Intelligent Dynamic Resource Allocation and Puncturing for Next Generation Wireless Networks

Walaa AlQwider, Aly Sabri Abdalla, Talha Faizur Rahman, and Vuk Marojevic

***Abstract*—As we progress from 5G to emerging 6G wireless, the spectrum of cellular communication services is set to broaden significantly, encompassing real-time remote healthcare applications and sophisticated smart infrastructure solutions, among others. This expansion brings to the forefront a diverse set of service requirements, underscoring the challenges and complexities inherent in next-generation networks. In the realm of 5G, Enhanced Mobile Broadband (eMBB) and Ultra-Reliable Low-Latency Communications (URLLC) have been pivotal service categories. As we venture into the 6G era, these foundational use cases will evolve and embody additional performance criteria, further diversifying the network service portfolio. This evolution amplifies the necessity for dynamic and efficient resource allocation strategies capable of balancing the diverse service demands. In response to this need, we introduce the Intelligent Dynamic Resource Allocation and Puncturing (IDRAP) framework. Leveraging Deep Reinforcement Learning (DRL), IDRAP is designed to balance between the bandwidth-intensive requirements of eMBB services and the latency and reliability needs of URLLC users. The performance of IDRAP is evaluated and compared against other resource management solutions, including Intelligent Dynamic Resource Slicing (IDRS), Policy Gradient Actor-Critic Learning (PGACL), System-Wide Tradeoff Scheduling (SWTS), Sum-Log, and Sum-Rate.The results show an improved Service Satisfaction Level (SSL) for eMBB users while maintaining the essential SSL threshold for URLLC services.**

***Index Terms*—Deep learning, eMBB, fairness, puncturing, Q-learning, reinforcement learning, scheduling, throughput, URLLC.**

I. Introduction

The rapid technological advancements in the cellular communication industry have led to a paradigm shift, particularly with the advent of 5th generation (5G) networks and the research on 6G technologies [1]. Next-generation networks are expected to not only enhance and extend the capabilities established by 5G but also introduce new functionalities and services. These include but are not limited to, real-time remote healthcare, autonomous cyber-physical systems, industry X.0, space connectivity, smart infrastructure, and environment management [2]. Such service diversity, which comes with unique requirements, highlights the complexity and challenges for designing and operating advanced wireless networks. The International Telecommunication Union (ITU) has categorized the services to be supported by 5G into three main categories: Enhanced Mobile Broadband (eMBB), Massive Machine Type Communications (mMTC), and Ultra-Reliable and Low Latency Communication (URLLC). Each of these service categories satisfies distinct connectivity needs ranging from high data rates in human-centric applications to mission-critical reliability and low latency [3]–[7]. These categories are set to evolve into further-eMBB (feMBB), ultra-mMTC (umMTC), and enhanced-URLLC (eURLLC) for enabling more sophisticated and complex operating scenarios [8].

The coexistence of diverse service types supported by the same network infrastructure poses a significant resource allocation challenge. This challenge is particularly pronounced between eMBB and URLLC services due to their conflicting demands: eMBB requires high throughput for data-intensive applications while URLLC necessitates immediate resource access to meet strict latency and reliability targets for critical tasks. mMTC generally refers to connecting a vast number of low-power, low-data-rate devices. However, certain mMTC devices may have service requirements that overlap with those of eMBB or URLLC, such as in industrial IoT. Industrial monitoring systems may collect and share large amounts of real time data, whereas industrial control systems may need to react quickly to optimize production and minimize disruption. This research focuses on the coexistence of high throughput and low-latency services and does not exclude future applications of MTCs. Without loss of generality and to be in alignment with current literature, we refer to the two service classes with conflicting resource allocation requirements as eMBB and URLLC

The authors are with the Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762 USA (e-mail: wq27@msstate.edu; asa298@msstate.edu; tfr42@msstate.edu; vm602@msstate.edu),(Corresponding author: Walaa AlQwider).

in this paper. Operators face the dilemma of satisfying the quality of service (QoS) requirements of URLLC users, which inadvertently impacts the resources available to eMBB users. The stochastic nature of URLLC, characterized by unpredictable traffic arrivals and the necessity to prioritize its demands due to strict latency and reliability requirements, further complicates resource allocation strategies. The Third Generation Partnership Project (3GPP) has therefore introduced two techniques, orthogonal slicing [9] and punctured scheduling [6], [10], which aim to effectively integrate eMBB and URLLC traffic within 5G New Radio (NR). These methods continue to be significant in the 5G Advanced framework and will gain prominence and evolve in 6G communication contexts.

Orthogonal slicing employs reservation-based scheduling to allocate a portion of the available bandwidth exclusively for URLLC traffic. This bandwidth can be reserved either statically or dynamically, each method bringing its own set of challenges. Specifically, dynamic reservation can lead to control signaling overheads, while both forms of reservation may result in underutilization of resources if the reserved bandwidth for URLLC remains unused in the absence of URLLC traffic. On the other hand, resource puncturing, allocates all available resources to eMBB users and accommodates URLLC demands by puncturing the ongoing eMBB transmissions, thereby avoiding the performance degradation and resource wastage of orthogonal slicing when URLLC traffic is absent. However, this approach introduces a potential risk to the service satisfaction levels (SSL) of eMBB users due to the aggressive puncturing of resources. Therefore, the challenge lies in optimally distributing punctured resources among eMBB users to minimize the impact of URLLC puncturing on eMBB SSL while simultaneously satisfying the stringent SSL requirements of URLLC services.

The development of a dynamic radio resource allocation framework that utilizes either orthogonal slicing or resource puncturing techniques to facilitate the simultaneous operation of eMBB and URLLC services within the same network is significant area of focus in contemporary research. This effort to strike a balance in the SSL of both service types becomes even more critical in the context of 5G advancement and the transition to 6G. The anticipated increase in URLLC device usage, particularly in scenarios characterized by a massive deployment of URLLC services (mURLLC) in a beyond 5 G/6 G scenario, highlights the urgent need for an effective radio resource allocation strategy. Such a strategy must adeptly accommodate the varied needs of all service classes, ensuring operational efficiency and the conservation of network resources [11], [12].

In this paper, we study the challenge of radio resource allocation for eMBB and URLLC services sharing the same network infrastructure, utilizing 3GPP's mini-slot based transmission and resource puncturing techniques. Our approach to resource allocation determines the bandwidth and transmission power allocation to eMBB users on time slots, whereas URLLC resource puncturing and power allocation occur at the granularity of mini-slots. To tackle this complex multi-timescale problem, we introduce a hierarchical deep reinforcement learning (DRL) framework comprising two specialized agents: the eMBB scheduling agent and the URLLC resource puncturing agent. The core contributions of this paper are outlined as follows:

- We formulate mixed-integer non-linear programming (MINLP) problems for both eMBB resource allocation and URLLC resource puncturing. The objectives are maximizing the eMBB sum data rate while meeting the minimum eMBB data rate requirements as well as minimizing the impact of eMBB data rate loss due to puncturing while satisfying the URLLC latency and reliability demands.
- We design a Markov Decision Process (MDP) for each problem. Actions for each problem are aligned with the constraints of their respective optimization problems, and the proposed reward function features penalty parameters to address QoS violations for each user type.
- We propose an Intelligent Dynamic Resource Allocation and Puncturing (IDRAP) framework based on DRL for eMBB scheduling and URLLC puncturing. It integrates two distinct DRL agents: the eMBB scheduling agent and the URLLC puncturing agent.
- We evaluate the proposed IDRAP framework for various penalty weights and traffic loads to assess the impact of these parameters on the AI model performance. SSL metrics for eMBB and URLLC traffic are captured in the numerical analysis and compared against those of established benchmarks and equivalent DRL based solutions. The results show the robustness of IDRAP for effectively balancing radio resources across heterogeneous services.

The rest of the paper is organized as follows: Section II summarizes the prior art. Sections III and IV present the system model and problem formulation. Section V introduces the design and operating principles of the proposed IDRAP framework. Section VI provides numerical results and performance evaluation. Section VII draws the conclusions.

TABLE I: Comparison of related work and the proposed work in this paper for eMBB and URLLC coexistence.

	Related Work																
	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	[24]	[25]	[26]	[27]	[28]	Proposed
Resource Puncturing	×	×	×	×	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓
Orthogonal Slicing	✓	✓	✓	✓	✓	✓	✓	✓	×	×	×	×	×	×	×	×	×
URLLC Latency	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
URLLC Reliability	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	✓
eMBB Min Data Rate	✓	×	×	✓	✓	✓	✓	✓	×	×	×	✓	×	×	✓	✓	✓
Power Alloc (eMBB)	✓	✓	×	×	×	×	×	×	×	×	✓	×	✓	✓	✓	×	✓
Power Alloc. (URLLC)	✓	✓	✓	×	×	×	×	×	×	×	×	×	×	✓	✓	×	✓
Resource Alloc. (eMBB)	✓	✓	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓
Instantaneous Channel State (eMBB)	✓	✓	✓	✓	NA	NA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×
Instantaneous Channel State (URLLC)	✓	✓	✓	✓	NA	NA	✓	✓	NA	NA	✓	✓	✓	✓	✓	✓	×

II. Related Work

The challenges associated with enabling the coexistence of eMBB and URLLC services in 5G NR have become a focal point for researchers in recent years. Designing a resource model incorporating orthogonal slicing and resource puncturing poses a complex problem, where various factors influence the performance in terms of meeting the QoS requirements for both URLLC and eMBB services. In this regard, orthogonal resource slicing for interlacing eMBB and URLLC traffic is proposed in [13]. The authors devise a resource optimization algorithm based on the sample average approximate technique and a distributed optimization method; thus, the algorithm maximizes the long-term total slice utility while considering the total transmission power and network bandwidth. Similarly, [14] studies a mixed-integer nonlinear programming (MINLP) problem with a specific focus on addressing eMBB and URLLC services for remote radio heads in 5G cloud radio access networks (C-RAN). The authors of [15] introduce a dynamic framework for bandwidth allocation through orthogonal slicing for eMBB and URLLC traffic. The URLLC service quality is defined in terms of queuing latency and eMBB service quality as the sum data rate. The original optimization problem, which is formulated as a MINLP problem, is addressed through the Lyapunov optimization, breaking it into two computationally efficient sub-problems that are solved separately in a sub-optimal manner.

