Intelligent Radio Access Network Slicing for Service Provisioning in 6G: A Hierarchical Deep Reinforcement Learning Approach

Jie Mei, *Member, IEEE*, Xianbin Wang, *Fellow, IEEE*, Kan Zheng, *Senior Member, IEEE*, Gary Boudreau, *Senior Member, IEEE*, Akram Bin Sediq, *Member, IEEE*, and Hatem Abou-Zeid, *Member, IEEE*

***Abstract*— Network slicing is a key paradigm in 5G and is expected to be inherited in future 6G networks for the concurrent provisioning of diverse quality of service (QoS). Unfortunately, effective slicing of Radio Access Networks (RAN) is still challenging due to time-varying network situations. This paper proposes a new intelligent RAN slicing strategy with two-layered control granularity, which aims at maximizing both the long-term QoS of services and spectrum efficiency (SE) of slices. The proposed method consists of an upper-level controller to ensure the QoS performance, which enforces loose control by performing adaptive slice configuration according to the long-term dynamics of service traffic. The lower-level controller is to improve SE of slices, by tightly scheduling radio resources to users at the small time-scale. To realize the proposed RAN slicing strategy, we propose a model-free deep reinforcement learning (DRL) framework, which is a hierarchical structure that collaboratively integrating the modified deep deterministic policy gradient (DDPG) and double deep-Q-network algorithm. Specifically, the lower-level control problem is a mixed-integer stochastic optimization problem with multiple constraints. This kind of problem is hard to be directly solved by the exiting DRL algorithms, since it involves searching for the solution in a vast set of mixed-integer action space, which will induce unbearable computational complexity. Thus, we propose a novel action space reducing approach, embedding the convex optimization tools into the DDPG algorithm, to speed up the lower-level control. Furthermore, simulation results confirm the effectiveness of our proposed intelligent RAN slicing scheme.**

***Index Terms*— Network slicing, 5G beyond, 6G, deep reinforcement learning, and radio resource management.**

Manuscript received November 13, 2020; revised April 19, 2021; accepted June 8, 2021. Date of publication June 18, 2021; date of current version September 16, 2021. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Program under Grant RGPIN2018-06254 and in part by the Canada Research Chair Program. The associate editor coordinating the review of this article and approving it for publication was D. Marabissi. *(Corresponding author: Xianbin Wang.)*

Jie Mei and Xianbin Wang are with the Department of Electrical and Computer Engineering, Western University, London, ON N6A 5B9, Canada (e-mail: jmei28@uwo.ca; xianbin.wang@uwo.ca).

Kan Zheng is with the Intelligent Computing and Communication (IC^2) Laboratory, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China (e-mail: zkan@bupt.edu.cn).

Gary Boudreau, Akram Bin Sediq, and Hatem Abou-Zeid are with Ericsson Canada Inc., Ottawa, ON K2K 2V6, Canada (e-mail: gary.boudreau@ericsson.com; akram.bin.sediq@ericsson.com; hatem.abou-zeid@ericsson.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2021.3090423.

Digital Object Identifier 10.1109/TCOMM.2021.3090423

I. Introduction

A. Background and Motivation

OUR society is undergoing a digitization transformation by integrating digital technologies into vertical industries with an expected plethora of emerging services and applications. These vertical services with far-reaching effects are characterized by a highly diverse set of quality of service (QoS) requirements [1]. To enable stringent and diverse QoS of future applications, one of the essential design considerations for future sixth-generation (6G) communications should be focused on diverse QoS provisioning techniques that can efficiently utilize the limited and constrained resources. For this reason, network slicing will be inherited as one of the key enabling technologies in 6G systems [2]. However, current network slicing schemes in 5G cannot fully meet the highly diverse and evolving QoS requirements of these emerging applications [3]. Thus, network slicing still needs to evolve for increased flexibility and adaptiveness, while enabling such various vertical services of increasingly connected industry and society.

However, network slicing will introduce much more complexity into the operation of wireless networks, which makes traditional mathematical model-based approaches to network operation no longer adequate [4], [5]. This challenge motivates us to enable network slicing with Artificial Intelligence (AI) capabilities, where network entities can self-determine suitable control policy based on the experience obtained by processing previous data [6], [7]. To further improve the effectiveness of AI-empowered network slicing, the primary motivation of this paper lies in the following consideration.

Network conditions within a specific region could be characterized and affected by both long-term trends (i.e., the dynamics of service traffic) and short-term trends (i.e., the dynamics of the physical layer) of networks [8]. Thus, we can utilize network resources more opportunistically and efficiently for concurrent QoS provisioning by exploiting hidden network dynamics and patterns in different timescales. Inspired by this idea, this paper aims at designing an intelligent RAN slicing scheme to achieve stringently and customized QoS, which can accommodate the time-varying network conditions with multiple time and resource granularities. By adopting multiple

control granularities, network slicing can be implemented with the tailored control effectiveness for diverse QoS requirements with the reduced resource management complexity.

B. State of the Art

In the following, we will first introduce the principle of network slicing, state the challenges faced by the RAN slicing, and then discuss the importance of applying deep reinforcement learning (DRL) in the context of RAN slicing.

1) Principle and Challenges of Network Slicing: According to 3GPP TR 28.801, a network slice instance includes a set of network functions and the resources for these network functions, which are arranged and configured to form a complete logical network that meets specific network characteristics required by a service instance [9]. Thanks to network function virtualization, network slices can be deployed with flexibility and elasticity [10]. Each network slice instance is an independent part of radio access network (RAN), transport, and core networks (CN). Current researches in the CN slicing are well studied and mainly focusing on the deployment of CN slices, as well as placement and management of virtualized network functions (VNFs) [11], [12].

Contrary to the CN slicing, there are still challenges in the design of radio management of RAN slicing need to be studied. These challenges include the potential radio resource sharing conflicts, the utilization efficiency of radio resources, the dynamics of service traffic flow, as well as the performance isolation between slices. Therefore, considering the impact of RAN slicing on the radio interface protocol architecture in 5G networks, a RAN slicing orchestration framework is proposed to create pools of resources that are shared and allocated among RAN slices [13]. The authors in [14] propose a RAN slicing framework, in which the service requirements are transformed into the demand of radio resources. In [15], a concept of hierarchical soft-slicing framework is proposed to support differentiated services with diverse QoS requirements, in which radio resources are pre-allocated to each cell in a large time scale and the pre-allocated radio resources are according to instantaneous service demands. Besides, considering both the cellular network topology and inter-cell/inter-slice interference, several network-wide radio resource allocation schemes for RAN slicing are proposed for providing guaranteed throughput in each network slice [16]–[18]. Furthermore, in the heterogeneous RAN, a dynamic radio resource management framework is proposed, which jointly considers the scheduling priority of RAN slices, radio resources, front/back-haul capacity [19], [20].

Based on our literature review and analysis, the above works on network slicing have the following limitations.

- In most of the existing works on RAN slicing, radio resources are shared based on a fixed assignment scheme. Arguably, this static resource slicing paradigm can avoid the potential radio resource sharing conflicts between the co-existing RAN slices and thus achieve perfect performance isolation among different slices. However, it is prone to result in low utilization efficiency of radio resources. Meanwhile, most works perform radio resource management with single resource granularity, which limits the flexibility of RAN slicing. Therefore, with the aim of improving the radio resource utilization efficiency, it is beneficial to design a RAN slicing control strategy with multiple time-resource granularities, in which the radio resources can be dynamically shared between network slices.
- Most network slicing schemes are regulated by conventional mathematical model-based approaches, which are derived from wireless communication theory or field measurement campaigns [4]. The fundamental premise of model-based approaches is to obtain a precise mathematical model to describe the system. Then, based on the precise system model, we can further analysis or optimize the system performance. However, mathematical models are inherently limited by a trade-off between modelling accuracy and computational tractability. Thus, in the context RAN slicing, when using the conventional model-based approaches, the network dynamics and its time variation pattern are difficult to be modeled in an accurate way, due to the limited computational capacity at the edge of networks. In consequence, it is reasonable for us to enable the RAN slicing control process through a model-free methodology, such as deep learning (DL).

