Bringing Fairness to Actor-Critic Reinforcement Learning for Network Utility Optimization

Jingdi Chen
Department of ECE
The George Washington University
DC 20052, USA
jingdic@gwu.edu

Yimeng Wang
Department of ECE
The George Washington University
DC 20052, USA
wangyimeng@gwu.edu

Tian Lan
Department of ECE
The George Washington University
DC 20052, USA
tlan@gwu.edu

***Abstract*—Fairness is a crucial design objective in virtually all network optimization problems, where limited system resources are shared by multiple agents. Recently, reinforcement learning has been successfully applied to autonomous online decision making in many network design and optimization problems. However, most of them try to maximize the long-term (discounted) reward of all agents, without taking fairness into account. In this paper, we propose a family of algorithms that bring fairness to actor-critic reinforcement learning for optimizing general fairness utility functions. In particular, we present a novel method for adjusting the rewards in standard reinforcement learning by a multiplicative weight depending on both the shape of fairness utility and some statistics of past rewards. It is shown that for proper choice of the adjusted rewards, a policy gradient update converges to at least a stationary point of general α-fairness utility optimization. It inspires the design of fairness optimization algorithms in actor-critic reinforcement learning. Evaluations show that the proposed algorithm can be easily deployed in real-world network optimization problems, such as wireless scheduling and video QoE optimization, and can significantly improve the fairness utility value over previous heuristics and learning algorithms.**

I. Introduction

Fairness is essential for many network optimization problems, e.g., congestion control, routing, power control, spectrum management, load balancing, and flow scheduling [1], [2], [3]. It is well-known that different notions of fairness, including proportional and max-min fairness [4], [5], can be achieved through the maximization of α-fair or isoelastic utility functions (for different choices of α values). An axiomatic framework of fairness for network optimization has been developed in [6], [7], [8].

Many algorithms have been proposed for network optimization to maximize fairness utility objectives. While certain properties, such as convergence and optimality gap, are often obtained in static settings, online optimization of fairness utility in dynamic environments is, in general, a challenging problem [9], [10], [11]. Recently, reinforcement learning (RL) has been successfully applied to autonomous online decision making in many network design and optimization problems, e.g., [12], [13], [14], [15], [16], [17]. However, most of them try to maximize the long-term (discounted) reward of all agents, without taking fairness utilities into account. Only a few proposals consider fairness, but either focus on the heuristic reward function design [18] (thus not able to optimize general fairness utilities) or rely on the Monte Carlo method to sample a large number of paths for estimating policy gradient [19] (thus resulting in extremely slow convergence).

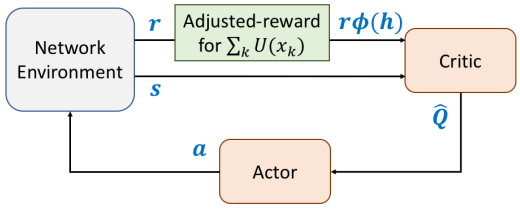

Figure 1: An illustration of our actor-critic reinforcement learning framework, which employs (multiplicative) adjusted-rewards for fairness utility optimization.

This paper aims to develop a family of algorithms that bring fairness to actor-critic reinforcement learning for optimizing general fairness utility functions. Actor-critic reinforcement learning [20] are Temporal Difference (TD) methods that have a separate structure to explicitly represent the policy independent of the value function. In particular, the critic evaluates the current TD error and typically uses a deep neural network to learn a value function. The value function is then used to update the actor's policy parameters in a direction improving the objective value. Actor-critic algorithms leveraging deep neural networks can solve online network optimization with large state/action space, and have fast convergence due to variance reduction. In this paper, we introduce a multiplicative-adjusted reward as shown in Figure 1 and analyze the resulting policy gradient for optimizing general fairness utility functions. We prove that the policy gradient update converges to at least a stationary point of α-fairness utility optimization for proper choice of adjusted rewards. It inspires the design of an actor-critic algorithm to learn a stochastic policy for fairness optimization using the adjusted rewards. To the best of our knowledge, this is the first proposal to achieve fairness utility optimization in actor-critic reinforcement learning.

To illustrate the challenge in taking fairness into reinforcement learning, we consider a network optimization problem that intends to maximize the utility of agents' average data rates. Let $r_{k,t}$ be the reward (i.e., data rate) for agent k in epoch t. It is not hard to see that optimizing the average reward $1/T\sum_{t,k} r_{k,t}$ by reinforcement learning would not maximize the α-fair utility $\sum_k U(1/T\sum_t r_{k,t})$ when the utility function

is non-linear (i.e., for any $\alpha > 0$). In fact, the Markovian property required for formulating the problem as an MDP is no longer satisfied. To overcome this, we propose a new method that adjusts the rewards in reinforcement learning with a multiplicative weight, $\phi(h_{\pi,t})$, which is defined through a uniformly-continuous function ϕ based on the shape of fairness utility and some statistics of past rewards $h_{\pi,t}$ up to epoch t under policy π. By analyzing an underlying MDP (that is obtained through a partially observable MPD [21]) and proving an alternative policy gradient theorem, we show that for proper choice of the function ϕ and statistics $h_{\pi,t}$, the new learning problem with adjusted rewards is guaranteed to converge to at least a stationary point of the α-fair utility optimization. The result naturally leads to a new class of actor-critic reinforcement learning algorithms, which can be applied to a wide range of network utility optimization problems. Unlike previous work [18], [19], the proposed algorithm does not rely on the Monte Carlo method for calculating gradient and has very fast convergence.

We note that many network optimization problems with fairness utilities can be solved efficiently using the proposed framework. It can learn an explicitly stochastic policy by generating the optimal probabilities of selecting various actions. This ability turns out to be useful especially when online optimization and non-convex problem structures are involved. We implement and validate the proposed learning framework in two real-world network optimization problems – wireless network scheduling and Quality-of-Experience (QoE) optimization in video streaming – both of which require online decision making in a dynamic network environment. We show that the proposed algorithm can achieve the optimal policy when the problem is convex (e.g., in wireless network scheduling) and significantly outperform several heuristic and learning algorithms when the problem is non-convex (e.g., in QoE optimization).

The main contributions of the paper are as follows:

- We propose a family of algorithms that introduce adjusted rewards and bring fairness to actor-critic reinforcement learning for optimizing general fairness utilities.
- A policy gradient theorem proven in this paper guarantees convergence to at least a stationary point of the α-fair utility optimization and inspires the design of actor-critic algorithms.
- Significantly improve the fairness utility value over baseline heuristics and learning algorithms.

II. Related Work

Network Optimization and Fairness. Fair resource allocation has been extensively studied in the past in network optimization [1], [2], [3]. Many well-known fairness measures have been proposed, including [22], [23], [24], [25], [26], [27], ranging from simple ratios to more sophisticated functions such as Jain's index [24] and entropy function [28]. A family of α-fair utility functions are developed to balance the fairness problem and the total throughput in network optimization. This includes the max-min fairness [4] and proportional fairness [5] as special cases when $\alpha = \infty$ and $\alpha = 1$, respectively. Later, a systematic framework of fairness measures in network optimization is developed in [6]. It proposes a set of five axioms for fairness measures and is shown to unify many existing notions of fairness. Fairness utilities have also been used in the context of federated training, i.e., [29], to reduce variance while maintaining the average accuracy during training.

Reinforcement Learning for Network Optimization. Deep Reinforcement Learning (RL) algorithms have become increasingly popular in solving network optimization problems, especially when online decision making in a dynamic environment is involved. For wireless networks, a multi-objective fully distributed strategy based on RL is proposed for LTE femtocell self-configuration and optimization [12], while a cell outage management framework for heterogeneous networks is developed in [13]. For traffic engineering problems, an RL based control framework is developed in [14], and an RL-based algorithm for link utilization optimization under different traffic matrices is proposed in [15]. Related work also include RL algorithms for Adaptive Bitrate video streaming [16], QoS-aware adaptive routing [30], [31], packet scheduling [17], and software-defined networks [32]. However, these algorithms relying on standard RL mainly focusing on maximizing the long-term (discounted) reward of all agents, without taking fairness into account.