In recent years, deep reinforcement learning (DRL) has been proposed for tackling the coexistence between URLLC and eMBB traffic in 5G networks [29], [30]. Researchers leverage the ability of DRL to learn from experiences and adjust to evolving network conditions. The authors of [16] employ a combination of reinforcement learning (RL) and long short-term memory (LSTM) for radio access network (RAN) slicing. An LSTM is utilized to predict the network traffic for the upcoming prediction windows (PWs) within each slice and RL to allocate resources to different slices in the current PW. In the related study [17], Deep Q-learning (DQL) is proposed for demand-aware resource allocation to network slices. The DQL is embedded into an intelligent controller xApp in Open RAN (O-RAN), which, based on the key performance metrics and available resources, selects the best-performing scheduling policy for each RAN slice and the number of resource blocks (RBs) allocated to each slice. The authors of [18] propose DeepSlicing based on alternating direction method of multipliers and DRL to address the diverse resource demands and performance metrics of each network slice. The problem is decomposed into a master problem solved through convex optimization and several slave problems handled by DRL. The goal is to learn the optimal resource allocation policies in the absence of closed-form utility functions, ensuring efficient network slicing.

The work presented in [19] introduces a two-tiered resource allocation system that employs a multi-agent DQL. In the first tier, a software defined network controller dynamically distributes RBs from a shared RB pool to multiple gNodeBs based on their reported demands. The second tier operates at a finer time scale, where each gNodeB allocates its assigned RBs among the associated eMBB and URLLC users to fulfill their specific QoS needs in terms of data rate and latency. A RAN slicing approach is proposed in [20] for high mobility users, leveraging a blend of Double Deep Q-Networks (DDQN), Dueling DQN, and action factorization. The authors integrate networks into their D3QN design to predict future channel states and enhance decision-making for resource slicing. The optimization focuses on optimizing user quality of experience (QoE), balancing data rate and latency.

Resources puncture has been utilized in different work in the literature. The study presented in [21] model the eMBB rate loss resulting from the dynamic puncturing for URLLC traffic in three different loss model:

linear, convex and threshold models. They adopt the linear loss model with introducing resource-proportional puncturing, where the resources punctured from eMBB users are proportional to their allocated resources for fair resource sharing. Reference [22] studies a risk-sensitive optimization framework that incorporates Conditional Value at Risk to minimize the risk on SSL of eMBB users caused by the resource puncturing for URLLC traffic. The work in [23] presents a null-space based spatial preemptive scheduler designed for joint URLLC and eMBB traffic. This scheduler targets cross-objective optimization, ensuring the fulfillment of URLLC's latency requirements while simultaneously maximizing eMBB's ergodic capacity. The research presented in [24] formulates the joint resource allocation of eMBB and URLLC resources as a utility maximization optimization problem, aiming at maximizing the minimum expected eMBB data rate. The optimization considers the rate loss incurred due to URLLC puncturing while meeting the strict URLLC latency requirements. The proposed solution decomposes the problem into two sub-problems, employing the penalty successive upper bound minimization for eMBB resources and the optimal transportation model for resource puncturing. The work presented in [25] addresses the dynamic multiplexing challenges of 5G, presenting a two-phase framework. The first phase employs the Decomposition and Relaxation Resource Allocation, utilizing an exponential utility function for the RBs and power allocation to eMBB users. In the second phase, a Policy Gradient Actor-Critic Learning (PGACL) algorithm handles the URLLC resource allocation dynamics. The integrated approach aims at maximizing the average data rate of eMBB users while minimizing the variance in data rates and satisfying the URLLC latency constraint.

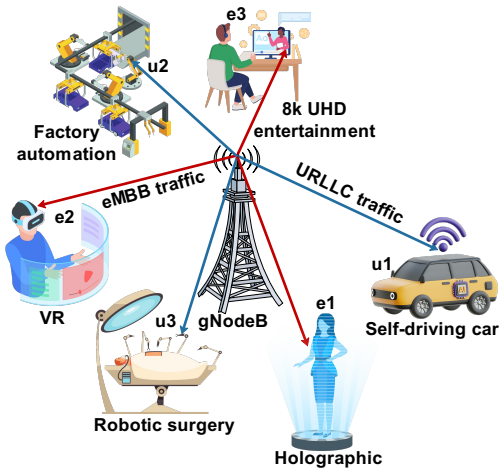
The authors of [26] employ Q-learning with co-training for the RB allocation to eMBB users. Subsequently, a Double Deep Q-Network (DDQN) is leveraged for handling URLLC scheduling over punctured eMBB slots with the goal of minimizing eMBB's data rate losses and satisfying URLLC's latency requirements. The research described in [27] addresses the coexistence of eMBB and URLLC services in a wireless powered communication network. It employs preemptive puncturing for URLLC traffic on eMBB transmissions, formulating an optimization problem to maximize the uplink eMBB sum rate. The considerations include the URLLC latency demand, the radio frequency, the user's battery capacity, and the subcarrier availability. A mixed-deep deterministic policy gradient (DDPG) approach is proposed to decompose the problem into discrete subproblems for subcarrier allocation and continuous subproblems for time and energy allocation, solving them alternately. The authors of [28] address the tradeoff between URLLC and eMBB QoS requirements in 5G networks. The proposed system optimizes the bandwidth allocation and resolves overlapping positions of URLLC users' traffic through a DDPG algorithm, considering channel variations and URLLC traffic arrivals. Table I compares our contribution to prior art in terms of resource management strategy, QoS constraints for each service category, optimization parameters, and whether real-time channel quality information is incorporated.

References [13]–[20] employ orthogonal slicing, often criticized for suboptimal resource utilization due to the sporadic nature of URLLC traffic. Additionally, these works predominantly focus on bandwidth allocation while neglecting power allocation. Conversely, references [21]–[28] implement resource puncturing to address the shortcomings of orthogonal slicing. These solutions come with their own limitations: References [21], [22] tend to treat URLLC traffic uniformly as opposed to addressing practical user and channel variations. While researches in [23]–[28] account for URLLC traffic load and channel variations, they rely on perfect and immediate knowledge of these dynamics, which is not feasible considering the unpredictable and bursty nature of URLLC service demands. References [21], [23], [26] primarily aim at maximizing eMBB data rates subject to resource puncturing without considering transmission reliability or minimum data rate requirements. Prior research focuses on meeting the URLLC latency requirements, overlooking the critical aspect of reliability. It is essential to ensure reliability by determining the necessary transmission power to maintain the signal-to-noise ratio (SNR) above a set threshold and avoid excessive packet error rates.

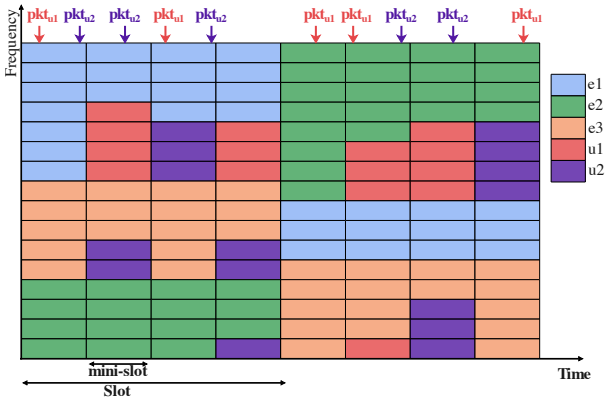
In response to these challenges, our proposed work adopts a more individualized approach to URLLC traffic, accounting for variable packet arrivals and channel conditions. We integrate URLLC reliability constraints into our optimization problem, ensuring a balanced consideration with the minimum data rate requirements of eMBB services while maximizing the system data rate. Our method dynamically allocates RBs and power to both eMBB and URLLC users based on prior channel conditions, a practical approach that reflects the inherent delays in acquiring and using such measurements in real-world scenarios.

III. System Model

We consider the downlink of an orthogonal frequency division multiple access (OFDMA) system where a gNodeB serves a set of eMBB users indexed by $\mathcal{E} = \{1, ..., e, ...E\}$ and a set of URLLC users indexed by $\mathcal{U} = \{1, ..., u, ...U\}$. Each eMBB user has a minimum data rate requirement R_e^{min} whereas URLLC users have a maximum delay requirement d_{max}. The available



(a)



(b)

Fig. 1: Coexistence of eMBB and URLLC traffic in a 5G network scenario: (a) System model depicting eMBB and URLLC traffic handled by a single gNodeB, (b) Illustration of the resource puncturing method for URLLC packet arrivals, demonstrating dynamic resource allocation.

bandwidth is evenly partitioned into M RBs, where each RB covers $B = 12$ consecutive subcarriers. The subcarrier spacing (SCS) and the RB bandwidth depend on the employed numerology μ. The time domain is partitioned into F equally spaced time frames denoted as $\mathcal{F} = \{1, 2, \ldots, f, \ldots, F\}$, where each time frame $f \in \mathcal{F}$ spans T time slots indexed by $\mathcal{T} = \{1, 2, \ldots, t, \ldots, T\}$. Each time slot covers 14 consecutive OFDM symbols. The value of T depends on the numerology index. A further subdivision occurs within a slot, yielding K mini-slots indexed by $\mathcal{K} = \{1, 2, \ldots, k, \ldots, K\}$. Each mini-slot has a duration of τ.

During each Transmission Time Interval (TTI), which corresponds to one time slot, the gNodeB allocates the available RBs to eMBB users to transmit their data. RBs are allocated to URLLC users on mini-slot basis, ensuring the prompt transmission of URLLC packets upon their arrival. This eliminates the need to wait until the end of the time slot for scheduling URLLC users.

The peak data rate of eMBB user $e \in \mathcal{E}$ in time slot t is defined as

$$R_e(t) = m_e(t) \cdot B \log_2 \left(1 + \frac{p_e(t) h_e(t) d_e^{-\alpha}(t)}{\sigma^2}\right), \quad (1)$$

where $p_e(t)$ and $m_e(t)$ represent the downlink transmission power and the number of RBs used by the gNodeB to transmit data to user e during time slot t, respectively, $h_e(t)$ denotes the small-scale channel fading factor, d_e captures the distance between the gNodeB and user e, α indicates the path loss exponent, and σ^2 represents the random Gaussian noise power. In our current system model, we consider the scenario of a single gNodeB, assuming that inter-cell interference is effectively managed by existing means or by allocating different frequency bands to neighboring gNodeBs. We thus assume that the impact of inter-cell interference is negligible for the purposes of this study. We will consider a dense cellular network model employing frequency reuse of one for developing and evaluating a multi-cell, multi-agent counterpart to IDRAP in future work.

