

THIS IS AN AUTHOR-CREATED POSTPRINT VERSION.

Disclaimer: This work has been accepted for publication in *IEEE Transactions on Communications*.

Copyright: © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

DOI: 10.1109/TCOMM.2025.3552296

MAREA: A Delay-Aware Multi-time-Scale Radio Resource Orchestrator for 6G O-RAN

Oscar Adamuz-Hinojosa, Lanfranco Zanzi, Vincenzo Sciancalepore, Xavier Costa-Pérez

Abstract—The Open Radio Access Network (O-RAN)-compliant solutions often lack crucial details for implementing effective control loops at various time scales. To overcome this, we introduce MAREA, an O-RAN-compliant mathematical framework designed for the allocation of radio resources to multiple ultra-Reliable Low Latency Communication (uRLLC) services. In the near-real-time (RT) control loop, MAREA employs a novel Martingales-based model to determine the guaranteed radio resources for each uRLLC service. Unlike traditional queuing theory approaches, this model ensures that the probability of packet transmission delays exceeding a predefined threshold—the violation probability—remains below a target tolerance.

Additionally, MAREA uses a real-time control loop to monitor transmission queues and dynamically adjust guaranteed radio resources in response to traffic anomalies. To the best of our knowledge, MAREA is the first O-RAN-compliant solution that leverages Martingales for both near-RT and RT control loops. Simulations demonstrate that MAREA significantly outperforms reference solutions, achieving an average violation probability that is $\times 10$ lower.

Index Terms—Multi-scale-time, O-RAN, Real-Time RIC, Martingales, uRLLC.

I. INTRODUCTION

In Sixth Generation (6G) networks, a critical challenge is managing the coexistence of multiple ultra-Reliable Low Latency Communication (uRLLC) services, which impose stringent latency and reliability requirements to ensure seamless and efficient operations [1]. One of the primary advancements driving the evolution of 6G is the virtualization of the Radio Access Network (RAN) [2], achieved through the deployment of virtualized RAN (vRAN) instances. These instances consist of fully configurable virtualized Base Stations (vBSs), tailored to the specific demands of diverse communication services.

Background. To address the increasing complexity of these networks, the Open Radio Access Network (O-RAN) Alliance has introduced an innovative, flexible architecture [3] that integrates the 3rd Generation Partnership Project (3GPP) functional split. This architecture distributes the vBS across

multiple network nodes, including the Centralized Unit (CU)-Control Plane (CP), CU-User Plane (UP), Distributed Unit (DU) and Radio Unit (RU). Additionally, O-RAN enhances network agility and autonomy through its unique inclusion of two RAN Intelligent Controllers (RICs): the non-Real Time (RT) RIC, which handles long-term network optimizations like policy computation and Machine Learning (ML) model management via third-party applications (*rApps*), and the near-RT RIC, which focuses on near-real-time RAN optimization and control (from 10 ms to 1 second) through *xApps*.

O-RAN also introduces three critical interfaces: the A1 interface, enabling non-RT RIC to manage policies and ML models for long-term optimization; the O1 interface, which allows the Service Management and Orchestration (SMO) framework to oversee Fault, Configuration, Accounting, Performance and Security (FCAPS) tasks for the vBS and near-RT-RIC; and the E2 interface, designed for near-RT-RIC to gather performance metrics from vBS components and make rapid adjustments to resource allocation and service quality. For more details on O-RAN, refer to [4].

However, despite these advances, several challenges persist in fully realizing the potential of O-RAN, particularly in the context of uRLLC services. The inherent architectural design of O-RAN imposes limitations on making ultra-low latency decisions, such as packet scheduling for uRLLC, which demands sub-millisecond response times [5]. Moreover, limited access to fine-grained, low-level information via the E2 interface (e.g., transmission queue states, real-time channel quality) and the communication latencies involved in gathering this information further compound the difficulty of achieving real-time optimization within the required timescales [6].

Motivation. This calls for an innovative RT control loop to monitor and orchestrate Medium Access Control (MAC) schedulers, especially in scenarios where multiple vBSs are deployed. In such cases, each vBS may operate with a dedicated DU featuring customized network functionalities. Building on the ongoing study of RT control loop mechanisms [6], *this paper establishes a foundation for future research into a comprehensive RT orchestration framework.*

One of the key challenges in O-RAN-compliant networks is ensuring that each uRLLC service meets its stringent delay requirement throughout its lifecycle. *This can be seen as an optimization problem, where the objective is to minimize for each service the probability the packet transmission delay w exceeds a given bound W , i.e., $\min \{\mathbb{P}[w > W]\}$, while ensuring that it remains below a predefined threshold ε , i.e., $\mathbb{P}[w > W] \leq \varepsilon$.* To achieve this, it is necessary to determine the optimal allocation of radio resources for each uRLLC

This work is part of grant PID2022-137329OB-C43 funded by MICIU/AEI/10.13039/501100011033. It has also been financially supported by the Ministry for Digital Transformation and of Civil Service of the Spanish Government through TSI-063000-2021-28 (6G-CHRONOS) project, and by the European Union through the Recovery, Transformation and Resilience Plan - NextGenerationEU, and in part by SNS JU Project 6G-GOALS (GA no. 101139232).

Oscar Adamuz-Hinojosa is with the Department of Signal Theory, Telematics and Communications, University of Granada, Granada, Spain (e-mail: oad-amuz@ugr.es). Lanfranco Zanzi, Vincenzo Sciancalepore, and Xavier Costa-Pérez are with NEC Laboratories Europe, Heidelberg, Germany. (e-mail: {name.surname}@neclab.eu). Xavier Costa-Pérez is also with i2CAT Foundation and ICREA, Barcelona, Spain (e-mail: {name.surname}@i2cat.net).

service at the near-RT scale. Additionally, the RT control loop must dynamically adjust the radio resource allocation computed in near-RT scale to prevent violations of these delay constraints due to unexpected traffic variations. Therefore, the effective coordination of near-RT and RT control loops is essential for optimizing radio resource allocation and maintaining the required performance for coexisting uRLLC services [4].

To ensure a packet's transmission delay w stays within a bound W with a violation probability ε , i.e., $\mathbb{P}[w > W] < \varepsilon$ it is crucial to estimate $\mathbb{P}[w > W]$ accurately. Stochastic Network Calculus (SNC) is a powerful tool that allows to calculate this probability under various traffic and cell capacity conditions. Previous work [7]–[9] applied SNC to estimate delay bounds for specific radio resource allocations in individual uRLLC services. In [10], we extended SNC for multi-service resource planning, though this approach, which dedicated Resource Blocks (RBs) per service, risked inefficient resource overprovisioning. Our latest work introduced an O-RAN-compliant framework [11] that builds on SNC, optimizing radio resource allocation across multiple uRLLC services using historical traffic and capacity data. Unlike earlier methods, we integrate O-RAN control loops, including a near-RT SNC-based controller that dynamically determines guaranteed RBs for each service to maintain violation probabilities within target limits. Additionally, we propose a RT control loop that adjusts RBs in response to transmission queue data, addressing traffic anomalies and further reducing delay violation probabilities while improving resource efficiency. For further details, we refer the reader to Section VII.

Contributions. While SNC-based models provide clear advantages in terms of latency guarantee, their conservative delay bound estimations may limit the number of deployable services [12]. To address this point, in this work, we explore the Martingales queueing methodology, which has shown notable improvements in delay bound estimation [13], [14]. Specifically, we propose a framework that builds upon our previous research [11], where we used an SNC-based approach, and extend it by incorporating a model based on Martingales Theory. Several research works, such as [15]–[18], have proposed radio resource allocation solutions using Martingales-Theory-based models for estimating delay bounds. However, many of these solutions depend on well-known statistical distributions, e.g., Poisson, Bernoulli, Markov on-off, which hardly match real traffic patterns that may be the realization of arbitrary distributions. Conversely, this work sheds the light on adapting a Martingales-Theory-based model to arbitrary traffic conditions. Specifically, we focus on the downlink (DL) operation of a single cell supporting multiple uRLLC services, each with specific packet delay budget and violation probability requirements. Nevertheless, our solution is adaptable to broader scenarios involving Uplink (UL) transmissions and multiple cells. The main contributions can be summarized as follows:

(C1) Building on the framework and findings of [11], we introduce MAREA as a complementary approach to support near-RT and RT control loops that can adapt to arbitrary traffic and channel statistical distributions.

MAREA incorporates a novel Martingales-based controller, replacing the original SNC-based controller in the near-RT control loop. This enhancement improves the allocation of guaranteed RBs for each uRLLC service while maintaining the probability of delay violation below a given target.

(C2) We provide a detailed description of the functional and building blocks that constitute the MAREA framework, focusing on its integration with the O-RAN architecture and operations in both near-RT and RT control loops. We also provide valuable insights related to MAREA's capabilities and interaction with the O-RAN ecosystem.

(C3) We provide a comprehensive evaluation of MAREA by means of an exhaustive simulation campaign involving realistic 5G-New Radio (NR) scenarios, comparing the Martingales-based approach against the previous proposed SNC-based model. Our results show that MAREA effectively achieves more accurate delay-bound estimations, which translates into the possibility to safely support a larger number of uRLLC services for the given cell configuration. Our evaluation also demonstrates improved execution times, making MAREA better suited for real-world applications.

The remainder of this paper is organized as follows. Section II defines MAREA. Section III introduces the proposed Martingales-based model. In Section IV, we describe how MAREA executes the multi-time-scale control loops. Section V evaluates MAREA's performance. Section VI discusses key challenges for implementing MAREA in commercial O-RAN deployments. Section VII discusses related works. Finally, Section VIII concludes the paper.

II. THE MAREA FRAMEWORK

In this paper, we consider a set \mathcal{M} of vBSs belonging to the same Mobile Network Operator (MNO) and deployed over the same cell. Each vBS is tailored to meet the performance requirements of a specific uRLLC service. The implementation of these vBSs is illustrated in Fig. 1, featuring both dedicated and shared components. Specifically, the CU-CP and the RU are shared among all vBSs [19], whereas the CU-UP and the DU are dedicated to each vBS [20].

The possibility of sharing CU-CP in O-RAN is crucial as to enable the centralized execution of Radio Resource Management (RRM) and Radio Resource Control (RRC) tasks across uRLLC services, preventing conflicting decisions and enhancing overall performances and scalability. Focusing on the RB allocation, we consider two hierarchical allocation levels. At the higher level, RBs are distributed centrally among different vBS-DUs, ensuring coordinated resource assignment across services. At the lower level, each vBS-DU assigns its allocated RBs to its attached User Equipments (UEs) at MAC level. Given the necessity of RT orchestration decisions for the high-level allocation, this process must occur close to the vBS-DUs. Specifically, we assume the high-level allocation is executed within the CU-CP, which ensures that RBs are assigned in a coordinated manner to all running services, improving individual service performance and preventing adverse impacts.

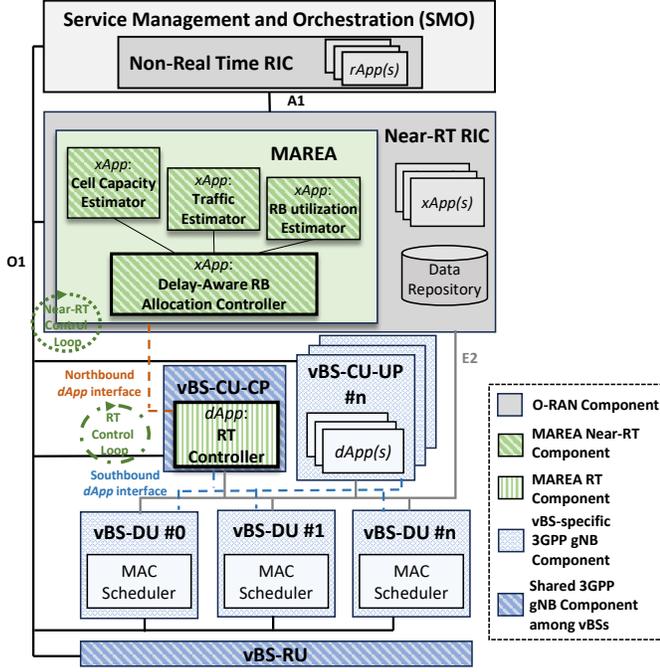


Fig. 1. Integration of MAREA (i.e., green blocks) in the O-RAN architecture. For simplicity, only the northbound and southbound interfaces of the RT Controller $dApp$ are shown. Other $dApps$ would have similar interfaces.

Under this scenario, we address the dynamic RB allocation problem among multiple uRLLC services, each with specific performance requirements regarding packet delay budget and violation probability. We focus on the operation of both the near-RT control loop and a new RT control loop. To achieve this, we develop a novel framework, namely MAREA, which leverages the O-RAN architecture [21], [22] and operates at different time scales, as illustrated in Fig. 1. Next, we explain how MAREA operates at near-RT and RT scales, followed by details of the MAREA's components.

