
Learning Fair Policies in Multiobjective (Deep) Reinforcement Learning with Average and Discounted Rewards

Umer Siddique¹ Paul Weng^{1,2} Matthieu Zimmer¹

Abstract

As the operations of autonomous systems generally affect simultaneously several users, it is crucial that their designs account for fairness considerations. In contrast to standard (deep) reinforcement learning (RL), we investigate the problem of learning a policy that treats its users equitably. In this paper, we formulate this novel RL problem, in which an objective function, which encodes a notion of fairness that we formally define, is optimized. For this problem, we provide a theoretical discussion where we examine the case of discounted rewards and that of average rewards. During this analysis, we notably derive a new result in the standard RL setting, which is of independent interest: it states a novel bound on the approximation error with respect to the optimal average reward of that of a policy optimal for the discounted reward. Since learning with discounted rewards is generally easier, this discussion further justifies finding a fair policy for the average reward by learning a fair policy for the discounted reward. Thus, we describe how several classic deep RL algorithms can be adapted to our fair optimization problem, and we validate our approach with extensive experiments in three different domains.

1. Introduction

The progress in artificial intelligence (AI) and its use in autonomous systems have created a lot of opportunities as well as challenges for human society. Indeed, a well-trained AI system can automate or solve some tasks better than humans (Pilarski et al., 2011; Silver et al., 2017). However,

¹UM-SJTU Joint Institute, Shanghai Jiao Tong University, Shanghai, China ²Department of Automation, Shanghai Jiao Tong University, Shanghai, China. Correspondence to: Paul Weng <paul.weng@sjtu.edu>.

current AI methods do not handle well situations where they impact many users.

The usual approach in those AI systems consists in maximizing a single overall utility (measuring for instance efficiency, accuracy, or task fulfillment). When an AI system affects many users, a *utilitarian* objective is generally adopted, where the individual utilities of all users are summed (or equivalently averaged). Yet, such approach does not guarantee that users are treated equitably. Indeed, in order to obtain an efficient global solution for the utilitarian objective, the utility of some users may be sacrificed. Thus, fairness considerations during the design of autonomous systems are critical if we want users to accept and trust them.

One typical solution to the previous issue is to resort to an *egalitarian* approach where the objective becomes to maximize the utility of the worse-off user. However, a direct application of this maxmin approach may not yield strictly efficient solution for all users because of the focus on only one user. In this paper, we adopt a more refined definition of fairness (Moulin, 2004) that relies on three properties: efficiency, impartiality, and equity (Section 2.3). In order to encode them, we use the generalized Gini evaluation function (Weymark, 1981) as the social welfare function (i.e., the function that defines the overall utility from all the user utilities).

In this work, we study the optimization of this fair welfare function in the context of (deep) reinforcement learning considering both discounted rewards and average rewards.

Our contributions can be summarized as follows:

1. We introduce this novel problem that we call “fair optimization in RL” (Section 3).
2. We investigate its theoretical properties (Section 3.1). Notably, (1) we establish the sufficiency of stationary Markov policies for finding fair solutions, (2) we discuss the possible state-dependency of fair optimality, and (3) we provide an approximation error bound for using a policy optimal for discounted rewards instead of one optimal for average rewards. Interestingly, this last result applied to single-objective RL leads to a novel, simple, and interpretable bound, which is of independent interest.
3. We adapt three deep RL algorithms for solving our fair

optimization problem (Section 4).

4. We provide extensive experimental results in three different domains (Section 5), which validate our propositions.

In the next section, we present the necessary background before presenting our contributions.

2. Background

In this section, we first recall Markov decision processes, and their extension to the multiobjective setting. Then, we motivate and review the welfare function called generalized Gini social welfare function, which encodes fairness.

Notations. Matrices are denoted in uppercase and vectors in lowercase. Both are written in bold. The identity matrix is denoted \mathbf{I} . Vectors are column vectors, except for those denoting probability distributions, which are row vectors.

2.1. Markov Decision Processes

A Markov Decision Process (MDP) (Puterman, 1994) is defined as a tuple of the following elements: a finite set of states \mathcal{S} , a finite set of actions \mathcal{A} , transition matrices \mathbf{P}_a for each $a \in \mathcal{A}$ where $\mathbf{P}_{a,ss'}$ denotes the probability of reaching state s' after performing action a in state s , reward vectors \mathbf{r}_a for each a where $r_{a,s}$ is the reward obtained after performing a in s , and probability distribution \mathbf{d}_0 over initial states. In this model, a *policy* π defines a procedure that specifies how actions are selected in states. A policy is *stationary* if the same procedure is used at every time steps. It is *Markov* if it selects actions only based on the current state. In this paper, unless otherwise stated, policies are stationary and Markov. A policy can be *deterministic* (i.e., $\forall s, \pi(s) \in \mathcal{A}$) or *stochastic* (i.e., $\forall s, a, \pi(a | s)$ denotes the probability of selecting a in s). Deterministic policies are special cases of stochastic ones. By extension, we write \mathbf{P}_π for $\mathbf{P}_{\pi,ss'} = \sum_a \pi(a | s) \mathbf{P}_{a,ss'}$ and \mathbf{r}_π for $\mathbf{r}_{\pi,s} = \sum_a \pi(a | s) \mathbf{r}_{a,s}$. A policy induces a *Markov reward process* whose transitions and rewards are resp. \mathbf{P}_π and \mathbf{r}_π .

In reinforcement learning (RL), \mathbf{P}_a 's and \mathbf{r}_a 's are usually unknown. The goal in an MDP or in RL is to find a policy that optimizes some performance measure, such as the *expected discounted total reward* or *expected average reward*.

Using the discounted-reward criterion, the value function \mathbf{v}_π of a policy π from an initial state s is defined by:

$$\mathbf{v}_{\pi,s} = \mathbb{E}_{\mathbf{P}_\pi} \left[\sum_{t=1}^{\infty} \gamma^{t-1} \mathbf{r}_t | s \right], \quad (1)$$

where $\mathbb{E}_{\mathbf{P}_\pi}$ is the expectation taken with respect to \mathbf{P}_π , $\gamma \in [0, 1)$ is a discount factor, and \mathbf{r}_t is the random variable that represents the reward obtained at time step t . Given

initial distribution \mathbf{d}_0 , an optimal policy is given by:

$$\operatorname{argmax}_{\pi} \mathbf{d}_0 \mathbf{v}_\pi. \quad (2)$$

Interestingly, the objective function can be rewritten as follows: $\mathbf{d}_0 \mathbf{v}_\pi = \mathbf{d}_{\pi,\gamma} \mathbf{r}_\pi$ where $\mathbf{d}_{\pi,\gamma}$ is the *discounted occupation distribution*¹ over states of π , which is defined as $\mathbf{d}_{\pi,\gamma} = \sum_{t=0}^{\infty} \gamma^t \mathbf{d}_0 \mathbf{P}_\pi^t$ with $\mathbf{P}_\pi^0 = \mathbf{I}$ and $\mathbf{P}_\pi^t = \mathbf{P}_\pi^{t-1} \mathbf{P}_\pi$. Value $\mathbf{d}_{\pi,\gamma,s}$ represents the total discounted probability of visiting state s under policy π from initial distribution \mathbf{d}_0 . A policy that is a solution of Problem (2) is called γ -*optimal* and denoted π_γ^* .

