Learning with Side Information: Elastic Multi-resource Control for the Open RAN

Xiaoxi Zhang, *Member, IEEE,* Jinhang Zuo, *Member, IEEE,* Zhe Huang, *Member, IEEE,* Zhi Zhou, *Member, IEEE,* Xu Chen, *Senior Member, IEEE,* and Carlee Joe-Wong, *Senior Member, IEEE*

***Abstract*—The open radio access network (O-RAN) architecture provides enhanced opportunities for integrating machine learning in 5G/6G resource management by decomposing RAN functionalities. Yet, generic learning mechanisms either do not fully exploit the disaggregated non-real-time and near-real-time RAN controllers or ignore the potential elasticity of application demands, another degree of freedom in managing RAN resources. We introduce a two-timescale framework aimed at optimizing users' long-term total QoS. Rather than reactive resource allocation, our approach proactively modifies multi-resource user demands using congestion indicators, prior to enforcing any allocation rules. Addressing the issue of insufficient user feedback on individual resource utilities, we employ a bandit-feedback version of the combinatorial multi-armed bandit framework to deduce resource-specific signals. Also, to compensate for insufficient and infrequent feedback, we've developed an algorithm that gleans side information from live network traffic to refine predictions on user resource sensitivities. This streamlines the algorithm's optimality convergence and leverages the two-tier O-RAN controller structure. We validate our algorithms' efficacy through analysis and 5G usage experiments, revealing our proposed method improves application utility by 13-60%, throughput by 8-19%, and reduces latency by 10-18%.**

***Index Terms*—Open Radio Access Network (Open RAN), elastic multi-resource management, online learning**

I. Introduction

The problem of managing resources such as spectrum blocks in wireless networks so as to optimize users' quality-of-service (QoS) has been examined over the past few decades [1], [2]. However, emerging applications have heterogeneous performance needs [3], e.g., enhanced mobile broadband (eMBB) use cases require high peak data rates, while ultra-reliable low latency communication (URLLC) traffic prioritizes ultra-high reliability and low latency [4]. As the key component of 5G/6G networks, the Open Radio Access Network (O-RAN) [5] faces the challenge of fulfilling heterogeneous, elastic, and multi-dimensional resource demands in a multi-tenant RAN system, e.g., allocating combinations of spectrum, power, and cell sites [6], [7], or RAN slices of radio spectrum, backhaul link bandwidth, and cache storage [8].

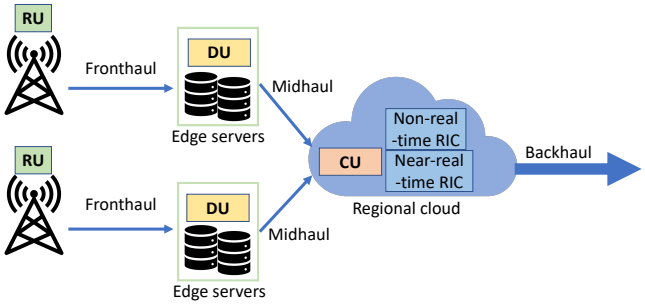


Fig. 1: Disaggregated O-RAN architecture with non-real-time and near-real-time RICs

As shown in Figure 1, O-RAN [9]–[11] decouples baseband unit (BBU) functions from Radio units (RUs) at each cell site into distributed units (DUs) for data plane operations and a centralized unit (CU) for global control. This hierarchical architecture further includes near-real-time and non-real-time RAN Intelligent Controllers (RICs) running at different timescales [12] interacted with the CU. It is therefore natural to ask: whether and how to exploit the elasticity of user demands and the two-tier architecture logic for multi-resource management? Recent machine learning (ML) advances can predict users' QoS needs to enable flexible resource allocation. For instance, schedulers may learn to allocate fewer but consistent cellular blocks to a smart factory application, which does not need high peak data rates like ultra-high definition video streaming. However, it is not obvious how to integrate ML with the two-tier O-RAN structure. Moreover, existing learning algorithms *rely on either large-scale training data-sets [13] or fine-grained instantaneous online feedback* (e.g., multi-armed bandits [14]). We address these shortcomings through: **1)** an outer-loop proactively reshaping elastic user demands by learned optimal control signals at a longer timescale, using only limited and possibly delayed application feedback; and **2)** an inner-loop transforming in-direct knowledge from real-time network data into side information for refining the outer-loop learning parameters. These control signals are designed to *induce congestion-aware traffic demands* from application APIs in order to improve the total QoS performance achieved by existing resource allocation schemes. Using control signals to regulate users can also be found in other scenarios such

Xiaoxi Zhang, Zhi Zhou, and Xu Chen are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, Guangdong 510275 China. Email: {zhangxx89,zhouzhi9,chenxu35}@mail.sysu.edu.cn.

Jinhang Zuo is with the Manning College of Information and Computer Sciences, University of Massachusetts–Amherst, Amherst, MA, US, and the Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA, US. Email:jhzuo@cs.umass.edu

Carlee Joe-Wong is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 US. Email:cjoewong@andrew.cmu.edu.

Zhe Huang is with Google, Princeton, New Jersey, US. Email:felix.zhe.huang@gmail.com

Jinhang Zuo is the corresponding author.

Manuscript received on January 7, 2023; revised on June 7, 2023 and September 2, 2023.

as demand response for smart grids [15] and dynamic cloud VM pricing [16], although they do not exploit two-tier control loops.

To this end, this work **is among the first learning approaches that can proactively control multi-resources for O-RAN systems under coarse-grained hindsight feedback and offer theoretical guarantees.** By "coarse-grained", we mean that the feedback is a single scalar of each user's aggregated QoS over the multiple types of acquired resources. While imposing less of a burden on the user application APIs, this kind of feedback impedes the learning convergence. We leverage our inner-loop to overcome this problem. Thus, unlike existing mechanisms for RAN resource scheduling that do not fully exploit the O-RAN promoted disaggregated architecture [10], [11], and unlike traditional online learning algorithms for resource allocation that require fine-grained user feedback [17], [18], we design a new learning algorithm with theoretical guarantees that fully exploits the two-tier nature of the O-RAN intelligent controllers to learn from coarse-grained feedback and reshape elastic user demands according to available resources. Other networked systems may also benefit from this approach (Section VI-A).

In introducing this control framework, we encounter several **research challenges**:

Uncertain and diverse application/user behaviors are hard to predict with the current RAN control logic, given that application APIs may only be able to provide limited feedback at sparse time points, e.g., few users will upload scores indicating their satisfaction with network QoS.

Ultra-low latency requirements for real-time resource management prohibit sophisticated allocation algorithms with high time-complexity, e.g., solving a combinatorial admission control problem. We must allocate resource combinations without knowing users' timely QoS feedbacks.

Elastic and multi-resource demands motivate proactive regulation of user demands when the network encounters congestion. However, multiple resources (e.g., spectrum/bandwidth on different links) greatly expand the space of demands encountered and control signals offered.

Online learning [14], [19] provides efficient solutions to optimize actions by learning through sequential feedback. We introduce new learning intelligence to address our research challenges, allowing us to make the following **technical contributions**.

First, we design a two-tier closed-loop control framework, which magnifies the value of the two-timescale O-RAN intelligent controllers. To reduce the complexity of solving elastic and multi-resource allocation, we propose an outer-loop online learning routine to first induce congestion-aware demands through control signals. In the inner loop, a gradient-based strategy (Algorithm 2) is designed to extract side information for improving our outer-loop learning ability. Lightweight allocation rules are thus integrated into the inner-loop as well for final resource provisioning (Section V), removing the need to create a clean-slate reinvention of O-RAN resource management in order to implement adaptive resource allocation.

Second, our C-MAB-based outer-loop can learn multi-dimensional decisions from single-dimensional feedback. It therefore uses much weaker assumptions than linear bandits [18] or semi-bandits C-MAB [20], [21], which require one feedback value for the control signal of each type of resource. We first reshape our original solution space and then parameterize the unknown QoS objective function in order to optimize it. Since we construct a one-hot format of the control signal for each type of resource, our linear parameterization admits an arbitrary QoS function with respect to (w.r.t.) the allocation and the control signal of any given resource.

Third, our learning scheme is a novel variant of contextual MAB algorithms. Unlike prior works [22]–[25], ours requires neither explicit contexts as parameters of the reward functions nor a context-reward relationship known a priori. Inspired by the widely adopted optimizers of neural network learning, we construct gradient-based contexts to improve our estimates of the uncertain reward values, i.e., to shrink the confidence region of the learning parameters which can reduce the sub-optimality of using the outer-loop control signals.

Finally, we validate our algorithm performance through both theoretical analysis and simulations on 5G data usage traces (Section V). Our analysis proves that the performance gap against the optimum increases with time in a poly-log scale. Although the gap is larger than conventional C-MAB [18], [21], unlike these algorithms ours does not need fine-grained semi-bandit feedback for each type of resource and integrates an inner loop to reduce the gap in practice. Our approach therefore provides a new perspective on utilizing side information to mitigate the effects caused by low-quality reward feedback. We further verify that our algorithm dramatically increases the total QoS compared with using various heuristics or standard MAB decisions, demonstrating the value of using our two-timescale method in latency-sensitive RAN systems.

We survey related works in Section II before outlining our novel control framework (Section III), learning algorithms with theoretical guarantees (Section IV), and empirical validation (Section V). We finally conclude in Section VI.

II. Related Work

Optimizing 5G/6G resource allocations with new technologies like edge computing and millimeter-wave interfaces has been one of the major objectives of 5G and 6G [6], [7], [10], [26], [27], driven by the increasing volume of traffic and computation demands. At the core of achieving this goal is more efficient and intelligent radio resource management [5], [6], [28] that can handle multi-access and multi-dimensional resources. Extensive studies have proposed to improve provisioning the RAN resources with goals of maximizing the total/mean throughput [29]–[31] or energy efficiency [32]–[34] of 5G/6G cellular networks by optimizing the allocation of spectrum blocks and/or transmit power. These approaches pre-specify fixed objectives and rely on classic optimization methods such as matching and water-filling, which cannot learn and adapt to uncertain resource demands or user QoS metrics. To this end, allocation problems in both offline and online scenarios have been extensively studied [35], [36]

Multi-armed bandits (MAB) can balance the *exploration-and-exploitation* trade-off that arises from our need to adapt

allocations to uncertain demands and QoS needs. Resource allocation problems, at their core, concern the strategic distribution of a finite set of resources among competing entities or tasks to optimize a specific performance metric. The inherent capability of the MAB model to strike a balance between exploration (assessing different allocation strategies) and exploitation (adhering to the most efficacious known strategy) makes it especially pertinent to such challenges. Recent works have proposed different MAB algorithms for resource allocation in communication networks such as wireless channel access scheduling [37], [38] and cloud resource provisioning [16]. However, these do not consider dual-timescale allocation structures, as we observe in the O-RAN intelligent controllers. Another line of works introduce contextual data into the MAB problem, leading to the development of **contextual bandits** for resource allocation; these have furthered the granularity of resource allocation, enabling more tailored decisions based on factors such as user profiles or system states [39]. Unlike the classic bandits for resource allocation schemes, our problem model falls into the category of **combinatorial MAB** (C-MAB), which has a much larger solution space than conventional MAB models. A few works devise new C-MAB frameworks by considering correlated bandits for general online resource allocation problems [40], [41]. Contrary to their semi-bandit feedback approach, we employ a bandit-feedback mechanism and leverage side information to expedite the learning process. In addition, although pioneering C-MAB works [18], [20], [21], [42] have offered algorithms with provable performance guarantees and good scalability, we target a different C-MAB setting for better capturing the structures of application feedbacks and O-RAN two-tier controllers. Namely, users may infrequently submit aggregate QoS scores (reward) over all types of received resources rather than fine-grained feedbacks (see the second contribution in Section I), but the controller needs to adjust multi-resource allocation in near-real time. Some works that propose **two-timescale learning-aided methods** are similar to our method [17], [43] but cannot handle the multi-dimensional resource allocation in a bandit-feedback scenario, where only a scalar feedback is revealed each time infrequently for allocation decisions made for each type of resource. In [44], a two-tier CMAB algorithm is devised to co-optimize the mode and resource allocations for both cellular user equipments (UEs) and device-to-device (D2D) UEs. Different from their definition of "two-tier", which selects the resource allocation for cellular and D2D UEs separately, ours is a two-timescale learning algorithm which leverages carefully abstracted side information in a finer timescale to help estimate latent, yet still uncertain, information about the rewards received in the longer timescale. Existing literature on MAB with side information mainly focuses on contextual bandits [22]–[24], [45], requiring stronger observation capabilities or assumptions on the context-reward mapping, e.g., observing a context before playing arms and knowing the reward function w.r.t. the context. We instead exploit *indirect* knowledge of demand-reward relationships to accelerate learning of the optimal control signals.

