

Resource Pooling Optimization for O-Cloud with Online Learning

Juhee Shin, Hyun-Min Yoo, Een-Kee Hong

Dept. of Electronics and Information Convergence Engineering

Kyung Hee University, Yongin, Republic of Korea

Emails: odong3094@khu.ac.kr, yhm1620@khu.ac.kr, ekhong@khu.ac.kr

Abstract—Running virtualized radio access networks (vRAN) over open cloud (O-Cloud) infrastructure is a promising and increasingly adopted direction, as it enables flexible, software-based sharing of general-purpose computing resources across base stations. However, conventional peak-based server dimensioning leaves the majority of these computational resources idle during off-peak hours, undermining the very efficiency that pooling promises. This paper presents a traffic diversity (TD)-driven resource pooling framework operating at two levels. At the gNB level, TD clustering groups virtualized gNB functions whose traffic peaks are mutually complementary. This flattens the aggregate demand of each shared O-Cloud pool and thereby lowers the compute capacity that must be provisioned to meet a target service-level agreement (SLA). At the platform level, LB-FTRL extends the standard follow-the-regularized-leader (FTRL) objective with a quadratic overload penalty. It distributes the pooled capacity across co-located application domains, so that RAN overload is suppressed while freed capacity is reclaimed by AI training workloads. A drift-triggered reconfiguration mechanism based on spectral multiplexing gain (SMG) keeps pools effective as traffic evolves. The pipeline is realized as an O-RAN rApp deployed through rappmanager. On real-world traffic from a Korean mobile network operator (MNO), the framework reduces the peak-to-average ratio (PAR) by 32.6% and the SLA-constrained capacity requirement by 17.4%, while cutting RAN overload incidents by 81.1%.

Index Terms—O-RAN, O-Cloud, resource pooling, online learning, FTRL, vRAN, closed-loop automation, rApp

I. INTRODUCTION

The commercialization of 5G and the prospective deployment of 6G networks have accelerated the transition from dedicated hardware base stations toward virtualized RAN (vRAN) architectures. In these architectures, baseband processing executes as software on general-purpose servers under the O-RAN Alliance framework [1]. Running such vRAN workloads over open cloud (O-Cloud) infrastructure has emerged as a promising and increasingly adopted direction, since the O-Cloud serves as a shared computing substrate that can dynamically reallocate processing resources across base stations, improving flexibility and hardware efficiency [2]–[4]. In practice, however, operators dimension O-Cloud servers for the worst-case peak traffic of individual gNBs, causing substantial and persistent underutilization that undermines this potential [5].

Real-world traffic collected from a Korean mobile network operator (MNO) deployment in a metropolitan business district of Seoul exhibits a peak-to-average ratio (PAR) of 3.56 when

each gNB is considered in isolation. Provisioning every gNB for its own peak therefore requires roughly 2.2 times its mean load, and the resulting capacity sits idle for much of the day. This translates directly into avoidable capital expenditure (CAPEX) and total cost of ownership (TCO). The economic case for pooling is also recognized at the industry level. The SK Telecom and NTT DOCOMO joint white paper [6] projects that horizontal pooling can reduce the required server count to as low as 26% of the unshared baseline, with commensurate power savings [7].

The insight enabling pooling is that gNBs serving different land-use types, such as business, residential, and entertainment zones, peak at systematically different times. Assigning gNBs with complementary peak profiles to the same O-Cloud pool yields an aggregate load far smoother than any individual load, allowing fewer servers to meet the combined demand without service-level agreement (SLA) degradation.

Translating this insight into operational practice requires resolving three technical challenges: (i) how to identify groups of gNBs with sufficiently complementary traffic patterns; (ii) how to continuously optimize the resource allocation policy within each pool under non-stationary traffic; and (iii) how to propagate allocation decisions to the physical infrastructure automatically and in conformance with O-RAN standards. This paper addresses all three challenges within a unified framework. Our principal contributions are summarized as follows:

- **TD Clustering**: a traffic-diversity-aware clustering algorithm that forms gNB pools by jointly optimizing coefficient of variation (CV), PAR, and inter-cell traffic complementarity.
- **LB-FTRL**: a variant of the follow-the-regularized-leader (FTRL) algorithm [8], [9] that incorporates a quadratic overload penalty, balancing resource efficiency against load imbalance across heterogeneous O-Cloud namespaces.
- **Adaptive Pool Reconfiguration**: a prediction-guided reconfiguration policy that uses SMG drift detection combined with autoregressive (AR(24)) or long short-term memory (LSTM) forecasting to suppress spurious reconfigurations.
- **End-to-End Standards-Compliant rApp**: a complete O-RAN rApp implementation deployed through rappmanager, integrating R1 model registration and live Kafka-

based performance management (PM) data ingestion.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. Deployment Architecture and Scope of Pooling