Using the average-reward criterion, the value function of a policy π is usually called *gain* and denoted \mathbf{g}_π . For an initial state s , it is defined by:

$$\mathbf{g}_{\pi,s} = \lim_{h \rightarrow \infty} \frac{1}{h} \mathbb{E}_{\mathbf{P}_\pi} \left[\sum_{t=1}^h \mathbf{r}_t | s \right]. \quad (3)$$

Given a distribution over initial states \mathbf{d}_0 , the expected average reward μ_π obtained by a policy π is defined by $\mu_\pi = \mathbf{d}_0 \mathbf{g}_\pi$. It can also be expressed as $\mu_\pi = \mathbf{d}_\pi \mathbf{r}_\pi$ where \mathbf{d}_π is the *stationary distribution* of policy π , which is defined as the Cesàro-limit² of $\mathbf{d}_0 \mathbf{P}_\pi^n$. Distribution \mathbf{d}_π represents the proportion of time policy π spends in each state. For the average-reward criterion, an optimal policy is obtained by:

$$\operatorname{argmax}_{\pi} \mu_\pi. \quad (4)$$

A policy that is a solution to this problem is called *average-optimal* and denoted π_1^* .

The average reward criterion is often preferred in problems where the interaction between agent and environment goes on for a long time horizon. However, the two criteria are intimately connected (Baxter & Bartlett, 2001), i.e., for any policy π , we have $\mathbf{d}_\pi \mathbf{v}_\pi = \frac{\mu_\pi}{1-\gamma}$.

In this paper, we assume that MDPs are *weakly communicating*³. Interestingly, in such MDPs, the optimal gain $\mathbf{g}_{\pi_1^*}$ is constant, i.e., independent of the initial state.

When the state or action space becomes too large or continuous, function approximation is needed to allow generalization. With parametric function approximation (e.g., neural networks or linear function), a function f is approximated

¹Technically speaking, it is not a probability distribution as it is not normalized.

²The Cesàro-limit of sequence u_n as $n \rightarrow \infty$ is given by $\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=0}^{n-1} u_i$. It is a generalized notion of limit and is equal to the standard limit, if the latter exists.

³An MDP is *weakly communicating* if its states can be partitioned into two classes: one in which all states are transient under every stationary policy, and the other in which any two states can be reached from each other under some stationary policy.

by $\hat{f}(\theta)$ where θ denotes the parameters to be learned. In RL, both value functions or policies can be approximated.

Standard deep RL methods are usually designed for discounted rewards. For instance, Deep Q Network (DQN) is an efficient extension of Q-Learning (Mnih et al., 2015). DQN combines bootstrapping, off-policy updates and function approximation. To improve the learning stability it relies on experience replay (Lin, 1991) and target networks. Two approximations of the Q value function $Q_\pi(s, a) = \mathbb{E}_{\mathcal{P}_\pi} [\sum_{t=1}^{\infty} \gamma^{t-1} r_t | s, a]$ are learned respectively parametrized by θ and θ' . The target network associated with θ' is periodically updated towards θ . To update θ , the regression target is: $\hat{Q}_\theta(s, a) = r + \gamma \max_{a' \in \mathcal{A}} \hat{Q}_{\theta'}(s', a')$ where (s, a, s', r) is a tuple drawn from the replay buffer respectively composed of a state, an action, a next state and a reward.

A policy can also be approximated and parametrized by θ . The policy gradient (Sutton et al., 2000) gives the direction in which the parameters should be updated: $\nabla_\theta J(\pi_\theta) = \mathbb{E}_{s \sim d_\pi, a \sim \pi_\theta(\cdot|s)} [A_\pi(s, a) \nabla_\theta \log \pi_\theta(a|s)]$ where $A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s)$ is the advantage function. Since this function is unknown, it needs to be estimated, which can be done in different manners. In Advantage Actor-Critic (A2C), the advantage is estimated by $\hat{A}_{\text{A2C}}(s_t, a_t) = \sum_{t=1}^{\infty} \gamma^{t-1} r_t - \hat{V}(s_t)$ where $\hat{V}(s_t)$ is approximated by a critic network (Mnih et al., 2016). The A2C actor update derives from the policy gradient obtained from $J_{\text{A2C}}(\pi_\theta) = \mathbb{E}_{s \sim d_\pi, a \sim \pi_\theta(\cdot|s)} [\hat{A}_{\text{A2C}}(s, a)]$. In Proximal Policy Optimization (PPO), the advantage \hat{A}_{PPO} is estimated with λ -returns (Schulman et al., 2017). It also derives from the policy gradient but with an additional constraint mitigating the policy changes. It is obtained from $J_{\text{PPO}}(\pi_\theta) = \mathbb{E}_{s \sim d_\pi, a \sim \pi_\theta(\cdot|s)} [\min(\rho_\theta \hat{A}_{\text{PPO}}(s, a), \bar{\rho}_\theta \hat{A}_{\text{PPO}}(s, a))]$ where $\bar{\rho}_\theta = \text{clip}(\rho_\theta, 1 - \delta, 1 + \delta)$, $\rho_\theta = \frac{\pi_\theta(a|s)}{\pi_b(a|s)}$, π_b is the policy that generates the transitions and δ is a hyperparameter to control the constraint.

2.2. Multiobjective Markov Decision Process

A multiobjective MDP (MOMDP) is an MDP where rewards are vectors (instead of scalars) whose components, called *objectives*, are interpreted as different criteria (e.g., length, cost, duration) in the multicriteria setting, and as individual utilities in the multi-agent setting. Formally, the reward function of an MOMDP is redefined as follows: $\mathbf{R}_{a,s} \in \mathbb{R}^D$ where D is the number of objectives. Consequently, value functions (now denoted \mathbf{V} , \mathbf{Q} , \mathbf{G}) also take values in \mathbb{R}^D .

All the previous definitions for MDPs extend naturally to MOMDPs. Notably, with discounted rewards, (1) becomes:

$$\mathbf{V}_{\pi,s} = \mathbb{E}_{\mathcal{P}_\pi} \left[\sum_{t=1}^{\infty} \gamma^{t-1} \mathbf{R}_t | s \right], \quad (5)$$

where \mathbf{V}_π can be seen as a $|\mathcal{S}| \times D$ matrix and \mathbf{R}_t represents the random vector reward obtained at time step t .