Learning for Fairness. However, they either focus on heuristic reward function design [18] (thus not able to optimize general fairness utilities or guarantee convergence) or rely on the Monte Carlo method to collect a large number of sampling paths for evaluating policy gradient [19] (thus resulting in extremely slow convergence). In particular, different RL algorithms have been developed for fair resource allocation on multi-core systems [33], maximizing reward distribution among Unmanned Aerial Vehicles [34], and balancing resource allocation in complex logistics networks [35]. Yet, these algorithms rely on heuristic design to improve fairness and cannot be used for general fairness utilities in network optimization. Recently, model-based and model-free RL algorithms for optimizing fairness objective is proposed in [19]. But it relies on the Monte Carlo method for gradient evaluation and thus suffers from extremely slow convergence. To the best of our knowledge, this work is the first proposal to achieve fairness utility optimization in actor-critic RL with provable convergence.

III. Problem Formulation

We consider a network utility optimization, where the network is modeled as an environment and users as learning agents. Agents interact with the environment and learn a policy to optimize an aggregate utility of the individual agents' reward (e.g., data rates). This is considered as an infinite horizon discounted Markov Decision Process (MDP) $\mathcal{M}$ that is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, K, r_1, r_2, \ldots, r_K, \gamma)$. Here $\mathcal{S}$ denotes a set of states, $\mathcal{A}$ a set of actions, and $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ the transition probability distribution. There are K agents, each associated with a reward $r_k : \mathcal{S} \times \mathcal{A} \to [0, 1]$ for $k = 1, \ldots, K$. When distributed execution is employed, each agent k maintains its own state space $\mathcal{S}_k$ and action space

$\mathcal{A}_k$. Then, the joint state space and action space become $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_K$ and $\mathcal{A} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_K$, respectively.

We use a stochastic policy $\pi(a|s)$ to determine the probability of selecting action $a \in \mathcal{A}$ in each state $s \in \mathcal{S}$. Let $x_{\pi,k}$ denote the average reward of agent k under policy π. Then, we have

$$x_{\pi,k} = \lim_{T\to\infty} \frac{1}{T} \mathbb{E}_\pi \left[\sum_{t=1}^{T} r_{k,t} \right], \tag{1}$$

where $r_{k,t} \sim r_k(a_t, s_t)$ is the reward, and we use the notation $\mathbb{E}_\pi[\cdot]$ to denote the expectation $\mathbb{E}_{s_0,a_0,s_1,\ldots}[\cdot]$ with $a_t \sim \pi(\cdot|s_t)$ and $s_t \sim P(\cdot|s_{t-1}, a_{t-1})$ under policy π.

In this paper, we consider the optimization of a class of α-fair utility functions, which are widely used in network optimization [6]. For some constant $\alpha \geq 0$, the α-fair utility is defined as:

$$U(x) = \begin{cases} x^{1-\alpha}/(1-\alpha) & \text{for } \alpha \neq 1, \\ \log(x) & \text{for } \alpha = 1. \end{cases} \tag{2}$$

Our goal is to find the optimal policy π that maximizes the aggregate utility $f = \sum_k U_k$ over the average per-step rewards $x_{\pi,k}$ of the K agents, i.e.,

$$\max_\pi \quad \sum_k U(x_{\pi,k}). \tag{3}$$

$$\text{s.t.} \quad x_{\pi,k} = \lim_{T\to\infty} \frac{1}{T} \mathbb{E}_\pi \left[\sum_{t=1}^{T} r_{k,t} \right] \tag{4}$$

It has been shown that the use of α-fair utility functions captures a wide range of notions of fairness including the well-known proportional fairness $f = \sum_k \log(x_k)$ for $\alpha = 1$, throughput maximization $f = \sum_k x_k$ for $\alpha = 0$, and max-min fairness $f = \min_k x_k$ for $\alpha \to \infty$. A unifying framework on fairness optimization is proposed in [6].

When it comes to online decision making using RL, however, maximizing general fairness functions is a challenging problem. RL algorithms typically are intended to maximize the discounted long-term reward of all agents - i.e., $\mathbb{E}_\pi[\sum_t \sum_k \gamma^t r_{k,t}]$ - and fail to consider fairness among different agents. It is easy to see that for non-linear utility functions such as the α-fair utility, maximizing the discounted long-term reward of all agents cannot optimize the network utility in Eq.(3).

IV. Learning Algorithm and Analysis

We propose a family of reinforcement learning algorithms that enable fairness utility optimization by introducing multiplicative-adjusted rewards to obtain an actor-critic implementation. We will first propose the RL algorithm concerning the adjusted rewards and characterize its policy gradient. Then, for proper choice of multiplicative-adjusted rewards, the derived policy is shown to be at least a stationary point for the network utility optimization under α-fair utility functions and inspires the design of an actor-critic algorithm. We assume that the Markov chain induced by any policy is irreducible and aperiodic.

A. Learning with Multiplicative-Adjusted Rewards

To optimize fairness utilities, it is easy to see that we need a mechanism to track past reward history in the learning algorithm. To demonstrate this, consider a simple example distributing a single unit of reward between two agents to achieve max-min fairness (which would assign an equal amount of reward to each agent at optimum). Under some policy π, if up to epoch t agent 1 receives more accumulative reward than agent 2, then we need to update policy π using a policy gradient that favors agent 2 over agent 1. Thus, the policy updates must depend on past reward history to optimize fairness utilities. To this end, we consider a new MDP, in which the rewards received by different agents are adjusted based on the shape of fairness utility and some statistics of past rewards.

Let $h_{\pi,t}$ be some statistics obtained from the current sample paths up to epoch t, under a policy π. We consider a uniformly-continuous function $\phi(h_{\pi,t})$ that depends on $h_{\pi,t}$. We define multiplicative-adjusted rewards for each time t by multiplying (actual) reward $r_{k,t}$ with a weight $\phi(h_{\pi,t})$, i.e.,

$$\hat{r}_{k,t} = r_{k,t} \cdot \phi(h_{\pi,t}). \tag{5}$$

Let $\hat{\rho}_\pi$ be the average per-step (multiplicative-adjusted) reward associated with policy π for this irreducible and aperiodic MDP [36]. With respect to the adjusted rewards $\hat{r}_{k,t}$, we denote the relative state- and action-value functions [37] associated with policy π by $\hat{V}_\pi : \mathcal{S} \to \mathbb{R}$ and $\hat{Q}_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which are defined respectively as:

$$\hat{V}_\pi(s) = \lim_{T\to\infty} \mathbb{E}_\pi \left[\sum_{t=1}^{T} \sum_{k=1}^{K} \hat{r}_{k,t} - \hat{\rho}_\pi \big|_{s_0=s} \right] \tag{6}$$

$$\hat{Q}_\pi(s,a) = \lim_{T\to\infty} \mathbb{E}_\pi \left[\sum_{t=1}^{T} \sum_{k=1}^{K} \hat{r}_{k,t} - \hat{\rho}_\pi \big|_{s_0=s,a_0=a} \right] \tag{7}$$

We note that $\hat{V}_\pi$ and $\hat{Q}_\pi$ are both bounded due to the subtraction of the average reward [37], [38]. Using the Laurent Series Expansion, it is easy to show that a state/action value in standard discounted reinforcement learning (with discount factor γ) can be decomposed into its average reward $\hat{\rho}_\pi$, the relative state/action value, and an additional error term that vanishes as $\gamma \to 1$ [38]. Further, we define the advantage function of policy π as the difference between $\hat{Q}_\pi$ and $\hat{V}_\pi$:

$$\hat{A}_\pi(s,a) = \hat{Q}_\pi(s,a) - \hat{V}_\pi(s). \tag{8}$$

We note that the multiplicative-adjusted rewards $\hat{r}_{k,t}$ depend on past statistics $h_{\pi,t}$ under policy π. Thus, the Markovian property that is required by the Policy Gradient Theorem (in its unrolling step) of standard RL is no longer satisfied with the adjusted rewards. The Policy Gradient Theorem cannot be directly applied to guarantee convergence in the actor-critic RL algorithm. To this end, we prove an alternative policy improvement theorem. Suppose the policy π_θ is approximated by a neural network with parameter θ. Let $\hat{\rho}_{\pi_\theta}$ be the average per-step adjusted reward under policy π_θ. We characterize the improvement in average per-step adjusted reward resulted by a small change in the current policy parameters by analyz-

ing an underlying MDP. It effectively establishes the a new policy gradient descent algorithm with multiplicative-adjusted rewards, which converges to at least a local maximum. We use $||\epsilon||_2$ to denote the L-2 norm of a vector ϵ..

