Appendix

No-regret Algorithms for Fair Resource Allocation

A. Sinha, A. Joshi, R. Bhattacharjee, C. Musco, M. Hajiesmaili

A Related Work

We provide a more comprehensive review of the fair machine learning literature in this section. Multiple different definitions have been used to quantify the fairness of machine learning algorithms. Hardt et al. [2016] introduced equality of opportunity as a fairness criterion, which ensures that individuals have an equal chance of being correctly classified by machine learning algorithms, regardless of their protected attributes like race or gender. Kleinberg et al. [2017] formalized three different notions of fairness and showed that no algorithm can satisfy these notions simultaneously, thus showing the inherent trade-offs in competing notions of fairness. Other prevalent fairness criteria include Price-of-fairness introduced by Bertsimas et al. [2011] which quantifies how much the aggregate utility is affected by enforcing fairness. Corbett-Davies and Goel [2016] discussed the limitations of different notions of fairness widely used in the fair machine learning literature. There have been different works that examined the societal harm caused by unfair algorithms in critical domains. For example, Chouldechova [2017] investigated the disparate impact of recidivism prediction algorithms used in the criminal justice system on different demographic groups. Since our work on online fair resource allocation is closely related to fairness in online algorithms, job scheduling, and matching, we provide a more detailed review of papers related to fair algorithms in each of these domains below.

Fairness in online algorithms: Several works have considered fairness specifically in the online setting using both regret and competitive ratio as performance metrics. Sinclair et al. [2022] considered the problem of allocating resources to individuals in an online setting, where the number and type of individuals arriving in each round are drawn from some fixed known distribution. The allocations must balance envy-freeness (requiring that each agent prefers the agent's own allocation over the other's allocation) and efficiency (the amount of consumed resources' budget). Banerjee et al. [2022] design competitive algorithms for the problem of allocating a set of divisible goods to agents in an online manner while maximizing the Nash social welfare, which is another metric that quantifies the trade-off between fairness and efficiency. They also study the setting when the algorithm is allowed to access some predictions related to the utility function of each agent over the goods. Arsenis and Kleinberg [2022] introduce notions of individual fairness in optimal stopping problems and provide competitive algorithms that satisfy these definitions of fairness.

There have been different works that consider fairness in terms of regret. Talebi and Proutiere [2018] studied a setting where tasks arriving dynamically need to be assigned to a set of servers where the probability of success of any job on any server is unknown. Using a stochastic multi-armed bandit setting, the authors showed that it is possible to achieve a proportionally fair allocation of jobs to servers in this setting. Multiple other works have studied the fairness problem in the stochastic multi-armed bandit setting [Patil et al., 2021, Joseph et al., 2016, Li et al., 2019, Chen et al., 2020]. There have also been some works recently that study algorithms with group fairness guarantees where we are concerned about being fair to a group of individuals (for example, a group might correspond to a race) instead of a single individual. [Freund et al., 2023] studies group fairness for algorithms used to assign refugees to different geographic locations. Baek and Farias [2021] studies the problem of sharing the cost of exploration in the stochastic multi-armed bandit model fairly among different groups.

Several prior works exist that design no-regret policies for specific non-additive functions. Blum and Burch [1997], Blum et al. [1999] used online learning techniques for solving the online paging and the Metrical Task System problem, which contains states. The notion of α-fairness in the online setting has been considered in a couple of works including Si Salem et al. [2022] as explained previously in the introduction. Wang et al. [2022] considered an online resource allocation problem where the

objective is to guarantee a sublinear regret for the allocation efficiency and a sublinear minimum guarantee violation penalty. The authors proposed an online policy that achieves this goal by using a weighted α-fair allocation on each round while sequentially tuning the weights and the exponents of the α-fairness function. Although they use round-wise α-fair allocation as a tool, their objective is not to optimize the α-fairness of the cumulative allocation, which is the focus of our paper.

Fairness in job scheduling: Apart from the papers cited above, the fair resource allocation problem in the context of job scheduling has also been extensively investigated by the wireless communication community, predominantly under stochastic assumptions [Stolyar, 2005a, Kushner and Whiting, 2002]. Furthermore, contrary to the online learning setting, these works typically assume that the current reward vectors (*e.g.,* the channel states) are available to the scheduler before it makes scheduling decisions for each round. In this setting, the classic proportional fair scheduling algorithm emerges as a stochastic approximation-type gradient ascent algorithm for maximizing the logarithmic utility function [Stolyar, 2005a, Kelly et al., 1998]. Other notions of fair utility functions that have been studied in the context of wireless communication include max-min utility [Bertsekas and Gallager, 1991, Jaffe, 1981] and α-fair utility [Mo and Walrand, 2000] (also called isoelastic utility function).

Fairness in matching: Fair matching algorithms have been studied in the context of resource allocation in online marketplaces in Bateni et al. [2021]. In this setting, a platform must dynamically allocate goods arriving at a platform in an online manner to customers who have their own utility and budget for each of these goods. The allocation of goods to customers must be fair to the customers while also maximizing the revenue for the platform. Different problems like online advertising where impressions must be served in an online manner to advertisers on a platform can be modeled under this setting. A proportionally fair stochastic approximation scheme was proposed in this paper. An algorithm with sublinear regret in a similar setting was proposed in Balseiro et al. [2021] where the fairness criteria were modeled as a regularizer for the objective function. Deng et al. [2023] studied how to incorporate machine-learned advice to improve fairness for bidders in the context of algorithmic bidding in online advertising. Competitive algorithms for optimizng both group and individual fairness in *online matching markets* which includes recommendation engines, crowdsourcing platforms etc. was proposed in Ma et al. [2023]. Fair matching algorithms for other domains like organ allocation Bertsimas et al. [2013] have also been proposed.