Fig. 2 showcases near-RT and RT control loops. At the near-RT scale, MAREA runs periodically every T_{OUT} Transmission Time Intervals (TTIs)(see point A). During each execution period, MAREA computes the amount of guaranteed RBs for each uRLLC service related to the following execution period (see point B). This calculation requires performance metrics from the previous T_{OBS} TTIs (see point C). At RT scale, within a specific TTI, some services may not fully utilize their guaranteed RBs, while others may require additional RBs beyond what is guaranteed. To optimize the use of available RBs in the cell, MAREA implements a RT control loop to redistribute previously assigned RBs to services that need them (see point D). Furthermore, if traffic anomalies are detected, the RT control loop may temporarily adjust the amount of guaranteed RBs to meet service requirements (see point E).

To perform near-RT and RT control loops, MAREA comprises four $xApps$ and one $dApp$ ¹ as shown in Fig. 1. The Cell Capacity Estimator, Traffic Estimator, RB utilization Estimator and Delay-Aware RB Allocation Controller $xApps$ are located in the near-RT RIC and are responsible for computing the

¹The concept of $dApps$, introduced in [6], is similar to $rApps$ and $xApps$, and enables fine-grained real-time control tasks. These $dApps$ can be deployed within the CU-CP, CU-UP, and/or DU.

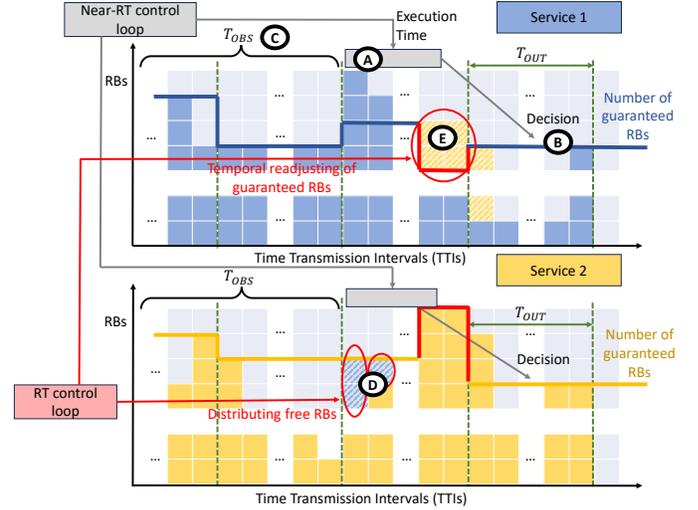


Fig. 2. Illustrative example of how MAREA performs dynamic RB allocation at near-RT and RT scales.

guaranteed RBs for multiple uRLLC services in a near-RT scale. The RT Controller $dApp$ is located in a CU-CP and it is responsible for fine-tuning RB allocation at RT scale. Details about these $Apps$ are provided below.

Traffic Estimator $xApp$: It analyzes the traffic generated by each service $m \in \mathcal{M}$ via the E2 interface over the last T_{OBS} TTIs, specifically tracking the number of bits generated by each service in each TTI. Using this data, this $xApp$ estimates the Probability Mass Function (PMF) of traffic generation for each service. Further details are provided in in Section III-C.

The O-RAN E2 Service Model (E2SM) [23] defines a metric to capture the number of bits entering the Radio Link Control (RLC) layer in the downlink for a single Dedicated Radio Bearer (DRB). This metric, measured in Kbits, can be filtered per network slice (and thus, per service) and measured periodically over a period referred to as *granularity period*. This period can be as short as 1 ms. The E2 interface does not need to transmit measurements of this metric every TTI, but can instead send them in bursts. The corresponding *reporting period* should be set such that the measurements arrive within the execution period of MAREA (see point A, Fig. 2) and allow it to be executed on time. Additionally, for the proposed Traffic Estimator $xApp$, assuming each measurement requires N_{bit} bits to represent the value of such metric and there is one measurement per service $m \in \mathcal{M}$ over T_{OBS} TTIs, the proposed $xApp$ must allocate a buffer of size $N_{bit} \times |\mathcal{M}| \times T_{OBS}$ to store the metrics captured via the E2 interface.

RB utilization Estimator $xApp$: Assuming a specific amount of guaranteed RBs for a service $m \in \mathcal{M}$, it estimates the PMF for the number of RBs that the service m uses beyond the guaranteed amount in an arbitrary TTI. To achieve this, it utilizes a Neural Network (NN) based on Mixture Density Network (MDN), considering the following inputs: (a) the incoming traffic demand in each TTI expressed in bits, (b) the enqueued bits in each TTI and (c) the candidate number of guaranteed RBs for the next T_{OUT} TTIs. These metrics are available from E2 interface. Details about the MDN implementation are provided in Section IV-B.

The O-RAN Near-Real-time RAN Intelligent Controller E2

Service Model (E2SM) specification [24] defines the concrete metrics to obtain inputs (b) and (c) through the E2 interface. Note that input (a) was previously explained in the context of the Traffic Estimator $xApp$. Input (b), the enqueued bits in each TTI, can be obtained from the metric *DL Buffer Occupancy* at the RLC layer, which is measured in Kbytes. Input (c), the candidate number of guaranteed RBs for the next T_{OUT} TTIs, can be obtained from the parameter *Slice PRB Quota*, which specifies, among other metrics, the number of guaranteed RBs allocated to each slice, and thus per service.

Cell Capacity Estimator $xApp$: It analyzes the packets transmitted via the radio interface as well as the Modulation and Coding Schemes (MCSs) used for these transmissions for each service $m \in \mathcal{M}$ during the last T_{OBS} TTIs. This data is collected from the vBS-DUs via E2 interface, as defined in the O-RAN specifications [24]. The MCSs are computed by the vBS-DUs using the UE's Channel State Information (CSI) reporting, which includes the Channel Quality Indicator (CQI) [25]. These measurements are aligned with the metrics specified in 3GPP TS 28.552 (Section 5.1.1.12) [26]. By considering these measurements, along with a guaranteed amount of RB and the PMF of potential RBs usage exceeding this guaranteed amount (provided by the RB utilization Estimator $xApp$), this $xApp$ generates the PMF for the served traffic. Specifically, it calculates the PMF of the number of bits served per TTI for each service $m \in \mathcal{M}$. Details on computing these PMFs are provided in Section IV-A.

Delay-Aware RB Allocation Controller $xApp$: Using inputs from previous $xApps$ and an iterative process, this $xApp$ calculates the amount of guaranteed RBs N_m^{min} for each service $m \in \mathcal{M}$, which ensures $\mathbb{P}[w > W_m] < \varepsilon_m$, where w is the packet transmission delay², W_m is a delay bound and ε_m is a violation probability. Note that $\sum_{m \in \mathcal{M}} N_m^{min} \leq N_{cell}^{RB}$, where N_{cell}^{RB} is the RBs available in the cell.

In our previous work [11] we make use of an SNC-based model to derive this probability. While this approach ensures conservative estimations of the delay probabilities, it may limit the number of services accommodated in the cell. To address this drawback, in this paper, we consider a complementary Martingales-based approach that achieves more accurate delay estimations. However, unlike the SNC-based model, such estimations are no longer conservative. This translates into the possibility of admitting more services given a more efficient resource allocation N_m^{min} for each service m , at the cost of a relaxed delay probability characterized by a small margin of error. We envision an interested MNO selecting the most appropriate model—whether the SNC-based or Martingales-based—depending on its business requirements. Section III describes the Martingales-based model, while the implementation of this $xApp$ is detailed in Section IV-C. A comparison of performances of using both the SNC-based model and the Martingales-based model is evaluated in Section V.

RT Controller $dApp$: It operates every TTI and ensures that each DU MAC scheduler, one per service $m \in \mathcal{M}$, has available at least N_m^{min} RBs, i.e., the amount computed

by the Delay-Aware RB Allocation Controller $xApp$. For a given TTI t , if a service m requires N_t RBs such as $N_t < N_m^{min}$, this $dApp$ allocates N_t RBs to such service. Otherwise, it first checks for free RBs, i.e., those allocated to other services but unused in the current TTI. If free RBs are found, the $dApp$ allocates N_m^{min} plus the available free RBs to the specified service. Additionally, this $dApp$ monitors the transmission buffers for traffic anomalies and, if detected, temporarily updates $N_m^{min} \forall m \in \mathcal{M}$ to mitigate the violation probability. To effectively monitor metrics at RT scale, vBS components and $dApps$ need interfaces similar to those used by RICs and vBS components (i.e., O1 and E2 interfaces). Northbound interfaces between $dApps$ and the near-RT RIC, and southbound interfaces between $dApps$ and programmable functions of DUs/CU as proposed in [6], facilitate control and data sharing, ensuring $dApps$ are platform-independent and interact seamlessly with other O-RAN components.

The O-RAN Alliance has not yet released specific guidelines detailing how these northbound and southbound interfaces should be implemented. Nevertheless, progress is being made in this direction, as evidenced by a recent O-RAN research report [27]. This report outlines the minimum architectural and interface requirements for $dApps$, including data exchange timelines, and explores their impact on control and user plane extensions.

MAREA in a Multi-Cell Scenario: To extend the proposed MAREA framework to a multi-cell scenario, it is sufficient to deploy our solution in a distributed manner. Specifically, one instance of MAREA can be deployed in each network cell. Each instance works independently using local performance metrics from its own cell, allowing the framework to scale efficiently as more cells are added. This setup ensures that our solution remains effective and scalable without needing major changes to its design or operation.

III. LATENCY MODEL BASED ON MARTINGALES

To assess the packet transmission delay for each service, we propose a delay model based on Martingales. This section first provides some fundamentals on Martingales. Then, we describe the steps to compute the delay bound $W_m \forall m \in \mathcal{M}$. Finally, we particularize these steps to our scenario.

A. Fundamentals on Martingales

Definition 1 (Martingale) [18]: Let $(\Omega, \mathcal{F}_\infty, P)$ be a probability space and let $\{\mathcal{F}_t, t \geq 0\}$ be a filtration, which is a nondecreasing sequence of σ -fields of \mathcal{F}_∞ , $\mathcal{F}_t \subset \mathcal{F}_{t+1}, t \geq 0$. A stochastic process $\{X(t), t \geq 0\}$ is adapted to the filtration $\{\mathcal{F}_t, t \geq 0\}$ if $X(t)$ is \mathcal{F}_t -measurable for all $t \geq 0$. If $\mathbb{E}[|X(t)|] < \infty \forall t \geq 0$ and $\mathbb{E}[X(t+1)|\mathcal{F}_t] = X(t) \forall t \geq 0$, then $\{X(t), t \geq 0\}$ is called a martingale. Moreover, if $\mathbb{E}[X(t+1)|\mathcal{F}_t] \geq X(t)$, then $\{X(t), t \geq 0\}$ is called a submartingale. On the contrary, if $\mathbb{E}[X(t+1) | \mathcal{F}_t] < X(t)$, then $\{X(t), t \geq 0\}$ is called a supermartingale.

Let $A(i, j) = \sum_{k=i+1}^j a_k$ denote the arrival process, representing the cumulative number of bits arriving at a network node over the time interval $(i, j]$, where i and j represent the TTI indexes. Furthermore, $S(i, j) = \sum_{k=i+1}^j s_k$ denotes

²Packet transmission delay is the waiting time of a Transport Block (TB) unit from entering the transmission buffer until it is fully transmitted.

the service process, representing the cumulative number of bits that this network node can process in the same interval. Both processes are driven by the stochastic processes a_k and s_k , representing the number of incoming and served bits in an arbitrary TTI, respectively. Henceforth, we define $A(j) := A(0, j)$ and $S(j) := S(0, j)$ to simplify notation.

Definition 2 (Arrival Martingale) [14]: The arrival process $A(j)$ admits an arrival martingale if for a free parameter $\theta > 0$ there is a $K_a \geq 0$ and a function $h_a : \text{rng}(a) \rightarrow \mathbb{R}^+$ such that the following process is a supermartingale

$$h_a(a_j) \exp[\theta(A(j) - jK_a)], \quad j \geq 0. \quad (1)$$

where ‘rng’ represents the range operator. The parameters K_a and h_a implicitly depend on θ ; for brevity, the notation $K_a(\theta)$ and $h_a(\theta)$ is omitted.

