Computer Communications 236 (2025) 108108

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Active queue management in 5G and beyond cellular networks using Machine Learning

Alexandros Stoltidis, Kostas Choumas *, Thanasis Korakis

Department of ECE, University of Thessaly, Volos, Greece

ARTICLE INFO

Keywords:
5G
QoS
AQM
AI
ML
LSTM
Disaggregated RAN
RIC
OpenAirInterface5G

ABSTRACT

This paper proposes a state-of-the-art framework for adapting Active Queue Management (AQM) in 5G and beyond cellular networks with disaggregated Radio Access Network (RAN) deployments. While existing AQM algorithms effectively mitigate bufferbloat in monolithic RAN deployments, their potential in disaggregated ones remains largely unexplored. This gap particularly relates to AQM algorithms relying on communication between layers distributed across distinct network entities to operate. Our research explores the current literature on AQM, identifies the gaps regarding disaggregated deployments, and introduces a comprehensive framework that employs Artificial Intelligence (AI) and Machine Learning (ML) within the RAN Intelligent Controller (RIC) for adapting AQM in such deployments. We evaluate our novel solution on a previously proposed AQM algorithm which requires cross-layer communication, using OpenAirInterface5G (OAI5G) to deploy a disaggregated RAN and a connected User Equipment (UE) that experiences realistic network conditions, including noise and mobility. Finally, we assess its accuracy through the Quality of Service (QoS) achieved for our disaggregated deployment on the NITOS testbed.

1. Introduction

The fifth generation (5G) of cellular networks is an evolution over its predecessors, prioritizing enhanced performance across diverse network conditions and requirements. To achieve this, 5G has introduced the concept of network slicing [1], which divides the physical network into several logical networks. Each network slice allocates dedicated resources for the specific requirements of the different services deployed within the network. As defined in the Release 15 specifications [2], 5G categorizes these services into three distinct categories [3], including Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communications (URLLC), and Massive Machine-Type Communications (mMTC). The eMBB service prioritizes throughput to support high-data-rate applications, while URLLC targets applications requiring highly reliable and low-latency communication, prioritizing latency. Lastly, mMTC facilitates communication across multiple devices, ensuring minimal energy consumption and efficient low-data-rate transmission.

Modern cellular networks rely on buffers to optimize resource utilization. However, when these buffers become congested, bufferbloat [4] occurs, causing significant issues, such as prolonged sojourn times, higher latency, and degraded network performance. Bufferbloat in 5G can arise when network slices support multiple services with inherently conflicting objectives, such as eMBB and URLLC services. While eMBB prioritizes high throughput by heavily filling the network queues to maximize resource utilization, this excessive buffering increases latency, undermining URLLC.

AQM is a network congestion control technique that prevents bufferbloat. It dynamically manages the buffers within the network components, replacing the simplistic passive queue management techniques, such as drop-tail and drop-head. More specifically, AQM maintains buffer occupancy at a reasonable size, preventing excessive buffering and avoiding congestion or delays. The existing literature has proposed various AQM algorithms that manage queues across different protocol stack layers within the 5G architecture. However, based on our research, most of these algorithms are latency-sensitive and were designed for the monolithic deployment of 5G, overlooking its potential for disaggregated deployments. In a disaggregated 5G deployment, the network protocol stack layers, including their functions and corresponding queues, are distributed across separate network entities, which may reside on different machines, introducing latency between them. As a result, previously introduced latency-sensitive AQM algorithms may prove inadequate in disaggregated 5G deployments.

Despite significant progress in 5G disaggregation research, the 5G specification does not address AQM in disaggregated 5G deployments, making its inclusion essential in future generations of cellular networks. In our previous study [5], we proposed a comprehensive solution for

* Corresponding author.
E-mail addresses: stalexandros@uth.gr (A. Stoltidis), kohoumas@uth.gr (K. Choumas), korakis@uth.gr (T. Korakis).

https://doi.org/10.1016/j.comcom.2025.108108
Received 24 May 2024; Received in revised form 25 January 2025; Accepted 19 February 2025
Available online 1 March 2025
0140-3664/© 2025 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

managing AQM algorithms in disaggregated 5G and beyond networks. However, a significant limitation of that work is that it serves primarily as a proof-of-concept, evaluated exclusively under ideal network conditions, undermining its effectiveness and reliability in unpredictable and suboptimal real-world scenarios. In this research, we address this limitation and provide a more comprehensive approach by:

- Incorporating realistic conditions into our experiments, including mobility and variable network quality.
- Expanding our prior implementation and enhancing the AI/ML integration within each RIC to address complex traffic patterns and network conditions.
- Demonstrating and evaluating the effectiveness of our approach and the performance improvements achieved under realistic network environments.

This paper comprises six primary sections. Section 1 provides a comprehensive overview of the bufferbloat phenomenon and AQM, along with an overview of our contribution. Section 2 delves into the literature on AQM in 5G networks. Sections 3 and 4 outline the proposed solution and introduce the implementation framework used for experimentation. Section 5 presents the evaluation framework and assesses the proposed solution. Finally, Section 6 concludes this research paper and discusses future directions for AQM beyond 5G networks.

2. Related work

All AQM algorithms aim to control congestion by actively managing the sizes of the queues within the network entities. They achieve this by intelligently discarding packets or employing strategies to handle the different types of traffic in distinct ways, preventing bufferbloat and ensuring the efficient utilization of the transmission channel. Studies have demonstrated that keeping queue sizes small can significantly decrease sojourn times, resulting in lower latency [6]. As this research primarily focuses on 5G and beyond cellular networks, we investigate previously proposed AQM algorithms that have been previously implemented and evaluated in cellular networks.

The literature on AQM in cellular networks is extensive, featuring numerous algorithms that operate on different principles. Some AQM algorithms function independently within individual queues. These algorithms operate based solely on the queue they manage, without any further knowledge of the entire network, effectively eliminating the overhead and the complexity associated with coordinated decision-making. Such examples include Random Early Detection (RED) [7], Controlled Delay (CoDel) [8], Proportional Integral Controller Enhanced (PIE) [9], Fair Queuing CoDel (FQ-CoDel) [10], RED-SP-CoDel [11], and RLRBM [12]. More specifically, RED probabilistically drops incoming packets as the queue length increases, while CoDel drops them based on their sojourn time. The application of RED and CoDel in Long-Term Evolution (LTE) 4G cellular networks has been extensively studied in [13] and in [6], respectively. PIE is an alternative to CoDel and performs similarly to RED as it probabilistically drops packets to maintain average queuing delay at a target value. FQ-CoDel is an extension of CoDel that manages individual flows independently, providing fair bandwidth distribution among concurrent flows. RED-SP-CoDel combines RED, Static Priority scheduling, and CoDel, offering flow prioritization and enhanced QoS. Finally, RLRBM uses Deep Reinforcement Learning (DRL) to find the optimal size of the buffers within the Radio Link Control (RLC) depending on the network conditions within the 5G network.

While these algorithms are adequate, they are not optimal as they do not consider the network as a whole, which has driven the development of more advanced algorithms. Such AQM algorithms require cross-layer communication to make informed and coordinated decisions to manage the queues of the network. These algorithms include Stochastic Fair Queuing (SFQ) [14], 5G Bandwidth Delay Product (5G-BDP) [15], UPF-SDAP Pacer (USP) [15], and Dynamic RLC Queue Limit (DRQL) [15]. SFQ was implemented in LTE and relies on communication between the RLC and PCDP layers to manage their respective buffers, preventing bufferbloat by deciding whether to forward packets from Packet Data Convergence Protocol (PDCP) to RLC. In 5G-BDP, the Service Data Adaptation Protocol (SDAP) layer communicates with the Medium Access Control (MAC) and the RLC layers to forward the optimal amount of data based on the available channel capacity, the remaining data on the RLC buffers and the current transmission state. USP requires communication between the SDAP and RLC layers and the User Plane Function (UPF) at the 5G Core Network (5G-CN). More specifically, the SDAP layer communicates with the RLC layer to find its capacity, referred to as its saturation point. Meanwhile, the UPF communicates with the SDAP to forward its packets, ensuring an optimized flow rate while respecting the saturation point of RLC.