B Additional Examples

Beyond online fair caching, the NOFRA problem can capture a fair version of several other problems in the literature. In the following, we concretely present online fair job scheduling and online fair matching problems.

Example B.1 (Online Job Scheduling [Even-Dar et al., 2009])**.** In this problem, there are m machines that play the role of agents. A single job arrives at each round. The reward accrued by assigning the incoming job at round t to the i^{th} machine is given by $x_i(t)$ where $x_i(t) \in [\delta, 1], \forall i \in [m]$, where $\delta > 0$ is a small positive constant. Before the rewards for round t are revealed, an online allocation policy selects a probability distribution $\boldsymbol{y}_t$ on m machines such that

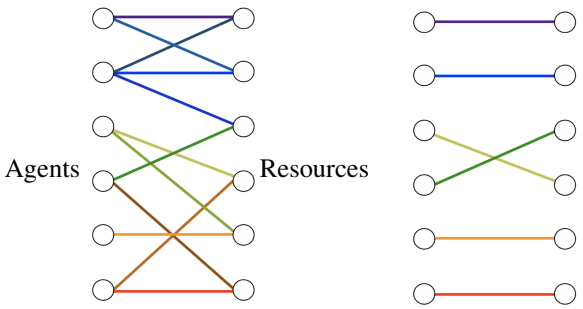
$$\sum_{i=1}^{m} y_i(t) = 1,\ y_i(t) \geq 0, \forall i \in [m].$$

The policy allocates a fraction $y_i(t)$ of the job to the i^{th} machine $\forall i \in [m]$. As a result, the i^{th} machine accrues a reward of $x_i(t)y_i(t)$ on round t. Hence, the cumulative reward accrued by the i^{th} machine in a time-horizon of length T is given by:

$$R_i(T) = 1 + \sum_{t=1}^{T} x_i(t)y_i(t),\ \forall i \in [m].$$

The objective of the online fair job scheduling problem is to design an online allocation policy that achieves a sublinear regret with respect to the α-fairness of the cumulative rewards defined in (4). From the above description, it is immediately clear that this problem is a special case of the general NOFRA problem. The online job scheduling problem occurs in many practical settings, *e.g.,* in the targeted ad-campaigning problem discussed in the introduction, the ads can be modelled as jobs, and different groups of users can be modelled as machines. In OFDMA wireless systems, the

fair allocation of frequency resource blocks to different users where the wireless channels change dynamically can be straightforwardly modelled as an instance of the online job scheduling problem.



A Bipartite graph $\mathcal{G}$ A Matching in $\mathcal{G}$

Figure 4: Illustrating the Online Fair Matching (OFM) problem for zero-one demands. The existence of an edge denotes unit demand, and likewise, the absence of any edge implies zero demand.

Example B.2 (Online Matching)**.** Consider an $m \times m$ bipartite graph where the vertices on the left denote the agents and the vertices on the right denote the resources. On every round, each agent can be matched with one resource only, where we allow fractional matchings. The $m \times 1$ demand vector of the agent i on round t is denoted by $x_i(t)$. The j^{th} component of the demand vector $x_i(t)$ denotes the potential reward accrued by the i^{th} agent had it been completely matched with the j^{th} resource on round t. Let the binary action variable $y_{ij}(t) \in [0, 1]$ denote the amount by which the agent i is matched with resource j on round t. Hence, the reward accrued by the agent i on time t is given by $\langle x_i(t), y_i(t) \rangle$. Let Δ denote the convex hull of all matchings. It is well known that Δ can be succinctly represented by the set of all $m \times m$ doubly stochastic matrices, *a.k.a.* the Birkhoff polytope [Ziegler, 2012]). In other words, in the OFM problem, the set of all feasible actions Δ consists of all $m \times m$ matrices $\left(y_{ij}\right)_{i,j}$ satisfying the following constraints [8]:

$$\sum_{i=1}^{m} y_{ij} = 1, \ \forall j, \quad \sum_{j=1}^{m} y_{ij} = 1, \ \forall i, \quad 0 \le y_{ij} \le 1, \ \forall i, j. \tag{13}$$

Let the variable $R_i(t)$ denote the cumulative rewards accrued by the i^{th} agent up to round t. Clearly,

$$R_i(t) = R_i(t-1) + \langle x_i(t), y_i(t) \rangle, \ R_i(0) = 1, \ \forall i \in [m], \tag{14}$$

It can be verified that Assumption 2 holds in this problem with $\mu = m^{-1}$ by noting that $m^{-1}\mathbf{1}_{m \times m} \in \Delta$. The objective of the Online Fair Matching (OFM) problem is to design an online matching policy that minimizes the c-Regret (4). From the above formulation, it can be immediately seen that the Online Matching problem is an instance of the NOFRA problem. Furthermore, it also generalizes the online scheduling problem described in Example B.1.

The online matching problem arises in numerous practical settings. For example, in the problem of online Ad Allocation, there are m different advertisers whose ads need to be placed in m different display slots on a webpage. Each slot can accommodate only one ad. On round t, a new user arrives and presents a reward vector $\boldsymbol{x}_i(t)$ for each advertiser. For example, the component $x_{ij}(t)$ could denote potential revenue accrued by the advertiser if the ad i is placed on the j^{th} slot on round t. The objective of the allocation policy is to match the ads to the slots on each round so that the total earned revenue is fairly distributed among the advertisers. Similar problems arise in designing recommendation systems for crowd-sourcing, online dating websites [Tu et al., 2014], and fair channel assignment in wireless networks [Altman et al., 2010].

[8] As before the fractional matching can be converted to a randomized integral matching via sampling. See Section C.6 for details.