With average reward, the gain (3) becomes:

$$\mathbf{G}_{\pi,s} = \lim_{h \rightarrow \infty} \frac{1}{h} \mathbb{E}_{\mathcal{P}_\pi} \left[\sum_{t=1}^h \mathbf{R}_t | s \right], \quad (6)$$

where \mathbf{G}_π can be seen as a $|\mathcal{S}| \times D$ -dimensional matrix.

Multiobjective optimization in MOMDPs amounts to solving the following problem: $\text{argmax}_\pi \mathbf{J}(\pi)$ where $\mathbf{J}(\pi)$ is the multiobjective version of either (2) or (4), and the vector maximization is with respect to *Pareto dominance*⁴. As there is no risk of confusion, Pareto dominance is simply denoted \geq for its weak form and $>$ for its strict form.

A policy whose value function (or gain) is not Pareto-dominated is called *Pareto-optimal*. The usual approach in MOMDPs is to compute all the Pareto-optimal solutions. However, the number of such solutions may be very large in some problems and such approach may be infeasible in general. Indeed, there are some MOMDP instances where the number of Pareto-optimal policies are exponential in the MOMDP size (Perny et al., 2013).

In practice, and especially in RL with an autonomous agent, one is rather interested to focus on one solution, typically one that finds a good balance between all the objectives. One naive way to focus on only one solution is to use a weighted sum to combine the objectives. However, this technique does not provide any control on how balanced the objective values are. A better method is to use a non-linear function to combine the objectives. In our context of multiple users, finding balanced solutions amounts to finding fair solutions. We detail our approach for fairness next.

2.3. Generalized Gini Social Welfare Function

In this paper, we require an optimal solution to satisfy three properties to qualify as a fair solution:

Efficiency A fair solution should be Pareto-optimal.

Impartiality A fair solution should satisfy the “*equal treatment of equals*” principle, which states that users with identical characteristics should be treated similarly.

Equity A fair solution should satisfy the *Pigou-Dalton principle* (Moulin, 2004). Intuitively, this principle states that given a utility vector $\mathbf{v} \in \mathbb{R}^D$, a transfer from a better-off user to a worse-off user yields a new vector that should be preferred. Formally, for any indices i and j , if $v_i > v_j$, then for any ϵ such that $v_i - v_j > \epsilon > 0$, the new vector $\mathbf{v} - \epsilon \mathbf{e}_i + \epsilon \mathbf{e}_j$ is preferred to \mathbf{v} , where \mathbf{e}_i denotes the i -th canonical basis vector⁵.

⁴ $\forall \mathbf{v}, \mathbf{v}' \in \mathbb{R}^D$, \mathbf{v} weakly Pareto-dominates $\mathbf{v}' \Leftrightarrow \forall i, v_i \geq v'_i$. Besides, \mathbf{v} Pareto-dominates $\mathbf{v}' \Leftrightarrow \forall i, v_i \geq v'_i$ and $\exists j, v_j > v'_j$.

⁵ Vector \mathbf{e}_i is such that $e_{ij} = 0$ for $i \neq j$ and $e_{ii} = 1$.

The first property is natural because choosing a Pareto-dominated solution would be irrational. The second one is reasonable in the context of fairness. It holds in our work by assumption: we assume that all the objectives are equal and should therefore be treated in the same way. The third one is the key property in the context of fair optimization, as it captures in a natural way the idea that we prefer solutions whose utility distribution over users is balanced.

In order to implement concretely those three principles, we resort to a welfare function called the *generalized Gini social welfare function* (GGF) (Weymark, 1981). GGF is defined as follows:

$$\text{GGF}_{\mathbf{w}}(\mathbf{v}) = \sum_{i=1}^D \mathbf{w}_i \mathbf{v}_i^{\uparrow}, \quad (7)$$

where $\mathbf{v} \in \mathbb{R}^D$, $\mathbf{w} \in \mathbb{R}^D$ is a fixed positive weight vector whose components are strictly decreasing (i.e., $\mathbf{w}_1 > \dots > \mathbf{w}_D$), and \mathbf{v}^{\uparrow} corresponds to the vector with the components of vector \mathbf{v} sorted in an increasing order (i.e., $\mathbf{v}_1^{\uparrow} \leq \dots \leq \mathbf{v}_D^{\uparrow}$). Furthermore, we assume without loss of generality that the GGF weight vector \mathbf{w} is normalized and sum to one (i.e., $\mathbf{w} \in [0, 1]^D$ and $\sum_{i=1}^D \mathbf{w}_i = 1$).

GGF satisfies the required three properties (Weymark, 1981). As the GGF weights are positive, GGF is monotonic with respect to Pareto dominance. It therefore satisfies the efficiency property. Because the components of \mathbf{v} are reordered in (7), GGF is symmetric with respect to its components. It therefore satisfies the impartiality property. Finally, because the GGF weights are positive and decreasing, GGF is Schur-concave (i.e., it is monotonic with respect to Pigou-Dalton transfers). It therefore satisfies the equity property.

Besides, GGF is a piecewise-linear concave function. Indeed, it is easy to check that GGF can be rewritten as follows thanks to its positive decreasing weights:

$$\text{GGF}_{\mathbf{w}}(\mathbf{v}) = \min_{\sigma \in \mathbb{S}_D} \mathbf{w}_{\sigma}^{\top} \mathbf{v}, \quad (8)$$

where \mathbb{S}_D is the symmetric group of degree D (i.e., set of permutations over $\{1, \dots, D\}$), σ is a permutation, and $\mathbf{w}_{\sigma} = (\mathbf{w}_{\sigma(1)}, \dots, \mathbf{w}_{\sigma(D)})$. Equation (8) holds since the minimum is attained by assigning the largest weight to the smallest component of \mathbf{v} , the second-largest weight to the second-smallest component of \mathbf{v} , and so on.

Although GGF is not the only fair welfare function, it enjoys nice properties: (1) simplicity, as it is a weighted sum in the Lorenz space (Chakravarty, 1990; Perny et al., 2013), (2) its well-understood properties axiomatized by Weymark (1981), (3) its generality.

GGF can cover various special cases by setting its weights appropriately, e.g.:

- If $w_1 \rightarrow 1, w_2 \rightarrow 0, \dots, w_D \rightarrow 0$, GGF corresponds to the maxmin egalitarian notion of fairness (Rawls, 1971).
- If $w_1 \rightarrow 1, w_2 \rightarrow \varepsilon, \dots, w_D \rightarrow \varepsilon$, GGF corresponds to the regularized maxmin egalitarian notion of fairness.
- If $w_1 \rightarrow 1/D, \dots, w_D \rightarrow 1/D$, GGF corresponds to the utilitarian approach.
- If $w_k/w_{k+1} \rightarrow +\infty$, GGF corresponds to the leximin notion of fairness (Rawls, 1971; Kurokawa et al., 2015).

