REINFORCEMENT LEARNING WITH EX-POST MAX-MIN FAIRNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider reinforcement learning with vectorial rewards, where the agent receives a vector of $K \geq 2$ different types of rewards at each time step. The agent aims to maximize the minimum total reward among the K reward types. Different from existing works that focus on maximizing the minimum expected total reward, i.e. *ex-ante max-min fairness*, we maximize the expected minimum total reward, i.e. *ex-post max-min fairness*. Through an example and numerical experiments, we show that the optimal policy for the former objective generally does not converge to optimality under the latter, even as the number of time steps T grows. Our main contribution is a novel algorithm, Online-ReOpt, that achieves near-optimality under our objective, assuming an optimization oracle that returns a near-optimal policy given any scalar reward. The expected objective value under Online-ReOpt is shown to converge to the asymptotic optimum as T increases. Finally, we propose offline variants to ease the burden of online computation in Online-ReOpt, and we propose generalizations from the max-min objective to concave utility maximization.

1 INTRODUCTION

The prevailing paradigm in reinforcement learning (RL) concerns the maximization of a *single scalar reward*. On one hand, optimizing a single scalar reward is sufficient for modeling simple tasks. On the other hand, in many complex tasks there are often multiple, potentially competing, rewards to be maximized. Expressing the objective function as a single linear combination of the rewards can be constraining and insufficiently expressive for the nature of these complex tasks. In addition, a suitable choice of the linear combination is often not clear a priori.

In this work, we consider the reinforcement learning with max-min fairness (RL-MMF) problem. The agent accumulates a vector of $K \geq 1$ time-average rewards $\bar{V}_{1:T} = (\bar{V}_{1:T,k})_{k=1}^{K} \in \mathbb{R}^K$ in T time steps, and aims to maximize $\mathbb{E}[\min_{k\in\{1,\ldots,K\}} \bar{V}_{1:T,k}]$. The maximization objective represents *ex-post* max-min fairness, in contrast to the objective of *ex-ante* max-min fairness by maximizing $\min_{k\in\{1,\ldots,K\}} \mathbb{E}[\bar{V}_{1:T,k}]$.

Our main contributions are the design and analysis of the Online-ReOpt algorithm, which achieves near-optimality for the *ex-post* max-min fairness objective. More specifically, the objective under Online-ReOpt converges to the optimum as T increases. Our algorithm design involves a novel adaptation of the multiplicative weight update method (Arora et al., 2012), in conjunction with a judiciously designed re-optimization schedule. The schedule ensures that the agent adapts his decision to the total vectorial reward collected at a current time point, while allowing enough time for the currently adopted policy to converge before switching to another policy.

En route, we highlight crucial differences between the *ex-ante* and *ex-post* max-min fairness objectives, by showing that an optimal algorithm for the former needs not converge to the optimality even when T increases. Finally, our results are extended to the case of maximizing $\mathbb{E}[g(\bar{V}_{1:T})]$, where g is a Lipschitz continuous and concave reward function.

2 RELATED WORKS

The Reinforcement Learning with Max-Min Fairness (RL-MMF) problem described is related to an emerging body of research on RL with *ex-ante* concave reward maximization. The class of *ex-ante* concave reward maximization problems include the maximization of $g(\mathbb{E}[\bar{V}_{1:T}])$, as well as its *ex-ante variants*, including the long term average variant $g(\mathbb{E}[\lim_{T\to\infty}\bar{V}_{1:T}])$ and its infinite horizon discounted reward variant. The function $g : \mathbb{R}^K \to \mathbb{R}$ is assumed to be concave.

The class of ex-ante concave reward maximization problems is studied by the following research works. Chow et al. (2017) study the case where g is specialized to the Conditional Value-at-Risk objective. Hazan et al. (2019) study the case when g models the entropy function over the probability distribution over the state space, in order to construct a policy which induces a distribution over the state space that is as close to the uniform distribution as possible. Miryoosefi et al. (2019) study the case of minimizing the distance between $\mathbb{E}[\bar{V}_{1:T}]$ and a target set in $\mathbb{R}^K$. Lee et al. (2019) study the objective of state marginal matching, which aims to make the state marginal distribution match a given target state distribution. Pareto optimality of $\mathbb{E}[\bar{V}_{1:T}]$ and its ex-ante variants are studied in (Mannor & Shimkin, 2004; Gábor et al., 1998; Barrett & Narayanan, 2008; Van Moffaert & Nowé, 2014). Lastly, a recent work Zahavy et al. (2021) provides a unifying framework that encompasses many of the previously mentioned works, by studying the problem of maximizing $g(\mathbb{E}[\bar{V}_{1:T}])$ and its ex-ante variants, where g is concave and Lipschitz continuous. Our contributions, which concern the ex-post max-min fairness $\mathbb{E}[\min_{k\in\{1,\ldots,K\}}\bar{V}_{1:T,k}]$ and its generalization to the ex-post concave case, are crucially different from the body of works on the ex-ante case. The difference is further highlighted in the forthcoming Section 3.2.

Additionally, a body of works Altman (1999); Tessler et al. (2019); Le et al. (2019); Liu et al. (2020) study the setting where g is a linear function, subject to the constraint that $\mathbb{E}[\bar{V}_{1:T}]$ (or its ex-ante variants) is contained in a convex feasible region, such as a polytope. There is another line of research works Tarbouriech & Lazaric (2019); Cheung (2019); Brantley et al. (2020) focusing on various online settings. The works Tarbouriech & Lazaric (2019); Cheung (2019) focus on the ex-post setting like ours, but they crucially assume that the underlying g is smooth, which is not the case for our max-min objective nor the case of Lipschitz continuous concave functions. In addition, the optimality gap (quantified by the notion of regret) degrades linearly with the number of states, which makes their applications to large scale problems challenging. Brantley et al. (2020) focus on the ex-ante setting, different from our ex-post setting, and their optimality gap also degrades linearly with the number of states.

3 MODEL

Set up. An instance of the Reinforcement Learning with Max-Min Fairness (RL-MMF) problem is specified by the tuple $(\mathcal{S}, s_1, \mathcal{A}, T, \mathcal{O})$. The set $\mathcal{S}$ is a finite state space, and $s_1 \in \mathcal{S}$ is the initial state. In the collection $\mathcal{A} = \{\mathcal{A}_s\}_{s\in\mathcal{S}}$, the set $\mathcal{A}_s$ contains the actions that the agent can take when he is at state s. Each set $\mathcal{A}_s$ is finite. The quantity $T \in \mathbb{N}$ is the number of time steps.

When the agent takes action $a \in \mathcal{A}_s$ at state s, he receives the array of stochastic outcomes $(s', U(s,a))$, governed by the outcome distribution $\mathcal{O}(s,a)$. For brevity, we abbreviate the relationship as $(s', U(s,a)) \sim \mathcal{O}(s,a)$. The outcome $s' \in \mathcal{S}$ is the subsequent state he transits to. The outcome $U(s,a) = (U_k(s,a))_{k=1}^K$ is a random vector lying in $[-1,1]^K$ almost surely. The random variable $U_k(s,a)$ is the amount of type-k stochastic reward the agent receives. We allow the random variables $s', U_1(s,a), \ldots U_K(s,a)$ to be arbitrarily correlated.

Dynamics. At time $t \in \{1, \ldots T\}$, the agent observes his current state s_t. Then, he selects an action $a_t \in \mathcal{A}_{s_t}$. After that, he receives the stochastic feedback $(s_{t+1}, V_t(s_t, a_t)) \sim \mathcal{O}(s_t, a_t)$. We denote $V_t(s_t,a_t) = (V_{t,k}(s_t,a_t))_{k=1}^K$, where $V_{t,k}(s_t,a_t)$ is the type-k stochastic reward received at time t. The agent select the actions $\{a_t\}_{t=1}^T$ with a policy $\pi = \{\pi_t\}_{t=1}^T$, which is a collection of functions. For each t, the function π_t inputs the history $H_{t-1} = \cup_{q=1}^{t-1}\{s_q, a_q, V_q(s_q,a_q)\}$ and the current state $\{s_t\}$, and outputs $a_t \in \mathcal{A}_{s_t}$. We use the notation a_t^π to highlight that the action is chosen under policy π. A policy π is stationary if for all t, H_{t-1}, s_t it holds that $\pi_t(H_{t-1}, s_t) = \bar{\pi}(s_t)$ for some function $\bar{\pi}$, where $\bar{\pi}(s) \in \mathcal{A}_s$ for all s. With a slight abuse of notation, we identify a stationary policy with the function $\bar{\pi}$.

Objective. We denote $\bar{V}^{\pi}_{1:t} = \frac{1}{t}\sum_{q=1}^{t} V_q(s_q, a^{\pi}_q)$ as the time average vectorial reward during time 1 to t under policy π. The agent's over-arching goal is to design a policy π that maximizes $\mathbb{E}[g_{\min}(\bar{V}^{\pi}_{1:T})]$, where $g_{\min} : \mathbb{R}^K \to \mathbb{R}$ is defined as $g_{\min}(v) = \min_{k\in\{1,\ldots,K\}} v_k$. Denoting $\bar{V}^{\pi}_{1:T,k}$ as the k-th component of the vector $\bar{V}^{\pi}_{1:T}$, the value $g_{\min}(\bar{V}^{\pi}_{1:T}) = \min_k \bar{V}^{\pi}_{1:T,k}$ is the minimum time average reward, among the reward types $1, \ldots, K$. The function $g_{\min}$ is concave, and is 1-Lipschitz w.r.t. $\|\cdot\|_\infty$ over the domain $\mathbb{R}^K$.

When $K = 1$, the RL-MMF problem reduces to the conventional RL problem with scalar reward maximization. The case of $K > 1$ is more subtle. Generally, the optimizing agent needs to focus on different reward types in different time steps, contingent upon the amounts of the different reward types at the current time step. Since the max-min fairness objective could lead to an intractable optimization problem, we aim to design a near-optimal policy for the RL-MMF problem.

3.1 REGRET

We quantify the near-optimality of a policy π by the notion of regret, which is the difference between a benchmark $\text{opt}(\mathsf{P}(g_{\min}))$ and the expected reward $\mathbb{E}[g_{\min}(\bar{V}^{\pi}_{1:T})]$. Formally, the regret of a policy π in a T time step horizon is

$$\text{Reg}(\pi, T) = \text{opt}(\mathsf{P}(g_{\min})) - \mathbb{E}[g_{\min}(\bar{V}^{\pi}_{1:T})]. \tag{1}$$

The benchmark $\text{opt}(\mathsf{P}(g_{\min}))$ is a fluid approximation to the expected optimum. To define $\text{opt}(\mathsf{P}(g_{\min}))$, we introduce the notation $p = \{p(s'|s,a)\}_{s\in\mathcal{S},a\in\mathcal{A}_s}$, where $p(s'|s,a)$ is the probability of transiting to s' from s, a. In addition, we introduce $v = \{v(s,a)\}_{s\in\mathcal{S},a\in\mathcal{A}_s}$, where $v(s,a) = \mathbf{E}[U(s,a)]$ is the vector of the K expected rewards. The benchmark $\text{opt}(\mathsf{P}(g_{\min}))$ is the optimal value of the maximization problem $\mathsf{P}(g_{\min})$. For any $g : \mathbb{R}^K \to \mathbb{R}$, we define

$$\mathsf{P}(g):\ \max_x\ g\left(\sum_{s\in\mathcal{S},a\in\mathcal{A}_s} v(s,a)x(s,a)\right)$$

$$\text{s.t.} \sum_{a\in\mathcal{A}_s} x(s,a) = \sum_{s'\in\mathcal{S},a'\in\mathcal{A}_{s'}} p(s|s',a')x(s',a') \quad \forall s\in\mathcal{S} \tag{2a}$$

$$\sum_{s\in\mathcal{S},a\in\mathcal{A}_s} x(s,a) = 1 \tag{2b}$$

$$x(s,a) \geq 0 \qquad \forall s\in\mathcal{S}, a\in\mathcal{A}_s. \tag{2c}$$

The concave maximization problem $\mathsf{P}(g_{\min})$ serves as a fluid relaxation to RL-MMF. For each $s \in \mathcal{S}, a \in \mathcal{A}_s$, the variable $x(s,a)$ can be interpreted as the frequency of the agent visiting state s and taking action a. The set of constraints (2a) stipulates that the rate of transiting out of a state s is equal to the rate of transiting into the state s for each $s \in \mathcal{S}$, while the sets of constraints (2b , 2c) require that $\{x(s,a)\}_{s\in\mathcal{S},a\in\mathcal{A}_s}$ forms a probability distribution over the state-action pairs. Consequently, $\text{opt}(\mathsf{P}(g_{\min}))$ is an asymptotic (in T) upper bound to the expected optimum.