The primary objective of eMBB services is here to achieve highest data rates while meeting the minimum data rate requirement for each user. We therefore introduce

$$\Delta_e(t) = \max(R_e^{min} - R_e(t), 0), \quad (2)$$

which quantifies the violation of not meeting the minimum data rate requirement for an eMBB user. The SSL of eMBB services is then defined as

$$\rho^{eMBB}(t) = \frac{1}{E} \sum_{e \in \mathcal{E}} \mathbb{1}(\Delta_e(t) = 0), \quad (3)$$

where $\mathbb{1}(x)$ is the indicator function that equals 1 if there is no violation to condition x, and 0 otherwise.

In addition to meeting the minimum data rate requirements of eMBB services, it is imperative to fulfill the reliability and latency prerequisites of URLLC services. Let $q_u^t(k)$ denote the traffic demand of user $u \in \mathcal{U}$ in mini-slot $k \in \mathcal{K}$ of time slot $t \in \mathcal{T}$. The latency requirement for URLLC user u in this mini-slot is satisfied if

$$R_u^t(k) \geq \frac{q_u^t(k)}{\tau}. \quad (4)$$

It establishes that the achieved data rate $R_u^t(k)$ must exceed the amount of data at the gNodeB awaiting transmission to this user in mini-slot k to meet the latency requirement τ.

Due to finite blocklength coding in URLLC, the short-packet transmission regime is used to approximate

$R_u^t(k)$ [31], [32] as

$$R_u^t(k) = \frac{m_u^t(k) \cdot B}{\ln 2} \left[\ln\left(1+\gamma_u^t(k)\right) - \sqrt{\frac{V_u^t(k)}{\tau m_u^t(k) f_b}} Q^{-1}(\epsilon) \right], \tag{5}$$

where

$$V_u^t(k) = 1 - \frac{1}{(1+\gamma_u^t(k))^2}, \tag{6}$$

and

$$\gamma_u^t(k) = \frac{p_u^t(k) h_u^t d_u^{-\alpha}(t)}{\sigma^2}. \tag{7}$$

Expression $m_u^t(k)$ represents the number of RBs allocated to URLLC user u in mini-slot k of time slot t. The SNR $\gamma_u^t(k)$ at URLLC user u considers the allocated transmission power $p_u^t(k)$, the small-scale channel fading factor h_u^t, and distance $d_u(t)$ to the gNodeB. Function $Q^{-1}(.)$ is the inverse of the Gaussian Q-function, whereas ϵ is the decoding error probability.

The allocated power to URLLC user u should result in an SNR greater than the threshold SNR γ_{min} [33] to ensure a reliability of $1-\epsilon$ for the downlink transmission at rate $R_u^t(k)$. In other words, if the allocated transmission power meets the SNR threshold, the expected rate of user u is calculated using (5); otherwise, the expected rate for user u is 0.

The primary goal of the URLLC scheduler is to allocate RBs and power to URLLC users requesting service in a given mini-slot so as to meet the given latency and reliability requirements. We introduce

$$\Delta_u^t(k) = \begin{cases} \max\left(\frac{q_u^t(k)}{\tau} - R_u^t(k), 0\right), & \text{if} \gamma_u^t(k) \geq \gamma^{min}, \\ \frac{q_u^t(k)}{\tau}, & \text{otherwise} \end{cases} \tag{8}$$

to measure the delay and reliability violation for URLLC user u in mini-slot k of time slot t. This user fulfills both reliability and latency requirements if $\Delta_u^t(k) = 0$. The SSL for URLLC services is then calculated as

$$\rho^{URLLC}(t) = \frac{1}{U \cdot K} \sum_{k \in \mathcal{K}} \sum_{u \in \mathcal{U}} \mathbb{1}(\Delta_u^t(k) = 0). \tag{9}$$

It indicates the proportion of URLLC users for whom both latency and reliability requirements have been met during time slot t.

IV. Problem Formulation

We employ the dynamic resource puncturing scheme to address the coexistence challenge between URLLC and eMBB users. The optimization problem is formulated with the primary focus on the allocation of RBs and transmission power. The objective is maximizing eMBB user data rates while ensuring that the minimum SSL thresholds are met for both URLLC and eMBB services.

For dynamic resource puncturing, the gNodeB initially allocates up to M available RBs to eMBB users at the beginning of a time slot. In case of a full resource allocation, the gNodeB interrupts ongoing eMBB transmissions to promptly serve URLLC requests within their mini-slots by reducing the transmission power to zero for the scheduled eMBB resources that are punctured. This allows the gNodeB to allocate the punctured resources to URLLC users. As a consequence, the punctured eMBB users experience a reduction in their data rates during the mini-slots where URLLC traffic is present. Two schedulers are employed in this scheme: the eMBB scheduler and the URLLC scheduler. The eMBB scheduler aims at maximizing the average data rate and fulfilling the SSL threshold for eMBB users. It achieves this by allocating the available bandwidth M and power P_E to eMBB users in every time slot. The optimization problem for

TABLE II: List of key notations

Notation	Description
$\mathcal{E}$	Set of eMBB users indexed by e
$\mathcal{U}$	Set of URLLC users indexed by u
R_e^{min}	Minimum data rate for eMBB user e
R_e	Data rate of eMBB user e
d_{max}	Maximum delay for URLLC users
M	Total number of RBs
μ	Numerology index
SCS	Subcarrier spacing
B	Bandwidth of each Resource Block (RB)
$\mathcal{F}$	Set of frames indexed by f
$\mathcal{T}$	Set of time slots in each frame
$\mathcal{K}$	Set of mini-slots in each time slot
τ	Time duration of one mini-slot
P_E	Transmission power for eMBB users
P_U	Transmission power for URLLC users
m_e	Number of RBs allocated to user e
p_e	Transmission power allocated to user e
q_u	Traffic demand of user u
R_u	Data rate of user u
m_u	Number of RBs allocated to user u
p_u	Transmission power allocated to user u
m_e^u	Number of RBs punctured by user u
pun_e	Number of RBs punctured from user e
R_e^l	Data rate reduction of user e
α	Path loss exponent
σ^2	Noise power
ϵ	Decoding error probability for URLLC
$\gamma^{\min}$	SNR threshold for URLLC users
ρ^{eMBB}	SSL for eMBB users
ρ^{URLLC}	SSL for URLLC users
ζ^{eMBB}	Penalty weight for eMBB users
ζ^{URLLC}	Penalty weight for URLLC users

the eMBB scheduler is then formulated as

$$\begin{aligned}\mathbf{P1}: \max_{\mathbf{M}_E(t),\mathbf{P}_E(t)} & \frac{1}{T}\sum_{t\in\mathcal{T}} R_e(t) \\ \text{s.t.} \quad C1: & \ \frac{1}{T}\sum_{t\in\mathcal{T}} \rho^{\text{eMBB}}(t) \geq \rho^{\text{eMBB}}, \\ C2: & \ \sum_{e\in\mathcal{E}} m_e(t) \leq M, \\ C3: & \sum_{e\in\mathcal{E}} p_e(t) \leq P_E.\end{aligned} \tag{10}$$

Vectors $\mathbf{M}_E(t) = [m_1(t), ..., m_e(t), ...m_E(t)]$ and $\mathbf{P}_E(t) = [p_1(t), ..., p_e(t), ...p_E(t)]$ capture the RB and power allocation for eMBB users in time slot t. Constraint $C1$ ensures that the eMBB SSL threshold is met, whereas Constraints $C2$ and $C3$ ensure that the sum of RBs and power allocated to eMBB users do not exceed the available RBs and transmission power, respectively.

The URLLC scheduler needs to be designed to meet the SSL threshold for URLLC users while minimizing the reduction of data rates of eMBB users caused by puncturing. This is achieved by determining for each URLLC user to be scheduled in a given mini-slot $k \in \mathcal{K}$ of time slot $t \in \mathcal{T}$ the number of RBs that URLLC user $u \in \mathcal{U}$ punctures from eMBB user $e \in \mathcal{E}$ as well as the transmission power allocated to this user.

Let $m_e^{u,t}(k)$ denote the number of RBs punctured by URLLC user u from eMBB user e in mini-slot k of time slot t, and let $pun_e(k) = \sum_{u\in\mathcal{U}} m_e^{u,t}(k)$ denote the total number of RBs punctured from eMBB user e in mini-slot k. The data rate reduction of eMBB user e due to RB puncturing then becomes [21]

$$R_e^l(k) = R_e(t) \cdot \left(\frac{pun_e(k)}{m_e(t)\cdot K}\right). \tag{11}$$

The above equation quantifies the impact of RB puncturing on the data rate of eMBB user e in time slot t.

The optimization problem for URLLC scheduling can then be formulated as

$$\begin{aligned}\mathbf{P2}: \min_{\mathbf{p}_U^t,\mathbf{M}_E^{U,t}} & \sum_{e\in\mathcal{E}} R_e^l \\ \text{s.t.} \quad C4: & \ \frac{1}{T\cdot K}\sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}} \rho^{\text{URLLC}}(k) \geq \rho^{\text{URLLC}}, \\ C5: & \ \frac{1}{T}\sum_{t\in\mathcal{T}} \rho^{\text{eMBB}}(t) \geq \rho^{\text{eMBB}}, \\ C6: & \ \sum_{u\in\mathcal{U}} P_u^t(k) \leq P_U, \\ C7: & \ \sum_{u\in\mathcal{U}} m_e^{U,t}(k) \leq m_e(t).\end{aligned} \tag{12}$$

Vector $\mathbf{p}_U^t$ captures the power allocations and $\mathbf{M}_U^{E,t}$ represents the RB puncturing matrix for URLLC users in mini-slot k. Constraints $C4$ and $C5$ ensure that the SSL threshold of URLLC and eMBB services are met. Constraint $C6$ ensures that the total power allocated to URLLC users is less than P_U. Constraint $C7$ limits the number of RBs punctured from eMBB user e to be equal or less than the RBs allocated to that user.

V. Intelligent Scheduling Framework

The optimization problems $P1$ and $P2$ are classified as MINLP problems because they incorporate both discrete (integer) and continuous variables within the context of nonlinear objectives and constraints [14], [26]. Specifically, the complexity in $P1$ arises from the requirement to balance the discrete nature of RB allocation, an integer decision, with the continuous allocation of power, alongside optimizing a nonlinear objective function that involves the sum data rate, which itself is a function of power and RB allocation. Problem $P2$ introduces integer decisions in the form of RB puncturing from eMBB users to accommodate URLLC transmissions as well as continuous decisions regarding URLLC transmit power allocation. It aims at minimizing an objective function that is inherently non-linear because of its dependence on transmission rates and power levels. The integer variables introduce combinatorial complexity, significantly expanding the solution space and complicating the search for optimal solutions. The nonlinear aspects of the $P1$ and $P2$ objectives arise from the relationships between data rates, RBs, and power allocations. Together, these characteristics render $P1$ and $P2$ as NP-hard, indicating the absence of polynomial-time solutions.

