IEEE INFOCOM 2025 - IEEE Conference on Computer Communications | 979-8-3315-4305-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/INFOCOM55648.2025.11044643

FairRIC: Real-time Fair Allocation in O-RAN with Shared Computing

Fatih Aslan*, Jose A. Ayala-Romero†, Andres Garcia-Saavedra†, Xavier Costa-Perez†‡, George Iosifidis*
*Delft University of Technology, The Netherlands, †NEC Laboratories Europe GmbH, Germany
‡i2CAT Foundation and ICREA, Spain

***Abstract*—The deployment of O-RAN systems on general-purpose computing platforms represents a significant paradigm shift, promising remarkable performance improvements. However, these architectures may potentially increase both the capital and operational expenses of the network. The processor pooling concept is a promising solution to address this problem, consisting of a set of processing units (PUs) in the O-Cloud shared by several virtualized BSs (vBSs). Nevertheless, this strategy requires sophisticated resource assignment mechanisms to provide the expected gains in terms of cost and reliability. This paper proposes a novel online learning framework that assigns computing resources to vBSs in real-time (e.g., every TTI), thus handling the burstiness of real traffic loads. Our algorithm relies on online convex optimization (OCO) theory, extending state-of-the-art approaches in long-term fairness and constrained optimization and allowing discrete decisions. Our method offers an intrinsic closed-form iteration, speeding up the computation process and consequently allowing real-time operation. Moreover, our solution has guarantees in terms of fairness among the vBSs while adhering to long-term energy constraints over the entire operation horizon. We validate our theoretical findings via simulation and evaluate experimentally the algorithms in an O-RAN platform.**

I. Introduction

A pivotal advancement in future mobile networks is the virtualization of the Radio Access Network (vRAN), particularly vBSs, and their deployment on general-purpose computing hardware [1]. vRANs offer unprecedented flexibility, enabling dynamic adaptation of vBSs to varying channel conditions and diverse user demands for throughput, latency, and other performance metrics. To meet these demands, the Open RAN (O-RAN) architecture incorporates computing pools (O-Cloud) comprising heterogeneous PUs, including CPUs and accelerators like ASICs, FPGAs, and GPUs. This approach facilitates efficient management of computational workloads across vBSs, representing a paradigm shift in RAN technology with the potential for substantial performance gains [1].

However, the virtualization of RANs is expected to increase the Operating Expenditures (OpEx) of networks. The reason is that, unlike legacy base stations, the energy required for executing software vBS functions becomes significant and can potentially exceed that of wireless transmissions [2]. This issue, compounded by the increasing densification of RANs, is expected to escalate the energy costs associated with vRANs to prohibitively high levels for future mobile networks [3], [4].

A potential direction to alleviate this problem is the concept of *processor pooling*. By default, PUs are co-located with and exclusively used by individual vBSs, leading to low utilization under real workloads [5]. Sharing multiple PUs among multiple vBSs can reduce energy consumption (OpEx) and distribute

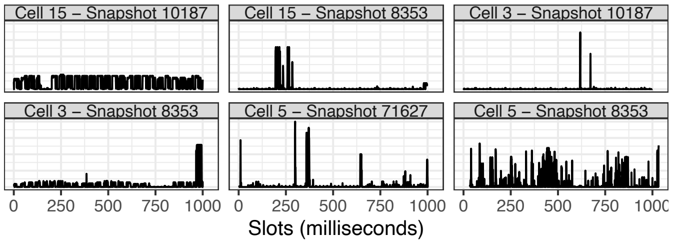

Fig. 1: Normalized load of 3 cells at small timescales.

the capital expenditure (CapEx) of such equipment, making the vBSs more affordable.

However, processor pooling necessitates sophisticated resource allocation mechanisms. Solving this problem is challenging due to several factors. First, the heterogeneity of PUs, with varying processing capacities and energy consumption profiles, makes them better suited for specific tasks (e.g., small data chunks or transport blocks (TBs)). Second, the bursty and time-varying nature of vBS load demands dynamic resource allocation. Third, the processing time for traffic is highly volatile, depending on parameters like signal modulation coding scheme (MCS), signal-to-noise ratio (SNR), and TB length. Moreover, incorrect resource allocation can lead to PU saturation, causing user data loss and degrading network QoS.

Previous works have addressed computing resource assignment in O-RAN systems using processor pooling [5], [6], [7]. These approaches align with O-RAN specifications by considering decision-making at both near-real-time (near-RT) and non-RT timescales [6], [7]. However, as shown in the literature [8] their decision granularity may be too coarse to adapt to the inherent burstiness of traffic. This is also confirmed by our own experiments. Using Falcon [9], we collected millisecond-resolution workload data from diverse cells operated by multiple providers in Madrid, Spain. Fig. 1 illustrates this burstiness, showing the instantaneous load of three production cells over 10 seconds at different times. The bursty nature of real-world RAN traffic at small timescales necessitates real-time control strategies to effectively exploit multiplexing opportunities. To manage traffic burstiness, the authors in [5] propose a real-time computational resource allocation framework. However, their algorithm relies on a greedy heuristic without performance guarantees, and throughput loss is not fairly distributed among vBSs when PUs are saturated.

This paper presents FairRIC, a novel real-time fair resource allocation algorithm for virtualized O-RAN platforms. Our solution dynamically and fairly assigns workloads to heterogeneous PUs while meeting long-term energy constraints.

By addressing both fairness and real-time operation, FairRIC aims to optimize resource utilization and enhance network performance, effectively handling the burstiness of real traffic loads. FairRIC builds upon OCO theory [10], [11], and specifically leverages the well-known Follow-The-Regularized-Leader (FTRL) framework [12]. We modify FTRL making it fit for our specific use case, namely, we optimize for long-term fairness of the vBSs sharing the O-Cloud resources following a primal-dual approach [7], [13] and restrict the power consumption using Lazy Lagrangians [14]. We propose a novel solution for long-term fairness that departs from the traditional primal-dual update approach, where both primal and dual variables are updated iteratively [7], [13]. Our method offers an intrinsic closed-form iteration, allowing for the direct calculation of the dual variable (for long-term fairness) solely based on the past primal variables and a concave utility function. This approach eliminates the need for separate updates of the dual variable, streamlining the computation process and allowing real-time operation. Moreover, we extend previous works on long-term fairness and constrained optimization and make the framework suitable for discrete decisions, required by real-time O-RAN computing resource assignment. Finally, we conduct an extensive set of simulation and experimental studies demonstrating significant improvements in terms of fairness and efficiency over existing methods.

II. System Model & Problem Statement

Background. The physical layer (PHY) of cellular base stations, responsible for baseband processing, operates under stringent deadlines [5], [8]. Radio scheduling decisions are made at each Transmission Time Interval (TTI) over TBs, which are indivisible data chunks sent between the vBSs and users. For the uplink, the vBS must decode the TBs coming from users within each TTI, utilizing PUs that are typically located at the O-Cloud. The O-Cloud, a central component of O-RAN, provides computational resources through a pool of heterogeneous PUs, including CPUs and hardware accelerators (HAs) such as GPUs and FPGAs. In our scenario, we consider these PUs to be shared among multiple vBSs, making fair and efficient resource allocation essential for meeting real-time processing requirements. The PUs available in the O-Cloud exhibit varying performance characteristics. CPUs, for instance, are cost-effective and energy-efficient but relatively slow for processing PHY signals. As a result, relying solely on CPUs may not meet strict deadlines. In contrast, HAs such as GPUs and FPGAs are more expensive (between 5 and 50$\times$ more expensive than a CPU core) and energy-hungry. While the opportunistic use of CPUs can reduce energy usage, it may also introduce delays in decoding. TBs not decoded within the deadline (e.g., 1 ms) are considered lost and require retransmission. Consequently, effective sharing of computational resources is vital for ensuring network reliability and maintaining users' QoS.

Notation. Vectors are denoted by boldface lowercase letters and matrices are denoted by boldface capital letters. Subscript denotes the time index and element of a vector/matrix. $[\cdot]_+$ denotes $\max\{0,\cdot\}$, and $\|\cdot\|_2, \|\cdot\|_\infty$ denote ℓ_2 and ℓ_∞ norm respectively. We use the shorthand notation $x_{1:T} = \sum_{t=1}^T x_t$ and write $\{x\}_{1:T}$ to denote the sequence $\{x_1, x_2, \cdots x_T\}$. All proofs are provided in the appendix.

We tackle the compute load assignment problem in O-RAN, where a set $\mathcal{I}$ of vBSs assign their workloads to a set $\mathcal{J}$ of PUs. In O-RAN, an algorithm must make decisions every 1 ms (matching the TTI) to qualify as a real-time algorithm. Hence, each vBS must be assigned to exactly one PU, as the TBs are atomic and cannot be split across multiple PUs. We define the assignment vector $\boldsymbol{x} \in \{0,1\}^{I\cdot J}$, with $x_{ij} = 1$ if vBS i is assigned to PU j, that belongs to set:

$$\mathcal{X}^d = \left\{ \boldsymbol{x} \in \{0,1\}^{I\cdot J}, \sum_{j\in\mathcal{J}} x_{ij} = 1, \ \forall i \in \mathcal{I} \right\}.$$

We model the utility gained from the decision $\boldsymbol{x}$ at time-slot t with utility vector $\boldsymbol{u}_t(\boldsymbol{x}) : \mathbb{R}^{I\cdot J} \mapsto \mathbb{R}^I$. We model the assignment decisions as an online (non)convex optimization problem [15], where the utility function can be chosen flexibly (for each type of O-RAN, for instance) as long as it is concave, positive, bounded, and Lipschitz continuous. One meaningful choice is to consider the normalized throughput, i.e., a portion of successfully decoded bits within the deadline of 1 ms of each vBS. Sec. IV provides details on how to calculate the normalized throughput.