We finally note that reinforcement learning (RL) has been proposed to optimize several decisions in 5G or edge networks [43], [46], including optimization of *prices* (e.g. [47]), which are conceptually similar to our regulation signals. Their methods do not permit provable performance guarantees or often require significant and frequent user feedbacks that can be very costly to obtain in 5G/6G RAN systems.

III. Problem Model

As shown in Figure 2, the outer-loop controller resides in the *non-real-time RAN intelligent controller* (non-RT RIC) and computes the control signals sent to users. The *near-real-time RAN intelligent controller* (near-RT RIC) learns real-time side information to improve the outer-loop and also computes the resources allocated to each user. Users complete the two control loops by sending their realized demands and satisfaction scores (feedback) to the controllers.

We consider that K types of cellular blocks (resources) are allocated to N users within the network over T evenly split time slots. Let $[X]$ denote the set $\{1, 2, \cdots, X\}$. Each user $i \in [N]$ can initiate an application session with arbitrary start and end times, between which it sends a request to the near-RT controller for d_{ikt} amount of type-k resource in each $t \in [T]$. These resource types can represent different 5G/6G cellular blocks that are shared by users with the help of advanced multi-access technologies [11]. They can also include transmit power, spectrum, code domains or shares of multiple network "slices", each providing a different QoS guarantee [4], [48]. Another example that may be realized in the future is that of augmented/virtual reality [49] or similar applications requesting both computing resources from edge servers and bandwidth/cellular resources on the 5G/6G connections to these edge servers, although the 5G RAN does not currently control edge server resources. Our framework is agnostic to the specific resources considered; thus, while we give several different examples here of possible resources that the RAN intelligent controllers might allocate, we use the generic term "resources" in the remainder of the paper. In Section V, we consider a scenario in which there are two "resources" representing the uplink and downlink cellular blocks.

We propose an online learning module at the **non-RT RAN intelligent controller** that computes the offered control signals every $\chi \in [1, T)$ timeslots, which not only allows the controller ample time to complete its computations but also requires a lower frequency of collecting feedback from users, as we elaborate below. In each time $t = 0, \chi, 2\chi, \cdots, \left\lfloor \frac{T}{\chi} \right\rfloor \chi$, the non-real-time controller computes a control signal p_{kt} for each type-k resource and posts it for the subsequent χ timesteps to each user i, who submits the demands $\{d_{ikt}(p_{kt})\}_{k\in[K]}$ to the near-RT controller as responses to p_{kt}.

The goal of creating p_{kt} is to indicate the current congestion at each resource, so that each application API can automatically moderate its demands accordingly. In practice, however, the reactions to the control signals may vary across applications, or even across users using the same application but in a different status. We thus treat resource demands as the outputs of a private response function of each user's application to the current signals and do not impose any

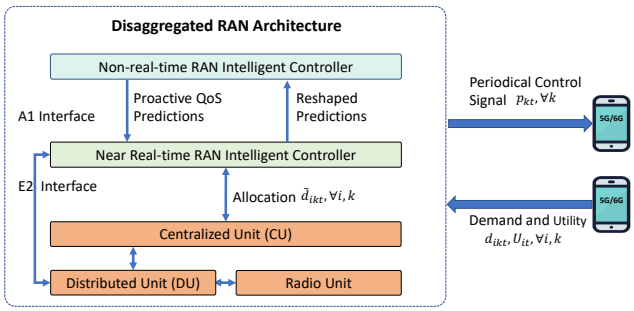


Fig. 2: Overview of our system, showing control signals $\{p_{kt}\}_{k\in[K]}$ for each resource k generated every χ timesteps on the outer loop at the non-real-time controller and near-real-time resource allocations $\tilde{\mathbf{d}}_t$ computed by the inner loop. Users' application APIs respond with their resource demands $\mathbf{d}_t$ for each t and an aggregate score $\sum_{\tau\in[t-\chi,t-1]} U_{i\tau}$ every χ timesteps, since our outer loop only needs this coarse-grained feedback. The orange blocks denote RAN modules that are not directly involved in the control logic, but convey information between the users and RICs.

structure on these functions. We associate each user with a single application; multiple applications at the same user may be modeled as separate users.

Formally, we aim to learn the optimal $\mathbf{p}_t = \{p_{kt}\}_{k\in[K]}$ that can induce each application i to submit a demand $\mathbf{d}_{it} = \{d_{ikt}\}_{k\in[K]}$ that maximizes the quality-of-service (QoS) of all users under the inner-loop allocation rule, and we update $\mathbf{p}_t$ every χ timesteps. In practice, the software agent of each i sends a demand vector $\mathbf{d}_{it}$ each timestep t and an aggregate *satisfaction score* $\sum_{i,\tau\in[t-\chi,t-1]} U_{i\tau}$ every χ timesteps to the near-RT RIC, which will decide the allocation results and transmit $\sum_{i,\tau\in[t-\chi,t-1]} U_{i\tau}$ to the non-real-time controller. More frequently submitted satisfaction feedbacks will not harm the outer-loop or inner-loop efficiency, and practically the parameter χ is tunable according to the computation overhead, solution optimality, and users' willingness to provide feedback. We refer to U_{it} as the *user utility* in the rest of the paper. The optimization problem that we seek to solve is defined as follows:

$$\text{maximize}_{\mathbf{p}_t,\forall t} \sum_{i,t} U_{it}(\mathbf{d}_{it}(\mathbf{p}_t), \tilde{\mathbf{d}}_{it}(\mathbf{p}_t)), \tag{1}$$

$$\text{with } \mathcal{O} : \left(\{\mathbf{d}_{it}(\mathbf{p}_t)\}_{i\in[N]}, f_t\right) \to \tilde{\mathbf{d}}_t(\mathbf{p}_t) \tag{2}$$

Here, the utility function $U_{it}(\cdot,\cdot)$ takes as inputs the submitted demands $\mathbf{d}_{it}(\mathbf{p}_t)$ from application APIs and the realized allocation $\tilde{\mathbf{d}}_{it}(\mathbf{p}_t)$ from the near-RT controller. The main challenge in solving (1) is to simultaneously learn the effects on U_{it} of the demand response functions d_{ikt}, allocation rule $\mathcal{O}$, and the network conditions f_t.

The **near-RT controller** runs a pre-determined, efficient oracle $\mathcal{O}$[1] in each $t \in [T]$ to realize a resource allocation $\tilde{\mathbf{d}}_t = \{\tilde{d}_{ikt}\}_{i\in[N],k\in[K]}$. The oracle $\mathcal{O}$ takes as input user demands d_{ikt} induced by our current control signals p_{kt} (Figure 2) and the network's feasible performance region (abstracted by f_t). For instance, $f_t \le 0$ can include linear constraints that the total allocation of each type-k resource should not exceed the capacity C_{kt}, namely, $\sum_i \tilde{d}_{ikt} \le C_{kt}$; or non-linear ones, such as that the latency L_i of each user i who occupies an MIMO channel should not exceed a threshold $L_i^{\max}$, i.e., $L_i = 1/\left(\sum_n B_{in}\log(1+\text{SNR}(p_{in}))\right) \le L_i^{\max}$ [50], with non-linear dependency between the allocated bandwidth B_{in} and power p_{in} on the nth channel incorporated in the signal-to-noise ratio (SNR$(\cdot)$).

To this end, allocation problems in both offline and online scenarios have been extensively studied [35], [36] (e.g., casting the allocation problem into an NP-hard combinatorial or a convex optimization problem). The allocation can be decided by rule-based [51], [52] or learning-based (e.g., reinforcement learning [16], [36], [46]) schemes, taking users' finally submitted demands $\mathbf{d}_{it}(\mathbf{p}_t)$ as inputs. The focus of this work is instead on *learning the optimal interaction signals between the two time-scale controls under coarse-grained feedback from application APIs*.

Each **application agent of user** i completes the inner and outer control loops by computing its demands $\mathbf{d}_{it}(\mathbf{p}_t)$ each t and an aggregate satisfaction score periodically. We emphasize that, although the true user QoS may depend on the (human) user's intrinsic preferences, these responses could be computed automatically by a SDK (software development kit) running in the background of the user's application without human involvement. Human users would then not need to spend time and energy on specifying their demands and utilities. Users' utility scores, for example, could be computed as a combination of various quality-of-experience metrics [53].

In practice, since user demands and utilities can be automatically calculated by an application SDK, user truthfulness in reporting their satisfaction is not a primary concern, as human users do not directly control the reported demands and utilities. When considering adapting our framework to other resource allocation problems facing human users' submitted demands, our framework guarantees truthfulness if the control signals represent prices that impose a *cost* on users, since our proposed logic becomes a posted-price mechanism [54], under which truthful user demands are provably optimal. Besides, while our framework for setting the control signals $\mathbf{p}_t$ does not make any assumptions on the demand functions $\mathbf{d}_{it}(\mathbf{p}_t)$, many prior works make the reasonable assumption that users select $\mathbf{d}_{it}$ so as to maximize their own utilities or QoS [54], [55]. We omit a full discussion of this point since it does not directly impact our design goal of achieving automatic and optimal interaction between RAN controllers and application APIs.

[1]Picking an efficient allocation rule $\mathcal{O}$ completes the proposed framework but is not our research contribution and thus not discussed. In Section V, we show that our learning algorithm can improve the resource allocation for a variety of oracle choices.

IV. A New Combinatorial-MAB Framework

To adaptively learn the outer-loop optimal control signals, we utilize the multi-armed bandits (MAB) framework, where an agent learns the optimal (i.e., maximizing the accumulated reward) decision (arm) out of a given set of possible arms, through carefully selecting arms in sequential trials. Variants of classical MAB frameworks, however, either *require more information on the chosen decisions than is available in our problem or scale poorly with the resource dimensions due to regarding the decision vectors as independent arms*. We propose a novel learning method to learn the optimal signals in our outer loop by incorporating an iterative gradient-based algorithm that carefully abstracts side information from the inner loop into a combinatorial MAB (C-MAB) framework with minimal feedback, (c.f. bandit feedback C-MAB). It is different from linear bandits [18] or semi-bandits C-MAB [20], [21] since we only access an aggregated feedback of all base arms [14]. Our novel use of inner-loop side information provides a new perspective of speeding up online learning using offline learning from indirect information.

A. Bandit-feedback C-MAB with Side Information

Solving (1) via multi-armed bandit algorithms introduces additional challenges from the continuous choice of control signals $\mathbf{p}_t$. Most MAB algorithms assume a discrete, finite set of decisions rather than a continuum. While one can adapt MAB techniques to search through a continuum of decisions, e.g., through adaptive discretization of the decision interval in Lipschitz bandit frameworks, such algorithms scale poorly as the dimension of the decision space (here, K, the number of resources) increases [25]. Thus, for ease of presentation, we consider that each control signal p_{kt} is discretized to Q distinct candidates values and refer the readers to existing work [56] on optimizing Q and analyzing the effect of this discretization on the achieved MAB reward. In general, a careful choice of Q can nearly match the performance of the offline optimum selection of $\mathbf{p}_t$ from the original continuous range, with the performance gap approaching zero as the number of timeslots $T \to \infty$ [56]. In our scenario, discretizing the signal values has an additional practical benefit: by limiting ourselves to Q signal values instead of a continuum, we may make it easier for users and application APIs to comprehend differences between the control signal values, and thus to react to them appropriately by adjusting their demands.

Given there are Q choices for each type of resource $k \in [K]$, we can certainly apply classic MAB framework by defining the arms as all feasible K-dimensional signal vectors. We then observe U_{it} as the reward of the chosen vector at t. However, this direct adoption leads to a total of Q^K arms and thus slow convergence to the optimum (see the lower-bound results in [14]).