In response to this complexity, we adopt a unified approach by modeling each problem as an MDP and leverage DRL, specifically the DDPG method to learn effective policies for optimizing the parameters of each scheduler with the goal of maximizing the long-term objective functions. DRL is particularly suited for addressing such complex decision-making problems where direct optimization is impractical, offering a viable path to achieving near-optimal solutions in a dynamic and stochastic environment like the one considered in our work.

A. Markov Decision Process Design

DRL empowers an agent to acquire optimal strategies to maximize a long-term reward function by interacting with its environment. The environment is modeled as an MDP, offering a mathematical framework for decision-making problems where outcomes are stochastic and controlled by an agent. The MDP comprises the tuple $(\mathcal{S}, \mathcal{A}, r, \gamma)$, where $\mathcal{S}$ is the state space containing environmental information, $\mathcal{A}$ is the action space defining

feasible actions, r represents the instantaneous reward function, and γ serves as the discount factor balancing between immediate and long-term rewards. At each time step t, the agent observes state s_t from $\mathcal{S}$, takes action a_t from $\mathcal{A}$, observes the new state s_{t+1}, and receives reward $r(s_t,\ a_t)$. In continuation, we formulate the state, the action, and the reward function for the eMBB scheduling and for the URLLC puncturing problems.

1) eMBB Scheduling MDP: The eMBB scheduling agent employs a dedicated DDPG agent. This agent is tasked to dynamically allocate the M RBs and P_E power that are available to eMBB users. The primary objective is maximizing the data rates of eMBB users, ensuring that each user meets the minimum data rate requirement which is captured by the SSL ρ^{eMBB}.

The state of the eMBB scheduler is formulated as

$$s_E(t) = [\gamma_e(t-1), R_e(t-1), x_e(t-1)], \forall e \in \mathcal{E}, \quad (13)$$

where $\gamma_e(t-1)$ represents the SNR of eMBB user e in the previous time slot, $R_e(t-1)$ denotes the previous data rate of user e, and $x_e(t-1)$ is a binary indicator signaling whether the user fulfills the minimum data rate requirement: $x_e(t-1) = 1$ if $\Delta_e(t-1) = 0$.

The agent's actions in time slot $t \in \mathcal{T}$ follow

$$\begin{aligned} &a_E(t) = [\alpha_e^M(t), \alpha_e^P(t)], \forall e \in \mathcal{E}, \\ &\textstyle\sum_{e\in\mathcal{E}} \alpha_e^M(t) = 1, \\ &\textstyle\sum_{e\in\mathcal{E}} \alpha_e^P(t) = 1. \end{aligned} \quad (14)$$

Parameters $\alpha_e^M(t)$ and $\alpha_P^e(t)$ represent the percentages of the M RBs and P_E transmission power allocated to eMBB user e in time slot t. The actually allocated RB and transmission power resources,

$$\begin{aligned} m_e(t) &= \lceil \alpha_e^M(t) \cdot M \rfloor, \\ p_e(t) &= \lceil \alpha_e^P(t) \cdot P_E \rfloor, \end{aligned} \quad (15)$$

are the result of the eMBB scheduling action, the notation $\lceil x \rfloor$ refers to rounding of x to the nearest integer.

The reward function

$$R_E(t) = \sum_{e\in\mathcal{E}} \left[R_e(t) - \zeta^E \cdot (1 - x_e(t)) \right] \quad (16)$$

applies penalty value ζ^E for every eMBB user whose minimum data rate requirement is violated.

2) URLLC Puncturing MDP: The URLLC puncturing agent aims at fulfilling the URLLC reliability and latency requirements while minimizing the impact on eMBB data rates caused by the puncturing process. Its state vector is defined as

$$\begin{aligned} s_U(k) = [&\gamma_u(t-1), q_u(k), x_u^\gamma(k-1), x_u^L(k-1), \\ &m_e(t), R_e^l(k-1), \delta_e(k-1)], \\ &\forall u \in \mathcal{U}, \forall e \in \mathcal{E} \end{aligned} \quad (17)$$

the first four elements are related to URLLC users and the remaining elements provide information about eMBB users, allowing the puncturing agent to assess the impact of puncturing on eMBB users up to mini-slot k. Parameter $m_e(t)$ represents the number of RBs allocated to eMBB user e, $R_e^l(k-1) = \sum_{n=1}^{k-1} R_e^{\mathrm{l}}(n)$ is the aggregated data loss of eMBB user e up to mini-slot $k-1$, and $\delta_e(k-1) = \max((R_e(t) - R_e^l(k-1)) - R_e^{\min}, 0)$ quantifies how far user e is from violating its data rate requirement. The data rate requirement is violated if $\delta_e(k-1) = 0$.

The action vector of the URLLC puncturing agent is

$$\begin{aligned} a_U(t) = &[\alpha_e^{pun}(k), \alpha_e^u(k), \alpha_u^P(k)], \forall u \in \mathcal{U}, \forall e \in \mathcal{E}, \\ &\alpha_e^{pun}(k) \le 1, \\ &\sum_{u\in\mathcal{U}} \alpha_e^u(k) = 1, \\ &\sum_{u\in\mathcal{U}} \alpha_u^P(t) = 1, \end{aligned} \quad (18)$$

where $\alpha_e^{pun}(k)$ denotes the percentage of RBs allocated to user e that are available for puncturing in mini-slot k. Parameters $\alpha_e^u(k)$ and $\alpha_u^P(k)$ are percentage values that are used to calculate the actual values of puncturing matrix $\mathbf{M}_U^{E,t}(k) = [m_1^1(k), ..., m_1^u(k), ..., m_e^u(k), ...m_E^U(k)]$ and power allocation vector $\mathbf{P}_U(k) = [p_1(k), p_2(k), ..., p_U(k)]$ as

$$\begin{aligned} pun_e(k) &= \lceil \alpha_e^{pun}(k) \cdot m_e(t) \rfloor, \\ m_e^u(k) &= \lceil \alpha_e^u(k) \cdot pun_e(t) \rfloor, \\ p_u(k) &= \lceil \alpha_u^P(k) \cdot P_U \rfloor. \end{aligned} \quad (19)$$

The reward function,

$$\begin{aligned} R_U(k) = -\Big(&\sum_{u\in\mathcal{U}} \zeta^U \cdot [x^\gamma u(k) + x^L u(k)] + \\ &\sum_{e\in\mathcal{E}} \cdot \sum_{k\in\mathcal{K}} \zeta^E R_e^{\mathrm{loss}}(k) \Big), \end{aligned} \quad (20)$$

is designed as a penalty, which increases with ζ^U for each URLLC user's violation of its reliability and latency requirements and the rate losses of eMBB users weighted by ζ^E.

B. IDRAP

We design a hierarchical intelligent resource allocation and puncturing strategy utilizing two DDPG agents: an eMBB scheduling agent and a URLLC puncturing agent. The eMBB agent addresses problem $P1$ with constraints $C1, C2$, and $C3$, and operates at time slot granularity. The URLLC agent, operates on mini-slots, addressing problem $P2$ with constraints $C4, C5$, and $C6$. At the beginning of each time slot $t \in \mathcal{T}$, the eMBB scheduling agent allocates the M available RBs and P_E power to

eMBB users to maximize the eMBB user data rates while meeting the individual data rate requirements. Simultaneously, within a time slot, the URLLC puncturing agent executes every mini-slot $k \in \mathcal{K}$ to determine the number of RBs that URLLC users need to puncture from eMBB users. This agent also allocates transmission power P_U to scheduled URLLC users to meet the URLLC reliability and latency requirements. The DDPG framework of the eMBB scheduling agent is composed of two neural networks: an actor network π_E and a critic network Q_E, both modeled as deep neural networks. Similarly, the URLLC puncturing agent employs an actor network π_U and a critic network Q_U. The actor networks take the current state as their inputs and select deterministic actions

$$a_E(t) = \pi_E(s_E(t)|\psi_E) + \mathcal{N}(t), \mathcal{N}(t) \sim N(\mu_n, \sigma_n^2) \tag{21}$$

and

$$a_U(k) = \pi_U(s_U(k)|\psi_U) + \mathcal{N}(k), \mathcal{N}(k) \sim N(\mu_n, \sigma_n^2), \tag{22}$$

where ψ_E and ψ_U represent the trainable parameters of the eMBB and URLLC actor networks, respectively. Symbols $\mathcal{N}(t)$ and $\mathcal{N}(k)$ represent noise terms that follow a normal distribution with a mean of μ_n and a variance of σ_n^2. These are added to the actions for exploration purposes. The critic networks are designed to predict Q-values $Q_E(s_E(t), a_E(t)|\phi_E)$ and $Q_U(s_U(k), a_U(k)|\phi_U)$ for the actions taken by the actors in states $s_E(t)$ and $s_U(k)$, thus evaluating the effectiveness of actions $a_E(t)$ and $a_U(k)$ based on the trainable parameters ϕ_E and ϕ_U of the eMBB and URLLC critic networks, respectively.

During the learning phase, both the eMBB scheduling and URLLC puncturing agents optimize the actor parameters (ψ_E, ψ_U) and the critic parameters (ϕ_E, ϕ_U). This optimization occurs over multiple training epochs. In each epoch, the actors take the current state (s_E for eMBB and s_U for URLLC), output actions a_E and a_U, and observe rewards R_E and R_U and the subsequent states s'_E for s'_U. The eMBB agent stores its complete experience, (s_E, a_E, R_E, s'_E), in its replay buffer $\mathcal{D}_E$; the URLLC agent stores its experience, (s_U, a_U, R_U, s'_U), in $\mathcal{D}_U$. Mini-batch experiences are sampled from these replay buffers for training: $\mathcal{B}_E = \{(s_E(i), a_E(i), R_E(i), s'_E(i))\}$ and $\mathcal{B}_U = \{(s_U(i), a_U(i), R_U(i), s'_U(i))\}$, where $i = (1, 2, ..., |\mathcal{B}|)$. The critic loss functions,

$$J(\phi_E) = \frac{1}{D}\sum_{i=1}^{D}\left(y_E(i) - Q_E(s_E(i), a_E(i)|\phi_E)\right)^2 \tag{23}$$

and

$$J(\phi_U) = \frac{1}{D}\sum_{i=1}^{D}\left(y_U(i) - Q_U(s_U(i), a_U(i)|\phi_U)\right)^2 \tag{24}$$

are used for updating the critic networks. Expressions

$$y_E(i) = R_E(i) + \gamma Q'_E(s_E(i+1)|\pi'_E(s_E(i+1)|\psi'_E)|\phi'_E) \tag{25}$$

and

$$y_U(i) = R_U(i) + \gamma Q'_U(s_U(i+1)|\pi'_U(s_U(i+1)|\psi'_U)|\phi'_U) \tag{26}$$

are the target Q-values for the eMBB and URLLC experiences, respectively. The networks π'_E and π'_U serve as the target actor models for the eMBB and URLLC agents, respectively, while Q'_E and Q'_U are the corresponding target critic models. Parameters ψ'_E and ϕ'_E are associated with the target actor and critic networks for the eMBB agent, whereas ψ'_U and ϕ'_U are linked to those of the URLLC agent.

