Towards Adaptive PRB Optimization in Open RAN via Near-Real-Time RIC Control

Suneet Kumar Singh[1], Dalibor Zeman[2], Marija Gajić[1], Stanislav Lange[1], Thomas Zinner[1]
[1]Norwegian University of Science and Technology, Norway
[2]VSB - Technical University of Ostrava, Czech Republic

***Abstract*—Beyond 5G (B5G) and 6G networks must support diverse applications with widely varying Quality of Service (QoS) requirements, ranging from ultra-low latency to extremely high throughput. Effectively meeting these demands calls for efficient resource management across applications. However, static resource allocation often struggles to adapt to rapid traffic fluctuations and volatile radio channel conditions, resulting in inefficiencies and performance degradation. This requires adaptive, application-aware control that can respond in real time at fine granularity. In this work, we propose a runtime resource control mechanism that enables application-aware resource allocation based on specific QoS requirements. We focus on dynamic Physical Resource Block (PRB) allocation and evaluate its impact on application-level throughput and latency through experimental validation on a B5G testbed, built using open-source RAN (srsRAN) and core network (Open5GS) implementation. Our results show that dynamic PRB allocation has a significant impact on overall performance. We also evaluate the control loop latency to understand the responsiveness and effectiveness of real-time adaptive resource allocation. This study contributes to the open-source community by advancing autonomous resource management, with the implementation released as open-source to support reproducibility.**

***Index Terms*—O-RAN, resource control, 5G/B5G, ML**

I. Introduction

With the advent of B5G and emerging 6G networks, delivering consistent QoS has become increasingly challenging due to the diversity and stringency of application demands. Latency-sensitive services such as immersive Extended Reality (XR) and Virtual Reality (VR) [1], [2] require not only high throughput but also ultra-low latency and reliable connectivity, which traditional best-effort scheduling cannot guarantee. Existing QoS mechanisms typically depend on static policies configured at the core network, where the 5G QoS Identifier (5QI) is assigned which defines QoS parameters such as packet delay budget and error rate. These 5QI values guide the Radio Access Network (RAN) in scheduling PRBs for each flow. In theory, this core-driven QoS model enables consistent traffic handling across network domains. However, this approach often fails to account for dynamic RAN conditions such as radio link quality, user mobility, and PRB utilization or congestion [3]. This disconnect can result in inefficient resource utilization or suboptimal service quality for time-critical flows. Real-time visibility at the RAN edge enables faster and more granular resource allocation, which directly affects QoS.

To overcome the limitations of static, core-centric QoS enforcement and enable intelligent, context-aware resource decisions at the network edge, the O-RAN Alliance [4] introduced the RAN Intelligent Controller (RIC) architecture. Building on the need for real-time visibility and application-aware control, the RIC comprises the Non-Real-Time RIC (Non-RT RIC) for policy and analytics and the Near-Real-Time RIC (Near-RT RIC) for latency-sensitive decision-making. The RIC [5] collects and monitors Key Performance Measurement (KPM) attributes and performs RAN control (RC) via the E2 interface using E2 Service Models (E2SM). These models, listed in Table I, provide rich telemetry and control capabilities to support learning-based adaptation. By analyzing these metrics, extensible applications (xApps) running on the Near-RT RIC can perform timely, QoS-aware actions, such as dynamic PRB allocation and Data Radio Bearer (DRB) mapping based on detected traffic characteristics. This framework provides the foundation for responsive and fine-grained resource control in modern RAN deployments.

Recent research has explored various techniques for improving resource management in the RAN. For example, [6] proposes a two-step algorithm to optimize PRB allocation in downlink slicing, though their approach assumes fixed service categories and predefined QoS requirements. Similarly, [7] introduce a PRB allocation method for adaptive RAN that reduces quality degradation for low-latency and high-capacity services while minimizing the information collected for control. Their approach, based on long-cycle estimation and short-cycle feedback, is validated via simulations. More recently, AI-driven approaches have been investigated. For instance, [8] propose an AI-driven framework in O-RAN that dynamically allocates PRBs across slices while detecting and mitigating malicious traffic. Their real-world testbed results show that intelligent PRB control preserves QoS for legitimate users even under network attacks. These works highlight the potential of AI-based control to improve QoS. However, prior work lacks a detailed exploration of dynamic PRB adaptation, particularly its impact on overall performance, and of control latency in fully open-source, real-time environments to enable context-aware resource allocation.