We measure the fairness of allocations using the generalized α-fairness function $F_\alpha(\boldsymbol{u}) = \sum_{i\in\mathcal{I}} f_\alpha(u_i)$, [16], [17], where

$$f_\alpha(u_i) = \begin{cases} \frac{u_i^{1-\alpha}-1}{1-\alpha}, & \text{for } \alpha \in \mathbb{R}_{\geq 0}\backslash\{1\}, \\ \log(u_i), & \text{for } \alpha = 1. \end{cases}$$

Note that we calculate the fairness directly over the utility function, capturing the more general utility-fair model.

OCO allows us to produce dynamic decisions as good as the best decision in hindsight even when the environment consists of adversarial (worst-case) perturbations, namely utility functions. This is quantified by the typical metric of regret [15]. Here, we are interested in the long-term fairness, and hence the regret needs to be redefined. In particular, following recent works such as [7], [13], we will be using the regret metric:

$$\mathcal{R}_{T,\alpha} \doteq F_\alpha\left(\frac{1}{T}\sum_{t=1}^T \boldsymbol{u}_t(\boldsymbol{x}^\star)\right) - F_\alpha\left(\frac{1}{T}\sum_{t=1}^T \boldsymbol{u}_t(\boldsymbol{x}_t)\right)$$

where $\boldsymbol{x}_t \in \mathcal{X}^d$ are the discrete decisions produced by our algorithm and $\boldsymbol{x}^\star$ is a (possibly randomized) benchmark policy (the oracle) selected with future knowledge by solving

$$\boldsymbol{x}^\star = \arg\max_{\boldsymbol{x}\in\mathcal{X}^c} F_\alpha\left(\frac{1}{T}\sum_{t=1}^T \boldsymbol{u}_t(\boldsymbol{x})\right), \text{ where } \mathcal{X}^c \text{ is}$$

$$\mathcal{X}^c = \left\{ \boldsymbol{x} \in [0,1]^{I\cdot J}, \sum_{j\in\mathcal{J}} x_{ij} = 1, \ \forall i \in \mathcal{I} \right\}.$$

Our goal is to ensure a vanishing regret, $\mathcal{R}_{T,\alpha} \to 0$ as $T \to \infty$.

Unfortunately, off-the-shelf algorithms cannot be applied directly to this problem, due to the time-averaging in the argument of $F_\alpha(\cdot)$ that hampers the required (for OCO) over-time decomposition [18]. To tackle this issue, we introduce the linearized proxy function:

$$\Psi_t(\boldsymbol{\theta}, \boldsymbol{x}) = (-F_\alpha)^*(\boldsymbol{\theta}) - \boldsymbol{\theta}^\top \boldsymbol{u}_t(\boldsymbol{x}),$$

where we introduce the dual variable vector $\boldsymbol{\theta}$ that, as will become clear in the sequel, track the long-term fairness metric. In the reformulation, we utilized the Fenchel-conjugate [19, Ch. 4], which for the considered function can be written as:

$$(-F_\alpha)^*(\boldsymbol{\theta}) = \begin{cases} \sum_{i=1}^{I} \frac{\alpha(-\theta_i)^{1-1/\alpha}-1}{1-\alpha}, & \text{for } \alpha \in \mathbb{R}_{\geq 0}\backslash\{1\}, \\ \sum_{i=1}^{I} -1 - \log(-\theta_i), & \text{for } \alpha = 1. \end{cases}$$

With this reformulation at hand, we follow a primal-dual approach over the proxy function so as to bound $\mathcal{R}_{T,\alpha}$. Namely, we first derive a bound for the continuous allocations, and then prove that the discrete ones maintain this bound on expectation.

First, we define the regret using the continuous allocations:

$$\hat{\mathcal{R}}_{T,\alpha} \doteq F_\alpha\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{u}_t(\boldsymbol{x}^\star)\right) - F_\alpha\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t)\right), \quad \hat{\boldsymbol{x}}_t \in \mathcal{X}^c.$$

Using the results from [7], [13], we express this metric as:

$$\hat{\mathcal{R}}_{T,\alpha} \leq \frac{\hat{\mathcal{R}}_T^x}{T} + \frac{\hat{\mathcal{R}}_T^\theta}{T} + \Sigma_T, \tag{1}$$

where $\hat{\mathcal{R}}_T^x, \hat{\mathcal{R}}_T^\theta$ are the primal and dual regrets, and Σ_T is the covariance between utilities $\boldsymbol{u}_t(\boldsymbol{x}^\star)$ and variables $\boldsymbol{\theta}_t$, i.e.,

$$\hat{\mathcal{R}}_T^x = \sum_{t=1}^{T} \Psi_t(\boldsymbol{\theta}_t, \boldsymbol{x}^\star) - \Psi_t(\boldsymbol{\theta}_t, \hat{\boldsymbol{x}}_t),$$
$$\hat{\mathcal{R}}_T^\theta = \sum_{t=1}^{T} \Psi_t(\boldsymbol{\theta}_t, \hat{\boldsymbol{x}}_t) - \Psi_t(\boldsymbol{\theta}, \hat{\boldsymbol{x}}_t),$$
$$\Sigma_T = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\theta}_t^\top \boldsymbol{u}_t(\boldsymbol{x}^\star) - \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{\theta}_t^\top \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{u}_t(\boldsymbol{x}^\star).$$

Σ_T is a residual term that depends on the perturbations and indicates that to achieve vanishing long-term fairness regret, the environment must adhere to perturbation restrictions – see impossibility result in [13]. Hereafter, we denote the growth rate of $\Sigma_T = \mathcal{O}(T^{\omega-1})$ where vanishing regret requires $\omega < 1$.

1) Primal Step: Unfolding the details of the algorithm, we perform the primal step using the FTRL algorithm [20] with entropic regularization due to the multi-simplex constraint $\mathcal{X}^c$:

$$\hat{\boldsymbol{x}}_{t+1} = \arg\min_{\boldsymbol{x}\in\mathcal{X}^c} \left\{ r_{1:t}(\boldsymbol{x}) - \boldsymbol{x}^\top \boldsymbol{g}_{1:t} \right\}, \tag{2}$$

where $r_{1:t}(\boldsymbol{x})$ is the aggregate regularization, and $\boldsymbol{g}_{1:t}$ the total gradient at slot t, with $\boldsymbol{g}_t = \nabla_{\hat{\boldsymbol{x}}_t}\Psi_t(\boldsymbol{\theta}_t, \hat{\boldsymbol{x}}_t) = \boldsymbol{\theta}_t^\top \nabla_{\hat{\boldsymbol{x}}_t}\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t)$. And the proposed regularizer is: $r_{1:t}(\boldsymbol{x}) =$

$$\eta_{1:t}\Big(I\log J + \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} x_{ij}\log x_{ij}\Big), \;\; \eta_{1:t} = \eta\sqrt{\sum_{\tau=1}^{t}\|\boldsymbol{g}_\tau\|_\infty^2} \tag{3}$$

Now, setting $\eta = \min\left\{1/2, \sqrt{2\sqrt{2}/\log J}\right\}$, $r_{1:t}(\boldsymbol{x})$ becomes 1-strongly-convex w.r.t. norm $\|\boldsymbol{x}\|_{(t)} = \|\boldsymbol{x}\|_1\big(\eta_{1:t}/I\big)^{1/2}$ and thus applying [7, Lemma 4.2], the primal regret is:

$$\hat{\mathcal{R}}_T^x \leq \left(\frac{2\sqrt{2}I}{\eta} + \eta I \log J\right)\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_\infty^2}.$$

Moreover, the closed-form solution for (2) is [7, Prop. 2]:

$$\hat{\boldsymbol{x}}_{t+1,Ji+j} = \frac{\exp(\boldsymbol{g}_{1:t,Ji+j}/\eta_{1:t})}{\sum_{j=1}^{J}\exp(\boldsymbol{g}_{1:t,Ji+j}/\eta_{1:t})}. \tag{4}$$

2) Dual Step: For the dual step, we utilize the strong convexity[1] of the proxy function and use the Follow-The-Leader (FTL) algorithm[2] to achieve logarithmic regret [20]:

$$\boldsymbol{\theta}_{t+1} = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}} \sum_{\tau=1}^{t} \Psi_\tau(\boldsymbol{\theta}, \hat{\boldsymbol{x}}_\tau). \tag{5}$$

Observe that $\Psi_t(\boldsymbol{\theta}, \hat{\boldsymbol{x}}_t)$ is 1-strongly convex w.r.t. norm $\|\boldsymbol{\theta}\|_{(t)} = \sqrt{t\phi}\|\boldsymbol{\theta}\|_2$ and using [20, Sec. 3.6] we get:

$$\hat{\mathcal{R}}_T^\theta \leq \frac{\alpha K^2}{2u_{\min}^{\alpha+1}}(1 + \log T),$$

where $\|\nabla_{\boldsymbol{\theta}}\Psi_t(\boldsymbol{\theta}, \hat{\boldsymbol{x}}_t)\|_2 \leq K$. The following new result provides the intrinsic closed-form solution for the dual iteration (5).