Lemma 1. *By changing the current policy parameters from θ to $\theta+\epsilon$ for some small vector ϵ, the resulting change in the average reward is given by*

$$\hat{\rho}_{\pi_{\theta+\epsilon}} - \hat{\rho}_{\pi_\theta} = \mathbb{E}_\pi\left[\epsilon \hat{A}_\pi(s,a)\nabla_\theta \ln \pi_\theta(a|s)\right] + O(||\epsilon||_2^2). \quad (9)$$

Proof. Since multiplicative-adjusted rewards $\hat{r}_{k,t}$ depend on past statistics $h_{\pi,t}$, the Markovian property that is needed to analyze average reward $\hat{\rho}_{\pi_\theta}$ no longer holds. To deal with this, we define a new state $z_t = [s_t, h_{\pi,t}]$, $\forall t$ and show that an underlying chain with states z_t, action a_t, and rewards $\hat{r}_{k,t}$ is an MDP. This approach was first introduced to analyze partially observable MDPs [21]. To this end, it is easy to see that the multiplicative-adjusted rewards $\hat{r}_{k,t} = r_{k,t}\phi(h_{\pi,t})$ only depends on states z_t and actions a_t. Further, the transitions between the new states z_t are also Markovian, since $h_{\pi,t}$ depends only on past history $h_{\pi,t-1}$ and the state/action at time $t-1$.

We will prove the result in this lemma by analyzing this underlying MDP. We use $p^a_{zz'}$ to denote the state transition probability of this underlying MDP, and $V_\pi(z)$ and $Q_\pi(z,a)$ its state- and action-value functions, respectively. Let $P^\pi(z|s)$ define the limit occupancy probabilities over the underlying states z for each given s (since $h_{\pi,t}$ and s_t are not necessary independent). For the state- and action-value functions in Eq.(7) and Eq.(6), we have

$$\hat{Q}_\pi(s,a) = \sum_{z\in s} Q^\pi(z,a) \text{ and } \hat{V}_\pi(s) = \sum_{z\in s} V_\pi(z), \quad (10)$$

where $z \in s$ denotes the states z that are consistent with s.

Next, we consider an auxiliary function defined on the underlying MDP, i.e.,

$$G_{\theta+\epsilon,\theta+\epsilon,\theta} = \sum_a \pi_{\theta+\epsilon}(a|s)\sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s) A_{\pi_\theta}(z,a), \quad (11)$$

where $A_\pi(z,a) = Q_\pi(z,a) - V_\pi(z,a)$ is the advantage value for the underlying MDP, and we have used the notation that the policy parameters on the left hand side (in $G_{\theta+\epsilon,\theta+\epsilon,\theta}$) correspond to the policy parameters on the right, respectively. We show that

$$\begin{aligned}
&G_{\theta+\epsilon,\theta+\epsilon,\theta} - G_{\theta+\epsilon,\theta+\epsilon,\theta+\epsilon} \\
&= \sum_a \pi_{\theta+\epsilon}(a|s)\sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s)[Q_{\pi_\theta}(z,a) - Q_{\pi_{\theta+\epsilon}}(z,a)] \\
&\quad - \sum_a \pi_{\theta+\epsilon}(a|s)\sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s)[V_{\pi_\theta}(z) - V_{\pi_{\theta+\epsilon}}(z)] \\
&= \sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s)\sum_{a,z'} \pi_{\theta+\epsilon}(a|s) p^a_{zz'}[V_{\pi_\theta}(z') - V_{\pi_{\theta+\epsilon}}(z')] \\
&\quad + \hat{\rho}_{\pi_{\theta+\epsilon}} - \hat{\rho}_{\pi_\theta} - \sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s)[V_{\pi_\theta}(z) - V_{\pi_{\theta+\epsilon}}(z)] \\
&= \sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s)\sum_{z'} p^{\pi_{\theta+\epsilon}}_{zz'}[V_{\pi_\theta}(z') - V_{\pi_{\theta+\epsilon}}(z')] \\
&\quad + \hat{\rho}_{\pi_{\theta+\epsilon}} - \hat{\rho}_{\pi_\theta} - \sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s)[V_{\pi_\theta}(z) - V_{\pi_{\theta+\epsilon}}(z)] \quad (12)
\end{aligned}$$

where we used $A_\pi(z,a) = Q_\pi(z,a) - V_\pi(z,a)$ in step 1 above, $\sum_a \pi_{\theta+\epsilon}(a|s)p^a_{zz'} = p^{\pi_{\theta+\epsilon}}_{zz'}$ for the state transition probabilities in step 3, the fact that $\sum_a \pi_{\theta+\epsilon}(a|s) = 1$ for the last term in step 2, and the Bellman's equation of the underlying MDP in step 2, i.e.,

$$Q_\pi(z,a) = \hat{r}(s,a) - \hat{\rho}_\pi + \sum_{z'} p^a_{zz'} V_\pi(z'). \quad (13)$$

We note that by weighting both sides of Eq.(12) for each state s with the limit occupancy probabilities $P^{\pi_{\theta+\epsilon}}(s)$ and by summing over all states, the first and the third term in the last step of Eq.(12) become exactly the same, thus canceling out. Furthermore, it is easy to verify that $G_{\theta+\epsilon,\theta+\epsilon,\theta+\epsilon} = 0$ since $\sum_a \pi(a|s)Q_\pi(z,a) = V_\pi(z)$ for any policy π. Thus, by re-arranging the terms and plugging in these results, Eq.(12) implies

$$\hat{\rho}_{\pi_{\theta+\epsilon}} - \hat{\rho}_{\pi_\theta} = \sum_s P^{\pi_{\theta+\epsilon}}(s) G_{\theta+\epsilon,\theta+\epsilon,\theta}. \quad (14)$$

This does not yet allow us to derive the change in average reward because $G_{\theta+\epsilon,\theta+\epsilon,\theta}$ still contains terms that are only available in the underlying MDP. However, when the change ϵ in the policy parameters is small, it can be shown [21] that the change in $P^{\pi_\theta}(z|s)$ and $P^{\pi_\theta}(s)$ can both be bounded by $C\cdot||\epsilon||_2$ for some constant C. Similarly, the corresponding change in the policy π_θ can be bounded by $\epsilon\nabla\pi_\theta(a|s) + O(||\epsilon||_2^2)$. Using these results on the right hand side of Eq.(14), we have

$$\begin{aligned}
&\hat{\rho}_{\pi_{\theta+\epsilon}} - \hat{\rho}_{\pi_\theta} \\
&= \sum_s P^{\pi_{\theta+\epsilon}}(s)\sum_a [\pi_\theta(a|s) + \epsilon\nabla_\theta\pi_\theta(a|s) + O(||\epsilon||_2^2)] \\
&\quad \cdot \sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s) A_{\pi_\theta}(z,a) \\
&= \sum_{s,a} P^{\pi_{\theta+\epsilon}}(s)\epsilon\nabla_\theta\pi_\theta(z|s)\sum_{z\in s} P^{\pi_{\theta+\epsilon}}(z|s) A_{\pi_\theta}(z,a) \\
&\quad + O(||\epsilon||_2^2) \\
&= \sum_{s,a} P^{\pi_\theta}(s)\epsilon\nabla_\theta\pi_\theta(a|s)\hat{A}_{\pi_\theta}(s,a) + O(||\epsilon||_2^2) \\
&= \sum_{s,a} P^{\pi_\theta}(s)\pi_\theta(a|s)\epsilon\nabla_\theta \ln\pi_\theta(a|s)\hat{A}_{\pi_\theta}(s,a) \quad (15)
\end{aligned}$$

where the second step uses $\sum_a \pi(a|s)Q_\pi(z,a) = V_\pi(z)$, or equivalently $\sum_a \pi(a|s)A_\pi(z,a) = 0$, the third step uses the bound on $P^{\pi_\theta}(z|s)$ and $P^{\pi_\theta}(s)$ for a small change in policy parameters. We note that the last step is exactly Eq.(9) since $\sum_{s,a} P^{\pi_\theta}(s)\pi_\theta(a|s)[\cdot]$ gives the desired expectation. □

