_i, \lambda_i] = r_i(s) + \sum_{s' \in S} \delta(a)(s') \cdot \lambda_i \cdot \mathbb{E}^{opt}_{s',A_{i-1}}[r_i, \lambda_i]$$

where $s = Src(a)$. Note that for every $\sigma \in \Sigma_{A_i}$ we have $\mathbb{E}^{\sigma}_{s}[r_i, \lambda_i] = \mathbb{E}^{opt}_{s,A_i}[r_i, \lambda_i]$. This is due to the properties of single-objective discounted reward in which taking any optimal action suffices to ensure the optimal values [14]. For every $a \in A_{i-1}(s)$ denote

$$v_{i,s,a} = \Big(\mathbb{E}^{opt}_{s,A_{i-1}}[r_i, \lambda_i]\Big) - \Big(r_i(s) + \sum_{s' \in S} \delta(a)(s') \cdot \lambda_i \cdot \mathbb{E}^{opt}_{s',A_{i-1}}[r_i, \lambda_i]\Big)$$

the loss when taking non-optimal optimal action (within $A_{i-1}(s)$) w.r.t. ith objective. We use $\varepsilon := \min\{v_{i,s,a} | v_{i,s,a} > 0\}$ to denote the least positive value among all $v_{i,s,a}$.

Now let k be a number such that for all i we have $w_i \cdot \varepsilon \cdot \lambda_i^k > \sum_{j=i+1}^{n} w_j \cdot U \cdot \lambda_j^k$. Such number certainly exists by the fact that $\lambda_i > \lambda_{i+1}$ and $w_i > 0$ for all i.

By induction in i we show that in order to be optimal, a strategy σ must pick actions from A_i on any path ws where $|ws| = \ell > k$. Suppose we have proved the claim for $i-1$. Note that if the strategy takes an action $a \in A_i$ in ws, then the optimal reward gained after ws is

$$\Big(\sum_{j=1}^{i-1} \lambda_j^{\ell} \cdot w_j \cdot \mathbb{E}^{opt}_{s,A_j}[r_j, \lambda_j]\Big) + \lambda_i^{\ell} \cdot w_i \cdot \mathbb{E}^{opt}_{s,A_i}[r_i, \lambda_i]\,,$$

while if a does not maximise the value, by the choice of k we get that

$$\begin{aligned}\sum_{i=1}^{n} \lambda_i^{\ell} \cdot \mathbb{E}^{\sigma}_{ws}[r_i, \lambda_i] &\leq \Big(\sum_{j=1}^{i-1} \lambda_j^{\ell} \cdot w_j \cdot \mathbb{E}^{opt}_{s,A_j}[r_j, \lambda_j]\Big) + \Big(w_i \cdot \lambda_i^{\ell} \cdot \mathbb{E}^{opt}_{s,A_i}[r_i, \lambda_i] - w_i \cdot \varepsilon \cdot \lambda_i^{\ell}\Big) \\ &\quad + \Big(\sum_{j=i+1}^{n} w_j \cdot \lambda_j^{\ell} \cdot \mathbb{E}^{opt}_{s}[r_j, \lambda_j]\Big) \\ &< \Big(\sum_{j=1}^{i-1} \lambda_j^{\ell} \cdot w_j \cdot \mathbb{E}^{opt}_{s,A_j}[r_j, \lambda_j]\Big) + w_i \cdot \lambda_i^{\ell} \cdot \mathbb{E}^{opt}_{s,A_i}[r_i, \lambda_i]\end{aligned}$$

Hence, actions from $A_i(s)$ give better value. Consequently, in order to be optimal, the strategy σ must, for any path ws with $|ws| > \ell$, only take actions from

A_n, and as we have argued above, any strategy which takes these actions suffice, meaning that we can pick an arbitrary deterministic eventually memoryless strategy which eventually plays actions from A_n. □

By Carathéodory's theorem [15], for an achievable point $\boldsymbol{u}$ there must be at most $n+1$ achievable points $\boldsymbol{v}_1 \dots \boldsymbol{v}_{n+1}$ of which $\boldsymbol{u}$ is a convex combination, and for which deterministic eventually memoryless strategies exist. This directly gives an algorithm which for any achievable point $\boldsymbol{u}$ returns "yes", and which does not halt otherwise: Set a step bound m, and try to "guess" n different deterministic strategies which become memoryless after at most m steps and which achieve points of which $\boldsymbol{u}$ (or some larger value) is a convex combination. There is only a finite number of such strategies, so we can guess by exploring all options. If the strategies are found, return that $\boldsymbol{u}$ is achievable. If the appropriate strategies can't be found, increase m and continue from the beginning.