We consider the O-RAN split deployment in which the O-RUs remain distributed at the cell sites, while the O-DU and O-CU functions of every gNB are virtualized and hosted centrally on a shared O-Cloud cluster. Because the virtualized O-DU/O-CU instances of many gNBs are co-located on the same physical servers, their compute demand can be served from a common capacity pool rather than from per-gNB dedicated hardware.

Pooling in this work operates at two distinct levels, and the two are complementary rather than alternatives.

Level 1 (gNB-level pooling). TD clustering partitions the N virtualized gNBs into K pools. Each pool is a group of vDU/vCU instances that share one capacity envelope. Because the pool is dimensioned for the *aggregate* demand of its members rather than for the sum of their individual peaks, grouping gNBs whose peaks occur at different times directly reduces the capacity that must be provisioned.

Level 2 (platform-level allocation). The compute capacity of the O-Cloud cluster is shared not only by RAN processing but also by co-located application domains: network management (SMO), AI/ML training, and RIC applications. LB-FTRL distributes the cluster capacity across these domains over time. Its role is not to reduce the provisioned capacity, but to ensure that the capacity freed by Level-1 flattening is neither wasted nor allowed to starve the RAN domain during peaks.

B. O-Cloud Resource Model

We model the O-Cloud cluster as J processing unit (PU) namespaces: RAN processing (ran), network management (smo), AI/ML training (traininghost), and RIC applications (nonrtrc). The total available CPU capacity is C_{total} , and the allocation vector $\mathbf{x} = [x_1, \dots, x_J]$ satisfies $\sum_j x_j = 1$, $x_j \geq 0$. At each time step t , the normalized load of pool i is $\ell_i^{(t)} \in [0, 1]$. The instantaneous overload of namespace j is $\max(0, \sum_i x_{ij} \ell_i^{(t)} - c_j)$, where c_j denotes the capacity fraction of namespace j .

C. Pool Aggregate Load and Capacity Provisioning

Let $\mathcal{C} = \{C_1, \dots, C_K\}$ be a partition of N gNBs into K pools. The aggregate load of pool k at time t is $L_k^{(t)} = \sum_{n \in C_k} \ell_n^{(t)}$, and its PAR is $\text{PAR}_k = \max_t L_k^{(t)} / \bar{L}_k$, where $\bar{L}_k$ is the time-averaged load.

To make the pooling gain concrete, we define the provisioned capacity of a load series as the smallest multiple γ of its mean load that keeps the SLA violation rate below a target ϵ . The SLA violation rate is the fraction of time steps whose demand exceeds the provisioned capacity:

$$\gamma(\ell) = \min \left\{ \gamma \mid \frac{1}{T} \sum_{t=1}^T \mathbf{1}[\ell^{(t)} > \gamma \bar{\ell}] \leq \epsilon \right\}, \quad (1)$$

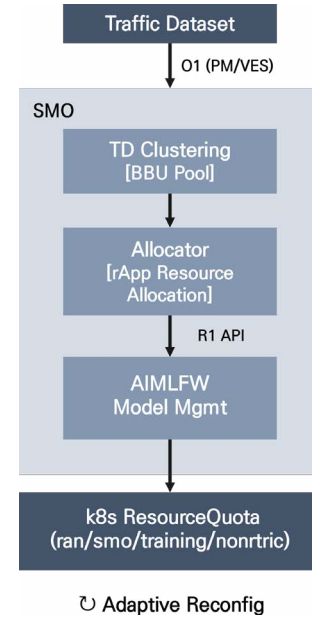


Fig. 1. End-to-end O-Cloud resource optimization pipeline.

with $\epsilon = 5\%$ throughout this paper. The objective of pooling is to find a partition $\mathcal{C}$ that minimizes the mean provisioned capacity $\frac{1}{K} \sum_k \gamma(L_k)$, since γ is what an operator must actually purchase.