Definition 3: (Service Martingale) [14]: The service process $S(j)$ admits a service martingale if for every $\theta > 0$ there is a $K_s \geq 0$ and a function $h_s : \text{rng}(s) \rightarrow \mathbb{R}^+$ such that the next process is a supermartingale

$$h_s(s_j) \exp[\theta(jK_s - S(j))], \quad j \geq 0. \quad (2)$$

Assuming the arrival process $A(j)$ and the service process $S(j)$ admit arrival and service martingales, respectively, the probability the packet’s transmission delay w exceeds a delay bound W , i.e., the *violation probability*, has been defined by Ciucu et al. [13], [28] as

$$\mathbb{P}[w \geq W] \leq \frac{\mathbb{E}[h_a(a_0)] \mathbb{E}[h_s(s_0)]}{H} \exp[-\theta^* K_s W], \quad (3)$$

where θ^* is computed as the supremum of the following set

$$\theta^* := \sup \{ \theta > 0 : K_a \leq K_s \}, \quad (4)$$

and H is the minimum value of the next set

$$H := \min \{ h_a(x) h_s(y) : x - y > 0 \}. \quad (5)$$

In Eq. (3), the delay bound W is given in terms of the number of TTIs. Based on this, we can reformulate Eq. (3) to estimate the delay bound W as shown in Eq. (6). Note that we have multiplied the resulting expression by the duration of a single TTI, t_{slot} , to convert the resulting delay bound into time units.

$$W \leq -\log \left[\frac{H \mathbb{P}[w \geq W]}{\mathbb{E}[h_a(a_0)] \mathbb{E}[h_s(s_0)]} \right] (\theta^* K_s)^{-1} t_{slot}. \quad (6)$$

B. Methodology for Delay Bound Computation with Martingales

Based on the fundamentals of Martingales discussed earlier, the following steps outline the procedure for estimating the delay bound W given a target violation probability $\mathbb{P}[w \geq W]$.

Step 1: Define the arrival and service martingales for the arrival and service processes $A(j)$ and $S(j)$, respectively. In this paper, we use Wald’s martingales, which are applied to sums of i.i.d. random variables [29].

Definition 4 (Wald’s Martingale) [29]: Let $\{x_j, j > 1\}$ be a sequence of i.i.d. random variables with a finite Moment Generating Function (MGF) $M_x(\theta) = \mathbb{E}[e^{\theta x}]$ for some $\theta > 0$, and let $Y(j) = \sum_{k=1}^j x_k$ with $Y(0) = 0$. The process

$M_Y^W(j) \forall j \geq 0$, defined by $M_Y^W(j) = M_x(\theta)^{-j} e^{\theta Y(j)}$, is Wald’s martingale, satisfying $\mathbb{E}[M_Y^W(j)] = 1 \forall j \geq 0$.

Step 2: Rewrite Wald’s martingales to conform with the arrival and service martingales as specified in Eqs. (1) and (2). From these reformulated expressions, extract the parameters $h_a(a_j)$, K_a , $h_s(s_j)$ and K_s .

Step 3: Using the extracted parameters, compute the threshold H , the expectations $\mathbb{E}[h_a(a_0)]$, $\mathbb{E}[h_s(s_0)]$, and θ^* .

Step 4: Finally, substitute the computed values into Eq. (6) to estimate the delay bound W .

Below, we particularize these steps to estimate the delay bound W_m for a uRLLC service $m \in \mathcal{M}$.

C. Arrival Process of a Downlink uRLLC Service

The arrival process $a_{m,j}$ represents the number of bits that arrive to the vBS’s transmission buffer for service m in the j -th TTI. We assume the PMF of the process $a_{m,j}$ can be estimated by using samples of the incoming bits per TTI in the last T_{OBS} TTIs. To that end, we define the sample vector $\vec{x}_{a_m} = \{a_{m,1}^{in}, a_{m,2}^{in} \dots a_{m,T_{OBS}}^{in}\}$, where $a_{m,i}^{in}$ denotes the number of bits that arrived to the transmission buffer in the TTI i for the service m . Additionally, $a_{m,i}^{in} = \sum_{k=1}^{J_{m,i}^{in}} l_k$, where $J_{m,i}^{in}$ is the number of incoming packets for service m in the TTI i and l_k the size of the packet k . Note that the computation of \vec{x}_{a_m} is a task performed by the Traffic Estimator $xApp$. Under these assumptions, we can compute the MGF for the process $a_{m,j}$ as

$$M_{a_m}(\theta) = (T_{OBS})^{-1} \sum_{i=1}^{T_{OBS}} \exp[\theta a_{m,i}^{in}]. \quad (7)$$

Based on the previous equation, we can build the Wald’s martingale for the arrival process as

$$\begin{aligned} M_{a_m}^W(\theta) &= M_{a_m}(\theta)^{-j} \exp[\theta A(j)] \\ &= \exp[-j \log[M_{a_m}(\theta)]] \exp[\theta A(j)] \\ &= \exp[\theta(A(j) - j \log[M_{a_m}(\theta)])]. \end{aligned} \quad (8)$$

Note that the final expression was obtained by applying the exponential operator to the logarithm of the term $M_{a_m}(\theta)^{-j}$, and then operating on the resulting expression. The obtained Wald’s martingale has the same form of the arrival-martingale defined in Eq. (1). Based on this observation, we can obtain the parameter $K_{a,m}$ as

$$K_{a,m} = \theta^{-1} \log \left[(T_{OBS})^{-1} \sum_{i=1}^{T_{OBS}} \exp[\theta a_{m,i}^{in}] \right]. \quad (9)$$

Note that the parameter $h_{a,m}(a_{m,j}) = 1$. Without loss of generality, we define the auxiliary parameter $K'_{a,m} = \theta K_{a,m}$.

D. Service Process of the Cell Capacity Provided to a Downlink uRLLC Service

The service process $s_{m,j}$ represents the number of bits that may be served by the cell for service $m \in \mathcal{M}$ in the j -th TTI. The negative MGF for $s_{m,j}$ is defined in Eq. (10). To that end, we consider the PMF of $s_{m,j}$ can be estimated by (a) using samples of the number of bits which may be transmitted in the last T_{OBS} TTIs and (b) considering the PMF of the

RB utilization. The latter captures the dynamics of the RT Controller *dApp*. Specifically, we consider the sample vectors $\vec{x}_{s_m,n} = \{s_{m,1}^{n,out}, s_{m,2}^{n,out}, \dots, s_{m,T_{m,n}}^{n,out}\} \forall n \in [0, N_{add}]$ where $s_{m,i}^{n,out}$ denotes the number of bits which may be transmitted by the cell for service m in a single TTI, i.e., considering the serving cell uses $n + N_m^{min}$ RBs for such service. Note that n represents the number of additional RBs allocated to service m , i.e., beyond the guaranteed ones. Furthermore, $N_{add} = N_{cell}^{RB} - N_m^{min}$ and $T_{m,n}$ is the number of samples. Additionally, we consider $\pi_{m,n}$ as the probability that the service m has a number of available RBs $n + N_m^{min}$ in an arbitrary TTI, conditioned to the fact that the service m requires n RBs more than N_m^{min} . The computation of $\vec{x}_{s_m,n}$, a task performed by the Cell Capacity Estimator *xApp*, is detailed in Section IV-A.

$$M_{s_m}(-\theta) = \sum_{n=0}^{N_{add}} \frac{\pi_{m,n}}{T_{m,n}} \sum_{i=1}^{T_{m,n}} \exp[-\theta s_{m,i}^{n,out}]. \quad (10)$$

Based on the previous equation, we can derive the Wald's martingale for the service process as shown in Eq. (11). Note that the final expression was obtained by applying the exponential operator to the logarithm of the term $M_{s_m}(-\theta)^{-j}$.

$$\begin{aligned} M_{s_m}^W(-\theta) &= M_{s_m}(-\theta)^{-j} \exp[-\theta S(j)] \\ &= \exp[-j \log[M_{s_m}(-\theta)]] \exp[-\theta S(j)] \\ &= \exp[\theta(-j\theta^{-1} \log[M_{s_m}(-\theta)] - S(j))]. \end{aligned} \quad (11)$$

The resulting Wald's martingale has the same form as the service martingale defined in Eq. (2). Based on that, we can obtain the parameter $K_{s,m}$ as Eq. (12) shown. Note that the parameter $h_{s,m}(s_{m,j}) = 1$. For convenience, we define the auxiliary parameter $K'_{s,m} = \theta K_{s,m}$.

$$K_{s,m} = -(\theta^{-1}) \log \left[\sum_{n=0}^{N_{add}} \frac{\pi_{m,n}}{T_{m,n}} \sum_{i=1}^{T_{m,n}} \exp[-\theta s_{m,i}^{n,out}] \right] \quad (12)$$

E. Delay Bound Estimation for an uRLLC Service

Since $h_{a,m}(a_{m,j}) = 1$ and $h_{s,m}(s_{m,j}) = 1$, the threshold $H_m = 1$ as well as the expectations $\mathbb{E}[h_{a,m}(a_{m,0})] = 1$ and $\mathbb{E}[h_{s,m}(s_{m,0})] = 1$. Using these results along with Eq. (6) and Eq. (12), we can define $W_m \forall m \in \mathcal{M}$ as follows

$$W_m \approx \frac{\log[\mathbb{P}[w \geq W_m]]}{\log \left[\sum_{n=0}^{N_{add}} \frac{\pi_{m,n}}{T_{m,n}} \sum_{i=1}^{T_{m,n}} \exp[-\theta_m^* s_{m,i}^{n,out}] \right]} \quad (13)$$

The previous delay bound depends on θ_m^* , which is the minimum upper bound for the set of values of θ where $K_{s,m} \geq K_{a,m}$ as Eq. (4) shown. Note this set of values is the same when considering the inequality $K'_{s,m} \geq K'_{a,m}$.

Before computing θ_m^* , we first present an example in Fig. 3, illustrating the evolution of the parameters $K'_{a,m}$ and $K'_{s,m}$ for various θ values. This example consider different scenarios, each with a specific number of guaranteed RBs for a single service $m \in \mathcal{M}$. If we examine the right plot, i.e., $N_m^{min} = 90$ RBs, $K'_{s,m}$ is greater than $K'_{a,m}$ for very small θ values. The range where $K'_{s,m} > K'_{a,m}$ decreases as the number of guaranteed RBs decreases, e.g., when $N_m^{min} = 60$

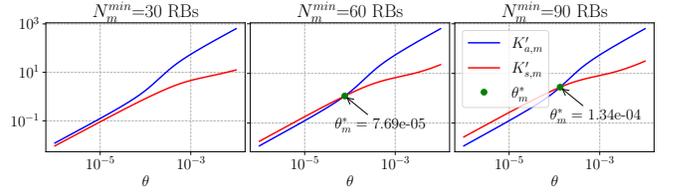


Fig. 3. Evaluation of $K'_{a,m}$ and $K'_{s,m}$ for a service $m \in \mathcal{M}$ with a specific number of guaranteed RBs N_m^{min} . This example uses the traffic pattern and channel conditions defined in the experimental setup (see Section V).

Algorithm 1: Search of θ_m^* for a service $m \in \mathcal{M}$.

```

1 Inputs: Sample vectors  $\vec{x}_{a,m}$  and  $\vec{x}_{s_m,n}$ ,  $T_{OBS}$ ,  $T_{m,n}$ ,  $\pi_{m,n}$ ;
2 Initialization:  $\theta_{old} = 1$ ,  $\Delta = 0.9$ ;
3 while True do
4   Compute  $\theta_{new} = \theta_{old}\Delta$ ;
5   if  $K'_{s,m}(\theta_{new}) - K'_{a,m}(\theta_{new}) \geq 0$  then
6     break;
7   else
8      $\theta_{old} = \theta_{new}$ ;
9     if  $\theta_{new} < 10^{-9}$  then
10      break;
11    end
12  end
13 end
14 if  $\theta_{new} \geq 10^{-9}$  then
15   Use bisection algorithm [30] to find  $\theta_m^* \in [\theta_{new}, \theta_{old}]$ ;
16   Output:  $\theta_m^*$ ;
17 else
18   Output:  $\theta_m^*$  does not exist;
19 end

```

RBs. Furthermore, when the reserved RBs are sufficiently small, e.g., $N_m^{min} = 30$, $K'_{a,m}$ never exceed $K'_{s,m}$ for any θ , indicating that the DL traffic for the service cannot be adequately served over time, resulting in overflow in the vBS's transmission buffer.

Another observation is $K'_{a,m}$ and $K'_{s,m}$ are not convex, indicating the need for a heuristic algorithm to search for θ_m^* . Based on the observed behavior of $K'_{a,m}$ and $K'_{s,m}$, we have proposed the heuristics defined in Algorithm 1. It begins by taking as input the sample vectors $\vec{x}_{a,m}$, $\vec{x}_{s_m,n}$ containing the traffic estimation for service m and the cell capacity for that service, respectively, as well as the sample sizes T_{OBS} , $T_{m,n}$ and the probability $\pi_{m,n}$ (line 1). With these inputs, an initial value is set for the upper limit of the interval where $\theta_m^* \in [\theta_{new}, \theta_{old}]$, and a parameter Δ is defined to determine the speed of the search for θ_m^* , i.e., the closer Δ is to 1, the more exhaustive the search (line 2). Based on experimental observations in Fig. 3, we set $\theta_{old} = 1$ and $\Delta = 0.9$. Then, we start an iterative procedure (lines 3-13). In each iteration, a value for the lower limit of the interval $[\theta_{new}, \theta_{old}]$ is computed (line 4). Then, we evaluate if $K'_{s,m}(\theta_{new}) - K'_{a,m}(\theta_{new}) > 0$. If so, the loop terminates with θ_{new} as the new lower bound (line 6). Otherwise θ_{old} is updated to θ_{new} and the loop continues (line 8). Note that if θ_{new} remains above 10^{-9} (line 10), the loop ends and the algorithm stop (lines 18). The value 10^{-9} has been experimentally established, based on Fig. 3, as sufficiently small to determine that θ_m^* does not exist due to the cell capacity being inadequate to meet the service demands. If θ_{new}

is above 10^{-9} , the bisection method [30], which is based on the intermediate value theorem, is used for searching the value of $\theta_m^* \in [\theta_{new}, \theta_{old}]$ (lines 15-16).