The actor networks are updated by applying the policy gradient method:

$$\begin{aligned}\nabla_{\psi_E} J(\psi_E) =& \frac{1}{D}\sum_{i=1}^{D}\nabla_{a_E} Q_E(s_E(i), a_E(i)|\phi_E) \\ &\times \nabla_{\psi_E}\pi_E(s_E(i)|\psi_E),\end{aligned} \tag{27}$$

$$\begin{aligned}\nabla_{\psi_U} J(\psi_U) =& \frac{1}{D}\sum_{i=1}^{D}\nabla_{a_U} Q_U(s_U(i), a_U(i)|\phi_U) \\ &\times \nabla_{\psi_U}\pi_U(s_U(i)|\psi_U).\end{aligned} \tag{28}$$

The target network parameters $\psi'_E, \phi'_E, \psi'_U, \phi'_U$ are updated periodically based on the main network parameters as follows [34]:

$$\psi'_E = \tau\psi_E + (1-\tau)\psi'_E, \tag{29}$$

$$\phi'_E = \tau\phi_E + (1-\tau)\phi'_E \tag{30}$$

$$\psi'_U = \tau\psi_U + (1-\tau)\psi'_U, \tag{31}$$

$$\phi'_U = \tau\phi_U + (1-\tau)\phi'_U. \tag{32}$$

The overall training process of IDRAP is outlined in Algorithm 1. It begins by initializing the weight parameters of the actor and critic networks, copying them to the target networks, and initializing the replay memories (lines 1-3). The training proceeds over a specified number of episodes, where each episode lasts F time frames. Each time frame f consists of T time slots. During each time slot t, the eMBB agent observes the state $s_E(t)$, takes action $a_E(t)$ based on the output of the actor network π_E (lines 7-9). Within each time slot t, the URLLC agent operates within multiple mini-slots k, where each time slot t spans

K mini-slots. In each mini-slot k, the URLLC agent observes the state $s_U(k)$, runs the actor network π_U, takes action $a_U(k)$, executes the action, receives reward $R_U(k)$, observes the new state $s_U(k+1)$, and stores the experience $(s_U(k), a_U(k), R_U(k), s_U(k+1))$ in the replay memory $\mathcal{D}_U$ (lines 11-16). The agent repeats these steps for each of the K mini-slots of the current time slot. At the end of each time slot, after the final mini-slot, the eMBB agent receives reward $R_E(t)$, observes the next state $s_E(t+1)$, and stores the experience $(s_E(t), a_E(t), R_E(t), s_E(t+1))$ in the replay memory $\mathcal{D}_E$ (lines 18-20). Upon completing the time frame f, both the eMBB and URLLC agents sample mini-batch experiences from their respective replay buffers and update the parameters of the actor and critic networks (lines 22-24). This parameter updating occurs once per time frame, and the learning process is repeated for each episode. Upon concluding the training phase, the eMBB scheduling and URLLC puncturing agents deploy the main actor networks with their trained parameters—the output of Algorithm 1—to make real-time decisions in every time slot and mini-slot, respectively. These agents continue to adapt and refine their strategies to to changing network conditions over time through interactions with the environment.

The inference stage of is illustrated in the enclosed flowchart presented in Fig. 2, the process begins with the eMBB agent observing the state $S_E(t)$ at the beginning of each slot t. The agent then executes its actor model to determine the action $a_E(t)$ and communicates it to the URLLC agent. In each mini-slot, if there is URLLC traffic, the URLLC agent observes its own state $S_U(k)$ influenced by the eMBB agent's actions. The URLLC agent then executes its actor model to take an action $a_U(k)$. After taking this action, the URLLC agent receives its immediate reward $r_U(k)$, whereas the eMBB agent receives its reward $r_E(t)$ at the end of the slot as a result of the actions of both agents and the environment feedback.

C. Design of the DDPG Agents

The eMBB scheduler's actor and critic networks implement neural networks with three hidden layers, each containing 256 neurons. The actor network's input layer is designed to match the eMBB scheduler's state space dimension of $3E$, as outlined in (13). The critic network's input layer is expanded to $3E + 2E$ neurons, accommodating the dimensions of both the state and action spaces as per (14). The actor network's output layer has a dimension of $2E$ and is tailored for determining the RB and power allocations for the E eMBB users, while the critic network outputs a single value indicative of the action's quality value. The hidden layers employ

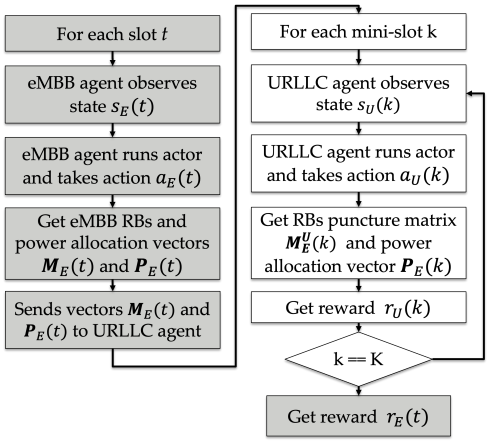


Fig. 2: Flowchart illustrating the IDRAP inference process, color-coded: light gray boxes represent the steps undertaken by the eMBB agent, and white boxes indicate the steps performed by the URLLC agent.

the ReLU activation functions, whereas the critic's input and output layers use linear functions. The actor's output layer uses a sigmoid function ensuring actions are bound within $[0, 1]$. The initial E elements of the actor's output, which correspond to the RB allocation, undergo normalization to ensure their sum is 1 before being scaled by M to derive the actual RB allocations $m_1(t), m_2(t), ..., m_E(t)$ for the E eMBB users. This satisfies Constraint $C2$ of (10). A similar normalization and scaling process is applied to the action vector related to the transmission power allocation. Fig. 3 depicts the architecture of the eMBB actor network, including the action post-processing.

The URLLC puncturing agent has three hidden layers for both the actor and critic networks, where the initial layer features 512 neurons and each subsequent layer 256 neurons. The actor network's input layer has $4U+3E$ neurons and aligns with the URLLC state space as defined in (17). The critic network's input layer has $4U+3E+E\cdot(U+1)+U$ neurons, factoring in the action space dimension as specified in (18). The actor's output layer is designed to match the action space (18). The first E elements of the action are scaled by $\mathbf{M}_E(t)$ to ascertain the number of RBs to be punctured from eMBB users. These are further distributed among URLLC users based on normalized action elements, calculating the specific RBs to be punctured by each URLLC user from each eMBB user. The last U elements of the action are scaled by P_U after being normalization to meet Constraint $C6$ of (12). This provides the power allocations for URLLC users. Fig. 4 illustrates the URLLC actor's framework along with its action modification post-processing. Both agent models use the Adam algorithm for learning optimization with a learning rate of 10^{-3},

a mini-batch size of 128 for random sampling, and a replay buffer size of $|\mathcal{D}_E|= |\mathcal{D}_U|= 10^5$.

The computational complexity of the IDRAP framework is influenced by the architecture of the DRL agents, the phases of operation (training vs. inference), and the hierarchical structure of the problem setup, including the number of episodes, frames, slots, and mini-slots. The training phase encompasses forward passes for action determination, backward passes for parameter updates, and replay buffer sampling. The complexity for each pass through the neural network is roughly $O(N_l \times N_n^2)$, where N_l represents the layer count, and N_n the neuron count in the largest layer. The eMBB scheduling agent's training complexity is of the order of $O(N_l \times N_n^2 \times L \times F \times T \times D)$, factoring in the episodes (L), frames (F), slots (T), and mini-batch size (D). The URLLC agent's complexity is of the order of $O(N_l \times N_n^2 \times L \times F \times T \times K \times D)$, where K accounts for the mini-slots within each time slot.

VI. Performance Evaluation

We evaluate the effectiveness of IDRAP in this section. The numerical analysis is centered around the SSLs of both types of users in addition to the aggregated data rate of eMBB users as incorporated in the objective function. The assessment covers different parameter settings and compares the performance of the proposed scheme against prior art and various benchmarks.

A. Simulation Environment Setup

We simulate a wireless network spanning an area with a radius of 500 m. It features a single gNodeB deployed at the center of the area. The simulation is performed using the PyTorch library running in Python 3.8. We consider $E = 12$ eMBB users and $U = 6$ URLLC users. These users are uniformly and randomly distributed across the area. We model the URLLC traffic load as a Poisson process with an average packet arrival rate of λ packets per millisecond, where each packet is 32 bytes. The eMBB users are assumed to have full buffers, implying that each eMBB user always has data in its buffer ready for transmission. The minimum data rate requirements of eMBB users are uniformly distributed with a mean value of R^{min}. We adopt the 15 kHz SCS, resulting in a bandwidth of 180 kHz for each RB, and a slot duration of 1 ms. Each slot is further divided into 7 mini-slots, each having a duration of 0.14 ms. The total system bandwidth is 20 MHz, or 100 RBs. Table III summarizes the simulation parameters. In our simulation setup, the IDRAP framework is trained for 5000 episodes to ensure the robust adaptation of our DRL agents to various network conditions. Each episode is designed to mirror a realistic operational scenario, consisting of 10,000 frames, where each frame encompasses 10 slots.