D. Online Resource Allocation

Given a fixed pool configuration, the platform-level allocation problem is formulated as a sequence of decisions $\mathbf{x}^{(t)}$ that minimize the cumulative regret:

$$R_T = \sum_{t=1}^T f(\mathbf{x}^{(t)}) - \min_{\mathbf{x}} \sum_{t=1}^T f(\mathbf{x}), \quad (2)$$

where f is the per-round loss. The non-stationarity of traffic makes offline optimization infeasible, motivating an online learning approach [8], [10]. Fairness across competing tenants on a virtualized O-RAN platform has likewise been studied as an online allocation problem [11].

III. PROPOSED FRAMEWORK

Fig. 1 shows the end-to-end pipeline. The traffic dataset feeds TD clustering, which forms baseband unit (BBU) pools inside the service management and orchestration (SMO). LB-FTRL then computes the allocation policy. This policy is registered to the AI/ML Framework (AIMLFW) model management service (ModelMgmtService) over the R1 interface, and finally enforced as Kubernetes ResourceQuota objects across the four O-Cloud namespaces.

A. TD Clustering

Standard K-means clustering groups gNBs by load magnitude. This tends to co-locate peak loads and therefore amplifies rather than attenuates aggregate PAR. TD clustering instead uses a three-dimensional diversity feature vector

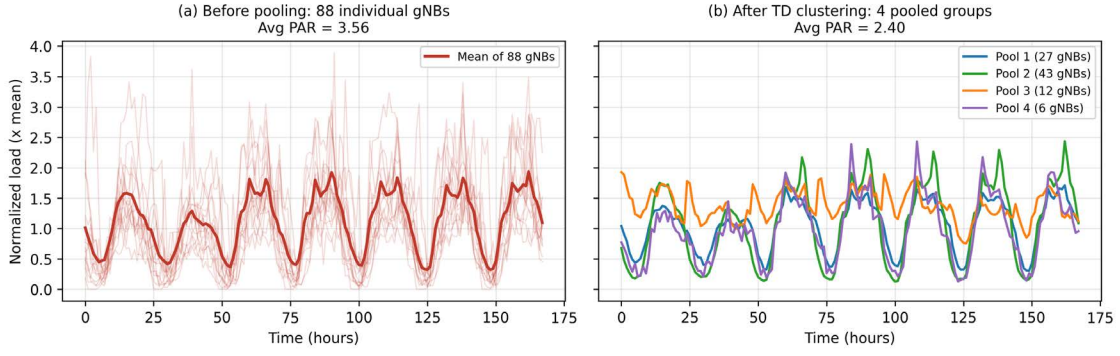


Fig. 2. Aggregate load before and after TD clustering. The four pools are markedly flatter than individual gNBs, lowering the capacity to be provisioned.

$\mathbf{v}_n = [\hat{\sigma}_n, \hat{\rho}_n, \hat{\phi}_n]$. Here, $\hat{\sigma}_n$, $\hat{\rho}_n$, and $\hat{\phi}_n$ are the min-max normalized CV, PAR, and mean pairwise Pearson correlation of gNB n with all other gNBs, respectively. K-means is then applied to $\{\mathbf{v}_n\}$ with $K = 4$ clusters. By incorporating inter-cell correlation explicitly, TD clustering tends to produce pools whose member gNBs peak at different times, yielding a flat aggregate profile.

B. LB-FTRL

Standard FTRL solves the online convex optimization problem in (2) by setting $f(\mathbf{x}) = -\eta(\mathbf{x})$, where η is a system efficiency metric [8]. Under this formulation, the algorithm tends to concentrate load on the highest-efficiency namespace. This induces severe overload on the RAN processing namespace while AI training resources remain idle, a configuration that is inconsistent with the AI-RAN convergence trajectory articulated in [6], [12].

LB-FTRL modifies the per-round objective to:

$$f_{\text{LB}}(\mathbf{x}) = -\eta(\mathbf{x}) + \mu \sum_{j=1}^J \left[\max \left(0, \sum_i x_{ij} \ell_i - c_j \right) \right]^2, \quad (3)$$

where $\mu = 0.3$ is a penalty strength coefficient selected through a comparative sweep across $\mu \in \{0.1, 0.5\}$. The quadratic penalty imposes a superlinear cost for severe overloads, steering the online learner away from allocations that would saturate any single namespace. The allocation is then updated by minimizing the cumulative loss with a proximal regularization term over the probability simplex, following the standard FTRL update rule [9]. The augmented objective f_{LB} remains convex and bounded over the simplex. As a result, LB-FTRL retains the sub-linear $\mathcal{O}(\sqrt{T})$ regret guarantee of standard FTRL [8], [9], so the overload penalty improves load balancing without degrading the asymptotic convergence rate.