Our goal is to design a policy π such that its regret[1] $\text{Reg}(T)$ satisfies

$$\text{Reg}(T) = \text{opt}(\mathsf{P}(g_{\min})) - \mathbb{E}[g_{\min}(\bar{V}^{\pi}_{1:T})] \leq \frac{D}{T^\gamma} \tag{3}$$

holds for all initial state $s_1 \in \mathcal{S}$ and all $T \in \mathbb{N}$, with parameters $D, \gamma > 0$ independent of T. We assume the access to an *optimization oracle* Λ, which returns a near-optimal policy given any scalar reward. For $\vartheta \in \mathbb{R}^K$, define the linear function $g_\vartheta : \mathbb{R}^K \to \mathbb{R}$ as $g_\vartheta(w) = \vartheta^\top w = \sum_{k=1}^K \vartheta_k w_k$. The oracle Λ inputs $\vartheta \in \mathbb{R}^K$, and outputs a policy π satisfying

$$\text{opt}(\mathsf{P}(g_\vartheta)) - \mathbb{E}[g_\vartheta(\bar{V}^{\pi}_{1:T})] = \text{opt}(\mathsf{P}(g_\vartheta)) - \mathbb{E}[\vartheta^\top \bar{V}^{\pi}_{1:T}] \leq \frac{D_{\text{lin}}}{T^\beta} \tag{4}$$

for all initial state $s_1 \in \mathcal{S}$ and all $T \in \mathbb{N}$, with parameters $D_{\text{lin}}, \beta > 0$ independent of T. By assuming $\beta > 0$, we are assuming that the output policy π is near-optimal, in the sense that the difference $\text{opt}(\mathsf{P}(g_\vartheta)) - \mathbb{E}[\vartheta^\top \bar{V}^{\pi}_{1:T}]$ converges to 0 as T tends to the infinity. A higher β signifies a

---

[1] We omit the notation with π for brevity sake

faster convergence, representing a higher degree of near-optimality. We refer to ϑ as a scalarization of v, with the resulting scalarized reward being $\vartheta^\top v(s,a)$ for each s, a.

Our algorithmic frameworks involve invoking Λ as a sub-routine on different ϑ's. In other words, we assume an algorithmic sub-routine that solves the underlying RL problem with scalar reward (the case of $K = 1$), and delivers an algorithm that ensures max-min fairness (the case of $K \geq 1$). Finally, while the main text focuses on g_{min}, our algorithm design and analysis can be generalized to the case of concave g, as detailed in Appendix C.

3.2 COMPARISON BETWEEN MAXIMIZING $\mathbb{E}[g_{\text{MIN}}(\bar{V}^{\pi}_{1:T})]$ AND $g_{\text{MIN}}(\mathbb{E}[\bar{V}^{\pi}_{1:T}])$

Before introducing our algorithms, we illustrate the difference between the objectives of maximizing $\mathbb{E}[g_{\text{min}}(\bar{V}^{\pi}_{1:T})]$ and $g_{\text{min}}(\mathbb{E}[\bar{V}^{\pi}_{1:T}])$ by the deterministic instance in Figure 1, with initial state $s_1 = s^o$.

The figure depicts an instance with $K = 2$. An arc represents an action that leads to a transition from its tail to its head. For example, the arc from s^o to s^ℓ represents the action $a^{o\ell}$, with $p(s^\ell \mid s^o, a^{o\ell}) = 1$. Likewise, the loop at s^ℓ represents the action $a^{\ell\ell}$ with $p(s^\ell \mid s^\ell, a^{\ell\ell}) = 1$. Each arc is labeled with its vectorial reward, which is deterministic. For example, with certainty we have $V(s^o, a^{o\ell}) = \binom{0}{0}$ and $V(s^\ell, a^{\ell\ell}) = \binom{0}{1}$.

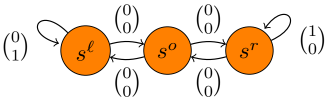

Figure 1: States and actions are represented by circles and arcs.

Consider two stationary policies π^ℓ, π^r, defined as $\pi^\ell(s^r) = a^{ro}, \pi^\ell(s^o) = a^{o\ell}, \pi^\ell(s^\ell) = a^{\ell\ell}$ and $\pi^r(s^r) = a^{rr}, \pi^r(s^o) = a^{or}, \pi^r(s^\ell) = a^{\ell o}$. The policy π^ℓ always seeks to transit to s^ℓ, and then loop at s^ℓ indefinitely, likewise for π^r. With certainty, $\bar{V}^{\pi^\ell}_{1:T} = \binom{0}{1-1/T}$, $\bar{V}^{\pi^r}_{1:T} = \binom{1-1/T}{0}$.

The objective $g_{\text{min}}(\mathbb{E}[\bar{V}^{\pi}_{1:T}])$ is maximized by choosing π_{ran} uniformly at random from the collection $\{\pi^\ell, \pi^r\}$. We have $\mathbb{E}[\bar{V}^{\pi_{\text{ran}}}_{1:T}] = \binom{1/2-1/(2T)}{1/2-1/(2T)}$, leading to the optimal value of $1/2 - 1/(2T)$. More generally, existing research focuses on maximizing $g(\mathbb{E}[\bar{V}^{\pi}_{1:T}])$ for certain concave g, and the related objectives of maximizing $g(\lim_{T\to\infty} \mathbb{E}[\bar{V}^{\pi}_{1:T}])$ or $g(\mathbb{E}[\sum_{t=1}^{\infty} \alpha^t V_t(s_t, a^\pi_t)])$, where $\alpha \in (0,1)$ is the discount factor. In these research works, a near-optimal policy π is constructed by first generating a collection Π of stationary policies, then sampling π uniformly at random from Π.

Interestingly, π_{ran} is sub-optimal for maximizing $\mathbb{E}[g_{\text{min}}(\bar{V}^{\pi}_{1:T})]$. Indeed, $\Pr(\bar{V}^{\pi_{\text{ran}}}_{1:T} = \binom{0}{1-1/T}) = \Pr(\bar{V}^{\pi_{\text{ran}}}_{1:T} = \binom{1-1/T}{0}) = 1/2$, so we have $\mathbb{E}[g_{\text{min}}(\bar{V}^{\pi_{\text{ran}}}_{1:T})] = 0$ for all T. Now, consider the deterministic policy π_{sw}, which first follows π^ℓ for the first $\lfloor T/2 \rfloor$ time steps, then follows π^r in the remaining $\lceil T/2 \rceil$ time steps. We have $\bar{V}^{\pi_{\text{sw}}}_{1:T,k} \geq 1/2 - 2/T$ for each $k \in \{1, 2\}$, meaning that $g_{\text{min}}(\bar{V}^{\pi_{\text{sw}}}_{1:T}) \geq 1/2 - 2/T$. Note that $g_{\text{min}}(\mathbb{E}[\bar{V}^{\pi_{\text{sw}}}_{1:T}]) \geq g_{\text{min}}(\mathbb{E}[\bar{V}^{\pi_{\text{ran}}}_{1:T}]) - 2/T$, so the policy π_{sw} is also near-optimal for maximizing $g_{\text{min}}(\mathbb{E}[\bar{V}^{\pi}_{1:T}])$.

Altogether, an optimal policy for maximizing $g_{\text{min}}(\mathbb{E}[\bar{V}^{\pi}_{1:T}])$ can be far from optimal for maximizing $\mathbb{E}[g_{\text{min}}(\bar{V}^{\pi}_{1:T})]$. In addition, for the latter objective, it is intuitive to imitate π_{sw}, which is to partition the horizon into episodes and run a suitable stationary policy during each episode. A weakness to π_{sw} is that its partitioning requires the knowledge on T. While our algorithm follows the intuition to imitate π_{sw}, we propose an alternate partitioning that allows does not require T as an input.

4 ONLINE-REOPT ALGORITHM FOR RL-MMF

We propose the Online-ReOpt algorithm, displayed in Algorithm 1. The algorithm runs in episodes. An episode $m \in \{1, 2, \ldots\}$ starts at time $\tau(m)$ (defined in Line 2), and ends at time $\tau(m+1) - 1$. Before the start of episode m, the algorithm computes the scalarization $\vartheta_{\tau(m)}$ based on the Multiplicative Weight Update (MWU), which we detail later. Then, the algorithm invokes the optimization oracle Λ, which returns a policy π_m that is near-optimal for the MDP with scalar rewards $r_m = \{r_m(s,a)\}_{s,a}$, where $r_m(s,a) = \vartheta^\top_{\tau(m)} v(s,a)$. Note that we only assume a black-box access to Λ, and the parameters D_{lin}, β do not need to be input to the Algorithm. Finally, the algorithm runs policy π_m during episode m. The Online Re-Opt algorithm is an *anytime* algorithm, since it does not require T as an input. Rather, it requires knowing T only during the terminal time step T. To complete the description of the algorithm, we provide the details about the scalarization.

Algorithm 1 Online-ReOpt for g_{min}

```
 1: Inputs: Optimization oracle $\Lambda$.
 2: Set $\tau(m) = \lfloor m^{3/2} \rfloor$ for $m \in \mathbb{N}$.
 3: for Episode $m = 1, 2, \ldots$ do
 4:     Define $\vartheta_{\tau(m)}$ according to (5).
 5:     Compute policy $\pi_m \leftarrow \Lambda(\vartheta_{\tau(m)})$.
 6:     for Time $t = \tau(m), \ldots, \tau(m+1) - 1$ do
 7:         Choose action $a_t = \pi_m(s_t)$.
 8:         Observe the outcomes $V_t(s_t, a_t)$ and the next state $s_{t+1}$.
 9:         if $t = T$ then
10:             Break the for loops and terminate the algorithm.
11:         end if
12:     end for
13: end for
```

Scalarization by MWU. At a time step t, we define the scalarization $\vartheta_t = \{\vartheta_{t,k}\}_{k=1}^{K}$ as

$$\vartheta_{t,k} = \frac{\exp\left[-\eta_{t-1} \sum_{j=1}^{t-1} V_{j,k}(s_j, a_j)\right]}{\sum_{\kappa=1}^{K} \exp\left[-\eta_{t-1} \sum_{j=1}^{t-1} V_{j,\kappa}(s_j, a_j)\right]}, \tag{5}$$

where

$$\eta_{t-1} = \frac{\sqrt{\log K}}{\max\{(t-1)^{2/3}, 1\}}. \tag{6}$$

In particular, at the start of each episode m, we apply (5) with $t = \tau(m)$ in Line 4. For the case $m = 1$, we have $\vartheta_{\tau(1),k} = 1/K$ for all $k \in \{1, \ldots, K\}$, meaning that all reward types are assigned with the same weight at the beginning. The exponent $2/3$ in the learning rate $\eta_{\tau(m)-1}$ in (5) is different from the conventional choice of $1/2$ (Arora et al., 2012). Our exponent is chosen for optimizing the resulting regret bound from our forthcoming analysis. We follow the approach in Chapter 7 in (Orabona, 2019) to define a time-varying learning rate.