Proposition 1. The closed-form solution for $\boldsymbol{\theta}_{t+1}$ in (5), is

$$\boldsymbol{\theta}_{t+1} = -\left(\frac{1}{t}\sum_{\tau=1}^{t}\boldsymbol{u}_\tau(\hat{\boldsymbol{x}}_\tau)\right)^{-\alpha}. \tag{6}$$

This result follows from the definition of $\Psi_t(\boldsymbol{\theta}, \boldsymbol{x})$ and Fenchel conjugate. We omit the details due to lack of space. Interestingly, due to using FTL, the dual regret can be ignored as it has much smaller growth rate of $\mathcal{O}(\log T)$ (instead of $\mathcal{O}(\sqrt{T})$) and also the algorithm does not require the knowledge of $u_{\min}$ and $u_{\max}$. What is more, the closed-form iteration admits an intuitive explanation: the algorithm penalizes the allocations by their accrued utilities, and the degree of this penalty increases with α. When $\alpha = 1$, the iteration resembles the Proportional Fair Scheduler [21], which takes into consideration not only the current utility but the total accrued utility until each time-slot.

Inserting these regret bounds to (1), we obtain:

$$\hat{\mathcal{R}}_{T,\alpha} \leq \frac{1}{T}\left(\frac{2\sqrt{2}I}{\eta} + \eta I \log J\right)\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_\infty^2} + \frac{\alpha I u_{\max}^2}{2u_{\min}^{\alpha+1}}\frac{1+\log T}{T} + \mathcal{O}(T^{\omega-1}).$$

which states that for the continuous allocations we achieve sublinear regret, conditioned on the perturbations satisfying the

[1]The Hessian matrix $\nabla_{\boldsymbol{\theta}}^2\Psi_t(\boldsymbol{\theta}, \hat{\boldsymbol{x}}_t) = \boldsymbol{H}_{I\times I}$ is diagonal with entries $H_{i,i} = (-\theta_i)^{-\frac{\alpha+1}{\alpha}}/\alpha \geq u_{\min}^{\alpha+1}/\alpha$.

[2]Regret guarantee in [7], [13] requires $\theta \in [-u_{\min}^{-\alpha}, -u_{\max}^{-\alpha}]^I$, where $u_{\max} = \max_{1\leq t\leq T}(\|\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t)\|_\infty)$, $u_{\min} = \min_{1\leq t\leq T}(\|\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t)\|_{-\infty})$. Indeed, $\boldsymbol{\theta}_{t+1}$ obtained with (6) adheres to this constraint.

Algorithm 1 Real-time Fair Algorithm

```
Require: $I$, $J$, $\alpha \geq 0$.
1: $\eta = \min\left\{1/2, \sqrt{\sqrt{2}/\log J}\right\}$
2: $\hat{\boldsymbol{x}}_1 \in \mathcal{X}^c$, sample $\boldsymbol{x}_1$ from $\hat{\boldsymbol{x}}_1$        ▷ Initial decision
3: for $t = 1$ to $T$ do
4:     Observe $\boldsymbol{u}_t(\cdot)$, calculate $\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t)$
5:     Compute $\boldsymbol{\theta}_{t+1}$ with (6)        ▷ Calculate dual var
6:     Compute $\hat{\boldsymbol{x}}_{t+1}$ with (4)        ▷ Calculate continuous primal var
7:     Sample $\boldsymbol{x}_{t+1}$ from $\hat{\boldsymbol{x}}_{t+1}$ using inverse transform sampling over each simplex
8: end for
```

impossibility result in [13] (last term). Now, using this result, we derive the regret bound for the discrete decisions $\{\boldsymbol{x}\}_{1:T}$ by taking the expectation of $\hat{\mathcal{R}}_{T,\alpha}$ over $\boldsymbol{x}$. This is formally stated in the next theorem, which we prove in the appendix.

Theorem 1. Alg. 1 ensures the expected regret bound $\mathbb{E}[\mathcal{R}_{T,\alpha}] = \hat{\mathcal{R}}_{T,\alpha}$ for linear utility functions, otherwise

$$\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathbb{E}\left[F_\alpha\left(\frac{1}{T}\sum_{t\in\mathcal{T}}\boldsymbol{u}_t(\boldsymbol{x}^\star)\right) - F_\alpha\left(\frac{1}{T}\sum_{t\in\mathcal{T}}\boldsymbol{u}_t(\boldsymbol{x}_t)\right)\right]$$
$$\leq \frac{1}{T}\left(\frac{\sqrt{2}I}{\eta} + \eta I \log J\right)\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_\infty^2}$$
$$+ \frac{\alpha I u_{\max}^2}{2u_{\min}^{\alpha+1}}\frac{1+\log T}{T} + \frac{\mathcal{O}(T^\omega)}{T} + \frac{L\sqrt{IJ}}{2},$$

where $L \geq \|\nabla_{\hat{\boldsymbol{x}}}\boldsymbol{u}_t(\hat{\boldsymbol{x}})\|_2, \forall t \leq T$ is the Lipschitz constant.

Discussion. There are some important notes in order here. First, observe that the first term in the bound grows adaptive to the gradients, and in the worst case it is $\mathcal{O}(\sqrt{T})$. Hence, when the environment is "easy" (small and smooth gradients), we achieve better results. Also, the second term is negligible in the growth rate (due to FTL and strong convexity), and the last two terms depend on the variation of the environment (beyond our control). We note that the restrictions stated in [13] regarding the third term hold here, since the dual variables are still bounded. The last term is the quantization error [22] due to the curvature of the utility functions, where the curvature of the fairness function does not affect this term. As a result, the growth rate without the quantization error is bounded as $\mathcal{O}(T^{\max\{-\frac{1}{2},\omega-1\}})$.

III. Accounting for Long-term Constraints

As shown in Sec. I, a major concern for vRANs is their energy consumption. A practical solution to this problem is capping the average power of PUs. However, merely limiting the power (e.g., setting fixed horizontal caps) is likely to yield unfair performance across the vBSs and users. Here, we extend our framework to allow the network to handle such long-term, or budget, constraints while ensuring long-term performance fairness. Naturally, this tool can be used to tackle the energy (and other budget-related problems) in a principled fashion.

We introduce the constraint violation vector $\boldsymbol{f}_t(\boldsymbol{x})$ of size M, where our goal is, additionally, to achieve sublinear accumulated constraint violation:

$$\mathcal{V}_T = \left\|\left[\sum_{t=1}^{T}\boldsymbol{f}_t(\boldsymbol{x}_t)\right]_+\right\|_2.$$

In practical applications, the constraints can be set as, e.g., the maximum average power consumption, as shown in Sec. IV-B. We assume that the violation function $\boldsymbol{f}_t(\boldsymbol{x})$ is linear (otherwise, it can be linearized [20]), i.e., $\boldsymbol{f}_t(\boldsymbol{x}) = \boldsymbol{W}_t\boldsymbol{x}$ where $\boldsymbol{W}_t$ is a matrix of size $M \times IJ$, and define the benchmark:

$$\boldsymbol{x}^\star = \arg\max_{\boldsymbol{x}\in\mathcal{X}_T^c} F_\alpha\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{u}_t(\boldsymbol{x})\right), \text{ where}$$
$$\mathcal{X}_T^c = \left\{\boldsymbol{x}\in\mathcal{X}^c \middle| \boldsymbol{f}_t(\boldsymbol{x}_t) \leq \boldsymbol{0}, \forall t \leq T\right\}.$$

Note that the benchmark is inevitably confined to satisfy the constraints at each slot (unlike the learner's decisions), due to the impossibility result in such OCO problems, see [14], [23].

The basis of our algorithm here is a new Lagrange function that tracks the budget via new dual variables. In particular, we use the regularized Lagrangian:

$$\mathcal{L}_t(\boldsymbol{x},\boldsymbol{\lambda}) \doteq r_t(\boldsymbol{x}) - \boldsymbol{g}_t^\top\boldsymbol{x} + \boldsymbol{\lambda}^\top\boldsymbol{W}_t\boldsymbol{x} - q_t(\boldsymbol{\lambda}), \qquad (7)$$

where $\boldsymbol{\lambda} \geq 0$ is the dual vector (not to be confused with $\boldsymbol{\theta}$) that penalizes the constraint violations when $\boldsymbol{W}_t\boldsymbol{x} \geq 0$, and $\boldsymbol{g}_t = \nabla_{\boldsymbol{x}}\Psi_t(\boldsymbol{\theta}_t,\boldsymbol{x})$ as in Sec. II, while $r_t(\boldsymbol{x})$ and $q_t(\boldsymbol{\lambda})$ are strongly convex primal and dual regularizers.

Using a saddle-point structure, we perform minimization in the primal step and maximization in the dual step to converge to an allocation with sublinear constraint violations and vanishing fairness regret. Note that we linearized the proxy function for fairness $\Psi_t(\boldsymbol{\theta}_t,\boldsymbol{x})$ in (7) by inclusion of $\boldsymbol{g}_t$. Due to the geometry of decision spaces, we again use negative entropy function as primal regularizer (3) and a non-proximal quadratic function as a dual regularizer,

$$q_{1:t}(\boldsymbol{\lambda}) = \underbrace{\sigma\max\left\{\sqrt{\sum_{\tau=1}^{t}\|\boldsymbol{W}_\tau\boldsymbol{x}_\tau\|_2^2}, t^\beta\right\}}_{\sigma_{1:t}}\frac{\|\boldsymbol{\lambda}\|_2^2}{2},$$

where $\beta \in [0,1)$ is the parameter of the algorithm that balances the regret and constraint violations. The decision updates are:

$$\hat{\boldsymbol{x}}_{t+1} = \arg\min_{\boldsymbol{x}\in\mathcal{X}^c}\left\{\sum_{\tau=1}^{t} r_\tau(\boldsymbol{x}) - \boldsymbol{g}_\tau^\top\boldsymbol{x} + \boldsymbol{\lambda}_\tau^\top\boldsymbol{W}_\tau\boldsymbol{x}\right\}, \qquad (8)$$

$$\boldsymbol{\lambda}_{t+1} = \arg\min_{\boldsymbol{\lambda}\geq\boldsymbol{0}}\left\{\sum_{\tau=1}^{t} q_\tau(\boldsymbol{\lambda}) - \boldsymbol{\lambda}^\top\boldsymbol{W}_\tau\hat{\boldsymbol{x}}_\tau\right\}. \qquad (9)$$