The lemma effectively establishes a policy gradient for optimizing average per-step adjusted reward $\hat{\rho}_{\pi_theta}$. We can leverage it to develop a policy gradient algorithm, whose convergence follows directly from the standard analysis of RL with proper estimates of advantage value $\hat{A}_\pi(s,a)$ [20] and is summarized in the lemma below. The algorithm continues to improve average per-step adjusted reward until $\nabla_\theta \hat{\rho}_{\pi_theta} = 0$, which constitutes a stationary point of the policy update.

Lemma 2. *Policy gradient algorithm using $\hat{A}_\pi(s,a)\nabla_\theta \ln \pi_\theta(a|s)$ and sufficiently small learning rate η converges to a stationary point with respect to the multiplicative-adjusted rewards.*

B. Solving Fairness Utility Optimization

Now we show that for proper choice of statistics $h_{\pi,t}$ and function $\phi(\cdot)$ that define the adjusted reward, the gradient policy algorithm indeed converges to a stationary point of the α-fair utility optimization problem in Eq.(2). Lemma 2 shows that the policy gradient algorithm converges to a stationary point of the adjusted MDP (with adjusted rewards). We denote the optimal policy parameter as θ^*, and thus optimal policy as π_{θ^*}. Then, the average per-step reward in terms of the original rewards $r_{k,t}$ (as defined in our α-fair utility optimization problem) is given by

$$x_{\pi_{\theta^*},k} = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}_\pi \left[\sum_{t=1}^{T} r_{k,t}\right]. \tag{16}$$

The above limit exists because the Markov Chain is irreducible and aperiodic (and thus ergodic) under policy π_{θ^*} [36]. We need to show that θ^* (and the optimal policy π_{θ^*}) is also a stationary point for the optimization of $\sum_k U(x_{\pi_\theta,k})$.

Algorithm 1 Fairness Policy Gradient Algorithm

1: Initialize: Learning rate η, policy parameter θ.
2: **for** *each episode* **do**
3: Generate one trajectory on policy π_θ:
$s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_T$.
4: Adjust reward $\hat{r}_{k,t}, \forall k,t$ using Eq.(5)
5: **for** $t = 1, 2, \ldots, T$ **do**
6: Estimate advantage $\hat{A}_\pi(s_t, a_t)$ using Eq.(8).
7: Update $\theta \leftarrow \theta + \eta\gamma^t \hat{A}_\pi(s_t,a_t)\nabla_\theta \ln \pi_\theta(a_t|s_t)$.
8: **end for**
9: **end for**
10: Output: Policy π_θ

We choose $\phi(\cdot)$ to be the first order derivative of the fairness utility function, i.e., $U'(\cdot)$. It is easy to see that the function is bounded and Lipschitz, as long as the per-step rewards $x_{\pi_{\theta^*},k}$ are bounded away from 0 and also upper-bounded, i.e., $|U'(x) - U'(y)| \le L|x-y|$ for some constant $L > 0$. Next, we consider an estimate of the per-step reward (in terms of the original rewards $r_{k,t}$) using the history up to time t, i.e., $h_{\pi_{\theta^*},t} = \sum_{\tau=1}^{t} r_{k,\tau}/t$. The adjusted rewards are then defined by

$$\hat{r}_{k,t} = r_{k,t}\phi(h_{\pi_{\theta^*},t}) = r_{k,t}U'\left(\frac{1}{t}\sum_{\tau=1}^{t} r_{k,\tau}\right). \tag{17}$$

We show that with this choice of adjusted rewards, the gradient policy algorithm converges to a stationary point of the α-fair utility optimization problem.

Theorem 1. *For the adjusted reward in Eq.(17), the gradient policy algorithm converges to a stationary point of the α-fair utility optimization.*

Proof. We need to prove that the policy parameter θ^* (and thus policy π_{θ^*}) is a stationary point for the α-fair utility $\sum_k U(x_{\pi_\theta,k})$. To this end, we note that due to Lemma 2, θ^* is a stationary point for the adjusted MDP, which implies that $\nabla_\theta \hat{\rho}_{\pi_\theta}|_{\theta=\theta^*} = 0$ due to Lemma 1. We need to analyze the relationship between per-step adjusted reward $\hat{\rho}_{\pi_\theta}$ and per-step reward $x_{\pi_\theta,k}$.

Using the adjusted rewards in Eq.(17), we have

$$\hat{\rho}_{\pi_\theta} = \lim_{T\to\infty} \frac{1}{T} \cdot \pi_\theta \left[\sum_{t=1}^{T}\sum_{k=1}^{K} r_{k,t}U'\left(\frac{1}{t}\sum_{\tau=1}^{t} r_{k,\tau}\right)\right] \tag{18}$$

where the expectation π_θ denotes $\mathbb{E}_{s_0,a_0,s_1,\ldots}[\cdot]$ with $a_t \sim \pi(\cdot|s_t)$ and $s_t \sim P(\cdot|s_{t-1}, a_{t-1})$ under policy π_θ. Since the Markov Chain is irreducible and aperiodic (and thus ergodic) under policy π_θ, we have $x_{\pi_\theta,k} = \lim_{T\to\infty} 1/T \sum_{t=1}^{T} r_{k,t}$. It implies that for any $\epsilon > 0$, there exists a sufficient large T, such that $|1/T\sum_{t=1}^{T} r_{k,t} - x_{\pi_\theta,k}| < \epsilon$. Combining this and the Lipschitz continuity of $U'(\cdot)$, we have

$$\begin{aligned}
&\left|\frac{1}{T}\sum_{t=1}^{T}\sum_k r_{k,t}U'\left(\frac{1}{t}\sum_{\tau=1}^{t} r_{k,\tau}\right) - \sum_k x_{\pi_\theta,k}U'(x_{\pi_\theta,k})\right| \\
&\le \sum_k \left|\frac{1}{T}\sum_{t=1}^{T} r_{k,t}U'\left(\frac{1}{t}\sum_{\tau=1}^{t} r_{k,\tau}\right) - x_{\pi_\theta,k}U'(x_{\pi_\theta,k})\right| \\
&\le \sum_k \left|\frac{1}{T}\sum_{t=1}^{T} r_{k,t}U'\left(x_{\pi_\theta,k}\right) - x_{\pi_\theta,k}U'(x_{\pi_\theta,k})\right| + \epsilon C_2 L \\
&\le \sum_k \left|\frac{1}{T}\sum_{t=1}^{T} r_{k,t} - x_{\pi_\theta,k}\right| \cdot \left|U'\left(x_{\pi_\theta,k}\right)\right| + \epsilon C_2 L \\
&\le \epsilon C_1 + \epsilon C_2 L
\end{aligned} \tag{19}$$

where C_1 is a bound for $|U'(x_{\pi_\theta,k})|$ and C_2 is a bound for the average reward $1/T\sum_{t=1}^{T} r_{k,t}$. We used bound C_2 in the second step above, as well as the Lipschitz continuity, i.e., $|U'(1/t\sum_{\tau=1}^{t} r_{k,\tau}) - U'(x_{\pi_\theta,k})| \le L|1/t\sum_{\tau=1}^{t} r_{k,\tau} - x_{\pi_\theta,k}| \le L\epsilon$.