3. Fair Policies in RL

By integrating GGF with MOMDPs, we can now formally formulate the *fair optimization* problem investigated in this paper, which is the problem of determining a policy that generates a fair distribution of rewards to D fixed users:

$$\operatorname{argmax}_{\pi} \text{GGF}_{\mathbf{w}}(\mathbf{J}(\pi)), \quad (9)$$

where $\mathbf{J}(\pi)$ can be defined with the discounted or average reward. As GGF is a concave function, (9) defines a convex optimization problem. This problem defined with discounted rewards is called *GGF- γ problem*, while that with average rewards is called *GGF-average problem*. Their solutions are respectively called *GGF- γ -optimal* and *GGF-average-optimal* policies.

In this paper, we aim at solving this problem in the RL setting. As GGF is a non-linear function, fair optimization is a non-linear convex optimization problem. This brings novel difficulties, which we discuss next.

3.1. Theoretical Discussions

In this part, we discuss three important points related to fair optimization in MOMDPs: (i) which subset of policies is guaranteed to contain an optimal solution, (ii) fair solution may depend on initial states, and (iii) how close is the GGF of the average vector reward of a GGF- γ -optimal policy to that of the optimal average vector reward. The proofs of our theoretical results can be found in Appendix A.

Sufficiency of Stationary Markov Policies. A first question related to Problem (9) is which types of policies are optimal among the set of all (possibly non-stationary) policies. The following lemma, which has not been stated and proved formally to the best of our knowledge, shows that there always exists a GGF- $(\gamma$ or average)-optimal stationary stochastic Markov policy for Problem (9).

Lemma 3.1. *For any MOMDP, Problem (9) admits a solution that is a stationary stochastic Markov policy.*

Note that this result holds in fact for any monotonic function, not only GGF in Problem (9). It implies that one can search for an optimal policy in the smaller set of stationary stochastic Markov policies instead of the set of all policies. Also,

note that contrary to the single-objective case, a deterministic policy may not be optimal (Busa-Fekete et al., 2017) because fairer solution can be obtained via randomization.

Possibly State-Dependent Optimality. For the GGF- γ problem, it is known that optimality depends on initial states or more generally on the distribution over initial states. (see Example 3.2).

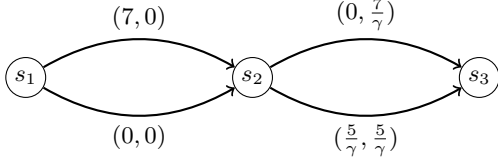


Figure 1. Example of MDP where optimality for Problem (9) with discounted rewards depends on states.

Example 3.2. The following example adapted from (Ogryczak et al., 2013) shows that a GGF- γ -optimal policy depends on initial states. Consider the deterministic bi-objective three-state MDP depicted in Figure 1 where arcs represent actions, arc weights correspond to vector rewards, and $\gamma \in [0, 1)$ is a discount factor. Each state has two actions (Up, Down), except state s_3 , which is an absorbing state. Assume $w = (5/9, 4/9)$. Then, from s_1 , the optimal policy π_1 chooses Up in s_1 , and Up in s_2 . However, viewed from s_2 , π_1 is not optimal, because action Down is preferred in s_2 .

This point raises a potential difficulty when applying a γ -optimal policy: depending on which state is visited, one may have an incentive to switch to another policy. In decision theory, this issue is called *dynamic inconsistency of preferences* (McClennen, 1990). Besides, it implies that the Bellman principle of optimality does not hold anymore and therefore dynamic programming cannot be directly used for computing fair optimal solutions. However, we can prove that for with average reward, preferences stay state-independent as the gain of the optimal policy is constant.

Lemma 3.3. For any weakly-communicating MOMDP, the GGF-average problem admits a solution that is a stationary stochastic Markov policy with constant gain.

Approximation Error A common practice in RL is to find an approximately optimal policy for the average reward by solving the related discounted reward problem. In the single-objective case, (Kakade, 2001) proved that doing so, the difference between the gain of a γ -optimal policy and the optimal gain can be bounded:

Theorem 3.4. Assume $P_{\pi_1^*}$ has n distinct eigenvalues. Let $U = (\mathbf{u}_1, \dots, \mathbf{u}_{|S|})$ be the matrix of its right eigenvectors with the corresponding eigenvalues $\lambda_1 = 1 > |\lambda_2| \geq \dots \geq |\lambda_{|S|}|$. Then,

$$\mu_{\pi_\gamma^*} \geq \mu_{\pi_1^*} - \kappa_2(\mathbf{U}) \|\mathbf{r}\| \frac{1 - \gamma}{1 - \gamma |\lambda_2|},$$

where $\kappa_2(\mathbf{U}) = \|\mathbf{U}\|_2 \|\mathbf{U}^{-1}\|_2$ is the condition number of \mathbf{U} , $\|\mathbf{U}\|_2 = \max_{\mathbf{x}: \|\mathbf{x}\|=1} \|\mathbf{U}\mathbf{x}\|$, and $\|\mathbf{x}\|$ is the Euclidean norm for any vector \mathbf{x} .

To the best of our knowledge, this is the only known bound for such approximation error. However, it only holds if $P_{\pi_1^*}$ has n distinct eigenvalues and it may be hard to interpret and evaluate in practice as it involves the condition number of the corresponding eigenvector matrix.

This motivates us to prove an alternative bound. Using a matrix decomposition approach based on Laurent series expansion and Drazin generalized inverse, Lamond & Puterman (1989) proved the following relation between the discounted value function and the gain of a stationary policy.

Theorem 3.5. For any MDP, any stationary policy π , and any $\gamma \in (\frac{\sigma(\mathbf{H}_{P_\pi})}{\sigma(\mathbf{H}_{P_\pi})+1}, 1)$,

$$\mathbf{v}_\pi = \frac{1}{1 - \gamma} \mathbf{g}_\pi + \frac{1}{\gamma} \sum_{n=0}^{\infty} \left(\frac{\gamma - 1}{\gamma} \right)^n \mathbf{H}_{P_\pi}^{n+1} \mathbf{r}_\pi, \quad (10)$$

where \mathbf{H}_{P_π} is the Drazin inverse of $\mathbf{I} - P_\pi$, which is given by $(\mathbf{I} - P_\pi + P_\pi^*)^{-1} (\mathbf{I} - P_\pi^*)$, P_π^* is the Cesàro-limit of P_π^n for $n \rightarrow \infty$, and $\sigma(\mathbf{H}_{P_\pi})$ is the spectral radius of matrix \mathbf{H}_{P_π} .

Using Th. 3.5, and assuming γ close enough to 1, we can prove an error bound for GGF (policies are GGF-optimal):

Theorem 3.6. For any weakly-communicating MOMDP:

$$\text{GGF}_w(\mu_{\pi_\gamma^*}) \geq \text{GGF}_w(\mu_{\pi_1^*}) - \bar{\mathbf{R}}(1 - \gamma) \left(\rho(\gamma, \sigma(\mathbf{H}_{P_{\pi_1^*}})) + \rho(\gamma, \sigma(\mathbf{H}_{P_{\pi_\gamma^*}})) \right),$$

where $\bar{\mathbf{R}} = \max_\pi \|\mathbf{R}_\pi\|_1$ and $\rho(\gamma, \sigma) = \frac{\sigma}{\gamma - (1 - \gamma)\sigma}$.