In this work, we develop an xApp for real-time, dynamic PRB allocation within the O-RAN architecture. By integrating resource control into the Near-RT RIC, our solution enables on-the-fly adaptation to live traffic conditions. The implementation is deployed and validated on a physical testbed comprising srsRAN, Open5GS, Software Defined Radio (SDR) hardware, and commercial User Equipments (UEs) running real-world application traffic. Our evaluation demonstrates how

978-3-903176-75-1 ©2025 IFIP

TABLE I
KPM AND RAN CONTROL (RC) ATTRIBUTES. RED CROSSES INDICATE UNSUPPORTED ATTRIBUTES.

Service Model	Supported Attributes	srsRAN	OAI
KPM	UE Throughput (UL/DL)	✓	✓
	RLC Delay (UL/DL)	✓	✓
	PRB Usage (UL/DL)	✓	✓
	RLC Packet Drop Rate (DL)	✓	✗
	PDCP SDU Packet Success Rate (UL)	✓	✗
	RLC SDU Transmitted Volume (UL)	✓	✗
	Delay on Over-the-Air Interface (UL)	✓	✗
	PDCP SDU Volume (UL/DL)	✗	✓
RC	UE RRC State Change	✗	✓
	QoS Flow Mapping Configuration	✗	✓
	PRB Allocation	✓	✗

varying PRB allocations impact application-level throughput and latency. We also quantify the control signaling latency involved in propagating resource changes, providing insights into ground-level performance responsiveness. Furthermore, we discuss how such runtime adaptability can be leveraged in real-time applications. As part of our forward-looking vision, we propose an architecture for cross-domain QoS orchestration, spanning both RAN and core network layers.

In summary, the main contributions of this article are:

- We provide a detailed analysis of the performance limitations introduced by default QoS settings in B5G networks.
- We design and implement an xApp capable of real-time monitoring and control, and evaluate the impact of dynamic PRB allocation on performance.
- We analyze the latency between triggering PRB reconfiguration and its effect on application performance to support future adaptation strategies for diverse use case requirements.
- We validate our approach through a complete testbed implementation using open-source platforms such as srsRAN and Open5GS, along with real-time hardware components including SDRs and commercial UEs.
- We release all code, documentation, and setup instructions in an open-source repository[1] to support reproducibility and enable further community development.

The remainder of this paper is organized as follows: Section II analyzes the limitations of default QoS policies based on experiments conducted on a physical testbed. Section III outlines the system architecture and the design of the proposed xApp workflow. Section IV presents the experimental evaluation of our solution. Section V explores how the insights gained can be applied in real-time scenarios and outlines directions for future work. Finally, Section VI concludes the paper.

II. LIMITATIONS OF DEFAULT QOS POLICIES

Using default QoS settings or failing to apply dynamic optimization often leads to inefficient resource usage and degraded performance. To understand this issue, we conducted an in-depth analysis using our B5G testbed (Fig. 2). This

[1]https://git.ntnu.no/suneetks/RIC-B5G

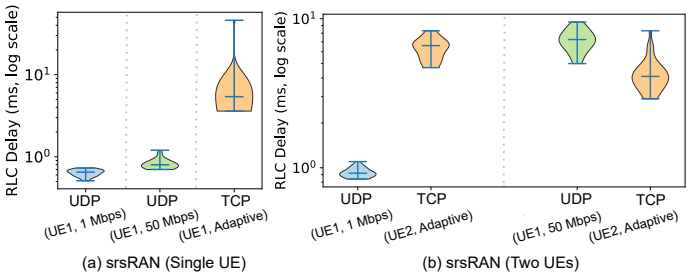

Fig. 1. RLC Delay Analysis at Varying Throughput Levels for Single vs. Dual UE in srsRAN Downlink.

setup includes commercial and open-source components such as Quectel USB modems (RM500/RM520), Nokia XR20 UEs, srsRAN [9] and OpenAirInterface (OAI) [10] gNBs with Ettus USRP X310 SDRs (40 MHz channel bandwidth), and 5G core implementations based on Open5GS [11]. Integration with O-RAN Software Community (SC) RIC [12] allows telemetry data to be captured and stored in time-series databases at regular intervals. Using this configuration, the peak throughput reached approximately 115 Mbps.