As T grows to infinity, Eq.(18) and Eq.(19) imply that $\hat{\rho}_{\pi_\theta} = \sum_k x_{\pi_\theta,k}U'(x_{\pi_\theta,k})$. It is easy to see that $(1-\alpha)U(x) = xU'(x)$ for α-fair utility functions[1].

[1]For $\alpha = 1$, optimization of the proportional fair utility $\log(x)$ can be considered as the limit of $(x^{1-\alpha} - 1)/(1-\alpha)$ as $\alpha \to 1$.

Thus, θ^* being a stationary point in the underlying MDP (i.e., $\nabla_\theta \left[x_{\pi_\theta,k} U'(x_{\pi_\theta,k})\right]|_{\theta=\theta^*} = 0$) implies that $\nabla_\theta \left[\sum_k U(x_{\pi_\theta,k})\right]|_{\theta=\theta^*} = 0$. We conclude that θ^* is also a stationary point for the α-fair utility optimization. □

The result inspires a new actor-critic algorithm to solve the α-fair utility optimization. In particular, to compute the adjusted rewards $\hat{r}_{k,t}$ in Eq.(17), we can leverage an estimate of the average of original rewards $r_{k,t}$ on the existing trajectory up to epoch t (instead of the true average under the current policy). Let $\hat{V}_\omega(s_t)$ be a neural network approximate of the state-value function parameterized by ω. We can train $\hat{V}_\omega(s_t)$ using the standard Temporal Different (TD) error with respect to adjusted rewards, i.e., $\delta_t = \sum_k \hat{r}_{k,t} + \gamma \hat{V}_\omega(s_{t+1}) - \hat{V}_\omega(s_t)$, as in standard actor-critic algorithms. By calculating the average of past original rewards, the algorithm obtains an estimate of the adjustment weights ϕ and updates the state-value as well as the policy parameters based on this estimate. Our proposed Fairness Actor-Critic (FAC) algorithm is summarized in Algorithm 2.

Algorithm 2 Fairness Actor-Critic (FAC) Algorithm

1: Initialize: Learning rate η_θ, η_ω, policy parameter θ, ω.
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Sample action $a_t \sim \pi_\theta(\cdot|s_t)$.
4: Sample reward $r_t \sim r_k(s_t, a_t)$.
5: Generate next state $s_{t+1} \sim P(\cdot|s_t, a_t)$.
6: Adjust reward $\hat{r}_{k,t}, \forall k$ using Eq.(17).
7: Compute $\hat{A}(a_t, s_t) = \sum_k \hat{r}_{k,t} + \gamma \hat{V}_\omega(s_{t+1}) - \hat{V}_\omega(s_t)$.
8: Update $\theta \leftarrow \theta + \eta_\theta \hat{A}_\pi(s_t, a_t) \nabla_\theta \ln \pi_\theta(a_t|s_t)$.
9: Compute TD error: $\delta_t = \hat{A}(a_t, s_t)$.
10: Update: $\omega \leftarrow \omega + \eta_\omega \delta_t \nabla_\omega \hat{V}_\omega(s_t)$.
11: **end for**
12: Output: Policy π_θ.

Remark 1: We note that when actor-critic algorithms are concerned, due to the error of policy evaluation steps and its impact on policy improvement steps, the convergence has only been established in special cases, e.g., if linear action-value function approximators or two-layer neural-network parameterization are used [39]. The convergence proof in general remains an open problem. Existing convergence results apply to our proposed fairness actor-critic algorithm

V. Evaluations

We implement the proposed Fairness Actor-Critic (FAC) algorithm and evaluate its performance in two real-world network optimization problems that require online decision making, i.e., wireless network scheduling and QoE optimization in video streaming. We show that the FAC algorithm can achieve the optimal policy when the problem is convex (e.g., in wireless network scheduling) and significantly outperform many heuristic and learning algorithms when the problem is non-convex (e.g., in QoE optimization).

A. Wireless Network Scheduling

Problem formulation. Fairness among different users' transmission rates is a very important design objective for opportunistic schedulers in wireless networks. This is often achieved by maximizing various α-fair utility functions [40], [41], [42], [43]. Consider a wireless network with a single base station and K users, , similar to [19]. Suppose that the wireless channel of each agent could only be in one of two states: $\{good, bad\}$. When its channel is in $good$ state, the agent has a higher transmission rate than in bad state. An example of data rates received by $K = 4$ agents (denoted by $r_i(t)$ for the agent i) in different channel states is shown in Table 1. We consider a dynamic channel model, where the state transition of each agent's channel is independent, and at each time step, the probability of remaining in the same state is $p_0 = 0.8$ while the probability of moving into a random state with equal probability is $p_1 = 0.2$. Let $s_i(t)$ denote the the channel state of agent i at time t. The state transitions are described by

$$s_i(t+1) = \begin{cases} s_i(t) & w.p.\ 0.8 \\ s \sim \text{Uniform}(\{good, bad\}) & w.p.\ 0.2 \end{cases} \quad (20)$$

User state	$r_1(t)$	$r_2(t)$	$r_3(t)$	$r_4(t)$
good	1.50	2.25	1.25	1.50
bad	0.768	1.00	0.384	1.12

Table I: Transmission rates of K=4 agents (obtained from a practical LTE network, in Mbps) under different channel states in our evaluation.

In our opportunistic scheduler, a single agent is selected to transmit at each time slot T. The problem of maximizing an α-fair utility over agent selection strategies can be formally defined as

$$\max \sum_{i=1}^{K} U(x_i), \quad \text{s.t. } x_i = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} r_i(t) \quad (21)$$

where $r_i(t)$ is the transmission rate of agent i at time t, x_i the average transmission rate of agent i, and U the utility function defined in Eq.(2) with some choice of parameter α corresponding to certain notion of fairness.

To recast this into a RL problem, we define the state space as $S(t) = \{s_1(t), \ldots, s_K(t)\}$, where $s_i(t)$ is the channel state of agent i at time t. Thus, there are $|S| = 2^K$ possible states for the learning problem. Since our opportunistic scheduler selects one agent to transmit at each t, the action at each time step t is a one-hot vector with one entry set at 1, corresponding to the agent selected for transmission. The action space A has size $|A| = K$. Finally, the reward corresponding to each state and action is the transmission rate of the active agent at time t, while remain agents receive zero rewards.

Evaluation results. We evaluate our FAC algorithm along with two learning algorithm baselines, which are denoted by "SARSA" and "FEN" policies. We also compute the optimal solution "OPT", which leverages convex optimization (which yet has exponential number of variables and constraints in general) to provide ground truth for our evaluation.

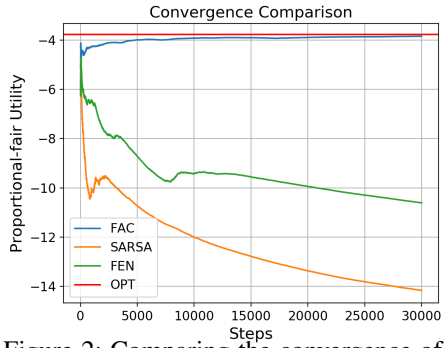

Figure 2: Comparing the convergence of different algorithms for proportional-fair utility $\sum_i \log(x_i)$. FAC converges to the optimal utility value.

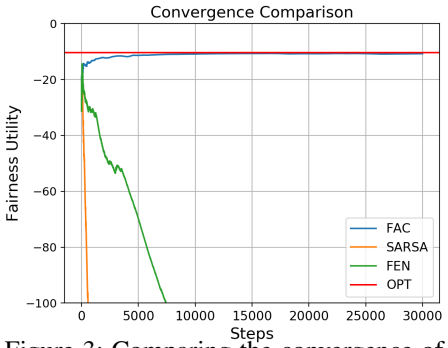

Figure 3: Comparing the convergence of different algorithms for fairness utility $\sum_i -1/x_i$ with $\alpha = 2$. FAC converges to the optimal utility value.