The computational complexity of Algorithm 1 is $\mathcal{O}([T_{m,n} \cdot N_{add} + T_{OBS}] \cdot N_{max})$, where $T_{m,n}$, N_{add} and T_{OBS} are input parameters that determine the number of operations performed by the algorithm. In each iteration, the algorithm computes two variables: $K'_{s,m}$ and $K'_{a,m}$. To calculate $K'_{s,m}$, i.e., Eq. (12), a double summation is performed over the indices $T_{m,n}$ and N_{add} , with each summation involving the calculation of an exponential function, which is assumed to take constant time. This results in a complexity of $\mathcal{O}(T_{m,n} \cdot N_{add})$. For $K'_{a,m}$, i.e., Eq. (9), a summation over T_{OBS} is performed, also involving the calculation of an exponential function in each term, yielding a complexity of $\mathcal{O}(T_{OBS})$. Since these calculations are independent, their complexities are added, giving $\mathcal{O}(T_{m,n} \cdot N_{add} + T_{OBS})$ per iteration. The algorithm includes a while loop, whose maximum number of iterations is bounded by $N_{max} = \left\lceil \frac{\log(10^{-9})}{\log(\Delta)} \right\rceil$, where Δ is a parameter that controls convergence. Therefore, the total complexity of Algorithm 1 is $\mathcal{O}([T_{m,n} \cdot N_{add} + T_{OBS}] \cdot N_{max})$, exhibiting a scalable (and computationally affordable) behavior.

IV. CONTROL LOOPS OF MAREA

In this section, we detail how MAREA executes the near-RT and RT control loops. We begin by explaining the computation of cell capacity samples and the estimation of probabilities $\pi_{m,n}$. Next, we describe how the Delay-Aware RB Allocation Controller $xApp$ determines N_m^{min} for all $m \in \mathcal{M}$ in the near-RT control loop. Finally, we outline how the RT Controller $dApp$ mitigates the violation probability.

A. Computation of the Cell Capacity Samples

The Cell Capacity Estimator $xApp$ performs the following steps to obtain a sample $s_{m,i}^{n,out} \in \vec{x}_{s_{m,n}}$.

Step 1: For each transmitted packet j , it considers (a) the packet size l_j , and (b) the amount of RBs required to transmit it, i.e., $N_{pkt,j}$. Note that we are assuming that a unique MCS value is adopted to transmit each packet. It means this $xApp$ can compute the number of bits transmitted per RB as $s_{RB,j} = l_j / N_{pkt,j}$. Based on that, it defines a vector $\vec{x}_{pkt,j} = \{s_{RB,j}, s_{RB,j} \dots, s_{RB,j}\}$, where the element $s_{RB,j}$ is repeated $N_{pkt,j}$ times.

Step 2: Performing the previous step for all the transmitted packets, this $xApp$ obtains a set of vectors $\vec{x}_{pkt,j} \forall j \in \mathcal{J}_{OBS}$, where \mathcal{J}_{OBS} denotes the set of packets transmitted in the last T_{OBS} TTIs. Based on that, this $xApp$ defines $\vec{x}_{con} = \{\vec{x}_{pkt,1}, \vec{x}_{pkt,2} \dots \vec{x}_{pkt,|\mathcal{J}_{OBS}|}\}$ as a vector which concatenates each measured vector $\vec{x}_{pkt,j}$.

Step 3: Based on \vec{x}_{con} , this $xApp$ groups its samples in set of $N_{m,n}^{set} = n + N_m^{min}$ consecutive samples. Note that $n \in [0, N_{add}]$ is the number of additional RBs that may be allocated to service m beyond the guaranteed number of RBs (see definition in Section III-D). In turn, each group of $N_{m,n}^{set}$ consecutive samples defines a vector $\vec{x}_{m,i}^{n,out}$, where $i \in [1, T_{m,n}]$ represents the i -th vector. We define $T_{m,n}$ as the

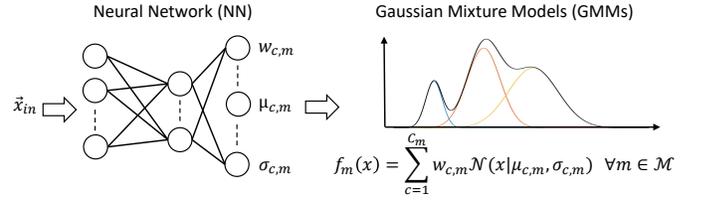


Fig. 4. Overview of the considered Mixture Density Network (MDN). Copied from [11].

total number of built vectors. Note we have one vector $\vec{x}_{m,i}^{n,out}$ per sample $s_{m,i}^{n,out}$, i.e., see Eq. (10).

Step 4: Finally, this $xApp$ obtains the sample $s_{m,i}^{n,out}$ as the sum of all the elements of the vector $\vec{x}_{m,i}^{n,out}$, i.e., $s_{m,i}^{n,out} = \sum_{z=1}^{N_{m,n}^{set}} \vec{x}_{m,i}^{n,out} \{z\}$.

Note that this $xApp$ generates a sample vector $\vec{x}_{s_{m,n}}$ for each value of n . Thus, steps 3-4 need to be repeated for every n . For instance, if a service m has $N_m^{min} = 10$ guaranteed RBs and the cell has $N_{cell}^{RB} = 25$, then n varies from 0 to 15.

B. Mixture Density Networks for Estimating the RB Utilization

The Delay-Aware RB Allocation Controller $xApp$ utilizes a MDN, a NN architecture for modeling probability distributions [31], to estimate $\pi_{m,n}$. Unlike a typical NN, which produces a single value, an MDN generates one or more parameterized mixture models in its output layer. These models are weighted combination of several component distributions. We consider $|\mathcal{M}|$ Gaussian Mixture Models (GMMs), one per service. It is proven the GMM accurately approximate any arbitrary distribution in the context of wireless networks [32]–[34]. Specifically, $\pi_{m,n}$ might not follow a single known statistical distribution and may vary over time. The GMM is described by the equation in Fig. 4, where C_m denotes the number of Gaussian distributions. In turn, the c -th distribution is characterized by the weight $w_{c,m}$, the mean $\mu_{c,m}$ and the standard deviation $\sigma_{c,m}$. Note that $\sum_{c=1}^{C_m} w_{c,m} = 1$. The considered MDN is summarized in Fig. 4. The inputs parameters \vec{x}_{in} are: (a) the RB utilization for each service, and (b) the 25th, 50th and 75th percentiles of incoming and enqueued bits. All of them are measured in the last T_{OUT} TTIs for each service $m \in \mathcal{M}$. Additionally, the MDN considers as input the target number of guaranteed RBs for each service $m \in \mathcal{M}$ in the next T_{OBS} TTIs. Based on them, the MDN provides an estimation of the parameters $w_{c,m}$, $\mu_{c,m}$ and $\sigma_{c,m}$. Finally, we compute the $\pi_{m,n}$ as Eq. 14 shows. Specifically, we split the GMM into N_{add} regions (i.e., see Section III-D) and compute $\pi_{m,n}$ as the probability of being in the n -th region.

$$\pi_{m,n} = \sum_{c=1}^{C_m} \int_{n-0.5}^{n+0.5} w_{c,m} \mathcal{N}(x | \mu_{c,m}, \sigma_{c,m}) dx. \quad (14)$$

Note that the considered MDN model was validated in our previous work [11].

C. MAREA Near-RT Control Loop

The Delay-Aware RB Allocation Controller $xApp$ aims to determine the optimal allocation of guaranteed RBs N_m^{min} for

each service $m \in \mathcal{M}$, such that the delay bound W_m is as close as possible to, or below, the target delay budget W_m^{th} , given the target violation probability ε_m . To achieve this, we formulate the ORCHESTRATION_URLLC_SERVICES problem, aiming to minimize the maximum ratio W_m/W_m^{th} across all $|\mathcal{M}|$ services as shown in Eq. (15). This optimization is subject to the constraint Eq. (16), ensuring the guaranteed RBs across all services does not exceed the available amount given the cell configuration.

Problem ORCHESTRATION_URLLC_SERVICES:

$$\min_{N_m^{min}} g(\vec{W}) = \max \left\{ \frac{W_1}{W_1^{th}}, \dots, \frac{W_{|\mathcal{M}|}}{W_{|\mathcal{M}|}^{th}} \right\}, \quad (15)$$

$$\text{s.t.} \quad \sum_{m=1}^{|\mathcal{M}|} N_m^{min} \leq N_{cell}^{RB}. \quad (16)$$

The objective function $g(\vec{W})$ depends on the computation of $W_m \forall m \in \mathcal{M}$. Specifically, for a given number of guaranteed RBs $N_m^{min} \forall m \in \mathcal{M}$, it is necessary to calculate $|\mathcal{M}|$ delay bounds W_m . This requires calculating θ_m^* for each service beforehand. As demonstrated in Fig. 3 (Section III-E), the search for θ_m^* occurs in a non-convex space, which necessitates using the heuristics from Algorithm 1 $|\mathcal{M}|$ times, one per service. For this reason, we propose Algorithm 2 to solve the ORCHESTRATION_URLLC_SERVICES problem.

This algorithm considers the target delay budget W_m^{th} , the target violation probability ε_m and the sample vectors $\vec{x}_{a,m}$, $\vec{x}_{s,m,n}$ as inputs. It begins with an equal distribution of the available RBs among the services, i.e., $N_{m,z}^{min} = \lfloor N_{cell}^{RB}/|\mathcal{M}| \rfloor$. Then, it initiates an iterative procedure to get $N_m^{min} \forall m \in \mathcal{M}$. In each iteration, considering $N_{m,z}^{min}$ guaranteed RBs for each service, it first estimates $\pi_{m,n}$ using the MDN described in Section IV-B (step 4). Then, it uses the Martingales-based model (see Section III-E) to estimate the delay bound $W_{m,z}$ for the target $N_{m,z}^{min}$ (step 5). Then, it evaluates the objective function, i.e., Eq. (15), considering $W_{m,z}$ and W_m (step 6). If the objective function has been reduced compared to the previous iteration (step 7), the algorithm updates the new values for N_m^{min} and W_m (step 8) and tries to reduce

Algorithm 2: Near-RT RB allocation

```

1 Inputs:  $W_m^{th}$ ,  $\varepsilon_m$ ,  $\vec{x}_{a,m}$ ,  $\vec{x}_{s,m,n}$ ;
2 Initialization:  $N_{m,z}^{min} = \lfloor N_{cell}^{RB}/|\mathcal{M}| \rfloor$ ,  $W_m = \infty$ ,
    $W_{m,z} = \infty$  stop = False;
3 while stop == False do
4   Estimate  $\pi_{m,n} \forall m \in \mathcal{M} \forall n \in [0, N_{add}]$  [Section IV-B];
5   Estimate  $W_{m,z} \forall m \in \mathcal{M}$  [See Eq. (13)];
6   Evaluate  $g(\vec{W}_z)$  and  $g(\vec{W})$  [See Eq. (15)];
7   if  $g(\vec{W}_z) < g(\vec{W})$  then
8     Update  $N_m^{min} = N_{m,z}^{min}$ ;  $W_m = W_{m,z} \forall m \in \mathcal{M}$ ;
9     Select  $m' | \frac{W_{m'}}{W_{m'}^{th}} \geq \frac{W_m}{W_m^{th}} \forall m \in \mathcal{M} \setminus \{m'\}$ ;
10    Select  $m'' | \frac{W_{m''}}{W_{m''}^{th}} \leq \frac{W_m}{W_m^{th}} \forall m \in \mathcal{M} \setminus \{m''\}$ ;
11    Compute  $N_{m',z}^{min} = N_{m'}^{min} + 1$ ;  $N_{m'',z}^{min} = N_{m''}^{min} - 1$ ;
12  else
13    stop = True;
14  end
15 end
16 return  $N_m^{min}$ ,  $W_m$ ;

```

the objective function. To that end, it selects the service m' with the best ratio $W_{m'}/W_{m'}^{th}$ and the service m'' with the worst ratio $W_{m''}/W_{m''}^{th}$ (steps 9-10). The algorithm then reallocates one guaranteed RB from service m'' to service m' (step 11). A new iteration of the algorithm starts if $N_{m,z}^{min} \forall m \in \mathcal{M}$ improves the objective function. Conversely, the iterative procedure ends if the objective function can not be further improved (step 13).