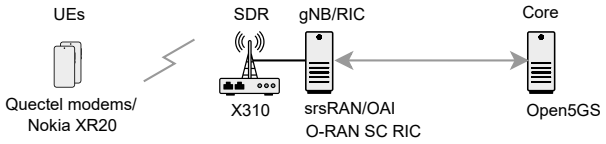

Fig. 2. B5G Testbed.

For this analysis, we focus on Radio Link Control (RLC) delay, which can be impacted by inefficient PRB allocation and, in turn, contributes to increased end-to-end latency. We begin our evaluation with single-UE experiments, where iPerf is used to generate UDP and TCP traffic flows, while monitoring the KPM attributes summarized in Table I. UDP flows at 1 Mbps and 50 Mbps consistently show low RLC delay (Fig. 1a), attributed to minimal congestion and adequate PRB allocation. However, TCP flows, particularly when using the CUBIC congestion control algorithm, result in noticeably higher RLC delay, often around 10 ms. This behavior arises from CUBIC's aggressive bandwidth probing, which leads to increased queuing. These results indicate that even under basic configurations, default QoS and scheduler settings can introduce significant latency variability. We then extend this analysis to dual-UE scenarios (Fig. 1b), where one UE runs a UDP flow and the other a TCP flow. At low UDP rates (e.g., 1 Mbps), the delay remains low and is comparable to the single-UE case. However, as the UDP rate increases (up to 50 Mbps), the RLC delay rises to around 10 ms due to insufficient PRB allocation, as the limited PRBs are equally distributed between both UEs under default QoS settings.

These findings reveal that without dynamic and intelligent resource control, even non-saturating traffic levels can lead to substantial QoS degradation, particularly for latency-sensitive applications such as AR/VR and VoIP. PRB resources are

inherently limited, and as demonstrated in Fig. 1b, default QoS settings fail to adapt to changing traffic conditions, resulting in noticeable performance drops when resources are shared across multiple UEs. To mitigate this, continuous network monitoring and traffic-aware PRB allocation become essential. Driven by these challenges, the following section presents our system architecture featuring the PRB Control Loop.

III. System Architecture

The overall system architecture is illustrated in Fig. 3. It comprises srsRAN as the access network, Open5GS as the 5G core, and the O-RAN SC RIC implementing the Near-RT RIC within the RIC framework. A detailed description of each component is provided below.

A. B5G Network Architecture

The B5G network setup consists of srsRAN (version 25.05) and Open5GS (version 2.7.1). The srsRAN Distributed Unit (DU) handles lower-layer radio functions, including MAC-layer scheduling, where PRB allocation is dynamically managed based on traffic load and radio conditions. It also supports RLC buffering and DRB configuration to ensure reliable and efficient data delivery. The Central Unit–User Plane (CU-UP) is responsible for the Service Data Adaptation Protocol (SDAP), Packet Data Convergence Protocol (PDCP), and GPRS Tunneling Protocol (GTP)-U layers, managing bearer mapping, security functions, and forwarding of user data. The CU-UP interfaces with the User Plane Function (UPF) via GTP-U tunnels, where QoS Flows identified by QFIs are mapped to DRBs associated with specific 5QIs. The Central Unit–Control Plane (CU-CP) oversees Radio Resource Control (RRC) and Next Generation Application Protocol (NGAP) procedures, handling signaling, UE context management, and mobility. To support high-throughput wireless communication in this setup, we use an SDR X310 configured with a 40 MHz channel bandwidth.

B. RIC Framework

The RIC framework consists of three main components: the Near-RT RIC, the xApp, and the ML processing thread. In our setup, the Near-RT RIC is implemented using the O-RAN SC RIC, which is deployed as a multi-container Docker-based application. It offers a modular and standardized platform for managing xApps and interacting with the RAN in real time.

The xApp is implemented in Python and is designed to monitor and manage RAN behavior through data exposed by the KPM and RC Service Models (SMs) within the E2 agent. It periodically collects KPM attributes, store in the time series database which is read by the integrated ML thread for real-time analysis. Based on the ML-driven decisions, the xApp issues control messages to the Near-RT RIC, enabling dynamic actions such as adaptive resource allocation within the RAN.