Arm space reshaping. To enable efficient learning that scales better with the size of solution space, we expand our K-dimensional decision $\mathbf{p}$ into a QK-dimensional indicator vector. More specifically, we define a set of binary vectors as our *super-arms*, i.e., $\mathbf{p}_t \in \mathcal{P} \triangleq \{0,1\}^{QK\times 1}$, where each element $p_l \triangleq p_{kqt}$ with $l = Q(k-1)+q$ represents whether to choose ($=1$) or not ($=0$) a *base-arm* of the qth signal candidate for resource k at timestep t. Our learning framework then falls into the category of combinatorial MAB [20] under bandit-feedback [14]. Unlike [18], [20], [21], we only observe an aggregated utility $\sum_{\tau=t-\chi}^{t-1} U_{i\tau}$ every χ timesteps, not the utilities from each individual base-arm in the chosen super-arm $\mathbf{p}_t$ each t.

We then need to learn the *latent reward of each base-arm* from the aggregated reward. Given our expanded decision $\mathbf{p}_t$, we then assume that the expected utility is a linear combination of the corresponding $\mathbf{p}_t$ and a *latent utility vector* $\mathbf{u} \in \mathbb{R}^{QK\times 1}$, namely, $\mathbb{E}[\sum_i U_{it}(\mathbf{p})] = \mathbf{u}^{\mathbf{T}}\mathbf{p} = \sum_{k,q} u_{kq}p_{kq}$ (bold superscription $^{\mathbf{T}}$ denotes transpose), where u_{kq} denotes an unobserved, arbitrary expected utility gained by the total allocation of type-k resource under the signal p_{kq}. Since our decision $\mathbf{p}_t$ consists of a one-hot formatted vector $\mathbf{p}_{kt}$ for each k, we allow arbitrary functions of U_{it} with respect to the value of the chosen decision: e.g., if $K=1$ and we discretize the set of feasible arms $\mathbf{p}$ as $\{0, 0.1, 0.2, \ldots\}$, then choosing a signal of 0.1 means $\mathbf{p}_t = \begin{bmatrix}0 & 1 & 0\ldots\end{bmatrix}$. The utility vector is defined $\mathbf{u} = \begin{bmatrix}U(0) & U(0.1) & \ldots\end{bmatrix}$ no matter what the values $U(0), U(0.1), \ldots$ are. We can further interpret the u_{kq} as unknown weights on each resource or performance metric, e.g., if we use linear regression to predict the aggregate utility from the control signal indicators $\mathbf{p}_t$. For example, if the utility is the negative of the end-to-end session latency, then it is simply the (negative) sum of the latency allocated on each link in a session path.

We then propose an **Online Learning Algorithm for choosing the optimal control Signals** (*OLAS*) to learn the optimal $\mathbf{p}$. The idea is to first construct a confidence region that contains $\mathbf{u}$ with high probability and then solve for the optimal $\mathbf{p}$ given the region of $\mathbf{u}$. Some combinatorial UCB algorithms also construct confidence regions/bounds for estimation [14], [20], [42], but ours is different in first constructing one initial confidence region of the K-dimensional latent vector using the aggregated reward of chosen super-arms, and then shrinking the region in a sub-routine (*GASH*) that exploits side information aggregated by the resource allocator: an uncertain inference mapping between the expected allocation $\mathbb{E}[\tilde{d}_{kq}]$ and u_{kq}. Intuitively, we translate our estimated $\mathbf{u}$ in the initial region to a range of estimated $\mathbb{E}[\tilde{d}_{kq}]$, so that the resource allocator can *correct* our estimated $\mathbf{u}$ using the inference mapping, since it knows better the real $\mathbb{E}[\tilde{d}_{kq}]$ from historical samples and can learn faster than we learn in the outer loop. Prior contextual MAB works consider side information as context [22], [23], but differ from ours in that: (i) the context is provided for each base-arm each time before taking the actions, (ii) the reward u_{ikt} of each base-arm is observable, and (iii) u_{ikt} follows a pre-specified function of the context.

Algorithm interpretation. As in Algorithm 1, every χ timesteps, we update our estimated latent utility vector $\mathbf{u}$ to be a linear least squares estimator $\hat{\mathbf{u}}_t$, with the intuition that $\hat{\mathbf{u}}_t$ minimizes the empirical prediction error of the observed aggregate utilities across all users i:

$$\hat{\mathbf{u}}_t = \text{argmin}_{\mathbf{u}} \sum_{\tau=1}^{t-1} (\mathbf{u}^{\mathbf{T}}\mathbf{p}_\tau - U_\tau)^2, \text{ where: } U_\tau \triangleq \sum_i U_{i\tau}$$

from which $\hat{\mathbf{u}}_t$ can be computed as in line 7 of Algorithm 1 and $\sum_{\tau=t-\chi}^{t-1} U_\tau$ is available each χ timesteps. Further, we construct a confidence region $\mathcal{E}_{0,t}$ of $\mathbf{u}$, so that each point $\mathbf{v} \in \mathcal{E}_{0,t}$ has a bounded empirical prediction error with respect to the error incurred by $\hat{\mathbf{u}}_t$, namely,

$$\left\{\mathbf{v} \middle| \sum_{\tau=1}^{t-1}(\mathbf{v}^{\mathbf{T}}\mathbf{p}_\tau - U_\tau)^2 - \sum_{\tau=1}^{t-1}(\hat{\mathbf{u}}_\tau^{\mathbf{T}}\mathbf{p}_\tau - U_\tau)^2 \leq \beta_t\right\}, \tag{3}$$

which can be rearranged to be the ellipse $\mathcal{E}_{0,t}$ formulated in (4) below, given the expressions of A_t and $\hat{\mathbf{u}}_t$ shown in lines 6-7 of Algorithm 1.

$$\mathcal{E}_{0,t} = \left\{\mathbf{v} | (\mathbf{v} - \hat{\mathbf{u}}_t)^{\mathbf{T}} A_t (\mathbf{v} - \hat{\mathbf{u}}_t) \leq \beta_t\right\} \tag{4}$$

The intuition on the choice of parameter β_t in each timestep t (line 9) is that β_t should be high enough such that the utility parameter vector $\mathbf{u}$ lies in $\mathcal{E}_{0,t}$ with high probability, but also low enough to ensure fast convergence since the regret bound (i.e., the difference between our achieved reward and the optimal reward) of timestep t can be shown to monotonically increase with β_t (Theorem 1). Given the initial confidence ellipse $\mathcal{E}_{0,t}$ (updated every χ timesteps), we call Algorithm 2 (*GASH*) to shrink $\mathcal{E}_{0,t}$ to be $\mathcal{E}_j$ in each subsequent timestep $j \in [t+1, t+\chi)$. To distinguish from the notation t in the outer-loop, we use j instead of t to denote the timesteps when we explain the inner-loop GASH algorithm. In the final step (line 4), $\mathbf{p}_t$ can be exactly solved by an offline oracle. For computation efficiency, an α-approximate solution of $\mathbf{p}$ may be preferred for large K, which yields similar algorithm regret as that shown in Theorem 1 when compared with an α-fraction of the offline optimum that always chooses the best $\mathbf{p}$ [20]. An illustration of the algorithm workflow is shown in Figure 3. Similar to the ConfidenceBall algorithms in [38], the main computational difficulty of Algorithm 1 comes from the optimization of the control signal $\mathbf{p}_t$ (line 4). As suggested by [30], if $|\mathcal{P}|$ is small, we can enumerate all possible $\mathbf{p}$ and solve $\max_{\mathbf{v}\in\mathcal{E}_{t-1}} \mathbf{v}^{\mathbf{T}}\mathbf{p}$ that is a linear programming problem, and total time complexity is $|\mathcal{P}| \cdot poly(QK)$. On the other hand, based on the discussion of ConfidenceBall2 in Section 3.4 of [38], this optimization problem can be seen as polynomial-time equivalent to the NP-hard negative definite linearly constrained quadratic programming problem, which might be computationally practical for large Q, K. However, compared with ConfidenceBall2, we shrink the confidence ellipse $\mathcal{E}_{t-1}$ using Algorithm 2 and we search for the optimal solution in a smaller feasible region. Thus, the computational complexity of Algorithm 1 can be further reduced, i.e., the factor absorbed in the $poly(QK)$ is smaller.

B. GASH: Shrinking the Confidence Region

Inspired by the classic ellipse algorithm for solving convex optimization problems [57], we design a subroutine algorithm *GASH* that searches for the maximizer $\mathbf{u}$ of some concave objective function G by: 1) initializing an original ellipse that contains $\mathbf{u}$ with high probability and 2) iteratively updating $\mathcal{E}_j$ to be an ellipse with the least volume that contains a half ellipse cut by an hyper-plane orthogonal to a (sub)gradient g_j of G at the center of $\mathcal{E}_j$ [57], formalized as (5). Step 2) is realized in *GASH* similar to the classical ellipse algorithm [57]: moving the center of the ellipse in the direction of g_j (line 7 of Algorithm 2) and updating β_j to be smaller (line 8); Step 1) is satisfied by using $\mathcal{E}_{0,t}$ in our Algorithm 1, which we show in Theorem 1. Algorithm 2 continues shrinking $\mathcal{E}_j$ in each iteration j until either $\mathcal{E}_j$ converges ($|\hat{\mathbf{u}}_{j,t} - \hat{\mathbf{u}}_{j-1,t}| \leq \epsilon$), or a pre-set maximum algorithm runtime in each timestep is reached or $g_j = 0$. The challenge is in fact to obtain g_j as side information from our resource allocator.

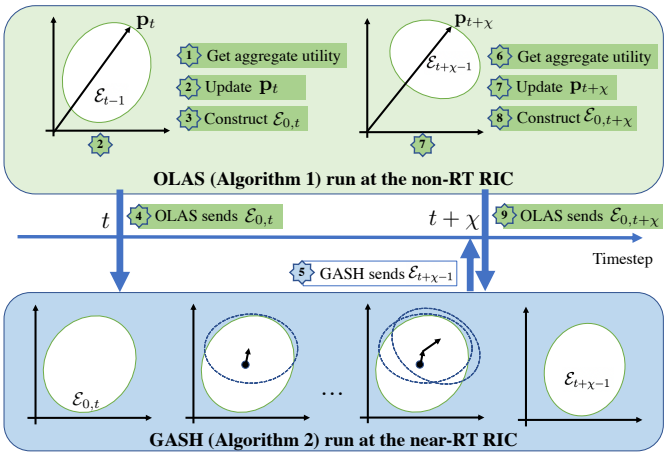


Fig. 3: Illustration of the workflow of our proposed Algorithms OLAS and GASH. The steps numbered from 1 to 9 show how the confidence region (ellipse) is updated at the near-RT RIC and non-RT RIC between two successive updates of the outer-loop OLAS algorithm.

Intuition to re-design g_j. As explained in Sec. IV-A, each element u_{kq} of $\mathbf{u}$ represents the (latent) expected utility gained from the total allocation $\tilde{d}_{kq}$ under the qth control signal candidate, which intuitively is determined by the expected allocation $\mathbb{E}[\tilde{d}_{kq}]$. Suppose that the resource allocator's oracle $\mathcal{O}$ solves for $\tilde{d}_{kqt}$ by maximizing some estimated total utility function $\hat{G}$ given the realized user demands $\{d_{ikqt}\}_{i\in[N]}$ under our chosen qth control signal candidate, and let G represent the deterministic certainty equivalent problem [58] of $\hat{G}$, of which $\mathbb{E}[\tilde{d}_{kq}]$ is the optimal solution. Note that $\hat{G}$, however, may not be known, even to the network operator. Therefore, getting a valid gradient of $\hat{u}_{j,t}$ to correcting the estimation of true expectation $\mathbf{u}$ is impossible. To tackle this, we show in Lemma 1 that it suffices to know the *sign* of g_j in each dimension. While existing works [57] have proved the convergence of the ellipse algorithm and adopted it to solve linear programming, *we are the first to integrate it into a learning framework with simplified gradient required by the learning algorithm.*

Lemma 1. *Conditioned on the event that the true expected latent vector $\mathbf{u}$ lies in $\mathcal{E}_{0,t}$, in each subsequent timestep $j+1$ ($j \geq t$), our shrunk ellipse $\mathcal{E}_{j+1}$ is a sub-region of the confidence region $\mathcal{E}_j$ and can be formulated in* (5) *below:*

$$\left\{\mathbf{v} | (\mathbf{v} - \hat{\mathbf{u}}_j)^{\boldsymbol{T}} A (\mathbf{v} - \hat{\mathbf{u}}_j) \leq \beta_t, g_j \cdot (\mathbf{v} - \hat{\mathbf{u}}_j) \geq 0\right\}, \tag{5}$$

if we use g_j defined in (6) *below, where $\texttt{sign}(\cdot)$ is a sign*

Algorithm 1: Outer-loop algorithm run at the non-real-time RIC: An Online Learning Algorithm for Optimizing the Control Signals (*OLAS*)