The DRQL algorithm requires communication between the SDAP and RLC layers. Due to this requirement for cross-layer communication and its simplicity, it makes the perfect example to demonstrate the challenges of implementing such algorithms in disaggregated network deployments. In DRQL, the MAC scheduler determines the capacity of the RLC buffers, while the SDAP layer deals with their occupancy. More specifically, the SDAP layer continuously queries the RLC to learn the capacity and occupancy of its buffers and determine whether to forward its packets. On the other hand, the MAC layer pulls as much data from the RLC buffers as the radio channel can support. When the MAC pulls the whole RLC buffer, it increases the capacity of the buffers, as it could have handled more data if not starved. In contrast, if the MAC scheduler manages to process part of the data in the RLC buffers, it adjusts their capacity to align with the available radio channel capacity.

Finally, [16] presents the *Antelope*, designed to switch between AQM algorithms, depending on the conditions of the environment. However, this system has not been implemented in 5G and overlooks the complexities that arise in disaggregated 5G networks. Similarly, the authors in TC-RAN [17] propose a general traffic control system to manage AQM algorithms in monolithic and disaggregated 5G and 6G networks. However, they do not address the potential challenges that arise with the introduction of latency in such disaggregated deployments, nor do they offer solutions for mitigating these issues. Our proposed scheme addresses these challenges, enabling the implementation of AQM algorithms that require cross-layer communication between layers distributed across separate network entities in disaggregated and high-latency 5G deployments.

3. AQM in disaggregated networks

While the literature on AQM in 5G and beyond networks is extensive, it primarily focuses on monolithic 5G-RAN deployments. In 5G cellular networks, the base station formerly known as Node-B (NB) is now referred to as Next-Generation NB (gNB), distinguishing it from the Enhanced NB (eNB) of 4G networks. One key difference between the gNB and the eNB is that the design of the gNB focuses on virtualization and cloud-native capabilities. To achieve this, eNB and gNB feature distinct architectures, with functional splits introduced by 3GPP [18] allowing the disaggregation of gNB into multiple entities, including the Centralized Unit (CU) and the Distributed Unit (DU). The 3GPP 7.2x split [19] is the most widely adopted functional split option. In this configuration, the CU manages the higher layers of the protocol stack, while the DU handles the lower layers of the disaggregated gNB. More specifically, the CU encompasses the SDAP, the Radio Resource Control (RRC), and the PDCP layers. The DU incorporates the RLC, the MAC, and the Physical (PHY) layers. Additionally, the CU comprises the control plane (CU-CP) and the user plane (CU-UP). The CU-CP contains the RRC layer and handles the exchange of control messages, while the CU-UP includes the SDAP layer and manages data flow.

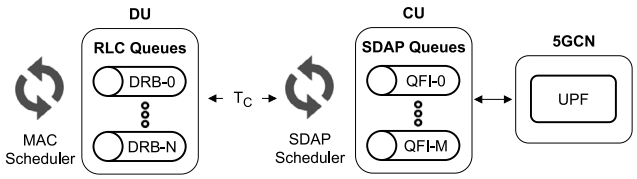

Fig. 1. RLC and SDAP queues in disaggregated gNB.

3.1. AI/ML assisted AQM

Integrating most AQM algorithms into the 5G-RAN requires the utilization of the QoS Flow Identifier (QFI) queues at the SDAP, alongside the Data Radio Bearer (DRB) queues at the RLC, as illustrated in Fig. 1. Each QFI queue corresponds to a different QoS flow, incorporating the downlink packets based on their respective QFI provided by the UPF. The SDAP scheduler maps the QFI queues to their respective DRB queues and forwards the packets, ensuring the appropriate prioritization to maintain the desired QoS.

As outlined previously, some AQM algorithms depend on cross-layer communication, such as DRQL, which relies on the interaction between the SDAP and RLC layers to operate. In a monolithic gNB, the CU and DU entities reside on a single host machine, enabling instantaneous communication between the SDAP and RLC layers. In contrast, a disaggregated gNB separates the CU and DU entities across different network nodes, increasing the communication latency and degrading the performance of latency-sensitive AQM algorithms.

Our proposed solution leverages AI/ML to overcome the AQM challenges caused by the communication latency in disaggregated gNB deployments. More specifically, we decouple and reallocate a portion of the control plane of the 5G-RAN to the RIC entities, adhering to the O-RAN specification [20]. The RIC entities employ AI/ML to forecast the state of DU, which is essential for the CU to support AQM. With this approach, the CU avoids the latency for direct information exchange with the DU, thereby improving performance and enabling support for AQM in disaggregated gNB deployments.

As is evident, the prediction accuracy significantly impacts the validity and performance of our approach. In our previous study, we developed a proof of concept, where the RIC entities monitored solely the state of the DU to predict its behavior for decision-making at the CU. By experimenting with simplified network conditions and traffic patterns, we found that the data from the DU alone was sufficient for accurate predictions, demonstrating significant performance improvements. However, as we more accurately reflect real-world scenarios, the network conditions and traffic patterns become increasingly complex, and the data from the DU alone is no longer sufficient for making accurate predictions.

We build on our previous approach by having the RIC entities monitor the overall 5G-RAN for more comprehensive historical data, including supplementary information from the DU and the CU entities. This additional information enables the RIC entities to comprehend more diverse and complex network conditions. As this study focuses specifically on the performance of our framework adapting DRQL, the RIC analyzes these historical patterns to forecast upcoming fluctuations of the DRB queues based on the current network conditions and mobility patterns. Subsequently, the SDAP leverages these forecasts to make informed decisions on packet forwarding. To thoroughly assess our approach, we comprehensively evaluate it with the DRQL algorithm under realistic network conditions in a disaggregated gNB deployment.

3.2. RIC assisted DRQL

As with most cross-layer AQM algorithms, DRQL requires seamless communication between the protocol stack layers, residing at CU and DU, to operate. More specifically, it necessitates that the SDAP layer at

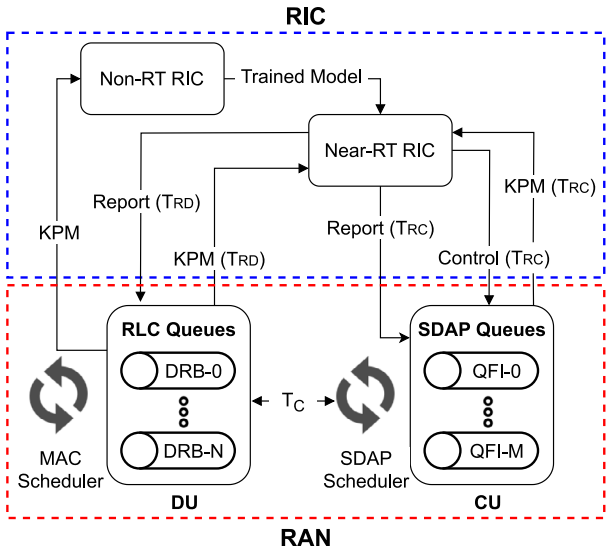

Fig. 2. Communication between RIC entities and the CU and DU.

the CU retrieves precise measurements of the status of the DRB queues within the RLC at the DU, consisting of their actual size and limit. The accuracy and efficiency of DRQL highly depend on the time-critical delivery of these measurements from the DU to the CU. When the CU requests these measurements from the DU, the communication latency, denoted as T_c, causes the provided information to reflect the status of the DRB queues in the past. More specifically, we observe that the information that CU requested at $t_0 - 2T_c$ and received at t_0 reflects the state of the DU at $t_0 - T_c$, leading to suboptimal forwarding decisions at the SDAP scheduler based on outdated information. Furthermore, since the SDAP relies on this information for every packet in its QFI queues to facilitate forwarding decisions, the latency in acquiring it hinders performance, resulting in reduced throughput and increased overall latency.

Our approach leverages RIC to forecast the status measurements of the DRB queues by analyzing historical data, eliminating the need for SDAP to query these measurements from RLC whenever a packet needs to be forwarded. The CU and DU entities periodically provide their current measurements to Non-Real Time RIC (Non-RT RIC) and Near-Real Time RIC (Near-RT RIC). The utilization of the different RIC entities depends on the control loops [21] defined in the O-RAN specification [22]. While the 3GPP specification does not define specific control loops for the CU and the DU entities, the O-RAN architecture introduces these loops to determine which RIC entity to employ, based on the time required to make and apply a decision. More specifically, the near-real-time loop or Loop 2, performed by CU entities and the Near-RT RIC, operates within time frames of 10 ms to 1s, making the Near-RT RIC particularly well-suited for model inference. On the other hand, the non-real-time loop or Loop 3, executed by the Non-RT RIC, has a duration longer than 1 s and is suited for model training.[1]

The Non-RT RIC trains the AI/ML models using the collected measurements and transfers them into the Near-RT RIC. The data collection for the training process is performed in a monolithic network deployment, ensuring the network conditions align with those DRQL was designed for and enabling its proper operational behavior. With this setup, the Non-RT RIC can monitor the behavior of the entire network under DRQL, including the fluctuations in the DRB queues that occur

[1] Additionally, the real-time control loop, or Loop 1, is used by the time-sensitive DU and RU entities as it operates on timescales of 1 ms.