C Proofs

C.1 Upper bound on the diameter of the admissible sets

Lemma 3. *For the online shared caching problem the diameter of the admissible action set can be bounded as follows:*

$$\mathit{Diam}(\Delta_k^N) \leq \sqrt{2k}.$$

Proof. Let $x, y \in \Delta_k^N$. We have

$$\|x-y\|_2^2 = \sum_{i=1}^{N}(x_i-y_i)^2 = \sum_{i=1}^{N}|x_i-y_i||x_i-y_i| \overset{(a)}{\leq} \sum_{i=1}^{N}|x_i-y_i| \overset{(b)}{\leq} \sum_{i=1}^{N}|x_i| + \sum_{i=1}^{N}|y_i| \overset{(c)}{=} 2k,$$

where, in (a), we have used the fact that $0 \leq x_i, y_i \leq 1$, in (b), we have used triangle inequality, and in (c), we have used the fact that the sum of the components of each feasible vector is k. □

Lemma 4. *For the online matching problem, the diameter of the admissible action set can be bounded as follows:*

$$\mathit{Diam}(\Delta) \leq \sqrt{2m}.$$

Let $x, y \in \Delta$, where Δ is the feasible set for the `OFM` problem. We have

$$\|x-y\|_2^2 = \sum_{i,j}(x_{ij}-y_{ij})^2 \leq \sum_{i,j}|x_{ij}-y_{ij}| \leq \sum_{ij}x_{ij} + \sum_{ij}y_{ij} \leq 2m,$$

where the inequalities follow from similar arguments as in the proof of the previous lemma.

C.2 Proof of Lemma 1

Proof. The expression for $\text{Regret}_T(\beta^{1-\alpha})$ from Eqn. (9) can be split into the difference of two terms A and B as follows:

$$\text{Regret}_T(\beta^{1-\alpha}) = \beta^{-\alpha}\Big[\underbrace{\sum_i \phi'(R_i(T))\sum_{t=1}^{T}\langle x_i(t), y_i^*\rangle}_{(A)} - \beta\underbrace{\sum_i \phi'(R_i(T))\sum_{t=1}^{T}\langle x_i(t), y_i(t)\rangle}_{(B)}\Big].$$

Also, denote the corresponding terms in the regret expression (10) for the surrogate learning problem by A' and B'. We will now separately bound each of the above two terms in terms of the corresponding terms in the regret expression (10) for the surrogate learning problem.

Proving $A \leq A'$: Recall that the utility function $\phi(\cdot)$ is concave. Hence, its derivative is non-increasing. Furthermore, under the action of any policy, the cumulative reward $R_i(\cdot)$ is non-decreasing for each user $i \in [m]$. Thus, we have $\phi'(R_i(t-1)) \geq \phi'(R_i(T))$ for all $t \in [T], i \in [m]$. Hence,

$$A \leq \sum_{t=1}^{T}\langle\sum_i \phi'\big(R_i(t-1)\big)x_i(t), y_i^*\rangle \leq A'. \tag{15}$$

Proving $B' \le (1-\alpha)^{-1}(B+m)$: We have

$$\begin{aligned}
B' &= \sum_i \sum_{t=1}^{T} \phi'(R_i(t-1))\langle x_i(t), y_i(t)\rangle \\
&= \sum_i \sum_{t=1}^{T} \phi'(R_i(t-1))\big(R_i(t) - R_i(t-1)\big) \\
&\overset{(a)}{\le} \sum_i \int_0^{R_i(T)} \phi'(R)dR \\
&= \sum_i \phi(R_i(T)) \\
&\overset{(b)}{=} (1-\alpha)^{-1} \sum_i \phi'(R_i(T))R_i(T) \\
&\overset{(c)}{=} (1-\alpha)^{-1} \sum_i \phi'(R_i(T))\Big(\sum_{t=1}^{T}\langle x_i(t), y_i(t)\rangle + 1\Big) \\
&\le (1-\alpha)^{-1}(B+m), \qquad (16)
\end{aligned}$$

where, in (a), we have used the fact that the derivative $\phi'(\cdot)$ is non-increasing and $R_i(t) - R_i(t-1) = \langle x_i(t), y_i(t)\rangle \le 1$ (see Appendix C.3 below for a proof by picture), in (b), we have used the explicit form of the α-fair utility function to substitute $x\phi'(x) = (1-\alpha)\phi(x)$, and in (c), we have used (2). Combining (15) and (16) and choosing $\beta = (1-\alpha)^{-1}$, we conclude that, letting L-Regret$_T$ be the surrogate regret defined as in (10),

$$\text{Regret}_T\big((1-\alpha)^{-(1-\alpha)}\big) \le (1-\alpha)^{\alpha} \cdot \text{L-Regret}_T + (1-\alpha)^{-(1-\alpha)}m.$$

□

C.3 Graphical proof of the inequality:

$$\phi'(R_i(t-1))\big(R_i(t) - R_i(t-1)\big) \le \int_{R_i(t-1)-1}^{R_i(t)-1} \phi'(R)dR. \qquad (17)$$

Proof. We use the fact that $\phi'(\cdot)$ is a non-increasing function and $0 \le R_i(t) - R_i(t-1) \le 1$. Then the inequality (17) follows straightforwardly from the schematic below.

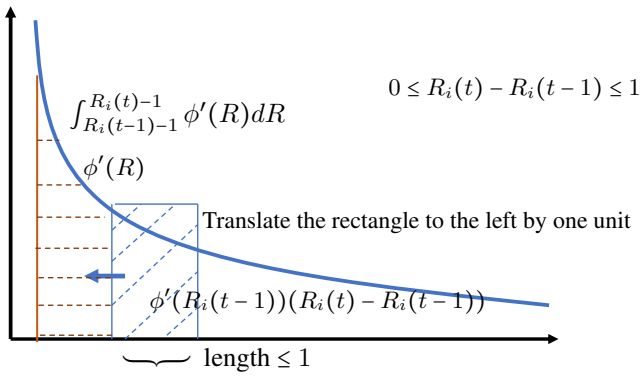


Figure 5: Graphical proof of (17)

The final inequality (a) then follows by summing up the above inequalities. □

C.4 Proof of Lemma 2

Proof. We will be using the following adaptive regret bound for the online gradient descent policy with an appropriate adaptive step sizes sequence. We will see that the norm of the gradients diminishes

at a steady rate under the action of the OFA policy. Hence, the data-dependent regret bound plays a central role in a tight regret analysis of the OFA policy.