Note that $\boldsymbol{g}_t$ includes $\boldsymbol{\theta}_t$ and keeps track of long-term fairness while satisfying the budget constraints. Primal (continuous) iteration ($\hat{\boldsymbol{x}}_{t+1}$) can be solved in closed-form by replacing $\boldsymbol{g}$ in (4) with $\boldsymbol{g} - \boldsymbol{W}^\top\boldsymbol{\lambda}$ and the dual iteration can be implemented as $\boldsymbol{\lambda}_{t+1} = \left[\sum_{\tau=1}^{t}\boldsymbol{W}_\tau\hat{\boldsymbol{x}}_\tau/\sigma_{1:t}\right]_+$ following [7, Proposition 1]. We define the constrained primal regret for $\boldsymbol{x}^\star \in \mathcal{X}_T^c$

Algorithm 2 Real-time Fair Alg. with Long-term Constraints

```
Require: $I$, $J$, $\alpha \geq 0$, $\beta \in [0,1)$, $\sigma$
1: $\eta = \min\left\{1/2, \sqrt{\sqrt{2}/\log J}\right\}$
2: $\hat{\boldsymbol{x}}_1 \in \mathcal{X}^c$, sample $\boldsymbol{x}_1$ from $\hat{\boldsymbol{x}}_1$
3: for $t = 1$ to $T$ do
4:     Observe $\boldsymbol{u}_t(\cdot), \boldsymbol{W}_t$, calculate $\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t), \boldsymbol{W}_t\hat{\boldsymbol{x}}_t$
5:     Compute $\boldsymbol{\lambda}_t$ with (9) (or provided closed-form)
6:     Compute $\boldsymbol{\theta}_{t+1}$ with (6)
7:     Compute $\hat{\boldsymbol{x}}_{t+1}$ with (8) (or provided closed-form)
8:     Sample $\boldsymbol{x}_{t+1}$ from $\hat{\boldsymbol{x}}_{t+1}$ using inverse transform sampling
       over each simplex
9: end for
```

as $\mathcal{R}_T^{x_T} = \sum_{t=1}^T \Psi_t(\boldsymbol{\theta}_t, \boldsymbol{x}^\star) - \Psi_t(\boldsymbol{\theta}_t, \boldsymbol{x}_t)$ and provide the following bounds on continuous primal variables $\{\hat{\boldsymbol{x}}\}_{1:T}$:

Lemma 1. Alg. 2 without the sampling step attains the following primal regret and constraint violations:

$$\hat{\mathcal{R}}_T^{x_T} \leq B_T = \frac{2\sqrt{2}I}{\eta}\sqrt{\sum_{t=1}^T \|\boldsymbol{g}_t - \boldsymbol{W}_t^\top \boldsymbol{\lambda}_t\|_\infty^2} + \eta I \log J \sqrt{\sum_{t=1}^T \|\boldsymbol{g}_t - \boldsymbol{W}_t^\top \boldsymbol{\lambda}_t\|_\infty^2 + \sum_{t=1}^T \frac{\|\boldsymbol{W}_t \hat{\boldsymbol{x}}_t\|_2^2}{\sigma_{1:t-1}}},$$

$$\hat{\mathcal{V}}_T \leq \sqrt{2\sigma_{1:T-1}(B_T - \hat{\mathcal{R}}_T^{x_T})}.$$

The next result provides the regret and constraint bounds.

Theorem 2. Alg. 2 ensures the expected long-term fairness regret and constraint violation bounds for linear utility functions:

$$\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathbb{E}\left[F_\alpha\left(\frac{1}{T}\sum_{t=1}^T \boldsymbol{u}_t(\boldsymbol{x}^\star)\right) - F_\alpha\left(\frac{1}{T}\sum_{t=1}^T \boldsymbol{u}_t(\boldsymbol{x}_t)\right)\right]$$
$$\leq \frac{1}{T}\left(\frac{\sqrt{2}I}{\eta} + \eta I \log J\right)\sqrt{\sum_{t=1}^T \|\boldsymbol{g}_t - \boldsymbol{W}_t \boldsymbol{\lambda}_t\|_\infty^2}$$
$$+ \frac{1}{T}\sum_{t=1}^T \frac{\|\boldsymbol{W}_t \hat{\boldsymbol{x}}_t\|_2^2}{\sigma_{1:t-1}} + \frac{\alpha I u_{\max}^2}{2u_{\min}^{\alpha+1}}\frac{1+\log T}{T} + \frac{\mathcal{O}(T^\omega)}{T},$$
$$\mathbb{E}[\mathcal{V}_T] \leq \sqrt{2\sigma_{1:T-1}(B_T - \hat{\mathcal{R}}_T^{x_T})},$$

while otherwise $\mathbb{E}[\mathcal{R}_{T,\alpha}]$ includes an additional quantization error of $L\sqrt{IJ}/2$.

Bound in Theorem 2 requires further analysis of convergence due to $\boldsymbol{\lambda}_t$ in the first term being unbounded. Next, we provide a detailed analysis of convergence and show that Alg. 2 achieves vanishing regret bound and sublinear constraint violation bound. In particular, we obtain the convergence guarantee by bounding the growth rate of $\boldsymbol{\Lambda}_T = \mathcal{O}(T^k)$ where $\boldsymbol{\Lambda}_T = \max\{\|\boldsymbol{\lambda}_t\|_\infty\}_{1:T+1}$ in the next theorem, which we prove in the appendix.

Theorem 3. Alg. 2 without the quantization error ensures

$$\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathcal{O}(T^{\max\{-\frac{1}{4}, \omega-1\}}), \mathbb{E}[\mathcal{V}_T] = \mathcal{O}(T^{\frac{3}{4}}), \text{ if } \beta \leq 1/2,$$
$$\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathcal{O}(T^{\max\{-\frac{\beta}{2}, \omega-1\}}), \mathbb{E}[\mathcal{V}_T] = \mathcal{O}(T^{\frac{1+\beta}{2}}), \text{ o/w.}$$

Discussion. There are some important notes in order here. First, the proposed algorithm guarantees vanishing long-term fairness regret (with the quantization error) and, at the same time, sublinear constraint violations independent of the utility functions' curvature. Fairness regret bound depends on the environment, i.e., the severity of the perturbations (ω) and Lipschitz constant (L). Namely, utility functions can change arbitrarily, as long $\omega < 1$ is satisfied. We define such an environment in Sec. IV-A and validate the theoretical results using Alg. 2. Moreover, we demonstrate the effects of quantization error using a real O-RAN platform in Sec. IV-B.

Second, Alg. 2 (i) does not store the decisions at each time-slot and only uses the accrued utilities and constraint violations to calculate the decision at the next time-slot, and (ii) does not include time costly operations; resulting in $\mathcal{O}(1)$ calculations and storage requirements.

Finally, depending on the value of ω, an operator can balance the fairness regret with constraint violations by setting the parameter β. In particular, if ω is known and within the interval $(1/2, 3/4]$, we can set $\beta = 2 - 2\omega$ and ensure $\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathcal{O}(T^{\omega-1})$, and $\mathbb{E}[\mathcal{V}_T] = \mathcal{O}(T^{\frac{3}{2}-\omega})$. If $\omega \leq 1/2$, we can discard the ω and obtain $\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathcal{O}(T^{-\frac{1}{4}})$ if $\beta \leq 1/2$, $\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathcal{O}(T^{-\frac{\beta}{2}})$ otherwise; and if $\omega > 3/4$, the bound in Theorem 3 becomes $\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathcal{O}(T^{\omega-1}), \mathbb{E}[\mathcal{V}_T] = \mathcal{O}(T^{\frac{3}{4}})$.

Practical considerations. There are some implementation aspects needed to bring Alg. 2 into practice. A requirement of our solution is having estimators of the PU performance in terms of computing time and energy consumption. In our case, we rely on open-source approximators provided in [5], which consider the characteristics of the incoming TB (i.e., size, SNR, and MCS) and the different types of PUs.

In our integration of Alg. 2 in an operational platform, there are a few steps that need to be executed at each time-slot t before line 4. First, the assignment decision should be set in the platform (AAL Broker, explained in the next section). Second, we observe the set of TBs coming from each vBS to be processed during the current time-slot. Third, we compute the following estimations: (i) the processing energy required by PU j to process the TBs of vBS i at time t, denoted by $e_{t,ij}$; (ii) the analogous processing time, denoted by $s_{t,ij}$. Based on this, we compute the lines 4-8 in Alg. 2. Note that the assignment decision for the next time-slot x_{t+1} is computed in line 8. Thus, at $t+1$, x_{t+1} is communicated with no extra delay, which enables real-time operation. Finally, we gather performance measurements from the system in terms of throughput, bit loss, and energy consumption.

IV. Performance Evaluation

We start off by evaluating our solution in different simulation settings where the utility and cost functions are chosen to emulate the real behavior of the system. This evaluation methodology allows us to validate the theoretical findings in terms of fairness regret and constraint violations.