Interestingly, Theorem 3.6 applied to the single-objective case ($D = 1$) yields an alternative approximation error bound, which is more general than that of Theorem 3.4.

Corollary 3.7. For any weakly-communicating MDP:

$$\mu_{\pi_\gamma^*} \geq \mu_{\pi_1^*} - \bar{\mathbf{r}}(1 - \gamma) \left(\rho(\gamma, \sigma(\mathbf{H}_{P_{\pi_1^*}})) + \rho(\gamma, \sigma(\mathbf{H}_{P_{\pi_\gamma^*}})) \right)$$

where $\bar{\mathbf{r}} = \max_\pi \|\mathbf{r}_\pi\|$.

The bounds in Theorem 3.6 and Corollary 3.7 clearly show that when $\gamma \rightarrow 1$, the approximation error tends to zero as expected. Like in the bound in Theorem 3.4, the error depends on instance-specific constants. Here, it mainly depends on the spectral radius of the Drazin inverse of $\mathbf{I} - P_\pi$ where π is either a GGF- γ -optimal or GGF-average-optimal policy. This spectral radius can be intuitively interpreted as a measure of how long the policy could spend in transient states (and therefore, as a measure of how long the policy takes to converge to its average reward). Indeed, a larger $\sigma(\mathbf{H}_{P_\pi})$ implies a larger $\rho(\gamma, \sigma(\mathbf{H}_{P_\pi}))$ and a larger bound.

4. Algorithms

In this section, we explain how to modify the DQN and policy gradient algorithms in order to solve Problem (9) with discounted rewards. As previously discussed, it can provide approximate solutions to that problem with average rewards for γ close enough to 1.

DQN. To optimize GGF, we modify the deep Q network (DQN) to take their values in $\mathbb{R}^{|\mathcal{A}| \times D}$ instead of $\mathbb{R}^{|\mathcal{A}|}$. DQN is trained to predict the multiobjective Q function. Note that directly predicting the GGF values would have prevented bootstrapping. Thus, the regression target of DQN becomes:

$$\hat{Q}_\theta(s, a) = r + \gamma \hat{Q}_{\theta'}(s', a^*),$$

where $a^* = \operatorname{argmax}_{a' \in \mathcal{A}} \operatorname{GGF}_{\mathbf{w}}(r + \gamma \hat{Q}_{\theta'}(s', a'))$. This adapted version of DQN is called *GGF-DQN*.

Note that ideally, a^* should be selected as $\operatorname{argmax}_{a' \in \mathcal{A}} \operatorname{GGF}_{\mathbf{w}}(\mathbb{E}_{s'}[r + \gamma \hat{Q}_{\theta'}(s', a')])$. However, it is hard to compute even if the expectation is estimated by a sample mean. Therefore, our modification of DQN optimizes in fact an expectation of GGF, and not a GGF of an expectation. By Jensen inequality, this implies that we are actually optimizing a lower bound of the correct objective function.

Policy Gradient Methods. A natural alternative approach for solving Problem (9) is to use a policy gradient method. Contrary to the adaptation of DQN, it directly optimizes the desired objective function. Another advantage is that it can learn a stochastic policy, which may strictly dominate a deterministic one for GGF.

The policy gradient is formulated as follows for GGF:

$$\begin{aligned} \nabla_{\theta} \operatorname{GGF}_{\mathbf{w}}(\mathbf{J}(\pi_{\theta})) &= \nabla_{\mathbf{J}(\pi_{\theta})} \operatorname{GGF}_{\mathbf{w}}(\mathbf{J}(\pi_{\theta})) \cdot \nabla_{\theta} \mathbf{J}(\pi_{\theta}) \\ &= \mathbf{w}_{\sigma}^{\top} \nabla_{\theta} \mathbf{J}(\pi_{\theta}), \end{aligned}$$

where $\nabla_{\theta} \mathbf{J}(\pi_{\theta})$ is a $D \times N$ matrix representing the classic policy gradient over the D objectives, \mathbf{w}_{σ} is sorted according to $\mathbf{J}(\pi_{\theta})$ and N is the number of policy parameters.

In the experiments, we applied it to PPO and A2C: $\nabla_{\theta} \mathbf{J}(\pi_{\theta})$ is respectively replaced by $\nabla_{\theta} \mathbf{J}_{\text{PPO}}(\pi_{\theta})$ and $\nabla_{\theta} \mathbf{J}_{\text{A2C}}(\pi_{\theta})$. In order to sort \mathbf{w} , the initial states are stored to empirically estimate $\mathbf{J}(\pi_{\theta}) = \mathbb{E}_{s_0 \sim d_0}[\hat{V}(s_0)]$ where $\hat{V} : \mathcal{S} \rightarrow \mathbb{R}^D$ is approximated with the critic. The resulting algorithms are called respectively *GGF-PPO* and *GGF-A2C*.

5. Experimental Results

To test our three algorithms, we carried out experiments in three different domains (for detailed descriptions, see Appendix B): (i) Species conservation (SC), (ii) Traffic light

control (TL), (iii) Data center control (DC). The first domain (SC) corresponds to a conservation problem encountered in ecology, where the goal is to maintain the populations of several interacting endangered species. We adapt the two-species model proposed by Chadès et al. (2012) to specifically take into account fairness with respect to the two species, namely an endangered species (sea otters) and its prey (northern abalone). A state encodes the population numbers of the species. The transition function is based on the population growth models for both species taking into account factors such as poaching (for abalones) or oil spills (for sea otters). In order to keep the two populations balanced, five actions are considered: do nothing introduce sea otters, enforce antipoaching, control sea otters, and one-half antipoaching and one-half control sea otters. Vector rewards correspond to scaled species densities (in m^{-2}).

The second domain (TL) corresponds to the classic traffic light control problem, in which an agent controls the traffic lights at one intersection in order to optimize traffic flow. The usual approach to this problem amounts to minimizing the expected sum of waiting times over all lanes. Instead, we propose to take into account fairness with respect to each road ($D = 4$ in our experiments). In other words, the goal is to learn a controller that optimizes the expected waiting times per road. More specifically, we consider an eight-lane intersection, where the four directions have 2 lanes. A state is composed of the total waiting time and density of cars (in $[0, 1]$) stopped at the intersection in each lane. It also contains the current phase (i.e., which lanes and directions have green/red lights) of the traffic lights. An action corresponds to a change of phase. Traffic randomly generated with fixed distributions. At each time step vehicles are emitted randomly with the given probability by following a binomial distribution. The binomially distributed flow approximates a Poisson distribution for small probabilities that a number of random events happens with certain rate independently. A reward is a vector whose components are the sums of negative waiting times for each lane.