The scalarization ϑ_t by (5) promotes max-min fairness. Consider two reward types k, k' with $\sum_{q=1}^{t-1} V_{q,k}(s_q, a_q) > \sum_{q=1}^{t-1} V_{q,k'}(s_q, a_q)$. We have $\vartheta_{t,k'} > \vartheta_{t,k}$, meaning that a higher weight is assigned to reward type k' than type k. This implies that there is a higher emphasis on increasing the type-k' reward, which is in shortage as compared to type k, than the type-k reward. Hence, max-min fairness is promoted.

4.1 THEORETICAL GUARANTEES

We provide the following theoretical guarantee for Online-ReOpt.

Theorem 1 *Consider the RL-MMF problem. Online-ReOpt, displayed in Algorithm 1, satisfies*

$$\textit{Reg}(T) \leq \frac{114\sqrt{\log K}}{T^{1/3}} + \frac{144 D_{\textit{lin}}}{T^{\beta/3}}, \tag{7}$$

where $D_{\textit{lin}}, \beta$ are parameters related to the optimization oracle Λ.

Theorem 1 is a generalization result, in the sense that it generalizes the ability of achieving near-optimality for the case of $K = 1$ to the case of $K \geq 1$. Indeed, as long as $\beta > 0$, meaning that the regret of the optimization oracle Λ diminishes with a growing T on any RL with scalar reward problem, the regret bound (7) in Theorem 1 also tends to zero as T increases.

Theorem 1 is proved in Appendix section C.3. The first regret term in (7) arises from two sources: (a) the regret of the MWU algorithm, (b) the update delay on the scalarization due to the episodic structure. To elaborate on (b), consider a time step t in episode m. Recall that the scalarization ϑ_t by (5) promotes max-min fairness, and ideally we should have employed the policy returned by $\Lambda(\vartheta_t)$ at time t. In contrast, in Online-ReOpt the action a_t is determined by π_m, the output of $\Lambda(\vartheta_{\tau(m)})$.

Item (b) accounts for the regret due to using $\vartheta_{\tau(m)}$ rather then ϑ_t. We crucially use the fact that ϑ_t are slowly changing in t so that the resulting regret is still diminishing with t.

The second term in (7) is due to the regret of the optimization oracle Λ. The exponent $\beta/3$ in the term is less than the exponent β in (4), as each policy π_m is run for only $\tau(m+1)-\tau(m) = O(\sqrt{\tau(m)})$ many time steps. Our design of $\{\tau(m)\}_{m=1}^M$ allows a shorter time frame for π_m to converge to its expected reward, as compared to running a policy for T steps in (4). When we increase the episode length $\tau(m+1)-\tau(m)$, the regret due to (b) increases, while the regret due to the second term in (7) decreases, and vice versa. Our design of $\{\tau(m)\}_{m=1}^M$ strikes an optimal (in terms of our analysis) balance between these two sources of regret.

The regret bound (7) does not feature a direct dependence on the sizes of the state and action spaces. The dependence on the hardness of the underlying MDP is only reflected through the parameters D_{lin}, β. Therefore, apart from the deterioration of the exponent β to $\beta/3$ and the first term in (7), our algorithm does not introduce any overhead in the generalization from the case of $K = 1$ to the case of $K \geq 1$. Improving the exponent $\beta/3$ in (7) is an interesting open question. Finally, Theorem 1 is generalized to the case of maximizing a concave utility objective $\mathbb{E}[g(\bar{V}^{\pi}_{1:T})]$, where g is Lipschitz continuous and concave. We detail the generalization in the model, algorithm and theoretical results in Appendix C.1.

4.2 OFFLINE VARIANTS TO ONLINE-REOPT

While the Online-ReOpt Algorithm achieves near-optimality, the efficiency of its implementation could be hindered by the need of online computation in Line 5 in Algorithm 1. Indeed, in order to compute π_m, the agent has to input the optimization oracle Λ and the scalarization $\vartheta_{\tau(m)}$, which is only known at the end of time step $\tau(m)-1$. In the case when the optimization oracle involves heavy computation, for example training deep neural networks, such online computation may not be realistic.

In this section, we propose Offline-ReOpt, which is a variant of Online-ReOpt that does not require invoking Λ during the horizon. The Offline-ReOpt is obtained from the Online-ReOpt by modifying two lines in Algorithm 1, as enumerated below. The full algorithm of Offline-ReOpt is provided in Appendix section A.1.

1. Replace the input of Λ in Line 1 with the input of the policy family $\Pi = \{(\vartheta, \pi^{(\vartheta)})\}_{\vartheta\in\Omega}$. The index set Ω is a finite subset of $\{\vartheta \in \mathbb{R}^K : \|\vartheta\|_1 = 1, \vartheta \geq 0\}$, the collection of all possible scalarizations. For each $\vartheta \in \Omega$, the policy $\pi^{(\vartheta)}$ is the output of $\Lambda(\vartheta)$.
2. Replace the online computation in Line 5 with these two lines:
   (a) Identify $\tilde{\vartheta}_{\tau(m)} \in \Omega$ that achieves $\min_{\vartheta\in\Omega} \left\|\vartheta - \vartheta_{\tau(m)}\right\|_1$.
   (b) Select policy $\pi_m = \pi^{(\tilde{\vartheta}_{\tau(m)})}$.

In item (1), all the policies in Π are computed *before* the execution of the algorithm, unlike the case in Online-ReOpt. Consequently, in item (2), the selection of policy π_m does not require invoking the optimization oracle Λ.

The main idea behind item (2) is that, in the case when the desired scalarization $\vartheta_{\tau(m)}$ does not lie in Ω, we chooses the surrogate scalarization $\tilde{\vartheta}_{\tau(m)}$ that is closest to $\vartheta_{\tau(m)}$, so that the resulting policy $\pi^{(\tilde{\vartheta}_{\tau(m)})}$ will be a reasonable approximation to the desired policy $\pi^{(\vartheta_{\tau(m)})}$.

In order for the surrogate $\tilde{\vartheta}_{\tau(m)}$ to be close to $\vartheta_{\tau(m)}$, it is desirable for the finite index set Ω to be so diverse that every scalarization $\vartheta_{\tau(m)}$ would be in a close neighborhood of a scalarization in Ω. We propose two families of Ω for the desired diversification. The first is the **random point family**, detailed in Appendix section A.2. The family is constructed by sampling random points in the domain $\{\vartheta \in \mathbb{R}^K : \|\vartheta\|_1 = 1, \theta \geq 0\}$ of all possible scalarizations. The second is the **imitation based family**, also detailed in Appendix section A.2. The family is constructed by first running Online-ReOpt multiple times, then collecting the scalarizations and the corresponding policies generated.

5 EXPERIMENTS

We evaluate our proposed algorithms and benchmark algorithms in a controlled queueing system involving vectorial rewards. For each of the algorithms, we first run the algorithm for $Z_{\text{po}} = Z_{\text{an}} \times \Xi$ independent trials, resulting in the Z_{po} average vectorial rewards[2] $\{\bar{V}_{1:T}^{(z_{\text{an}},\xi)}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}, 1\leq \xi \leq \Xi}$. We plot the following three quantities against T:

- **Ex-post Fairness:** $\bar{\Psi} = \frac{1}{Z_{\text{an}}}\frac{1}{\Xi}\sum_{z_{\text{an}}=1}^{Z_{\text{an}}}\sum_{\xi=1}^{\Xi} g_{\text{min}}\left(\bar{V}_{1:T}^{(z_{\text{an}},\xi)}\right)$, an estimate to $\mathbb{E}[g_{\text{min}}(\bar{V}_{1:T})]$.
- **Ex-ante Fairness:** $\bar{\Gamma} = \frac{1}{Z_{\text{an}}}\sum_{z_{\text{an}}=1}^{Z_{\text{an}}} g_{\text{min}}\left(\frac{1}{\Xi}\sum_{\xi=1}^{\Xi}\bar{V}_{1:T}^{(z_{\text{an}},\xi)}\right)$, an estimate to $g_{\text{min}}(\mathbb{E}[\bar{V}_{1:T}])$.
- **Type k rewards for each $1 \leq k \leq K$:** $\bar{\Phi}_k = \frac{1}{Z_{\text{an}}}\frac{1}{\Xi}\sum_{z_{\text{an}}=1}^{Z_{\text{an}}}\sum_{\xi=1}^{\Xi}\bar{V}_{1:T,k}^{(z_{\text{an}},\xi)}$, an estimate to $\mathbb{E}[\bar{V}_{1:T,k}]$.

We define the upper and lower error bars respectively as the 75 and 25-percentiles of the data, see Appendix section B.1 for details. For the forthcoming discussions, we denote $\mathbf{e}_k$ as the k-th standard basis vector for $k \in \{1, \ldots K\}$ in $\mathbb{R}^K$. In addition, we denote $\mathbf{1}_K$, $\mathbf{0}_K$ as the all one vector and the all zero vector in $\mathbb{R}^K$.

5.1 QUEUING NETWORK

Queuing problems are studied extensively due to their relevance in fields such as manufacturing and in communication systems. In our evaluation, we focus on a discrete-time queuing system. The queuing network that we have tested our algorithms on, consisting of two servers and four queues arranged in a bidirectional fashion, has been previously studied in works by Rybko & Stolyar (1992), Kumar & Seidman (1990), Chen & Meyn (1998), de Farias & Van Roy (2003) and Banijamali et al. (2019). This network is shown in Figure 2.

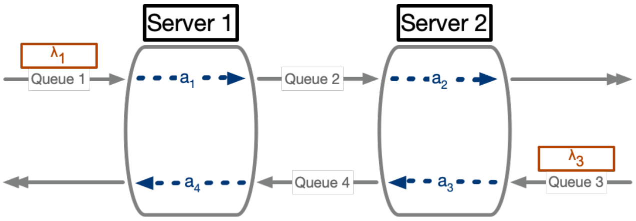

Figure 2: A bi-directional four-queue network

There are two servers in the system. Server 1 only serves Queue 1 *or* Queue 4 with service rates $\mu_1 = 0.3$ and $\mu_4 = 0.3$ respectively, and where Server 2 similarly only serves Queue 2 *or* Queue 3 at the rates of $\mu_2 = 0.3$ and $\mu_3 = 0.3$. Arrivals occur at a rate of $\lambda_1 = 0.2$ and $\lambda_2 = 0.2$ at Queues 1 and 3 respectively. An arrival that gets served at Queue 1 by Server 1 then progresses to Queue 2, and only leaves the system after it has been served by Server 2. Likewise, arrivals at Queue 3 have to be served by Server 2, before moving on to Queue 4 to be served by Server 1 in order to leave the system. Each queue i has a maximum length of $L_i = 9$, and a customer is rejected at a queue if the queue is at its full capacity. Conversely, an empty queue remains at length 0 even if an action is taken to serve that queue.