Regarding the computational complexity of Algorithm 2, the bottleneck lies in line 5, which depends on the execution of Algorithm 1. The time complexity results in $\mathcal{O}([T_{m,n} \cdot N_{add} + T_{OBS}] \cdot N_{max} \cdot |\mathcal{M}|)$, which corresponds to the complexity of Algorithm 1 multiplied by the number of services $|\mathcal{M}|$. The overall complexity of Algorithm 2 is then $\mathcal{O}([T_{m,n} \cdot N_{add} + T_{OBS}] \cdot N_{max} \cdot |\mathcal{M}| \cdot N_{ite})$, where N_{ite} represents the number of iterations of the while loop. This parameter depends on the number of services $|\mathcal{M}|$ as well as the total number of radio resources N_{cell}^{RB} . As shown later in Section V-B, the value of N_{ite} increases with the number of radio resources. Although this is not explicitly reflected in the results for N_{ite} , a similar trend has been experimentally observed with respect to the number of services $|\mathcal{M}|$.

D. RT Control Loop

Every T_{OUT} TTIs, the RT Controller *dApp* receives the new value of $N_m^{min} \forall m \in \mathcal{M}$ from the Delay-Aware RB Allocation Controller *xApp* (see point B in Fig. 2). Based on them, the RT Controller *dApp* operates at each TTI as follows. First, it tries to drain the transmission queue of each service m by using N_m^{min} RBs. After, a subset $\mathcal{M}' \subset \mathcal{M}$ of services will have drained their queues, while the remaining $\mathcal{M}'' = \mathcal{M} - \mathcal{M}'$ services may still have pending transmissions. Assuming the $|\mathcal{M}'|$ services have not fully consumed their guaranteed amount of RBs N_m^{min} , the total remaining free RBs can be represented as N_{fr} . These spare resources, which comprise the sum of the resources not utilized by the $|\mathcal{M}'|$ services, can then be allocated to the remaining $|\mathcal{M}''|$ services (see point D in Fig. 2). Specifically, the N_{fr} RBs will be allocated among the $|\mathcal{M}''|$ services following the Earliest Deadline First (EDF) discipline [35] since it minimizes the number of packets whose transmission delay is above the target delay budget. Note EDF does not consider the violation probability [36]. For this reason, the RT Controller *dApp* combines EDF with the establishment of guaranteed RBs per service. Since the latter are decided by the Delay-Aware RB Allocation Controller *xApp*, our framework ensures $\mathbb{P}[w > W_m] \leq \varepsilon_m \forall m \in \mathcal{M}$ as long as the traffic and channel conditions do not change with respect to the samples $\vec{x}_{a,m}$, $\vec{x}_{s,m,n}$.

If the traffic and/or channel conditions change, $\mathbb{P}[w > W_m]$ may be greater than the violation probability ε_m . To avoid it whenever possible, the RT Controller *dApp* executes Algorithm 3 to temporarily adjust the number of guaranteed RBs N_m^{min} of the set \mathcal{M} of services (see point E in Fig. 2).

This algorithm monitors the waiting time of the first packet of each service in the transmission queue. Then, if the waiting time is close to the delay budget, the algorithm increases (if possible) the amount of guaranteed RBs for the corresponding

Algorithm 3: Mitigating $w > W_m^{th}$ at TTI i

```

1 Inputs:  $N_m^{min}$ ,  $Q_{T,m}^U$ ,  $Q_{T,m}^L$ ,  $\vec{s}_{i-1}$ ;
2 Compute  $q_{i,m} \forall m \in \mathcal{M}$ ;
3 Update states  $\vec{s}_i$  and  $N_{m,i}^{req}$  according to Fig. 5;
4  $\vec{v}_a = \{m' \mid \forall m' \mid \vec{s}_i\{m'\} = A\}$ ;
5  $\vec{v}_b = \{m'' \mid \forall m'' \mid \vec{s}_i\{m''\} = B \cup C\}$ ;
6 if  $\vec{v}_d \neq \emptyset$  then
7   Set  $N_{ite} = \sum_{m''} N_{m'',i}^{req}$  and  $N_{m,i}^{min} = N_m^{min} \forall m \in \mathcal{M}$ ;
8   Set  $j_a = 0$  and  $j_b = 0$ ;
9   for  $u$  from 1 to  $N_{ite}$  do
10    Determine  $n' = \vec{v}_a\{j_a\}$  and  $n'' = \vec{v}_b\{j_b\}$ ;
11    Update  $N_{n',i}^{min} = N_{n',i}^{min} - 1$ ;  $N_{n'',i}^{min} = N_{n'',i}^{min} + 1$ ;
12    Update  $j_a = j_a + 1$  and  $j_b = j_b + 1$ ;
13    if  $j_a == |\vec{v}_a|$  then
14       $j_a = 0$ 
15    end
16    if  $j_b == |\vec{v}_b|$  then
17       $j_b = 0$ 
18    end
19  end
20 end
21 return:  $N_{m,i}^{min}$ ,  $N_{m,i}^{req}$ ;

```

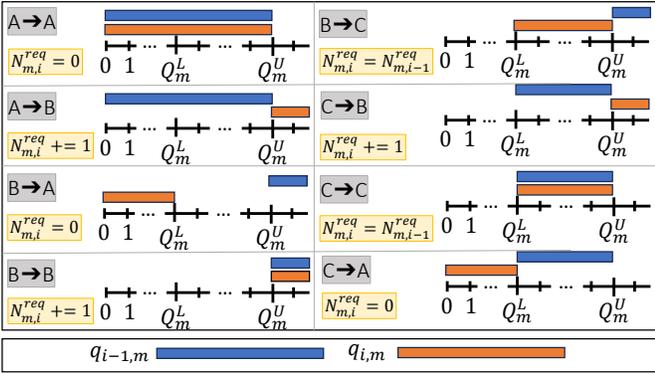


Fig. 5. Transitions of the finite-state machine to control the RB allocation for service m . From [11].

service. To this end, Algorithm 3 relies on a finite-state machine of three states $\{A, B, C\}$ based on two thresholds $Q_{T,m}^U$ and $Q_{T,m}^L$. The threshold $Q_{T,m}^U = \eta Q_{T,m}$ indicates the waiting time of a packet is close to the delay budget, whereas $Q_{T,m}^L = \tau Q_{T,m}$ indicates the waiting time is far to the delay budget. The parameter $Q_{T,m} = \lfloor W_m^{th}/t_{slot} \rfloor$ is the maximum number of TTIs that a packet can wait in the queue before crossing the delay budget. Additionally, $Q_{T,m}^U > Q_{T,m}^L$. Note that $\eta \in (0, 1]$ and $\tau \in (0, 1]$ can be tuned by the MNO. Regarding the states, the state A indicates the RT Controller $dApp$ allocates to the service m the amount of guaranteed RBs decided by the Delay-Aware RB Allocation Controller $xApp$, i.e., $N_{m,i}^{min} = N_m^{min}$. Note that we define $N_{m,i}^{min}$ as the amount of guaranteed RBs decided by the RT Controller $dApp$ in the TTI i . The state B indicates the waiting time w of the first packet of service m is very close to the delay budget (i.e., $w/t_{slot} > Q_{T,m}^U$), thus the RT Controller $dApp$ may increase the amount of guaranteed RBs for such service. Specifically, it may increase $N_{m,i}^{req}$ RBs. In state B , $N_{m,i}^{req}$ increases by one RB with respect to the previous TTI. The state C indicates the waiting time of the first packet is lower than in state B ,

but not enough to go back to $N_{m,i}^{min} = N_m^{min}$. In such case, Algorithm 3 keeps the same value of $N_{m,i}^{req}$ with respect to the previous TTI. In Fig. 5 we summarize the possible transitions among states. Considering such transitions, we define \vec{s}_i as a vector containing the state for each service at TTI i .

Based on N_m^{min} , $Q_{T,m}^L$, $Q_{T,m}^U$ and \vec{s}_{i-1} , Algorithm 3 initially computes the state of the first packet of each service as $q_{i,m} = w_m^{pkt}/t_{slot}$ (step 2). Note that w_m^{pkt} is the waiting time of such a packet. Then, it updates the states \vec{s}_i and $N_{m,i}^{req}$ according to the transitions depicted in Fig. 5 (step 3). Later, Algorithm 3 needs to check if the amount of RBs defined in $N_{m,i}^{req}$ can be allocated, in addition to N_m^{min} , to the corresponding services. The policy considered by Algorithm 3 is that only the services whose state is A can donate RBs to those which require more RBs. Considering this policy, Algorithm 3 iteratively re-allocates the amount of guarantees RBs from services in state A to services in states B or C (steps 4-20).

The computational complexity of Algorithm 3 can be generically expressed as $\mathcal{O}(|\mathcal{M}| + N_{ite})$. In particular, lines 2-5 involve checking the transmission queue state for each service $m \in \mathcal{M}$, while the remaining lines depend on the number of iterations N_{ite} . In turn, N_{ite} is governed by two key factors. First, there must exist services capable of donating radio resources (see line 4 in Algorithm 3), meaning their transmission queues must be in state A . If this condition is met, then N_{ite} depends on the number of additional resources requested by the services whose transmission queues are in state B or C (see line 7 in Algorithm 3). Both $|\mathcal{M}|$ and N_{ite} depend on the number of services as well as the state of their transmission queues at the time Algorithm 3 is executed.

V. PERFORMANCE RESULTS

We conducted an extensive simulation campaign to validate MAREA and assess its performance using a Python-based simulator running on a computing platform with 16 GB RAM and a quad-core Intel Core i7-7700HQ @ 2.80 GHz. The simulator models a single cell utilizing an Orthogonal Frequency-Division Multiple Access (OFDMA) scheme with $N_{cell}^{RB} \in [50, 100]$ RBs and $t_{slot} = 1$ ms. To simulate the traffic and cell capacity of each uRLLC service, we use realistic traces collected from an operational RAN with the FALCON tool [37]. FALCON enables the decoding of the Physical Downlink Control Channel (PDCCH) of a vBS, providing insight into the number of active users and their allocated resources. Fig. 6 illustrates the incoming bits per TTI measured by FALCON, along with the corresponding PMF in the upper plots. To emulate the incoming traffic for three distinct uRLLC services, we sorted the active UEs and divided them into equally sized groups based on their aggregated traffic demand. The resulting PMFs for these groups are also shown in the lower plots of Fig. 6. For these services, we set a delay budget $W_m^{th} = \{5, 10, 15\}$ ms and target violation probability $\varepsilon_m = \{10^{-5}, 10^{-4}, 10^{-3}\}$ [38]–[40].

A. Validation Martingales-based Model

In the first experiment, we examine a cell hosting a single service m with incoming traffic measured by FALCON (i.e.,

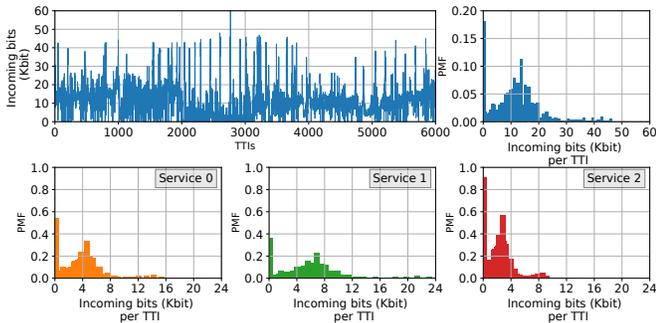


Fig. 6. PMFs of incoming bits from FALCON traces. Copied from [11].

blue plots in Fig. 6). Assuming a target violation probability $\varepsilon_m = 10^{-3}$, we estimate the delay bound using the proposed Martingales-based model and compare it with the real delay bound obtained through simulation. We also compare this with delay bounds estimated using the SNC-based model proposed in [11]. The Delay-Aware RB Allocation Controller $xApp$ (a) allocates $N_m^{min} \in [40, 100]$ RBs for the service, and (b) uses $T_{OBS} \in \{1, 2, 3, 4, 5, 6\} \cdot 1000$ TTIs to obtain the sample vectors $\vec{x}_{a,m}$ and $\vec{x}_{s,m,n}$.

Fig. 7 illustrates the evolution of the delay bound as the Delay-Aware RB Allocation Controller $xApp$ allocates a specific number of RBs to the service. The left plot depicts the delay bound estimation using the SNC-based model, while the right plot represents the estimation from the proposed Martingales-based model. Both plots include red boxplots representing the delay bounds measured experimentally through simulation. For each specific RB allocation, the delay bound was experimentally measured 50 times, with each measurement based on a simulation of 4 million TTIs.

The results show that delay bounds from the SNC model are significantly less accurate than those from the proposed Martingales-based model, which closely aligns with real values. To quantify this difference, Fig. 8 illustrates the relative error between the actual delay bounds (measured in simulations) and those estimated by both models. The average relative error ranges from 15% to 25% for the Martingales-based model, compared to around 300% for the SNC model, demonstrating superior accuracy.