Algorithm 1 Intelligent Dynamic Resources Allocation and Puncturing

Initialize randomly the actors and critics training parameters ψ_E, ψ_U, ϕ_E and ϕ_U a;
Copy the values of ψ_E, ψ_U, ϕ_E and ϕ_U to ψ'_E, ψ'_U, ϕ'_E and ϕ'_U ;
Initialize replay memories $\mathcal{D}_E$ and $\mathcal{D}_U$ to $\mathcal{D}_{size}$;
for all episodes **do**
  for f = 1 to F **do**
    for t = 1 to T **do**
      Observe $s_E(t)$ based on (13) ;
      Select action $a_E(t)$ based on (21);
      Get $\mathbf{M}_E(t)$ and $\mathbf{P}_E(t)$ based on (15);
      for k = 1 to K **do**
        Observe $s_U(k)$ based on (17);
        Select action $a_U(k)$ based on (22);
        Get $\mathbf{M}_E^U(k)$ and $\mathbf{P}_U(k)$ based on (19);
        Get the reward $R_U(k)$ based on (20);
        Observe new state $s_U(k+1)$ based on (17);
        Store $(s_U(k), a_U(k), R_U(k), s_U(k+1))$ in replay memory $\mathcal{D}_U$;
      end for
      Get the reward $R_E(t)$ based on (16);
      Observe new state $s_E(t+1)$ based on (13);
      Store $(s_E(t), a_E(t), R_E(t), s_E(t+1))$ in reply memory $\mathcal{D}_E$;
    end for
    Randomly sample $\mathcal{B}_E$ and $\mathcal{B}_U$ of size D from $\mathcal{D}_E$ and $\mathcal{D}_U$;
    Update ϕ_E, ϕ_U, ψ_E ψ_U based on (23), (24), (27) and (28);
    Update $\psi'_E, \phi'_E, \psi'_U, \phi'_U$ based on (29), (30),(31) and (32);
  end for
end for
Output: Trained π_E and π_U;

Moreover, to evaluate the performance of our framework in varying network conditions, we executed 500 independent simulation runs during the inference phase. Each of these runs spans 10,000 frames. The results presented in this section are averaged over these simulation runs.

B. Benchmarking Schemes

We implement five benchmarking schemes for comparative analysis. The first is a dynamic resource slicing framework based on orthogonal slicing. It employs three DDPG agents. We coin it Intelligent Dynamic Resource Slicing (IDRS). IDRS operates on a hierarchical model with specialized roles for each agent: The bandwidth slicing agent dynamically partitions the available band-

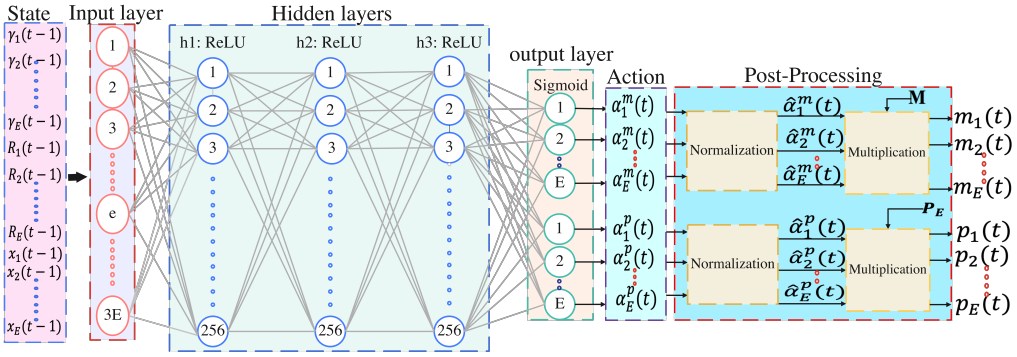


Fig. 3: eMBB scheduling agent—actor network architecture.

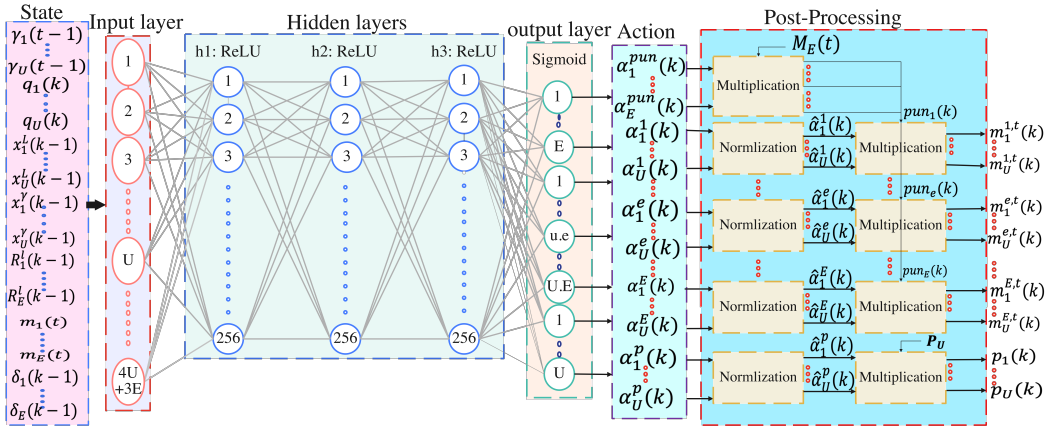


Fig. 4: URLLC scheduling agent—actor network architecture.

h!

TABLE III: Simulation parameters.

Parameter	Value	Parameter	Value
E	12	P_E	30 dBm
U	6	P_U	20 dBm
μ	0	α	3
SCS	15 kHz	σ^2	-114 dBm
B	180 kHz	τ	0.15 ms
M	100 RBs	ϵ	10^{-5}
F	1000 frames	γ_{min}	5 dB
T	10 slots/frame	ζ^{URLLC}	0.99
K	7 minislots/slot	ζ^{eMBB}	0.91

width between eMBB and URLLC slices. The eMBB resource allocation agent is responsible for allocating RBs and transmission power to eMBB users. The URLLC resource scheduling agent allocates resources to URLLC users in accordance to their specific traffic and reliability needs. This framework adopts the same MDP design as IDRAP, ensuring consistency in the decision-making processes.

The second scheme employs decomposition and relaxation-based resource allocation for eMBB users and policy gradient based actor-critic learning (PGACL) for servicing URLLC users through resource puncturing. It is introduced in [25]. Its primary objective is maximizing the average eMBB data rate while minimizing the variance of the achieved data rate among eMBB users. This is achieved by satisfying the URLLC latency requirements through the use of a puncturing scheme based on the policy gradient model.

The third benchmark is SWTS, which is described in [28]. It formulates the coexistence problem as a utility function, combining the summation of the SSLs of both eMBB and URLLC users. A single DDPG agent allocates RBs to eMBB users and determines the number of RBs to be punctured from each eMBB user in every mini-slot.

The forth benchmark is Sum-Log, which allocates resources to eMBB users maximizing the weighted sum of logarithms of eMBB data rates. Resource proportionality as in [21] is applied for resource puncturing, meaning that each URLLC user punctures a number of RBs from each eMBB user in proportion to its RB allocation.

The final benchmark is Sum-Rate, which allocates resources to eMBB users maximizing the sum data rate. It prioritizes eMBB users that have good channel

conditions. More precisely, it allocates more resources to users with good channel conditions and punctures resources from users with poor channel conditions.

C. Results and Analyses

We conduct several simulation runs with different eMBB and URLLC penalty weights—ζ^E and ζ^U—to study their impact on the models' performances in terms of URLLC and eMBB SSLs.

1) URLLC SLL: Fig. 5a plots the SSL of URLLC users over varying URLLC packet arrival rate λ. A common trend is observed across all models: as λ increases, the traffic load intensifies, demanding more resources and subsequently leading to a SSL decrease. Setting a lower URLLC penalty weight compared to the eMBB penalty weight, such as in the case of IDRS(8,2) and IDRP(8,2), where $\zeta^E = 8$ and $\zeta^U = 2$, makes the models more susceptible to increasing traffic loads. This results in poor performance in terms of SSL, failing to reach the threshold value of 0.99 even for low URLLC packet arrival rates. Conversely, using a higher ζ^U compared to ζ^E, as is the case for IDRS(4,8) and IDRP(4,8), leads to a good SSL at a moderate packet rate of 6 packets/ms. The performance however deteriorates for packet rates of 8 packets/ms and above due to the scarce resources for serving all incoming packets. In cases with equal penalty weights of $\zeta^E = \zeta^U = 4$, as captured by IDRS(4,4) and IDRP(4,4), the models exhibit intermediate performance, leaning more towards the high URLLC penalty scenario.

Fig. 5b illustrates the impact of increasing the minimum required data rate R_{min} of eMBB users on the URLLC SSL for different penalty weight values. The observed trend aligns with the previous results: as R_{min} increases, more resources are needed to meet the quality requirements of eMBB users. Consequently, fewer resources are allocated to URLLC users, leading to a decrease in the URLLC SSL. This is less pronounced for high penalty weights, indicating higher prioritization of URLLC services. Conversely, the URLLC SLL decrease is substantial when setting a high eMBB penalty weight that prioritizes eMBB services. This is the result of more resources being assigned to eMBB users and fewer resources being punctured for supporting URLLC traffic to maintain a high eMBB quality. For scenarios with equal penalty weights, the performance falls in between, leaning more towards the high URLLC penalty weight scenario. The URLLC SSL remains high and close to the threshold value of 0.99 for low to moderate R_{min}. However, the performance starts to decline for high R_{min} (15 Mbps and above), deviating from the set threshold.

Fig. 6a plots the average RLLC SSL over URLLC packet arrival rate λ for IDRAP and five benchmarks. For this comparison, we set the minimum eMBB data rate for

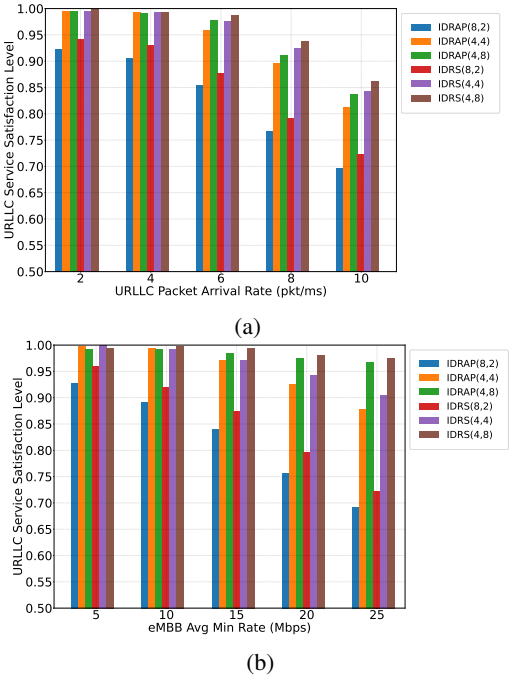


Fig. 5: Average SSL for URLLC users for different values of penalty weight (ζ^E,ζ^U) over URLLC packet arrival rate (a) and over minimum rate requirement per average eMBB user (b).