The third domain (DC) is a data center control problem, where a centralized controller manages a computer network that is shared by a certain number of hosts in order to optimize the bandwidths of each host (Ruffy et al., 2019). Here, fairness is expressed with respect to hosts ($D = 16$ in our experiments). This kind of problem can typically occur in software-defined networking (SDN) for instance. A state encodes network statistics (i.e., queue length, derivative over time of queue length, number of packet drops, and queue length above some limit) and the current bandwidth allocation to hosts. An action is a vector of bandwidth allocation. Traffic between hosts is randomly generated. A reward is a vector whose components are bandwidths per host penalized by a sum of queue lengths (in order to avoid bufferbloat).

The three domains are roughly ranked in increasing complexity (with also increasing number of objectives). The first two domains have discrete state-action spaces, while the third has continuous state-action spaces. As they are all episodic problems, they are all communicating (MO)MDPs.

On those domains, we typically ran DQN, PPO, A2C, and their adaptations to GGF (i.e., GGF-DQN, GGF-PPO, GGF-A2C). The hyperparameters of the algorithms were optimized (Appendix C) and all the experiments were conducted using Lightweight HyperParameter Optimizer (LHPO), an open source library used to run parallel experiments on a cluster (Zimmer, 2018). Two computers with double CPU sockets have been used (Intel Xeon CPU E5-2678 v3). The unnormalized GGF coefficients are defined as $w_i = \frac{1}{2^i}$ from 0 to $D - 1$. All the experimental results (e.g., plots) are averaged over 50 (resp. 20) runs with different seeds for SC and TL (resp. DC as it is a complex environment).

We now present the main results of our experiments (for more, see Appendix D). They have been designed to answer the following questions: **(A)** What is the impact of optimizing GGF instead of the average of the objectives? **(B)** How do the algorithms adapted to GGF compare with each other and with their standard versions? **(C)** How do fair deterministic and stochastic policies compare? **(D)** What is the effect of γ with respect to GGF-average optimality? **(E)** How do those algorithms perform in continuous domains?

Question (A) In order to answer (A), we discuss the experimental results of DQN, A2C, PPO and their GGF counterparts in the SC domain. We can first compare them in terms of the average over the two accumulated densities during learning phase (see Figure 2). As a sanity check, the figure also includes the uniformly random policy. As densities are accumulated over an episode, drops happen in the curves of all algorithms. As expected, the random policy performs the worse and standard RL algorithms are the best because the average density is roughly what is optimized in those algorithms. An algorithm adapted for GGF would normally perform worse than its original version in terms of the average of the objectives, since it trade-offs between efficiency and equity.

More interestingly, for our purposes, we can compare the algorithms in terms of the GGF score. After training, the obtained policies are applied 50 times in the environment. This score is the GGF of the sample average vector rewards of the generated trajectories. Figure 3 shows the distribution of this score for the policies learned by DQN, A2C, PPO and their GGF algorithms. The number of steps during training and testing is 3650 (corresponding to 10 years). As expected, all the three GGF algorithms have higher GGF score than their original algorithms.

As the GGF score does not directly give a clear picture of

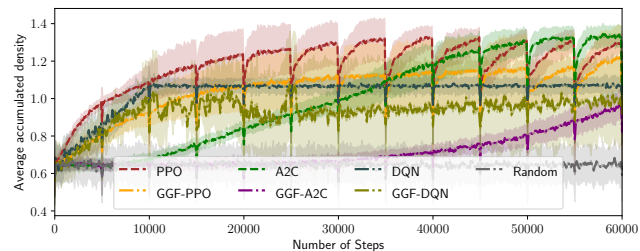


Figure 2. Average accumulated densities of DQN, A2C, PPO and their GGF versions during the learning phase with those of the random policy in the SC domain.

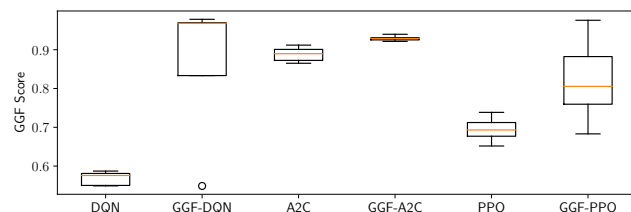


Figure 3. GGF scores of DQN, A2C, PPO and their GGF algorithms during the testing phase in the SC domain.

how balanced the objectives are, we also provide the plots of non-aggregated accumulated densities estimated after training (Figure 4), which can easily be done for the SC domain as it is bi-objective. We can observe again that standard RL algorithms obtain higher total accumulated densities than their GGF counterparts. However, the individual densities of the two species for the standard approaches are much more unequal than those obtained with our approach, which provides much fairer solutions.

Because we cannot easily display the non-aggregated objectives in all the domains, we introduce additional statistics to evaluate fairness. Notably, the *Coefficient of Variation* (CV), which can be understood as a simple measure of inequality. In Figure 5, every algorithm optimizing GGF have a lower CV and a higher minimum density.

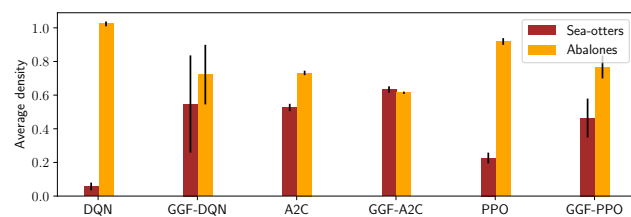


Figure 4. Individual densities for DQN, A2C, PPO and their GGF versions during the testing phase in the SC domain.

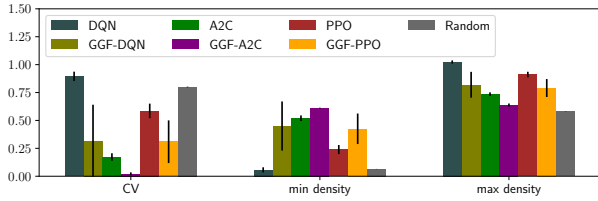


Figure 5. CV, minimum and maximum densities of DQN, A2C, PPO and their GGF counterparts during the testing phase in the SC domain.

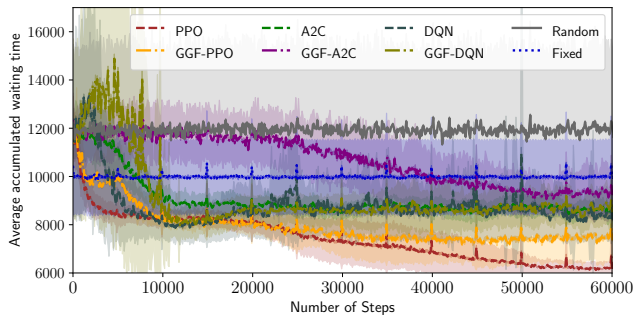


Figure 6. Average waiting times of DQN, A2C, PPO, and their GGF counterparts during learning phase, and those of the fixed and random policies in the TL domain.