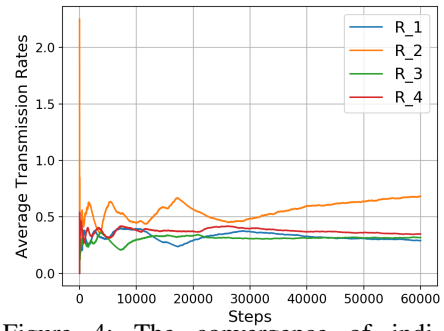

Figure 4: The convergence of individual (average) transmission rates under the proposed FAC algorithm for proportional-fair utility.

- Algorithm "FAC": This is the Fairness Actor-Critic algorithm we proposed in this paper. It maximizes the fairness utility concerning adjusted rewards over a finite horizon.
- Algorithm "OPT": This is the optimal utility that is computed through a convex optimization to provide the ground truth for our evaluation (shown as a horizontal line). More precisely, We denote $y_{i,s}$ as the probability of selecting agent i for transmission in state s, satisfying $\sum_s y_{i,s} = 1$. It is not hard to see that optimization is convex, albeit having $K \cdot 2^K$ optimization variables and 2^K constraints in general.
- Algorithm "SARSA": We implement the SARSA RL algorithm to optimize the long-term reward as a baseline without considering fairness. It uses a standard Policy Gradient method with $\sum_i r_i(t)$ as the aggregate step reward and a single Actor neural network for learning.
- Algorithm "FEN": This implements the RL algorithm in [18] for fair-efficient reward optimization. In particular, it leverages a hueristic method to construct an adjusted reward $\hat{r}_i(t) = \bar{R}(t)/[c(\epsilon + |\bar{R}_i(t)/\bar{R}(t) - 1|)]$, where $\bar{R}_i(t)$ is the average reward of agent i over elapsed timesteps and $\bar{R}(t)$ the average sum reward of all agents. The term $\bar{R}(t)/c$ could be seen as the resource utilization of the whole system, while $|\bar{R}_i(t)/\bar{R}(t) - 1|$ measures an agent's utility deviation from the average.

In our experiments, the actor and critic neural networks consist of one hidden layer with 250 neurons and the ReLu activation function. We use Adam optimizer to train the network for 30,000 steps for the learning algorithms. We choose the learning rate of the actor-network as 0.001, while the critic network has a learning rate of 0.01. This small difference makes the critic network learn faster than the actor-network. A slower learning rate of the actor could allow it to obtain feedback from the critic in each step, making the learning process more robust. Finally, we use $\gamma = 0.9$ in the critic network to discount the reward and calculate the advantage functions.

We first compare FAC with the baseline algorithms using proportional-fair utility function $\sum_i \log(x_i)$, which is a special case of the α-fair utility for $\alpha = 1$. We trained SARSA, FEN, and FAC algorithms for 30,000 steps and $K = 4$ agents and plot the convergence of the proportional fair utility value in Figure 2. We note that the SARSA algorithm has the worst performance because it aims to optimize the long-term (discounted) reward $\sum_i x_i$ (i.e., the overall throughput) without taking fairness into account. The proportional-fair utility value eventually goes to minus infinity for SARSA, since some agents would receive a zero transmission rate in this throughput optimal solution. The performance of FEN is slightly better than SARSA. This is because FEN leverages a heuristic design to balance the throughput efficiency and the variance of different agents' transmission rates. More precisely, in the adjusted reward of FEN algorithm, the term $\bar{R}(t)/c$ could be seen as the resource utilization of the whole system, encouraging agents to become more efficient, while $|\bar{R}_i(t)/\bar{R}(t) - 1|$ measures an agent's utility deviation from the average, punishing any agent if its average reward moves above or below the average. Our proposed FAC algorithm can converge within 10,000 steps and is shown to achieve the optimal proportional-fair utility value (as obtained by OPT despite having an exponential number of variables and constraints in general) in this convex wireless network scheduling problem.

Next, to demonstrate FAC algorithm's ability to optimize different utility functions, we consider the α-fair utility with $\alpha = -1$, i.e, $\sum_i -1/x_i$. Since x_i is the average transmission rate of agent i, maximizing the utility $\max \sum_i -1/x_i$ can be regarded as minimizing the aggregate latency of all agents i.e., $\min \sum_i 1/x_i$ assuming a single unit of data to transmit. The convergence of different algorithms is shown in Figure 3. The α-fair utility value for SARSA and FEN drops more quickly, while FAC again can converge to the optimal solution.

Finally, Figure 4 shows the convergences of individual agents' average transmission rates $\bar{R}_i(t)$ in the FAC algorithm for the proportional fair utility function. We note that the adjusted reward in FAC is given by $r_i(t)/\bar{R}_i(t)$, where $\bar{R}_i(t)$ is the average reward of agent i up to current time t. Thus, FAC is encouraged to choose an agent with a smaller $\bar{R}_i(t)$ to transmit. After that, the agent's adjusted reward would rise, until it becomes more favorable to select other agents. As we have proven in this paper, the FAC algorithm converges to an

optimal policy of the α-fair utility optimization for arbitrary α.

B. QoE Optimization in Video Streaming

Problem Formulation. We now consider a QoE optimization problem in video streaming. The streaming stall time, which is defined as the accumulated waiting time between a video chunk is played and the time the next chunk is downloaded is used as our metric for QoE [44], [45]. We note that the optimization of stall time over dynamic bandwidth allocation among multiple agents leads to a non-convex online problem that is very difficult to solve.

Consider a video streaming server connected with a set of K agents, who continuously request various video clips on the server through a shared link with bandwidth B. Let $t_k(v)$ denote the stall time ratio (i.e., stall time normalized by video play time) experienced by user k when watching video v. We consider the optimization of bandwidth allocation among the K agents – denoted by $\pi = (\pi_1, \ldots, \pi_K)$ (such that $\pi_k B$ is the bandwidth assigned to agent k) – with QoE rewards defined by $f(t_k(v)) = 2 - t_k(v)$. Thus a smaller stall time ratio correspond to a higher reward.

Our goal is to optimize a proportional-fair utility of the per-step average QoE rewards over bandwidth allocation $\pi = (\pi_1, \ldots, \pi_K)$, i.e.,

$$\max \sum_{k=1}^{K} log(x_k), \quad \text{s.t. } x_k = \lim_{T\to\infty} \frac{1}{T} \sum_{l=1}^{T} r_k(l) \tag{22}$$

where $r_k(l) = f(t_k(v)) = 2 - t_k(v)$ is the QoE value received by agent k at time slot l (over video v). The choice of proportional-fair utility(and QoE reward) ensures that (i) Videos with lower stall time are given higher rewards, while all agents' stall time are bounded by 2; and (ii) Reducing the stall time for heavily stalled users achieves higher reward improvement than for lightly stalled users due to the shape of logarithm function.

We implement a network with K=5 agents and both high- and low-resolution videos. Three agents prefer to watch high-resolution videos with bitrates of 8Mbps (1080p) and 5Mbps (720p) (similar to Youtube videos), while the other two agents consume 2.5Mbps (480p) and 1Mbps (360p) videos randomly. The exact setup of different agents' video bitrates and probabilities are shown in Table II. We divide the timeline into slots according to the agents' video switches, i.e, each time slot ends when an agent switches its video/bitrate selection, and a new time slot starts right after. Similar to [45], we assume that the duration of time slots follow an i.i.d. exponential distribution and that agents independently select the next video according to the probabilities specified in Table II. When a new time slot l is detected, a new bandwidth allocation decision π is generated to allocate the shared bandwidth of B to agents.

To recast this into a learning problem, we define the system state at time slot l by $s = (\overline{v}(l), \overline{d}(l), \overline{z}(l), \overline{c}(l))$, in which the four K-dimension vectors represent the video bitrates ($\overline{v}(l)$), assigned download speeds ($\overline{d}(l)$), accumulated stall time for current videos ($\overline{z}(l)$), and the number of residue video chunks for all users ($\overline{c}(l)$). To adjust bandwidth allocation decisions on the fly while maintaining a small action space, we introduce two reinforcement learning modules, one of which selects an agent with a "down" action (a_-) to reduce its assigned bandwidth by one unit, and the other selects an agent with an "up" action (a_+) to increase its assigned bandwidth by one unit. Thus the total bandwidth consumption remains equal to B. By iteratively running these two RL modules until $a_- = a_+$ is detected, a new bandwidth allocation policy π can be obtained. The action space for each RL module is kept at $|A| = K$, i.e., linear to the number of agents.