2) Need of Deep Reinforcement Learning in RAN Slicing: As mentioned above, due to the high complexity of RAN slicing and untenability of the conventional model-based approaches, there have been a few pioneer works to design intelligent RAN slicing architecture, which aims at incorporating DL (deep learning) technologies into network slicing and realizing the automatic RAN slicing control. DL is a model-free methodology, which can extract the inherent features of the system and make decisions by processing historical data set. Specifically, DRL can perform the self-optimization of system performance by interacting with an unknown environment, which has received tremendous research attention [4], [21].

In general, there are two categories of DRL: 1) DRL based on value functions and 2) DRL based on control policy search. One typical representative of the first category of DRL is Deep-Q-Network (DQN). DQN constructs a value function model for measuring how good each state or state-action pair is. Then, DQN obtains the suitable control policy implicitly by optimizing the value function model. Although DQN has a good sample efficiency and stable performance, the main disadvantage of DQN is that its action space must be a discrete variable set (i.e., it can only solve integer stochastic optimization problem). The discrete action space not only brings quantization error for the continuous stochastic optimization problem, but also induces an exponential increase in the size of action spaces [22]. On the other hand, the second category of DRL directly searches the optimal control policy by estimating of the gradient with respect to the parameters of the control policy. One typical representative of the second category of DRL is deep deterministic policy gradient (DDPG) algorithm, which is a DRL algorithm with actor-critic framework. Compared to DQN, DDPG can work well

for the continuous stochastic optimization problem, which is more common in network resource management.

Based on double-DQN algorithm, a service demand-aware radio resource slicing scheme is proposed to realize a dynamic spectrum sharing among different RAN slices at a large timescale [23], [24]. Furthermore, a two-timescale radio resource scheduling strategy is proposed in [25], where long short term memory (LSTM) neural network is utilized to predict the demand of radio resources of each slice on a long-term timescale, and then RL algorithm is used to perform the real-time radio resource scheduling among users in each slice. From a game-theoretic perspective, in [26], the RAN slicing is modeled as a non-cooperative stochastic game, where each slice is realized by a DQN and aims to selfishly maximize its own long-term QoS performance.

While DRL brings the good potential to realize the automatic operation of the RAN slicing system, it does not imply the DRL-based approaches can apply to all kinds of network operation problems. The existing DRL-based algorithms still need enhancement due to the following main technical challenge. Practically, the problem of RAN slicing involves jointly optimizing the slice configuration (discrete variables), radio resource allocation (discrete variables), and transmit power allocation (continuous variables), which is essentially a mixed integer stochastic optimization problem with multiple constraints. However, as mentioned above, existing DRL algorithms can only efficiently deal with pure-integer/pure-continuous stochastic optimization problems. Thus, it is ineffective to directly utilize existing DRL algorithms in a "plug and play" way [4]. As a result, it is imperative for us to design a new DRL framework by combining the advantages of the classical optimization-based tools and the conventional DRL.

C. Contribution

In this paper, considering a single-cell downlink scenario that contains several network slices sharing available radio resources, we propose an intelligent RAN slicing scheme with self-configuration and self-optimization capabilities, which is enabled by a customized and model-free DRL framework, to maximize the long-term QoS and spectrum efficiency (SE) of slices. The technical contributions of this paper can be summarized as follows:

- Designing a RAN slicing strategy with multiple time and resource granularities to accommodate the time-varying network conditions and diverse QoS requirements. It consists of an upper-level controller and a lower-level controller. According to the dynamics of service traffic at the large time-scale, the upper-level controller aims to improve QoS performance of services by enforcing adaption of slice configuration (i.e., loose control by restricting the data rate of users in each slice). Meanwhile, with the goal of improving the SE of slices, the lower-level controller performs the tight control by scheduling radio resources to active users at the small time-scale, where radio resource scheduling is conditioned by the slice configuration determined by the upper-level controller.
- Customizing a hierarchical DRL framework to realize the proposed RAN slicing control strategy. In the proposed strategy, the slice configuration made in the upper-level control will condition the lower-level controlling process; Meanwhile, the performance of the lower-level control will also affect the decision-making of the upper-level controlling process, which makes it very difficult to be implemented by conventional RL frameworks. Therefore, based on twin-timescale Markov decision process, we propose a model-free DRL framework with hierarchical structure, which is the convergence of deep deterministic policy gradient (DDPG) and double deep-Q-Network (Double-DQN). Firstly, the lower-level control policy is learned by a modified DDPG algorithm. Then, based on the well-performed lower-level control policy, the upper-level control policy is learned by the Double DQN algorithm.
- A joint utilization of DDPG algorithm and convex optimization tools for the real-time lower-level control. The lower-level control can be seen as a mixed integer stochastic optimization problem with multiple constraints. This kind of problem is hard to be solved by the exiting DRL algorithms directly, since it involves searching for the optimal solution in a vast set of mixed integer space (i.e., unbearable computational complexity). To ensure the timeliness of the lower-level control, a novel action space reducing approach is proposed. It can significantly reduce the lower-level control's action space size. Firstly, this approach converts the searching of the optimal solution into solving a pre-defined weighted sum-rate optimization problem, where the weights of UEs (i.e., continuous variable set) are determined by the DDPG algorithm. Under this setting, the pre-defined problem is solved by the Dual Lagrangian method and Karush-Kuhn-Tucker (KKT) conditions.

The rest of the paper is organized as follows. Section II introduces the system model. Section III presents the proposed hierarchical RAN resource slicing control strategy and formulates the target problem. In Section IV, based on the twin timescale Markov decision process, we propose a hierarchically structured DRL framework. Then in Section V, we present a system-level simulation of the proposed scheme. Finally, Section VI concludes the paper.

II. System Model

Consider a typical downlink cellular network system with a single Base Station (BS). The time dimension is partitioned into Transmission Time Interval (TTI) of 1 ms, indexed by $t \in \{1, 2, \ldots\}$. The bandwidth is divided into a set of physical resource blocks (PRBs), denoted as $\mathcal{F} = \{1, 2, \ldots, F\}$, for each TTI. Assume that the cellular network is split into a set $\mathcal{N} = \{1, 2, \ldots, N\}$ of network slices. Specifically, the UEs associated with slice $n \in \mathcal{N}$ is denoted as the set $\mathcal{U}_n$, where $\mathcal{U} = \cup_{\forall n} \mathcal{U}_n$. It is noteworthy that, in our system model, a UE can concurrently associate with multiple slices, which is consistent with actual use cases.

A. Signal Transmission Model

In conventional services, such as eMBB with large transmitted packet size, we can directly utilize Shannon's capacity

formulation to estimate data rate. However, unlike the conventional services, one distinctive feature of emerging uRLLC or MTC service is short-sized packet (ranging from 32 to 200 Bytes) transmission [27]. In the short packet transmission, the data rate cannot be accurately captured by Shannon's capacity theory. Instead, the achievable data rate of short packet transmission can be approximated by finite block-length theory [28]. Therefore, the data rate can be modeled as follows.

In network slice n, the achievable data rate of UE $i \in \mathcal{U}_n$ for PRB $j \in \mathcal{F}$ at the t-th TTI can be approximated as in formula (1) and (2), as shown at the bottom of the page, where $p_{i,j,t}$ is the transmit power on PRB j at the t-th TTI. Let $h_{i,j,t}$ be the channel coefficient, which contains path loss, shadowing effect and Rayleigh fast fading between UE i and the BS on PRB j. N_0 is the power of additive white Gaussian noise (AWGN) in each PRB. In formula (2), $Q^{-1}(\cdot)$ is the inverse of the Gaussian Q-function, ρ_n is the block-length of packet (i.e., number of transmit symbols), ε is the effective decoding error probability, and $V_{i,j,t}$ is the so-called channel dispersion, which depicts the stochastic variability of the channel relative to a deterministic channel with the same capacity, denoted as,

$$V_{i,j,t} = 1 - \left(1 + p_{i,j,t}|h_{i,j,t}|^2 \Big/ N_0\right)^{-2}. \tag{2a}$$

Hence, the instantaneous data rate of UE $i \in \mathcal{U}_n$ can be written as

$$r_{i,t} = \sum\nolimits_{j=1}^{F} s_{i,j,t} \cdot r_{i,j,t}, \quad (\text{bits per TTI}), \tag{3}$$

where binary variable $s_{i,j,t} = 1$ indicates the j-th PRB is allocated to UE i and $s_{i,j,t} = 0$, otherwise. Furthermore, each PRB can be allocated to at most one UE.