Next, we evaluate our algorithm in a real O-RAN platform where we can measure its performance in terms of energy consumption and throughput. The platform comprises an O-RAN Distributed Unit (DU) and an O-RAN O-Cloud with two PUs: an Nvidia GPU V100 and a CPU Intel Xeon Gold

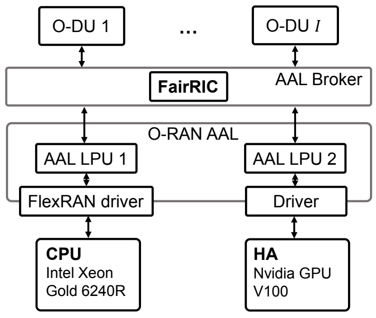

Fig. 2: Scheme of our experimental platform consisting of 2 PUs: a CPU and a GPU. The DUs interface with the AAL Broker, where FairRIC runs in real-time. AAL Broker assigns the load of each individual DU to the LPUs, which abstract the HAs.

6240R, from which 4 cores are dedicated to processing tasks. We implement the O-RAN Acceleration Abstraction Layer (AAL) to abstract the O-Cloud computing resources as Logical Processing Units (LPUs). To implement the AAL, we use Intel DPDK BBDev[3] according to specifications [24]. On top of the AAL, we implemented the *AAL Broker* as detailed in [5]. The AAL Broker operates in real-time and executes our algorithm, assigning the workload of each vBS to one LPU at each TTI.

The data generation and the channel quality of the user are based on traffic traces collected from real BS using [9]. In particular, the traffic traces model the data arrivals from which the DU performs the scheduling decisions creating the TB. Then, we add noise to the TB based on the SNR in the traces and inject the TB into the system. The platform processes the incoming signals using the open-access software libraries, e.g., Intel FlexRAN [25] in the case of the CPU. We measure the energy consumption using the drivers of each PU, i.e., `RAPL` and `nvidia-smi` for the CPU and GPU, respectively. Fig. 2 presents a schematic of our experimental platform.

A. *Simulation Study*

For the simulation study, we model the normalized utility of vBS i and the normalized cost of PU j as:

$$u_{t,i}(\boldsymbol{x}_t) = \sum_{j=1}^{J} u_{t,ij}(\boldsymbol{x}_t) = \sum_{j=1}^{J} x_{t,ij}(1-\pi_{t,j}),$$

$$c_{t,j}(\boldsymbol{x}_t) = \sum_{i=1}^{I} c_{t,ij}(\boldsymbol{x}_t) = \sum_{i=1}^{I} x_{t,ij}e_{t,ij},$$

where $\pi_{t,j}$ denotes the probability of PU j to be over its capacity at time slot t, and $e_{t,ij}$ denotes the energy required to process the load of vBS i using the PU j at time slot t. Let $b_{t,j}$ be the cost constraint (energy budget) for PU j at time t. To ensure the feasibility of the solution for a given $b_{t,j}$ in the simulation study, we set the constraints as a ratio of the energy demand per PU, i.e., $b_{t,j} = R\sum_{i=1}^{I} e_{t,ij}$, where $R \geq 1/J$. We define two simulation scenarios based on how $\pi_{t,j}$ and $e_{t,ij}$ are chosen and we fix in all the cases $I = 20$ vBS and $J = 10$ PUs, and $R = 0.15$:

- *Simulation scenario 1.* The initial values of the parameters ($t = 1$) are drawn randomly from uniform distributions: $\pi_{1,j} \sim \mathcal{U}[0, 0.2)$, $e_{1,ij} \sim \mathcal{U}[0, 1)$. Then, for the next time slots ($t > 1$): $e_{t,ij} = 1 - e_{t-1,ij}$, $\pi_{t,j} = 1 - \pi_{t-1,j}$. This setting is known as the ping-pong scenario and is considered the worst-case scenario in the OCO literature [26].
- *Simulation scenario 2.* We modify simulation scenario 1 to ensure vanishing fairness regret ($w < 1$ in Theorem 2). For that purpose, we randomly select $\lfloor\sqrt{T}\rfloor$ time slots to change $\pi_{t,j}$ instead of changing it at each time slot.

[3]https://doc.dpdk.org/guides/prog guide/bbdev.html

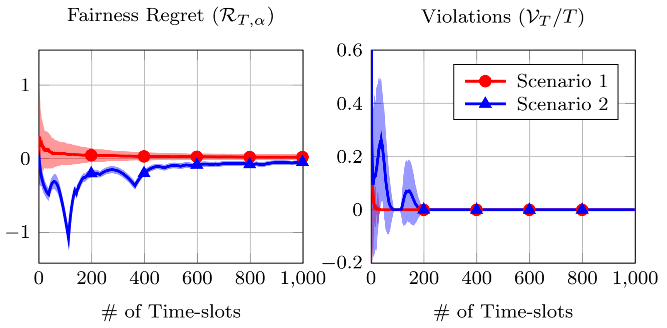

Fig. 3: Evolution of long-term fairness regret (left) and constraint violations (right) within simulations. Mean and ± 1.96 standard deviation (shaded area) of 50 independent runs.

Fig. 3 shows the fairness regret and the time-averaged constraint violations for 50 independent runs of 1000 time-slots, where $\alpha = 1$ and $\beta = 0.75$. The fairness regret of our algorithm converges to a small positive constant value in simulation scenario 1, while vanishing (negative) fairness regret is achieved in simulation scenario 2. These results validate our theoretical findings (see Sec. III). It is proven in [13] that ensuring vanishing fairness regret in scenario 1 is theoretically not possible ($\omega = 1$). Moreover, our algorithm achieves vanishing constraint violation in both cases, while the convergence is faster in simulation scenario 1. When we restrict the environment and ensure vanishing fairness regret (scenario 2), our algorithm needs to balance long-term fairness with the constraint violations. It is evident in Fig. 3 that the constraint violations peak when the fairness regret dips in scenario 2. Hence, our algorithm finds the balance between the two metrics, i.e., fairness and constraint violations, and the negative fairness regret indicates that our algorithm results in a more fair allocation than the best static decision when we restrict the environment (scenario 2).

B. *Experimental Evaluation*

Next, we evaluate the algorithm on an O-RAN-compliant platform. Based on the energy and time estimations defined in Sec. III, $e_{t,ij}$ and $s_{t,ij}$, respectively, the expected processing time and energy consumption of the PU j at time-slot t is given by:

$$r_{t,j} = \sum_{i=1}^{I} x_{t,ij}s_{t,ij} \text{ ms}, \quad c_{t,j} = \sum_{i=1}^{I} x_{t,ij}e_{t,ij} \text{ mJ},$$

respectively. We design the utility and cost functions for this real setting as proxies of the normalized throughput and PU power peak. Specifically, the utility is a piecewise linear function modelling that vBSs assigned to saturated

	Light, x1 Load	Medium, x4 Load	Heavy, x8 Load
Sparse, 1 User	vBS1	vBS2	vBS3
Medium, 4 Users	vBS4	vBS5	vBS6
Dense, 8 Users	vBS7	vBS8	vBS9

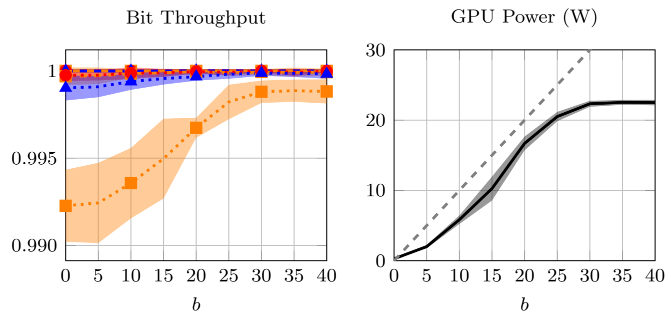

Fig. 4: Table describing the vBS setting in the experiment (top). Normalized throughput per vBS (bottom left) and GPU power consumption (bottom right) as a function of the maximum power constraint, b. FairRIC respects the long-term constraints and distributes the throughput fairly among vBSs. Mean and ± 1.96 standard deviation (shaded area) of 10 independent runs, $\beta = 0, \alpha = 1$.

PUs will experience reduced throughput, similar to [7]. Moreover, the higher the traffic load of a vBS, the higher the probability of bit loss. Considering $\pi_{t,j}$ from Sec. IV-A as $\pi_{t,j} = \max\{0, r_{t,j} - 1\}$ and the vBS load characteristics, the utility of vBS i assigned to PU j is given by $u_{t,ij}(\boldsymbol{x}_t) =$

$$x_{t,ij}(1 - v_{t,i}\pi_{t,j}) = x_{t,ij} \min\{1, 1 - v_{t,i}(r_{t,j} - 1)\},$$

where $v_{t,i}$ is the ratio between the number of TBs coming from vBS i and the total number of TBs at time-slot t. The total utility of vBS i at time-slot t ($u_{t,i}(\boldsymbol{x}_t) = \sum_{j=1}^{J} u_{t,ij}(\boldsymbol{x}_t)$) represents the normalized throughput of vBS i, ranging in $[0, 1]$. In order to ensure that FairRIC respects the long-term power constraints, we model the peak power by dividing the energy consumption of PU with minimum processing time per PU:

$$p_{t,j} = \frac{c_{t,j}}{q_{t,j} \sum_{i=1}^{I} s_{t,ij}},$$

where $q_{t,j}$ is the utilization of the PU j at time-slot t ($q_{t,j}(\boldsymbol{x}_t) = \sum_{i=1}^{I} u_{t,ij}(\boldsymbol{x}_t)/I$). We denote the maximum power consumption of the GPU per time-step as b (W). Note that our algorithm guarantees to satisfy this constraint on average (see Sec. III), i.e., $\sum_{t=1}^{T} p_{t,j}/T \leq b$.

Using the real network traces presented in Fig. 1, we build a heterogeneous environment consisting of 9 vBSs with different numbers of users and traffic loads, described in the Table shown in Fig. 4 (top). The rows represent the number of users, while the columns denote the load amplifier assigned to each user. For instance, *x8 load* implies that the number of bits in each user's data arrival, within the traffic traces, is scaled by a factor of 8. The scaling is performed to emulate higher load vBSs in the future, using the current traffic patterns.