Input : σ, K, Q, ϵ, χ, $\mathcal{P}$
Output : $\mathbf{p}_t$, $\forall t$
Initialize: $\hat{\mathbf{u}}_0 = 0$, $\mathbf{p}_0 = 0$

while *each time* $t = \chi\hat{t}, \hat{t} = 1, 2, \cdots, \lfloor \frac{T}{\chi} \rfloor$ *starts* **do**
- `/* Solve for the optimal` $\mathbf{p}_t$ `given updated predictions */`
- Get the updated feasible region $\mathcal{P}$ of $\mathbf{p}_t$ and collect utility $\sum_{\tau=t-\chi}^{t-1} U_\tau$ under $\mathbf{p}_{t-\tau}$;
- Get the shrunk ellipse $\mathcal{E}_{t-1}$ from Algorithm 2;
- Solve for the control signals $\mathbf{p}_t = \text{argmax}_{\mathbf{p}\in\mathcal{P}, \mathbf{v}\in\mathcal{E}_{t-1}} \mathbf{v}^\mathbf{T}\mathbf{p}$;
- Broadcast $\mathbf{p}_t$ and use it for the next χ timesteps;
- Update the parameters of initial confidence region $\mathcal{E}_{0,t}$: $A_t = I + \sum_{\tau<t} \mathbf{p}_\tau \mathbf{p}_\tau^\mathbf{T}$;
- `/* Estimate the latent utility parameter vector */`
- Update the center of $\mathcal{E}_{0,t}$: $\hat{\mathbf{u}}_t = A_t^{-1} \sum_{\tau<t} U_\tau \mathbf{p}_\tau$;
- Get an aggregate utility score U_t from the near-real-time RIC;
- Update the parameter of $\mathcal{E}_{0,t}$: $\beta_t = \max\left(128QK \ln \hat{t} \ln\left(\frac{\hat{t}^2}{\sigma}\right), \left(\frac{8}{3}\ln\left(\frac{\hat{t}^2}{\sigma}\right)\right)^2\right)$;
- `/* Construct the initial confidence region of time` t. `*/`
- $\mathcal{E}_{0,t} = \left\{\mathbf{v} | (\mathbf{v} - \hat{\mathbf{u}}_t)^\mathbf{T} A_t (\mathbf{v} - \hat{\mathbf{u}}_t) \le \beta_t\right\}$;
- Call Algorithm 2 to start to shrink $\mathcal{E}_{0,t}$: *GASH*$(\mathbf{p}_t, A_t, \beta_t, \hat{\mathbf{u}}_t, Q, K, \epsilon)$;

Algorithm 2: Inner-loop algorithm run at the real-time RIC: A Gradient-based Algorithm to Shrink the Confidence ellipse (*GASH*)

Input : $\mathbf{p}_t, A_t, \beta_t, \hat{\mathbf{u}}_t, Q, K, \epsilon, \mathcal{O}$
Output : $\mathcal{E}_{t+\chi-1}$
Initialize: $\mathbf{p} = \mathbf{p}_t, A = A_t$, $\hat{\mathbf{u}}_0 = \hat{\mathbf{u}}_t$, $j = 0$, $\beta_0 = \beta_t$

Broadcast $\mathbf{p}$ to application APIs and receive their real-time demands;
while *each timestep* $j \in \{t, t+\chi-1\}$ *starts* **do**
- Conduct resource allocation using a pre-defined solver $\mathcal{O}$;
- **if** $|\hat{\mathbf{u}}_{j,t} - \hat{\mathbf{u}}_{j-1,t}| > \epsilon$ **then**
  - Calculate our simplified gradient g_j using (6) and break if $g_j = \mathbf{0}$;
  - Normalize the simplified gradient: $\tilde{g}_j = g_j / \sqrt{(g_j)^\mathbf{T} A^{-1} \beta_j g_j}$;
  - Update the predicted center: $\hat{\mathbf{u}}_{j+1} = \hat{\mathbf{u}}_j + \frac{1}{(QK+1)} A^{-1} \beta_j \tilde{g}_j$;
  - Update the parameter: $\beta_{j+1} = \frac{Q^2K^2}{Q^2K^2-1}\left(\beta_j - \frac{2}{QK+1}\beta_j \tilde{g}_j \left(\tilde{g}_j\right)^\mathbf{T} A^{-1} \beta_j\right)$;
  - Update the timestep of the inner-loop: $j = j + 1$;
- Define $\beta_t^{\min}$ to be the smallest β_j over all j;
- `/* Choose the ellipse with the smallest width */`
- Return the refined confidence region $\mathcal{E}_{t+\chi-1} = \left\{\mathbf{v} | (\mathbf{v} - \hat{\mathbf{u}}_j)^\mathbf{T} A (\mathbf{v} - \hat{\mathbf{u}}_j) \le \beta_t^{\min}\right\}$;

function:

$$\exists\ \alpha_{kq} > 0,\ g_{jkq} = \alpha_{kq} \cdot sign(u_{kq} - \hat{u}_{jkq}),\ \forall k, q. \quad (6)$$

Besides, each updated confidence region $\mathcal{E}_{j+1}$ must contain $\mathbf{u} = \{u_{kq}\}_{k\in[K], q\in[Q]}$ given that $\mathbf{u} \in \mathcal{E}_{0,t}$ and is the least volume of ellipse that contains the sliced half ellipse $\mathcal{E}_j$.

Proof of Lemma 1. First, based on the expression of the initial ellipse $\mathcal{E}_{0,t}$ and our expression of updating $\hat{u}_{j+1}$ and parameter β_{j+1} in lines 7–8, one can verify that the ellipse $\mathcal{E}_j$ is formulated as $\left\{\mathbf{v} | (\mathbf{v} - \hat{\mathbf{u}}_j)^\mathbf{T} A (\mathbf{v} - \hat{\mathbf{u}}_j) \le \beta_t\right\}$. Hence, $\mathcal{E}_{j+1}$ defined in (5) is thus the joint region of $\mathcal{E}_j$ and the half space $\{\mathbf{v} | g_j \cdot (\mathbf{v} - \hat{\mathbf{u}}_j) \ge 0\}$. Next, (6) means that g_i is a vector, where each element indicates the direction of $u_{kq} - \hat{u}_{jkq}$. Taking (6) and $\mathbf{v} = \mathbf{u}$ into (5), it's easy to verify that the inequalities of (5) hold, which proves that $\mathcal{E}_{j+1}$ contains $\mathbf{u}$ as $\mathcal{E}_j$. Finally, lines 7–8 combined with condition (5) yield that *GASH* inherits the property of the ellipse algorithm [57]. Therefore, $\mathcal{E}_{j+1}$ is the least volume of ellipse containing half ellipse of $\mathcal{E}_j$. □

Estimating the g_j defined in (6). Given Lemma 1's results, we now discuss how to get the indicator sign$(u_{kq} - \hat{u}_{jkq})$ for each k, q in any iteration j of Algorithm 2. Theoretically, the side information of such sign$(\cdot)$ does not require the knowledge on the exact value or range of the u_{kq} that we need to estimate, but instead *only requires that the system can answer queries that indicate whether our estimated $\hat{u}_{kq}$ is larger or smaller than u_{kq}*. This requirement is much weaker than that of knowing the true value of u_{kq}. It can, for example, be thought of as analogous to relative feedback models in other learning scenarios such as recommendation systems, where it is much easier for the user to tell us whether our estimated score of each item is higher or lower compared to that of another item than to directly tell us the absolute score of the item [59], [60], although our overall method and problem are very different from those of recommendation systems. Below, we provide two specific methods to design such queries in practice. We also show in the proof of Theorem 1 that the estimation error of g_j does not change the order of the overall performance gap (regret) of Algorithm 1.

The first method leverages the monotonicity of some measurable performance metric with respect to the utility score $U(\mathbf{p})$. More specifically, we assume that certain applications (denoted as a set ϕ) achieve some performance using their allocated resources under $\mathbf{p}$ is monotonic w.r.t. $U(\mathbf{p})$, such as the throughput, a common performance metric in RAN resource allocation [6], [51]. In some cases, the throughput itself may be the user utility function, though in general we cannot simply maximize throughput as some users may have utilities that are more complex functions of throughput

or other directly measurable performance metrics. We note that knowing the allocation of each type of resource or the measurable performance is far from knowing the actual value of the utility gained by getting each type of resource; instead, we are only able to compare two aggregated application utility scores, based on the relative difference of the measurable performance. Then, we can obtain $\text{sign}(u_{kq} - \hat{u}_{jkq})$ as long as the RIC or application side can measure the performance of applications in set ϕ and compare their (a) expected measurable performance achieved using the expected type-k allocated resource each time under the qth price; and (b) the mean performance in those historical timesteps over which the average received utility score equals $\hat{u}_{jkq}$, or the average of measurable performances in two groups of timesteps when the received average utility scores constitute the smallest range containing $\hat{u}_{jkq}$.

We can design another method to find $\texttt{sign}(u_{kq} - \hat{u}_{jkq})$ for our Algorithm 2 if we can infer (e.g., from prior knowledge or historical samples) a range[2], denoted by $[u^-_{kq}, u^+_{kq}]$ of each expected utility entry u_{kq} of the utility vector. With $[u^-_{kq}, u^+_{kq}]$ obtained, we replace the function $\texttt{sign}(\cdot)$ by $\texttt{sign}^{\text{approx}}$ defined below, which guarantees that Lemma 1 is still satisfied. The only difference is that when the last case of (7) occurs, $\texttt{sign}\left(u_{kq} - \hat{u}_{jkq}\right)$ could be -1 or 1 while $\texttt{sign}^{\text{approx}}\left(u_{kq} - \hat{u}_{jkq}\right) = 0$, meaning that GASH could stop updating earlier if we use $\texttt{sign}^{\text{approx}}(\cdot)$ instead of $\texttt{sign}(\cdot)$.

$$\texttt{sign}^{\text{approx}}\left(u_{kq} - \hat{u}_{jkq}\right) = \begin{cases} 1 & \text{if } u^+_{kq} < \hat{u}_{jkq} \\ -1 & \text{if } u^-_{kq} > \hat{u}_{jkq} \\ 0 & \text{Otherwise} \end{cases} \quad (7)$$

Intuitively, a smaller inference range can lead to a smaller confidence region since we might have more iterations to correct our estimated $\hat{u}_{jkq}$. However, a smaller inference range in turn imposes a stronger assumption on the real-time allocator, e.g., using more historical samples to obtain such inference ranges. We test their impact on the performance of our OLAS and GASH algorithms algorithm in Section V-B. In the following, we also discuss potential methods for computing the inference range of each u_{kq} but leave systematic methods as our future work. For each k, q, the larger end point u^+_{kq}, of the inference range of u_{kq}, can be obtained in three steps. (i) We discretize historical U_t's to certain levels, denoted as U'_l (l indexes the level). We associate each U'_l with the average corresponding allocated resource $\hat{d}_{kqt}$ over historical timesteps. (ii) We estimate the expected allocation $\hat{d}_{kq}$ by its upper confidence bound (UCB) [20], [38]. (iii) Then u^+_{kq} is approximated by the U'_l that has the associated $\hat{d}_{kqt}$ closest to the UCB of $\hat{d}_{kq}$. The choice of using U'_l as an upper-bound of u^+_{kq} is natural as U'_ls are the total utility scores achieved by all the types of allocated resources which is larger than that gained by the single type associated with $\hat{d}_{kq}$. The smaller end point of the interval, u^-_{kq}, can be approximated by zero.

We next analyze the regret of Algorithms 1 and 2, given that we have a method to estimate the subgradients g_j in Algorithm 2's inner loop.

[2]We call the obtained range of each utility u_{kq} *inference range* and show its impact on our algorithm performance in Section V.