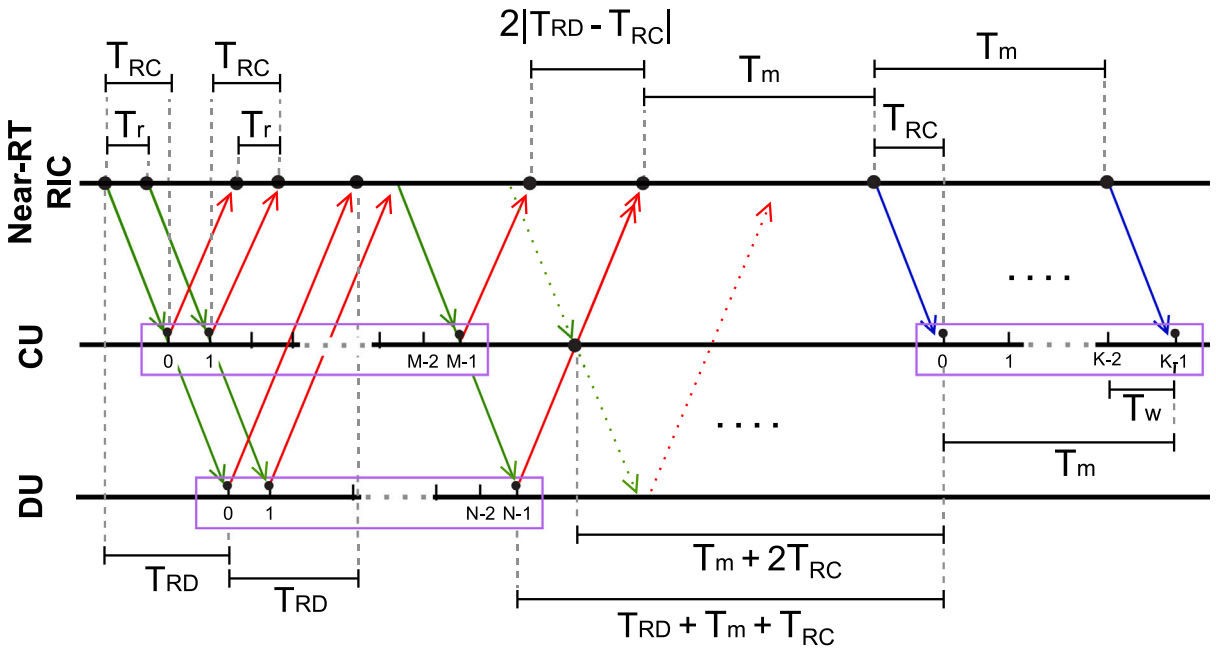

Fig. 3. Timing of the communication between Near-RT RIC and the CU and the DU entities.

when the MAC scheduler pulls data from them, every Transmission Time Interval (TTI). With this comprehensive data, the Non-RT RIC trains predictive AI/ML models to identify the network patterns that influence the fluctuations of the DRB queues.

The Near-RT RIC utilizes the trained AI/ML models to make informed control decisions through its microservices, named xApps, based on the current received measurements, as Fig. 2 depicts. It continuously collects the necessary measurements by requesting E2SM-KPM [23] through report messages, forecasts the anticipated fluctuations of the DRB queues, and then broadcasts these predictions to the CU. Finally, the CU employs the prediction data from the Near-RT RIC to avoid the T_c latency.

3.3. The challenge of accurate time forecasting

For effective model training and inference, the communication latencies between the Near-RT RIC and the CU or the DU, denoted as T_{RC} and T_{RD}, **must be considered**. As the left part of Fig. 3 illustrates, the Near-RT RIC periodically sends report messages to the CU and the DU entities every T_r to retrieve their KPM. These report messages undergo a latency of T_{RC} and T_{RD} to reach the CU and the DU, respectively. Upon receiving a report request, each entity promptly responds to the Near-RT RIC with its KPM. Therefore, the Near-RT RIC receives these KPM responses at regular intervals of T_r, each undergoing a latency T_{RC} or T_{RD}, depending on its source.

As shown in the right part of Fig. 3, the Near-RT RIC periodically collects M and N KPM from the CU and the DU over a duration MT_r and NT_r, respectively. It then forecasts K values, representing the status measurements of the DRB queues for a duration of KT_w. The predictive models at the Near-RT RIC ensure accurate time alignment and synchronization with the 5G operation by factoring in the T_{RD}, T_{RC} latencies alongside the inference time, denoted as T_m, in their predictions. More specifically, the Near-RT RIC delivers its predictions to the CU after a cumulative elapsed time of $T_{RD} + T_m + T_{RC}$ and $T_m + 2T_{RC}$, measured from the time reference of the most recent measurements received by the DU and the CU before forecasting. By incorporating T_{RD}, T_{RC}, and T_m into the prediction process, our approach ensures that any timing misalignment between the predicted values and the actual state of the DRB queues is corrected. As a result, when the CU receives the predictions every T_m, it can rely on them to make decisions, as they accurately reflect the current state of the DU.

In DRQL, however, the SDAP scheduler cannot use these predictions directly, as they are discrete values, each representing the status of the DRB queues for a duration of $T_w = T_m/K$. Instead, SDAP forwards packets as they arrive at the QFI queues, potentially transmitting multiple packets within a single T_w interval. Similarly to our previous research [5], the SDAP scheduler mitigates this issue by linearly interpolating the intermediate values within the T_w interval to estimate the measurements of the DRB queues based on the received predictions.

4. Implementation framework

This section describes the implementation framework we employed to develop and evaluate our proposed solution, including the 5G configuration and AI/ML training.

4.1. 5G deployment

Our implementation of DRQL in 5G, called OAI5G-DRQL, is an extension of the original OAI5G [24], a 5G-RAN implementation, where we introduced several enhancements:

- Enhancement of the SDAP layer to include multiple SDAP queues and a Round-Robin (RR) scheduler for packet forwarding.
- Modification of the RLC layer, enabling it to dynamically adjust its DRB queue limits based on the volume of downlink packets it receives.
- Extension of the currently implemented KPM sent from DU to CU to accommodate additional, not-yet available, RLC and SDAP queue-specific information, as expected in E2SM-KPM.

In our research, we rely on the Radio Frequency (RF) Simulator to simulate exclusively part of the PHY layer of the 5G-RAN. While the NITOS testbed supports the deployment of the 5G-RAN using software-defined radios, it lacks the hardware necessary to emulate realistic channel conditions, including channel noise and mobility. The RF Simulator bridges this gap by accurately replicating these phenomena in software, delivering results that closely approximate those obtained from hardware-based approaches. Moreover, NR-UE emulates a commercial off-the-shelf (COTS) UE connected to 5G-RAN. A `Python3`

Non-RT RIC performs data preprocessing and model training, while FlexRIC [25] is the Near-RT RIC as it connects to the 5G-RAN nodes via the E2 interface, allowing monitoring and control via report and control messages. To accurately configure the latencies between network components, we deploy the CU, DU, and UE on a shared NITOS computational node. Furthermore, since the Near-RT RIC and the Non-RT RIC do not work simultaneously, we deployed both on a dedicated NITOS node with high computational capacity. Additionally, we deploy the 5G-CN, using OAI5G Core Network [26], on a separate NITOS node. Finally, we employ traffic control (`tc`) to control the latencies between each node within the NITOS network.