Question (B) To answer (B), we turn to the TL domain, a more complex environment. Figure 6 shows the waiting times averaged over all the lanes obtained during the learning phase by the six different RL algorithms we considered. As a reference, we added the performances of the random policy and a fixed policy. This latter policy cycles between all the phases at a fixed frequency, which has been optimized over many simulations. The fixed policy naturally performs better than the random one. When training ends, it is worse than all RL algorithms. As expected, all standard algorithms performs better than their GGF counterparts, because the average accumulated waiting times correspond to the measure optimized by the original algorithms.

Figure 7 depicts the GGF score (over minus waiting times) computed after training. It also includes results for PPO and GGF-PPO with two different values of γ (i.e., 0.99 and close to 1), which we discuss in the answer to (D). Although, the fixed policy is not included for space reasons as it has the worse GGF-performance, we note that GGF-DQN performs better than the fixed policy, which shows that GGF-DQN has indeed optimized for fairness. All the three GGF algorithms have better GGF scores than their original counterparts. GGF-PPO achieves the best score.

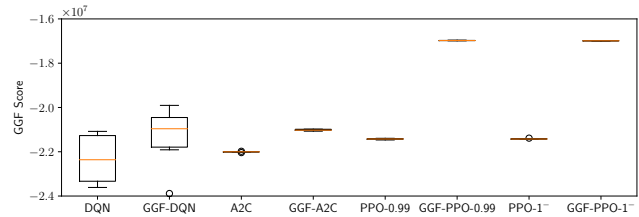


Figure 7. GGF scores of DQN, A2C, PPO, and their GGF versions, with those of PPO and GGF-PPO when γ is close to 1, during the testing phase in the TL domain.

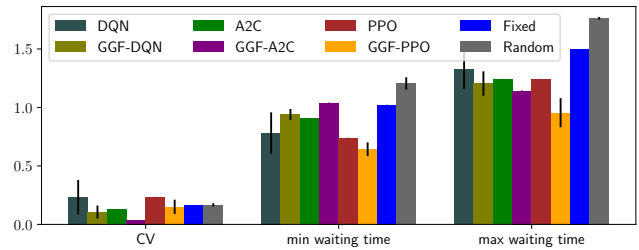


Figure 8. CV, minimum and maximum waiting times of DQN, A2C, PPO and their GGF counterparts during the testing phase in the TL domain. The minimum and maximum waiting times have been divided by 3000 to be displayable with the CV.

To confirm that those high GGF scores correspond to fairer solutions, we can indeed observe in Figure 8 that our proposed algorithms always achieve a lower CV than their original algorithms. Among all the algorithms, PPO performs the best as it has the lowest waiting time. Similar to PPO, GGF-PPO also has lower waiting times in all roads but with more balanced distributions of waiting times in each road.

Question (C) Algorithms optimizing a stochastic policy often perform better than DQN in terms of the average (or total) of the objectives (see Figure 2, Figure 6, or Figure 23 in Appendix). However, in terms of GGF, for a simple domain like SC, GGF-DQN actually performs well, while the conclusion is reversed for a more complex domain such as TL. This may be due to the partial observability of the domain and the use of GGF, which calls for stochastic policies for fairer solutions (as suggested by our theoretical discussion). Figure 6 indicates that the price of fairness (i.e., loss in terms of the average of the objectives for optimizing GGF instead of a utilitarian criterion) is limited.

Question (D) We also ran the algorithms with γ very close to 1, i.e., $\gamma = 0.99999$. The last two boxplots of Figure 7 show that the results for GGF-PPO are very similar to those with $\gamma = 0.99$. This indicates that the policy found

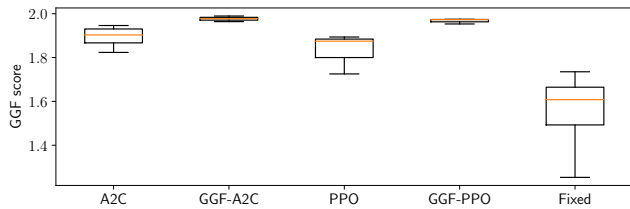


Figure 9. GGF scores of A2C, PPO, and their GGF versions, with those of a fixed policy, during the testing phase in the DC domain.

by GGF-PPO is close to GGF-average optimal. Also, this suggests that in practice, except for difficult MDP structures, using $\gamma = 0.99$ is sufficient.

Question (E) Although our theoretical discussion concerned finite MOMDPs, we conjecture that similar results could be obtained in continuous spaces by adding some usual technical conditions (Arapostathis et al., 1993). Therefore, we also tried our approach (with A2C and PPO) on the DC domain, whose states and actions are continuous.

Figure 9 illustrates the offline performance of A2C, PPO, GGF-A2C, and GGF-PPO in terms of GGF score. As a reference, it also includes the GGF score of a fixed policy. As expected, the fixed policy has the lowest GGF score. GGF-A2C and GGF-PPO have better GGF scores than their original counterparts. This shows clearly that if a fair policy is important, the usual approach based on a weighted sum to aggregate the objectives (with equal weights) is insufficient.

Figure 10 illustrates the performances of different RL algorithms and their GGF algorithms with those of the fixed and the random policy in terms of their CV, minimum and maximum of bandwidths. As expected, the random policy performs worse as it has the lowest minimum and maximum bandwidths and also obtain the lowest accumulated bandwidth. The fixed policy performs better than random one. The GGF versions of A2C and PPO have lower CV which indicates that their are more fairer than their original algorithms.

For more experimental details and results, see the appendix.

6. Related Work

Fair optimization in applied mathematics, operations research, and theoretical computer science is an active research direction (Ogryczak et al., 2014). Numerous classic continuous and combinatorial optimization problems (Hurkala & Sliwinski, 2012; Ogryczak et al., 2013; Nguyen & Weng, 2017; Lesca et al., 2019) have been extended to take into account fairness. However, these works assume that the whole model is known and therefore only

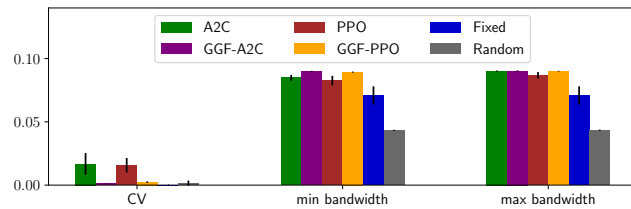


Figure 10. CV, minimum and maximum bandwidths of A2C, PPO and their GGF counterparts during the testing phase in the DC domain.

focuses on the fair optimization problem since no learning is needed. One notable work among them (Ogryczak et al., 2013) solves Problem (9) with discounted rewards using a linear-programming approach, which is possible only if the transition and reward functions are known. In contrast, in our work, we solve Problem (9) in the RL setting considering both discounted and average rewards and also tackling problems with large or even continuous state space.