In the ML thread, the trained classification model is loaded and used to infer application types in real-time. This thread periodically reads metrics from the database, extracts the relevant features, and performs predictions at one-second

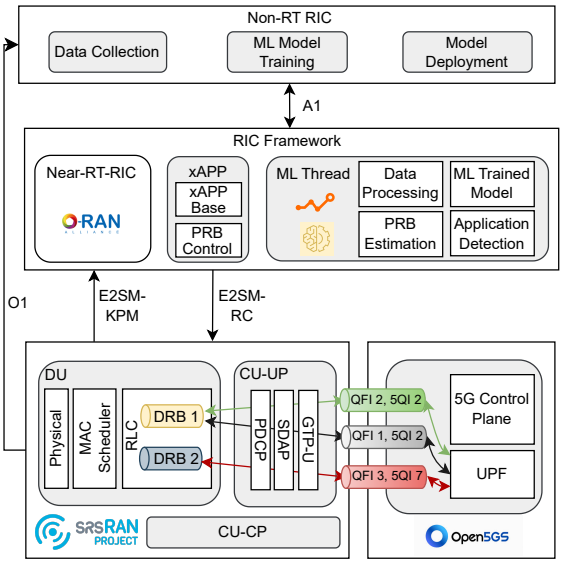

Fig. 3. System Architecture with PRB Control Loop.

intervals—a frequency currently supported by the open-source RIC framework. However, this interval can be adjusted dynamically depending on the QoS sensitivity of specific use cases. Upon classifying the traffic, control actions such as PRB reallocation can be triggered immediately after one or two consistent predictions. For instance, bandwidth-intensive applications like cloud gaming may require prompt and higher PRB provisioning compared to less critical services.

C. Non-RT RIC

The Non-RT RIC continuously collects telemetry data from the RAN via the O1 interface, including traffic patterns, QoS metrics, and control-plane statistics. This data is used to dynamically train and update ML models that learn traffic behavior over time. Once trained, the updated models are deployed to the Near-RT RIC via the A1 interface, enabling real-time inference and control decisions that directly impact the data plane (e.g., PRB allocation, QoS enforcement).

The O-RAN SC [13] provides a reference implementation of the Non-RT RIC, which can be used for data collection, model training, and policy management, offering a solid foundation for such workflows. However, a key challenge lies in the efficiency of runtime model training and deployment, specifically, how quickly models can be trained, validated, and pushed to the Near-RT RIC for enforcement. This becomes especially critical for latency-sensitive use cases, where delays in model updates may impact the system's responsiveness.

In this work, we deploy a B5G network in our laboratory using real-time equipment and the open-source platforms discussed above. The Near-RT RIC is integrated with the B5G setup, specifically connected to the gNB for monitoring and resource allocation. Our primary focus is on a detailed analysis of PRB allocation and its impact on performance, including latency measurements associated with triggering control signals for resource adjustments. While our implementation