4.2. Dataset creation

To train the AI/ML models and predict the status of the DRB queues with supervised learning, we create a comprehensive dataset [27] under a range of network requirements and conditions. The developed dataset encompasses the behavior of network entities under DRQL in different bufferbloat scenarios. To cause bufferbloat, we use a single network slice along with `iperf3` [28] and `ping` [29], emulating the conflicting high-throughput and low-latency requirements of eMBMS and URLLC. Subsequently, we emulate the different bufferbloat scenarios by incorporating diverse application requirements. By adjusting the downlink transmission rate in `iperf3`, we replicate applications such as file downloading, video streaming, and cloud gaming. Furthermore, adjusting the `ping` interval enables the emulation of applications such as online gaming and video conferencing.

To more closely mirror real-world conditions, we encompass complex and diverse channel quality to the produced dataset, enhancing its realism and relevance. More specifically, we explored a couple of channel and UE mobility models and incorporated them into the dataset. The first model featured a stationary UE on a noise-free channel, while the second one involved a UE that follows a movement pattern according to the normal distribution on a channel containing Additive White Gaussian Noise (AWGN). We refer to these models as *real-wave*, designating the former as the *static-wave* and the latter as the *dynamic-wave*. To simulate the UE movement on the *dynamic-wave*, we use channel models, a channel simulation feature integrated into the latest versions of the RF Simulator. More specifically, we adjust the path loss parameter to account for changes in signal strength as the UE moves through different propagation environments, ensuring a realistic representation of dynamic channel conditions. For the mobile UE on the *dynamic-wave*, we adjusted the path loss parameter following the normal distribution, with values ranging from a minimum of 6dB to a maximum of 30dB to emulate a similar channel quality range to that of dataset [30]. On the other hand, for the stationary UE on the *static-wave*, we set the default path loss parameter of 0 dB, reflecting ideal channel conditions.

Finally, the developed dataset contains the Key Performance Measurements (KPM) of the 5G-RAN, sampled at $T_r = 1$ ms under DRQL in a monolithic network deployment, characterized by $T_c = 0$. To create distinct network downlink conditions, we employ combinations of different parameters in `iperf3` and `ping` alongside the previously introduced *real-wave models*. These combinations include UDP traffic at $[1, 10, 40, 100]$ Mbps alongside ICMP transmission with intervals of $[1, 10, 100]$ ms, ensuring the accuracy and relevance of the trained models and their predictions across diverse applications. Therefore, the compiled dataset includes entries representing the KPM with their respective timestamps.

4.3. Data preprocessing

Accurately forecasting the fluctuations of the precise limits and actual size measurements of the DRB queues on the *dynamic-wave* presents an inherent complexity, necessitating adjustments to our AI/ML approach. In contrast to our previous *dual-model* approach, where the Non-RT RIC used a couple of separate models to predict the actual size and limit of the DRB queues based solely on their respective historical fluctuation, now we propose a new *single-model* method that uses only one predictive model. The Non-RT RIC multiplexes the actual size with the limit of the DRB queues, which depend on each other, into a single variable called the remaining size. We define the remaining size as the difference between the limit and the actual size. In this way, it reduces the additional computing power required by the two AI/ML models, suggested in our preliminary study, to avoid facilitating parallel model inference for a more consistent forecasting horizon, minimizing frequent fluctuations in its inference duration. Similarly, the Near-RT RIC computes this remaining size from its KPM before performing inference.

According to the new *single-model* approach, the process begins with preprocessing the provided dataset for model training, which includes the KPM of the entire network. More specifically, we transform the prediction into a multivariate time series forecasting problem, considering the latencies that the Near-RT RIC has with the CU and the DU entities. As illustrated in Fig. 3 for the same report request, sent by the Near-RT RIC to the CU and DU entities, the received KPM differ by $2|T_{RD} - T_{RC}|$. The Non-RT RIC should incorporate this timing difference in its preprocessing stage and accurately align the measurements before training. It achieves this by deriving the feature-label pairs with a rolling window approach, preserving causality by refraining from data shuffling. For the measurements arriving from the DU, the Non-RT RIC applies the rolling window technique, considering the $T_{RD} + T_m + T_{RC}$ time interval to account for the DU to Near-RT RIC, model inference, and Near-RT RIC to CU latencies. Likewise, the Non-RT RIC performs the same technique to the measurements received from the CU, considering the $T_m + 2T_{RC}$ time interval to incorporate the CU to Near-RT RIC, model inference, and Near-RT RIC to CU latencies.

Since not every KPM is equally critical in the prediction process, we can prune unnecessary KPM from the dataset to avoid overfitting and save on hardware resources. To find which KPM are essential, we employ a *Random Forest* regressor, an extension of the Random Forest algorithm, with 100 decision trees to classify which input features affect the state of the DRB queues the most. The output of the regressor indicates that the model is primarily affected by the latest KPM values, specifically the limit and actual size of the DRB queues. Additionally, several other factors significantly impact the accuracy of the model. These include KPM regarding the rate at which the RLC receives its data from the upper layers and transmits it to the lower ones, the size and occupancy of the QFI queues, and the block error rate at the MAC layer. Furthermore, the amount of data transmitted and received within a MAC frame and slot, the detected signal-to-noise ratio, and the data volume reported in the MAC buffer status report also affect the accuracy of the predictive model.

Some KPM values are aggregated values that can increase indefinitely, causing inconsistencies in the scaling procedure since the scaling procedure partly relies on the highest value of each KPM to scale it. We compute the difference between two consecutive observations of these KPM values, creating the modified features to feed into the time series forecasting model. The Near-RT RIC follows a similar procedure on these KPM before performing inference. In the final step of the preprocessing stage, the Non-RT RIC scales the data in the $[0, 1]$ range using `MinMaxScaler`, provided by scikit-learn [31], and splits the dataset into 70% training, 20% validation, and 10% test sets. The Near-RT RIC also scales its KPM before performing inference with the same scaler.

4.4. Model architecture and training

Our preliminary studies and experiments indicate that Long Short-Term Memory (LSTM) models outperform other predictive models, including Seasonal Autoregressive Integrated Moving Average (SARIMA) and Random Forests for this forecasting. The Non-RT RIC

employs TensorFlow [32] with Keras [33] to handle the development and training of the LSTM model, while KerasTuner optimizes training hyperparameters for precise forecasting results. We employ `tensorflow.keras.sequential` for both models to stack multiple layers. We specify that the input layer for the model is an LSTM layer consisting of N nodes. KerasTuner determines the parameters for the subsequent LSTM layers, including their quantity and configurations. The final layers, also determined by KerasTuner, are fully connected dense layers that produce predictions aligned with the dimensions of our target output labels, denoted as K in this context. We utilize the Mean Squared Error (MSE) as our chosen loss function, effectively penalizing errors and providing remarkable overall model performance. The choice of optimizer algorithm is also pivotal in shaping how errors propagate through the network. The Adam optimizer is an excellent choice since it strikes an optimal balance between learning the most relevant and less frequent features.

To avoid overfitting, we add a Dropout layer between the LSTM and the Dense layers, in the range of 0 to 0.5 and a step size of 0.1. The Dropout layer prevents overfitting by randomly deactivating a fraction of neurons during training. Additionally, we simplify the model by restricting KerasTuner to a maximum of three LSTM layers and three Dense layers, each containing a small number of neurons within the range of 32 to 512. We also implement early stopping during training to monitor the model accuracy on the validation set and halt training when the performance starts to degrade.

The dimensions of the output labels highly depend on the inference duration of the model since $KT_w \approx T_m$, which we calculate as the mean inference duration obtained from multiple observations of model inference. If these two do not align, the model is invalid, as it fails to accurately capture the timing necessary for reliable predictions and subsequent actions within the system. Therefore, aligning the duration the output labels represent with the inference duration ensures that the predicted labels align with the temporal dynamics of the system, allowing for accurate forecasting and effective decision-making based on the inferred results. Finally, for simplicity in our experiments, we assume $T_w = T_r$.

5. Evaluation results

The effectiveness of our implementation relies significantly on the precision of the trained models utilized by the Near-RT RIC to forecast the missing information for the CU to support OAI5G-DRQL. The subsequent Section 5.1 assesses the level of precision achieved by the models. Sections 5.2 and 5.3 provide a performance evaluation of the effective throughput and latency, with the latter measured as the Round Trip Time (RTT). Throughput and latency are the fundamental metrics commonly used in the literature to effectively and sufficiently capture the performance of an AQM algorithm. In our study, these metrics are the most significant evaluation criteria as they directly reflect on the accuracy of the predictions, with inaccuracies adversely impacting them the most. More specifically, Section 5.2 evaluates our implementation in the *static-wave* scenario, where the channel quality remains consistently strong, making forecasting straightforward. This experiment demonstrates the performance improvements our contribution provides under optimal network conditions. Section 5.3 showcases our implementation under the realistic network conditions of the *dynamic-wave*. The significant increase in performance of our implementation in this experiment further establishes the effectiveness of our approach as a robust solution for integrating AQM under disaggregated 5G-RAN deployments.