Theorem 3 (Theorem 4.14 of Orabona [2020]). *Let $\Delta \subset \mathbb{R}^d$ be a convex set with a diameter D. Let us consider a sequence of linear reward functions with gradients $\{\boldsymbol{g}_t\}_{t\geq 1}$. Run the Online Gradient Ascent policy with step sizes $\eta_t = \frac{D}{\sqrt{2\sum_{\tau=1}^{t}\|\boldsymbol{g}_\tau\|_2^2}}, 1 \leq t \leq T$. Then the standard regret under the OGA policy can be upper-bounded as follows:*

$$\textit{Regret}_T \leq D\sqrt{2\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_2^2}. \tag{18}$$

Note that the gradient component g_i corresponding to the i^{th} user for the surrogate problem (10) on round t is given by the vector $g_i(t) = \phi'(R_i(t))x_i(t)$. Using the above data-dependent static regret bound (18), the regret achieved by the OGA policy for the surrogate problem for any round $T \geq 1$ can be upper-bounded as follows:

$$\text{L-Regret}_T = O\left(\sqrt{\sum_{t=1}^{T}\sum_i \phi'(R_i(t))^2}\right) = O\left(\sqrt{\sum_{t=1}^{T}\sum_i \frac{1}{R_i(t)^{2\alpha}}}\right), \tag{19}$$

where we have used the fact that the demand vectors at each round are bounded.

Clearly, the regret bound (19) depends on the sequence of the cumulative rewards $\{\boldsymbol{R}(t)\}_{t\geq 1}$, which is implicitly controlled by the past actions of the online policy itself. By the definition of regret, for any fixed allocation $y^\star \in \Delta_k^N$, we have for any time step T:

$$\sum_{t=1}^{T}\sum_i \langle \phi'(R_i(t-1))x_i(t), y_i(t)\rangle \geq \sum_{t=1}^{T}\sum_i \langle \phi'(R_i(t-1))x_i(t), y_i^*\rangle - \text{L-Regret}_T, \tag{20}$$

where L-Regret_T denotes the worst-case cumulative regret of the adaptive OGD policy up to time T. Since the cumulative reward of each user is monotonically non-decreasing, we have:

$$R_i(T) \geq 1, \forall i, \forall T. \tag{21}$$

Substituting this bound to the regret bound in (19), we obtain our first (loose) upper-bound for the regret of the fair allocation problem (10):

$$\text{L-Regret}_T = O(\sqrt{T}). \tag{22}$$

Note that this bound might be too loose as the offline benchmark itself could be smaller in magnitude than this regret bound. A closer inspection reveals the root cause for this looseness - the cumulative reward lower bound (21) is too loose for our purpose, as cumulative rewards grow steadily with time, depending on the online policy. We now strengthen the above upper-bound using a novel *bootstrapping* method, that *simultaneously* tightens the lower bound for the cumulative rewards (21) and, consequently, improves the regret upper bound (19).

Using the fact that $\langle x_i(t), y_i(t)\rangle = R_i(t) - R_i(t-1)$, and following the same calculations up to the fourth step of (16), we have

$$\sum_i \sum_{t=1}^{T} \phi'(R_i(t-1))\langle x_i(t), y_i(t)\rangle \leq \sum_i \phi(R_i(T)). \tag{23}$$

Furthermore, lower bounding $\phi'(R_i(t-1))$ by $\phi'(R_i(T))$, we have

$$\sum_{t=1}^{T}\sum_i \langle \phi'(R_i(t-1))x_i(t), y_i^*\rangle \geq \sum_i \phi'(R_i(T))R_i^*(T), \tag{24}$$

where $R_i^*(T) \equiv \sum_{t=1}^{T}\langle x_i(t), y_i^*\rangle$ is the cumulative reward accrued by a fixed allocation $y^\star \in \Delta$ up to time T. Using Assumptions 1 and 2 and choosing $y_i^* = \mu \mathbb{1}_N$, we have $R_i^*(T) \geq \mu\delta T, \forall i$. Hence, combining (23), (24), and (20), we have

$$\sum_i \phi(R_i(T)) \geq \mu\delta T \sum_i \phi'(R_i(T)) - \text{L-Regret}_T. \tag{25}$$

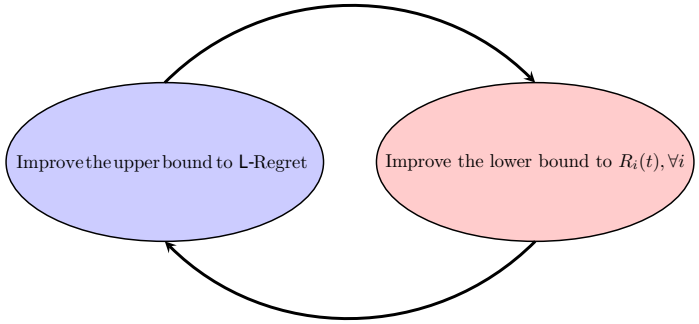


Figure 6: Illustrating the new *Bootstrapping* technique used in the proof of Lemma 2