The state of the system is thus defined by the vector $\boldsymbol{x}_t = (x_{t,1}, x_{t,2}, x_{t,3}, x_{t,4})$ whereby $x_{t,i}$ represents the length of the queue i at time t. At each time step t, a decision has to be made by each server to serve *only one* or *neither* of its queues, which we can represent by a 4-component vector $\boldsymbol{a}_t = (a_{t,1}, a_{t,2}, a_{t,3}, a_{t,4}) \in \{0, 1\}^4$, where $a_{t,i} = 1$ indicates the decision to serve Queue i at time t, and $a_{t,i} = 0$ otherwise. Note that the condition of being able to only serve *one* queue at each server naturally imposes the constraints $a_{t,1} + a_{t,4} \leq 1$ and $a_{t,2} + a_{t,3} \leq 1$ at each t, meaning that $\mathcal{A}_s = \{a \in \{0, 1\}^4 : a_1 + a_4 \leq 1, a_2 + a_3 \leq 1\}$ for each s.

---

[2] To avoid clutter, we omit the upper-script for the algorithm.

The transition dynamics for the system can then defined by the following equation when $0 < x_{t,i} < L_i$, where $\boldsymbol{e}_i$ refers to the basis vector in $\mathbb{R}^4$:

$$\boldsymbol{x}_{t+1} = \begin{cases} \boldsymbol{x}_t + \boldsymbol{e}_1 & \text{with probability } \lambda_1 \\ \boldsymbol{x}_t + \boldsymbol{e}_3 & \text{with probability } \lambda_2 \\ \boldsymbol{x}_t + \boldsymbol{e}_2 - \boldsymbol{e}_1 & \text{with probability } \mu_1 a_1 \\ \boldsymbol{x}_t - \boldsymbol{e}_2 & \text{with probability } \mu_2 a_2 \\ \boldsymbol{x}_t + \boldsymbol{e}_4 - \boldsymbol{e}_3 & \text{with probability } \mu_3 a_3 \\ \boldsymbol{x}_t - \boldsymbol{e}_4 & \text{with probability } \mu_4 a_4 \\ \boldsymbol{x}_t & \text{otherwise} \end{cases} \tag{8}$$

We define the type-i reward at time t as $V_{t,i}(\boldsymbol{x}, \boldsymbol{a}) = 1 - \frac{x_{t,i}}{L_i}$, for $i \in \{1, \dots, 4\}$. Recall that $x_{t,i}$ is the queue length of Queue i at time t. The reward $V_{t,i}(\boldsymbol{x}, \boldsymbol{a})$ is equal to 1 if Queue i is empty, and the reward $V_{t,i}(\boldsymbol{x}, \boldsymbol{a})$ decreases linearly with the length of Queue i at time t. In particular, $V_{t,i}(\boldsymbol{x}, \boldsymbol{a}) = 0$ if Queue i is full. Altogether, the agent's reward for Queue i at time t positively correlates with the degree of idleness of the Queue. The maximization of $g_{\text{min}}(\bar{V}_{1:T}) = \min_{1 \le i \le 4} \bar{V}_{1:T,i}$ is equivalent to the minimization of time-average queue lengths among all queues, hence enforcing all queues to be stable simultaneously.

5.1.1 SIMULATION RESULTS

In our simulation, we evaluate 5 algorithms. Three of them are our proposed algorithms, namely Online-ReOpt, Offline-ReOpt with Random Point Family and Offline-ReOpt with Imitation based family. The other two are existing baselines. The Meta Algorihtm by Zahavy et al. (2021) is the state-of-the-art for maximizing $g_{\text{min}}(\mathbb{E}[\bar{V}_{1:T}])$, while the longer queue first heuristic is a well-established algorithm in the queuing theory literature. As the name suggests, each server serves the longer of the two queues at each time round. We ran each of the algorithms with the following parameters: $T = 100000$, $Z_{\text{an}} = 10$ and $\Xi = 100$, meaning $Z_{\text{po}} = Z_{\text{an}} \times \Xi = 1000$.[3] All of the algorithms employ the same optimization oracle Λ, with the same hyper-parameters and architecture, a double deep Q-learning network algorithm (Double DQN) by Hasselt et al. (2016).

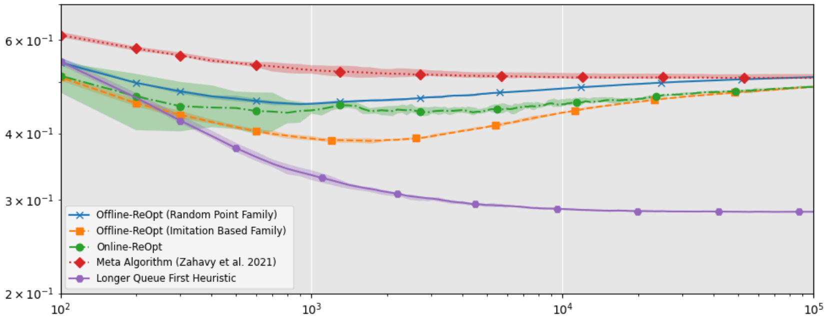

Figure 3: Ex-ante Fairness of various algorithms in a queuing network

Figure 3 plots the quantity $\bar{\Psi}$ against T under the 5 algorithms. Notice in Figure 3 how the Offline and Online-ReOpt algorithms, as well as the Meta Algorithm by Zahavy et al. (2021) perform similarly well in terms of *ex-ante* fairness. Among them, the Meta Algorithm has the best performance, since the Re-Optimization schedule in our proposed algorithms compromises the *ex-ante* fairness objective. All algorithms demonstrate converging behavior, in the sense that the error bars diminishes as T grows.

---

[3] Except for Online-ReOpt, where we set $\Xi = 5$ since running an online algorithm for 1000 trials is not as practical as running an offline algorithm, which only needs to be trained once.

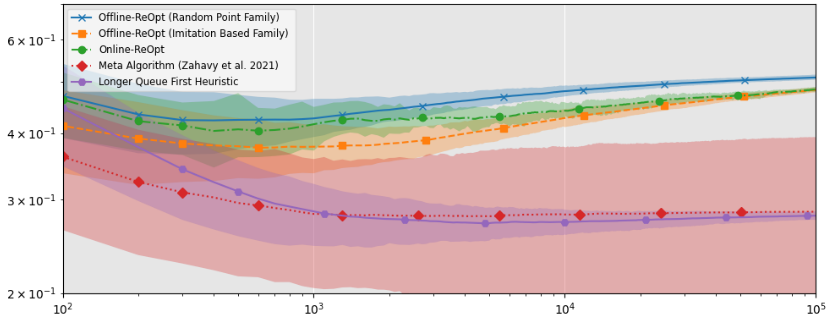

Figure 4: Ex-post Fairness of various algorithms in a queuing network

Figure 4 plots the quantity $\bar{\Gamma}$ against T under the 5 algorithms. In terms of *ex-post* fairness, the Offline and Online-ReOpt algorithms perform significantly better than Meta Algorithm and the Longer Queue First Heuristic. The sub-optimality of the Meta Algorithm corroborates with Section 3.2 that policies designed for the *ex-ante* fairness objective could be far from optimal for the *ex-post* fairness objective. While the Meta Algorithm has a similar performance to the Longer Queue First heuristic, the former has a significantly wider error bar than the latter, meaning that the latter is more stable.

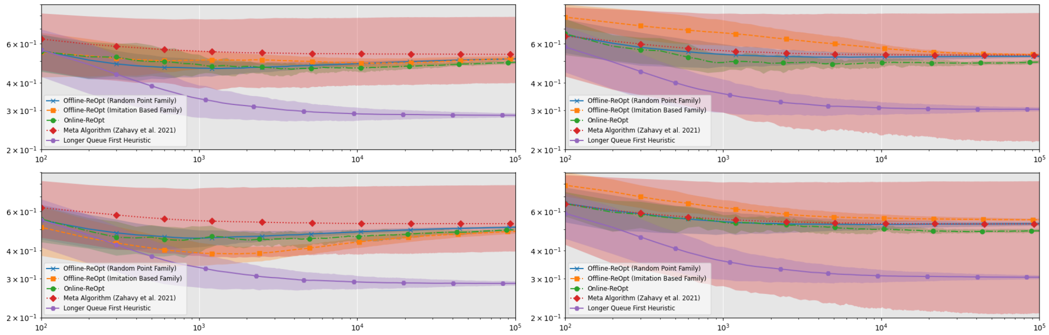

Figure 5: Type k-rewards for $1 \leq k \leq K$ of various algorithms in a queuing network. Figure is read from left to right, top to bottom.

Figure 5 plots Φ_i against T for $i \in \{1, \ldots, 4\}$. In a nutshell, the plotted lines explain the trends in Figure 3, while the error bars shed light on the trends in Figure 4. Firstly, the plotted lines indicate that the Meta Algorithm has the highest (or close to the highest) individual average reward Φ_i for each queue, signifying that the Meta Algorithm has the highest $\mathbb{E}[\bar{V}_{1:T,i}]$ for each $i \in \{1, \ldots, 4\}$. This explains the superiority of the Meta Algorithm shown in Figure 3.

When we focus on the error bars, the plots in Figure 5 tell a different story. Notably, the error bars for the Meta Algorithm is significantly wider than others, meaning that the Z_{po} trials of the Meta Algorithm have significantly different results from one another.[4] When we unpack the summands in $\Phi_1, \ldots, \Phi_4$ and compute the minimum reward in each trial, it results in Figure 4, which is vastly different from Figure 3, signifying the ex-ante and ex-post objectives are fundamentally different.

As a final remark, our numerical experiments do not imply that the Longer Queue Heuristic is a worse algorithm than the other 4 algorithms. Indeed, the Longer Queue Heuristic does not require the knowledge of $\lambda_1, \lambda_2, \mu_1, \ldots, \mu_4$, whereas the other 4 algorithms crucially uses these parameters for generating their policies. In addition, the Longer Queue Heuristic is computationally much less onerous than the others. Finally, the Longer Queue Heuristic demonstrates converging behaviors in all the plots, in the sense that the error bars diminish when T increases.

---

[4] It is helpful to revisit Section 3.2, where Z_{po} trials would result in $\approx Z_{\text{po}}/2$ outcomes of $\binom{0}{1-1/T}$ and $\approx Z_{\text{po}}/2$ outcomes of $\binom{1-1/T}{0}$.

REFERENCES

E. Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

Ershad Banijamali, Yasin Abbasi-Yadkori, Mohammad Ghavamzadeh, and Nikos Vlassis. Optimizing over a restricted policy class in mdps. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89, pp. 3042–3050. PMLR, 16–18 Apr 2019.

Leon Barrett and Srini Narayanan. Learning all optimal policies with multiple criteria. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pp. 41–47, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390162. URL `https://doi.org/10.1145/1390156.1390162`.

Kianté Brantley, Miroslav Dudík, Thodoris Lykouris, Sobhan Miryoosefi, Max Simchowitz, Aleksandrs Slivkins, and Wen Sun. Constrained episodic reinforcement learning in concave-convex and knapsack settings. In *Advances in Neural Information Processing Systems 33*, 2020.

R.-R. Chen and S. Meyn. Value iteration and optimization of multiclass queueing networks. In *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, volume 1, pp. 50–55 vol.1, 1998. doi: 10.1109/CDC.1998.760588.

Wang Chi Cheung. Regret minimization for reinforcement learning with vectorial feedback and complex objectives. In *Advances in Neural Information Processing Systems 32*, pp. 724–734, 2019.

Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *J. Mach. Learn. Res.*, 18(1):6070–6120, January 2017. ISSN 1532-4435.

D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

Zoltán Gábor, Zsolt Kalmár, and Csaba Szepesvári. Multi-criteria reinforcement learning. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pp. 197–205, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1558605568.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pp. 2094–2100. AAAI Press, 2016.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2681–2691. PMLR, 09–15 Jun 2019.

P.R. Kumar and T.I. Seidman. Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *IEEE Transactions on Automatic Control*, 35(3): 289–298, 1990. doi: 10.1109/9.50339.

Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 3703–3712. PMLR, 09–15 Jun 2019.

Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric P. Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *CoRR*, abs/1906.05274, 2019. URL `http://arxiv.org/abs/1906.05274`.

Yongshuai Liu, Jiaxin Ding, and Xin Liu. Ipo: Interior-point policy optimization under constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4940–4947, Apr. 2020. doi: 10.1609/aaai.v34i04.5932. URL `https://ojs.aaai.org/index.php/AAAI/article/view/5932`.

Shie Mannor and Nahum Shimkin. A geometric approach to multi-criterion reinforcement learning. *J. Mach. Learn. Res.*, 5:325–360, 2004.

Sobhan Miryoosefi, Kianté Brantley, Hal Daume III, Miro Dudik, and Robert E Schapire. Reinforcement learning with convex constraints. In *Advances in Neural Information Processing Systems 32*, pp. 14093–14102. 2019.

Francesco Orabona. A modern introduction to online learning. *CoRR*, abs/1912.13213, 2019.

A. N. Rybko and Aleksandr Stolyar. On the ergodicity of stochastic processes describing functioning of open queueing networks. *Problemy Peredachi Informatsii*, (3):3–26, July 1992. ISSN 0555-2923.

Shai Shalev-Shwartz. Online learning: Theory, algorithms, and applications, Jul 2007.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Found. Trends Mach. Learn.*, 4(2):107–194, 2012.

Jean Tarbouriech and Alessandro Lazaric. Active exploration in markov decision processes. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019*, volume 89, pp. 974–982. PMLR, 2019.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2019.

Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *J. Mach. Learn. Res.*, 15(1):3483–3512, January 2014. ISSN 1532-4435.

Tom Zahavy, Brendan O'Donoghue, Guillaume Desjardins, and Satinder Singh. Reward is enough for convex mdps. *ArXiv*, abs/2106.00661, 2021.

A SUPPLEMENTARY DETAILS

A.1 A FULL DESCRIPTION OF OFFLINE-REOPT FOR g_{MIN}

The full description is provided in Algorithm 2, with the modifications colored in blue.

Algorithm 2 Offline-ReOpt for $g_{\min}$

```
Inputs: policy family $\Pi = \{(\vartheta, \pi^{(\vartheta)})\}_{\vartheta \in \Omega}$, where $\Omega \subseteq \{\vartheta \in \mathbb{R}^K : \|\vartheta\|_1 = 1, \vartheta \geq 0\}$.
Set $\tau(m) = \lfloor m^{3/2} \rfloor$ for $m \in \mathbb{N}$.
for Episode $m = 1, 2, \dots, M$ do
    Define $\vartheta_{\tau(m)}$ according to (5).
    Identify $\tilde{\vartheta}_{\tau(m)} \in \Omega$ that achieves $\min_{\vartheta \in \Omega} \left\| \vartheta - \vartheta_{\tau(m)} \right\|_1$ .
    Select policy $\pi_m = \pi^{(\tilde{\vartheta}_{\tau(m)})}$.
    for Time $t = \tau(m), \dots, \tau(m+1) - 1$ do
        Choose action $a_t = \pi_m(s_t)$.
        Observe the outcomes $V_t(s_t, a_t)$ and the next state $s_{t+1}$.
        if $t = T$ then
            Break the For loops and terminate the algorithm.
        end if
    end for
end for
```

A.2 CONSTRUCTIONS OF THE RANDOM POINT AND IMITATION BASED FAMILIES

The random point family and the imitation based family are constructed according to Algorithms 3 and 4 respectively.

Algorithm 3 Construction of an Random Point Family

```
Inputs: Trial numbers $N$, rejection threshold $\epsilon > 0$, optimization oracle $\Lambda$.
Initialize $\Pi \leftarrow \emptyset$.
while $|\Pi| < N$ do
    Sample $\vartheta$ from $\{\vartheta \in [0, 1]^K : \|\vartheta\|_1 = 1\}$ uniformly at random.
    if $\min_{\vartheta' \in \Omega} \|\vartheta' - \vartheta\|_1 > \epsilon$ then        ▷ When $\Pi = \emptyset$, define $\min_{\vartheta' \in \Omega} \|\vartheta' - \vartheta\|_1 = \infty$
        Apply the optimization oracle $\Lambda(\vartheta) \rightarrow \pi^{(\vartheta)}$
        Update $\Pi \leftarrow \Pi \cup \{(\vartheta, \pi^{(\vartheta)})\}$.
    end if
end while
Return $\Pi$.
```

The motivation behind Algorithm 3 is to generate a collection of scalarizations that covers $\{\vartheta \in [0, 1]^K : \|\vartheta\|_1 = 1\}$ through random sampling. We impose a rejection procedure to remove redundant polices.

Algorithm 4 Construction of an Imitation-Based Family

```
Inputs: Trial numbers $N$, and the inputs for the Online-ReOpt Algorithm.
Initialize $\Pi \leftarrow \emptyset$.
for Trial $n = 1, 2, \dots, N$ do
    Run Algorithm 1, which generates scalarizations $\{\vartheta^{(n)}_{\tau(m)}\}_{m=1}^{M}$ as well as policies $\pi^{(n)}_m$,
    where $\pi^{(n)}_m = \Lambda(\vartheta^{(n)}_{\tau(m)})$.        ▷ $M$ is the number of episode in the run.
    Update $\Pi \leftarrow \Pi \cup \left( \cup_{m=1}^{M} \left\{ \left( \vartheta^{(n)}_{\tau(m)}, \pi^{(n)}_m \right) \right\} \right)$.
end for
Return $\Pi$.
```

The motivation behind Algorithm 4 is simple - by saving each scalarization and stationary policy pair generated at specific episodic intervals determined during Online-ReOpt, we would expect the policy family to contain scalarizations that would be reasonable proxies to the true scalarizations that would be encountered during the deployment of the Offline-ReOpt algorithm, should we look up the policy family for a proxy scalarization at the very same episodic intervals found during the running of Onine-ReOpt, and then deploy the corresponding stationary policy associated with this proxy.

In a way, we are simulating the trajectory to be taken in the environment by running the Online-ReOpt algorithm multiple times, and then meticulously following these ”well-trodden” trajectories again in run-time. A similar analogy to this would be how people navigate in cities by memorizing landmarks/waypoints and the corresponding series of directions to take upon reaching them, so as to be able to retrace a path previously used in the future.

B ADDITIONAL DETAILS ABOUT NUMERICAL EXPERIMENTS

B.1 DETAILS ABOUT PLOTS AND ERROR BARS

Recall that in each of these environments and for each of the algorithms, we first run the algorithm for $Z_{\text{po}} = Z_{\text{an}} \times \Xi$ independent trials, resulting in the Z_{po} average vectorial rewards[5] $\{\bar{V}_{1:T}^{(z_{\text{an}},\xi)}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}, 1\leq\xi\leq\Xi}$. For our plots, we use three sets of data $\Psi, \Gamma, \{\Phi\}_{k\in\{1,\ldots,K\}}$, with the quantities to be evaluated in bold.

- **Ex-post Fairness:** $\Psi = \{\Psi_{(z_{\text{an}},\xi)}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}, 1\leq\xi\leq\Xi}$, where

$$\Psi_{(z_{\text{an}},\xi)} = g_{\min}\left(\bar{V}_{1:T}^{(z_{\text{an}},\xi)}\right) \qquad \text{for each } (z_{\text{an}},\xi) \in \{1,\ldots,Z_{\text{an}}\} \times \{1,\ldots,\Xi\}.$$

- **Ex-ante Fairness:** $\Gamma = \{\Gamma_{z_{\text{an}}}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}}$, where

$$\Gamma_{z_{\text{an}}} = g_{\min}\left(\frac{1}{\Xi}\sum_{\xi=1}^{\Xi}\bar{V}_{1:T}^{(z_{\text{an}},\xi)}\right) \qquad \text{for each } z_{\text{an}} \in \{1,\ldots,Z_{\text{an}}\}.$$

- **Type k-reward:** $\Phi_k = \{\Phi_{(z_{\text{an}},\xi),k}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}, 1\leq\xi\leq\Xi}$, where

$$\Phi_{(z_{\text{an}},\xi),k} = \bar{V}_{1:T,k}^{(z_{\text{an}},\xi)} \qquad \text{for each } (z_{\text{an}},\xi) \in \{1,\ldots,Z_{\text{an}}\} \times \{1,\ldots,\Xi\}.$$

Also, recall that we plot these three quantities against T:

- **Ex-post Fairness:** $\bar{\Psi} = \frac{1}{Z_{\text{an}}}\frac{1}{\Xi}\sum_{z_{\text{an}}=1}^{Z_{\text{an}}}\sum_{\xi=1}^{\Xi}\Psi_{(z_{\text{an}},\xi)}$, an estimate to $\mathbb{E}[g_{\min}(\bar{V}_{1:T})]$.
- **Ex-ante Fairness:** $\bar{\Gamma} = \frac{1}{Z_{\text{an}}}\sum_{z_{\text{an}}=1}^{Z_{\text{an}}}\Gamma_{z_{\text{an}}}$, an estimate to $g_{\min}(\mathbb{E}[\bar{V}_{1:T}])$.
- **Type k-reward:** $\bar{\Phi}_k = \frac{1}{Z_{\text{an}}}\frac{1}{\Xi}\sum_{z_{\text{an}}=1}^{Z_{\text{an}}}\sum_{\xi=1}^{\Xi}\Phi_{(z_{\text{an}},\xi),k}$, an estimate to $\mathbb{E}[\bar{V}_{1:T,k}]$.

In each plot, the upper and lower error bars are defined as the 75-percentile and the 25-percentile of the data points. More precisely:

- **Ex-post Fairness:** Order the quantities $\{\Psi_{(z_{\text{an}},\xi)}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}, 1\leq\xi\leq\Xi}$ in an increasing order to get $\Psi_{(1)} \leq \Psi_{(2)} \leq \ldots \leq \Psi_{(Z_{po})}$. The 25-percentile and 75-percentile are respectively $\Psi_{(\lfloor 0.25\times Z_{\text{po}}\rfloor)}, \Psi_{(\lfloor 0.75\times Z_{\text{po}}\rfloor)}$.
- **Ex-ante Fairness:** Order the quantities $\Gamma_1,\ldots,\Gamma_{Z_{\text{an}}}$ in an increasing order to get $\Gamma_{(1)} \leq \Gamma_{(2)} \leq \ldots \leq \Gamma_{(Z_{\text{an}})}$. The 25-percentile and 75-percentile are respectively $\Gamma_{(\lfloor 0.25\times Z_{\text{an}}\rfloor)}, \Gamma_{(\lfloor 0.75\times Z_{\text{an}}\rfloor)}$.
- **Individual Reward for type k:** Order the quantities $\{\Phi_{(z_{\text{an}},\xi),k}\}_{1\leq z_{\text{an}}\leq Z_{\text{an}}, 1\leq\xi\leq\Xi}$ in an increasing order to get $\Phi_{(1),k} \leq \Phi_{(2),k} \leq \ldots \leq \Phi_{(Z_{po}),k}$. The 25-percentile and 75-percentile are respectively $\Phi_{(\lfloor 0.25\times Z_{\text{po}}\rfloor),k}, \Phi_{(\lfloor 0.75\times Z_{\text{po}}\rfloor),k}$.