When comparing measurements for different T_{OBS} values, both models provide reliable estimates when $T_{OBS} \geq 4000$. Below this threshold, estimations are less accurate, especially with fewer RBs are allocated. For the Martingales-based model, we can see that the delay bound estimation is significantly higher than the real value for the blue and orange curves ($T_{OBS} = 1000$ and $T_{OBS} = 2000$, respectively). For the SNC model, even with the green curve ($T_{OBS} = 3000$), the estimation is no longer conservative (i.e., the estimated value exceeds the real value). Note that the SNC-based model guarantees accurate upper-bound estimations only when the PMF for both arrival and service processes is correctly captured [11]. Due to insufficient samples when $T_{OBS} < 4000$, this is not the case. This fact also explains why the SNC model shows a "lower" relative error with few samples. However, this lower error is misleading; the estimation is not better but appears so because, with insufficient samples, the estimates

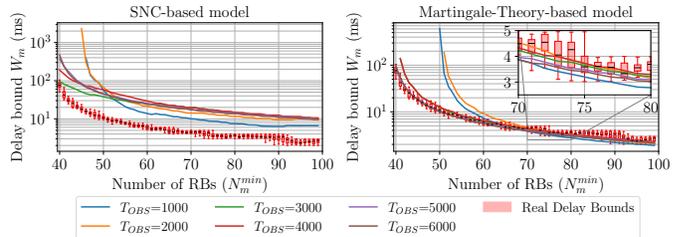


Fig. 7. Evaluation of delay bound W_m for a service m based on the number of guaranteed RBs N_m^{min} and considering a specific metric collection period size T_{OBS} .

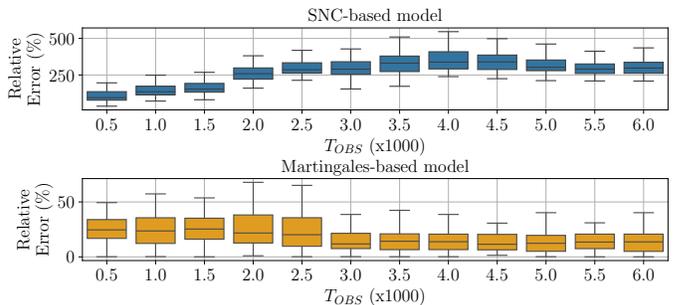


Fig. 8. Evaluation of the relative error between the estimated delay bound and the delay bound measured through simulation, considering a specific metric collection period size T_{OBS} .

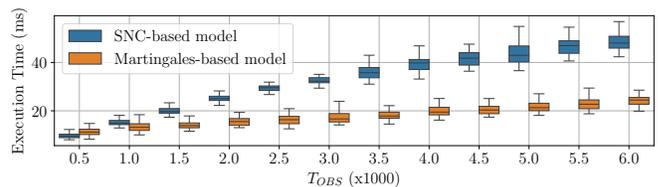


Fig. 9. Execution time comparison between the Martingales-based model and the SNC-based model for estimating the delay bound W_m .

and actual values intersect at lower RB values, creating a false impression of improved accuracy.

The execution times of the Martingales-based model and the SNC-based model were also compared, as shown in Fig. 9. The boxplot illustrates that both models exhibit a linear increase in execution time with T_{OBS} , but the SNC-based model's time grows more rapidly. For $T_{OBS} = 6000$, the SNC-based model's execution time is twice that of the Martingales-based model. This is due to the Martingales-based model only needing to optimize one parameter, θ_m^* , while the SNC-based model optimizes two parameters [11].

Analyses of the previous results show that the Martingales-based model provides better delay bound estimations and faster execution times than the SNC-based model, making it ideal for integration into the proposed framework.

B. Performance Analysis of the Delay-Aware RB Allocation Controller $xApp$

In a third set of experiments, we focus on a single decision period of the Delay-Aware RB Allocation Controller $xApp$ when it consider both, the Martingales-based model and the SNC-based model [11] to estimate the delay bound.

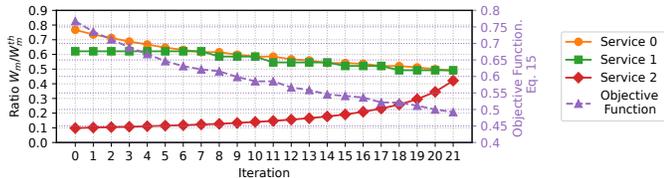


Fig. 10. Convergence Analysis of Algorithm 2.

TABLE I
PERFORMANCE COMPARISON OF DELAY-AWARE RB ALLOCATION
CONTROLLER $xApp$ USING SNC-BASED AND MARTINGALES-BASED
MODELS WITH A BRUTE FORCE ALGORITHM.

N_{cell}^{RB}	60	70	80	90	100
Relative error (%) SNC-based model	23.07	21.43	23.05	21.43	17.35
Relative error (%) Maringale-based model	0	0	0	0	0
Brute Force (iterations)	1711	2346	3081	3916	4851
Algorithm 2 (iterations) SNC-based model	12	14	16	19	22
Algorithm 2 (iterations) Martingales-based model	12	14	16	20	22
Ratio iterations SNC-based model	142.58	167.57	192.56	206.10	220.5
Ratio iterations Martingales-based model	142.58	167.57	192.56	195.8	220.5
Avg. Execution Time per iteration (ms) when SNC-based model	184.62	179.09	176.39	173.76	171.85
Avg. Execution Time per iteration (ms) when Martingales-based model	106.38	99.67	91.63	88.95	87.90

Specifically, we assume this $xApp$ computes N_m^{min} for the three services described at the beginning of Section V. Under this scenario, we evaluate the convergence of the heuristics proposed in Algorithm 2, the computational complexity of such heuristics, and the accuracy of the obtained solution with respect to the optimal.

First, we evaluate the convergence of Algorithm 2. We consider the Delay-Aware RB Allocation Controller $xApp$ applies the Martingales-based model to derive the delay bound. Fig. 10 depicts the objective function value $g(\vec{W})$, i.e., purple curve, and the ratios $W_m/W_m^{th} \forall m \in \mathcal{M}$, i.e., orange, green and red curves, across iterations. The heuristics progressively reduce $g(\vec{W})$ until achieving the optimal solution.

Second, we compare the heuristic solution with the optimal one obtained via brute force for $N_{cell}^{RB} \in [60, 100]$ RBs. The heuristics employ both the Martingales-based model and the SNC-based model [11]. Table I shows that the heuristic with the Martingales-based model reaches the optimal solution, while the SNC-based model yields a near-optimal solution with a 20% relative error. This confirms that our heuristic can provide near-optimal solutions efficiently, avoiding the computationally expensive brute force method.

Furthermore, we observe the number of iterations increases with N_{cell}^{RB} due to the larger search space, while the average execution time per iteration decreases slightly. This is because estimating W_m (via Algorithm 1 to determine θ_m^* and W_m) becomes faster with more RBs. Specifically, fewer iterations are needed as N_{cell}^{RB} increases, as previously shown in Fig 3. Comparing execution times, the Delay-Aware RB Allocation Controller $xApp$ performs iterations faster with the

Martingales-based model than with the SNC-based model, as the former estimates delay bounds more quickly as previously depicted in Fig. 9.

C. Service Accommodation Capability of the Delay-Aware RB Allocation Controller $xApp$

In this experiment, we evaluated the service accommodation capability of the Delay-Aware RB Allocation Controller $xApp$, which can use either the proposed Martingales-based model or the SNC-based model presented in [11]. Specifically, we assessed each controller's ability to allocate radio resources to multiple coexisting uRLLC services within a cell, ensuring that the delay requirements of these services are met simultaneously. This means the objective function $g(\vec{W})$ defined in Eq. (15) must be less than or equal to 1.

Results are presented in Fig. 11, which shows the value of the objective function $g(\vec{W})$ when deploying a specific number of uRLLC services given a fixed number of RBs available in the cell. The left plot displays the objective function value for the SNC-based controller, while the right plot shows the value for the Martingales-based controller. Note that both controllers were evaluated under the same conditions, including identical services with the same latency requirements, traffic generation distribution, and channel conditions.

The results indicate that the Martingales-based controller can accommodate up to 4 services simultaneously with 50 RBs, whereas the SNC-based controller can only accommodate a single service with the same number of RBs. Even with 90 RBs, the SNC-based controller can handle only 3 services, which is fewer than the number of services accommodated by the Martingales-based controller with 50 RBs. Although services orchestrated by the SNC-based controller would experience lower delay bounds (see results of Section V-A), the conservative estimations of the SNC-based model limits the number of services that can be deployed by the MNO. In contrast, the Martingales-based model provides more accurate delay bound estimations, allowing the proposed controller to deploy a greater number of services. This highlights the significant advantage of using a Martingales-based model over an SNC-based model for orchestrating multiple uRLLC services.

D. Performance Analysis of MAREA Framework

In the last experiment, we evaluated the performance of MAREA against three reference solutions. These solutions are

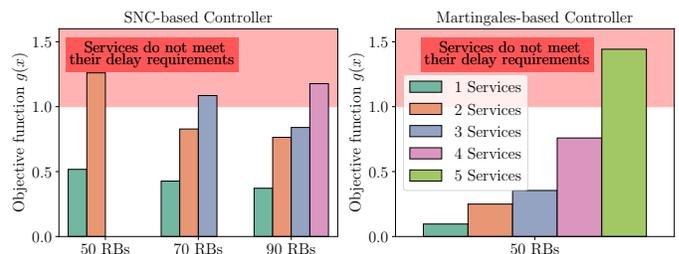


Fig. 11. Analysis of the service accommodation capability when the cell has available a specific amount of RBs.

based on different resource allocation mechanisms and control strategies, as detailed below:

- **Reference solution #1:** only considers a single RT-Controller (*dApp*) in the O-RAN architecture, located in the CU-CP. The RT-Controller implements an EDF scheduler across all $|\mathcal{M}|$ services. For each service $m \in \mathcal{M}$, a transmission queue is considered, and the scheduler monitors the waiting time of the first packet in each queue. The packet whose waiting time is closest to its corresponding delay bound W_m^{th} is transmitted first.
- **Reference solution #2:** the Queue Length and Delay Requirement (QLDR)-Based Algorithm, proposed in [41], operates in two main steps. First, for each service $m \in \mathcal{M}$, the average transmission queue length (in bits) and the average spectral efficiency (in bits/s/Hz) are computed over the past T TTIs. Then, the average queue length is normalized by the average spectral efficiency and further divided by the service's target delay bound W_m^{th} . Finally, the available RBs N_{RB}^{cell} are allocated among these services in proportion to the previous normalized values. Note that $N_m^{min} \forall m \in \mathcal{M}$ is updated every T TTIs.
- **Reference solution #3:** utilizes all the *xApps* proposed in MAREA framework, except for the RT Controller *dApp*. Specifically, the Delay-Aware RB Allocation Controller *xApp* is responsible for periodically calculating the RBs required by each service $m \in \mathcal{M}$, as in MAREA. However, a key distinction is that this solution does not allow for RB sharing between services during the assignment period. Each service $m \in \mathcal{M}$ is allocated dedicated N_m^{min} RBs, and those resources that are not being used by a service (in specific TTIs) cannot be shared with other services, unlike MAREA, which allows sharing of unused RBs between services within the same assignment period.
- **Reference solution #4:** the RT Controller *dApp* is included, along with the remaining MAREA's *xApps*, enabling the sharing of RBs among services. Specifically, if certain services are not utilizing their full allocated RBs at specific TTIs, those unused RBs can be reassigned to other services with higher instantaneous traffic load. However, the RB sharing is implemented using an EDF scheduler, without considering Algorithm 3, which is a key aspect of MAREA. In our framework, the EDF scheduler is complemented by Algorithm 3, which takes into account the state of the transmission queues for each service $m \in \mathcal{M}$, providing a more sophisticated approach to resource sharing.