B. Queuing Model of UE

In slice n, the distribution of packet size and packet inter-arrival time determined by the pattern of service and network controller has no prior statistical information of service traffic pattern. It is assumed that each UE owns a data queue at the BS to buffer the arriving packets, in which the packet is delivered according to the first-come-first-serve (FCFS) policy. The queue length (in number of packets) of UE $i \in \mathcal{U}_n$ at the t-th TTI is denoted as $q_{i,t}$ and evolved as follows,

$$q_{i,t+1} = \max\{q_{i,t} - r_{i,t}/Z_n, 0\} + A_{i,t}, \tag{4}$$

where Z_n is the total packet size (in bits), and $A_{i,t}$ is the instantaneous packet arrival for UE i during the t-th TTI. Furthermore, the set of UEs with nonzero queue length ($\mathcal{U}_n^{\text{actv}} = \{i | q_{i,t} > 0, i \in \mathcal{U}_n\} \subseteq \mathcal{U}_n$) are defined as the active UEs in slice n.

Generally, the packet delay consists two parts, containing the transmission-related latency and the scheduling-related latency (queuing latency). Specifically, the transmission latency is decided by the data rate of UE in formula (3), while the queuing latency is decided by the scheduling policy. In our system model, the packet delay is defined as the summation of the transmission delay and the queuing delay. Thus, the delay of the m-th ($m = 1, 2, \cdots$) packet arriving at the i-th UE's buffer is modeled as

$$D_{i,m} = W_{i,m} + \delta_{i,m}, \quad (\text{in TTI}) \tag{5}$$

where queuing latency of the m-th packet is $W_{i,m}$ and the transmission latency is $\delta_{i,m}$. In the slice n, when the average packet arrival rate of UE is low, the queuing delay is close to zero, which makes that the packet delay is dominated by the transmission delay. however, with increasing the packet arrival rate of UE, the queuing delay will reach a high level due to the backlog of packets in the buffer, which makes the packet delay is determined by both the queuing latency and transmission latency.

From the view of service provisioning, applications identify one data packet as dropout, when the delay of this packet is exceeding a predefined maximum tolerant packet delay [28]. Normally, the communication reliability is determined by packet dropouts. Therefore, in the system model, the packet drop rate (PDR) of the m-th arriving packet at the i-th UE's buffer is defined as the probability that the packet delay is exceeding a preset maximum packet delay threshold. Then, the communication reliability expression of UE i is

$$\beta_{i,t} = \Pr\{D_{i,m} > D_n^{\max}\}, \quad i \in \mathcal{U}_n, \tag{6}$$

where $D_n^{\max}$ denotes the maximum tolerant packet delay of each UE in slice n.

In the RAN slicing, its overall goal is to optimize the QoS performance of service. In the service provisioning, the packet delay of UE, defined in formula (5), should be considered as one important metric of QoS. Meanwhile, the PDR of UE, defined in formula (6), quantifies the communication reliability, which should be considered as another important metric of QoS. Thus, in the next section, we will use the packet delay and PDR as two key metrics to evaluate the QoS performance of service.

III. Hierarchical Control Strategy and Problem Formulation

In this study, as an extension of our original work [7], a RAN slicing control strategy, denoted as $\pi = \{\pi_{\text{C}}, \pi_{\text{R}}\}$, is proposed. As shown in Figure 1, the upper-level controlling policy π_{C} is responsible for adjusting the configuration of

$$r_{i,j,t} = \begin{cases} \Delta t \cdot B \log_2\left(1 + \dfrac{p_{i,j,t}|h_{i,j,t}|^2}{N_0}\right), \text{ for long packet transmission,} & (1) \\ \Delta t \cdot B\left[\log_2\left(1 + \dfrac{p_{i,j,t}|h_{i,j,t}|^2}{N_0}\right) - \sqrt{\dfrac{V_{i,j,t}}{\rho_n}} \cdot \dfrac{Q^{-1}(\varepsilon)}{\ln 2}\right], \text{ for short packet transmission,} & (2) \end{cases}$$

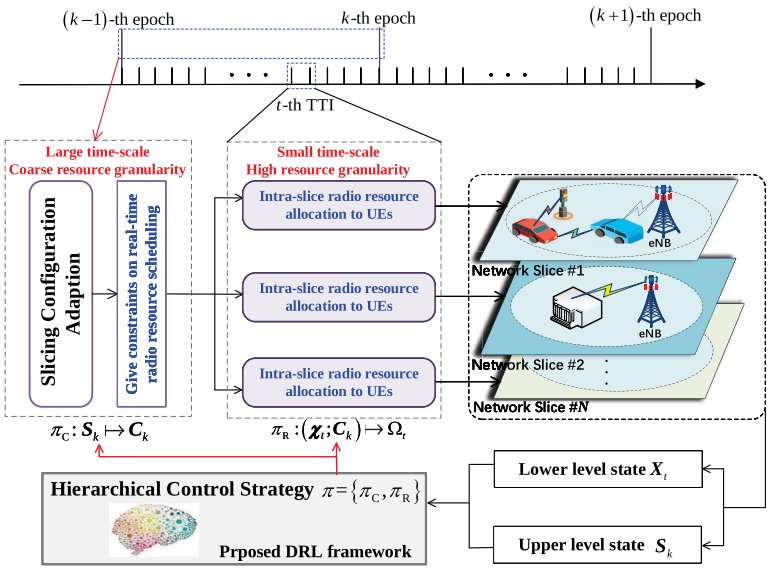


Fig. 1. Schematic of the intelligent RAN slicing control strategy. At the beginning of each epoch, the upper-level controller tunes the slice configuration (i.e., loose control by restricting the maximum/minimum data rate of UE in each slice) according to the dynamics of service traffic. On the other hand, conditioned by the slice configuration determined by the upper-level controller, the lower-level controller performs the scheduling of radio resource at each TTI.

slices according to the dynamics of service traffic at the large time-scale. Specifically, the basic time unit of upper-level control is defined as an epoch, indexed by $k \in \{1, 2, \ldots\}$, each epoch is corresponding to ΔT consecutive TTIs. Based on the configuration determined by the upper-level, the lower-level controlling policy π_{R} directly performs the radio resource scheduling according to the dynamics of physical layer at the small timescale.

A. Upper-Level Control

The upper-level controller will tune the slice configuration to improve the QoS performance of services. It is noteworthy that the upper-level controller is not directly involved in the real-time radio resource scheduling. Essentially, the upper-level control has two main functions,

- *Ensuring QoS Requirements of Slices*: It transforms the dynamics of service traffic as well as the QoS requirements of services into the data rate constraint of users in each slice. Here, the guaranteed bit rate (GBR) of UEs in each slice is used to ensure the QoS performance.
- *Ensuring Performance Isolation Among Slices*: It guarantees that the traffic overload in one slice does not negatively affect the QoS experienced by UEs in other slices. Here, each slice is imposed with the maximum bit rate (MBR) of UEs to ensure performance isolation.

At the beginning of each epoch, the upper-level control policy π_{C} obtains the upper-level state, which is the status of the service traffic and corresponding QoS performance. Specifically, the upper-level state of the entire system at the k-th epoch is defined as,

$$\boldsymbol{S}_k = \{\boldsymbol{S}_{n,k} \,|\forall n \in \mathcal{N}\} \in \mathcal{S},$$

where

$$\boldsymbol{S}_{n,k} = \left(\bar{A}_{n,k}, \bar{D}_{n,k}, \bar{\beta}_{n,k}\right), \quad n \in \mathcal{N},$$

is the upper-level state of slice n. Term $\bar{A}_{n,k}$ is the average packet arrival rate of UE, $A_{i,t}$, in slice n within the last epoch; $\bar{D}_{n,k}$ is average packet delay of active UEs, $D_{i,m}$ defined in formula (5), in slice n during the last epoch; and $\bar{\beta}_{n,k}$ is the average PDR of active UEs, $\beta_{i,t}$ defined in formula (6), in slice n within the last epoch.