Finally, we give an algorithm that for $\boldsymbol{u}$ which is not achievable returns "no", and does not halt otherwise. If $\boldsymbol{u}$ is not achievable, then by Lemma 2 there is τ such that $\boldsymbol{u} - \tau$ is not achievable. Let us pick m such that $\lambda_i^m \cdot U \leq \tau/3$ for all i. Then any strategy satisfies that within m steps, the reward accumulated w.r.t. ith reward is at most $\boldsymbol{u}_i - 2 \cdot \tau/3$, and we know that from that point on no more than $\tau/3$ can be accumulated. This means that m witnesses that $\boldsymbol{u}$ is not achievable. Hence, our algorithm fixes a number m and verifies whether for all strategies it is the case that there is i such that the reward accumulated up to m steps is at most $\boldsymbol{u} - 2 \cdot \lambda_i^m \cdot U/3$. This problem can be expressed using a formula over reals with addition and multiplication, for which the satisfiability problem is decidable [16]. If we find out that all strategies satisfy the condition, we return that $\boldsymbol{u}$ is not achievable, otherwise we increase m and continue.

The two above algorithms can be run in parallel, giving an algorithm that eventually terminates for any input. This allows us to establish the following corollary.

Corollary 1. *For the reward-discount model, the problem of achievability of a given vector is decidable.*

Remark 1. Our analysis does not yield any upper complexity bound. The limiting factor of our analysis is that in Theorem 1 we don't have any information about the vector $\boldsymbol{w}$. If we were able to bound the coordinates in of $\boldsymbol{w}$, then later in the proof we could give a bound on k, and hence establish the upper number of steps after which the strategies start behaving memoryless. Nevertheless, there is no obvious way how to achieve this.

4 Results for Graphs and Pure Strategies

In this section we study the decidability of the problem of achievability of a given vector for graphs. We show decidability of this problem and existence of finite-memory strategies even for the unrestricted model where the discounts depend both on the state as well as the objective function. However, we require all of

these discount functions λ_i to be *inverse-integer*, which means that for each i and $s \in S$ there is some $m \in \mathbb{N}$ such that $\lambda_i(s) = 1/m$. Under this restriction, we are able to represent the integer thresholds as periodic sequences, yielding a finite-state system. A similar approach was used in a different setting by [2].

Finally, we show that if not all of the discount reward functions are inverse-integer then any strategy for a given achievable vector may require an infinite amount of memory and we leave the decidability of that case as an open problem.

For the rest of this section, fix a graph $M = (S, A, Act, \delta)$, i.e. $\delta(a)$ is Dirac for all $a \in A$, i.e. $\delta(a)$ is Dirac for all $a \in A$.

Theorem 3. *Let $(r_1, \lambda_1), \dots (r_n, \lambda_n)$ be discounted reward structures with inverse-integer discount factors and let $\boldsymbol{v} \in \mathbb{Q}^n$ be the bound to be achieved, where n is fixed and all constants are given in unary. The problem whether there exists a pure strategy σ achieving $\boldsymbol{v}$ can be solved in polynomial time.*

Proof. We start by reducing this problem for arbitrary rational rewards functions r_i and lower threshold $\boldsymbol{v}_i$ to integer rewards and thresholds. To do so for each i we multiply the denominator of $\boldsymbol{v}_i$ by the denominators of all rational numbers $r_i(s)$ for every $s \in S$ to obtain some number d_i. We now define $r'_i(s) = d_i \cdot r_i(s)$ and $\boldsymbol{v'}_i = d_i \cdot \boldsymbol{v}_i$. It is easy to see that $r'_i(\omega) = d_i \cdot r_i(\omega)$ for any ω and so for any σ we have $\mathbb{E}^\sigma_{s_0}[r_i, \lambda_i] \geq \boldsymbol{v}_i$ iff $\mathbb{E}^\sigma_{s_0}[r'_i, \lambda_i] \geq \boldsymbol{v'}_i$. Moreover, notice that all the numbers defining r'_i and $\boldsymbol{v'}_i$ are of polynomial size, because n is fixed.