C. Regret Analysis

Every χ timesteps, our Algorithm 1 produces a signal vector $\mathbf{p}_t$ which is used by the inner-loop Algorithm 2 for the subsequent $\chi - 1$ timesteps, given that Algorithm 1 is only able to receive one utility feedback every χ timesteps. To theoretically analyze the performance of our Algorithm 1 with subroutine Algorithm 2, we adopt the classic metric in online learning, the *regret* [14], which is defined as the gap in expected total reward between using the optimal super-arm $\mathbf{p}$ and our chosen $\mathbf{p}_t, \forall t$:

$$\text{Regret} = T \max_{\mathbf{p}} \mathbf{u}^{\mathbf{T}}\mathbf{p} - \sum_{t\in[T]} \mathbf{u}^{\mathbf{T}}\mathbf{p}_t$$

We prove in Theorem 1 that the regret is sub-linear with the time-horizon.

Theorem 1 (Regret Bound)**.** *Suppose the minimum gap of the utility per time between using the optimal control signal vector and any sub-optimal one is Δ, and the maximum total utility per time is U_{max}. Then our online learning algorithm OLAS achieves a **polylog(T)** regret, more specifically, a regret at most $O\left(\frac{\chi}{\Delta} QK\Lambda\beta_T \ln(T/\chi)U_{max}\right)$, where:*

$$\beta_T = \max\left(128QK\ln\frac{T}{\chi}\ln\left(\frac{T^2}{\sigma\chi^2}\right), \left(\frac{8}{3}\ln\left(\frac{T^2}{\sigma\chi^2}\right)\right)^2\right),$$

$$\Lambda = \max_t \left(\min\left(1, \max\left(\prod_{j=0}^{\chi-1}\frac{1+(QK-1)\Gamma_{jt}}{Q^2K^2-1}, 0\right)\right)\right)$$

$$\Gamma_{jt} = \frac{g_j}{\sqrt{g_j^{\boldsymbol{T}} A_t^{-1} g_j}}\left(\frac{g_j}{\sqrt{g_j^{\boldsymbol{T}} A_t^{-1} g_j}}\right)^{\boldsymbol{T}} A_t^{-1} \quad (8)$$

The regret bound is polynomial in Q and K, in contrast to $O(Q^K)$ if using conventional MAB algorithms that ignore the combinatorial structure of the control signal decisions [61]. Note that Λ is smaller than 1 by design and denotes the multiplicative factor of the regret reduced by our Algorithm 2. Moreover, a larger χ, the number of timesteps within each outer-loop duration, leads to a higher regret, compared to an optimal algorithm that can adjust its decisions in each timestep. This dependence on χ can be explained as that more frequent feedback leads to a higher learning efficiency but also greater burden on the user's application to provide such feedback. Our regret does not increase dramatically with infrequent feedback, due to the fact that: although each mistake OLAS makes at any $t = \lfloor\frac{T}{\chi}\rfloor$ (a played sub-optimal super-arm) will incur χ mistakes since we do not change $\mathbf{p}_t$ in the subsequent $\chi - 1$ timesteps, the diameter of the initial confidence region updated by OLAS increases more slowly than updating $\mathbf{p}_t$ every $t \in [T]$, and it can be further reduced by our subroutine GASH algorithm. In addition, the total number of *GASH* iterations (within each χ-timestep duration), in which the confidence region is successfully shrunk depends on the inference capability that the real-time allocator provides. E.g., the number of inner-loop timesteps for which the system is able to answer the query of getting $\text{sign}(u_{kq} - \hat{u}_{jkq}) \neq 0$. We leave how to improve the capability of the system to provide such useful side information in realistic 5G and 6G O-RAN

systems to our future work.

Proof of Theorem 1. The main idea is that with probability at least $1-\sigma$ (the parameter σ is an input to define β_t in Algorithm 1) our regret is at most $O\left(\frac{\chi}{\Delta}QK\Lambda\beta_T\ln(T/\chi)U_{\max}\right)$ in the event that $\{u_{kq}\}_{k,q}$ is within our confidence ellipse, and is no greater than $U_{\max}T$ with probability at most σ otherwise. We can tune the parameter σ small enough so that $U_{\max}T\sigma$ can be ignored.

More specifically, we first upper-bound the regret for the event where $\{u_{kq}\}_{k,q}$ lies in $\mathcal{E}_t,\ \forall t$, which happens at least probability $1-\sigma$. This can be proven in two steps. First, based on Hoeffding-Azuma inequality [42], our prediction $\{u_{kq}\}_{k,q}$ lies in $\mathcal{E}_{t0},\ \forall t \in [\lfloor\frac{T}{\chi}\rfloor]$, with at least probability $1-\hat{\sigma}$, where $\hat{\sigma}$ is tunable parameter affecting the parameter β_t in Algorithm 1. Second, for each iteration j of Algorithm 2, if we don't have a precise g_j as Lemma 1 assumes, there exists a $\sigma_j^{'} \in (0,1]$ so that with a probability of at least $1-\sigma_j^{'}$, the shrunk ellipse $\mathcal{E}_j$ still contains $\mathbf{u}$, conditioned on that $\mathbf{u}$ lies in the ellipse updated in $j-1$ and $\mathcal{E}_{t0}$ updated by Algorithm 1. Therefore, there exists a $\sigma>0$ that $1-\sigma=(1-\hat{\sigma})\prod_{j=1}^{J}(1-\sigma_j^{'})$.

Below we then upper-bound the regret for normalized $U_{\max}\le 1$; the bound multiplied by the original non-normalized $U_{\max}$ completes the proof. Let $\beta_{j,t}$ denote the β_j in Algorithm 2 for time t. Below we upper-bound the regret per time, defined as $[U^*-U_t]^+ \triangleq \max\left\{0, \mathbf{u}^T(\mathbf{p}^*)-\mathbf{p}_t\right\}$, which is the regret each time we choose a wrong decision $p_t, \forall t \in \lfloor\frac{T}{\chi}\rfloor$, if (U^*-U_t) is non-negative. Since we do not change our chosen signal $\mathbf{p}_t$ for χ times, i.e., within each round of the outer-loop, the regret each time is at most multiplied by χ. Recall the ellipse structure of $\mathcal{E}_t$, our temporary assumption $U_{\max} \triangleq \max_t U_t \le 1$, and the definition of Δ, we have $[U^*-U_t]^+ \ge \Delta$ if $[U^*-U_t]^+>0$, and then we can upper-bound $[U^*-U_t]^+$, by:

$$\frac{(U^*-U_t)^2}{\Delta} \le \frac{4}{\Delta}\left(\min\{1,\sqrt{\beta_{m,t}\mathbf{p}_t^{\mathbf{T}}A^{-1}\mathbf{p}_t}\}\right)^2 \tag{9}$$

$$\le \frac{4\beta_{m,t}}{\Delta}\left(\min\{1,\sqrt{\mathbf{p}_t^{\mathbf{T}}A^{-1}\mathbf{p}_t}\}\right)^2$$

$$= \frac{4}{\Delta}\beta_{m,t}\left(\min\{1, p_t^TA^{-1}p_t\}\right) \tag{10}$$

$$\le \frac{8\beta_{m,t}}{\Delta}\ln\left(1+\mathbf{p}_t^{\mathbf{T}}A^{-1}\mathbf{p}_t\right). \tag{11}$$

The first inequality (9) can be proved by upper-bounding the per-time regret as follows.

$$[U^*-U_t]^+ = \mathbf{u}^T(\mathbf{p}^*-\mathbf{p}_t) \le (\mathbf{u}-\mathbf{v}^*)^T\mathbf{p}_t \tag{12}$$

$$\le |(\mathbf{u}-\hat{\mathbf{u}})^T\mathbf{p}_t| + |(\hat{\mathbf{u}}-\mathbf{v}^*)^T\mathbf{p}_t| \tag{13}$$

$$\le 2||A_t^{\frac{1}{2}}(\mathbf{u}-\hat{\mathbf{u}})||\cdot||A_t^{-\frac{1}{2}}\mathbf{p}|| \le 2\sqrt{\beta_{mt}\mathbf{p}_tA_t^{-1}\mathbf{p}_t}, \tag{14}$$

where p^* is the optimal solution and $\mathbf{v}^*$ is the selected $\mathbf{v}$ in line 4 of Algorithm 1. Therefore, (12) is due to the fact that $\mathbf{u}^T\mathbf{p}^* \le \mathbf{v}^*\mathbf{p}_t$ according to the optimization in line 3. The first inequality in (14) is due to that $|(\mathbf{u}-\hat{\mathbf{u}})^T\mathbf{p}_t| \le ||A_t^{\frac{1}{2}}(\mathbf{u}-\hat{\mathbf{u}})||\cdot||A_t^{-\frac{1}{2}}\mathbf{p}||$ and $|(\hat{\mathbf{u}}-\mathbf{v}^*)^T\mathbf{p}_t| \le ||A_t^{\frac{1}{2}}(\mathbf{u}-\hat{\mathbf{u}})||\cdot||A_t^{-\frac{1}{2}}\mathbf{p}||$, which we prove as follows.

For any v in our confidence region ellipse $\mathcal{E}_t$ including $\mathbf{u}$, since A_t is symmetric, we have:

$$\begin{aligned}|(\mathbf{v}-\hat{\mathbf{u}})^T\mathbf{p}_t| \le& ||(A_t^{\frac{1}{2}}(\mathbf{v}-\hat{\mathbf{u}}))^T||\cdot||A_t^{-\frac{1}{2}}\mathbf{p}_t||\\ =& ||(A_t^{\frac{1}{2}}(\mathbf{v}-\hat{\mathbf{u}}))^T||\sqrt{\mathbf{p}_t^TA_t^{-1}\mathbf{p}_t}.\end{aligned} \tag{15}$$

We can prove the upper-bound of $|(\hat{\mathbf{u}}-\mathbf{v}^*)^T\mathbf{p}_t|$ similarly. Finally, combined with the definition of our ellipse $\mathcal{E}_t$, (15) leads to the last inequality of (14) .

Further, we can upper-bound (11) by first analyzing the parameter matrix A_t. We have:

$$\begin{aligned}\text{trace}\left(A_t\right) &= \text{trace}\left(I+\sum_{\tau=1}^{t-1}\mathbf{p}_t\mathbf{p}_t^T\right) \le QKt,\\ \det\left(A_{t+1}\right) &= \prod_{\tau=1}^{t}\left(1+\mathbf{p}_t^{\mathbf{T}}A_\tau^{-1}\mathbf{p}_t\right).\end{aligned} \tag{16}$$

Therefore, we have $\det\left(A_t\right) \le t^{QK}$. Based on (11), the total regret over T timesteps is then $\sum_{t\in[\lfloor\frac{T}{\chi}\rfloor]}\chi[U^*-U_t]^+$ and can be upper-bounded by:

$$\begin{aligned}&\frac{8}{\Delta}\chi\max_t(\beta_t)\sum_{t=1}^{\lfloor T/\chi\rfloor}\ln\left(1+\mathbf{p}_t^{\mathbf{T}}A^{-1}\mathbf{p}_t\right)\\ =&\frac{8}{\delta}\chi\max_t(\beta_t)\ln\left(\prod_{t=1}^{\lfloor T/\chi\rfloor}\left(1+\mathbf{p}_t^{\mathbf{T}}A^{-1}\mathbf{p}_t\right)\right)\\ =&\frac{8}{\Delta}\chi\max_t(\beta_t)\ln\left(\det(A_T)\right) \le \frac{8}{\Delta}\chi\max_t(\beta_t)QK\ln\lfloor T/\chi\rfloor,\end{aligned}$$

where $\max_t\beta_t = \prod_{j=T}^{T+\chi-1}\frac{\beta_{j+1}}{\beta_j}$. In order to maximize the regret, we want to use the smallest β_t in any χ-timestep duration. We first examine if just using the confidence ellipse shrunk in the last iteration j rather than using the ellipse with the least β_j. We then have that $\frac{\beta_{t+\chi-1}}{\beta_t}$ equals:

$$\begin{aligned}&\prod_{j=t}^{t+\chi-1}\frac{\beta_{j+1}}{\beta_j} = \prod_{j=t}^{t+\chi-1}\frac{Q^2K^2}{Q^2K^2-1}\left(1-\frac{2\Gamma_j}{QK+1}\right)\\ &= \prod_{j=t}^{t+\chi-1}\frac{1+(QK-1)\Gamma_j}{Q^2K^2-1} := \prod_{j=t}^{t+\chi-1}\Lambda_j\\ &\text{where: } \Gamma_j = \frac{g_j}{\sqrt{g_j^TA_t^{-1}g_j}}\left(\frac{g_j}{\sqrt{g_j^TA_t^{-1}g_j}}\right)^TA_t^{-1}\end{aligned} \tag{17}$$

Finally, the additional number of sub-optimal $\mathbf{p}_t$ chosen due to the mistakes in estimating g_j by potential inference methods discussed in Section IV-B is at most $O(QK\log T)$, since we can use the UCB algorithm in each near-RT timestep for estimation in each corresponding method (e.g., throughput and allocations $\hat{d}_{kq}$ in our two methods to estimate g_j). Thus, such inference error does not increase the order of our regret shown in (8).