Based on the upper-level state, the slice configuration adaption is determined as follows,

Definition 1 (Upper-Level Control Policy): Define the upper-level controlling policy π_{C} to be a mapping from the global state of the whole network $\boldsymbol{S}_k$ to a suitable slice configuration $\boldsymbol{C}_k$, which is given by

$$\pi_{\mathrm{C}}: \ \boldsymbol{S}_k \rightarrow \boldsymbol{C}_k,$$

where the slice configuration at k-th epoch is defined as

$$\boldsymbol{C}_k = \left\{R_{n,k}^{\min}, R_{n,k}^{\max} \,|\forall n \in \mathcal{N}\right\} \in \mathcal{C}.$$

Here, $R_{n,k}^{\min}$ is the Guaranteed Bit Rate (GBR) of active UE in slice n, which is used to ensure the QoS requirement of service, and $R_{n,k}^{\max}$ is the Maximum Bit Rate (MBR) of active UE in slice n, which can avoid that too many radio resources occupied by one slice. Besides, $\mathcal{C}$ is the finite set of all candidate slice configurations. After choosing slice configuration $\boldsymbol{C}_k$, the data rate of active UE in slice n is restricted by

$$R_{n,k}^{\min} \leq r_{i,t} \leq R_{n,k}^{\max}, \forall i \in \mathcal{U}_{n,t}^{\mathrm{actv}}, k\Delta T \leq t < (k+1)\,\Delta T, \tag{7}$$

where $\mathcal{U}_{n,t}^{\mathrm{actv}}$ is the set of active UEs in slice n at the t-th slot.

B. Lower-Level Control

During the k-th epoch (i.e., from TTI $k\Delta T$ to TTI $(k+1)\,\Delta T$), once the slice configuration $\boldsymbol{C}_k$ is chosen by the upper-level control policy π_{C}, then the radio resource allocation scheme at each TTI is restricted by constraint (7). Thus, the remaining problem is to allocate PRBs and transmit power to active UEs according to the lower-level state of the network at the t-th TTI, which is defined as

$$\boldsymbol{X}_t = \{\boldsymbol{X}_{n,t} \,|\forall n \in \mathcal{N}\} \in \mathcal{X},$$

where

$$\boldsymbol{X}_{n,t} = \{q_{i,t}, \boldsymbol{H}_{i,t} \,|\forall i \in \mathcal{U}_n\},$$

is the lower-level state of slice n at the t-th TTI. Term $q_{i,t}$ is the queue length of UE i and $\boldsymbol{H}_{i,t}$= $\{h_{i,j,t} \,|\forall j \in \mathcal{F}\}$ represents the channel state information of UE i, which is a vector the channel gain $h_{i,j,t}$ on PRB $j \in \mathcal{F}$. The details of the lower-level control policy π_{R} are described as follows,

Definition 2 (Lower-Level Control Policy): Define the lower-level control policy π_{R} to be a mapping from the lower-level state of the network $\boldsymbol{X}_t$ to the PRB and power allocation of active UEs in each slice, which is expressed as

$$\pi_{\mathrm{R}} : (X_t; C_k) \rightarrow \Omega_t,\ k\Delta T \leq t < (k+1)\,\Delta T,$$

where Ω_t is the radio resource allocation scheme for slice n, which is given by

$$\Omega_t = \left\{s_{i,j,t}, p_{i,j,t} \,\middle|\forall n \in \mathcal{N}, i \in \mathcal{U}_{n,t}^{\mathrm{actv}}, j \in \mathcal{F}\right\}.$$

Remark 1 (High Flexibility of the Proposed RAN Slicing Strategy): Different from the existing RAN slicing strategies that assigns a fixed number of PRBs to each slice, the proposed strategy enables dynamic sharing of the PRBs between slices in each TTI according to the network situations, which can significantly improve the radio resource utilization efficiency.

C. Problem Formulation

In this study, we set the comprehensive utilization of the network slice, which is related to the QoS performance of service and the spectrum efficiency (SE) of the network slice. Then, the utility function of slice n at the k-th epoch is given by

$$U_{n,k} = U_{n,k}^{\mathrm{QoS}}\left(S_{n,k}\right) + \sum\nolimits_{t=k\Delta T+1}^{(k+1)\Delta T} U_{n,t}^{\mathrm{SE}}\left(X_{n,t}, \Omega_t\right), \quad (8)$$

where $U_{n,k}^{\mathrm{QoS}}(\cdot)$ is the QoS satisfaction of slice n, which is measured by the average packet latency and the average PDR of active UEs in slice n,

$$U_{n,k}^{\mathrm{QoS}}\left(\boldsymbol{S}_{n,k}\right) = \alpha_{n,1} \cdot \exp\left(-\bar{D}_{n,k}\right) + \alpha_{n,2} \cdot \exp\left(-\bar{\beta}_{n,k}\right), \quad (8a)$$

To better stabilize the training procedure of the DRL frameworks developed in the next section, we choose the exponential function to compose function $U_{n,k}^{\mathrm{QoS}}(\cdot)$, whose value does not diverge dramatically [26]. On the other hand, the SE of slice n, $U_{n,k}^{\mathrm{SE}}(\cdot)$, is defined as the normalized average data rate of UEs during k-th epoch,

$$U_{n,t}^{\mathrm{SE}}\left(X_{n,t}, \Omega_t\right) = \alpha_{n,3} \cdot \frac{1}{\Delta T\left|\mathcal{U}_{n,t}^{\mathrm{actv}}\right|} \cdot \sum\nolimits_{i \in \mathcal{U}_{n,t}^{\mathrm{actv}}} r_{i,t}, \quad (8b)$$

Besides $\alpha_{n,1}$, $\alpha_{n,2}$ and $\alpha_{n,3}$ are positive weighting factors of slice n, which denotes the importance of average packet latency, average PDR of UE and SE, respectively.

The objective of this paper is to obtain the optimal RAN slicing control strategy, which can maximize the expected long-term utility function of all network slices while fulfilling the constraints on radio resource scheduling. Mathematically, we have

$$\mathcal{P}: \max_{\pi=\{\pi_{\mathrm{C}}, \pi_{\mathrm{R}}\}} \left\{J(\pi) = \left[\sum\nolimits_{k=0}^{\infty} \lambda^k \cdot \left(\sum\nolimits_{n \in \mathcal{N}} U_{n,k}\right)\right]\right\}$$

$$\text{subject to :}$$

$$\mathrm{C1}: \sum\nolimits_{n \in \mathcal{N}} \sum\nolimits_{i \in \mathcal{U}_n^{\mathrm{actv}}} s_{i,j} \leq 1,\ \forall j \in \mathcal{F},$$

$$\mathrm{C2}: \sum\nolimits_{n \in \mathcal{N}} \sum\nolimits_{i \in \mathcal{U}_n^{\mathrm{actv}}} \left(\sum\nolimits_{j \in \mathcal{F}} s_{i,j} p_{i,j}\right) \leq P_{\max},$$

$$\mathrm{C3}: p_{i,j,t}|h_{i,j,t}|^2/N_0 \geq s_{i,j,t} \cdot \gamma_{\mathrm{TH}}, \forall i \in \mathcal{U}_n^{\mathrm{actv}}, j \in \mathcal{F},\ n \in \mathcal{N},$$

$$\mathrm{C4}: R_{n,k}^{\min} \leq r_{i,t} \leq R_{n,k}^{\max},\ \forall i \in \mathcal{U}_{n,t}^{\mathrm{actv}}, k\Delta T \leq t < (k+1)\,\Delta T,$$

$$\mathrm{C5}: s_{i,j} \in \{0,1\},\ \forall i \in \mathcal{U}_n^{\mathrm{actv}}, j \in \mathcal{F},\ n \in \mathcal{N},$$

where the objective function $J(\pi)$ of the target problem $\mathcal{P}$ is an infinite horizon discounted reward, λ is a discount factor and λ^k approaches to zero when k is large enough. Constraint C1 and C2 limit the PRB resources and total transmit power of BS, respectively. Constraint C3 ensures the signal-to-noise-ratio (SNR) of each transmission above a predefined SNR threshold, denoted as γ_{TH}. Constraint C3 can enable that the UE decodes the signal with high reliability. Constraint C4, i.e., constraint (7), ensures that the data rate $r_{i,t}$ of UE $i \in \mathcal{U}_{n,t}^{\mathrm{actv}}$ is between the GBR (guaranteed bit rate) and MBR (maximum bit rate). It is noteworthy that constraint C4 is determined by the slice configuration $\boldsymbol{C}_k$. The last constraint ensures binary-valued $s_{i,j}$.