We first focus on the case $n = 1$. It is well-known that the problem in this setting can be solved in polynomial time even when all the inputs are given in binary by finding a solution to a linear program which gives the optimal value as well as the optimal deterministic strategy for the controller. However, this approach does not generalise to the $n > 1$ case. Instead, we will use automata theory based approach to solve the case $n = 1$ which works in polynomial-time if all the constants are represented in unary, and can be exponential otherwise. The advantage of this approach is that it allows us to solve the general case by using a cross product construction of the automaton generated for each of the reward structure.

Now, let the single inverse-integer reward structure be (r, λ) and the lower threshold bound be v. Also, let $a_{\max} = \lceil \max_{s \in S} r(s)/(1 - \lambda(s)) \rceil$ and $a_{\min} = \lfloor \min_{s \in S} r(s)/(1 - \lambda(s)) \rfloor$. Let $\delta(s)$ be the set of all possible successors of state s, i.e. $\delta(s) = \{s' \in S \mid \exists_{a \in Act(s)} \delta(a)(s') = 1\}$. We construct a deterministic automaton $A = (Q, \Sigma, \Delta)$ with the set of states $Q = S \times \{a_{\min}, a_{\min}+1, \dots, a_{\max}\} \cup \{\top, \bot\}$, action alphabet $\Sigma = A$ and transition function $\Delta : S \times A \to S$. The initial state of A is (s_0, v). Intuitively, if A is in state (s, x) after reading the k-th letter of the word, then x denotes how big the discounted reward of the tail of this word should be in order for the whole word to satisfy the condition $\geq v$. Formally, we define the transition function Δ as follows. If the current state is (s, x) then notice that $y := (x - r(s))/\lambda(s) \in \mathbb{Z}$, because λ is an inverse-integer discount. Now, if $y \in \{a_{\min}, a_{\min}+1, \dots, a_{\max}\}$ then the automaton for each action a such that $Src(a) = s$ have a transition to the state $(\delta(a), y)$ which reads letter a. However, if $y > a_{\max}$ then the only transition from (s, x) is to state $\top$

and if $y < a_{\min}$ then the only transition is to $\bot$. Also, from $\top$ the only transition is to $\top$ and from $\bot$ the only transition is to $\bot$. These special transitions can read any action letter.

Lemma 3. *There exists a pure strategy σ such that $\mathbb{E}^{\sigma}_{s_0}[r,\lambda] \geq v$ iff there exists an infinite word such that the corresponding run of the automaton never reaches $\top$ (a safety accepting condition).*

Proof. ($\Rightarrow$) Let σ be any pure strategy such that $\mathbb{E}^{\sigma}_{s_0}[r,\lambda] \geq v$ and let it generate a run $\omega = s_1 a_1 s_2 a_2 \ldots$, where $s_1 = s_0$. That is we have $r(\omega) = \mathbb{E}^{\sigma}_{s_0}[r,\lambda] \geq v$. Let $(s_1,x_1)(s_2,x_2)\ldots$ be the sequence of states visited by A while reading the word $a_1 a_2 \ldots$, where $s_1 = s_0$ and $x_1 = v$. Let us modify this sequence by replacing $\top$ and $\bot$ states by the actual states of the form (s,x) that would have be visited if these special states $\top$ and $\bot$ were not present. In other words, $s_{k+1} = \delta(a_k)$ and $x_{k+1} = (x_k - r(s_k))/\lambda(s_k)$ hold for any k even if $x_k \notin \{a_{\min},\ldots,a_{\max}\}$. We will show that it cannot be the case that $x_k > a_{\max}$ for some k and as a result $\top$ did not occur in the original sequence.

The proof is by contradiction; let l be the first step for which $x_l > a_{\max} = \max_s r(s)/(1-\lambda(s))$. It is easy to see that there has to be some constant $\alpha > 1$ such that $x_l \geq \max_s r(s)/(1-\alpha\cdot\lambda(s))$. In fact, we can pick α to be $\min_s 1/\lambda(s) - r(s)/(\lambda(s)\cdot x_l)$. This is because from the definition of α for any s we would then have $r(s)/(1-\alpha\cdot\lambda(s)) \leq r(s)/(1-(1-r(s)/x_l)) = x_l$. Note that $\alpha > 1$ because from the definition of $a_{\max}$ if $x_l > a_{\max}$ then $(x_l - r(s))/\lambda(s) > x_l$ for all s. Finally, we get $x_{l+1}/x_l = (x_l - r(s_l))/(x_l\cdot\lambda(s_l)) = 1/\lambda(s_l) - r(s_l)/(\lambda(s_l)\cdot x_l) \geq \alpha$. Therefore $x_{l+1} \geq \alpha x_l \geq x_l$, but the expression for α increases as x_l increases and so for any k we have $x_{l+k} \geq \alpha x_{l+k-1} \geq \ldots \geq \alpha^k \cdot x_l$. In conclusion $x_k \to \infty$ as $k \to \infty$.