Since $0 < R_i(T) \leq T, \forall i$, and $\phi(\cdot)$ is monotone non-decreasing, for any user $i \in [m]$, the above inequality yields:

$$\frac{m}{(1-\alpha)T^{\alpha}} \geq \frac{\mu\delta}{R_i^{\alpha}(T)} - \frac{\text{L-Regret}_T}{T}.$$

i.e.,

$$R_i^{\alpha}(T) \geq \mu\delta\left(\frac{\text{L-Regret}_T}{T} + \frac{m}{1-\alpha}T^{-\alpha}\right)^{-1} = \Omega(T^{\min(1/2,\alpha)}), \forall i \in [m], \tag{26}$$

where we have used our previous upper bound (22) on the regret. Eqn. (26) is our key equation for carrying out the bootstrapping process as it lower bounds the minimum cumulative reward of users in terms of the worst-case regret. Now we consider two cases:

Case-I: $0 \leq \alpha \leq 1/2$

In this case, we immediately have $R_i(T) = \Omega(T), \forall i, T$. Substituting this bound on (19), we have

$$\text{L-Regret}_T = \begin{cases} O(\sqrt{\log T}) & \text{if } \alpha = 1/2 \\ O(T^{1/2-\alpha}) & \text{if } 0 < \alpha < 1/2. \end{cases}$$

Case-II: $1/2 < \alpha < 1$

In this case, using the bound (26), we have $R_i(T) = \Omega(T^{1/2\alpha}), \forall i, T$. Using the regret bound (19), the regret can be bounded as

$$\text{L-Regret}_T = O(\sqrt{\sum_{t=1}^{T}\frac{1}{t}}) = O(\sqrt{\log T}).$$

Substituting this bound again in (26), we have $R_i(T) \geq \Omega(T)$. This, in turn, yields the following regret bound:

$$\text{L-Regret}_T = O\Big(\sqrt{\sum_{t=1}^{T}\frac{1}{t^{2\alpha}}}\Big) = O(1).$$

□

C.5 Simplified Pseudocode for the Online Fair Allocation Policy for the Caching Problem

Algorithm 2 Online Fair Allocation Caching (`OPFC`)

Input: Fairness parameter $0 \leq \alpha < 1$. Online file request sequence $\{x_i(t)\}_{t=1}^{T}$ from user $i, i \in [m]$, Euclidean projection oracle $\Pi_{\Delta_k^N}(\cdot)$ onto the feasible convex set Δ_k^N.
Output: Sequence of file inclusion probabilities $\{y_t\}_{t=1}^{T}$ and samples from this distribution
$R_i \leftarrow 1, \forall i \in [m], S \leftarrow 0.$ ▷ *Initialization*
for each round $t = 2 : T$: **do**
  $g \leftarrow \sum_{i=1}^{m} \frac{x_i(t-1)}{R_i^{\alpha}}$ ▷ *Computing the gradient*
  $S \leftarrow S + \|g\|_2^2,$ ▷ *Accumulating the norm of the gradients*
  $y \leftarrow \Pi_{\Delta_k^N}\left(y + \frac{k}{\sqrt{S}} g\right)$ ▷ *Updating the inclusion probabilities using OGA*
  The users reveal their file requests for round t $\{x_i(t), i \in [m]\}$.
  $R_i \leftarrow R_i + \langle x_i(t), y \rangle, \ \forall i \in [m].$ ▷ *Updating cumulative hits*
  Sample a subset of k files by invoking Algorithm 3
end for each

C.6 High-probability regret bound for randomized integral allocations

Theorem 2 establishes $c_\alpha \equiv (1-\alpha)^{-(1-\alpha)}$-regret guarantee for the Online Fair Allocation (`OFA`) policy for fractional allocations where the set of admissible actions Δ is the convex hull of integral allocations. In the following, we show that a sub-linear c_α-regret guarantee holds with high probability when randomized integral actions are chosen according to the optional step (9) in the `OFA` policy. Recall that for integral allocation, on each round t we independently sample an integral allocation $Y_t \in \{0,1\}^{mN}$, that matches with the fractional allocation y_t in expectation, *i.e.,* $\mathbb{E}[Y_t] = y_t$ where y_t is the fractional allocation vector recommended by the `OFA` policy. A fractional allocation vector y_t in the convex hull of integral allocations can be turned into a randomized integral allocation by expressing y_t as a convex combination of integral allocations and randomly sampling one of the integral allocations with appropriate probabilities. As Appendix C.7 shows, for many problems with structure, such a decomposition can be efficiently computed.

Regret Analysis: By appealing to [Cesa-Bianchi and Lugosi, 2006, Lemma 4.1], it is enough to consider an oblivious adversary that fixes the sequence of demand vectors $\{\boldsymbol{x}(t)\}_{t\geq 1}$ before the game commences. Let the random variable $\mathsf{R}_i(T)$ denote the random cumulative reward obtained by the i^{th} user under the randomized integral allocation policy and the deterministic variable $R_i(T)$ is defined as in Eqn. (2). By construction, we have $\mathbb{E}[\mathsf{R}_i(T)] = R_i(T)$. Furthermore, since $\mathsf{R}_i(T)$ is the sum of T independent random variables, each of magnitude at most one, from the standard Hoeffding's inequality, we have

$$\mathbb{P}(|\mathsf{R}_i(T) - R_i(T)| \geq \lambda) \leq \exp(-2\lambda^2/T), \forall i \in [m].$$

Hence, with probability at least $1 - \texttt{poly}(1/T)$, the aggregate α-fair utility (3) accrued by the randomized policy can be lower bounded as:

$$\begin{aligned}(1-\alpha)^{-1}\sum_i \mathsf{R}_i^{1-\alpha}(T) \quad &\geq \quad (1-\alpha)^{-1}\sum_i \left(R_i(T) - O(\sqrt{T\log T})\right)^{1-\alpha} \\ &\overset{(a)}{\geq} \quad (1-\alpha)^{-1}\sum_i R_i^{1-\alpha}(T) - O\big((T\log T)^{\frac{1-\alpha}{2}}\big), \end{aligned} \tag{27}$$

where in inequality (a), we have used the fact that $(x+y)^{1-\alpha} \leq x^{1-\alpha} + y^{1-\alpha}, \forall 0 \leq \alpha \leq 1, x, y \geq 0$. Combining the above with the approximate regret bound in Theorem 1, and taking the dominant term, we conclude that the c_α-regret for the randomized integral allocation policy is upper-bounded by $O\big((T\log T)^{\frac{1-\alpha}{2}}\big)$ w.h.p. for all $0 \leq \alpha \leq 1$. Unfortunately, unlike Theorem 1, the integral allocation policy incurs a non-trivial (but sublinear) c_α-regret for the entire range of the fairness parameter $0 \leq \alpha < 1$.