Table II: User preferences in our experiment.

User	Resolutions	Bitrates	Probabilities
1	1080p	$8Mbps$	0.5
	720p	$5Mbps$	0.5
2	1080p	$8Mbps$	0.5
	720p	$5Mbps$	0.5
3	1080p	$8Mbps$	0.5
	720p	$5Mbps$	0.5
4	480p	$2.5Mbps$	0.5
	360p	$1Mbps$	0.5
5	480p	$2.5Mbps$	0.5
	360p	$1Mbps$	0.5

Evaluation Results. We develop a simulation testbed similar to [44], [45] to generate the stall time in video streaming. The following algorithms, including both heuristic and learning methods, are compared:

- "FAC": The proposed FAC algorithm with adjusted rewards for optimizing proportional fair utilities.
- "Even": A heuristic policy that evenly distributes the shared bandwidth to all agents.
- "Adaptive": A heuristic policy that dynamically splits the bandwidth in proportion to agents' video bitrates.
- "NMPG": The NMPG reinforcement learning algorithm in [19] for optimizing fairness utilities. It leverages the Monte Carlo method for policy gradient evaluation and thus has slow convergence.

First, we show the convergence of total QoE reward on 2500KB/S download link in Figure 5. Since NMPG uses the Monte Carlo method to evaluate policy gradient in each episode, it takes 10,000 steps per episode for each Monte Carlo evaluation and 5,000 episodes of gradient updates – thus 50 million steps in total – to converge to an optimal policy. On the other hand, the FAC algorithm that leverages an actor-critic structure converges in only 6,000 steps. Further, because FAC adjusts the bandwidth distribution after each step, its achieved utility quicks ramp up, while NMPG only makes incremental utility improvement after each episode. The convergence of FAC is plotted almost as a vertical line(with barely visible transient to optimal) in Figure 5 and quickly arrives at the optimal utility value. Several orders of magnitude improvement in convergence speed are observed in this video streaming application.

Figures 6 and 7 show the performance comparison of different algorithms with a Constant Bitrate (CBR) streaming policy, for 2000 KB/S and 2500 KB/S download links, respectively. In particular, we run the NMPG algorithm with a training time

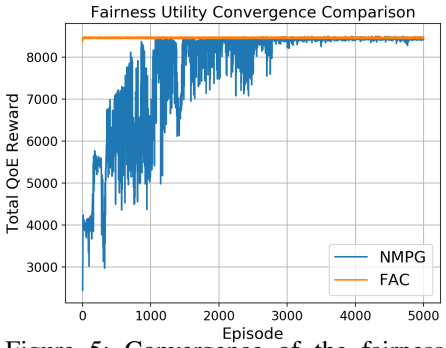

Figure 5: Convergence of the fairness utility using FAC and NMPG algorithms. FAC converges much faster(with barely visible transient to optimal) than that of NMPG.

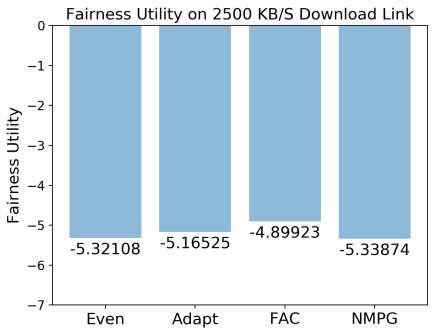

Figure 6: FAC achieves the highest fairness utility for download bandwidth of 2500 KB/s. Note that y-axis is in log due to the use of proportional fairness.

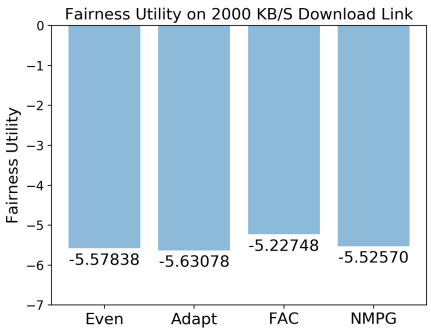

Figure 7: The benefit of FAC becomes more significant for download bandwidth of 2000 KB/s. Note that y-axis is in log due to the use of proportional fairness.

Table III: Reward breakdown for fairness function on $2500KB/s$ download link.

Policy	Total Reward	Agent Average		
		Agent	Stall Ratio	Reward
"Even"	5188.13	1, 2, 3 (HD) 4, 5 (LD)	0.3959 0.0761	1.6041 1.9239
"Adaptive"	5343.36	1, 2, 3 (HD) 4, 5 (LD)	0.1789 0.2808	1.8210 1.7191
"FAC"	5632.58	1, 2, 3 (HD) 4, 5 (LD)	0.2257 0.2672	1.8943 1.8528
"NMPG"	5161.22	1, 2, 3 (HD) 4, 5 (LD)	0.3396 0.2409	1.6604 1.7591

limit of 1000 episodes and 3000 steps per episode (i.e., 3 million steps in total). Our proposed FAC algorithm performs better than all other baselines in terms of achieved fairness objective $\sum_{k=1}^{K} log(R(t_k(v)))$. On the 2000 KB/S download link, FAC outperforms the static "Even", dynamic "Adaptive" and NMPG algorithms by 6.29%, 7.16%, and 5.40% in terms of the fairness utility value, respectively. On the 2500 KB/S download link, FAC outperforms the static "Even", dynamic "Adaptive" and NMPG policies by 7.93%, 5.15%, and 8.23% respectively. We note that these improvements in the fairness utility should be interpreted in the "decibel" sense due to the use of the logarithm utility function. For instance, as the fairness utility improves from -5.63(achieved by "Adaptive") to -5.22(achieved by "FAC"), the 7.16% improvement in utility corresponds to $10^{(-5.22+5.63)/5} - 1 = 21\%$ improvement in the geometric-mean reward.

Table III shows the breakdown of stall time $t_k(v)$ and the original reward without normalization for different algorithms on 2500KB/S download link. Again, we run the NMPG algorithm with a training time limit of 1000 episodes and 3000 steps per episode. We note that the "Adaptive" policy achieves similar stall time for both HD and LD users, while the "Even" policy tends to slightly reduce the stall time of LD users at the cost of the performance loss of HD users. Since the reward loss of HD users is lower than the gain of LD users, the "Even" policy could have a higher overall reward than "Adaptive". We also note that due to the use of the Monte Carlo method, the NMPG algorithm has very slow convergence and suffers much lower utility if the training time budget further decreases. FAC can achieve the highest performance compared to all three baselines since it dynamically makes bandwidth adjustment decisions based on the current network states and with respect to the fairness utility objective. When a user in FAC has enough chunks loaded ready for future playing, its bandwidth can be temporarily hand over to another user who is just switching to a new video or is already waiting too long for its current video. The stall time of both HD and LD users is optimized jointly concerning the proportional fair utility objective.

VI. Conclusions

For optimizing general fairness utility functions, we propose a family of algorithms that bring fairness to actor-critic reinforcement learning. By proving an alternative policy gradient theorem, it is shown that RL with adjusted rewards (obtained through a uniformly-continuous function depending on the shape of fairness utility and some statistics of past rewards) can converge to at least a stationary point of general α-fairness utility optimization. Our analysis inspires the design of an actor-critic algorithm for α-fair utility optimization. We implement and evaluate the proposed algorithm on two real-world applications, i.e., wireless scheduling and video QoE optimization. It is shown to substantially outperform existing heuristic and learning algorithms in terms of the optimal fairness utility value. For future work, we plan to extend our framework to consider off-policy training and to investigate fully distributed multi-agent algorithms.