D. Twin Timescale Markov Decision Process

In this study, twin time-scale Markov Decision Process (MDP) can better model and characterize the target problem $\mathcal{P}$ [29]. In essence, the upper-level control process is an infinite horizon MDP at the large timescale, while the lower-level controlling is a finite (ΔT-TTIs) horizon MDP at the small timescale. The details are described as follows.

Definition 3 (Action-Value Function of Lower-Level Control): During the k-th epoch, under a given slice configuration $\boldsymbol{C}_k \in \mathcal{C}$ and lower-level control policy π_{R}, the action-value function (Q function) is defined in (9a), as shown at the bottom of the next page, where $\tau_{>t} = \left(\boldsymbol{X}_t, \Omega_t, \ldots, \boldsymbol{X}_{(k+1)\Delta T}, \Omega_{(k+1)\Delta T}\right)$ denotes a sample trajectory of lower-level states and actions (i.e., radio resource

allocation schemes) after t-th TTI. Furthermore, under the lower-level control policy π_{R} and slice configuration $\boldsymbol{C}_k \in \mathcal{C}$, the expected cumulative reward of lower-level control during the k-th epoch is defined as

$$J_{\mathrm{L}}\left(\pi_{\mathrm{R}};\boldsymbol{C}_k\right)=\mathbb{E}_{\tau}\left[Q_{\mathrm{L}}\left(\boldsymbol{X}_{k\Delta t},\Omega_{k\Delta t};\boldsymbol{C}_k\right)|\boldsymbol{C}_k\right], \tag{9b}$$

where $\tau=\left(\boldsymbol{X}_{k\Delta T},\Omega_{k\Delta T},\ldots,\boldsymbol{X}_{(k+1)\Delta T},\Omega_{(k+1)\Delta T}\right)$ is a sample trajectory of lower-level states and actions during k-th epoch.

Definition 4 (Action-Value Function of Upper-Level Control): Under the given policy π and the upper-level state $\boldsymbol{S}_k$, the Q-function of the upper-level control is defined in (10a), as shown at the bottom of the page, where $\tau_{\mathrm{U}}=(\boldsymbol{S}_k,\boldsymbol{C}_k,\boldsymbol{S}_{k+1},\boldsymbol{C}_{k+1},\ldots)$ is a full sample trajectory of upper-level states and actions (i.e., slice configuration). Then, the value function of the upper-level control is defined as

$$V_{\mathrm{U}}\left(\boldsymbol{S}_k\right)=\mathbb{E}_{\boldsymbol{C}_k}\left[Q_{\mathrm{U}}\left(\boldsymbol{S}_k,\boldsymbol{C}_k\right)|\pi_{\mathrm{R}}\right] \tag{10b}$$

Remark 2 (The Nested Structure of RAN Slicing Strategy $\pi=\{\pi_{\mathrm{C}},\pi_{\mathrm{R}}\}$): From ***Definition 3***, the slice configuration $\boldsymbol{C}_k$ defined by the upper-level control policy π_{C} will condition on the radio resource allocation Ω_t $(k\Delta T\le t<(k+1)\Delta T)$ determined by the lower-level control policy π_{R}. Meanwhile, from ***Definition 4***, the value function of the upper-level control depends on the lower-level control policy π_{R}.

Based on ***Definition 3*** and ***Definition 4***, the objective function $J(\pi)$ of target problem $\mathcal{P}$ can be rewritten as (11), shown at the bottom of the page.

Unfortunately, twin timescale MDP is difficult in general [29]. Herein, We want to emphasize that the target problem $\mathcal{P}$ is particularly hard to solve due to

Challenge 1 (Convergence Issue in the Training of RAN Slicing Strategy π): When the upper-level control policy π_{C} and the lower-level control policy π_{R} are trained simultaneously, the transition probabilities between the upper-level states and the distribution of the objective function $J(\pi)$ will continue to change if the lower-level control policy π_{R} continues to be updated. Thus, the existing DRL algorithms will struggle to learn the RAN slicing control strategy π, due to the non-stationary distribution of upper-level states and objective function $J(\pi)$.

IV. Solution Based on Hierarchical Deep Reinforcement Learning Framework

As discussed in ***Challenge 1***, target problem $\mathcal{P}$ is challenging to solve. Hence, in this section, a hierarchically structured DRL framework is proposed, in which the lower-level control policy and the upper-level control policy are learned separately in two stages. Firstly, in stage I, the converged lower-level policy is leaned by joint use of the convex optimization tools and the policy gradient method. In stage II, with the converged lower-level control policy, the upper-level control policy is obtained based on the Double-DQN. Finally, we present the implementation of the proposed hierarchical DRL framework.

A. Hierarchical DRL Framework for Solving Problem $\mathcal{P}$

Learning the proposed RAN slicing control strategy π is still possible as the distributions of the upper-level states will be stabilized under the converged lower-level policy π_{R}. Furthermore, according to the objective function $J(\pi)$ in formula (11), we can decouple the target problem $\mathcal{P}$ into two parts: i). the optimization of the lower-level control policy; and ii). the optimization of the upper-level control policy. Followed this idea, we propose a hierarchical DRL framework, which is comprised of two stages.

Stage I (Learning of the Lower-Level Control Policy): Learning the converged lower-level policy π_{R}^{*} under all candidate slice configuration $\boldsymbol{C}_k\in\mathcal{C}$, which can be achieved by solving the following problem,

$$\begin{aligned}\mathcal{P}_1:\ \pi_{\mathrm{R}}^{*}=\ & \arg\max_{\pi_{\mathrm{R}}}\left\{J_{\mathrm{L}}\left(\pi_{\mathrm{R}};\boldsymbol{C}_k\right)|\boldsymbol{C}_k\in\mathcal{C}\right\}\\ & \text{subject to}:\ \mathrm{C1}-\mathrm{C5},\end{aligned}$$

where function $J_{\mathrm{L}}\left(\pi_{\mathrm{R}};\boldsymbol{C}_k\right)$, defined in formula (9b) of ***Definition 3***, represents the expected SE of all network slices in the k-th epoch under lower-level control policy π_{R} and slice configuration $\boldsymbol{C}_k$. Besides, the constraints of problem $\mathcal{P}_1$ are the same as the target problem $\mathcal{P}$.

Problem $\mathcal{P}_1$ is a constrained MDP with huge mixed-integer action space. To fit problem $\mathcal{P}_1$ into a typical MDP and reduce the computational complexity, a novel action space reducing approach is proposed based on convex optimization tools. Then, the converged lower-level control policy π_{R}^{*} is obtained based on a modified Deep Deterministic Policy Gradient (DDPG) algorithm (Section IV-B).

Stage II (Learning of the Upper-Level Control Policy): Then, with the converged lower-level policy π_{R}^{*}, learning of the upper-level policy π_{C} can be realized by solving the reduced version of the target problem $\mathcal{P}$, which is

$$\mathcal{P}_2:\pi_{\mathrm{C}}^{*}=\arg\max_{\pi_{\mathrm{C}}}\left\{V_{\mathrm{U}}\left(\boldsymbol{S}_0\right)|\pi_{\mathrm{R}}^{*}\right\}.$$

$$Q_{\mathrm{L}}\left(\boldsymbol{X}_t,\Omega_t;\boldsymbol{C}_k\right)=\mathbb{E}_{\tau_{>t}}\left[\sum\nolimits_{t'=t}^{(k+1)\Delta T-1}\sum\nolimits_{n\in\mathcal{N}}U_n^{\mathrm{SE}}\left(\boldsymbol{X}_{n,t'},\Omega_{t'}\right)\middle|\boldsymbol{X}_t,\Omega_t,\boldsymbol{C}_k\right],k\Delta T\le t<(k+1)\Delta T, \tag{9a}$$

$$Q_{\mathrm{U}}\left(\boldsymbol{S}_k,\boldsymbol{C}_k\right)=\mathbb{E}_{\tau_{\mathrm{U}}}\left[\sum\nolimits_{k'=k}^{\infty}\lambda^{k'-k}\cdot\left(J_{\mathrm{L}}\left(\pi_{\mathrm{R}};\boldsymbol{C}_{k'}\right)+\sum\nolimits_{n\in\mathcal{N}}U_{n,k'}^{\mathrm{QoS}}\right)\middle|\boldsymbol{C}_k,\pi_{\mathrm{R}}\right], \tag{10a}$$

$$J\left(\pi\right)=\mathbb{E}\left[Q_{\mathrm{U}}\left(\boldsymbol{S}_1,\boldsymbol{C}_1\right)|\pi_{\mathrm{C}},\pi_{\mathrm{R}}\right]=\mathbb{E}_{\tau_{\mathrm{U}}}\left[\underbrace{\sum\nolimits_{k=0}^{\infty}\lambda^{k}\cdot\sum\nolimits_{n=1}^{N}U_{n,k}^{\mathrm{QoS}}}_{\text{QoS of slices (upper level control)}}+\underbrace{\sum\nolimits_{k=0}^{\infty}\lambda^{k}\cdot J_{\mathrm{L}}\left(\pi_{\mathrm{R}};\boldsymbol{C}_k\right)}_{\text{SE of slices (lower level control)}}\right]. \tag{11}$$

Since the unconstrained problem $\mathcal{P}_2$ is a standard MDP problem, the Double DQN algorithm can be utilized to solve problem $\mathcal{P}_2$ (Section IV-C).