---

[5] To avoid clutter, we omit the upper-script for the algorithm.

B.2 DETAILS ABOUT THE ALGORITHMS

In the experiment performed, we ran a total of 4 different algorithms, the Online-ReOpt algorithm, the Offline-ReOpt algorithm with a random point family, the Offline-ReOpt algorithm with an imitation based family, the Meta Algorithm by Zahavy et al. (2021), and lastly a Longer Queue First heuristic, for a total of $T = 100000$ time steps per trajectory.

B.2.1 ONLINE-REOPT

This algorithm was run exactly as described in 1, with a double Deep Q Network (Double DQN) as its optimization oracle Λ.

B.2.2 OFFLINE-REOPT (RANDOM POINT FAMILY)

The policy family Π used for this algorithm was generated using Algorithm 3, whereby $N = 2000$, and $\epsilon = 0.0001$.

The algorithm was then run exactly as described in Algorithm 2.

B.2.3 OFFLINE-REOPT (IMITATION BASED FAMILY)

The policy family Π used for this algorithm was generated using Algorithm 4, whereby $N = 5$.

The algorithm was then run exactly as described in Algorithm 2.

B.2.4 META ALGORITHM BY ZAHAVY ET AL. (2021)

This algorithm was run exactly as described in the paper by Zahavy et al. (2021), with the policy player set as the same optimization oracle λ used across all the algorithms that we have run, and the cost player set as the MWU scalarization shown in 5.

The algorithm is run for a total of $K = 2000$ iterations, generating a policy bag with the same number of policies. At the start of each trajectory, one policy is random sampled from this policy bag and then used for the entirety of the trajectory (for T time steps).

B.2.5 LONGER QUEUE FIRST HEURISTIC

This algorithm (or more simply a stationary policy) is based on a simple heuristic that makes the agent select, at any time step, the longer queue at each queue to be served. For example, at a certain time step if Server 1 has two queues of length 5 and 9 respectively, the agent would select the action to serve the queue of length 9. Should both queues be of the same queue length, a random action will be taken to serve either of these queues.

Note that there is no ’learning’ process involved in this heuristic.

B.3 DETAILS ABOUT THE OPTIMIZATION ORACLE Λ

All of the algorithms described above use the same optimization oracle Λ. For our experiments with the queuing network, we have employed a double deep Q-learning network (Double DQN) algorithm by Hasselt et al. (2016).

The architecture used is a simple fully-connected neural net consisting of 2 hidden layers, with 16 and 32 units respectively, and ReLU activation functions. The neural net learning rate is set at 0.00475. Updates to the primary net occurs every 5 steps with batch sizes of 128 from a replay buffer of size 10000 and synchronization across both nets occur every 100 steps.

A decaying epsilon-greedy method of exploration is used, with $\epsilon_{\text{start}} = 0.9$ and $\epsilon_{\text{end}} = 0.2$ decaying across 200 intervals. 50 trajectories of 300 steps each are run for each call to the optimization oracle.

C GENERALIZATION TO CONCAVE g

In this Appendix section, we consider the Reinforcement Learning with Concave Rewards (RL-CR) problem, which is a generalization of the RL-MMF problem. We first explain the model and goal of RL-CR. Then, we provide the Online-ReOpt algorithm for RL-CR, as displayed in the forthcoming Algorithm 5. After that, we state and prove a regret bound of the Online-ReOpt algorithm. Along the way, we highlight how the model and algorithm can be specialized to recover the model and algorithm we developed for RL-MMF in the main text. In addition, Theorem 1 is proved by specializing the analysis of Algorithm 5 to the case of RL-MMF.

C.1 MODEL OF RL-CR

An instance of the RL-CR problem is specified by the tuple $(\mathcal{S}, s_1, \mathcal{A}, T, \mathcal{O}, g)$. The first five quantities are defined in the same way as in the **Set-up** as the RL-MMF problem in Section 3. The last quantity $g : [-1, 1]^K \to \mathbb{R}$ is a concave function, which is L-Lipschitz continuous with respect to the norm $\| \cdot \|$ over $[-1, 1]^K$. That is, for all $u, v \in [-1, 1]^K$ it holds that

$$|g(u) - g(v)| \leq L\|u - v\|.$$

The RL-CR problem has the same **Dynamics** as that the RL-CR problem, while the RL-CR problem's **Objective** is a generalization of the RL-MMF's. In the RL-CR problem, the agent's overarching goal is to design a policy π that maximizes[6] $\mathbb{E}[g(\bar{V}^{\pi}_{1:T})]$. Evidently, RL-MMF is a special case of RL-CR, by specializing $g = g_{\text{min}}$, which is 1-Lipschitz continuous w.r.t. $\| \cdot \|_\infty$.

Similar to the case of RL-MMF, we set our goal as to design a policy π such that

$$\text{Reg}(\pi, T) = \text{opt}(\mathsf{P}(g)) - \mathbb{E}[g(\bar{V}^{\pi}_{1:T})] \leq \frac{D}{T^\gamma} \tag{9}$$

holds for all initial state $s_1 \in \mathcal{S}$ and all $T \in \mathbb{N}$, with parameters $D, \gamma > 0$ independent of T. The benchmark $\text{opt}(\mathsf{P}(g))$ is defined in Section 3.1 as the optimal value of the concave maximization problem $\mathsf{P}(g)$. Similar to the case of RL-MMF, we assume the access to an *optimization oracle* Λ, which satisfies (4) for all initial state $s_1 \in \mathcal{S}$ and all $T \in \mathbb{N}$, with parameters D_{min}, β independent of T.

C.2 ONLINE-REOPT FOR RL-CR

The Online-ReOpt Algorithm for RL-CR is shown in Algorithm 5. Apart from Lines 1, 4, 5, Algorithm 5 is identical to Algorithm 1. Thus, we focus our discussion on Lines 1, 4, 5 in Algorithm 5.

Algorithm 5 Online-ReOpt for concave g

```
 1: Inputs: Scalarization oracle $\Theta$, regularizer $F$, optimization oracle $\Lambda$.
 2: Set $\tau(m) = \lfloor m^{3/2} \rfloor$ for $m \in \mathbb{N}$.
 3: for Episode $m = 1, 2, \ldots, M$ do
 4:     Set $\theta_{\tau(m)} \leftarrow \Theta(\{V_j(s_j, a_j)\}_{j=1}^{\tau(m)-1}, g, F)$.
 5:     Set $\vartheta_{\tau(m)} = -\theta_{\tau(m)}$.
 6:     Compute policy $\pi_m \leftarrow \Lambda(\vartheta_{\tau(m)})$.
 7:     for Time $t = \tau(m), \ldots, \tau(m+1) - 1$ do
 8:         Choose action $a_t = \pi_m(s_t)$.
 9:         Observe the outcomes $V_t(s_t, a_t)$ and the next state $s_{t+1}$.
10:         if $t = T$ then
11:             Break the For loops and terminate the algorithm.
12:         end if
13:     end for
14: end for
```

In Line 1, the Online-ReOpt algorithm requires the input of a scalarization oracle Θ and a regularizer F, in addition to the optimization oracle Λ that is also assumed in the case of RL-MMF. The

---

[6] In the remaining of the Appendix section, we omit the superscript on the policy π for brevity.

inputs Θ, F are used to produce the scalarization $\vartheta_{\tau(m)}$ at each episode m in Lines 4, 5, hinging on the vectorial rewards $\{V_j(s_j, a_j)\}_{j=1}^{\tau(m)-1}$ received and the reward function g. The choice of the *regularizer* F crucially depends on g, and we elaborate on F in the forthcoming section. The scalarization oracle Θ is constructed based on the Follow-The-Regularized-Leader (FTRL) algorithm Shalev-Shwartz (2012) with time varying learning rates (see Chapter 7 in Orabona (2019)). We first describe the scalarization oracle Θ in details in Section C.2.1, then we explain how Algorithm 5 specializes to Algorithm 1 in Section C.2.2.

C.2.1 SCALARIZATION ORACLE Θ

In a nutshell, the scalarization oracle Θ produces the scalarizations by imitating the mechanism of FTRL. We start off the description of Θ by recalling relevant basic concepts in convex optimization.

Convex optimization basics. For a norm $\|\cdot\|$ over $\mathbb{R}^K$, we denote $B_{\|\cdot\|}(L) = \{v \in \mathbb{R}^K : \|v\| \leq L\}$. The dual norm $\|\cdot\|_*$ of $\|\cdot\|$ is defined as $\|\theta\|_* = \max_{v \in B_{\|\cdot\|}(1)}\{\theta^\top v\}$. It is well known that $\|\cdot\|_{**} = \|\cdot\|$.

The Fenchel dual $(-g)^* : \mathbb{R}^K \to \mathbb{R}$ for $-g$, which is convex and L-Lipschitz continuous over $[-1,1]^K$, is defined as

$$(-g)^*(\theta) = \max_{x \in [-1,1]^K} \left\{x^\top \theta + g(x)\right\}.$$

Clearly, the function $(-g)^*$ is convex over the domain $\mathbb{R}^K$, and the function $(-g)^*$ is $\|\mathbf{1}_K\|$-Lipschitz continuous over the domain $B_{\|\cdot\|_*}(L)$. The following Theorem sheds light on why we are interested in considering the Fenchel dual.

Theorem 2 *Suppose that* $g : [-1,1]^K \to \mathbb{R}$ *is concave and* L*-Lipschitz continuous w.r.t. norm* $\|\cdot\|$ *over the domain* $[-1,1]^K$*. For any* $v \in [-1,1]^K$*, it holds that*

$$g(v) = \min_{\theta \in B_{\|\cdot\|^*}(L)} \{(-g)^*(\theta) - \theta^\top v\}.$$

In our discussions with the regularizer F, it is convenient to work with an extended function $H : B \to \mathbb{R} \cup \{\infty\}$, and we denote $\text{dom}(H) = \{\theta \in B : H(\theta) \in \mathbb{R}\}$.

Finally, we recall the notion of sub-gradient. Consider a convex function $f : B_{\|\cdot\|_*}(L) \to \mathbb{R}$, and let $\theta \in B_{\|\cdot\|_*}(L)$. We say that w is a sub-gradient of f at θ, if for all $\theta' \in B_{\|\cdot\|_*}(L)$ it holds that $f(\theta') \geq f(\theta) + w^\top(\theta' - \theta)$. We denote the set of sub-gradients of f at θ as $\partial f(\theta)$, which is non-empty by the assumption of convexity. In addition, if f is L'-Lipschitz continuous w.r.t. to $\|\cdot\|_*$, then it holds that $\|w\| \leq L'$ for all $w \in \partial f(\theta)$.