Fairness considerations have recently become an important topic in machine learning (Busa-Fekete et al., 2017; Speicher et al., 2018; Agarwal et al., 2018; Heidari et al., 2018; Jiang & Lu, 2019; Weng, 2019). Most work focuses on the impartiality aspect of fairness. However, a few notable exceptions consider fair optimization in sequential decision-making problems. Busa-Fekete et al. (2017) investigate a problem similar to ours but in the multi-armed bandit setting. Jiang & Lu (2019) consider the problem of learning fair policies in multi-agent RL where fairness is defined over agents and encoded with a different welfare function. Besides, their focus is on learning decentralized policies in a distributed way using a consensus mechanism. Finally, another recent work deserves to be mentioned here although it does not specifically deal with fairness. Cheung (2019) investigates a problem more general than ours in the UCRL setting (Jaksch et al., 2010) where the focus is on regret minimization for solving efficiently the exploration-exploitation dilemma. That work also deals with non-stationary deterministic policies in tabular MDPs, contrary to our work.

7. Conclusion

In this work, we introduced the novel problem of fair optimization in RL, which we theoretically discussed. We proposed adaptations of three deep RL algorithms to solve large-scale problems and provided an extensive empirical validation. As future work, we may consider other fair welfare functions (Ogryczak et al., 2014), extend to distributed control, or directly adapt RL algorithms for average reward.

Acknowledgements

This work is supported in part by the program of National Natural Science Foundation of China (No. 61872238), the program of the Shanghai NSF (No. 19ZR1426700), and a Yahoo FREP grant.

References

- Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., and Wallach, H. A reductions approach to fair classification. In *ICML*, 2018.
- Arapostathis, A., Borkar, V., Fernandez-Gaucherand, E., Ghosh, M. K., and Marcus, S. I. Discrete-time controlled Markov processes with average cost criterion: a survey. *SIAM Journal on Control and Optimization*, 31(2):282–344, 1993. ISSN 03630129. doi: 10.1137/0331018.
- Baxter, J. and Bartlett, P. L. Infinite-horizon policy-gradient estimation. *JAIR*, 15:319–350, 2001.
- Busa-Fekete, R., Szörenyi, B., Weng, P., and Mannor, S. Multi-objective bandits: Optimizing the generalized Gini index. In *ICML*, 2017.
- Chadès, I., Curtis, J. M., and Martin, T. G. Setting realistic recovery targets for two interacting endangered species, sea otter and northern abalone. *Conservation Biology*, 26(6):1016–1025, 2012.
- Chakravarty, S. R. *Ethical Social Index Numbers*. Springer Verlag, 1990.
- Cheung, W. C. Regret Minimization for Reinforcement Learning with Vectorial Feedback and Complex Objectives. In *NeurIPS*, 2019.
- Heidari, H., Ferrari, C., Gummadi, K. P., and Krause, A. Fairness behind a veil of ignorance: A welfare analysis for automated decision making. In *NeurIPS*, 2018.
- Hurkala, J. and Sliwinski, T. Fair flow optimization with advanced aggregation operators in wireless mesh networks. In *Federated Conference on Computer Science and Information Systems*, pp. 415–421, 2012.
- Jaksch, T., Ortner, R., and Auer, P. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600, 2010.
- Jiang, J. and Lu, Z. Learning Fairness in Multi-Agent Systems. In *NeurIPS*, 2019.
- Kakade, S. Optimizing average reward using discounted rewards. In *COLT*, 2001.
- Kurokawa, D., Procaccia, A. D., and Shah, N. Leximin Allocations in the Real World. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, EC '15, pp. 345–362, Portland, Oregon, USA, June 2015. Association for Computing Machinery. ISBN 978-1-4503-3410-5. doi: 10.1145/2764468.2764490.
- Lamond, B. F. and Puterman, M. L. Generalized Inverses in Discrete Time Markov Decision Processes. *SIAM Journal on Matrix Analysis and Applications*, 10(1):118–134, jan 1989. ISSN 0895-4798. doi: 10.1137/0610009.
- Lesca, J., Minoux, M., and Perny, P. The fair owa one-to-one assignment problem: Np-hardness and polynomial time special cases. *Algorithmica*, 81(1):98–123, 2019.
- Lin, L. Programming robots using reinforcement learning and teaching. In *AAAI*, pp. 781–786, 1991.
- McClennen, E. *Rationality and dynamic choice: Foundational explorations*. Cambridge university press, 1990.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016.
- Moulin, H. *Fair Division and Collective Welfare*. MIT Press, 2004.
- Nguyen, V. H. and Weng, P. An efficient primal-dual algorithm for fair combinatorial optimization problems. In *COCOA*, 2017.
- Ogryczak, W., Perny, P., and Weng, P. A compromise programming approach to multiobjective Markov decision processes. *International Journal of Information Technology & Decision Making*, 12:1021–1053, 2013.
- Ogryczak, W., Luss, H., Pióro, M., Nace, D., and Tomaszewski, A. Fair optimization and networks: A survey. *Journal of Applied Mathematics*, 2014, 2014.
- Perny, P., Weng, P., Goldsmith, J., and Hanna, J. Approximation of Lorenz-optimal solutions in multiobjective Markov decision processes. In *International Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013.
- Pilarski, P. M., Dawson, M. R., Degris, T., Fahimi, F., Carey, J. P., and Sutton, R. S. Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning. In *IEEE International Conference on Rehabilitation Robotics*, 2011.

- Puterman, M. *Markov decision processes: discrete stochastic dynamic programming*. Wiley, 1994.
- Rawls, J. *The Theory of Justice*. Harvard university press, 1971.
- Ruffy, F., Przystupa, M., and Beschastnikh, I. Iroko: A framework to prototype reinforcement learning for data center traffic control. In *Workshop on ML for Systems at NeurIPS*, 2019. URL <http://arxiv.org/abs/1812.09975>.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL <http://arxiv.org/abs/1707.06347>.
- Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.
- Speicher, T., Heidari, H., Grgic-Hlaca, N., Gummadi, K. P., Singla, A., Weller, A., and Zafar, M. B. A unified approach to quantifying algorithmic unfairness: Measuring individual & group unfairness via inequality indices. In *KDD*, pp. 2239–2248, 2018. ISBN 978-1-4503-5552-0. doi: 10.1145/3219819.3220046. URL <http://doi.acm.org/10.1145/3219819.3220046>.
- Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, 2000.
- Weng, P. Fairness in reinforcement learning. In *AI for Social Good Workshop at IJCAI*, 2019.
- Weymark, J. Generalized Gini inequality indices. *Mathematical Social Sciences*, 1:409–430, 1981.
- Zimmer, M. *Apprentissage par renforcement développemental*. PhD thesis, University of Lorraine, January 2018.