C.7 Efficiently sampling an integral allocation from a mixed allocation in the convex hull

1. Online shared caching: For the online shared caching problem, the admissible set Δ is given by Eq. (5). Clearly, incidence vectors of all k-sets, containing k 1's and $N-k$ zeros, belong to Δ. Furthermore, given any vector in $\boldsymbol{y} \in \Delta$, a randomized integral allocation can be sampled in linear time using Madow's sampling scheme, described in Algorithm 3, which yields the vector $\boldsymbol{y}$ in expectation.

Algorithm 3 MADOW-SAMPLE (p)

Input: A universe $[N]$ of size N, the cardinality of the sampled set k, and a marginal inclusion probability vector $p \in \Delta_k^N$.
Output: A random k-set S with $|S| = k$ such that, $\mathbb{P}(i \in S) = p_i, \forall i \in [N]$
Define $\Pi_0 = 0$, and $\Pi_i = \Pi_{i-1} + p_i, \forall 1 \leq i \leq N$.
Sample a uniformly distributed random variable U from the interval $[0, 1]$.
$S \leftarrow \varnothing$
for each $i \leftarrow 0$ to $k-1$ **do**
    Select the element j if $\Pi_{j-1} \leq U + i < \Pi_j$.
end for each
return S

2. Online job scheduling: In the online job scheduling problem, the admissible action set Δ is given by the N-simplex. Given any point on the N-simplex, it is trivial to randomly sample a coordinate using a single uniform random variable.

3. Online matching: In the online matching problem, the admissible action set Δ is given by the Birkhoff polytope (13). Given any point $\boldsymbol{y}$ in the Birkhoff polytope, it can be efficiently decomposed as a convex combination of a small number of matchings using the Birkhoff-von-Neumann (BvN) decomposition algorithm [Valls et al., 2021]. Using the decomposition coefficients, a matching can be randomly sampled that exactly matches the point $\boldsymbol{y}$ in expectation.

C.8 Non-Convexity of the optimal offline benchmark for the online job scheduling problem

Consider the job scheduling problem in the reward maximization setting as described in Example B.1. Clearly, the α-fairness metric accumulated by the online policy is concave in the regime $0 \leq \alpha \leq 1$. However, the following proposition shows that the offline static benchmark for this problem is *non-convex* for the α-fair utility function in the regime $0 < \alpha < 1$.

Proposition 1. *The offline optimal α-fair reward function for the job scheduling problem is non-convex for $0 < \alpha < 1$.*

Proof. Let the probability distribution $\boldsymbol{y}^*$ be an optimal static offline allocation vector for the reward sequence $\{\boldsymbol{x}_t\}_{t \geq 1}$. Also, let $R_T(i) \equiv \sum_t x_i(t)$ be the cumulative reward observed by the i^{th} machine, $1 \leq i \leq m$. Then the optimal allocation $\boldsymbol{y}^*$ maximizes the following objective:

$$(1-\alpha)^{-1} \sum_i (y_i R_T(i))^{1-\alpha}. \tag{28}$$

s.t. the constraints $\sum_i y_i = 1, \boldsymbol{y} \geq \boldsymbol{0}$. Since $0 < \alpha < 1$, the objective function is strongly concave and the optimal solution $\boldsymbol{y}^*$ is unique. Using the standard Hölder's inequality with the conjugate norms $p = 1/1-\alpha, q = 1/\alpha$, we can upper bound the objective (28) as

$$\begin{aligned} (1-\alpha)^{-1} \sum_{i=1}^{m} (y_i R_T(i))^{1-\alpha} &\leq (1-\alpha)^{-1} (\sum_i y_i)^{1-\alpha} (\sum_i R_T(i)^{\frac{1-\alpha}{\alpha}})^{\alpha} \\ &= (1-\alpha)^{-1} (\sum_i R_T(i)^{\frac{1-\alpha}{\alpha}})^{\alpha}, \end{aligned} \tag{29}$$

where we have used the constraint $\sum_i y_i = 1$ in the last equality. The upper-bound is achieved by the following distribution

$$y_i^* = \frac{R_i^{\frac{1-\alpha}{\alpha}}}{\sum_i R_i^{\frac{1-\alpha}{\alpha}}}, \ \forall i \in [m]. \tag{30}$$

Hence, the optimal offline reward (28) is given by

$$\mathcal{R}^* = (1-\alpha)^{-1}\Big(\sum_i R_T(i)^{\frac{1-\alpha}{\alpha}}\Big)^{\alpha}. \tag{31}$$

To show that $\mathcal{R}^*$ is non-convex in the cumulative reward vector $\boldsymbol{R}_T$ in the regime $0 < \alpha < 1$, consider the case of two users. Letting $x = R_T(1), y = R_T(2)$ and ignoring the positive pre-factor, the function under consideration is given as follows:

$$f(x,y) = \big(x^{\frac{1-\alpha}{\alpha}} + y^{\frac{1-\alpha}{\alpha}}\big)^{\alpha}. \tag{32}$$