To measure the performance of these solutions, we consider the Complementary Cumulative Distribution Function (CCDF) of the metric $(w - W_m^{th})/W_m^{th}$. Note the random variable w represents the transmission delay of an arbitrary packet. Additionally, when this metric is equal to 0, the CCDF value represents the violation probability. In Fig. 12 we observe that the reference solution #3 provides the worst performance, i.e., a violation probability 11.46 and 178.30 times larger than MAREA for services 1 and 2. Note that this probability is equal to 0 for scenario 3 when we consider MAREA. These

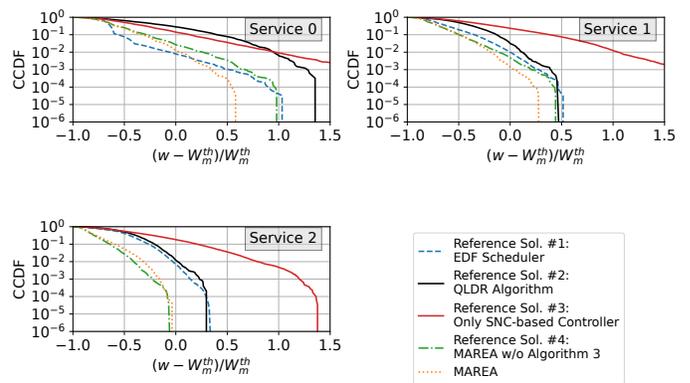


Fig. 12. Complementary Cumulative Distribution Function (CCDF) for $(w - W_m^{th})/W_m^{th}$. In MAREA, we have set $\eta = 0.75$ and $\tau = 0.3$.

results are due to the fact the reference solution #2 only considers dedicated RBs. Reference solution #2 also considers dedicated RBs, but outperforms reference solution #3. This is because its dedicated RB allocation is updated every 10 TTIs, whereas in reference solution #3, it is updated every 1000 TTIs. As a result, despite not using a delay estimation model, reference solution #2 better accounts for the current transmission queue state, capturing the impact of unexpected traffic behavior on queue occupancy and the current radio channel capacity for these users. The remaining solutions consider sharing RBs among the services. Comparing them, the reference solution #1 usually provides a greater violation probability with respect to MAREA. Although EDF ensures the packet with the earliest deadline are transmitted first, e.g., we observe for service 0 the CCDF is lower for reference solution #1 when $(w - W_m^{th})/W_m^{th} \leq 0$, it does not provide any guarantees in terms of violation probability. To consider such probability, the MAREA framework establishes guaranteed RBs in a near-RT and allocates share free RBs among services using EDF in a RT scale. It improves the performance with respect to the remaining solutions. Specifically, MAREA provides lower violation probabilities for service #1 and service #2. Finally, we observe the consideration of Algorithm 5 in MAREA improves the obtained violation probability if we compare the results with the ones obtained by the reference solution #4. The improvement is most significant when the metric $(w - W_m^{th})/W_m^{th}$ is higher. This is due to Algorithm 5 performing its actions when the waiting time of a packet in the transmission queue is closer to the delay budget. We can also observe the reference solution #4 has a similar behavior as MAREA for service 2. The packet transmission delay is never above the delay budget for this service, as happens for services 0 and 1.

VI. CHALLENGES FOR IMPLEMENTING MAREA IN REAL-WORLD O-RAN DEPLOYMENTS

The deployment of MAREA in an O-RAN testbed would allow to evaluate its performance in real radio conditions based on specific deployment scenarios. However, implementing MAREA in such an environment presents several challenges that require extensive investigation and are thus beyond the scope of this work.

Real-Time Task Execution. A key challenge is ensuring real-time task execution across the different functional blocks of MAREA. Its two control loops, working at near-RT and RT timescales, need fast processing to make sure radio resource allocations happen on time. In particular, the RT control loop needs hardware optimization to adjust radio resource allocation at each TTI or every few TTIs, depending on the transmission queue status of uRLLC services. This can be done using hardware accelerators like FPGAs and/or GPUs to run these time-critical tasks efficiently. The advantages of such hardware acceleration in O-RAN systems have been shown in testbeds like X5G [42], [43], which uses NVIDIA Aerial to offload physical layer tasks, improve performance through parallel processing, and enable AI/ML-based optimization.

Compatibility with standardized interfaces. To integrate MAREA’s *xApps* and *dApps* with existing O-RAN components, their functionalities need to be adapted so that their input and output data match the O-RAN standardized APIs. This involves adjusting data formats and ensuring they can interact with the E2 interface for real-time radio resource control [23], [24], [44], [45].

Access to Licensed Spectrum. Access to licensed spectrum is one of the key challenges in integrating MAREA into a commercial O-RAN platform. Obtaining licenses for 5th Generation (5G) frequency bands is often complicated due to regulatory restrictions. However, in some countries, certain frequency bands, such as those in the 3.5 GHz range, have been partially allocated for industrial and research use, offering more flexibility for testing [46]. These frequency allocations, available through regulatory sandboxes or collaborative testbed initiatives [47], could enable more adaptable experimentation.

VII. RELATED WORKS

Several studies have explored radio resource allocation in networks where enhanced Mobile Broadband (eMBB) and uRLLC services coexist, leveraging puncturing to minimize uRLLC packet transmission delay. Bairagi et al. [48] formulate an optimization problem to maximize the minimum expected eMBB rate while ensuring efficient uRLLC allocation. Their approach combines Penalty Successive Upper Bound Minimization (PSUM) for eMBB scheduling with a Transportation Model (TM) for uRLLC, improving fairness and minimum achieved rates. Similarly, Alsenwi et al. [49] propose a risk-sensitive slicing framework that optimizes eMBB puncturing probability while maintaining reliability. Their method models the tail distribution of eMBB rates and employs convex relaxation for iterative optimization. For a comprehensive review of uRLLC puncturing, see [38]. However, these works do not address networks with only uRLLC services or integrate O-RAN architectural constraints.

Other authors have addressed radio resource allocation in networks with multiple uRLLC services. In the context of O-RAN, Abedin et al. [50] propose an actor-critic framework to minimize the probability of Internet of Things (IoT) devices exceeding an age of information threshold, though it overlooks air interface transmission delay. Karbalaee et al. [51] introduce an iterative algorithm for joint radio resource

and power allocation, while Rezazadeh et al. [52] address the problem via federated learning—both focusing primarily on average delay. Polese et al. [53] evaluate Deep Reinforcement Learning (DRL) agents within a non-RT control loop in O-RAN. Despite their contributions, these works lack details on multi-time-scale control loop interactions.

Non-RT and near-RT solutions often rely on queuing theory to model packet transmission delay [54]–[56], but these models only yield average values under complex arrival and channel capacity distributions. Alternatively, some works [7]–[9] use SNC to estimate a delay bound W of type $\mathbb{P}[w > W] < \varepsilon$, where w is the delay and ε a target tolerance. However, they do not account for multiple uRLLC services or cross-interference. In [10], we proposed an SNC-based controller for planning multiple uRLLC services, but it assumes dedicated resources, potentially leading to inefficiencies, and is limited to Poisson batch arrivals. Recently, we introduced an O-RAN-compliant framework based on a novel SNC model for multi-service uRLLC allocation [11], incorporating real traffic and capacity metrics. However, SNC-based delay bounds often deviate significantly from actual values, restricting the number of deployable services. Martingale queueing methods have demonstrated significantly improved delay bound estimations. Poloczek and Ciucu derive tight stochastic delay bounds for Markovian sources over ALOHA and CSMA/CA [13], [14], while Zhao et al. optimize delay and energy efficiency in Machine Type Communication (mMTC) using differentiated ALOHA [15]. Other works analyze end-to-end delay in multi-hop networks [57] and multiplexing of uRLLC and eMBB traffic using reconfigurable intelligent surfaces [17]. Yu et al. propose frameworks for bandwidth abstraction and network reliability under strict latency constraints [58]. Despite their advancements, these studies assume well-known traffic distributions (Poisson, Bernoulli, or Markov on-off), limiting their applicability to real-world traffic patterns with arbitrary distributions.

Considering RT solutions, [35], [59], [60] use schedulers based on EDF to assign priorities to packets based on their deadlines. EDF ensures the packets with the earliest deadline are transmitted first. However, EDF does not consider the probability the packet transmission delay exceeds a delay budget [61]. Other solutions such as [62]–[64] rely on ML models. Although they are effective in managing scenarios with intricate traffic patterns and channel conditions, their performance is primarily reliant on the similarity between the measured patterns and those used during training.

O-RAN specifications mention a RT control loop for optimizing tasks such as packet scheduling or interference recognition [65]. However, at the moment of writing this paper, such a control loop has not been defined. In the same row, the authors of [6] introduce the concept of *dApps* to implement fine-grained RT control tasks. Despite implementing a proof-of-concept, they omit to detail how multiple uRLLC services can be orchestrated in a RT scale, and how the near-RT control loop interacts with the *dApps*.

VIII. CONCLUSIONS

In this paper, we tackled the challenge of implementing efficient multi-time-scale control loops in O-RAN-based deployments for uRLLC services. We introduced MAREA, an O-RAN-compliant framework specifically designed to address the radio resource allocation problem at both near-RT and RT scales. Central to our approach is the use of a novel Martingales-based model, to accurately compute the required number of guaranteed RBs per service, ensuring that the violation probability—i.e., the probability that packet transmission delays exceed a predefined threshold—remains below the target tolerance. An additional key innovation in MAREA is the integration of an RT control loop, which continuously monitors the transmission queues of each service and dynamically adjusts the allocation of guaranteed RBs to address traffic anomalies in real time. Through extensive simulation, MAREA demonstrated significant improvements over existing solutions, achieving an average violation probability that is $\times 10$ lower than reference methods. This highlights the potential of our framework to enhance the reliability and efficiency of O-RAN-based uRLLC deployments, making it a promising approach for future networks.

REFERENCES

- [1] P. Popovski *et al.*, “Wireless Access for Ultra-Reliable Low-Latency Communication: Principles and Building Blocks,” *IEEE Netw.*, vol. 32, no. 2, pp. 16–23, 2018.
- [2] B. Tang, V. K. Shah, V. Marojevic, and J. H. Reed, “AI Testing Framework for Next-G O-RAN Networks: Requirements, Design, and Research Opportunities,” *IEEE Wirel. Commun.*, vol. 30, no. 1, pp. 70–77, 2023.
- [3] A. S. Abdalla, P. S. Upadhyaya, V. K. Shah, and V. Marojevic, “Toward Next Generation Open Radio Access Networks: What O-RAN Can and Cannot Do!” *IEEE Netw.*, vol. 36, no. 6, pp. 206–213, 2022.
- [4] M. Polese, L. Bonati, S. D’Oro, S. Basagni, and T. Melodia, “Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges,” *IEEE Commun. Surv. Tutor.*, pp. 1–1, 2023.
- [5] L. Zanzi, V. Sciancalepore, A. Garcia-Saavedra, H. D. Schotten, and X. Costa-Pérez, “LACO: A Latency-Driven Network Slicing Orchestration in Beyond-5G Networks,” *IEEE Trans. Wirel. Commun.*, vol. 20, no. 1, pp. 667–682, 2021.
- [6] S. D’Oro, M. Polese, L. Bonati, H. Cheng, and T. Melodia, “dApps: Distributed Applications for Real-Time Inference and Control in O-RAN,” *IEEE Commun. Mag.*, vol. 60, no. 11, pp. 52–58, 2022.
- [7] C. Xiao *et al.*, “Downlink MIMO-NOMA for Ultra-Reliable Low-Latency Communications,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 4, pp. 780–794, 2019.
- [8] S. Schiessl, M. Skoglund, and J. Gross, “NOMA in the Uplink: Delay Analysis With Imperfect CSI and Finite-Length Coding,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3879–3893, 2020.
- [9] Á. A. Cardoso, M. V. G. Ferreira, and F. H. T. Vieira, “Delay bound estimation for multicarrier 5G systems considering lognormal beta traffic envelope and stochastic service curve,” *Trans. Emerg. Telecommun. Technol.*, p. e4281, 2021.
- [10] O. Adamuz-Hinojosa, V. Sciancalepore, P. Ameigeiras, J. M. Lopez-Soler, and X. Costa-Pérez, “A Stochastic Network Calculus (SNC)-Based Model for Planning B5G uRLLC RAN Slices,” *IEEE Trans. Wirel. Commun.*, vol. 22, no. 2, pp. 1250–1265, 2023.
- [11] O. Adamuz-Hinojosa, L. Zanzi, V. Sciancalepore, A. Garcia-Saavedra, and X. Costa-Pérez, “ORANUS: Latency-tailored Orchestration via Stochastic Network Calculus in 6G O-RAN,” in *IEEE INFOCOM*, 2024, pp. 61–70.
- [12] M. Fidler and A. Rizk, “A Guide to the Stochastic Network Calculus,” *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 92–105, 2015.
- [13] F. Ciucu, F. Poloczek, and J. Schmitt, “Sharp per-flow delay bounds for bursty arrivals: The case of FIFO, SP, and EDF scheduling,” in *IEEE INFOCOM*, 2014, pp. 1896–1904.
- [14] F. Poloczek and F. Ciucu, “Service-martingales: Theory and applications to the delay analysis of random access protocols,” in *IEEE INFOCOM*, 2015, pp. 945–953.
- [15] L. Zhao, X. Chi, and Y. Zhu, “Martingales-Based Energy-Efficient D-ALOHA Algorithms for MTC Networks With Delay-Insensitive/URLLC Terminals Co-Existence,” *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1285–1298, 2018.
- [16] H. Yan, X. Chi, W. Yang, and Z. Han, “End-to-end delay analysis for multi-hop wireless links based on martingale theory,” *IEEE Trans. Veh. Technol.*, pp. 1–7, 2024.
- [17] H. Peng, C.-C. Hsia, Z. Han, and L.-C. Wang, “A Generalized Delay and Backlog Analysis for Multiplexing URLLC and eMBB: Reconfigurable Intelligent Surfaces or Decode-and-Forward?” *IEEE Trans. Wireless Commun.*, vol. 23, no. 5, pp. 4049–4068, 2024.
- [18] B. Yu, X. Chi, S. Li, X. Liu, and S. Ren, “Martingale-Based URLLC Slice Customization for the Provisioning of Reliability With Regard to End-to-End Latency,” *IEEE Internet Things J.*, vol. 11, no. 1, pp. 1311–1327, 2024.
- [19] J. Ordonez-Lucena, P. Ameigeiras, L. M. Contreras, J. Folgueira, and D. R. López, “On the Rollout of Network Slicing in Carrier Networks: A Technology Radar,” *Sensors*, vol. 21, no. 23, 2021.
- [20] C.-Y. Chang and N. Nikaiein, “Closing in on 5G Control Apps: Enabling Multiservice Programmability in a Disaggregated Radio Access Network,” *IEEE Veh. Technol. Mag.*, vol. 13, no. 4, pp. 80–93, 2018.
- [21] O-RAN Working Group 3, “Near-Real-time RAN Intelligent Controller, Use Cases and Requirements 3.0,” Mar. 2023.
- [22] O-RAN Working Group 1, “Use Cases Analysis Report 10.0,” Mar. 2023.
- [23] O-RAN Alliance, “O-RAN E2 Service Model (E2SM) KPM 5.0,” *Technical Specification*, no. O-RAN.WG3.E2SM-KPM-R003-v05.00, June 2024.
- [24] O-RAN Alliance, “O-RAN E2 Service Model (E2SM), RAN Control 6.0,” June 2024.
- [25] E. Markova *et al.*, “Performance-Utilization Trade-Offs for State Update Services in 5G NR Systems,” *IEEE Access*, vol. 12, pp. 129 789–129 803, 2024.
- [26] 3GPP TS 28.552, “Management and orchestration; 5G performance measurements (Release 19),” no. V19.2.0, Dec. 2024.
- [27] O-RAN next Generation Research Group (nGRG), “dApps for Real-Time RAN Control: Use Cases and Requirements,” Northeastern University, NVIDIA, Mavenir, MITRE, Qualcomm, Tech. Rep. RR-2024-10, October 2024, contributed Research Report. [Online]. Available: <https://mediastorage.o-ran.org/ngrg-rr/nGRG-RR-2024-10-dApp%20use%20cases%20and%20requirements.pdf#page=17.09>
- [28] F. Poloczek and F. Ciucu, “Scheduling analysis with martingales,” *Performance Evaluation*, vol. 79, pp. 56–72, 2014, special Issue: Performance 2014.
- [29] A. Wald, “On cumulative sums of random variables,” *The Annals of Mathematical Statistics*, vol. 15, no. 3, pp. 283–296, 1944.
- [30] R. L. Burden and J. D. Faires, “2.1 The bisection algorithm,” *Numerical analysis*, vol. 3, 1985.
- [31] C. M. Bishop, “Mixture Density Networks,” 1994.
- [32] B. Selim, O. Alhussein, S. Muhaidat, G. K. Karagiannidis, and J. Liang, “Modeling and Analysis of Wireless Channels via the Mixture of Gaussian Distribution,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 10, pp. 8309–8321, 2016.
- [33] K. Plataniotis and D. Hatzinakos, “Gaussian Mixtures and Their Applications to Signal Processing,” *Advanced Signal Processing Handbook*, CRC Press, pp. 89–124, 2017.
- [34] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake, “Finite Mixture Models,” *Annual Review of Statistics and Its Application*, vol. 6, no. 1, pp. 355–378, 2019.
- [35] T. Guo and A. Suárez, “Enabling 5G RAN Slicing With EDF Slice Scheduling,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2865–2877, 2019.
- [36] A. Elgabli, H. Khan, M. Krouka, and M. Bennis, “Reinforcement Learning Based Scheduling Algorithm for Optimizing Age of Information in Ultra Reliable Low Latency Networks,” in *2019 IEEE ISCC*, 2019, pp. 1–6.
- [37] R. Falkenberg and C. Wietfeld, “FALCON: An Accurate Real-Time Monitor for Client-Based Mobile Network Data Analytics,” in *2019 IEEE GLOBECOM*, 2019, pp. 1–7.
- [38] M. E. Haque, F. Tariq, M. R. A. Khandaker, K.-K. Wong, and Y. Zhang, “A Survey of Scheduling in 5G URLLC and Outlook for Emerging 6G Systems,” *IEEE Access*, vol. 11, pp. 34 372–34 396, 2023.