B. Stage I–Desgin and Learning of Lower-Level Control Policy

In problem $\mathcal{P}_1$, to learn the lower-level control policy π_{R}, one main challenge arises as follows:

Challenge 2 (Vast Mixed Integer Action Space of the Lower-Level Control): In the lower-level control, the action space, i.e., the feasible solution space of radio resource allocation scheme Ω_t, is extremely huge. This makes problem $\mathcal{P}_1$ a mixed integer stochastic optimization problem with multiple constraints $\mathrm{C1}-\mathrm{C5}$. It makes problem $\mathcal{P}_1$ hard to be directly solved by the existing model-free DRL algorithms.

1) Reducing Action Space of Lower-Level Control: The radio resource allocation scheme Ω_t consists of the following $|\mathcal{U}| \times F$ PRB allocation matrix and $|\mathcal{U}| \times F$ power allocation matrix. Therefore, the size of the mixed-integer action space for the lower-level control is $\Omega_t \in \mathcal{O}\left(|\mathcal{U}|^F\right) \times \mathbb{R}_F$. As we can see, it is infeasible to search for the optimal action Ω_t^* in such a huge action space. ***Challenge 2*** will severely deteriorate the learning performance of conventional DRL algorithms, especially for the deep Q-learning algorithms, which needs to search the action space to obtain the optimal action directly.

To address ***Challenge 2***, similar to the work in [30], we propose an action space reducing approach, which can significantly reduce the size of action space from $\mathcal{O}\left(F^{|\mathcal{U}|}\right) \times \mathbb{R}^F$ to $\mathbb{R}^{|\mathcal{U}|}$. Here, we define the reduced action space as the scheduling priority factor of UEs $\boldsymbol{w}_t \in \mathbb{R}^{|\mathcal{U}|}$. Specifically, *the action space reducing approach* converts the searching for the suitable action Ω_t in the original mixed-integer action space into solving an action space reducing sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$. Because problem $\mathcal{P}_1$ aims at maximizing the long-term SE of all slices in the k-th epoch, we define sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ as a weighted sum-rate optimization problem with respect to the data rate of all active UEs in the network, $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t) : \Omega_t$, as shown at the bottom of the page, where $w_i \in \mathbb{R}^+$ is the scheduling weight factor of UE $i \in \mathcal{U}$. The scheduling weight factor of UEs $\boldsymbol{w}_t$ are determined by the lower-level control policy π_{R}, which is a deterministic policy.

Remark 3 (Practical Value of the Proposed Action Space Reducing Approach): Firstly, the lower-level control policy π_{R} maps the lower-level state of the whole network $\boldsymbol{X}_t$ into the reduced action space (i.e., the scheduling priority factor of UEs $\boldsymbol{w}_t \in \mathbb{R}^{|\mathcal{U}|}$). Then, we solve sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ to obtain the PRB and transmit power allocation Ω_t.

$$\pi_{\mathrm{R}}(\boldsymbol{X}_t;\boldsymbol{C}_k) = \boldsymbol{w}_t \rightarrow \text{Solve Problem } \mathcal{P}_1^{\mathrm{R}}(w_t) \rightarrow \Omega_t. \quad (12)$$

Notably, we want to highlight two advantages of the action space reducing approach in the following,

- The lower-level control policy π_{R} only needs to adjust the scheduling priorities of UEs, i.e., $\boldsymbol{w}_t \in \mathbb{R}^{|\mathcal{U}|}$ to maximize the expected SE of all slices in the epoch, i.e., $J_{\mathrm{L}}(\pi_{\mathrm{R}};\boldsymbol{C}_k)$;
- The action space reducing sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ is a deterministic optimization problem, which can be solved by the conventional convex optimization tools.

Thus, now the key point is to solve sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ and derive the optimal solution. Although the non-convexity of the short packet transmission rate (2), under the proper setting of SNR threshold γ_{TH} in constraint C3, the action space reducing sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ can be approximated as a convex optimization problem, which satisfies Slater's condition. The dual decomposition method can guarantee the optimal solution as the strong duality holds for this case. Therefore, we utilize the dual decomposition method and Karush-Kuhn-Tucker (KKT) optimality conditions [31] to obtain the optimal solution to the sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$.

Proposition 1 (PRB and Transmit Power Allocation): In this paper, we set the SNR threshold γ_{TH} in constraint C3 greater than 10 dB, in order to ensure high SNR of UEs. The optimal PRB and power allocation scheme for sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ can be decided based on the following criterion,

$$p_{i,j}^* = \begin{cases} \Lambda_{i,j}, & \text{if} \Lambda_{i,j} \geq \gamma_{\mathrm{TH}} \cdot N_0 / |h_{i,j}|^2 \\ 0, & \text{otherwise}, \end{cases} \quad (13a)$$

$$s_{i,j}^* = \begin{cases} 1, & i = \arg\ \max_i \{\Xi_{i,j}\} \\ 0, & \text{otherwise} \end{cases} \quad (13b)$$

where

$$\Lambda_{i,j} = \frac{B\Delta t}{\xi \ln 2} \cdot (w_i + \eta_i - \nu_i) - \frac{1}{|h_{i,j}|^2} \cdot N_0,$$

$$\Xi_{i,j} = r_{i,j}(w_i + \eta_i - \nu_i) - \beta p_{i,j}^*, \forall i \in \mathcal{U}_n^{\mathrm{actv}}, n \in \mathcal{N}, j \in \mathcal{F},$$

and $\boldsymbol{\eta} = \{\eta_i \,|\forall i\}$ and $\boldsymbol{\nu} = \{\nu_i \,|\forall i\}$ are the positive Lagrange multiplier vectors associated with the GBR (guaranteed bit rate) and MBR (maximum bit rate) constraints in C4, i.e., (7), and ξ is the Lagrange multiplier with power constraint C2. An iterative sub-gradient method in [31] is used to update these Lagrange multipliers,

$$\eta_i \leftarrow \eta_i - \phi \cdot \left(r_i - R_{n,k}^{\min}\right), \quad \forall i \in \mathcal{U}_n^{\mathrm{actv}}, \quad (13c)$$

$$\nu_i \leftarrow \nu_i - \phi \cdot \left(R_{n,k}^{\max} - r_i\right), \quad \forall i \in \mathcal{U}_n^{\mathrm{actv}}, \quad (13d)$$

$$\xi \leftarrow \xi - \phi \cdot \left(P^{\max} - \sum_{n \in \mathcal{N}} \sum_{i \in \mathcal{U}_n^{\mathrm{actv}}} \sum_{j \in \mathcal{F}} s_{i,j}^* \cdot p_{i,j}^*\right), \quad (13e)$$

where $\phi \ll 0$ is the iteration step size. The process of updating the PRB and power allocation, and Lagrange multipliers is repeated until convergence or a predefined maximum number of iterations. The asymptotic computational complexity can be shown to be of $\mathcal{O}(|\mathcal{U}|F)$.