5.1. Evaluation of model accuracy

The validity of our implementation heavily depends on the accuracy of the trained models, which we measure using Normalized Root Mean Square Error (NRMSE). NRMSE is the optimal metric for assessing model accuracy for our research as it is sensitive to large deviations, highlighting the critical prediction errors that can lead to congestion or inefficient utilization of the DRB queues and significantly impact network performance. Additionally, the NRMSE normalizes the error, enabling a clear performance comparison across the different models and varying conditions.

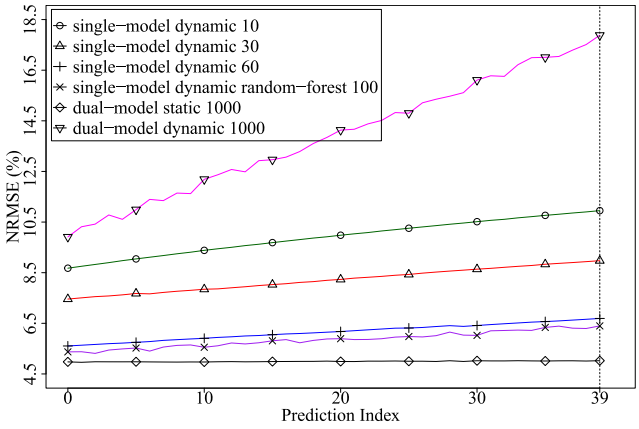

Fig. 4. NRMSE of predictions.

Fig. 4 depicts the NRMSE of each of the $K = 40$ predictions of the trained models, compared to their actual respective RLC measurements. This comparison highlights the performance of the models and provides the basis for selecting the optimal predictive model. Each line depicts the NRMSE of the $0 \leq k^{\text{th}} \leq K-1$ prediction within a future horizon of K predictions for each model, under various conditions: *single* or *dual model*, using *Random Forest* or not, within *dynamic* or *static wave*, as well as how many $N = M$ KPM used for forecasting. It is evident that as the prediction horizon extends, the corresponding NRMSE also increases.

The accuracy of our predictions, derived from the *dual-model* approach in our previous study for the *static-wave* with $N = M = 1000$ input features, remains consistently low, illustrated by the diamond-pointed line, with an error rate below 5%. However, the same approach does not perform nearly as accurately on the *dynamic-wave*, at least doubling the NRMSE to 10.5% and increasing it linearly with the less recent predictions (with higher k), as illustrated by the inverted-triangle-pointed line. In contrast, the new *single-model* approach outperforms the previous one within the *dynamic-wave*, whether using as input features the KPM from the overall 5G-RAN stack or the pruned KPM produced by the *Random Forest* algorithm.

As the problem involves multivariate time series forecasting, the hardware memory requirements increase exponentially with the number of input features used. Fig. 4 also illustrates the relationship between the number of input features and the model accuracy. Firstly, we evaluate the model trained using the KPM of the overall 5G-RAN stack. When using the $N = M = 60$ latest KPM as input features, the NRMSE, represented by the cross-pointed line, is the lowest for the *dynamic-wave* at approximately 5.5%. For a lower number of input features, $N = M = 30$ and $N = M = 10$, represented by the triangle-pointed and circle-pointed lines, the NRMSE slightly increases to around 7.5% and 8.5%. Considering we have prevented overfitting, this minimal increase in NRMSE indicates that the most recent KPM values primarily impact the future behavior of the DRB queues in RLC. Therefore, by reducing the number of input features, we conserve memory resources without significantly sacrificing accuracy.

Furthermore, we enhanced the accuracy further by incorporating more input features with reduced information from each KPM. More specifically, we use the pruned KPM produced by the *Random Forest* algorithm and increase the number of input features to $N = M = 100$ without requiring additional memory. The model that uses 100 pruned KPM, marked by the x-pointed, achieves a lower NRMSE than the

60 KPM model, marked by the cross-pointed line. We attribute this reduction in NRMSE to the increased crucial information fed into the network for forecasting. This optimized model is the model we will be using in the subsequent experiments.

Considering model accuracy, inaccurate predictions do not necessarily imply invalid decision-making under real-time execution. To evaluate the accuracy of the decision-making process, we define two additional metrics regarding the percentage of false positives and false negatives. These metrics more accurately reflect the effect of the predictions in our approach as they directly correlate to the percentage of incorrect decisions our implementation makes under realistic network conditions. False positives might occur when the prediction over-evaluates the underlying channel and exceeds the expected remaining size, which indicates that the SDAP might forward packets to the DRB queues even if they do not fit, potentially resulting in packet drops. False negatives, on the other hand, may emerge when the predictions under-evaluate the channel and are lower than the expected remaining size, suggesting that the SDAP will not forward packets even though it could. Fig. 5 depicts the percentage of remaining size relative to the actual limit of the DRB queues under real-time monitoring on the *dynamic-wave*, sampled every 1 ms for a time horizon of 500 ms. The prediction value used by the SDAP scheduler regarding the remaining size of the DRB queues, marked by the triangle-pointed line, follows closely the trend of the expected remaining size, marked by the circle-pointed line. Moreover, even when the prediction over-evaluates the channel and exceeds the expected remaining size, the average percentage of false positives is measured below $8\% \pm 0.5\%$ for all the experiments conducted. Similarly, when that channel is under-evaluated, the average percentage of false negatives is even less and approximately equal to $1\% \pm 0.8\%$.

This discrepancy between the percentages of false positives and false negatives emerges due to the lack of KPM that sufficiently describe the latency between the CU and the DU. False positives arise as the SDAP overestimates the capacity of the DRB queues and forwards packets without accounting for those residing on the buffers of the interprocess communication channels between the CU and the DU into consideration. For false negatives to occur, however, two conditions must be met. Firstly, the SDAP must correctly underestimate the capacity of the DRB queues and fail to consider that this capacity might increase within the next TTI. Secondly, the buffers of the interprocess communication channels must not be able to forward enough packets to fill the DRB queues within the next TTI. Due to the nature of the high-throughput experiments within the high-latency environment, the buffers on the interprocess communication channels between the CU and the DU entities remain congested. This congestion, along with the properties of these high-throughput interprocess buffers, decreases the probability of false negatives by reducing the likelihood that the second condition occurs while simultaneously increasing the probability of false positives.

5.2. Evaluation of performance on the static-wave

5.2.1. Experiment description

This experiment investigates our proposed AQM solution for OAI5G-DRQL in *static-wave*, assessing its performance in terms of effective throughput and latency under bufferbloat. Specifically, we connect a single UE to the 5G-RAN that establishes only one Packet Data Unit (PDU) session with distinct QFI queues for the UDP and the ICMP packets, both mapping to a single RLC Acknowledge Mode (AM) DRB, as depicted in Fig. 6. We then generate 50 Mbps of downlink UDP traffic from the 5G-CN while transmitting ICMP packets every 100 ms to the UE. Finally, we measure the throughput of the UDP traffic and the RTT of the ICMP traffic at the UE.

We conduct three sets of experiments with the *static-wave* and a monolithic and a disaggregated gNB. The monolithic gNB is distinguished by $T_c \approx 0$ while the disaggregated gNB by $T_c \approx 10$ ms [34],

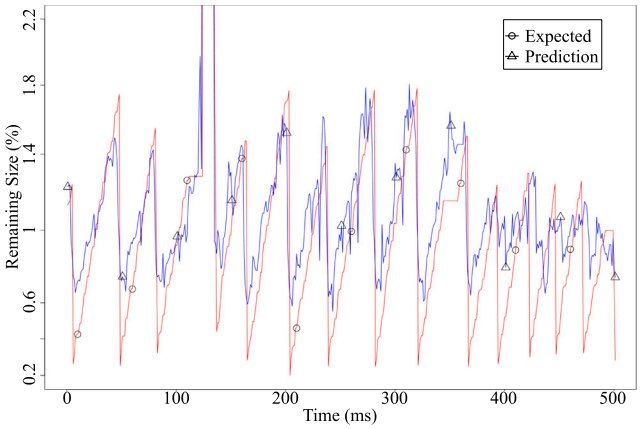

Fig. 5. Real-time predictions and the expected percentage of the remaining size.