Fig. 4 (bottom) shows the normalized throughput for each vBS (left) and the GPU power consumption (right) as a function of the maximum power constraint and for $T = 10^4$ time-slots. We observe that the average power constraint is always satisfied. When $b = 0$, only the CPU can be used and some vBSs inevitably suffer throughput loss due to the lack of computing

TABLE I: Average vBS throughput and GPU power consumption of algorithms. Average over 10 independent runs ($\pm$ standard deviation).

Algorithm	b (W)	Avg. Thr. (%)	Fairness ($\alpha = 1$)	Avg. GPU Power (W)
FairRIC	5	99.900	-0.0090 (0.0016)	2.003 (0.122)
	15	99.937	-0.0057 (0.0014)	10.256 (0.868)
	∞	99.984	-0.0014 (0.0003)	22.324 (0.219)
Oracle	-	99.998	-0.0002 (0)	27.499 (0.090)
Greedy [5]	-	99.990	-0.0009 (0.0003)	23.091 (0.229)

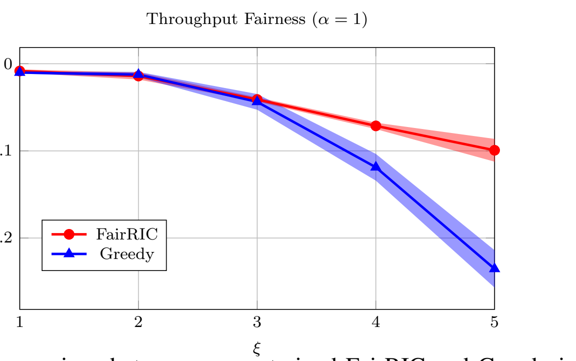

Fig. 5: Comparison between unconstrained FairRIC and Greedy in terms of fairness over normalized throughput as a function of the GPU processing capacity. Mean and ± 1.96 standard deviation (shaded area) of 10 independent runs.

capacity (e.g., vBS9). As the constraint takes higher values, the GPU utilization increases, and therefore the throughput loss diminishes. Note that, even with FairRIC's fair resource allocation, the vBSs with more load inevitably have a higher probability of throughput loss when a PU is saturated.

Next, we evaluate the fairness, aggregated throughput, and power consumption of our algorithm against two benchmarks:

- **Oracle** is the optimal static vBS to PU allocation, found by exhaustive search. The oracle can be computed as we assume stationary traffic loads.
- **Greedy** is inspired by the LPU allocation algorithm in [5]. This algorithm assigns vBS to PU sequentially at each time-slot. For each vBS, the algorithm generates the set of feasible PUs (i.e., the ones that can process the traffic load within the deadline based on an estimation of the processing time). Then, from the set of feasible PUs, the lowest energy consumption is assigned (based on an estimation of the energy consumption of the PUs). This benchmark tries to minimize energy consumption while achieving maximum throughput.

Table I shows the performance of the algorithms for three values of the power constraint: 5 W (minimal GPU usage), 15 W, ∞ (unconstrained). We observe that our algorithm satisfies the power constraints in all the cases. Moreover, when the constraint is set to a very tight value $b = 5$ W (i.e., the use of the GPU, the fast processor, is very restricted), FairRIC allocates the resources in a way that neither the throughput nor the fairness drops significantly. Higher values of b allow FairRIC to provide higher throughput and fairness.

Note that the benchmarks do not allow to set power constraints and therefore can only be compared with the unconstrained version of FairRIC. Oracle provides the highest throughput and fairness but also the higher power consumption due to the higher utilization of the PUs. However, the Oracle solution can only be computed when the traffic loads are known a priori. We also observe no statistical differences between Greedy and unconstrained FairRIC. The reason is that, without power constraints, the computing resources are enough to process the load, resulting in normalized throughput close to 1 and consequently high fairness for all vBSs. The results in Table I indicates that the quantization error does not affect the performance of FairRIC.

Now, we evaluate these algorithms when the computing resources are constrained. For that purpose, we decrease the processing power of the GPU by a factor ξ. That is, $\xi = 1$ implies normal operation, while $\xi = 2$ doubles the processing time. Fig. 5 shows the α-fairness of the throughput with $\alpha = 1$ for different values of GPU speed. Values of throughput fairness lower than zero indicate throughput losses at the vBSs. We observe that for $\xi > 3$, FairRIC provides higher throughput fairness compared to the Greedy benchmark. In other words, when the computing resources are scarce, FairRIC fairly distributes the losses across the different vBSs.

V. Literature Review

Resource allocation in mobile networks. There are many previous works addressing resource management problems in mobile networks. For instance, some rely on analytical functions to characterize the performance of the network (e.g., [27], [28]). Unfortunately, these solutions require some input parameters that are often unknown in vRANs. Some others propose techniques that adapt to network conditions and user demands (e.g., [29], [30]). For instance, Bayesian learning is used for the optimization of video analytics [29], and Reinforcement Learning is applied for spectrum management and wireless scheduling [30], among others. These approaches, however, either have high computing overhead such as the ones relying on Bayesian learning, or do not provide any guarantees in terms of convergence or regret.

The most closely related works to this paper address the computing resource assignment in O-RAN systems with processor pooling [5], [6], [7]. While some of them tackle the problem in near-RT or non-RT, the granularity of the decision-making in these approaches may be very coarse to adapt to the intrinsic burstiness of the traffic [5], [8]. To alleviate this issue, the authors in [5] propose a framework to assign the computational resources of the O-Cloud in real time. Nevertheless, the proposed algorithm in this framework is a greedy heuristic, which lacks performance guarantees. Furthermore, fairness among vBSs or users is not addressed in this work. In contrast, we propose a framework with long-term fairness and constraint violation guarantees utilizing OCO theory. Finally, antenna pooling is studied in [31] and orthogonal to this work.

Online learning and fairness. Fairness is a crucial metric in resource management, with extensive applications in cloud computing [32], communication systems [16], and other domains [33]. Recent studies have explored max-min throughput fairness in RANs through spectrum management [34], fair allocation of computing capacity to vRAN functions and edge services [35], and cost-fairness in multi-tenant O-RANs [36]. However, these works often overlook the dynamic nature of vRANs and fail to provide fairness guarantees, making the problem theoretically challenging. Previous research has considered slot fairness, addressing fairness independently in each decision round [37]. Although simpler, this approach leads to a higher price of fairness [38]. More advanced methods aim for long-term fairness [13], or enforce fairness across multiple time-scales [39]. There are some works that define fairness function as a regularizer [40] or as a constraint [41]. Recent long-term fairness studies typically assume known or non-adversarial utility functions [39], [42], or compare with easier benchmarks [43]. Our framework, however, drops these assumptions and extends the work of [7] by respecting long-term constraints and allowing discrete allocations.

Producing discrete decisions within the OCO framework is not a new idea and studied extensively by [44], [45], [46], where the proposed algorithms used Follow-The-Perturbed-Leader. However, none of these works considered fairness or constraints. Constrained OCO framework utilizes penalty functions [47], [48] or Lazy Lagrangians [14], [49]. Some other works utilize Lyapunov optimization to meet the stochastic constraints at each time-slot instead of considering an average constraint [50], [51].

Our approach relies on the seminal FTRL framework that receives increasing attention [20]. We extend FTRL to accommodate budget constraints and ensure low computation and memory requirements while producing discrete assignment decisions; as deemed necessary for the considered real-time decision problem in O-RAN.

VI. Conclusion

The development of computing resource assignment strategies becomes crucial in O-RAN networks where several vBSs share a common O-Cloud. We address this problem with FairRIC, a novel resource assignment algorithm that operates in real-time to handle the intrinsic burstiness of the traffic. FairRIC relies on OCO theory, extending state-of-the-art approaches on long-term fairness and constrained optimization, and allowing discrete decisions. Our method offers an intrinsic closed-form iteration streamlining the computation process and allowing real-time operation. Moreover, our algorithm has guarantees in terms of fairness among the vBSs while adhering to long-term constraints in terms of energy over the entire operation horizon. We validate our approach via simulations, and experimentally in an O-RAN platform.

Acknowledgment

This work has been supported by (i) the European Commission through Grant No. 101139270 (ORIGAMI), and Grant No. 101192462 (Flecon-6G), (ii) the National Growth Fund through the Dutch 6G flagship project "Future Network Services", and (iii) NextGeneration EU through UNICO I+D Grant No. TSI-063000-2021 (OPEN6G).