Proof: Detailed derivations are given in **Appendix A**. ■

$$\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t) : \Omega_t = \arg\max_{\Omega_t} \left\{ \sum_{n \in \mathcal{N}} \sum_{i \in \mathcal{U}_n^{\mathrm{actv}}} w_{i,t} \cdot r_{i,t} \,|\boldsymbol{C}_k,\ \boldsymbol{w}_t \right\},\ \text{subject to} : \mathrm{C1}-\mathrm{C5},\ k\Delta T \leq t < (k+1)\Delta T.$$

2) Learning Lower-Level Control Policy by Policy Gradient Method: After reducing the original mixed-integer action space to the continuous space $\mathbb{R}^{|\mathcal{U}|}$, Deep Deterministic Policy Gradient (DDPG) can effectively deal with problem $\mathcal{P}_1$, since DDPG is explicitly adapted to RL problems with continuous action spaces [32]. While the Deep Q-Network (DQN) can only work in the discrete action space, DDPG extends DQN to continuous action space with the actor-critic framework. In DDPG, it uses the Bellman equation to learn the Q-function and utilizes it to learn a deterministic control policy. In this paper, based on the twin timescale MDP, a modified DDPG algorithm is proposed.

Firstly, the optimal Q-function of the lower-level control, denoted as Q_{L}^{*}, can be straightforwardly obtained from the Bellman equation, which is given by

$$\begin{aligned} &Q_{\mathrm{L}}^{*}\left(\boldsymbol{X}_t, \Omega_t; \boldsymbol{C}_k\right) \\ &= \mathbb{E}\left[\sum\nolimits_{n\in\mathcal{N}} U_n^{\mathrm{SE}}\left(\boldsymbol{X}_{n,t}, \Omega_t\right) + Q_{\mathrm{L}}^{*}\left(\boldsymbol{X}_{t+1}, \Omega_{t+1}; \boldsymbol{C}_k\right)\right], \\ &\quad k\Delta T \le t < (k+1)\,\Delta T. \end{aligned} \tag{14}$$

Furthermore, a deep Q network (DQN), denoted as $\hat{Q}_{\mathrm{L}}$, is used to approximate the Q function of the lower-level control,

$$Q_{\mathrm{L}}\left(\boldsymbol{X}_t, \Omega_t; \boldsymbol{C}_k \,|\theta\right) \approx \hat{Q}_{\mathrm{L}}\left(\boldsymbol{X}_t, \boldsymbol{w}_t; \boldsymbol{C}_k \,|\boldsymbol{\theta}\right),$$

where vector $\boldsymbol{\theta}$ stands for the weights of DQN $\hat{Q}_{\mathrm{L}}$.

Then, based on Bellman equation (14), $\hat{Q}_{\mathrm{L}}\left(\boldsymbol{X}_t, \boldsymbol{w}_t; \boldsymbol{C}_k \,|\boldsymbol{\theta}\right)$ is trained by updating weights $\boldsymbol{\theta}$ at each learning step to approximate the real optimal Q-function of the lower-level control, Q_{L}^{*}. Thus, we set up a mean-squared error (MSE) loss function, which can measure how closely the DQN $\hat{Q}_{\mathrm{L}}$ comes to satisfying the Bellman equation (14):

$$L\left(\boldsymbol{\theta}\right) = \frac{1}{2}\cdot\mathbb{E}\left[\left(y_{\mathrm{L}}^{\mathrm{target}} - \hat{Q}_{\mathrm{L}}\left(\boldsymbol{X}_t, \boldsymbol{w}_t; \boldsymbol{C}_k \,|\boldsymbol{\theta}\right)\right)^2\right], \tag{15}$$

where

$$y_{\mathrm{L}}^{\mathrm{target}} = \sum_{n\in\mathcal{N}} U_n^{\mathrm{SE}}\left(\boldsymbol{X}_{n,t}, \Omega_t\right) + \hat{Q}_{\mathrm{L}}\left(\boldsymbol{X}_{t+1}, \boldsymbol{w}_{t+1}; \boldsymbol{C}_k \,|\boldsymbol{\theta}_{-}\right)$$

is referred to as the target Q value of lower-level control. In the learning procedure, the DDPG algorithm adopts a "target neural network" with weights $\boldsymbol{\theta}_{-}$, which is a copy of DQN $\hat{Q}_{\mathrm{L}}$, to calculate the target Q value of the lower-level control. The target neural network can enable the learning procedure of DDPG more stable. Furthermore, in each learning step, the weights of the target neural network $\boldsymbol{\theta}_{-}$ are updated by having them slowly track the weights $\boldsymbol{\theta}$, i.e., $\boldsymbol{\theta}_{-} \leftarrow (1-\iota)\,\boldsymbol{\theta}_{-} + \iota\boldsymbol{\theta}$ with $\iota \ll 0$. Then, the weights $\boldsymbol{\theta}$ of the DQN $\hat{Q}_{\mathrm{L}}$ can be updated by performing stochastic gradient descent (SGD) to minimize the loss function $L\left(\boldsymbol{\theta}\right)$.

Secondly, the lower-level control policy $\pi_{\mathrm{R}}\left(\boldsymbol{X}_t; \boldsymbol{C}_k\right)$ is represented by a deep neural network,

$$\pi_{\mathrm{R}}\left(\boldsymbol{X}_t; \boldsymbol{C}_k \,|\boldsymbol{\varphi}\right) = \boldsymbol{w}_t,$$

where $\boldsymbol{\varphi}$ represents the weights of the neurla network.

To obtain the optimal lower-level control policy π_{R}^{*}, the modified DDPG algorithm is to adjust the weights $\boldsymbol{\varphi}$ in the direction of the gradient of function $J_{\mathrm{L}}\left(\pi_{\mathrm{R}}; \boldsymbol{C}_k\right)$ in $\mathcal{P}_1$.

Proposition 2 (DDPG for the Twin Timescale MDP): Based on the principle of DDPG in [32], the gradient of the function $J_{\mathrm{L}}\left(\pi_{\mathrm{R}}; \boldsymbol{C}_k\right)$ in problem $\mathcal{P}_1$ with respect to the weights $\boldsymbol{\varphi}$ is given by

$$\begin{aligned} &\nabla_{\boldsymbol{\varphi}} J_{\mathrm{L}}\left(\pi_{\mathrm{R}}; C_k\right) = \\ &\mathbb{E}\left[\nabla_{\boldsymbol{w}} \hat{Q}_{\mathrm{L}}\left(\boldsymbol{X}_t, \boldsymbol{w}; \boldsymbol{C}_k\right)\big|_{\boldsymbol{w}=\pi_{\mathrm{R}}\left(\boldsymbol{X}_t; \boldsymbol{C}_k|\boldsymbol{\varphi}\right)} \nabla_{\boldsymbol{\varphi}} \pi_{\mathrm{R}}\left(\boldsymbol{X}_t; \boldsymbol{C}_k \,|\boldsymbol{\varphi}\right)\right] \\ &k\Delta T \le t < (k+1)\,\Delta T, \quad \boldsymbol{C}_k \in \mathcal{C}, \end{aligned} \tag{16}$$

where the DQN $\hat{Q}_{\mathrm{L}}$ is substituted for the real Q-function of the lower-level control and DQN $\hat{Q}_{\mathrm{L}}$ is differentiable with respect to the scheduling priority of UEs, $\boldsymbol{w} \in \mathbb{R}^{|\mathcal{U}|}$. Moreover, it is noteworthy that the weights $\boldsymbol{\theta}$ of the DQN $\hat{Q}_{\mathrm{L}}$ are treated as constants in formula (16).

Remark 4 (Advantage of Proposition 2): Although the distribution of lower-level state $\boldsymbol{X}_t$ depends on the lower-level control policy π_{R} and the slice configuration $\boldsymbol{C}_k$, the policy gradient $\nabla_{\boldsymbol{\varphi}} J_{\mathrm{L}}\left(\pi_{\mathrm{R}}; C_k\right)$ in formula (16) can be calculated based on Monte Carlo methods. It means that Proposition 2 does not need the detailed knowledge of the upper-level control policy π_{C}, including the transition probability between upper-level states, or the policy to choose $\boldsymbol{C}_k$.

Based on ***Proposition 2***, the weights $\boldsymbol{\varphi}$ of lower-level control policy π_{R} can be adjusted by using stochastic gradient ascent. Then, the converged policy π_{R}^{*} can be obtained. The learning procedure of the lower-level control policy π_{R} is summarized in **Algorithm 1**. Furthermore, DDPG is an off-policy algorithm. It means that we can maintain an "experience replay buffer," which contains "old experiences" (i.e., samples of lower-level control), even though these "experiences" are collected using an outdated lower-level control policy.