REFERENCES

[1] C. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks," ser. SenSys '04. ACM, 2004, pp. 148–161.

[2] E. Hahne, "Round-robin scheduling for max-min fairness in data networks," *IEEE journal on selected areas in communications*, vol. 9, no. 7, pp. 1024–1039, 1991.

[3] A. Sinha and A. Anastasopoulos, "A general mechanism design methodology for social utility maximization with linear constraints," 2015.

[4] B. Radunovic and J.-Y. Le Boudec, "A unified framework for max-min and min-max fairness with applications," *IEEE/ACM transactions on networking*, vol. 15, no. 5, pp. 1073–1083, 2007.

[5] F. Kelly, "Charging and rate control for elastic traffic," *European transactions on telecommunications*, vol. 8, no. 1, pp. 33–37, 1997.

[6] T. Lan, D. T. H. Kao, M. Chiang, and A. Sabharwal, "An axiomatic theory of fairness," *CoRR*, vol. abs/0906.0557, 2009. [Online]. Available: http://arxiv.org/abs/0906.0557

[7] C. Joe-Wong, S. Sen, T. Lan, and M. Chiang, "Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework," Jun. 2012.

[8] T. Lan and M. Chiang, "Measuring fairness: Theory and applications," *In proceedings of Allerton Conference on Communication, Control and Computing (Allerton)*, 2011.

[9] S. de Jong, K. Tuyls, K. Verbeeck, and N. Roos, "Considerations for fairness in multi-agent systems," 2007.

[10] C. Zhang and J. A. Shah, "Fairness in multi-agent sequential decision-making," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS 14. Cambridge, MA, USA: MIT Press, 2014, pp. 2636–2644.

[11] A. Beynier, N. Maudet, and A. Damamme, "Fairness in Multiagent Resource Allocation with Dynamic and Partial Observations: Extended Abstract," in *the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018)*, Stockholm, Sweden, Jul. 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01808984

[12] G. Alnwaimi, S. Vahid, and K. Moessner, "Dynamic heterogeneous learning games for opportunistic access in lte-based macro/femtocell deployments," *IEEE Transactions on Wireless Communications*, vol. 14, no. 4, pp. 2294–2308, 2015.

[13] O. Onireti, A. Zoha, J. Moysen, A. Imran, L. Giupponi, M. Ali Imran, and A. Abu-Dayya, "A cell outage management framework for dense heterogeneous networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2097–2113, 2016.

[14] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1871–1879.

[15] A. Valadarsky, M. Schapira, D. Shahaf, and A. Tamar, "Learning to route," in *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XVI. New York, NY, USA: Association for Computing Machinery, 2017, pp. 185–191. [Online]. Available: https://doi.org/10.1145/3152434.3152441

[16] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM 17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 197–210. [Online]. Available: https://doi.org/10.1145/3098822.3098843

[17] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1946–1960, 2017.

[18] J. Jiang and Z. Lu, "Learning fairness in multi-agent systems," 2019.

[19] M. Agarwal and V. Aggarwal, "Reinforcement learning with non-markovian rewards," 2019.

[20] D. D. Castro and R. Meir, "A convergent online single time scale actor critic algorithm," *CoRR*, vol. abs/0909.2934, 2009.

[21] T. Jaakkola, S. P. Singh, and M. I. Jordan, "Reinforcement learning algorithm for partially observable markov decision problems," in *in Proceedings of the 7th International Conference on Neural Information Processing Systems*, 1994.

[22] M. A. Marsan and M. Gerla, "Fairness in local computing networks," 1982.

[23] J. Wong, J. Sauve, and J. Field, "A study of fairness in packet-switching networks," *IEEE Transactions on Communications*, vol. 30, no. 2, pp. 346–353, 1982.

[24] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," 1998.

[25] M. Dianati, X. Shen, and S. Naik, "A new fairness index for radio resource allocation in wireless networks," vol. 2. IEEE, 2005, pp. 712–717 Vol. 2.

[26] C. Koksal, H. Kassab, and H. Balakrishnan, "An analysis of short-term fairness in wireless media access protocols," *ACM SIGMETRICS*, 03 2001.

[27] M. Bredel and M. Fidler, "Understanding fairness and its impact on quality of service in ieee 802.11," in *IEEE INFOCOM 2009*, 2009, pp. 1098–1106.

[28] A. Rényi, "On measures of entropy and information," in *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. Berkeley, Calif.: University of California Press, 1961, pp. 547–561. [Online]. Available: https://projecteuclid.org/euclid.bsmsp/1200512181

[29] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019.

[30] S. Lin, I. F. Akyildiz, P. Wang, and M. Luo, "Qos-aware adaptive routing in multi-layer hierarchical software defined networks: A reinforcement learning approach," in *2016 IEEE International Conference on Services Computing (SCC)*. Los Alamitos, CA, USA: IEEE Computer Society, jul 2016, pp. 25–33. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SCC.2016.12

[31] Q. Xu, Y. Zhang, K. Wu, J. Wang, and K. Lu, "Evaluating and boosting reinforcement learning for intra-domain routing," in *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2019, pp. 265–273.

[32] K. Rusek, J. Suárez-Varela, A. Mestres, P. Barlet-Ros, and A. Cabellos-Aparicio, "Unveiling the potential of graph neural networks for network modeling and optimization in sdn," *Proceedings of the 2019 ACM Symposium on SDN Research*, Apr 2019. [Online]. Available: http://dx.doi.org/10.1145/3314148.3314357

[33] R. Jain, P. R. Panda, and S. Subramoney, "Cooperative multi-agent reinforcement learning-based co-optimization of cores, caches, and on-chip network," *ACM Trans. Archit. Code Optim.*, vol. 14, no. 4, Nov. 2017. [Online]. Available: https://doi.org/10.1145/3132170

[34] J. Cui, Y. Liu, and A. Nallanathan, "Multi-agent reinforcement learning-based resource allocation for uav networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 2, pp. 729–743, Feb 2020.

[35] X. Li, J. Zhang, J. Bian, Y. Tong, and T.-Y. Liu, "A cooperative multi-agent reinforcement learning framework for resource balancing in complex logistics network," 2019.

[36] R. A. Howard, "Dynamic programming and markov processes," 1960.

[37] B. L. Miller and A. F. Veinott, "Discrete dynamic programming with a small interest rate," *The Annals of Mathematical Statistics*, vol. 40, no. 2, pp. 366–370, 1969. [Online]. Available: http://www.jstor.org/stable/2239451

[38] M. L. Puterman, "Markov decision processes," 1994.

[39] L. Wang, Q. Cai, Z. Yang, and Z. Wang, "Neural policy gradient methods: Global optimality and rates of convergence," 2019.

[40] R. Margolies, A. Sridharan, V. Aggarwal, R. Jana, N. K. Shankaranarayanan, V. A. Vaishampayan, and G. Zussman, "Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 355–367, 2016.

[41] R. Kwan, C. Leung, and J. Zhang, "Proportional fair multiuser scheduling in lte," *IEEE Signal Processing Letters*, vol. 16, no. 6, pp. 461–464, 2009.

[42] T. Bu, L. Li, and R. Ramjee, "Generalized proportional fair scheduling in third generation wireless data networks," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, 2006, pp. 1–12.

[43] X. Li, R. Shankaran, M. A. Orgun, G. Fang, and Y. Xu, "Resource allocation for underlay d2d communication with proportional fairness," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6244–6258, 2018.

[44] T. Mangla, N. Theera-Ampornpunt, M. H. Ammar, E. Zegura, and S. Bagchi, "Video through a crystal ball: effect of bandwidth prediction quality on adaptive streaming in mobile environments share on," *MoVid '16: in Proceedings of the 8th International Workshop on Mobile Video*, 2016.

[45] A. O. Al-Abbasi, V. Aggarwal, T. Lan, Y. Xiang, M. Ra, and Y. R. Chen, "Fasttrack: Minimizing stalls for cdn-based over-the-top video streaming systems," *CoRR*, vol. abs/1807.01147, 2018.