Description of Θ. We start with the description of the regularizer F, which is an extended function $F : B_{\|\cdot\|_*}(L) \to (-\infty, \infty]$. Given that g is L-Lipschitz continuous over $[-1,1]^K$ with respect to $\|\cdot\|$, we require F to be 1-strongly convex with respect to the norm $\|\cdot\|_*$ over $B_{\|\cdot\|_*}(L)$, that is,

$$\forall\, \theta, \theta' \in \text{dom}(F), \quad \forall\, w \in \partial F(\theta'), \quad F(\theta) \geq F(\theta') + w^\top(\theta - \theta') + \frac{1}{2}\|\theta - \theta'\|_*^2.$$

We display the execution of the scalarization at a time step t, namely $\theta_t \leftarrow \Theta(\{V_q(s_q, a_q)\}_{q=1}^{t-1}, g, F)$, in Algorithm 6. We also inductively assume that the scalarizations $\theta_1, \ldots, \theta_{t-1}$ have been computed. While for Online-ReOpt we only need to apply Θ at time steps $\{\tau(m)\}_{m=1}^M$, we display Θ for general t since the generality is needed for our analysis.

While Line 5 could appear onerous, in actual implementation the sub-gradients $w_1, \ldots, w_T$ are cached so that it only requires the computation of 1 sub-gradient per time step. In equation (11) in Line 7, the argmax is achieved by a unique θ_t, due to the strong convexity of F. In the argmax, we focus on the domain $\text{dom}(F)$ without loss of generality, since the argument is equal to $-\infty$ at any θ outside of $\text{dom}(F)$. The dynamic learning rate follows the approach for FTRL (see Chapter 7 in Orabona (2019) for example). Similar to the case of RL-MMF, the exponent involved in the learning rate is $2/3$ instead of the conventional $1/2$. The chosen value of the exponent turns out to optimize the performance guarantee order bound in our analysis.

To relate Θ to FTRL, it is useful to think about an online convex optimization where convex functions $f_1, \ldots, f_T$ arrive sequentially. At time step t, firstly the online agent has already observed

Algorithm 6 Scalarization Oracle Θ

```
Input: vectorial rewards $\{V_q(s_q, a_q)\}_{q=1}^{t-1}$, reward function $g$.
Input: regularizer $F : B_{\|\cdot\|_*}(L) \to (-\infty, \infty]$.    $\triangleright$ 1-strongly convex w.r.t. $\|\cdot\|_*$.
Input: previous scalarizations $\{\theta_q\}_{q=1}^{t-1}$.
for $q = 1, 2, \ldots, t-1$ do
    Extracts a sub-gradient $w_q \in \partial f_q(\theta_q)$, where we define
        $f_q(\theta) = (-g)^*(\theta) - \theta^\top V_q(s_q, a_q).$    (10)
end for
Compute $\theta_t$ as follows:
    $\theta_t = \text{argmax}_{\theta\in\text{dom}(F)} \left\{ -\eta_{t-1} \cdot \theta^\top \left[ \sum_{q=1}^{t-1} w_q \right] - F(\theta) \right\}.$    (11)
    The dynamic learning rate $\eta_{t-1}$ is defined as
    $\eta_{t-1} = \frac{R}{\|\mathbf{1}_K\| \cdot \max\{(t-1)^{2/3}, 1\}}$, where $R = \sqrt{\max_{\theta\in\text{dom}(F)} F(\theta) - \min_{\theta\in\text{dom}(F)} F(\theta)}.$    (12)
Return $\theta_t$.
```

convex functions $f_1, \ldots, f_{t-1}$. Secondly, the online agent determines θ_t according to Algorithm 6, contingent upon $f_1, \ldots, f_{t-1}$. Finally, the online agent receives f_t as the feedback. Note that the learning rates are non-increasing. In addition, and each f_t is $2\|\mathbf{1}_K\|$-Lipschitz continuous with respect to $\|\cdot\|_*$ over $B_{\|\cdot\|_*}(L)$. Consequently, we can invoke the classical results on FTRL to achieve the following inequality, which is crucial for our analysis. We adopt Corollary 7.9 in Orabona (2019) for the following statement:

Proposition 1 *(Orabona, 2019) Consider the sequence of convex functions $\{f_t\}_{t=1}^T$ defined in (10), and the sequence $\{\theta_t\}_{t=1}^T$ (see 11) generated by applying the scalarization oracle Θ at each t. The following inequalities hold:*

$$\min_{\theta\in B_{\|\cdot\|_*}(L)} \left\{ \sum_{t=1}^T f_t(\theta) \right\} \geq \sum_{t=1}^T f_t(\theta_t) - \left[\frac{R^2}{\eta_{t-1}} + 2\|\mathbf{1}_K\|^2 \sum_{t=1}^T \eta_{t-1} \right]$$
$$\geq \sum_{t=1}^T f_t(\theta_t) - 7R\|\mathbf{1}_K\| \cdot T^{2/3}. \tag{13}$$

The last bound in (13) is by applying the definition of η_t in (12).

C.2.2 SPECIALIZATION TO $g = g_{\text{MIN}}$: FROM FTRL TO MULTIPLICATIVE WEIGHT UPDATES

We assert that we can recover Algorithm 1, the Online-ReOpt algorithm for g_{min}, by specializing that for the scalarization oracle Θ the inputs:

$$g = g_{\text{min}} \text{ and } F(\theta) = \sum_{k=1}^K (-\theta_k)\log(-\theta_k) + I_{-\Delta^K}(\theta), \tag{14}$$

where $I_{-\Delta^K}(\theta) = 0$ if $\theta \in -\Delta^K = \{\theta' \in [-1, 0]^K : \sum_{k=1}^K \theta'_k = -1\}$, and $I_{\Delta^K}(\theta) = \infty$ if $\theta \notin -\Delta^K$. The set $-\Delta^K$ is the negation of the probability simplex, in the sense that for any $\theta \in -\Delta^K$, the negation $-\theta$ is a probability distribution over the K coordinates.

We justify our assertion by showing that under the inputs in (14), the scalarization $\theta_t = \{\theta_{t,k}\}_{k=1}^K$ returned by Θ is

$$\theta_{t,k} = -\frac{\exp\left[-\eta_{t-1}\sum_{j=1}^{t-1} V_{j,k}(s_j, a_j)\right]}{\sum_{\kappa=1}^K \exp\left[-\eta_{t-1}\sum_{j=1}^{t-1} V_{j,\kappa}(s_j, a_j)\right]}, \tag{15}$$

where $\eta_{t-1} = \frac{\sqrt{\log K}}{\max\{(t-1)^{2/3}, 1\}}$. Consequently, we have $\vartheta_{\tau(m)} = -\theta_{\tau(m)}$ as dictated in Line 5, which is in agreement with the scalarization defined in Algorithm 1.

We demonstrate (15) by walking through Algorithm 6. Firstly, we claim that $w_q = \mathbf{1}_K - V_q(s_q, a_q) \in \partial f_q(\theta_q)$ for each q. Indeed, it can be checked that $\mathbf{1}_K \in \partial(-g_{\min})^*(\theta)$ for any $\theta \in \Delta_K$. Consequently, the optimization problem in (11) specializes to

$$\begin{aligned}
&\text{argmax}_{\theta \in -\Delta^K} \left\{ -\eta_{t-1}\theta^\top \left[\sum_{q=1}^{t-1} \mathbf{1}_K - V_q(s_q, a_q) \right] - \sum_{k=1}^{K} (-\theta_k) \log(-\theta_k) \right\} \\
=&\text{argmax}_{\theta \in -\Delta^K} \left\{ (-\theta)^\top \left[-\eta_{t-1} \sum_{q=1}^{t-1} V_q(s_q, a_q) \right] - \sum_{k=1}^{K} (-\theta_k) \log(-\theta_k) \right\} \\
=&-\text{argmin}_{\theta \in \Delta^K} \left\{ \theta^\top \left[-\eta_{t-1} \sum_{q=1}^{t-1} V_q(s_q, a_q) \right] - \sum_{k=1}^{K} \theta_k \log \theta_k \right\}, \qquad (16)
\end{aligned}$$

where we denote $\Delta^K = \{\theta' \in [0,1]^K : \sum_{k=1}^K \theta'_k = 1\}$, and

$$\eta_{t-1} = \frac{R}{\|\mathbf{1}_K\|_\infty \cdot \max\{(t-1)^{2/3}, 1\}} = \frac{\sqrt{\log K}}{\max\{(t-1)^{2/3}, 1\}}.$$

The latter equality is due to the fact that $\max_{\theta \in \text{dom}(F)} F(\theta) = 0$ and $\min_{\theta \in \text{dom}(F)} F(\theta) = -\log K$. Finally, an application of the Lagranage multiplier shows us that θ_t defined in (15) is indeed equal to (16).

C.3 PERFORMANCE GUARANTEES AND ANALYSIS OF ONLINE-REOPT FOR CONCAVE g

We provide the following performance guarantee for the Online-ReOpt on Lipchitz continuous and concave g:

Theorem 3 *Consider the RL-CR problem. Online-ReOpt, displayed in Algorithm 5, satisfies the regret bound:*

$$\textit{Reg}(\pi, T) \leq \frac{114\|\mathbf{1}_K\|R}{T^{1/3}} + \frac{144 D_{\textit{lin}}}{T^{\beta/3}},$$

where $D_{\textit{lin}}, \beta$ are parameters pertaining to the optimization oracle.

Theorem 3 is a generalization result, in the sense that it assumes an optimization oracle that solves scalar reward MDPs to near-optimality (in the sense of regret diminishing with a growing T), and provides an algorithmic framework that solves the RL-CR problem to near-optimality. Theorem 3 specializes to Theorem 1 for the RL-MMF problem. Indeed, under the specialization in (14), we have shown that Algorithm 5 specializes to Algorithm 1, as shown in Section C.2.2. In addition, for $g_{\min}$ the underlying norm is $\|\cdot\|_\infty$, meaning that $\|\mathbf{1}_K\| = 1$, and the choice of regularizer in (14) implies that $R = \sqrt{\log K}$. Consequently, Theorem 3 specializes to Theorem 1.

In the following, we provide a proof to Theorem 3, which also proves the specialization Theorem 1. The proof of Theorem 3 first involves developing two terms (†, ‡) that constitute a lower bound to the expected reward $\mathbb{E}[g(\bar{V}_{1:T})]$. Then, we bound each of the terms (†, ‡) from below respectively according to (20, 21). These lower bounds are justified in Sections C.3.1, C.3.2 respectively, which finishes the proof.

Proof of Theorem 3. We first bound $\mathbb{E}[g(\bar{V}_{1:T})]$ from below as follows:

$$\begin{aligned}
\mathbb{E}[g(\bar{V}_{1:T})] &= \mathbb{E}\left[\min_{\theta \in B_{\|\cdot\|^*}(L)} \{(-g)^*(\theta) - \theta^\top \bar{V}_{1:T}\} \right] \qquad (17) \\
&= \mathbb{E}\left[\min_{\theta \in B_{\|\cdot\|^*}(L)} \left\{ \frac{1}{T} \sum_{t=1}^{T} (-g)^*(\theta) - \theta^\top V_t(s_t, a_t) \right\} \right] \\
&\geq \mathbb{E}\left[\frac{1}{T} \sum_{t=1}^{T} (-g)^*(\theta_t) - \theta_t^\top V_t(s_t, a_t) \right] - \frac{7\|\mathbf{1}\|_K R}{T^{1/3}}. \qquad (18)
\end{aligned}$$

Step (18) is by the bound (13) in Proposition 1. Recall the notation that $f_t(\theta) = (-g)^*(\theta) - \theta^\top V_t(s_t, a_t)$. We decompose the first term in (18) as follows:

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} f_t(\theta_t)\right] = \underbrace{\mathbb{E}\left[\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1}\big(f_t(\theta_t) - f_t(\theta_{\tau(m)})\big)\right]}_{(\dagger)} + \underbrace{\mathbb{E}\left[\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} f_t(\theta_{\tau(m)})\right]}_{(\ddagger)}. \tag{19}$$

We claim that

$$(\dagger) \geq -\frac{107\|\mathbf{1}_K\|R}{T^{1/3}}, \tag{20}$$

$$(\ddagger) \geq \mathrm{opt}(\mathsf{P}(g)) - \frac{144 D_{\text{lin}}}{T^{\beta/3}}, \tag{21}$$

where the parameters D_{lin}, β are parameters involved in the assumption of the optimization oracle.