Note that if QK, the dimension of our decision region is sufficiently large, we will have $\Lambda_j \le 1$, which determines the fraction of regret we can reduce by shrinking the confidence ellipse. For small QK, we may encounter that the smallest width of the ellipse is larger than the initial one and thus stop

running Algorithm 2, leading to a regret with $\Lambda_j = 1$. □

V. Experimental Evaluation

We validate our algorithm performance with a 5G usage dataset [62] and user demand and reward parameters drawn from real traces [63].

A. Simulation Set-up

For our Algorithm 1 (*OLAS*), we choose $K = 2$ resource types, e.g., two network slices, $Q = 5$ candidates of control signals p_{kt}, $N = 5$ user applications, and $T = 500$ timesteps as defaults. The entire time span of our experiment can simulate hours or days in the real O-RAN 5G system. Given that each timestep of our Algorithm 1 can be much longer than that of the near-real-time resource allocation, which is in practice below 100ms, the processing time of each instance of Algorithm 1 is negligible considering the length of each outer-loop timestep.

Different environments. Since we assume that the dynamics of the network characteristics and users' demands are i.i.d. in Section IV-C to ensure an upper-bounded regret for our Algorithm 1, we first simulate an environment called **Scenario I**: the latent utility entries u_{kq} are randomly selected from a distribution of the *utility coefficients* extracted from mobile usage traces in [63]. We simulate the user utility scores by sampling the realized **reward** of each timestep from a Gaussian distribution with mean equal to the sum of the (u_{kq}) of the chosen signal levels q over all k. For Algorithm 2's subroutine, we randomly generate the *inference range* of the $\hat{u}_{jkq}$ for calculating the side information g_i according to (6).

Our **Scenario II** is based on a real-world 5G usage dataset [62]. We consider two types of resources (downlink and uplink throughputs) and three different real applications (File Download, Amazon Prime, and Netflix). Users' (not necessarily i.i.d.) demands are the downlink and uplink bitrates of these applications provided in the dataset. We take $Q = 10$ control signals to affect the final allocation $\hat{d}_{ikq} = \max(d_{ik} - \eta \cdot q, 0)$, where d_{ik} is the raw demand of user i for resource k before reacting to our control signals, $\eta = 1$ is a decay factor, and $q = 0.2x$ is the control signal with $x = 1, 2, \cdots, Q$. We consider two types of utility functions: linear utility and α-fair utility. For the former case, the utility of application i on resource k is given by a simple linear function $u_{ikq} = 5\hat{d}_{ikq} - q \cdot \hat{d}_{ikq}$. For the latter case, the utility of application i on resource k is given by an α-fair utility function $u_{ikq} = \frac{c_k(\hat{d}_{ikq})^{1-\alpha_k}}{1-\alpha_k} - q \cdot \hat{d}_{ikq}$, with $\alpha_k = 0.95$ and c_k sampled from a uniform distribution $U(2, 2.5)$ according to the data in [63]. The choices of the resource allocation and utility functions are intended to test the robustness of our assumption in Section IV that the near-RT RIC allocates the resources by maximizing the users' total utility, while in this scenario we consider an allocation rule that is agnostic to users' utility functions.

Our last environment, called **Scenario III** combines the 5G usage dataset [62] and customized inner-loop resource allocation rules. The utility function of resource k is still $u_{ikq} = \frac{c_k(\hat{d}_{ikq})^{1-\alpha_k}}{1-\alpha_k} - q \cdot \hat{d}_{ikq}$, while the final allocation $\hat{d}_{ikq}$ is not only affected by the control signal q but also inner-loop allocation rules (e.g., the water-filling algorithm and others elaborated later) under randomly generated resource capacities in each inner timestep t. These allocation rules simulate the near-RT RIC's dynamic allocation of resources according to capacity constraints.

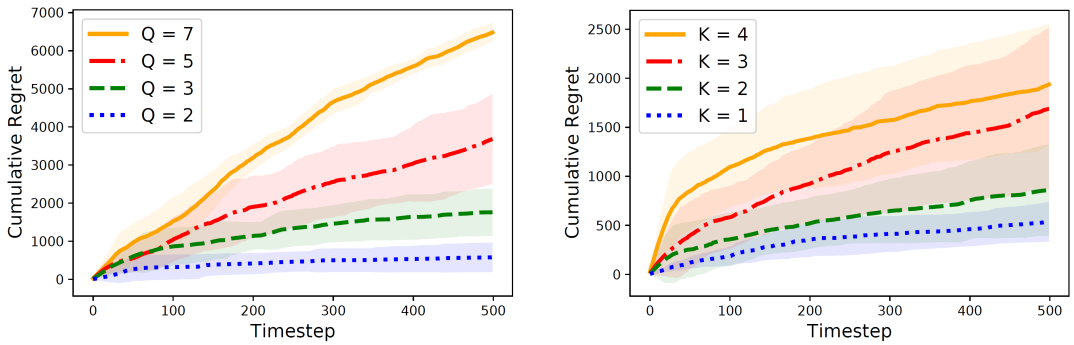
B. Comparison to Other Bandit Approaches

We next compare the regret incurred by our Algorithm 1 to that incurred with heuristics and a classic MAB algorithm, UCB, validating Theorem 1's result. We omit the comparison with contextual bandit algorithms as their rewards rely on contexts (demands, allocations, etc.) accessed *before* playing the arms in each timestep, which are not reasonable in our O-RAN 5G environments. We consider regret in only Scenario I as the optimal benchmark for calculating the regret is not available for other scenarios where non-i.i.d. trace data is used.

Effect of Q and K. As shown in Figures 4a and 4b, our Algorithm 1 achieves a regret sublinear with the number of timesteps for various Q and K, with a larger regret for a larger number of signal candidates Q (Figure 4a) or number of resources K (Figure 4b), verifying Theorem 1. Larger values of Q and K expand the dimension of the decision space, which would intuitively increase the regret. A larger Q may also admit higher overall *reward* (not shown) due to more flexibility in the choices of control signals.

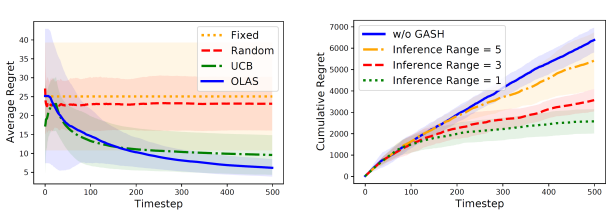
Superiority of our algorithms. We compare Algorithm 1 with four benchmarks: **Random**, which randomly selects $\mathbf{p}_t$ each t; **Fixed**, which uses a randomly selected $\mathbf{p}_0$ until the last timestep; **UCB** [14], a classic MAB algorithm that requires Q^K arms (explained in Sec. IV-A); and our Algorithm 1 without the Algorithm 2 subroutine (**w/o GASH**). Figure 5a shows that **Random**, **Fixed** and **UCB** incur a higher regret by 273%, 305% and 55% compared to Algorithm 1 in Scenario I. Figure 5b indicates a higher improvement in reward compared to (**w/o GASH**) with a smaller *inference range* of $\hat{u}_{jkq}$, i.e., more informative side information. Utilizing side information from the near-RT controller thus allows us to achieve a higher total utility (equivalent to a lower regret) in the long run, compared to ignoring this information.

We then compare our Algorithm 1 to the fixed, random, and UCB baselines in Scenario II, with demands taken from 5G data traces [62]. Figure 6 shows that Algorithm 1 achieves significantly higher reward than all three baseline algorithms for both linear and α-fair utility functions. In fact, its cumulative reward is briefly super-linear with the number of timesteps, indicating that the algorithm successfully learns the control signals that yield higher utility. Once it has learned these signals, it extracts a higher reward per timestep than before, as it no longer needs to explore suboptimal signal values. The **UCB** algorithm achieves higher reward than **Random** or **Fixed**, as we would expect since it can balance exploration and exploitation of different control signals, but lower reward than OLAS as it does not take advantage of side information. Figures 7 and 8 indicate that Algorithm 1 outperforms other baselines by achieving 8%-19% higher throughput and 10%-18% reduced latency when measured with α-fair utility functions. While all baselines show a rapid increase in throughput



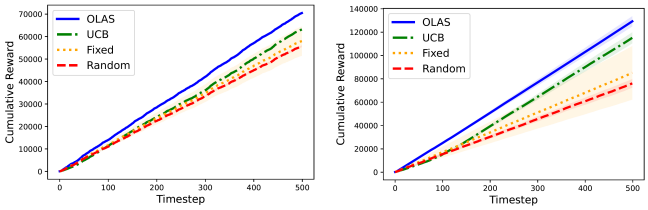
(a) Varying Q (K = 2) (b) Varying K (Q = 2)

Fig. 4: Algorithm 1 (OLAS) achieves sublinear mean regret (lines) that increases with (a) Q and (b) K. Shaded regions show one standard deviation of the regret.



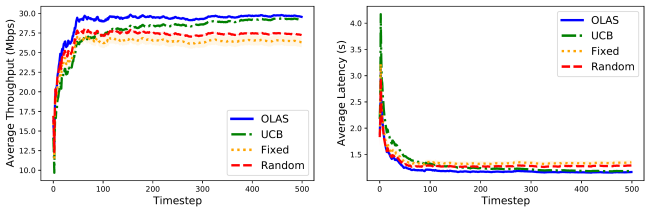
(a) Comparison with baselines (b) Varying the inference range

Fig. 5: Algorithm 1 (OLAS) achieves achieves lower regret than the baselines in Scenario I (5a) and lower regrets for smaller inference ranges, which allow for more shrinkage of the confidence region in Algorithm 2 (GASH)'s inner-loop subroutine (5b). Changed (a) to average regret



(a) Linear utility (b) α-fair utility

Fig. 6: Algorithm 1 (OLAS) achieves higher rewards than baseline algorithms on a 5G usage dataset [62] (Scenario II). Shaded regions show one standard deviation of the reward.

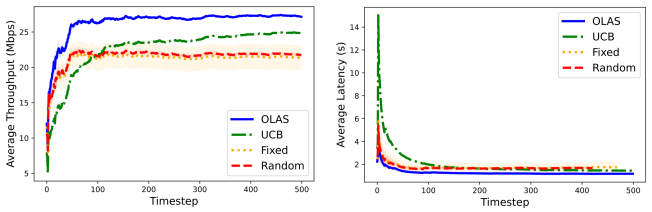


(a) Throughput (b) Latency

Fig. 7: Algorithm 1 (OLAS) achieves higher throughput and lower latency than baseline algorithms on a 5G usage dataset [62] with α-fair utility (Scenario II). Shaded regions show one standard deviation. New plots show network-related metrics (Revision 1).

in the initial timesteps, likely due to an increase in raw demand from the data trace, **OLAS** ultimately learns a better allocation than the **Fixed** and **Random** baselines that do not learn or adapt. This distinction is particularly apparent in Figure 8's results with decay factor $\eta = 0.5$, for which the control signal has less influence on user demand and it is therefore more difficult to learn the optimal signals. Figure 9 details the cumulative rewards attained by each algorithm post 500 timesteps when considering 20, 60, 100 users for each application type (resulting in total counts of 60, 180, and 300 users). The reward for each application aggregates the utility for all its users. As user numbers rise, the reward gap between Algorithm 1 and other baselines also widens.