C. Adaptive Pool Reconfiguration

As traffic patterns evolve over time, the pool configuration produced by TD clustering may become suboptimal. We

quantify pooling effectiveness via the spectral multiplexing gain (SMG):

$$\text{SMG}(C_k) = 1 - \frac{\text{CV}(\sum_{n \in C_k} \ell_n)}{\frac{1}{|C_k|} \sum_{n \in C_k} \text{CV}(\ell_n)}, \quad (4)$$

where $\text{CV}(\cdot)$ denotes coefficient of variation. A higher SMG indicates that pooling provides greater variance reduction relative to individual gNB variance.

Drift is declared when the current SMG drops more than 15% relative to the reference SMG measured at the time of pool formation, i.e., $(\text{SMG}_{\text{ref}} - \text{SMG}^{(t)})/\text{SMG}_{\text{ref}} > 0.15$. This threshold is set operationally rather than derived analytically. Upon drift detection, the algorithm invokes an automatically selected prediction model to forecast future traffic over a 24-hour horizon. If the predicted CV is less than 90% of the current value, the drift is classified as transient and reconfiguration is suppressed. Otherwise, greedy reconfiguration is performed by relocating the gNB that maximizes SMG improvement. The predictor is chosen from traffic characteristics: AR(24) if the 24-lag autocorrelation exceeds 0.8, indicating strong diurnal periodicity, and an LSTM network if the mean CV exceeds 0.5.

D. End-to-End Closed-Loop Automation

The allocation policy produced by LB-FTRL is registered to the AIMLFW ModelMgmtService via the O-RAN R1 interface every 10 time steps. A daemon process (pool_actuator) polls the R1 model discovery endpoint at 30-second intervals, computes Kubernetes ResourceQuota manifests proportional to the allocation vector, and applies them to each namespace. The rApp is packaged as a TOSCA-compliant cloud service archive (CSAR) and deployed through the O-RAN software community (SC) rappmanager, reaching PRIMED and DEPLOYED lifecycle states. A Kafka consumer thread also subscribes to the pmreports topic, feeding live PM data into an online LB-FTRL update path.

The initial pool configuration and LB-FTRL training are performed once at deployment time, completing in approximately 30 seconds. Thereafter, the closed-loop policy update cycle—comprising R1 model registration, pool_actuator

polling, and ResourceQuota actuation—incur a worst-case latency of 32 seconds per cycle, consistent with the Non-RT RIC control timescale defined by the O-RAN Alliance.

IV. EXPERIMENTAL EVALUATION

A. Dataset and Setup

The framework is evaluated on a measurement dataset from a Korean MNO covering 88 commercial 5G NR gNBs deployed across a metropolitan business district of Seoul, a heterogeneous area combining commercial, residential, and entertainment zones. Each gNB operates on the n78/n79 bands with 100 MHz bandwidth and four MIMO layers.

Per-gNB PM counters report the downlink and uplink data volume delivered in each collection interval. The traffic load $\ell_n^{(t)}$ used throughout the evaluation is the total (downlink plus uplink) transported byte volume, sampled at a one-hour granularity over 720 consecutive hours (30 days), giving a 88×720 load matrix. Loads are min-max normalized per gNB before clustering so that pool formation reflects the *shape* of each traffic profile rather than absolute cell size. Across the 88 gNBs, the mean downlink rate is 78.2 Mbps and the mean peak downlink rate is 288.2 Mbps, confirming the strongly peaked profile that motivates this work. The first 360 hours are used to form the pools and the remaining 360 hours to evaluate them.

The O-RAN testbed runs on a Kubernetes cluster with four namespaces (ran, smo, traininghost, nonrtic), hosting O-RAN SC components including SMO, AIMLFW, Non-RT RIC, and rappmanager.

B. Pooling Gain: Where the Saving Comes From

Fig. 2 contrasts the load seen before and after TD clustering. Individually, the 88 gNBs are sharply peaked (average PAR 3.56, CV 0.66). After clustering, the four pools—containing 27, 43, 12, and 6 gNBs—exhibit an average PAR of 2.40 and CV of 0.50, corresponding to a 32.6% PAR reduction and a 24.5% CV reduction. Pool 3 in particular peaks out of phase with the others, which is precisely the complementarity that TD clustering is designed to exploit.