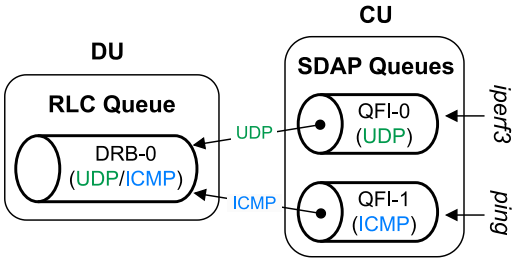

Fig. 6. ICMP and UDP QFI queues mapped to a single DRB.

which is the maximum fronthaul latency allowed in 5G. **In the first set of experiments, we use the original OAI5G**, which supports a drop-tail AQM in the DRB queues, restricting the queue sizes to a predetermined limit by discarding incoming packets that would surpass it. However, it does not support an AQM algorithm that requires cross-layer communication. In OAI5G, the SDAP forwards the packets from its QFI queues, as they become available, to their respective DRB queue without querying any information from the RLC. This experiment highlights the baseline performance, demonstrating the effectiveness of the original OAI5G system while also revealing the issues that arise when bufferbloat occurs, emphasizing the importance of AQM algorithms that consider the entire 5G-RAN to operate. **In the second set of experiments, we use OAI5G-DRQL without RIC**, which required direct communication between the CU and the DU entities. This experiment demonstrates the performance degradation of DRQL when introducing latency between the CU and DU entities. Finally, **the third set of experiments assesses OAI5G-DRQL with RIC**, where the Near-RT RIC employs the trained multivariate time series forecasting model to provide the anticipated status measurements of the DRB queues to the CU. Since the Near-RT RIC is collocated with the CU, the latency between Near-RT RIC and DU is $T_{RD} = T_c \approx 10$ ms, while the latency between Near-RT RIC and CU is $T_{RC} \approx 0$.

In OAI5G-DRQL with RIC, the model at the Near-RT RIC uses the latest $N = M = 100$ KPM, representing the metrics of the entire 5G-RAN over a duration of $100T_r = 100$ ms, to forecast $K = 40$ values. These forecasts indicate the remaining size of the DRB queues for the next $40T_w = 40$ ms, starting from the moment the CU receives them. To ensure that the predictions accurately align with the actual DRB queue measurements over time, the model incorporates the inference time $T_m \approx 40$ ms and the reporting latencies $T_{RC} \approx 0$ and $T_{RD} \approx 10$ ms. Therefore, the model uses 100 ms of historical data to predict the subsequent 40 ms, following a time interval of $T_{RD} + T_m + T_{RC} \approx 50$ ms from the generation of the latest KPM received by the Near-RT RIC.

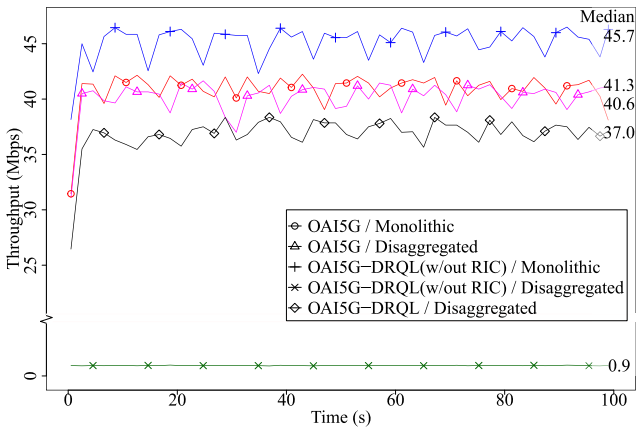

Fig. 7. Effective throughput on the *static-wave* experiments.

5.2.2. Experiment results

Each set of experiments was conducted twelve times, with the best and worst runs discarded to minimize outliers. The results presented reflect the average performance observed across the remaining runs, along with the statistical accuracy, measured as the mean variability of the metrics across the remaining ten experiments. Fig. 7 illustrates the effective throughput measured at the UE for each set of experiments. As the DRB queues are empty at the beginning of each experiment, the MAC scheduler has less data to process, resulting in a relatively low initial throughput. As the DRB queues fill up, the MAC scheduler processes more data, gradually increasing throughput. Eventually, the throughput stabilizes at a higher level, representing the maximum achievable throughput the MAC scheduler can process, adhering to the capacity limitations of the wireless channel.

In the first set of experiments, the original OAI5G shows little to no impact on throughput from the monolithic or disaggregated deployment of the gNB, as it remains relatively unaffected regardless of the deployment. We observe that the effective throughput is $41.3 \pm 0.4\%$ Mbps for a monolithic gNB and $40.6 \pm 0.4\%$ Mbps for a disaggregated one, represented by the red circle-pointed and the pink triangle-pointed lines. This consistency in throughput alongside the $\pm 0.4\%$ mean variability across our experiments demonstrates the minimal influence of the deployment option in the original OAI5G, as the CU maintains a steady packet forwarding rate, regardless of the latency between the CU and the DU.

In the second set of experiments, the OAI5G-DRQL without RIC in a monolithic gNB deployment achieves higher throughput than the original OAI5G, with an average of $45.7 \pm 0.8\%$ Mbps, as shown by the blue cross-pointed line. The observed increase in throughput is driven by the dynamic fluctuation of the DRB queues in response to the available radio channel capacity, allowing the MAC scheduler to process strictly the amount of data needed within every single TTI. More specifically, in OAI5G-DRQL, the MAC scheduler avoids processing the unnecessarily large RLC queues that the original OAI5G would, containing data that would eventually be dropped and retransmitted due to insufficient channel capacity. Therefore, it saves on resources, increasing the overall effective throughput while avoiding unnecessary starvation and overflow of the DRB queues. On the other hand, OAI5G-DRQL without RIC features very low throughput at almost $0.9 \pm 0.1\%$ Mbps in a disaggregated gNB deployment, represented by the green x-pointed line. This decrease in throughput occurs as every packet at the QFI queues undergoes the high-latency communication between the CU and the DU entities for DRQL to operate. This latency causes a bottleneck at CU and leads to starvation of the DRB queues, making DRQL completely redundant which justifies the low variability across our experiments.

In the third set of experiments, we observe that the OAI5G-DRQL with RIC eliminates the throughput degradation observed in OAI5G-DRQL without RIC experiences in disaggregated gNB deployments. Specifically, OAI5G-DRQL with RIC effectively avoids the direct latency-heavy communication between its CU and DU entities, making its decisions based on the predictions received by the Near-RT RIC and achieving $37 \pm 1.2\%$ Mbps throughput, as depicted by the black diamond-pointed line. Its throughput is slightly lower than the original OAI5G, which is an acceptable tradeoff given the significantly decreased RTT it achieves, as presented below.

To assess the effectiveness of OAI5G-DRQL with RIC, we also investigate the RTT of the ICMP packets as they are affected by the high throughput in the same DRB queues. In Fig. 8, we present the RTT in a logarithmic scale, with the actual average RTT measurements on the y-axis, to better visualize its lower and higher values within the same graph. In the first set of experiments, we observe the negative impact on RTT due to the absence of an AQM algorithm that does not consider the whole network to operate, such as the drop-tail approach in the original OAI5G. In the original OAI5G, the SDAP immediately forwards packets to the DRB queues, regardless of their QFI. Since the volume of the UDP packets is higher than that of the ICMP packets, they accumulate in front of the ICMP packets in the DRB queues. As the ICMP packets lag behind the UDP packets, the RTT increases linearly regardless of whether the gNB deployment is monolithic or disaggregated, as illustrated by the circular and triangular-pointed lines, respectively. In the disaggregated gNB deployment, the RTT growth rate increases slightly compared to the monolithic deployment, primarily due to higher initial acceleration in the accumulation of UDP packets in the DRB queues.