Finally, we compare our Algorithm 1 to UCB under two different inner-loop allocation rules in Scenario III, with demands taken from 5G data traces [62] and utility coefficients taken from [63]. Different from Scenario II, we define $\bar{d}_{ikp} = \max(d_{ik} - \eta \cdot q, 0)$ as the requested demand of user i to resource k under control signal q. For each resource k, we consider a random capacity B_{kq}, which is 80–100% of the total requested demand from all users (i.e., $\sum_i \bar{d}_{ikp}$). Thus, the final allocation $\hat{d}_{ikp}$ depends on the requested demand $\bar{d}_{ikp}$, the capacity B_{kq}, and the allocation scheme. We first simulate a **Proportional** allocation scheme that allocates each type of resource in proportion to the users' requested demands $\bar{d}_{ikp}$, i.e., $\hat{d}_{ikp} = \frac{\bar{d}_{ikp}}{\sum_i \bar{d}_{ikp}} B_{kq}$. The second allocation scheme, **Water-Filling**, allocates resource k to users with the objective of maximizing $\sum_i \log(1+\hat{d}_{ikp})$, while satisfying $\sum_i \hat{d}_{ikp} \leq B_{kq}$ and $\hat{d}_{ikp} \leq \bar{d}_{ikp}$ for all i. We consider 10 inner-loop allocations for each outer-loop step (i.e., $\chi = 10$). Figure 10 shows that our algorithm **OLAS** outperforms the UCB, fixed, and random baseline algorithms under both allocation rules for $3,000$ inner-loop timesteps. Notice that the obtained reward under the Water-Filling rule is less than that under the Proportional rule, as Water-Filling puts more emphasis on fairness across users and thus may decrease the allocation proportions of users with higher demands and utility coefficients in our scenario.



(a) Throughput (b) Latency

Fig. 8: (New results for Revision 2) Algorithm 1 (OLAS) achieves higher throughput and lower latency than baseline algorithms on a 5G usage dataset [62] with α-fair utility and decay factor $\eta = 0.5$, which makes the learning task more difficult (Scenario II). Shaded regions show one standard deviation.

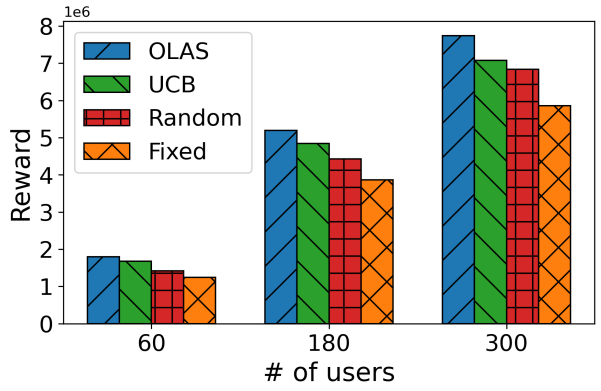
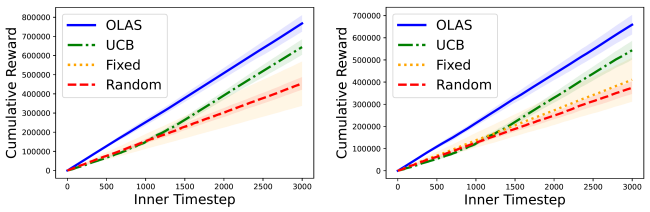


Fig. 9: Algorithm 1 (OLAS) achieves higher reward than baseline algorithms on a 5G usage dataset [62] (Scenario II) when varying the total number of users. New plot shows the scalability of our algorithm.

C. Future Implementation Plan for the O-RAN Use-case

Our proposed model and algorithm can be implemented using features in the O-RAN software community (OSC) [64] Cherry (the third and latest version) software release from the O-RAN alliance. As shown in Figure 11, the near-real-time RIC in this implementation includes a KPIMON module that collects the per-UE (user equipment) based metrics through the E2 termination from the O-CU and O-DU, a shared data layer (SDL) that stores these metrics, a QoS Prediction (QP) and a QP-driver module that performs holistic utility estimation, and the Traffic Steering module that performs resource allocation (i.e., traffic engineering) based on the policy issued by the A1 policy coordinator in the non-real-time RIC. The UEs send their demands and utility scores through the O-RU and then O-DU to the near-real-time RIC controller, which pass the scores with some aggregated side information to the non-real-time controller. Our oracle function $\mathcal{O}$ in (1) will be implemented as logic in the traffic steering module, and the inference mapping function used in the non-real-time learning strategy (Section IV-A) will be produced by the QP modules. The non-real-time RIC can utilize the Operations, Administration, and Maintenance (OAM) configuration, performance and fault



(a) Proportional (b) Water-Filling

Fig. 10: With (a) proportional and (b) water-filling inner-loop allocation rules on 5G data traces (Scenario III), Algorithm 1 (OLAS) achieves higher rewards than baseline algorithms. Shaded regions show one standard deviation of the reward.

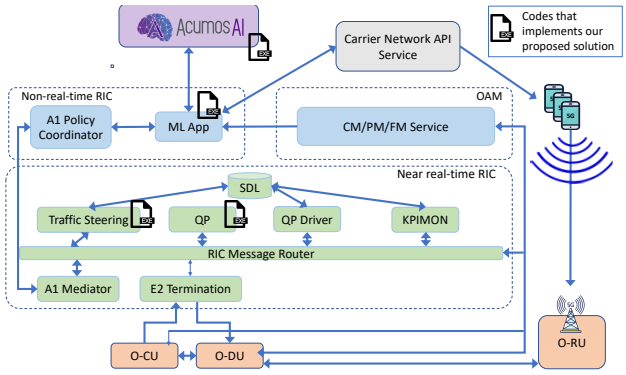


Fig. 11: Implementation roadmap of Section III's framework using OSC Cherry release software, showing the architecture components where our proposed solution would reside.

management service to collect the resource allocation results from the traffic steering module, and the side information from the QP modules. A ML app running in the non-real-time RIC platform creates a signal optimization problem instance as in (1), and offloads it to an off-site ML cluster based on the Linux foundation Acumos platform [65]. The results will be collected by the ML app and later sent to the traffic steering module through the A1 interface. Meanwhile, the control signals will also be available to user applications through the 5G carrier's network API service (e.g., Verizon Thingspace API [66], AT&T API marketplace [67]).

VI. Concluding Remarks and Other Use-Cases

Taking advantage of the new 5G O-RAN architecture with disaggregated functionalities, we design a two-tier closed-loop control framework that can learn the outer-loop signals to adjust user behaviors and utilize side information from the near real-time inner loop resource allocation routine, together optimizing the long-term aggregate QoS performance. Our design is among the first to embed this architecture logic into a learning framework, with proved performance guarantees. We propose a combinatorial multi-armed bandits algorithm, which can learn the latent structure of the reward across different resource dimensions from only the realized samples of the total reward. A novel integration of a gradient-based subroutine effectively improves the learning optimality by carefully using the side information. In the future, we will adapt our

combinatorial learning methods to optimize other types of controls in various networks, where high-dimensional uncertain inputs have latent effects on the optimization objectives and arbitrary dependencies with the control variables, including the optimiztion of taxi mobility and resource placement/planning of the edge plus cloud network.

A. Other Potential Use-cases

Network slicing is one of the key techniques to provide service-oriented architectures for 5G networks. A network slice is typically a virtual network along with logically isolated resources built on top shared physical resources [4]. Such slicing then requires both inter- and intra-slice resource allocation, which should satisfy a set of complex constraints that characterize different requirements of both service providers (e.g., low latency, high bandwidth, and ultra-high reliability for a critical IoT application) and infrastructure providers (e.g., physical resource constraints) [68]. Our two-tiered learning framework can simplify the bi-level inter- and intra-slice management problem by providing an outer loop learning algorithm for regulating the demands of different slices that share the same physical infrastructure, with an inner loop that schedules users within the slice and constructs side information for the outer loop. To further decouple intra-slice management across different slices, our method can be extended to multiple inner loop controllers that run in parallel at different slices.

Serverless computing technologies (e.g., Amazon's AWS Lambda[3] and Microsoft's Azure serverless[4]) enable today's cloud computing platforms to automatically provision and scale resources for applications running in the cloud, abstracting this infrastructure provisioning from application developers. Eliminating the overhead of users' infrastructure management allows serverless applications to run faster than traditional cloud services. Realizing these benefits, however, requires new control intelligence that can fit the *dual timescale* interaction between the resource scaling run at a smaller timescale for specific users and the platform improvement (e.g., by adjusting the offered resource prices or performing user admission control) run at a longer timescale. This outer loop can further leverage event-triggered application feedback on resource utilization from the inner loop [69]. Our proposed control framework addresses these challenges: our control signals can be regarded as the real prices of serverless computing resources. Cloud providers can apply any inner-loop resource allocation rule, while our method designs the optimal pricing strategy in the outer loop, using side information from the inner loop resource allocation.

VII. Acknowledgement

This work was supported in part by NSFC grants (No. 62102460, No. U20A20159, and No. 61972432), Guangzhou Science and Technology Plan Project (No. 202201011392), Guangdong Basic and Applied Basic Research Foundation (No. 2023A1515012982 and No. 2021B151520008), Young Outstanding Award under the Zhujiang Talent Plan of Guangdong Province, ONR grant W911NF1910036, and NSF grant 21-03024.

[3]aws.amazon.com/serverless/

[4]azure.microsoft.com/en-us/solutions/serverless/

References

[1] B. Agarwal, M. A. Togou, M. Ruffini, S. Member, and G.-M. Muntean, "A comprehensive survey on radio resource management in 5g hetnets: Current solutions, future trends and open issues," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 4, 2022.

[2] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek, "A resource allocation model for qos management," in *Proceedings Real-Time Systems Symposium*. IEEE, 1997, pp. 298–307.

[3] M. Harishankar, S. Pilaka, P. Sharma, N. Srinivasan, C. Joe-Wong, and P. Tague, "Procuring spontaneous session-level resource guarantees for real-time applications: An auction approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 7, pp. 1534–1548, 2019.

[4] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi, "5g wireless network slicing for embb, urllc, and mmtc: A communication-theoretic view," *IEEE Access*, vol. 6, pp. 55 765 – 55 779, 2018.

[5] C.-Y. Chang, N. Nikaein, and T. Spyropoulos, "Radio access network resource slicing for flexible service execution," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2018.

[6] "A multi-dimensional intelligent multiple access technique for 5g beyond and 6g wireless networks."

[7] F. Zhou, P. Yu, L. Feng, X. Qiu, Z. Wang, L. Meng, M. Kadoch, L. Gong, and X. Yao, "Automatic network slicing for iot in smart city," *IEEE Wireless Communications*, vol. 27, no. 6, pp. 108 – 115, 2020.

[8] P. L. Vo, M. N. H. Nguyen, T. A. Le, and N. H. Tran, "Slicing the edge: Resource allocation for ran network slicing," *IEEE Wireless Communications Letters*, vol. 7, no. 6, pp. 970–973, 2018.

[9] K. Liang, L. Zhao, X. Chu, and H. Chen, "An integrated architecture for software defined and virtualized radio access networks with fog computing," *IEEE Network*, vol. 31, no. 1, pp. 80–87, 2017.

[10] S. Niknam, A. Roy, H. S. Dhillon, S. Singh, R. Banerji, J. H. Reed, N. Saxena, and S. Yoon, "Intelligent o-ran for beyond 5g and 6g wireless networks."

[11] A. S. Abdalla, P. S. Upadhyaya, V. K. Shah, and V. Marojevic, "Toward next generation open radio access networks—what o-ran can and cannot do!" *IEEE Network Magazine*, vol. early access, 2022.

[12] "O-ran: Towards an open and smart ran," https://www.o-ran.org/s/O-RAN-WP-FInal-181017.pdf, 2018.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks (alexnet)," in *Proc. of NIPS*, 2013.

[14] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Foundation and Trends in Machine Learning*, vol. 5, no. 1, pp. 1–22, 2012.

[15] R. Lu, S. H. Hong, and X. Zhang, "A dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach," *Applied Energy*, vol. 220, pp. 220–230, 2018.

[16] X. Zhang, C. Wu, Z. Huang, and Z. Li, "Occupation oblivious pricing of cloud jobs via online learning," in *Proc. of INFOCOM*, 2018.

[17] C. He, Y. Wu, M. Huang, S. Feng, and F. Shu, "Machine learning-based two-stage task offloading optimization for power distribution internet of things," *Wireless Communications and Mobile Computing*, no. 3365279, 2022.

[18] R. Combes, M. S. Talebi, A. Proutiere, and M. Lelarge, "Combinatorial bandits revisited," in *Proc. of NIPS*, 2015.

[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[20] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework, results and applications," in *Proc. of ICML*, 2013.

[21] B. Kveton, Z. Wen, A. Ashkan, and C. Szepesvari, "Tight regret bounds for stochastic combinatorial semi-bandits," in *Proc. of AISTATS*, 2015.