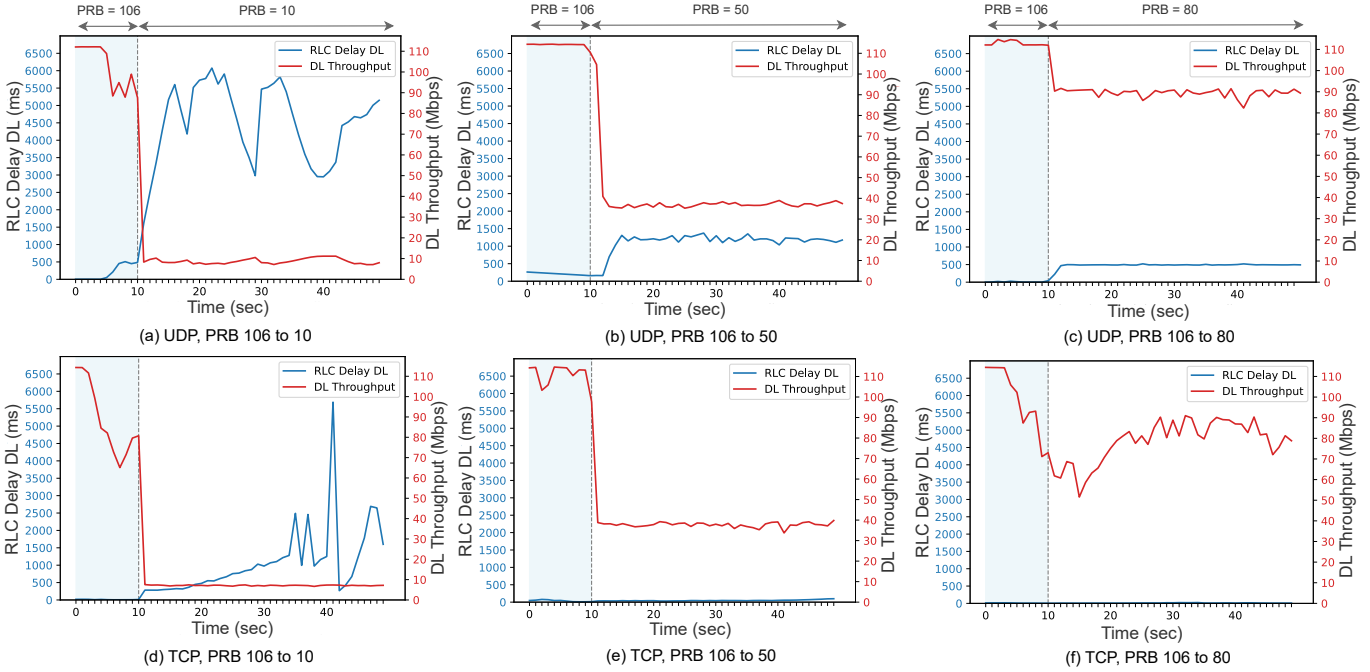

Fig. 4. Runtime PRB Adaptation in a B5G Network with Analysis of Downlink RLC Delay and UE Throughput.

supports the integration of an ML thread for real-time traffic classification—available in our source code repository—the aspects related to classification and the use of the Non-RT RIC are left for future work.

IV. Experimental Evaluation

In this section, we address the following key research questions:

1) How does dynamic PRB allocation impact application-level throughput and latency in a real-time B5G network deployment?
2) How do different traffic types influence the performance outcomes of dynamic PRB allocation?
3) What is the response time between triggering a PRB allocation change and observing its effect on application performance?

The detailed analysis and findings related to these questions are presented in the following subsections.

A. Experiment Setup

1) Testbed: Our B5G lab environment consists of two primary servers, both built on the x86 architecture and powered by Intel Core i7-6700 processors, each featuring 4 physical cores and 8 threads with hyper-threading. The CPUs operate at a base frequency of 3.40 GHz. One server hosts the srsRAN and RIC framework and is connected to an Ettus X310 SDR and a Quectel USB modem. The second server is dedicated to the Open5GS core network deployment.

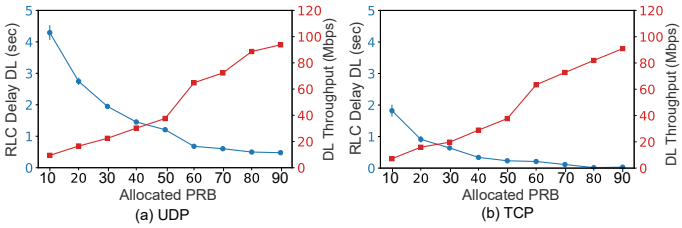

Fig. 5. Aggregated RLC Delay and UE Throughput vs. PRB Allocation (with 95% Confidence Intervals).

2) Evaluation Metrics: To evaluate the impact of varying PRB allocations, we use throughput, latency, and RLC layer delay as key performance indicators. Downlink (DL) throughput is measured in Mbps, while RLC layer delay is analyzed in both milliseconds (ms) and seconds (sec). The response time from the RIC to the RAN, representing the control signaling latency, is also measured in ms. In our setup, the total number of PRBs is 106. This corresponds to a 40 MHz channel bandwidth, of which 38.16 MHz is effectively used for transmission, with the remainder reserved as guard bands. The system operates with a 30 kHz subcarrier spacing (SCS). Since each PRB consists of 12 subcarriers, the total number of PRBs is calculated as $\frac{38.16\ \text{MHz}}{12 \times 30\ \text{kHz}} = 106$.