In the second set of experiments, we observe that OAI5G-DRQL without RIC significantly reduces RTT compared to the original OAI5G, highlighting the importance of cross-layer AQM algorithms. Although it may seem counterintuitive, OAI5G-DRQL without RIC in a disaggregated gNB deployment, depicted by the green x-pointed line, features a lower RTT of $40.5 \pm 0.1\%$ ms compared to the $101.5 \pm 0.9\%$ OAI5G-DRQL without RIC in a monolithic deployment, shown by the blue cross-pointed line. As our previous study thoroughly explains, this occurs due to the nature of the Round-Robin SDAP scheduler and the latency between the CU and the DU entities. More specifically, in DRQL, the SDAP scheduler experiences this T_c latency for every packet in its QFI queues. When $T_c \approx 0$, the SDAP scheduler predominantly forwards UDP traffic, as the QFI queue for the UDP packets almost always contains UDP packets due to their higher transmission frequency and volume. In contrast, the QFI queue for ICMP packets remains mostly empty, as the ICMP packets arrive less frequently. Therefore, UDP packets accumulate in front of the ICMP packets at the DRB queues, increasing RTT. However, when $T_c \approx 10$, the SDAP scheduler experiences at least $2T_c \approx 20$ ms on each UDP packet to decide whether to forward it. As the SDAP scheduler experiences this latency, it forwards the UDP packets at a notably slower pace while the ICMP queues begin to fill up. Since both QFI queues have data, the SDAP scheduler interchangeably forwards the UDP and the ICMP traffic. Thus, the ICMP packets do not lag behind the UDP packets at the DRB queues, decreasing RTT at the expense of significantly reduced throughput, as already shown.

In the third set of experiments, we compare the performance of OAI5G-DRQL with RIC in a disaggregated gNB deployment with the OAI5G-DRQL without RIC in a monolithic deployment. In both cases, downlink traffic avoids the additional DRQL communication overhead, preventing the impedance of the SDAP scheduler and ensuring robust decision-making for packet forwarding. Specifically, OAI5G-DRQL with RIC achieves an RTT of $80.1 \pm 1.1\%$ ms, depicted by the black diamond-pointed line, approximately 20 ms higher than the monolithic OAI5G-DRQL without RIC. This 20 ms increase in RTT occurs as each ICMP packet undergoes $2T_c \approx 20$ ms round-trip delay between the CU and the DU.

The experiments demonstrate the importance of our approach to integrating sophisticated AQM algorithms that require cross-layer communication in disaggregated high-latency deployments. We showcased

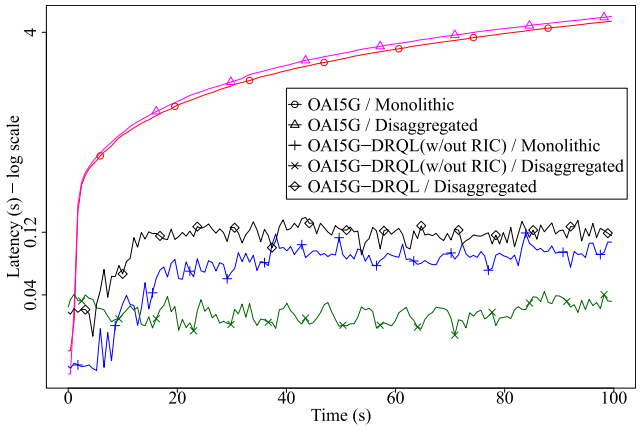

Fig. 8. Achieved RTT on the *static-wave* experiments.

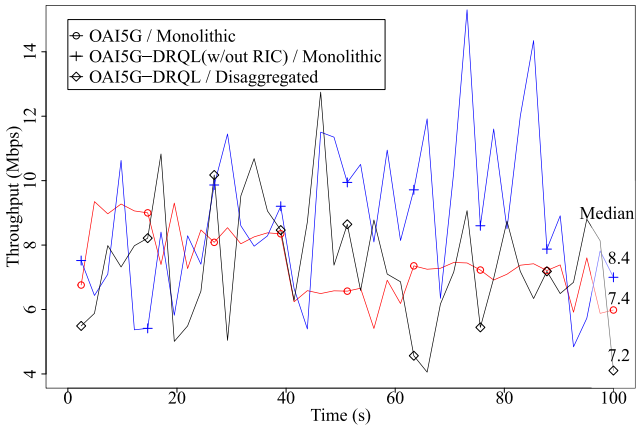

Fig. 9. Effective throughput on the *dynamic-wave* experiments.

that our proposed OAI5G-DRQL with RIC enhances throughput compared to OAI5G-DRQL without RIC in disaggregated deployments, approaching the performance of a monolithic gNB while maintaining relatively low RTT values. These improvements mitigate bufferbloat with improved AQM in disaggregated deployments with minimal performance degradation. Therefore, our solution enables AQM algorithms, initially designed for monolithic gNB deployments, to be employed in disaggregated *static-wave* ones while maintaining stable latency and high throughput levels.

5.3. Evaluation of performance on the dynamic-wave

5.3.1. Experiment description

To evaluate our study under realistic network conditions, we conducted the previous sets of experiments with the *dynamic-wave* across different deployment options. More specifically, we evaluated our approach with the monolithic deployment of the original OAI5G, the previously proposed monolithic OAI5G-DRQL without RIC, and our proposed disaggregated OAI5G-DRQL with RIC. These implementations are particularly significant as the original OAI5G demonstrates the necessity for AQM, whereas our OAI5G-DRQL with RIC aims to achieve comparable performance in disaggregated gNB deployments to that of OAI5G-DRQL without RIC. Finally, the primary distinction in this set of tests is the *dynamic-wave* network conditions, as opposed to the previously used *static-wave*.

5.3.2. Experiment results

To correctly calculate the statistical accuracy, we perform these experiments twelve times under consistent noise and mobility conditions and remove two outliers. In line with the previous experiments, Fig. 9 presents the throughput achieved for each set of deployment options, specifically within the *dynamic-wave*. The introduction of noise and mobility in the *dynamic-wave* reduces throughput from the 25 to 46 Mbps initial range, observed in the *static-wave*, down to a range of 4 to 15 Mbps. More specifically, the overall reduction in throughput occurs due to channel noise, with its fluctuations resulting from mobility.

As illustrated in the previous experiments, with the *static-wave*, the monolithic OAI5G-DRQL without RIC and the original OAI5G with a monolithic gNB outperform our proposed by OAI5G-DRQL with RIC by approximately 19% and 10.4%, respectively. Similarly, with the *dynamic-wave*, our proposed OAI5G-DRQL with RIC performs nearly as well as the OAI5G-DRQL without RIC and the original OAI5G, with a slight performance decrease of 2.7% and 16.6%, respectively. Additionally, we observe that this performance is consistent as the mean variabilities of OAI5G-DRQL with RIC, OAI5G-DRQL without RIC, and the original OAI5G are ±3.8%, ±2.8%, and ±2.7%, respectively.

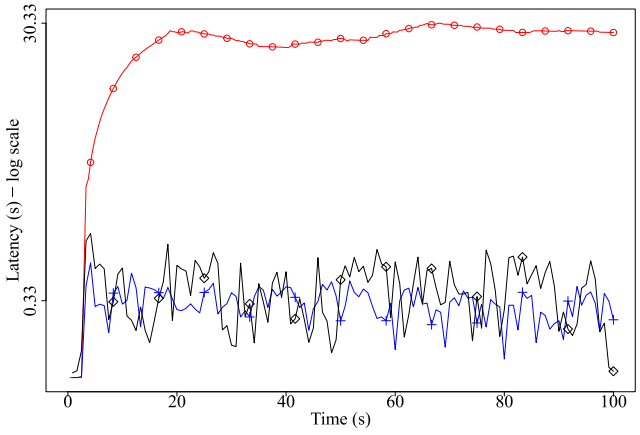

Fig. 10. Achieved RTT on the *dynamic-wave* experiments.

Therefore, this reduction in throughput is justified as it enables the disaggregation of the 5G-RAN while maintaining low latency, as indicated by decreased RTT values, as presented below.

Fig. 10 depicts the RTT values of the distinct gNB deployments within the *dynamic-wave* in logarithmic scale with the actual average RTT measurements on the y-axis. We observe that the RTT of the original OAI5G deployment increases similarly to the previous *static-wave* experiments, even in the *dynamic-wave*. This increase in RTT suggests that the traffic volume impacts the network performance more significantly than the properties of the underlying channel, including noise and mobility.