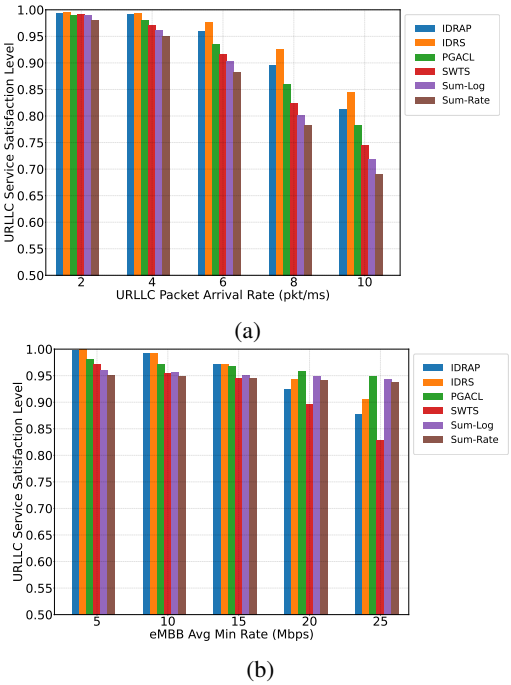


Fig. 6: Average SSL for URLLC users for the proposed IDRAP scheme and five benchmarks over URLLC packet arrival rate (a) and average minimum rate requirement per eMBB user (b).

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3422350

eMBB demand to $R_{min} = 10$ Mbps and use $\zeta^E = \zeta^U = 4$. The figure highlights the superior performance of IDRAP and IDRS in ensuring URLLC service satisfaction. A key factor contributing to IDRAP's performance in maintaining high SSLs across different packet arrival rates is the inclusion of reliability metrics in the power allocation decision process. Unlike SWTS, Sum-Rate, and Sum-Log, which lack a sophisticated power control mechanism and distribute power uniformly among URLLC users, IDRAP and IDRS dynamically allocate power with the awareness of the URLLC reliability requirements. While incorporating a power control strategy, PGACL primarily focuses on optimizing power allocation for eMBB users without directly addressing URLLC reliability needs. In PGACL's model, URLLC users inherit the power allocated to eMBB users whose resources they puncture. This indirect approach to power allocation does not specifically aim to meet URLLC reliability standards, as the power optimization does not directly factor in the threshold error probability of URLLC services. Fig. 6b illustrates the URLLC SSL as a function of the average minimum eMBB data rate demand R_{min}. The inclusion of the eMBB data rate requirement significantly affects the URLLC SSL. IDRAP, IDRS, and SWTS incorporate the eMBB data rate requirements in their optimization constraints. This results in a noticeable decline in URLLC SSL as the minimum average data rate requirement for eMBB users increases. This decline reflects the inherent trade-off between satisfying the high data rate demands of eMBB services and the strict latency and reliability requirements of URLLC traffic. In contrast, PGACL, Sum-Log, and Sum-Rate do not incorporate eMBB data rate requirements in their optimization objectives and this results in a relatively stable URLLC SSL across different eMBB data rate requirements. This stability, however, comes at the cost of failing to meet the service level expectations of eMBB users, as shown in Fig. 7b.

2) eMBB SLL: Figs. 7a and 7b numerically evaluate the SSL of eMBB users for various penalty weights. Consistent trends are observed in both subplots as before, showing the decline of the SLL over sλ and R_{min}, respectively across all penalty weight values. When ζ^U is higher than ζ^E, the eMBB SSL experiences a significant decline when λ or R_{min} increases. This is so because of the prioritization of URLLC over eMBB traffic. For scenarios where ζ^E and ζ^U are equal, the model effectively sustains the SSL levels above the set SSL threshold of 0.99 for low to moderate λ and R_{min}. However, for an R_{min} of 15 Mbps or higher, the SSL drops below the threshold even for high ζ^E. This outcome is attributed to the fact that despite prioritizing eMBB traffic, insufficient resources are available to meet high QoS demands.

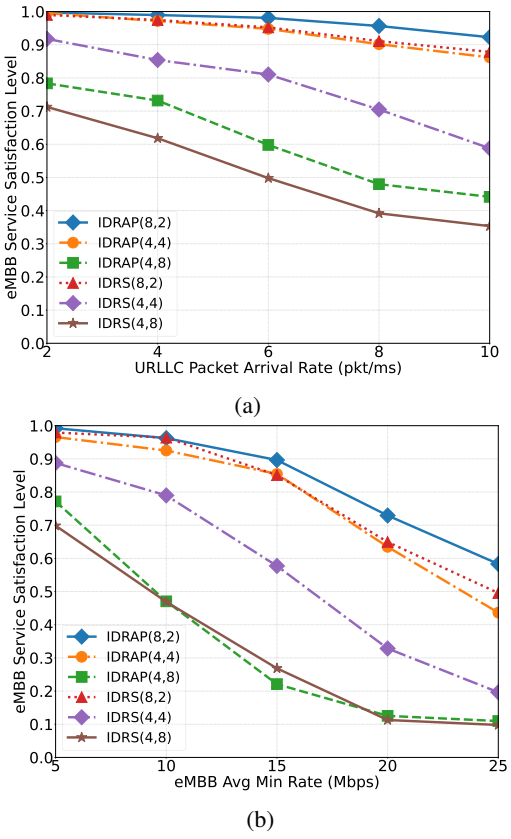


Fig. 7: Average SSL for eMBB users for different values of penalty weight (ζ^E,ζ^U) over URLLC packet arrival rate (a) and over minimum rate requirement per average eMBB user (b).

Fig. 8a compares the average SSL of eMBB users of the six solutions as a function of the URLLC packet arrival rate λ. For this comparison, we set the minimum required data rate for eMBB users to $R_{min} = 5Mbps$ and use $\zeta^E = \zeta^U = 4$. The observed trend indicates that an increase in the URLLC packet rate negatively affects the SSL of eMBB users across all schemes. However, this impact is notably lower for IDRAP and moderate for PGACL and SWTS. For a higher λ (8 packets/ms and above), IDRAP achieves approximately 12.43% higher SSL compared to PGACL and SWTS and a remarkable improvement of around 76.5% and 103% over Sum-Rate and Sum-Log, respectively. On the other hand, IDRS achieves lower SSL than and SWTS but outperforms Sum-Rate and Sum-Log. The Sum-Rate and Sum-Log schemes prioritize URLLC packets over eMBB services, leading to puncturing all eMBB resources to fulfill the URLLC requirements; this contributes to their poorer performance.

Fig. 8b illustrates the relation between the average minimum required eMBB data rate R_{min} and the eMBB SSL. Here we set the URLLC packet arrival rate to $\lambda = 2$ packets/ms and use pre-trained models with

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3422350

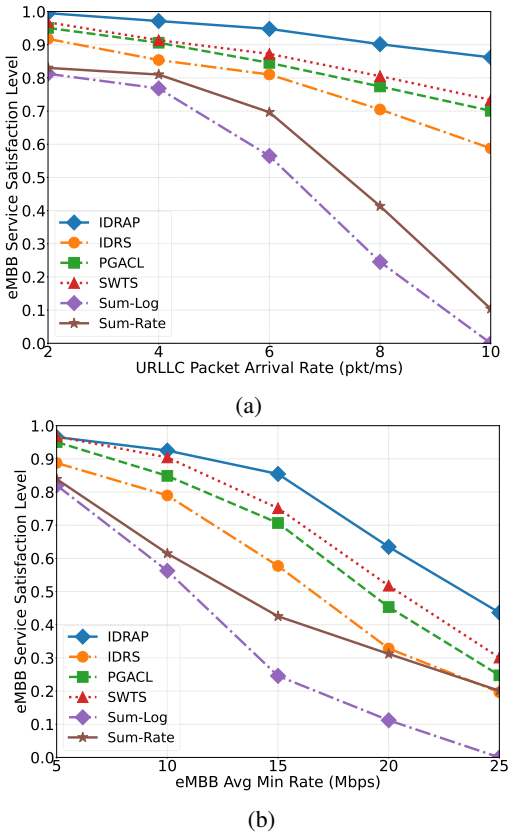


Fig. 8: Average SSL for eMBB users for the proposed IDRAP scheme and five benchmarks over URLLC packet arrival rate (a) and average minimum rate requirement per eMBB user (b).

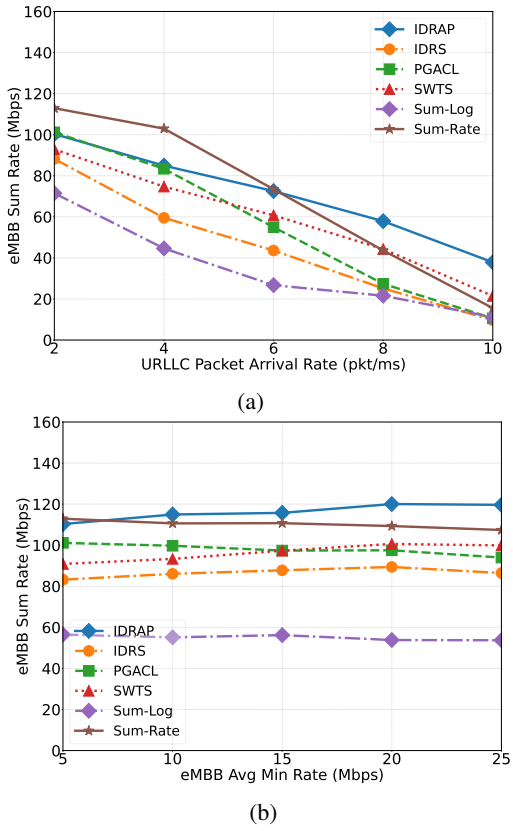


Fig. 9: eMBB Sum Rate for the proposed IDRAP scheme and five benchmarks versus different URLLC packet arrival rate (a) and different mean minimum rate requirement per eMBB user (b).