B. Performance Evaluation

The performance analysis is divided into two parts: the first examines application-level performance under varying PRB allocations at runtime, and the second focuses on analyzing the response time from the control request to observable

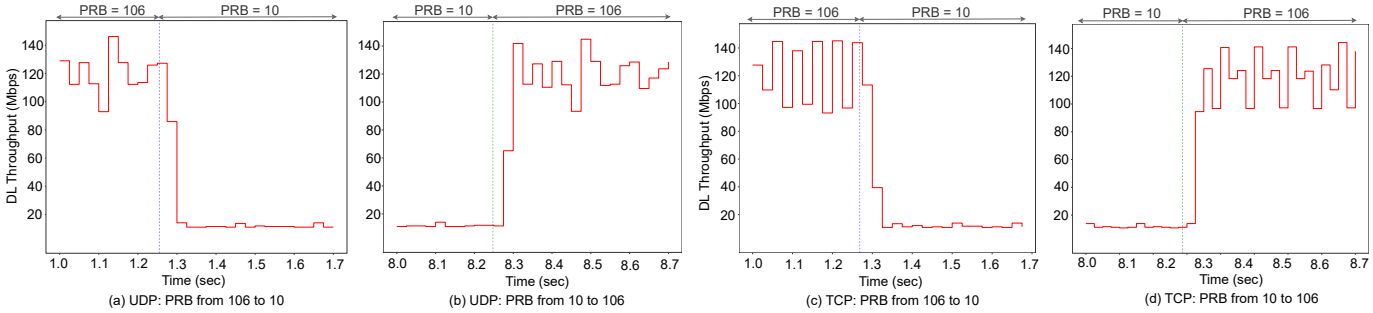

Fig. 6. Response time from PRB drop trigger to restoration, reflecting UE throughput changes. Throughput is aggregated in 25 ms bins.

performance changes in the RAN. The detailed analysis is presented below.

1) Performance Under Dynamic PRB Allocation: The PRB is the smallest unit of resource allocation in the frequency-time grid of the RAN. As mentioned in the previous section, depending on the system bandwidth and configuration, a typical RAN deployment may support up to 106 PRBs per time slot. The number of PRBs required to meet a specific throughput demand depends on several parameters including the modulation and coding scheme (MCS), signal quality (e.g., SINR), and the overhead introduced by protocol layers.

Once the traffic flows are identified and their QoS requirements (e.g., latency sensitivity or guaranteed bit rate) are known, it becomes possible to dynamically allocate an appropriate number of PRBs per UE. This approach enables more efficient resource utilization and better QoS satisfaction in real-time scenarios. To better understand the performance behavior under different PRB allocations, we conducted controlled iPerf experiments with TCP and UDP flows to generate higher-throughput traffic. This allowed us to perform a detailed evaluation of how PRB allocation affects KPM attributes such as RLC delay and UE throughput under different traffic conditions.

Results: Fig. 4 presents the experimental results for both UDP and TCP flows, evaluating DL RLC delay (in ms) and throughput (in Mbps) under different PRB allocations. We conducted tests using PRB values in multiples of 10, with the maximum achievable throughput around 115 Mbps. Initially, for the first 10 seconds, the system operated with the default PRB allocation (106 PRBs). After this period, the xApp began dynamically updating PRB allocations by sending control messages to the RAN. For UDP flows (Fig. 4a–c), a sharp drop in throughput and a corresponding spike in RLC delay was observed immediately after PRB reduction. A similar trend was noted for TCP flows (Fig. 4d–f); however, TCP demonstrated lower RLC delay, particularly with PRB 50 and 80. This suggests that TCP's congestion control helps stabilize delay under constrained conditions, whereas UDP continuously sends packets at a rate of 115 Mbps without control, leading to RLC queue buildup and increased delay.

Fig. 5 provides an aggregated view of DL RLC delay and UE throughput across a range of PRB values from 10 to 90. This helps to understand the overall relationship between PRB allocation and performance metrics. The results reveal a generally linear trend. However, in the lower PRB range (e.g., 10–30), performance degrades more sharply—RLC delay rises rapidly and throughput drops significantly.