On the other hand, for any k let us denote the discounted-reward of ω until step k by $r(\omega_k) := \sum_{i=1}^{k} r(s_i)\cdot\prod_{j=1}^{i-1}\lambda(s_j)$. Notice that $x_1 = v$, $x_2 = (v - r(s_1))/\lambda(s_1) = v/\lambda(s_1) - r(s_1)/\lambda(s_1)$, $x_3 = v/(\lambda(s_1)\cdot\lambda(s_2)) - r(s_1)/(\lambda(s_1)\cdot\lambda(s_2)) - r(s_2)/\lambda(s_2)$, and by induction $x_k = v/\prod_{j=1}^{k-1}\lambda(s_j) - \sum_{i=1}^{k-1} r(s_i)/\prod_{j=i}^{k-1}\lambda(s_j)$. In other words, $x_k\cdot\prod_{j=1}^{k-1}\lambda(s_j) = v - r(\omega_k) \leq r(\omega) - r(\omega_k) = \sum_{i=k+1}^{\infty} r(s_i)\cdot\prod_{j=1}^{i-1}\lambda(s_j)$, which means $x_k \leq \lambda(s_k)\cdot\sum_{i=k+1}^{\infty} r(s_i)\cdot\prod_{j=k+1}^{i-1}\lambda(s_j) \leq (\max_s\lambda(s)\cdot\max_s r(s))/(1-\max_s\lambda(s))$, but the right hand side is a constant while we just showed that $x_k \to \infty$ as $k \to \infty$; a contradiction.

($\Leftarrow$) Let $\omega = a_1 a_2 \ldots$ be any infinite word for which A does not enter state $\top$ and let $(s_1,x_1)(s_2,x_2)\ldots$ be the sequence of states visited by A along this word. If $\bot$ state is in this sequence (and as a result only $\bot$ occurs from that moment on), then note that $(a_{\min} - r(s))/\lambda(s) < a_{\min}$ for any $s \in S$ and so when A is starting at any state (s,x) such that $x < a_{\min}$, no state (s',x') with $x' > a_{\max}$ can be reached. Therefore, $x_k < a_{\max}$ holds for every k in either case. As we will now show, this condition suffices for the word ω to be accepted and so the automaton should accept any word once it enters $\bot$ no matter what the tail of that word is. Again, let us denote the discounted-reward of ω until step k by $r(\omega_k) := \sum_{i=1}^{k} r(s_i)\cdot\prod_{j=1}^{i-1}\lambda(s_j)$ and notice that $x_k\prod_{j=1}^{k-1}\lambda(s_j) = v - r(\omega_k)$. We know that for any k we have $x_k \leq a_{\max}$ which means that $v - r(\omega_k) \leq$

$a_{\max} \cdot \prod_{j=1}^{k-1} \lambda(s_j)$, and so $v - a_{\max} \cdot \prod_{j=1}^{k-1} \lambda(s_j) \leq r(\omega_k)$. Now taking the limit as $k \to \infty$ we get that $v \leq r(\omega)$, because $a_{\max}$ is a constant, and $\prod_{j=1}^{k-1} \lambda(s_j) \to 0$ and $r(\omega_k) \to r(\omega)$ as $k \to \infty$. □

Now, to deal with more than one reward structures, we will use a cross product of all the automata constructed for each of the reward structures. Let $A_i = (Q_i, \Sigma_i, \Delta_i)$ be the automaton generated using the previous construction for the reward structure (r_i, λ_i). The cross-product automaton A' is defined as follows $A' = (Q', \Sigma', \Delta')$, where the set of states $Q' = \prod_i Q_i$, the letter alphabet $\Sigma' = A$ and the transition function Δ' is defined as follows: $\Delta((s_1, \ldots, s_n), a) = (\Delta_1(s_1, a), \Delta_2(s_2, a), \ldots, \Delta_n(s_n, a))$. The initial state of the automata A' is $s'_0 = ((s_0, \boldsymbol{v}_1), (s_0, \boldsymbol{v}_2), \ldots, (s_0, \boldsymbol{v}_n))$. Notice that the size of the automata A' is polynomial in the size of the original graph, because n is fixed. Technically we can further reduce the size of A by noticing that any reachable state $((s_1, \boldsymbol{v}_1), (s_2, \boldsymbol{v}_2), \ldots, (s_n, \boldsymbol{v}_n))$ satisfies $s_1 = s_2 = \ldots = s_n$. A run of A' is accepting if it does not reach the $\top$ state in any of the component automata. This is a safety condition and deciding the existence of a safe run can be done in time linear in the size of the automaton [9].