The practical consequence is a smaller capacity purchase. Applying the provisioning rule of (1) with $\epsilon = 5\%$, an unpooled gNB must be provisioned at $\gamma = 2.17 \times$ its mean load, whereas a TD-formed pool needs only $\gamma = 1.80 \times$. The compute capacity—and hence the number of O-Cloud servers—that must be installed to serve the same traffic under the same SLA therefore falls by 17.4%. This is the CAPEX saving reported in Table I. Equally important for operators, the standard deviation of γ across sites drops from 0.39 to 0.16, a 58.4% reduction. Pooled demand is thus not only lower but far more predictable, so less safety margin must be held.

We stress what this saving is *not*. It is a dimensioning gain, obtained because the pooled peak is lower than the sum of individual peaks; it does not assume that servers are powered down at run time. We also estimate TCO with a simple power model (30 W idle, up to 100 W under load, USD 0.10/kWh). This yields a further OPEX benefit and a combined 16.9%

TABLE I
O-CLOUD RESOURCE OPTIMIZATION RESULTS (KOREAN MNO)

Metric	Baseline	Proposed
<i>Pooling efficiency</i>		
PAR (mean)	3.56	2.40
CV (mean)	0.66	0.50
Idle time ratio	38.2%	11.3%
<i>Cost</i>		
Provisioned capacity γ	$2.17 \times$	$1.80 \times$
Capacity spread (std)	0.39	0.16
Relative TCO	220.1	182.9
<i>Resource allocation</i>		
RAN overload (vs. FTRL)	1.00	0.19
traininghost utilization	$\approx 0\%$	48.9%
LB-FTRL final regret	—	0.4356
Closed-loop cycle latency	—	32 s

TABLE II
POOL RECONFIGURATION STRATEGY COMPARISON

Strategy	SMG Mean	SMG Std	Reconfigs
Static	0.2252	0.0255	0
Periodic	0.2252	0.0255	1
Threshold	0.2263	0.0769	14
Adaptive+AR(24)	0.2322	0.0201	2

TCO reduction, but the dominant term remains the reduced capacity that must be installed in the first place.

C. Platform-Level Allocation

Given the flattened pools, LB-FTRL allocates cluster capacity across the four namespaces. Relative to the standard FTRL baseline, it reduces mean RAN overload by 81.1% while raising traininghost utilization from near-zero to 48.9%. The quadratic penalty in (3) is what produces this behavior: standard FTRL concentrates allocation on the most efficient namespace and starves the others, whereas the penalty makes saturating any single namespace superlinearly costly. The freed RAN capacity is thus reclaimed by AI training rather than left idle, which is the co-scheduling behavior anticipated in [6].

D. Adaptive Reconfiguration

Table II compares four pool management strategies on the 360-hour evaluation window. The threshold strategy reconfigures whenever SMG drops at all: it performs 14 reconfigurations yet ends with a *lower* mean SMG and a three-fold higher variance than the adaptive strategy. This is not a paradox but the cost of churn—each reconfiguration disrupts the pool and itself induces a transient SMG dip. The adaptive strategy detects 6 drift events, classifies 4 of them as transient using the forecast test, and reconfigures only twice, achieving the highest mean SMG (0.2322) with the lowest variance (0.0201). Prediction accuracy supports this. On this dataset the 24-lag autocorrelation exceeds 0.8, so the selector picks AR(24). AR(24) attains a mean absolute error of 0.061 in 0.01 s of training—better than LSTM (0.091, 31 s) and TCN (0.125, 2 s), confirming that a lightweight linear model suffices for periodic urban traffic.

E. Hybrid O-Cloud Deployment

Finally, we validated a hybrid configuration in which the AIMLFW ModelMgmtService is hosted on an AWS EC2 instance (t3.medium, Seoul region) while all RAN and SMO components remain on-premise. Transitioning between the local and cloud-hosted endpoint requires no code modification; updating the R1_BASE environment variable is sufficient, demonstrating that the R1 interface abstraction provides genuine deployment independence. Because raw traffic data is retained on-premise and only derived model artifacts cross the boundary, the arrangement also satisfies the data sovereignty constraints common in telecommunications regulation, while offering a path toward the AI-RAN infrastructure envisioned in [6].