2) Control-to-Performance Response Time Analysis: This section focuses on measuring the time it takes for control signals (i.e., PRB reconfiguration requests) from the RIC to produce observable effects on application performance in the RAN. This metric is critical for evaluating how responsive the system is to dynamic network conditions and traffic demands. A shorter response time enables faster adaptation, making it suitable for latency-sensitive or mission-critical applications. This analysis also helps in identifying bottlenecks in the control loop between RIC and RAN.

In the test scenario, we begin by generating TCP or UDP traffic using iPerf with a full allocation of 106 PRBs for the initial period. Then, a control request is dynamically sent to reduce the allocation to 10 PRBs. After a few seconds, the xApp sends another signal to restore the allocation back to 106 PRBs. This process is repeated for both TCP and UDP traffic flows. We record the timestamp immediately before triggering each control signal to accurately measure the response time.

Results: Fig. 6 provides a detailed examination of the effects of dynamic PRB allocation on UE throughput, shown across four subfigures. The x-axis is divided into 100-ms intervals, with throughput aggregated in 25-ms bins. Fig. 6a illustrates the case for UDP traffic when PRB allocation is reduced from 106 to 10. Following the control signal trigger, throughput sharply decreases to about 10 Mbps within approximately 50 ms. Conversely, Fig. 6b shows the recovery process as PRBs increase from 10 back to 106, with throughput returning to near-original levels within approximately 50 ms. For TCP traffic, Fig. 6c reveals a similar throughput reduction pattern after PRB reduction, while Fig. 6d demonstrates that throughput restoration to baseline levels occurs within roughly 30 ms after PRB increase. Taken together, these observations indicate that the total response time—from initiating the control message to fully reflected changes in throughput—lies between 30 and 50 ms. This response latency underscores the system's capability to dynamically adjust resource allocation promptly, a crucial factor for sustaining QoS in real-time scenarios.

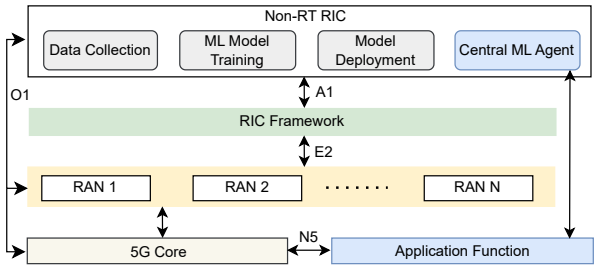

Fig. 7. Cross-Domain QoS Optimization Architecture.

V. Discussion and Future Work

A key insight from the performance evaluation (Fig. 6) is the response time of approximately 30–50 ms. While this level of responsiveness is generally sufficient for applications such as adaptive video streaming, interactive media, and many industrial automation tasks, it still falls short of the ultra-low latency requirements (e.g., below 10 ms) demanded by safety-critical use cases like autonomous vehicles or remote control systems. Also, the current evaluation was conducted with a limited number of UEs under controlled traffic scenarios. In practical large-scale deployments, scalability becomes a significant challenge. While the E2SM-RC sends control requests in batches rather than individually per UE, the total signaling load will still grow with the number of UEs and cells involved. This increase can introduce additional delays in the control loop due to signaling congestion, processing overhead, or scheduling conflicts at the RIC and RAN layers.

In addition to response time, we observe that insufficient PRB allocation to application flows (Fig. 4 and 5) can significantly degrade both delay and throughput performance, highlighting the critical importance of timely and adaptive resource optimization. Rigid slicing approaches often lead to PRB imbalances across applications, especially under dynamic traffic conditions. In contrast, runtime PRB reallocation driven by RIC feedback improves both responsiveness and allocation precision by aligning resources with real-time service demands. The xApp can synchronize with 5QI assignments and monitoring data to compute and update PRB distribution dynamically. Moreover, ML and Reinforcement Learning (RL) can learn optimal PRB allocation strategies by adapting to dynamic network states, traffic patterns, and 5QI configurations from the core network.