□

Theorem 4. *Let $(r_1, \lambda_1), \ldots (r_n, \lambda_n)$ be inverse-integer reward structures given in binary, and let $\boldsymbol{v} \in \mathbb{Q}^n$ be a bound. The problem whether there exists a pure strategy σ achieving $\boldsymbol{v}$ can be solved in* PSPACE. *If such σ exists, then there is also a finite-memory one.*

Proof. We use the same cross-product automaton A' construction as in the proof of Theorem 3. The size of the automata A' is now exponential in the size of the original graph, but every state can be represented using polynomial space, because we just need to remember the current state, (s, x), of each component automaton and the value of x has at most polynomially many bits. Also the number of states of A' can be represented using polynomially many bits. Notice that A' has an accepting run iff there exists a cycle starting s'_0 which never reaches a $\top$ state. So a simple NPSPACE (which is =PSPACE) algorithm can be given as follows: we simply simulate the transitions while counting the number of steps made, and we stop and reject if we reach $\top$, and stop and accept if we already made more steps than the number of states in A'. The latter implies that we already formed a cycle and a run that never reaches $\top$ exists. A safe strategy can be reconstructed from the accepting path of our algorithm by looking at the transitions taken, and repeating a pattern based on them forever. Notice that the size of the memory such a strategy requires is at most equal to the number of states in the automaton A'. □

Proposition 1. *There are uniform-discount rewards $(r_1, \lambda), (r_2, \lambda)$ that are not inverse-integer such that for some $\boldsymbol{v} \in \mathbb{Q}^2$: any pure strategy σ which achieves $\boldsymbol{v}$ requires infinite memory.*

Proof. We will show this already for a system with just two states. Let us denote the states by s_1 and s_2 and actions by a_1 and a_2. The uniform discount is set to

$\lambda = \frac{2}{3}$. From any state the action a_i leads to state s_i. We set $r_1(s_1) = r_2(s_1) = 0$, $r_1(s_2) = 1$, $r_2(s_2) = -1$, and the thresholds to $\boldsymbol{v}_1 = 3/2$ and $\boldsymbol{v}_2 = -3/2$. Notice that since $r_1(\omega) + r_2(\omega) = 0$ for any infinite path ω, the conditions $\mathbb{E}^{\sigma}_{s_0}[r_i, \lambda_i] \geq \boldsymbol{v}_i$ are satisfied iff $r_1(\omega) = 3/2$. Because we are only looking at finite-memory pure strategies σ, we can represent the unique run ω it generates by a string $\omega' = b_1 \dots b_k \cdot (c_1 \dots c_l)^{\omega}$, where $k \geq 0$, $l \geq 1$, $b_i, c_i \in \{0, 1\}$ and the j-th position in ω' is 0 iff a_1 is used at the j-th step of the run ω and otherwise it is equal to 1. Notice that we have

$$r_1(\omega) = b_1 + \frac{2}{3}b_2 + \ldots + \left(\frac{2}{3}\right)^{k-1} b_k + \left(\frac{2}{3}\right)^{k} \frac{1}{1 - (\frac{2}{3})^l} \left(c_1 + \frac{2}{3}c_2 + \ldots + \left(\frac{2}{3}\right)^{l-1} c_l \right)$$

and if we multiply both sides by $3^k \cdot (3^l - 2^l)$ the right-hand side will be an integer, but the left-hand side will not be an integer as $r_1(\omega) = 3/2$ and $3^k \cdot (3^l - 2^l)$ is an odd number for $l \geq 1$.