As expected, the RTT of the OAI5G-DRQL with and without RIC decreases and remains stable throughout the experiments, compared to the original OAI5G. Similarly to the previous experiments, we observe approximately 20 ms increase in latency for our proposed OAI5G-DRQL with RIC compared to the previously proposed OAI5G-DRQL without RIC. This increase in latency, from $320.4 \pm 2.5\%$ ms to $348.3 \pm 3.9\%$ ms, occurs due to the additional $2T_c \approx 20$s ms round-trip latency that the ICMP packets experience. Finally, when we compare the *static-wave* with the *dynamic-wave*, we observe that the overall RTT increases in the *dynamic-wave*. This increase in RTT is due to the unstable nature of the underlying physical channel as the*dynamic-wave* has higher packet drop rates due to high noise and mobility. High drop rates lead to more retransmissions, which consequently lower the rate at which the MAC layer can push packets downlink to the UE and reduce the size of the DRB queues, eventually increasing the overall latency.

The *dynamic-wave* experiment results show that our proposed OAI5G-DRQL with RIC provides performance benefits comparable to

those of the previous experiments, achieving throughput and latency levels similar to those of the monolithic OAI5G-DRQL without RIC. Additionally, our implementation demonstrated a stable and low RTT compared to the high RTT of the original OAI5G, with only a slight decrease in throughput. These improvements verify the validity of our approach, addressing bufferbloat with improved AQM in realistic disaggregated high-latency deployments in realistic environments *dynamic-wave* with noise and mobility. Thus, our solution enables the adaptation of various AQM algorithms, such as DRQL, from monolithic 5G networks to disaggregated ones.

6. Conclusions & future work

This paper introduces an innovative approach that addresses AQM in disaggregated 5G and beyond cellular networks where communication between the CU and the DU entities is required. Our approach addresses realistic and complex network conditions by leveraging the ability to incorporate AI/ML into the RIC-based architecture of the beyond 5G networks. This integration allows us to effectively manage previously proposed AQM algorithms in disaggregated deployments by considering the latency of communication exchange between the CU and the DU entities and the dynamic nature of modern cellular networks. Future work will focus on further refining the proposed AQM solution by incorporating AI/ML techniques on networks containing multiple mobile UE entities and multiple DU entities. This refinement will enable our solution to more effectively deliver a comprehensive, unified approach rather than focusing on individual network entities in isolation.

CRediT authorship contribution statement

Alexandros Stoltidis: Writing – original draft, Software. **Kostas Choumas:** Writing – review & editing, Conceptualization. **Thanasis Korakis:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The research leading to these results has received funding from the European Horizon 2020 Programme for research, technological development and demonstration under Grant Agreement Number No. 101008468 (H2020 SLICES-SC). The European Union and its agencies are not liable or otherwise responsible for the contents of this document; its content reflects the view of its authors only.

Data availability

No data was used for the research described in the article.

References

[1] Shunliang Zhang, An overview of network slicing for 5G, IEEE Wirel. Commun. 26 (3) (2019) 111–117.
[2] ETSI, Release 15, 2018, https://www.etsi.org/deliver/etsi_tr/138900_138999/138912/15.00.00_60/tr_138912v150000p.pdf.
[3] P. Popovski, K.F. Trillingsgaard, O. Simeone, G. Durisi, 5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view, IEEE Access 6 (2018) 55765–55779.
[4] M. Irazabal, E. Lopez-Aguilera, I. Demirkol, R. Schmidt, N. Nikaein, Preventing RLC buffer sojourn delays in 5G, IEEE Access 9 (2021) 39466–39488.
[5] Alexandros Stoltidis, Kostas Choumas, Thanasis Korakis, Active queue management in disaggregated 5G and beyond cellular networks using machine learning, in: Proc. WONS, 2024.
[6] M. Irazabal, E. Lopez-Aguilera, I. Demirkol, Active queue management as quality of service enabler for 5G networks, in: Proc. EuCNC, 2019.
[7] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Trans. Netw. 1 (4) (1993) 397–413.
[8] K.M. Nichols, V. Jacobson, A. McGregor, J.R. Iyengar, Controlled delay active queue management, RFC 8289 (2018) 1–25.
[9] R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, B. VerSteeg, PIE: A lightweight control scheme to address the bufferbloat problem, in: Proc. IEEE High Performance Switching and Routing, HPSR, 2013.
[10] T. Høiland-Jørgensen, P.E. McKenney, D. Täht, J. Gettys, E. Dumazet, The flow queue CoDel packet scheduler and active queue management algorithm, RFC 8290 (2018) 1–25.
[11] Osel Lhamo, Mingyu Ma, Tung V. Doan, Tobias Scheinert, Giang T. Nguyen, Martin Reisslein, Frank H.P. Fitzek, RED-SP-CoDel: Random early detection with static priority scheduling and controlled delay AQM in programmable data planes, Comput. Commun. 214 (2024) 149–166.
[12] Huihui Ma, Du Xu, RLRBM: A reinforcement learning-based RAN buffer management scheme, in: Proc. of Mobility, Sensing and Networking, MSN, 2022.
[13] A.K. Paul, H. Kawakami, A. Tachibana, T. Hasegawa, An AQM based congestion control for eNB RLC in 4G/LTE network, in: Proc. IEEE Canadian Conference on Electrical and Computer Engineering, CCECE, 2016.
[14] P.E. McKenney, Stochastic fairness queueing, in: Proc. IEEE INFOCOM, 1990.
[15] M. Irazabal, E. Lopez-Aguilera, I. Demirkol, N. Nikaein, Dynamic buffer sizing and pacing as enablers of 5G low-latency services, IEEE Trans. Mob. Comput. 21 (3) (2022) 926–939.
[16] Jianer Zhou, Xinyi Qiu, Zhenyu Li, Qing Li, Gareth Tyson, Jingpu Duan, Yi Wang, Heng Pan, Qinghua Wu, A machine learning-based framework for dynamic selection of congestion control algorithms, IEEE/ACM Trans. Netw. 31 (4) (2022) 1566–1581.
[17] Mikel Irazabal, Navid Nikaein, TC-RAN: A programmable traffic control service model for 5G/6G SD-RAN, IEEE JSAC 42 (2) (2024) 406–419.
[18] ETSI, NG-RAN; Architecture Description, Technical Report TS 138 401, ETSI, 2018, Version 15.2.0.
[19] N. Makris, C. Zarafetas, P. Basaras, T. Korakis, N. Nikaein, L. Tassiulas, Cloud-based convergence of heterogeneous RANs in 5G disaggregated architectures, in: Proc. IEEE ICC, 2018.
[20] O-RAN specifications, https://www.o-ran.org/specifications.
[21] Bharath Balasubramanian, E. Scott Daniels, Matti Hiltunen, Rittwik Jana, Kaustubh Joshi, Rajarajan Sivaraj, Tuyen X. Tran, Chengwei Wang, RIC: A RAN intelligent controller platform for AI-enabled cellular networks, IEEE Internet Comput. 25 (2) (2021) 7–17.
[22] O-RAN Alliance, O-RAN Architecture Description, Technical Report O-RAN.WG1.OAD-R003-v12.00, O-RAN Alliance, 2024, Technical Specification R003.
[23] O-RAN near-real-time RAN intelligent controller E2 service model (E2SM) KPM 2.0, 2021, O-RAN Working Group 3.
[24] Open air interface (OAI), https://openairinterface.org/.
[25] R. Schmidt, M. Irazabal, N. Nikaein, Flexric: An SDK for next-generation SD-RANs, in: Proc. CoNEXT, 2021.
[26] Open air core network 5G, https://gitlab.eurecom.fr/oai/cn5g/oai-cn5g-fed.
[27] Alexandros Stoltidis, Kostas Choumas, Thanasis Korakis, 5G RAN network-wide KPM under DRQL AQM for ML/AI training, 2024-2025, http://dx.doi.org/10.5281/zenodo.14678983.
[28] iPerf, https://iperf.fr/.
[29] Ping, https://linux.die.net/man/8/ping.
[30] K.M. Karthick Raghunath, Ravi Kumar Lanke, 5G-network-metrics-for-high-traffic-event, 2023.
[31] Fabian Pedregosa, et al., Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[32] Martin Ababi, et al., TensorFlow: Large-scale machine learning on heterogeneous distributed systems, 2016.
[33] Keras, https://keras.io/.
[34] Parallel Wireless, 5G functional splits, https://www.parallelwireless.com/wp-content/uploads/5G-Functional-Splits-V3.pdf.