V. CONCLUSION

This paper presented a traffic diversity-driven resource pooling framework for O-Cloud optimization operating at two levels: TD clustering groups virtualized gNBs with complementary traffic peaks, and LB-FTRL distributes the resulting pooled capacity across co-located application domains. On real-world Korean MNO traffic, TD clustering reduces PAR by 32.6% and, more consequentially, lowers the SLA-constrained capacity that must be provisioned by 17.4% while making that requirement substantially more predictable. LB-FTRL then suppresses RAN overload by 81.1% and raises AI training utilization to 48.9%, so the freed capacity is reclaimed rather than left idle. An SMG-guided reconfiguration policy avoids the churn that degrades naive threshold schemes. The complete system is realized as a deployable O-RAN rApp with a 32-second closed-loop policy update cycle, and is portable to hybrid on-premise/cloud O-Cloud deployments.

Future work will extend LB-FTRL to incorporate traffic forecasts proactively, transforming the current reactive learner into a predictive resource manager, and will investigate explicit co-scheduling of RAN and AI workloads on shared O-Cloud infrastructure, aligned with the AI-RAN convergence trajectory.

ACKNOWLEDGMENT

This work was supported by the IITP (Institute of Information & Communications Technology Planning & Evaluation) - ITRC (Information Technology Research Center) (IITP-2026-RS-2021-II212046, 50%) grant funded by the Korea government (Ministry of Science and ICT) and in part by IITP under YKCS Open RAN Global Collaboration Center (IITP-2026-RS-2024-00434743, 50%) grant funded by the Korea government (MSIT).

REFERENCES

[1] O-RAN Alliance, "O-RAN Architecture Description," O-RAN.WG1.OAD-R003-v10.00, Technical Specification, 2024, o-RAN Alliance Working Group 1.
[2] Y. Zhang, F. Barusso, D. Collins, M. Ruffini, and L. A. DaSilva, "Dynamic Allocation of Processing Resources in Cloud-RAN for a Virtualised 5G Mobile Network," in *Proc. 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 787–791.

[3] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz, "vrAIn: A Deep Learning Approach Tailoring Computing and Radio Resources in Virtualized RANs," in *Proc. 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2019, pp. 1–16.
[4] L. Lo Schiavo, G. Garcia-Aviles, A. Garcia-Saavedra, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "CloudRIC: Open Radio Access Network (O-RAN) Virtualization with Shared Heterogeneous Computing," in *Proc. 30th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2024, pp. 1–15.
[5] E. Ramagiddaiah, G. V. Preethi, S. V. R. Anand, C. R. Murthy, and R. Husbands, "Traffic-Aware Energy Efficiency Optimization in Practical Open RAN 5G Networks," in *Proc. 2nd ACM Workshop on Open and AI RAN (OpenRan)*, 2025, pp. 56–58.
[6] SK Telecom and NTT DOCOMO, "Key Enabling Features for the Evolution of vRAN and the Path to AI-RAN," SK Telecom and NTT DOCOMO, White Paper, Mar. 2026.
[7] K. Cho, J. Kim, and S. Jung, "Enhancing Cost-Effective 5G Virtualized RAN Pooling Gain on Clouds: An Intelligent Auto-Scaling Approach," *IEEE Access*, vol. 12, pp. 111 322–111 336, 2024.
[8] M. Kalntis, G. Iosifidis, and F. A. Kuipers, "Adaptive Resource Allocation for Virtualized Base Stations in O-RAN With Online Learning," *IEEE Transactions on Communications*, vol. 73, no. 3, pp. 1787–1802, Mar. 2025.
[9] H. B. McMahan, "Follow-the-Regularized-Leader and Mirror Descent: Equivalence Theorems and L1 Regularization," in *Proc. 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, ser. JMLR W&CP, vol. 15, 2011, pp. 525–533.
[10] J.-B. Monteil, G. Iosifidis, and I. Dusparic, "Reservation of Virtualized Resources with Optimistic Online Learning," *arXiv preprint arXiv:2303.08772*, 2023.
[11] F. Aslan, G. Iosifidis, J. A. Ayala-Romero, A. Garcia-Saavedra, and X. Costa-Perez, "Fair Resource Allocation in Virtualized O-RAN Platforms," *Proc. ACM on Measurement and Analysis of Computing Systems (POMACS)*, vol. 8, no. 1, pp. 1–34, Feb. 2024.
[12] S. D. A. Shah, M. Hafeez, A. Salama, and S. A. R. Zaidi, "Proactive AI-and-RAN Workload Orchestration in O-RAN Architectures for 6G Networks," *arXiv preprint arXiv:2507.09124*, 2025.