- [39] M. U. A. Siddiqui, H. Abumarshoud, L. Bariah, S. Muhaidat, M. A. Imran, and L. Mohjazi, "URLLC in Beyond 5G and 6G Networks: An Interference Management Perspective," *IEEE Access*, vol. 11, pp. 54 639–54 663, 2023.
- [40] B. S. Khan, S. Jangsher, A. Ahmed, and A. Al-Dweik, "URLLC and eMBB in 5G Industrial IoT: A Survey," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 1134–1163, 2022.
- [41] J. Dai, L. Li, R. Safavinejad, S. Mahboob, H. Chen, V. V. Ratnam, H. Wang, J. Zhang, and L. Liu, "O-RAN-Enabled Intelligent Network Slicing to Meet Service-Level Agreement (SLA)," *IEEE Trans. Mob. Comput.*, vol. 24, no. 2, pp. 890–906, 2025.
- [42] D. Villa *et al.*, "An Open, Programmable, Multi-Vendor 5G O-RAN Testbed with NVIDIA ARC and OpenAirInterface," in *IEEE INFOCOM*, 2024, pp. 1–6.
- [43] D. Villa *et al.*, "X5G: An Open, Programmable, Multi-vendor, End-to-end, Private 5G O-RAN Testbed with NVIDIA ARC and OpenAirInterface," *arXiv*, pp. 1–15, June 2024.
- [44] O-RAN Alliance, "O-RAN E2 Service Model (E2SM) 6.0," *Technical Specification*, no. O-RAN.WG3.E2SM-R004-v06.00, October 2024.
- [45] O-RAN Alliance, "O-RAN E2 Service Model (E2SM) Cell Configuration and Control 4.0," 2024.
- [46] F. Pedersen, R. Högman, M. Buchmayer, and A. Zaid, "5G Spectrum for Local Industrial Networks," <https://www.ericsson.com/497977/assets/local/reports-papers/white-papers/2024/5g-spectrum-for-local-industrial-networks.pdf>, 2024, accessed: February 2025.
- [47] P. S. Upadhyaya, N. Tripathi, J. Gaeddert, and J. H. Reed, "Open AI Cellular (OAIC): An Open Source 5G O-RAN Testbed for Design and Testing of AI-Based RAN Management Algorithms," *IEEE Netw.*, vol. 37, no. 5, pp. 7–15, 2023.
- [48] A. K. Bairagi, M. S. Munir, M. Alsenwi, N. H. Tran, S. S. Alshamrani, M. Masud, Z. Han, and C. S. Hong, "Coexistence Mechanism Between eMBB and uRLLC in 5G Wireless Networks," *IEEE Trans. Commun.*, vol. 69, no. 3, pp. 1736–1749, 2021.
- [49] M. Alsenwi, N. H. Tran, M. Bennis, A. Kumar Bairagi, and C. S. Hong, "eMBB-URLLC Resource Slicing: A Risk-Sensitive Approach," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 740–743, 2019.
- [50] S. F. Abedin, A. Mahmood, N. H. Tran, Z. Han, and M. Gidlund, "Elastic O-RAN Slicing for Industrial Monitoring and Control: A Distributed Matching Game and Deep Reinforcement Learning Approach," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 10 808–10 822, 2022.
- [51] M. Karbalaee Motalleb, V. Shah-Mansouri, S. Parsaefard, and O. L. Alcaraz López, "Resource Allocation in an Open RAN System Using Network Slicing," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 1, pp. 471–485, 2023.
- [52] F. Rezazadeh, L. Zanzi, F. Devoti, H. Chergui, X. Costa-Pérez, and C. Verikoukis, "On the Specialization of FDRL Agents for Scalable and Distributed 6G RAN Slicing Orchestration," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 3473–3487, 2023.
- [53] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "CoO-RAN: Developing Machine Learning-based xApps for Open RAN Closed-loop Control on Programmable Experimental Platforms," *IEEE Trans. Mob. Comput.*, pp. 1–14, 2022.
- [54] W. Zhang, M. Derakhshani, G. Zheng, C. S. Chen, and S. Lambotaran, "Bayesian Optimization of Queuing-Based Multichannel URLLC Scheduling," *IEEE Trans. Wirel. Commun.*, vol. 22, no. 3, pp. 1763–1778, 2023.
- [55] B. Shi, F.-C. Zheng, C. She, J. Luo, and A. G. Burr, "Risk-Resistant Resource Allocation for eMBB and URLLC Coexistence Under M/G/1 Queuing Model," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6279–6290, 2022.
- [56] P. Yang, X. Xi, T. Q. S. Quek, J. Chen, X. Cao, and D. Wu, "RAN Slicing for Massive IoT and Bursty URLLC Service Multiplexing: Analysis and Optimization," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14 258–14 275, 2021.
- [57] R. Fantacci, T. Pecorella, B. Picano, and L. Pierucci, "Martingale Theory Application to the Delay Analysis of a Multi-Hop Aloha NOMA Scheme in Edge Computing Systems," *IEEE/ACM Trans. Netw.*, vol. 29, no. 6, pp. 2834–2842, 2021.
- [58] B. Yu, X. Chi, and H. Sun, "Bandwidth abstraction and instantiation under closed-loop latency constraint for tactile slice based on martingale theory," *Comput. Commun.*, vol. 160, pp. 274–283, 2020.
- [59] I. Hadar, L.-O. Raviv, and A. Leshem, "Scheduling For 5G Cellular Networks With Priority And Deadline Constraints," in *IEEE ICSEE*, 2018, pp. 1–5.
- [60] L.-O. Raviv and A. Leshem, "Joint Scheduling and Resource Allocation for Packets With Deadlines and Priorities," *IEEE Commun. Lett.*, vol. 27, no. 1, pp. 248–252, 2023.
- [61] F. Capozzi, G. Piro, L. Grieco, G. Boggia, and P. Camarda, "Downlink Packet Scheduling in LTE Cellular Networks: Key Design Issues and a Survey," *IEEE Commun. Surv. Tutor.*, vol. 15, no. 2, pp. 678–700, 2013.
- [62] J. Zhang, X. Xu, K. Zhang, B. Zhang, X. Tao, and P. Zhang, "Machine Learning Based Flexible Transmission Time Interval Scheduling for eMBB and uRLLC Coexistence Scenario," *IEEE Access*, vol. 7, pp. 65 811–65 820, 2019.
- [63] A. A. Esswie, K. I. Pedersen, and P. E. Mogensen, "Online Radio Pattern Optimization Based on Dual Reinforcement-Learning Approach for 5G URLLC Networks," *IEEE Access*, vol. 8, pp. 132 922–132 936, 2020.
- [64] M. Alsenwi, N. H. Tran, M. Bennis, S. R. Pandey, A. K. Bairagi, and C. S. Hong, "Intelligent Resource Slicing for eMBB and URLLC Coexistence in 5G and Beyond: A Deep Reinforcement Learning Based Approach," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 7, pp. 4585–4600, 2021.
- [65] O-RAN Working Group 2, "O-RAN AI/ML workflow description and requirements 1.03," July 2021.



slicing, 6G radio access networks (RAN), and deterministic networks, with a focus on mathematical modeling.

Oscar Adamuz-Hinojosa received the B.Sc., M.Sc., and Ph.D. degrees in telecommunications engineering from the University of Granada, Granada, Spain, in 2015, 2017, and 2022, respectively. He was granted a Ph.D. fellowship by the Education Spanish Ministry in September 2018. He is currently an Interim Assistant Professor with the Department of Signal Theory, Telematics, and Communication, University of Granada. He has also been a Visiting Researcher at NEC Laboratories Europe on several occasions. His research interests include network slicing, 6G radio access networks (RAN), and deterministic networks, with a focus on mathematical modeling.



Lanfranco Zanzi received the B.Sc. and M.Sc. degrees in telecommunication engineering from the Polytechnic of Milan, Italy, in 2014 and 2017, respectively, and the Ph.D. degree from the Technical University of Kaiserslautern, Germany, in 2022. He works as a Senior Research Scientist with NEC Laboratories Europe. His research interests include network virtualization, machine learning, blockchain, and their applicability to 5G and 6G mobile networks.



Vincenzo Sciancalepore (M'15–SM'19) received his M.Sc. degree in Telecommunications Engineering and Telematics Engineering in 2011 and 2012, respectively, whereas in 2015, he received a double Ph.D. degree. Currently, he is a Principal Researcher at NEC Laboratories Europe, focusing his activity on reconfigurable intelligent surfaces. He is an Editor of the *IEEE Transactions on Wireless Communications* (since 2020) and *IEEE Transactions on Communications* (since 2024).



Xavier Costa-Pérez (M'06–SM'18) is a Research Professor in ICREA, Scientific Director at the i2Cat Research Center and Head of 5G/6G Networks R&D at NEC Laboratories Europe. He has served on the Organizing Committees of several conferences, published papers of high impact, and holds tens of granted patents. Xavier received his Ph.D. degree in Telecommunications from the Polytechnic University of Catalonia (UPC) in Barcelona and was the recipient of a national award for his Ph.D. thesis.