Appendix

A. Proof of Theorem 1

Proof. Define the regret on discrete decisions $\mathcal{R}_T^x \doteq \sum_{t=1}^T \Psi_t(\boldsymbol{\theta}_t, \boldsymbol{x}^\star) - \Psi_t(\boldsymbol{\theta}_t, \boldsymbol{x}_t)$. Dual variables $\{\boldsymbol{\theta}\}_{1:T}$ are calculated using the continuous variables $\{\hat{\boldsymbol{x}}\}_{1:T}$ and does not include randomization over decisions $\{\boldsymbol{x}\}_{1:T}$. Therefore, we can write

$$\mathbb{E}[\mathcal{R}_{T,\alpha}] = \mathbb{E}\left[F_\alpha\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{u}_t(\boldsymbol{x}^\star)\right) - F_\alpha\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{u}_t(\boldsymbol{x}_t)\right)\right]$$
$$\leq \frac{\mathbb{E}[\mathcal{R}_T^x]}{T} + \frac{\hat{\mathcal{R}}_T^\theta}{T} + \Sigma_T.$$

For linear utility functions, it is sufficient to show that

$$\mathbb{E}[\mathcal{R}_T^x] = \sum_{t=1}^{T}\mathbb{E}[\Psi_t(\boldsymbol{\theta}_t,\boldsymbol{x}^\star) - \Psi_t(\boldsymbol{\theta}_t,\boldsymbol{x}_t)] = \sum_{t=1}^{T}\mathbb{E}[\boldsymbol{g}_t^\top(\boldsymbol{x}^\star - \boldsymbol{x}_t)]$$
$$\overset{(a)}{=} \sum_{t=1}^{T}\boldsymbol{g}_t^\top(\boldsymbol{x}^\star - \mathbb{E}[\boldsymbol{x}_t]) \overset{(b)}{=} \sum_{t=1}^{T}\boldsymbol{g}_t(\boldsymbol{x}^\star - \hat{\boldsymbol{x}}_t) = \hat{\mathcal{R}}_T^x,$$

where in (b) we used the fact that inverse transform sampling is unbiased ($\mathbb{E}[\boldsymbol{x}_t] = \hat{\boldsymbol{x}}_t$) [52], [53]. For concave and L-Lipschitz continuous utility functions, (a) is not true since $\boldsymbol{g}_t = \nabla_{\hat{\boldsymbol{x}}_t}\Psi_t(\boldsymbol{\theta}_t,\hat{\boldsymbol{x}}_t) = \boldsymbol{\theta}_t^\top\nabla_{\hat{\boldsymbol{x}}_t}\boldsymbol{u}_t(\hat{\boldsymbol{x}}_t)$ depends on $\hat{\boldsymbol{x}}_t$, and $\mathbb{E}[\mathcal{R}_T^x] \leq \hat{\mathcal{R}}_T^x + L\sqrt{IJ}/2$ from [22, Theorem 1]. Bound follows from the proof of [7, Lemma 4.2]:

$$\hat{\mathcal{R}}_T^x \leq \left(\frac{2\sqrt{2}I}{\eta} + \eta I\log J\right)\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_\infty^2}.$$

☐

B. Proof of Lemma 1

Proof. We use the rationale of [14, Theorem 1] as starting point, but we have a different setting. First, we use negative entropy as the primal regularizer, which is non-proximal and we cannot directly apply the strategy of using Be-the-Leader [26, Lemma 3.1] to bound the regret with respect to prescient actions. Instead, we derive the bound by using regret bounds from [7] (B_T^x, B_T^r). We follow the same approach as [14, Theorem 1] until Be-the-Leader step, and bound the regret as:

$$-\sum_{t=1}^{T}\boldsymbol{g}_t^\top\hat{\boldsymbol{x}}_t \leq B_T^\lambda + \sum_{t=1}^{T}q_t(\boldsymbol{\lambda}_t) + \sum_{t=1}^{T}\mathcal{L}_t(\hat{\boldsymbol{x}}_t,\boldsymbol{\lambda}_t)$$
$$\leq B_T^\lambda + \sum_{t=1}^{T}q_t(\boldsymbol{\lambda}_t) + \sum_{t=1}^{T}\mathcal{L}_t(\boldsymbol{x}^\star,\boldsymbol{\lambda}_t)$$
$$+ \underbrace{\frac{2\sqrt{2}I}{\eta}\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t - \boldsymbol{W}_t^\top\boldsymbol{\lambda}_t\|_\infty^2}}_{B_T^x}$$
$$= B_T^\lambda + B_T^x + \sum_{t=1}^{T}r_t(\boldsymbol{x}^\star) - \boldsymbol{g}_t^\top\boldsymbol{x}^\star + \boldsymbol{\lambda}_t^\top\boldsymbol{W}_t\boldsymbol{x}^\star$$
$$\overset{(a)}{\leq} B_T^\lambda + B_T^x + \underbrace{\eta I\log J\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t - \boldsymbol{W}_t^\top\boldsymbol{\lambda}_t\|_\infty^2}}_{B_T^r} - \sum_{t=1}^{T}\boldsymbol{g}_t^\top\boldsymbol{x}^\star,$$

where (a) follows from $\boldsymbol{W}_t\boldsymbol{x}^\star = \boldsymbol{f}_t(\boldsymbol{x}^\star) \leq 0, \forall t \leq T$, and B_T^x, B_T^r follows from [7, Lemma 4.2]. Rearranging, we obtain the regret bound. In a similar fashion, we bound the constraint violations as:

$$\boldsymbol{\lambda}^\top\sum_{t=1}^{T}\boldsymbol{W}_t\hat{\boldsymbol{x}}_t - q_{1:T-1}(\boldsymbol{\lambda}) \leq B_T^\lambda + \sum_{t=1}^{T}q_t(\boldsymbol{\lambda}_t) + \boldsymbol{g}_t^\top\hat{\boldsymbol{x}}_t$$
$$+ \sum_{t=1}^{T}\underbrace{r_t(\hat{\boldsymbol{x}}_t) - \boldsymbol{g}_t^\top\hat{\boldsymbol{x}}_t + \boldsymbol{\lambda}_t^\top\boldsymbol{W}_t\hat{\boldsymbol{x}}_t - q_t(\boldsymbol{\lambda}_t)}_{\mathcal{L}_t(\hat{\boldsymbol{x}}_t,\boldsymbol{\lambda}_t)}$$
$$\leq B_T^\lambda + \sum_{t=1}^{T}q_t(\boldsymbol{\lambda}_t) + \boldsymbol{g}_t^\top\hat{\boldsymbol{x}}_t + \mathcal{L}_t(\boldsymbol{x}^\star,\boldsymbol{\lambda}_t) + B_T^x$$
$$\leq B_T^\lambda + B_T^x + \sum_{t=1}^{T}r_t(\boldsymbol{x}^\star) + \boldsymbol{\lambda}_t^\top\boldsymbol{W}_t\boldsymbol{x}^\star - \underbrace{\sum_{t=1}^{T}\boldsymbol{g}_t^\top(\boldsymbol{x}^\star - \hat{\boldsymbol{x}}_t)}_{\hat{L}_T^x}$$

Using the fact that $\boldsymbol{W}_t\boldsymbol{x}^\star = \boldsymbol{f}_t(\boldsymbol{x}^\star) \leq 0, \forall t \leq T$, and $\hat{L}_T^x \geq \hat{\mathcal{R}}_T^x$ from [20], we obtain the constraint bound. ☐

C. Proof of Theorem 2

Proof. Regret is bounded same as Theorem 1, and $\mathbb{E}[\mathcal{V}_T]$ is bounded using the proof of Lemma 1 as:

$$\mathbb{E}[\boldsymbol{\lambda}^\top\sum_{t=1}^{T}\boldsymbol{W}_t\boldsymbol{x}_t - \frac{\sigma_{1:T-1}}{2}\|\boldsymbol{\lambda}\|_2^2] \leq \mathbb{E}[B_T^x + B_T^r + B_T^\lambda - L_T^x],$$
$$\boldsymbol{\lambda}^\top\sum_{t=1}^{T}\boldsymbol{W}_t\mathbb{E}[\boldsymbol{x}_t] - \frac{\sigma_{1:T-1}}{2}\|\boldsymbol{\lambda}\|_2^2 \leq B_T^x + B_T^r + B_T^\lambda - \mathbb{E}[L_T^x],$$
$$\boldsymbol{\lambda}^\top\sum_{t=1}^{T}\boldsymbol{W}_t\hat{\boldsymbol{x}}_t - \frac{\sigma_{1:T-1}}{2}\|\boldsymbol{\lambda}\|_2^2 \leq B_T^x + B_T^r + B_T^\lambda - \hat{L}_T^x.$$

☐

D. Proof of Theorem 3

Proof. From the closed-form solution for $\boldsymbol{\lambda}$ and $\|\cdot\|_\infty \leq \|\cdot\|_2$, and triangle inequality,

$$\|\boldsymbol{\lambda}_{T+1}\|_\infty = \left\|\left[\frac{\sum_{t=1}^{T}\boldsymbol{W}_t\hat{\boldsymbol{x}}_t}{\sigma_{1:t}}\right]_+\right\|_\infty \leq \frac{\hat{\mathcal{V}}_T}{\sigma_{1:T}}, \tag{10}$$

$$\sqrt{\sum_{t=1}^{T}\|\boldsymbol{g}_t - \boldsymbol{W}_t\boldsymbol{\lambda}_t\|_\infty^2} \leq (\check{\boldsymbol{g}} + \check{\boldsymbol{W}}\check{\boldsymbol{\lambda}}_T)\sqrt{T}, \tag{11}$$

where $\check{\boldsymbol{g}} = \max\{\|\boldsymbol{g}_t\|_\infty\}$, $\check{\boldsymbol{W}} = \max\{\|\boldsymbol{W}_t\|_\infty\}$ are constants, and $\check{\boldsymbol{\lambda}}_T = \max\{\|\boldsymbol{\lambda}_t\|_\infty\}$ Expanding $\hat{\mathcal{V}}_T$ in (10) we have:

$$\|\boldsymbol{\lambda}_{T+1}\|_\infty \leq \frac{\sqrt{2\sigma_{1:T-1}(B_T - \hat{\mathcal{R}}_T^{x_T})}}{\sigma_{1:T}} \leq \sqrt{\frac{2(B_T - \hat{\mathcal{R}}_T^{x_T})}{\sigma_{1:T}}} \tag{12}$$

Following [14, Theorem 2], we can bound

$$\frac{1}{\sigma_{1:T}} = \mathcal{O}(T^\gamma), \qquad \gamma = \min\{-\beta, -1/2\} \leq 0, \tag{13}$$
$$B_T^\lambda = \mathcal{O}(T^\zeta), \qquad \zeta = \min\{1-\beta, 1/2\} \leq 1. \tag{14}$$

Finally, inserting these values into B_T in (12), we have $k = (1+\gamma)/2$ and $B_T = \mathcal{O}(T^{1+\frac{\gamma}{2}})$ and $\hat{\mathcal{V}}_T = \mathcal{O}(T^{\frac{1-\gamma}{2}})$. ☐

References

[1] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in o-ran for data-driven nextg cellular networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, 2021.