Future work should develop scalable control architectures and optimization algorithms for low-latency resource allocation across multiple UEs. ML-driven predictive and adaptive management in Near-RT and Non-RT RICs can enhance real-time decisions and extend toward Quality of Experience (QoE) to align optimization with user experience. As illustrated in Fig. 7, RIC-based optimization can extend beyond the RAN to the core network, where a centralized ML agent aggregates RAN insights and coordinates with the Application Function (AF) to dynamically update QoS policies, enabling cross-domain optimization for improved end-to-end performance.

VI. Conclusions Remarks

This work presented a practical evaluation of dynamic resource allocation in a real-time B5G RAN environment using a standards-compliant O-RAN architecture. Through a series of controlled experiments, we analyzed the impact of varying PRB allocations on application-level performance metrics such as throughput and RLC layer delay for both TCP and UDP traffic. Additionally, we quantified the control-to-performance response time, showing that observable changes in throughput can occur within 30–50 ms of a control signal being triggered. Our findings highlight the importance of timely and adaptive resource management to meet the diverse QoS requirements of modern applications. Dynamic PRB reallocation can serve as an effective mechanism for prioritizing critical traffic, but it also requires careful consideration of signaling overhead and responsiveness, especially in multi-UE scenarios. This study's insights can guide intelligent, scalable RAN control strategies. Future work will integrate ML to enhance network efficiency.

Acknowledgment

This work received partial funding from the Research Council of Norway via the SFI Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS), project no. 310105, and was co-funded by the European Union through the REFRESH project – Research Excellence for Region Sustainability and High-tech Industries, ID No. CZ.10.03.01/00/22_003/0000048.

References

[1] T. Taleb, Z. Nadir, H. Flinck, and J. Song, "Extremely Interactive and Low-Latency Services in 5G and Beyond Mobile Systems," *IEEE Communications Standards Magazine*, vol. 5, no. 2, pp. 114–119, 2021.

[2] F. G. Vogt, F. E. R. Cesen, A. G. de Castro, S. K. Singh, M. C. Luizelli, C. E. Rothenberg, and G. Antichi, "Video Streaming QoE Meets Programmable Data Planes: The Case of In-Network QoE for 360°VR," *IEEE Network*, vol. 39, no. 2, pp. 176–183, 2025.

[3] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on ran: Flexibility and resources abstraction," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102–108, 2017.

[4] O.-R. ALLIANCE, "O-RAN E2 Service Model (E2SM) 7.0," https://specifications.o-ran.org/specifications, 2025, accessed: 2025-04-26.

[5] B. Balasubramanian, E. S. Daniels, M. Hiltunen, R. Jana, K. Joshi, R. Sivaraj, T. X. Tran, and C. Wang, "RIC: A RAN Intelligent Controller Platform for AI-Enabled Cellular Networks," *IEEE Internet Computing*, vol. 25, no. 2, pp. 7–17, 2021.

[6] M. Karbalaee Motalleb, V. Shah-Mansouri, S. Parsaeefard, and O. L. Alcaraz López, "Resource allocation in an open ran system using network slicing," *IEEE Transactions on Network and Service Management*, vol. 20, no. 1, pp. 471–485, 2023.

[7] H. Hirayama, Y. Tsukamoto, and H. Shinbo, "Feedback control for qos-aware radio resource allocation in adaptive ran," *IEEE Access*, vol. 10, pp. 21 563–21 573, 2022.

[8] A. Alchaab, A. Younis, and D. Pompili, "Demo: Secure edge server for network slicing and resource allocation in open ran," in *2025 IEEE 26th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2025, pp. 169–171.

[9] "srsRAN: Open Source 5G CU/DU," 2025, accessed: 2025-04-25. [Online]. Available: https://www.srsran.com/5g

[10] O. S. Alliance, "OpenAirInterface," https://openairinterface.org/, 2024.

[11] "Open5GS: Open Source 5G Core and EPC Implementation," 2025, accessed: 2025-04-25. [Online]. Available: https://open5gs.org/

[12] srsran, "ORAN SC RIC," https://github.com/srsran/oran-sc-ric/, 2025.

[13] O-RAN Software Community, "O-RAN SC Non-RT RIC Project Overview," https://docs.o-ran-sc.org/projects/o-ran-sc-nonrtric/en/cherry/overview.html, 2025, accessed: Aug. 6, 2025.