Finally, we just need to show now that there exists an infinite sequence $\omega = b_1 b_2 \dots$ such that $b_i \in \{0, 1\}$ for all i and

$$r_1(\omega) = b_1 + \frac{2}{3}b_2 + \ldots + \left(\frac{2}{3}\right)^{k-1} b_k + \ldots = \frac{3}{2}. \tag{1}$$

In other words there exists a pure winning strategy σ which uses an infinite amount of memory. We use the following algorithm to generate this sequence ω. We initialise $x := \frac{3}{2}$. At the k-th step, starting with $k := 1$, if $x \geq 1$ then we set $b_k := 1$ and update $x := \frac{3}{2}(x - 1)$, and otherwise set $b_k := 0$ and update $x := \frac{3}{2}x$. We then move to the next step $k := k + 1$. Intuitively the value of x at the k-th step tell us what the value of $\left(\frac{2}{3}\right)^k b_{k+1} + \ldots$ should be in order for the total discount reward to be equal to $\frac{3}{2}$. Also, based on the rules how x gets updated, at any step we have $x \geq 0$ and $x \leq \frac{3}{2}$. From this fact and using similar reasoning to Lemma 3, we can show that this process will generate an infinite sequence $\omega = b_1 b_2 \dots$ such $b_i \in \{0, 1\}$ for all i and the condition (1) holds.

5 Conclusions

We have studied MDPs with multiple discounted objectives. We have extended the results of [6] by considering a more expressible class of models, which allow to specify discount factors dependent on states and/or objectives.

As we have already mentioned in the introduction, there are several interesing and challenging open questions. Except for these, it is also of interest to obtain a complexity bound for the reward-discound model.

Acknowledgements The authors are grateful to Aisis Šimaitis for his stimulating discussions on the topic. K. Chatterjee is supported by Austrian Science Fund (FWF) Grant No P 23499-N23, FWF NFN Grant No S11407-N23 (RiSE), ERC Start grant (279307: Graph Games), and Microsoft faculty fellows

award. V. Forejt is also affiliated with Faculty of Informatics, Masaryk University in Brno, and was supported by a Royal Society Newton Fellowship and EPSRC project EP/J012564/1. D. Wojtczak is supported by the grant EPSRC EP/H046623/1.

References

1. E. Altman. *Constrained Markov Decision Processes (Stochastic Modeling)*. Chapman & Hall/CRC, 1999.
2. U. Boker and T. A. Henzinger. Determinizing discounted-sum automata. In *CSL*, pages 82–96, 2011.
3. S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge Univ. Press, 2004.
4. T. Brázdil, V. Brozek, K. Chatterjee, V. Forejt, and A. Kucera. Two views on multiple mean-payoff objectives in markov decision processes. In *LICS*, pages 33–42. IEEE Computer Society, 2011.
5. K. Chatterjee, L. Doyen, H. Gimbert, and T. A. Henzinger. Randomness for free. In P. Hlinený and A. Kucera, editors, *MFCS*, volume 6281 of *Lecture Notes in Computer Science*, pages 246–257. Springer, 2010.
6. K. Chatterjee, R. Majumdar, and T. A. Henzinger. Markov decision processes with multiple objectives. In B. Durand and W. Thomas, editors, *STACS*, volume 3884 of *Lecture Notes in Computer Science*, pages 325–336. Springer, 2006.
7. T. Chen, V. Forejt, M. Kwiatkowska, A. Simaitis, and C. Wiltsche. On stochastic games with multiple objectives. In *Proc. 38th International Symposium on Mathematical Foundations of Computer Science (MFCS'13)*. Springer, 2013. To appear.
8. T. Chen, M. Kwiatkowska, A. Simaitis, and C. Wiltsche. Synthesis for multi-objective stochastic games: An application to autonomous urban driving. In *In Proc. 10th International Conference on Quantitative Evaluation of SysTems (QEST 2013)*. IEEE CS Press, 2013. To appear.
9. E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. The MIT Press, 1999.
10. K. Etessami, M. Kwiatkowska, M. Vardi, and M. Yannakakis. Multi-objective model checking of Markov decision processes. *Logical Methods in Computer Science*, 4(4):1–21, 2008.
11. V. Forejt, M. Kwiatkowska, G. Norman, D. Parker, and H. Qu. Quantitative multi-objective verification for probabilistic systems. In P. Abdulla and K. Leino, editors, *Proc. 17th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'11)*, volume 6605 of *LNCS*, pages 112–127. Springer, 2011.
12. V. Forejt, M. Kwiatkowska, and D. Parker. Pareto curves for probabilistic model checking. In S. Chakraborty and M. Mukund, editors, *Proc. 10th International Symposium on Automated Technology for Verification and Analysis (ATVA'12)*, volume 7561 of *LNCS*, pages 317–332. Springer, 2012.
13. C. H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *FOCS*, pages 86–92. IEEE Computer Society, 2000.
14. M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 1994.
15. R. Rockafellar. *Convex Analysis*. Princeton University Press, 1997.
16. A. Tarski. *A decision method for elementary algebra and geometry*. Rand report. Rand Corporation, 1948.