C.3.1 PROVING (20), WHICH BOUNDS $(\dagger)$ FROM BELOW

To proceed, we first bound each summand from below as

$$f_t(\theta_t) - f_t(\theta_{\tau(m)}) \geq -2\|\mathbf{1}_K\| \cdot \|\theta_t - \theta_{\tau(m)}\|_*,$$

by the $(2\|\mathbf{1}_K\|)$-Lipschitz continuity of f_t with respect to $\|\cdot\|_*$. We then focus on upper bounding the difference $\|\theta_t - \theta_{\tau(m)}\|_*$, which implies a lower bound on the summand. We first make a crucial observation about θ_t.

$$\begin{aligned}\theta_t &= \mathrm{argmax}_{\theta\in\mathrm{dom}(F)}\left\{-\eta_{t-1}\cdot\theta^\top\left[\sum_{q=1}^{t-1} w_q\right] - F(\theta)\right\}\\ &= \nabla F^*\left(-\eta_{t-1}\sum_{q=1}^{t-1} w_q\right),\end{aligned} \tag{22}$$

where F^* is the Fenchel dual of F, exactly same as the way $(-g)^*$ is the Fenchel dual of $-g$:

$$F^*(w) = \max_{\theta\in\mathrm{dom}(F)}\left\{\theta^\top w - F(\theta)\right\}.$$

The connection between θ_t and ∇F_* is in fact well established when the research community makes connection between FTRL and online mirror descent algorithms. To proceed, we use the following result in convex optimization, which can be found in Lemma 15 item 3 in Shalev-Shwartz (2007) for example.

Proposition 2 *(Shalev-Shwartz, 2007) Let F be 1-strongly convex over $B_{\|\cdot\|_*}(L)$ with respect to $\|\cdot\|_*$. Then its Fenchel dual is 1-smooth with respect to $\|\cdot\|$, in the sense that for any $w, u \in \mathbb{R}^K$, it holds that*

$$\|\nabla F^*(w) - \nabla F^*(u)\|_* \leq \|w - u\|.$$

Combining (22) and Proposition 2, we arrive at

$$\|\theta_t - \theta_{\tau(m)}\|_* \leq \left\|-\eta_{t-1}\sum_{q=1}^{t-1} w_q + \eta_{\tau(m)-1}\sum_{q=1}^{\tau(m)-1} w_q\right\|.$$

Subsequenty, we have

$$\begin{aligned}&\left\|-\eta_{t-1}\sum_{q=1}^{t-1} w_q + \eta_{\tau(m)-1}\sum_{q=1}^{\tau(m)-1} w_q\right\|\\ \leq &\sum_{j=\tau(m)-1}^{t-2}\left\|-\eta_{j+1}\sum_{q=1}^{j+1} w_q + \eta_j\sum_{q=1}^{j} w_q\right\|\end{aligned} \tag{23}$$

$$\leq \sum_{j=\tau(m)-1}^{t-2} \eta_{j+1}\|w_{j+1}\| + (\eta_j - \eta_{j+1}) \left\|\sum_{q=1}^{j} w_q\right\| \tag{24}$$

$$\leq 2\|\mathbf{1}_K\| \sum_{j=\tau(m)-1}^{t-2} [\eta_{j+1} + j(\eta_j - \eta_{j+1})] \tag{25}$$

$$\leq \frac{10R}{3} \sum_{j=\max\{\tau(m)-1,1\}}^{t-2} \frac{1}{j^{2/3}} \leq \frac{10R}{3} \cdot \frac{t-\tau(m)}{\max\{(\tau(m)-1)^{2/3},1\}}. \tag{26}$$

Steps (23, 24) is by the triangle inequality. Step (25) is by the fact that $\|w_q\| \leq 2\|\mathbf{1}_K\|$, since $w_q \in \partial f_q(\theta_q)$, and f_q is $2\|\mathbf{1}_K\|$-Lipschitz continuous. Step (26) is by applying the definition of η_t.

Altogether, we have

$$f_t(\theta_t) - f_t(\theta_{\tau(m)}) \geq -\frac{20\|\mathbf{1}_K\|R}{3} \cdot \frac{t-\tau(m)}{\max\{(\tau(m)-1)^{2/3},1\}},$$

and the bound on (†) is proved by summing over m, t:

$$\begin{aligned}
&\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} \big(f_t(\theta_t) - f_t(\theta_{\tau(m)})\big) \\
\geq& -\frac{20\|\mathbf{1}_K\|R}{3} \cdot \frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} \frac{t-\tau(m)}{\max\{(\tau(m)-1)^{2/3},1\}} \\
=& -\frac{20\|\mathbf{1}_K\|R}{3} \cdot \frac{1}{T}\left[1 + \sum_{m=2}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} \frac{t-\tau(m)}{(\tau(m)-1)^{2/3}}\right] \\
\geq& -\frac{10\|\mathbf{1}_K\|R}{3} \cdot \frac{1}{T}\left[2 + \sum_{m=2}^{M} \frac{(\tau(m+1)-\tau(m))^2}{(\tau(m)-1)^{2/3}}\right]. \qquad (27)
\end{aligned}$$

Now, for $m \geq 2$, it is routine to check that $\tau(m+1) - \tau(m) \leq 3\sqrt{m+1}$, and $(\tau(m)-1)^{2/3} = (\lfloor m^{3/2} \rfloor - 1)^{2/3} \geq (m^{3/2}/4)^{2/3} \geq m/3$, so we can continue to bound (27) from below:

$$\begin{aligned}
(27) \geq& -\frac{10\|\mathbf{1}_K\|R}{3} \cdot \frac{1}{T}\left[2 + \sum_{m=2}^{M} \frac{9(m+1)}{m}\right] \\
\geq& -\frac{10\|\mathbf{1}_K\|R}{3} \cdot \frac{1}{T}[2 + 15M] \\
\geq& -\frac{107\|\mathbf{1}_K\|R}{T^{1/3}}, \qquad (28)
\end{aligned}$$

where (28) is because M satisfies $\lfloor M^{3/2} \rfloor \leq T \leq \lfloor (M+1)^{3/2} \rfloor$ implies that $M \leq 2T^{2/3}$. Altogether, the bound for (†) is shown.

C.3.2 PROVING (21), WHICH BOUNDS (‡) FROM BELOW

We bound (‡) by invoking the property of the optimization oracle. Now,

$$\begin{aligned}
(‡) =& \mathbb{E}\left[\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} f_t(\theta_{\tau(m)})\right] \\
=& \mathbb{E}\left[\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} (-g)^*(\theta_{\tau(m)}) - \theta_{\tau(m)}^\top V_t(s_t,a_t)\right] \qquad (29) \\
=& \mathbb{E}\left[\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} (-g)^*(\theta_{\tau(m)}) - \theta_{\tau(m)}^\top v(s_t,a_t)\right]. \qquad (30)
\end{aligned}$$

Step (29) is by recalling that $f_t(\theta) = (-g)^*(\theta) - \theta^\top V_t(s_t, a_t)$ from (10). To this end, recall that the actions in the trajectory $\{s_t, a_t)\}_{t=\tau(m)}^{\tau(m+1)-1}$ are chosen under policy π_m, which is the output of $\Lambda(\vartheta_{\tau(m)})$ and $\vartheta_{\tau(m)} = -\theta_{\tau(m)}$. By our assumption of the optimization oracle (see equation (4)), we know that

$$\frac{1}{\tau(m+1) - \tau(m)}\mathbb{E}\left[(-\theta_{\tau(m)})^\top \sum_{t=\tau(m)}^{\tau(m+1)-1} v(s_t, a_t)\right] \geq \mathsf{opt}(\mathsf{P}(g_{-\theta_{\tau(m)}})) - \frac{D_{\text{lin}}}{[\tau(m+1) - \tau(m)]^\beta}, \tag{31}$$

where $g_{-\theta_{\tau(m)}}(w) = -\theta_{\tau(m)}^\top w$. Combining (30, 31) yields the lower bound

$$(\ddagger) \geq \mathbb{E}\left[\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} (-g)^*(\theta_{\tau(m)}) + \mathsf{opt}(\mathsf{P}(g_{-\theta_{\tau(m)}}))\right] - \frac{1}{T}\sum_{m=1}^{M} D_{\text{lin}}[\tau(m+1) - \tau(m)]^{1-\beta}. \tag{32}$$

We first lower bound the second term in (32), which follows routine calculations:

$$-\frac{1}{T}\sum_{m=1}^{M} D_{\text{lin}}[\tau(m+1) - \tau(m)]^{1-\beta} \geq -\frac{D_{\text{lin}}}{T}\sum_{m=1}^{\lfloor 2T^{2/3}\rfloor} [3(m+1)]^{\frac{1-\beta}{2}} \geq -\frac{144 D_{\text{lin}}}{T^{\beta/3}}.$$

finally, we lower bound the first term in (32). We claim that

$$\frac{1}{T}\sum_{m=1}^{M}\sum_{t=\tau(m)}^{\tau(m+1)-1} (-g)^*(\theta_{\tau(m)}) - \mathsf{opt}(\mathsf{P}(g_{-\theta_{\tau(m)}})) \geq \mathsf{opt}(\mathsf{P}(g))$$

with certainty, by showing that, for any $\theta \in B_{\|\cdot\|_*}(L)$, it holds that

$$(-g)^*(\theta) + \mathsf{opt}(\mathsf{P}(g_{-\theta})) \geq \mathsf{opt}(\mathsf{P}(g)). \tag{33}$$

To show (33), let's define $y_\theta^* = \{y_\theta^*(s, a)\}_{s\in\mathcal{S}, a\in\mathcal{A}_s}$ as an optimal solution to $\mathsf{P}(g_{-\theta})$, and $x^* = \{x^*(s, a)\}_{s\in\mathcal{S}, a\in\mathcal{A}_s}$ as an optimal solution to $\mathsf{P}(g)$. Crucially, note that the optimization problems $\mathsf{P}(g_{-\theta})$, $\mathsf{P}(g)$ have the same feasible region. Since y_θ^* is optimal for the linear program $\mathsf{P}(g_{-\theta})$, we have the following crucial inequality:

$$\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_s} (-\theta)^\top v(s, a) y_\theta^*(s, a) \geq \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_s} (-\theta)^\top v(s, a) x^*(s, a). \tag{34}$$

Consequently, for any $\theta \in B_{\|\cdot\|_*}(L)$,

$$\begin{aligned}
&(-g)^*(\theta) + \mathsf{opt}(\mathsf{P}(g_{-\theta}))\\
=&(-g)^*(\theta) + \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_s} (-\theta)^\top v(s, a) y_\theta^*(s, a)\\
\geq&(-g)^*(\theta) + \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_s} (-\theta)^\top v(s, a) x^*(s, a)\\
\geq&\min_{\theta'\in B_{\|\cdot\|_*}(L)} \left\{(-g)^*(\theta') + \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_s} (-\theta')^\top v(s, a) x^*(s, a)\right\}\\
=&\mathsf{opt}(\mathsf{P}(g)).
\end{aligned}$$

Thus, the bound for (‡) is established, and Theorem 3 is proved.