[22] J. Langford and T. Zhang, "The epoch-greedy algorithm for contextual multi-armed bandits," in *Proc. of NIPS*, 2007.

[23] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. of WWW*, 2010.

[24] L. Qin, S. Chen, and X. Zhu, "Contextual combinatorial bandit and its application on diversified online recommendation," in *Proc. of SDM*, 2014.

[25] A. Nika, S. Elahi, and C. Tekin, "Contextual combinatorial volatile multi-armed bandit with adaptive discretization," in *International Conference on Artificial Intelligence and Statistics*, 2020, pp. 1486–1496.
[26] N. Hassan, K. Yau, and C. Wu, "Edge computing in 5g: A review," *IEEE Access*, vol. 7, pp. 127 276 – 127 289, 2019.
[27] Z. Zhou, Q. Wu, and X. Chen, "Online orchestration of cross-edge service function chaining for cost-efficient edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1866–1880, 2019.
[28] J. Mei, X. Wang, and K. Zheng, "An intelligent self-sustained ran slicing framework for diverse service provisioning in 5g-beyond and 6g networks," *Intelligent and Converged Networks*, vol. 1, no. 3, pp. 281 – 294, 2020.
[29] M. Kaneko, T. Nakano, K. Hayashi, T. Kamenosono, and H. Sakai, "Distributed resource allocation with local csi overhearing and scheduling prediction for ofdma heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1186–1199, 2017.
[30] N. Mokari, F. Alavi, S. Parsaeefard, and T. Le-Ngoc, "Limited-feedback resource allocation in heterogeneous cellular networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2509–2521, 2016.
[31] S. T. X. Huang, D. Zhang, "Fairness-based distributed resource allocation in two-tier heterogeneous networks," *IEEE Access*, vol. 7, p. 40000–40012, 2019.
[32] D. Gao, Z. Liang, H. Zhang, O. Dobre, and G. Karagiannidis, "Stackelberg game-based energy efficient power allocation for heterogeneous noma networks," in *Proc. of GLOBECOM*, 2018.
[33] J. Tang, D. K. C. So, and E. Alsusa, "Energy-efficient heterogeneous cellular networks with spectrum underlay and overlay access," *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2439–2453, 2018.
[34] X. Sun and S. Wang, "Resource allocation scheme for energy saving in heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 8, p. 4407–4416, 2015.
[35] K. Wang, K. Yang, and C. S. Magurawalage, "Joint energy minimization and resource allocation in c-ran with mobile cloud," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 760–770, 2018.
[36] N. Liu, Z. Li, J. Xu, Z. Xu, S. Lin, Q. Qiu, J. Tang, Y. Wang, and Y. Wang, "A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning," in *Proc. of IEEE ICDCS*, 2017.
[37] Y. Liu and M. Liu, "An online approach to dynamic channel access and transmission scheduling," in *Proc. of MobiHoc*, 2015.
[38] J. Zuo, X. Zhang, and C. Joe-Wong, "Observe before play: Multi-armed bandit with pre-observations," in *Proc. of AAAI*, 2020.
[39] S. Agrawal and N. R. Devanur, "Linear contextual bandits with knapsacks," in *Proc. of NIPS*, 2016.
[40] K. A. Sankararaman and A. Slivkins, "Combinatorial semi-bandits with knapsacks," in *Proc. of AISTATS*, 2018.
[41] S. Gupta, J. Zuo, C. Joe-Wong, G. Joshi, and O. Yagan, "Correlated combinatorial bandits for online resource allocation," in *In Proc. of MobiHoc*, 2022.
[42] V. Dani, T. P. Hayes, and S. M. Kakade, "Stochastic linear optimization under bandit feedback," in *Proc. of COLT*, 2008.
[43] J. Kong, Z.-Y. Wu, M. Ismail, E. Serpedin, and K. A. Qaraqe, "Q-learning based two-timescale power allocation for multi-homing hybrid rf/vlc networks," *IEEE Wireless Communications Letters*, vol. 9, no. 4, 2020.
[44] A. Ortiz, A. Asadi, M. Engelhardt, A. Klein, and M. Hollick, "Cbmos: Combinatorial bandit learning for mode selection and resource allocation in d2d systems," *IEEE Journal on Selected Areas in Communications,*, vol. 37, no. 10, 2019.
[45] W. Chu, L. Li, L. Reyzin, and R. E. Schapire, "Contextual bandits with linear payoff functions," in *Proc. of AISTATS*, 2011.
[46] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang, and L.-C. Wang, "Deep reinforcement learning for mobile 5g and beyond: Fundamentals, applications, and challenges," *IEEE Veh. Tech. Magazine*, vol. 14, no. 2, pp. 44–52, 2019.
[47] X. Foukas, M. K. Marina, and K. Kontovasilis, "Iris: Deep reinforcement learning driven shared spectrum access architecture for indoor neutral-host small cells," *IEEE JSAC*, vol. 37, no. 8, pp. 1820–1837, 2019.
[48] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.
[49] C. Westphal, "IETF presentation: Challenges in networking to support augmented reality and virtual reality," 2017.
[50] Y. Xu, G. Gui, M. L. Guoquan Li, H. Gacanin, and F. Adachi, "A survey on resource allocation for 5g heterogeneous networks: Current research, future trends and challenges," *IEEE Communications Surveys and Tutorials*, 2020.
[51] S. Fu, J. Gao, and L. Zhao, "Collaborative multi-resource allocation in terrestrial-satellite network towards 6g," *IEEE Transactions on Wireless Communications*, vol. 20, no. 11, pp. 7057–7071, 2021.
[52] S. He, Z. An, J. Zhu, J. Zhang, Y. Huang, and Y. Zhang, "Beamforming design for multiuser urllc with finite blocklength transmission," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 8096–8109, 2021.
[53] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *IEEE Network*, vol. 24, no. 2, pp. 36–41, 2010.
[54] X. Zhang, Z. Huang, C. Wu, Z. Li, and F. C. Lau, "Online auctions in iaas clouds: Welfare and profix maximization with server costs," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 1034–1047, 2017.
[55] ——, "Online stochastic buy-sell mechanism for vnf chains in the nfv market," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 392–406, 2017.
[56] A. Singla and A. Krause, "Truthful incentives in crowdsourcing tasks using regret minimization mechanisms," in *Proc. of WWW*, 2013.
[57] S. Boyd, "Ellipsoid method," https://stanford.edu/class/ee364b/lectures/ellipsoid_method_notes.pdf, 2018.
[58] A. Vinel, "Mathematical programming techniques for solving stochastic optimization problems with certainty equivalent measures of risk," pp. xii, 184 pages, 2015.
[59] S. Kalloori and F. Ricci, "Improving cold start recommendation by mapping feature-based preferences to item comparisons," in *In Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, 2017.
[60] S. Kalloori and T. Li, "Modeling user preferences using relative feedback for personalized recommendations," in *The Thirty-Third International FLAIRS Conference*, 2019.
[61] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, pp. 4–22, 1985.
[62] D. Raca, D. Leahy, C. J. Sreenan, and J. J. Quinlan, "Beyond throughput, the next generation: a 5g dataset with channel and context metrics," in *Proceedings of the 11th ACM Multimedia Systems Conference*, 2020, pp. 303–308.
[63] C. Joe-Wong, S. Sen, and S. Ha, "Sponsoring mobile data: Analyzing the impact on internet stakeholders," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1179–1192, 2018.
[64] "O-ran software community," https://o-ran-sc.org, 2020.
[65] "Linux foundation acumos," https://www.acumos.org/, 2020.
[66] "Verizon thingspace api," https://thingspace.verizon.com/resources/documentation/connectivity/API_Reference, 2020.
[67] "At&t api marketplace," https://apimarket.att.com, 2020.
[68] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.
[69] Z. Al-Ali, S. Goodarzy, E. Hunter, S. Ha, R. Han, E. Keller, and E. Rozner, "Making serverless computing more serverless," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*. IEEE, 2018, pp. 456–459.

Xiaoxi Zhang (S'13, M'18) is an associate professor with the School of Computer Science and Engineering, Sun Yat-sen University. Before joining SYSU, she was a Postdoctoral researcher of the Department of Electrical and Computer Engineering at Carnegie Mellon University. She received her B.E. degree in Electronics and Information Engineering from the Huazhong University of Science and Technology and Ph.D. degree in Computer Science from The University of Hong Kong in 2013 and 2017, respectively. She is broadly interested in optimization and algorithm design for networked systems, including cloud and edge computing networks, NFV systems, and distributed machine learning systems.

Jinhang Zuo (M'22) is a joint postdoc at University of Massachusetts Amherst and California Institute of Technology. He received his Ph.D. in Electrical and Computer Engineering from Carnegie Mellon University in 2022. His main research interests include online learning, networked systems, and resource allocation. He was a recipient of the CDS Postdoctoral Fellowship from UMass Amherst and the Carnegie Institute of Technology Dean's Fellowship.

Zhe Huang (M'13) received the BEng, MPhil, and PhD degrees, all from the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong. He was a postdoctoral research scholar with the Department of Electrical Engineering, Princeton University. He joined AT&T Labs Research, in 2016. He has been actively contributing to Linux foundation Open Network Automation Platform project and Linux foundation Akraino edge computing platform project. He worked at Amazon Web Services at 2021 and joined Google in 2022. His research interests include edge/fog computing, cloud computing, network function virtualization, network automation, and 5G networks.

Zhi Zhou (M'18) received the B.S., M.E., and Ph.D. degrees in 2012, 2014, and 2017, respectively, all from the School of Computer Science and Technology at Huazhong University of Science and Technology (HUST), Wuhan, China. He is currently an associate professor in the School of Data and Computer Science at Sun Yat-sen University, Guangzhou, China. In 2016, he was a visiting scholar at University of Göttingen. He was nominated for the 2019 China Computer Federation CCF Outstanding Doctoral Dissertation Award, the sole recipient of the 2018 ACM Wuhan & Hubei Computer Society Doctoral Dissertation Award, and a recipient of the Best Paper Award of IEEE UIC 2018. His research interests include edge computing, cloud computing, and distributed systems.

Xu Chen (M'12, SM'20) Xu Chen received the Ph.D. degree in information engineering from the Chinese University of Hong Kong in 2012. He is a Full Professor with Sun Yat-sen University, Guangzhou, China, Director of Institute of Advanced Networking and Computing Systems, and the Vice Director of the National and Local Joint Engineering Laboratory of Digital Home. He was a Post-Doctoral Research Associate with Arizona State University, Tempe, USA, from 2012 to 2014, and a Humboldt Scholar Fellow with the Institute of Computer Science, University of Goettingen, Germany, from 2014 to 2016. He was a recipient of the Prestigious Humboldt Research Fellowship awarded by Alexander von Humboldt Foundation of Germany, the 2014 Hong Kong Young Scientist Runner-Up Award, the 2020 IEEE Computer Society Best Paper Awards Runner-Up, the 2017 IEEE Communication Society Asia–Pacific Outstanding Young Researcher Award, the 2017 IEEE ComSoc Young Professional Best Paper Award, the Honorable Mention Award of 2010 IEEE international conference on Intelligence and Security Informatics, the Best Paper Runner-Up Award of 2014 IEEE International Conference on Computer Communications (INFOCOM), and the Best Paper Award of 2017 IEEE International Conference on Communications. He is currently an Area Editor of IEEE Open Journal of the Communications Society, an Associate Editor of the IEEE Transactions Wireless Communications, IEEE Internet of Things Journal, IEEE Transactions on Vehicular Technology, and IEEE Journal on Selected Areas in Communications (JSAC) Series on Network Softwarization and Enablers.

Carlee Joe-Wong (S'11, M'16, SM'22) is the Robert E. Doherty Associate Professor of Electrical and Computer Engineering at Carnegie Mellon University. She received her A.B. degree (magna cum laude) in Mathematics, and M.A. and Ph.D. degrees in Applied and Computational Mathematics, from Princeton University in 2011, 2013, and 2016, respectively. Her research interests lie in optimizing various types of networked systems, including applications of machine learning and pricing to cloud computing, mobile/wireless networks, and ridesharing networks. From 2013 to 2014, she was the Director of Advanced Research at DataMi, a startup she co-founded from her research on mobile data pricing. She received the NSF CAREER award in 2018 and the ARO Young Investigator award in 2019.