Recall that a function is convex if and only if the determinants of all leading principal minors of its Hessian matrix are non-negative. A straightforward computation yields the following expression for the determinant of the Hessian of the function $f(x,y)$:

$$\det(\nabla^2 f(x,y)) = (2\alpha-1)\frac{(\alpha-1)^2 x^{1/\alpha-1} y^{1/\alpha-1}(x^{1/\alpha-1}+y^{1/\alpha-1})^{2\alpha}}{(yx^{1/\alpha}+xy^{1/\alpha})^2}.$$

The determinant becomes strictly negative in the regime $0 < \alpha < \frac{1}{2}$. This shows that the function (32) is non-convex for $0 < \alpha < \frac{1}{2}$. On the other hand, we have

$$\frac{\partial^2 f(x,y)}{\partial x^2} = (\alpha-1)\frac{yx^{1/\alpha-2}(x^{1/\alpha-1}+y^{1/\alpha-1})^{\alpha}(\alpha^2 yx^{1/\alpha}+2\alpha xy^{1/\alpha}-xy^{1/\alpha})}{\alpha(yx^{1/\alpha}+xy^{1/\alpha})^2}.$$

In particular, at the point $(1,1)$ the above second partial derivative evaluates to be

$$\frac{\partial^2 f(x,y)}{\partial x^2}\Big|_{(x,y)=(1,1)} = -\frac{2^{\alpha-2}(1-\alpha)}{\alpha}(\alpha^2+2\alpha-1),$$

which is strictly negative for $\sqrt{2}-1 < \alpha < 1$. Taking the above two results together, we conclude that the offline optimal reward function (32) is non-convex for $0 < \alpha < 1$. □

C.9 Proof of Theorem 2 (lower bounding the approximation factor)

Consider the following instance of the online shared caching problem with $m = 2$ users and a cache of unit capacity. Assume that the library's size (N) is sufficiently large. Let the total length of the request sequence be T rounds and let $\eta \in [0,1]$ be some constant fraction to be fixed later. Consider two different file request sequences:

- **Instance 1:** For the first ηT rounds, user 1 always requests file 1 and user 2 always requests file 2. For the next $(1-\eta)T$ rounds, user 1 always requests file 2 and user 2 requests a file chosen uniformly at random from the library.
- **Instance 2:** For the first ηT rounds, as in Instance 1, user 1 always requests file 1 and user 2 always requests file 2. However, for the next $(1-\eta)T$ rounds, user 1 requests a random file chosen uniformly at random from the library and user 2 always requests file 1.

We now lower bound the approximation factor achievable by any online policy for the above two request sequences.

Optimal static offline policy: It is easy to see that the optimal offline strategy is to cache file 2 for all T rounds. Hence, the total α-fairness objective accrued by the static offline policy is

$$\text{Offline } \alpha\text{-fairness} = (1-\alpha)^{-1}T^{1-\alpha}\big[\eta^{1-\alpha} + (1-\eta)^{1-\alpha}\big].$$

Clearly, the same fairness objective is achieved for the second instance by a static policy that always caches file 1.

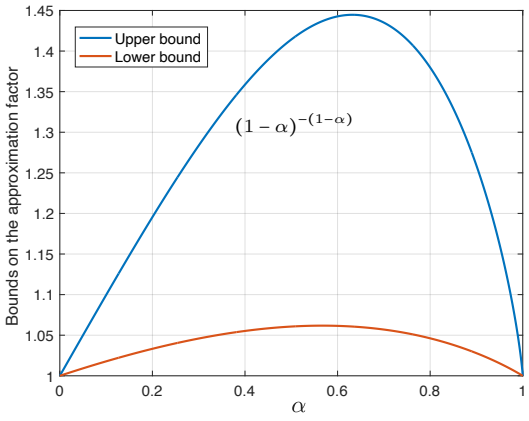


Figure 7: Comparison between the upper and lower bounds of the approximation ratio

Online Policy: Suppose that during the first ηT rounds, the online policy caches file 1 for γ fraction of the time and file 2 for $1-\gamma$ fraction of the time. Since the policy is online, the fraction γ remains the same for both instances. Clearly, for the next $(1-\eta)T$ time slots, an optimal online policy caches file 2 for instance 1 and file 1 for instance 2. Taking the worst of the two instances, any online policy achieves the following fairness objective:

$$\text{Online } \alpha\text{-fairness} = (1-\alpha)^{-1} T^{1-\alpha} \min\big(g(\gamma), g(1-\gamma)\big),$$

where $g(\gamma) \equiv \big(\gamma\eta + (1-\eta)\big)^{1-\alpha} + \big((1-\gamma)\eta\big)^{1-\alpha}$. From simple calculus, it follows that the function $g(\gamma)$ is non-increasing for $0 \le \eta \le 1/2, 0 \le \alpha < 1$. This implies that the online α-fairness is maximized when $\gamma^* = \frac{1}{2}$. Hence, the α-fairness metric achieved by any online policy for the worst of the above two instances is upper bounded by:

$$\text{Online } \alpha\text{-fairness} \le (1-\eta/2)^{1-\alpha} + (\eta/2)^{1-\alpha}.$$

Hence, the achievable approximation ratio c_α for any online policy with a sublinear c_α-regret is lower bounded as follows:

$$\text{Approx. ratio} \ge \frac{\text{Offline } \alpha\text{-fairness}}{\text{Online } \alpha\text{-fairness}} = \frac{\eta^{1-\alpha} + (1-\eta)^{1-\alpha}}{(1-\eta/2)^{1-\alpha} + (\eta/2)^{1-\alpha}}.$$

Taking the maximum of the RHS with respect to the parameter η yields the desired result.

D Additional Numerical Experiments

In this section, we provide a more detailed description of how the CDN dataset was preprocessed and additional numerical experiments on fair scheduling.