[2] J. A. Ayala-Romero, I. Khalid, A. Garcia-Saavedra, X. Costa-Perez, and G. Iosifidis, "Experimental evaluation of power consumption in virtualized base stations," in *Proc. of IEEE ICC*, pp. 1–6, 2021.

[3] GSMA Association, "5g energy efficiencies: Green is the new black," *White Paper*, 2020.

[4] China Mobile Limited, "2021 sustainability report," *White Paper*, 2021.

[5] L. L. Schiavo, G. Garcia-Aviles, A. Garcia-Saavedra, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "Cloudric: Open radio access network (o-ran) virtualization with shared heterogeneous computing," in *Proc. of ACM MOBICOM*, pp. 558–572, 2024.

[6] J. A. Ayala-Romero, L. Lo Schiavo, A. Garcia-Saavedra, and X. Costa-Perez, "Mean-field multi-agent contextual bandit for energy-efficient resource allocation in vrans," in *Proc. of IEEE INFOCOM*, pp. 911–920, 2024.

[7] F. Aslan, G. Iosifidis, J. A. Ayala-Romero, A. Garcia-Saavedra, and X. Costa-Perez, "Fair resource allocation in virtualized o-ran platforms," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 8, no. 1, pp. 1–34, 2024.

[8] X. Foukas and B. Radunovic, "Concordia: Teaching the 5g vran to share compute," in *Proc. of ACM SIGCOMM*, pp. 580–596, 2021.

[9] R. Falkenberg and C. Wietfeld, "FALCON: An accurate real-time monitor for client-based mobile network data analytics," in *Proc. of IEEE GLOBECOM*, pp. 1–7, 2019.

[10] E. Hazan, "Introduction to online convex optimization," *Foundations and Trends in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.

[11] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[12] S. Shalev-Shwartz and Y. Singer, "A primal-dual perspective of online learning algorithms," *Machine Learning*, vol. 69, no. 2-3, pp. 115–142, 2007.

[13] T. Si Salem, G. Iosifidis, and G. Neglia, "Enabling long-term fairness in dynamic resource allocation," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 6, no. 3, pp. 1–36, 2022.

[14] D. Anderson, G. Iosifidis, and D. J. Leith, "Lazy lagrangians for optimistic learning with budget constraints," *IEEE/ACM Transactions on Networking*, vol. 31, no. 5, pp. 1935–1949, 2023.

[15] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proc. of ICML*, pp. 928–936, 2003.

[16] E. Altman, K. Avrachenkov, and A. Garnaev, "Generalized α-fair resource allocation in wireless networks," in *Proc. of IEEE CDC*, pp. 2414–2419, 2008.

[17] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556–567, 2000.

[18] S. Agrawal and N. Devanur, "Bandits with concave rewards and convex knapsacks," in *Proc. of ACM EC*, pp. 989–1006, 2014.

[19] A. Beck, "First-order methods in optimization," *MOS-SIAM Series on Optimization*, 2017.

[20] H. B. McMahan, "A survey of algorithms and analysis for adaptive online learning," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1–50, 2017.

[21] T. Ramjee, T. Bu, and L. Li, "Generalized proportional fair scheduling in third generation wireless data networks," in *Proc. of IEEE INFOCOM*, pp. 1–12, 2006.

[22] A. Lesage-Landry, J. A. Taylor, and D. S. Callaway, "Online convex optimization with binary constraints," *IEEE Transactions on Automatic Control*, vol. 66, no. 12, pp. 6164–6170, 2021.

[23] S. Mannor, J. N. Tsitsiklis, and J. Y. Yu, "Online learning with sample path constraints," *Journal of Machine Learning Research*, vol. 10, pp. 569–590, 2009.

[24] O-RAN ALLIANCE, "O-ran acceleration abstraction layer general aspects and principles. o-ran.wg6.aal-ganp-v01.00," 2021.

[25] Intel, "Flexran lte & 5g nr fec software development kits," 2019.

[26] N. Cesa-Bianchi and G. Lugosi, *Prediction, learning, and games*. Cambridge university press, 2006.

[27] D. Bega, A. Banchs, M. Gramaglia, X. Costa-Pérez, and P. Rost, "Computation-aware scheduling in virtualized radio access networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 12, pp. 7993–8006, 2018.

[28] K. Wang, X. Yu, W. Lin, Z. Deng, and X. Liu, "Computing-aware scheduling in mobile edge computing system," *Wireless Networks*, vol. 27, pp. 4229–4245, 2021.

[29] A. Galanopoulos, J. A. Ayala-Romero, D. J. Leith, and G. Iosifidis, "Automl for video analytics with edge computing," in *Proc. of IEEE INFOCOM*, pp. 1–10, 2021.

[30] J. J. Alcaraz, J. A. Ayala-Romero, J. Vales-Alonso, and F. Losilla-López, "Online rl for adaptive interference coordination," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 10, pp. 1–24, 2020.

[31] J. Mendes, X. Jiao, A. Garcia-Saavedra, F. Huici, and I. Moerman, "Cellular access multi-tenancy through small cell virtualization and common rf front-end sharing," in *Proceedings of the 11th Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*, pp. 35–42, 2017.

[32] T. Bonald and J. W. Roberts, "Multi-resource fairness: Objectives, algorithms and performance," in *Proc. of ACM Sigmetrics*, pp. 31–42, 2015.

[33] D. Nace and M. Pioro, "Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 5–17, 2008.

[34] F. Mehmeti and T. F. La Porta, "Reducing the cost of consistency: Performance improvements in next generation cellular networks with optimal resource reallocation," *IEEE Transactions on Mobile Computing*, vol. 21, no. 7, pp. 2546–2565, 2022.

[35] S. Tripathi, C. Puligheddu, S. Pramanik, A. Garcia-Saavedra, and C. F. Chiasserini, "Fair and scalable orchestration of network and compute resources for virtual edge services," *IEEE Transactions on Mobile Computing*, vol. 23, pp. 2202–2218, 2023.

[36] S. Mondal and M. Ruffini, "Fairness guaranteed and auction-based x-haul and cloud resource allocation in multi-tenant o-rans," *IEEE Transactions on Communications*, vol. 71, no. 6, pp. 3452–3468, 2023.

[37] S. R. Sinclair, G. Jain, S. Banerjee, and C. L. Yu, "Sequential fair allocation: Achieving the optimal envy-efficiency tradeoff curve," *Operations Research*, vol. 71, no. 5, pp. 1689–1705, 2022.

[38] D. Bertsimas, V. F. Farias, and N. Trichakis, "The price of fairness," *Operations Research*, vol. 59, no. 1, pp. 17–31, 2011.

[39] E. Altman, K. Avrachenkov, and S. Ramanath, "Multiscale fairness and its application to resource allocation in wireless networks," *Computer Communications*, vol. 35, no. 7, pp. 820–828, 2012.

[40] S. Balseiro, H. Lu, and V. Mirrokni, "Regularized online allocation problems: Fairness and beyond," in *Proc. of ICML*, pp. 630–639, PMLR, 2021.

[41] X. Wu and B. Li, "Achieving regular and fair learning in combinatorial multi-armed bandit," in *Proc. of IEEE INFOCOM*, pp. 361–370, 2024.

[42] L. Liao, Y. Gao, and C. Kroer, "Nonstationary dual averaging and online fair allocation," in *Proc. of NeurIPS*, pp. 37159–37172, 2022.

[43] A. Sinha, A. Joshi, R. Bhattacharjee, C. Musco, and M. Hajiesmaili, "No-regret algorithms for fair resource allocation," *Proc. of NeurIPS*, vol. 36, pp. 48083–48109, 2024.

[44] A. Kalai and S. Vempala, "Efficient algorithms for online decision problems," *Journal of Computer and System Sciences*, vol. 71, no. 3, pp. 291–307, 2005.

[45] A. Cohen and T. Hazan, "Following the pertubed leader for online structured learning," in *Proc. of ICML*, pp. 1034–1042, 2015.

[46] N. Mhaisen, A. Sinha, G. Paschos, and G. Iosifidis, "Optimistic no-regret algorithms for discrete caching," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 6, no. 3, pp. 1–28, 2022.

[47] W. I. Zangwill, "Nonlinear programming via penalty functions," *Management Science*, vol. 13, no. 5, pp. 344–358, 1967.

[48] D. J. Leith and G. Iosifidis, "Penalized ftrl with time-varying constraints," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 311–326, Springer, 2022.

[49] M. Mahdavi, R. Jin, and T. Yang, "Trading regret for efficiency: online convex optimization with long term constraints," *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 2503–2528, 2012.

[50] H. Yu, M. J. Neely, and X. Wei, "Online convex optimization with stochastic constraints," *Proc. NeurIPS*, vol. 30, pp. 1–11, 2017.

[51] H. Yu and M. J. Neely, "Learning-aided optimization for energy-harvesting devices with outdated state information," *IEEE/ACM Transactions on Networking*, vol. 27, no. 4, pp. 1501–1514, 2019.

[52] L. Devroye, "Nonuniform random variate generation," *Handbooks in operations research and management science*, vol. 13, pp. 83–121, 2006.

[53] S. M. Ross, *Simulation*. academic press, 2022.