$\zeta^E = \zeta^U = 4$. Similar to the previous case, increasing R_{min} negatively affects all schemes and most schemes experience a dramatic decrease in SSL as R_{min} increases. This substantial decline does not stem from the resources allocated to URLLC packets, as the packet arrival rate is only 2 packets/ms. The decrease results from the limited resources that are available to fulfill the high QoS demands of eMBB users. IDRAP manages to maintain SSL values above the threshold for low to moderate R_{min} while most other schemes fail to do so and all experience a rapid SSL decline. IDRS exhibits better performance than Sum-Log and Sum-Rate, but it lags behind PGACL and SWTS.

3) eMBB Sum Rate: Fig. 9a and 9b plot the average sum data rate of eMBB users over λ and R_{min}, respectively. We employing the same pre-trained model as for the previous results with $\zeta^E = \zeta^U = 4$, $R_{min} = 5$ Mbps in Fig. 9a, and $\lambda = 2$ packets/ms in Fig. 9b.

We observe that the URLLC packet rate λ has a significant effect on the sum rate, whereas the mean required data rate R_{min} does not. As λ increases, more resources are allocated to URLLC users in the bandwidth slicing case and more resources are punctured from eMBB users in the puncturing case. Both lead to a decrease in the achieved sum data rate of eMBB users. For low λ, the Sum-Rate scheme attains the highest sum data rate since this strategy prioritizes users with the best channel conditions. However, as λ increases, the Sum-Rate scheme experiences a steep decrease in performance as more resources are punctured to URLLC users to meet their requirements, resulting in a decrease in the achieved eMBB data rates. In contrast, IDRAP outperforms all schemes for moderate to high packet loads due to the agent's strategy of maintaining a balance between URLLC and eMBB services.

Parameter R_{min} has no impact on the sum rate for PGACL, Sum-Log, and Sum-Rate as it is not included in their objective functions. PGACL aims at increasing the average data rate of eMBB users while decreasing their variance with the goal of a fair resource allocation. Sum-Log aims to increase the weighted sum rate, considering the proportional fair index to ensure fair resource allocation across eMBB users. Parameter R_{min} is part of the objective functions of IDRS, IDRAP, and SWTS. Therefore, an increase in its value with a constant value of λ implies more resources are needed for eMBB

users to maintain their QoS demands. Consequently, these schedulers assign more resources to eMBB users, resulting in a slight increase in their data rates and an overall increase in the sum rate.

VII. Conclusions

In this paper, we have explored the effectiveness of intelligent dynamic resource allocation and puncturing for a better coexistence between eMBB and URLLC services. We have developed distinct optimization models for both resource allocation and puncturing, each tailored to the unique resource allocation needs of eMBB and URLLC services, while taking into account their specific performance requirements. After carefully designing the optimization problems, we have proposed a DRL based solution, leveraging the DDPG model. We have designed the state, action, and reward functions for each agent to align with the parameters defined in our optimization models. Furthermore, we have incorporated a penalty in the reward functions to enhance the reliability of services provided to both eMBB and URLLC users.

Through comprehensive simulations, we have validated that the proposed IDRAP successfully meets the SSL thresholds for both eMBB and URLLC services for scenarios with moderate URLLC traffic load and minimum rate requirements for eMBB users. The proposed scheme outperforms existing solutions as well as a newly introduced benchmarks which are not able to balance as effectively the two services across network settings. The overall performance of the proposed approach can be improved by incorporating complex neural networks, such as convolutional neural networks and LSTM, into the agents' designs to improve the framework's adaptability to dynamic environments with high-dimensional state spaces. This is in our future plans. Furthermore, we plan to extend our framework to encompass multi-agent systems, particularly to cater to multi-gNodeB scenarios, while exploring the implementation of federated learning to enhance the multi-agent learning processes. These enhancements will enable addressing the intricate and demanding KPI requirements of emerging use cases for next-generation wireless networks.

Acknowledgment

This work was supported in part by the National Science Foundation under grant number ECCS-2030291.

References

[1] C.-X. Wang, X. You, X. Gao, X. Zhu, Z. Li, C. Zhang, H. Wang, Y. Huang, Y. Chen, H. Haas, J. S. Thompson, E. G. Larsson, M. D. Renzo, W. Tong, P. Zhu, X. Shen, H. V. Poor, and L. Hanzo, "On the road to 6G: Visions, requirements, key technologies, and testbeds," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 905–974, 2023.

[2] I. F. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133 995–134 030, 2020.

[3] ITU-R, "IMT vision-framework and overall objectives of the future development of IMT for 2020 and beyond." International Telecommunication Union, Tech. Rep. M.2083-0, 2015.

[4] M. Fuentes *et al.*, "5G new radio evaluation against IMT-2020 key performance indicators," *IEEE Access*, vol. 8, pp. 110 880–110 896, 2020.

[5] F. W. Vook, A. Ghosh, E. Diarte, and M. Murphy, "5G new radio: Overview and performance," in *52nd Asilomar Conf. on Signals, Systems, and Computers*. IEEE, 2018, pp. 1247–1251.

[6] The Third Generation Partnership Project (3GPP), "Study on physical layer enhancements for NR ultra-reliable and low latency case (URLLC) (Release 16)," Technical Report (TR) 38.824, 03 2019, Version 16.0.0.

[7] R. Kumar, D. Sinwar, and V. Singh, "QoS aware resource allocation for coexistence mechanisms between eMBB and URLLC: Issues, challenges, and future directions in 5G," *Computer Communications*, vol. 213, pp. 208–235, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366423003894

[8] ITU-R, "Future technology trends of terrestrial international mobile telecommunications systems towards 2030 and beyond." International Telecommunication Union, Tech. Rep. M.2516-0, 2022.

[9] 3GPP, "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G," Technical Report (TR) TR 21.915, 10 2019, Version 15.0.0.

[10] ——, "Final Report of 3GPP TSG RAN WG1 Meeting 88 Version 1.0.0," Feb 2017.

[11] B. S. Khan, S. Jangsher, A. Ahmed, and A. Al-Dweik, "Urllc and embb in 5g industrial iot: A survey," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 1134–1163, 2022.

[12] G. Pocovi, R. Abreu, P. Andres, M. Deghel, K. Hugl, T. Jacobsen, K. Jayasinghe, P. Kela, J. Korhonen, P.-H. Kuo, L. Kuru, Z. Li, T. Lunttila, E. Peralta-Calvo, C. Rosa, and T. Tao, "Further enhanced urllc and industrial iot support with release-17 5g new radio," *IEEE Communications Standards Magazine*, vol. 7, no. 4, pp. 12–19, 2023.

[13] P. Yang, X. Xi, T. Q. S. Quek, J. Chen, X. Cao, and D. Wu, "How should i orchestrate resources of my slices for bursty URLLC service provision?" *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 1134–1146, 2021.

[14] M. Setayesh, S. Bahrami, and V. W. Wong, "Joint PRB and power allocation for slicing eMBB and URLLC services in 5G C-RAN," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[15] L. Feng, Y. Zi, W. Li, F. Zhou, P. Yu, and M. Kadoch, "Dynamic resource allocation with RAN slicing and scheduling for uRLLC and eMBB hybrid services," *IEEE Access*, vol. 8, pp. 34 538–34 551, 2020.

[16] M. Yan, G. Feng, J. Zhou, Y. Sun, and Y.-C. Liang, "Intelligent resource scheduling for 5G radio access network slicing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7691–7703, 2019.

[17] R. Li *et al.*, "Deep reinforcement learning for resource management in network slicing," *IEEE Access*, vol. 6, pp. 74 429–74 441, 2018.

[18] Q. Liu, T. Han, N. Zhang, and Y. Wang, "DeepSlicing: Deep reinforcement learning assisted resource allocation for network slicing," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[19] A. Filali, Z. Mlika, S. Cherkaoui, and A. Kobbane, "Dynamic sdn-based radio access network slicing with deep reinforcement learning for urllc and embb services," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2174–2187, 2022.

[20] S. Choi, S. Choi, G. Lee, S.-G. Yoon, and S. Bahk, "Deep reinforcement learning for scalable dynamic bandwidth allocation in ran slicing with highly mobile users," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 1, pp. 576–590, 2024.

[21] A. Anand, G. De Veciana, and S. Shakkottai, "Joint scheduling of URLLC and eMBB traffic in 5G wireless networks," in *IEEE INFOCOM 2018 - IEEE Conf. on Computer Communications*, 2018, pp. 1970–1978.

[22] M. Alsenwi, N. H. Tran, M. Bennis, A. K. Bairagi, and C. S. Hong, "EMBB-URLLC resource slicing: a risk-sensitive approach," *IEEE Communications Letters*, vol. 23, pp. 740–743, 2019.

[23] A. A. Esswie and K. I. Pedersen, "Opportunistic spatial preemptive scheduling for URLLC and eMBB coexistence in multi-user 5G networks," *IEEE Access*, vol. 6, pp. 38 451–38 463, 2018.

[24] 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 Transactions on Communications*, vol. 69, no. 3, pp. 1736–1749, 2021.

[25] 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 Transactions on Wireless Communications*, vol. 20, no. 7, pp. 4585–4600, 2021.

[26] R. M. Sohaib, O. Onireti, Y. Sambo, R. Swash, S. Ansari, and M. A. Imran, "Intelligent resource management for eMBB and URLLC in 5G and beyond wireless networks," *IEEE Access*, vol. 11, pp. 65 205–65 221, 2023.

[27] X. Jiang, K. Liang, X. Chu, C. Li, and G. K. Karagiannidis, "Multiplexing eMBB and URLLC in wireless powered communication networks: A deep reinforcement learning-based approach," *IEEE Wireless Communications Letters*, vol. 12, no. 10, pp. 1716–1720, 2023.

[28] J. Li and X. Zhang, "Deep reinforcement learning-based joint scheduling of eMBB and URLLC in 5G networks," *IEEE Wireless Communications Letters*, vol. 9, no. 9, pp. 1543–1546, 2020.

[29] J. A. Hurtado Sánchez, K. Casilimas, and O. M. Caicedo Rendon, "Deep reinforcement learning for resource management on network slicing: A survey," *Sensors*, vol. 22, no. 8, pp. 1–32, 2022.

[30] N. C. Luong *et al.*, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[31] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.

[32] J. Scarlett, V. Y. F. Tan, and G. Durisi, "The dispersion of nearest-neighbor decoding for additive non-Gaussian channels," *IEEE Transactions on Information Theory*, vol. 63, no. 1, pp. 81–92, 2017.

[33] A. Destounis and G. S. Paschos, "Complexity of URLLC scheduling and efficient approximation schemes," in *2019 International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, 2019, pp. 1–8.

[34] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. M. O. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *CoRR*, vol. abs/1509.02971, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:16326763