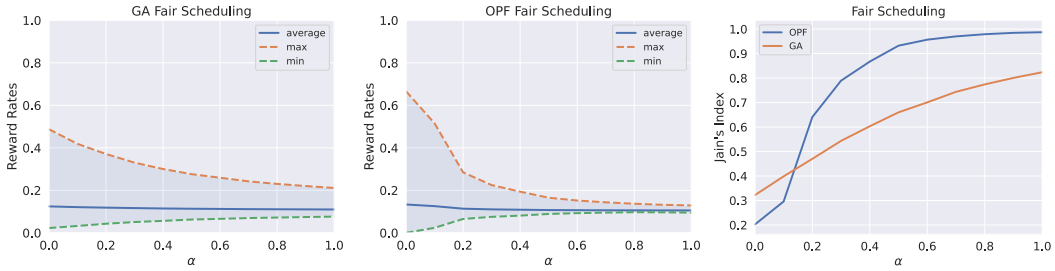
D.1 Preprocessing the CDN dataset

We give additional details regarding preprocessing the CDN dataset for the experiments here. As explained in section 4 we set $T = 400$ rounds, number of users $m = 4$, cache size $C = 10$ and, library size $N = 50$. For preprocessing, first, we sort the data according to the timestamp and discard all the files with an ID greater than N, and then only consider the first $m \times T = 1600$ requests among the remaining requests. We then relabel the files from 1 to N in descending order of the number of requests such that among these 1600 requests, file 1 is the most requested file, followed by the second file, and so on. We then assign the requests sequentially to users in decreasing order of priority from users 1-4, i.e., we first try to assign a request to user 1, failing which we try to assign it to user 2, and then to users 3 and 4. We can only assign a request to a user if the total number of requests assigned to that user is less than or equal to $T = 400$ and the following conditions are met for users 1-3: (1)

user 1: file ID between 1-6; (2) user 2: file ID between 1-9; (3) user 3: file ID between 1-12. User 4 can be assigned any file. Thus, we allocate files to users in decreasing order of their popularity, such that user 1 tends to request common files, while user 4 tends to request less common files. This skews the file request distribution, making a fair cache allocation for all users difficult.

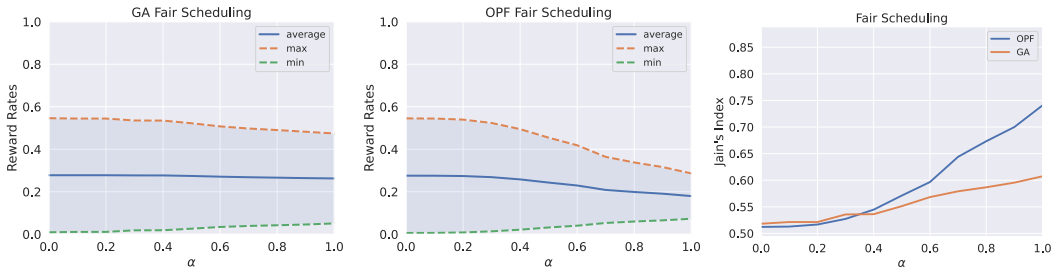
D.2 Fair Scheduling

Dataset: The datasets are generated by simulating a wireless channel. For the first dataset (*WLAN*), the simulator generates channel gains and shadowing values for each user, computes the received power and noise for each user, adds Rayleigh fading to the signal-to-noise ratios (SNRs), and then normalizes the SNRs by the maximum value to generate rewards. We take the horizon $T = 5000$ and number of users $m = 5$. The second dataset (*MCS*) is generated using Modulation and Coding Scheme table which defines the data rates for each user depending on the radio link quality. Here, the number of users is $m = 2$ and the horizon is at $T = 2000$. Although regret guarantees for $\alpha > 1$ become vacuous, to clearly demonstrate the fairness characteristics of algorithms, we run simulations for $\alpha \in [0, 2]$.



(a) Reward Rates: GA (b) Reward Rates: OFA (c) Jain's Index

Figure 8: Scheduling (WLAN)



(a) Reward Rates: GA (b) Reward Rates: OFA (c) Jain's Index

Figure 9: Scheduling (WLAN)

Comparison: We compare our policy with the Gradient Algorithm (GA) described by Stolyar [2005b]. The Gradient Algorithm is a widely used in fairly scheduling transmission of multiple users via a time-varying wireless channel. Stolyar considers a model where a finite set of queues are served by a switch. The switch has a finite number of switch states M. The sequence of states at each time steps form an irreducible Markov chain. At each state m, a scheduling decision $k \in K(m)$ is made, where $K(m)$ is a finite set of decisions. Each scheduling decision has a corresponding vector of service rates given by $\mu^m(k)$. Let V be the set of all long-term average service rate vectors. The goal is to have a scheduling policy that gives an average service rate solving the maximization problem

$$\max_{u \in V} H(u), \tag{33}$$

where $H(\cdot)$ is a strictly concave smooth utility function. The Gradient Algorithm makes a scheduling decision

$$k(t) \in \underset{k}{\operatorname{argmax}} \langle \nabla H(X(t)), \mu^m(k) \rangle,$$

where $X(t)$ is updated as

$$X(t+1) = (1-\beta)X(t) + \beta\mu^m(k),$$

for some initial value of $X(0)$ and a fixed parameter $\beta > 0$. The Gradient Algorithm converges to an optimal u^* that maximizes (33) as $\beta \to 0$.

Figures 8a, 8b, 9a and 9b compares the average, minimum and maximum reward rates of both the policies as α increases. The Jain's Index of both the policies is shown in Figures 8c and 9c. It can be clearly observed that for $\alpha \in [0, 1)$, the OFA policy performs better in terms of fairness, but for larger α's, the GA outperforms the OFA policy by a small margin. Recall that Stolyar only provides theoretical guarantees for a stochastic setting whereas we give a stronger regret bound in an adversarial setting at the expense of slight performance degradation in practice.