C. Stage II–Desgin and Learning of Upper-Level Control Policy

Under the converged lower-level control policy π_{R}^{*}, problem $\mathcal{P}_2$ is a typical infinite-horizon discrete-time MDP with discrete action space. Here, we restore to using double DQN algorithm to solve problem $\mathcal{P}_2$.

The optimal upper-level control policy denoted as π_{R}^{*}, can be obtained from solving the Bellman's equation of Q-function of the upper-level control (formula (10a) of ***Definition 4***),

$$\begin{aligned} &Q_{\mathrm{U}}\left(\boldsymbol{S}_k, \boldsymbol{C}_k\right) \\ &\quad = \mathbb{E}\left[\sum\nolimits_{n\in\mathcal{N}} U_{n,k} + \gamma\cdot\max\nolimits_{C'}\left\{Q_{\mathrm{U}}\left(\boldsymbol{S}_{k+1}, \boldsymbol{C}'\right)\right\}\right] \end{aligned} \tag{17}$$

where $\boldsymbol{C}'$ is the resulting slice configuration at the subsequent epoch.

In the double DQN, the Q function of upper-level control Q_{U} is represented by a DQN,

$$Q_{\mathrm{U}}\left(\boldsymbol{S}_k, \boldsymbol{C}_k\right) \approx \hat{Q}_{\mathrm{U}}\left(\boldsymbol{S}_k, \boldsymbol{C}_k \,|\boldsymbol{\psi}\right), \quad \boldsymbol{C}_k \in \mathcal{C},$$

where $\boldsymbol{\psi}$ stands for the weights associated with the DQN $\hat{Q}_{\mathrm{U}}$. The DQN $\hat{Q}_{\mathrm{U}}\left(\boldsymbol{S}_k, \boldsymbol{C}_k \,|\boldsymbol{\psi}\right)$ is trained towards the target value by minimizing the MSE loss function $L\left(\boldsymbol{\psi}\right)$ at each learning

Algorithm 1 DDPG-based learning procedure of the lower-level control policy π_{R}

Initialization:

- Initialize the DQN $\hat{Q}_{\mathrm{L}}$ and the lower-level control policy π_{R} with the weights $\boldsymbol{w}$ and $\boldsymbol{\varphi}$.
- Set weights of target neural networks to the main weights, i.e., $\boldsymbol{\varphi}_{-} \leftarrow \boldsymbol{\varphi}$ and $\boldsymbol{\theta}_{-} \leftarrow \boldsymbol{\theta}$.

Repat:

1) Observe lower-level state $\boldsymbol{X}_t$ and generate the scheduling priority factor of active UEs $\boldsymbol{w}_t$.

2) Obtain radio resource allocation scheme Ω_t by solving the action space reducing sub-problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ based on ***Proposition 1*** and observe the new lower-level state $\boldsymbol{X}_{t+1}$.

3) Store sample data $\left(\boldsymbol{X}_t, \boldsymbol{w}, \sum_{n=1}^{N} U_n^{\mathrm{SE}}, \boldsymbol{X}_{t+1}\right)$ in the "experience replay buffer".

4) Get a random minibatch of samples form the replay buffer.

5) Update the weights $\boldsymbol{\theta}$ of the DQN $\hat{Q}_{\mathrm{L}}$ by performing stochastic gradient descent (SGD) to minimize the loss function $L(\boldsymbol{\theta})$ in formular (15).

6) Update the weights $\boldsymbol{\varphi}$ of the lower-level control policy π_{R} in ***Proposition 2***.

7) Update the target neural networks: $\boldsymbol{\theta}_{-} \leftarrow (1-\iota)\boldsymbol{\theta}_{-} + \iota\boldsymbol{\theta}$ and $\boldsymbol{\varphi}_{-} \leftarrow (1-\iota)\boldsymbol{\varphi}_{-} + \iota\boldsymbol{\varphi}$.

until convergence

step. Like the loss function defined in formula (15), $L(\boldsymbol{\psi})$ can be defined as

$$L(\boldsymbol{\psi}) = \frac{1}{2} \cdot \mathbb{E}\left[\left(y_{\mathrm{U}}^{\mathrm{target}} - \hat{Q}_{\mathrm{U}}(\boldsymbol{S}_k, \boldsymbol{C}_k | \boldsymbol{\psi})\right)^2\right]. \quad (18)$$

In the basic Q-learning, both choosing an action and evaluating the chosen action use the same Q-Network, which leads to the over-optimistic estimation of the real Q function of upper-level control. Therefore, to relieve the over-estimation issue, the target Q-Network is adopted, and the target Q value of upper-level control is defined as

$$\begin{aligned} & y_{\mathrm{U}}^{\mathrm{target}} \\ &= \sum\nolimits_{n \in \mathcal{N}} U_{n,k} \\ &\quad + \lambda \cdot \hat{Q}_{\mathrm{U}}\left(\boldsymbol{S}_{k+1}, \arg\max_{\boldsymbol{C}'}\left\{\hat{Q}_{\mathrm{U}}(\boldsymbol{S}_{k+1}, \boldsymbol{C}' | \boldsymbol{\psi})\right\} \middle| \boldsymbol{\psi}_{-}\right), \end{aligned}$$

where $\boldsymbol{\psi}_{-}$ is the weights of the target DQN. In formula (18), the selection of slice configuration $\boldsymbol{C}'$ for the $(k+1)$-th epoch, i.e., the right-hand side of (18), is generated from the target DQN, which can significantly reduce the over-estimation issues of Q-learning and make the training procedure more stable. Specifically, the weights $\boldsymbol{\psi}_{-}$ of the target DQN are updated as $\boldsymbol{\psi}_{-} \leftarrow \boldsymbol{\psi}$ every G learning steps. Thus, the weights $\boldsymbol{\psi}$ of the DQN $\hat{Q}_{\mathrm{U}}$ are updated by using stochastic gradient descent (SGD).

Based on the DQN $\hat{Q}_{\mathrm{U}}$, the upper-level control policy π_{R} is expressed as,

$$\begin{aligned} & \pi_{\mathrm{C}}(\boldsymbol{C}_k | \boldsymbol{S}_k) \\ &\quad = \begin{cases} 1-\epsilon, \text{ if } \boldsymbol{C}_k = \arg\max_{\boldsymbol{C}}\left\{\hat{Q}_{\mathrm{U}}(\boldsymbol{S}_k, \boldsymbol{C} | \boldsymbol{\psi})\right\}, \\ \epsilon / (|\mathcal{C}| - 1), \text{ otherwises}, \end{cases} \end{aligned}$$

In the learning procedure of DQN $\hat{Q}_{\mathrm{U}}$, and ϵ-greedy rule is used in formula (10), where the upper-level control performs exploration either by randomly selecting the slice configuration $\boldsymbol{C}_k \in \mathcal{C}$ or by selecting the slice configuration $\boldsymbol{C}_k$ with the maximum value of DQN $\hat{Q}_{\mathrm{U}}$. Furthermore, term ϵ decays exponentially according to the factor $\epsilon_{\mathrm{decay}}$ by each learning step until the minimal value $\epsilon_{\min}$ is reached.

In summary, the learning procedure of the upper-level control policy π_{C} can be stated as in **Algorithm 2**. Similar to **Algorithm 1**, we also maintain an "experience replay buffer," which contains samples generated during the upper-level controlling process.

Algorithm 2 Double-DQN based learning procedure of the upper-level control policy π_{C}

Initialization:

- Initialize the DQN $\hat{Q}_{\mathrm{U}}$ with the weights $\boldsymbol{\psi}$.
- Set weights of target neural networks, i.e., $\boldsymbol{\psi}_{-} \leftarrow \boldsymbol{\psi}$ and $\epsilon = 1$.

Repat:

1) Observe upper-level state $\boldsymbol{S}_k$, generate slice configuration $\boldsymbol{C}_k$ based on ϵ-greedy rule in formula (10), and observe the new upper-level state $\boldsymbol{S}_{k+1}$.

2) Update ϵ-greedy rule: $\epsilon \leftarrow \max\{\epsilon \cdot \epsilon_{\mathrm{decay}}, \epsilon_{\min}\}$.

3) Store sample data $\left(\boldsymbol{S}_k, \sum_{n=1}^{N} U_{n,k}, \boldsymbol{S}_{k+1}\right)$ in the "experience replay buffer".

4) Update the weights $\boldsymbol{\psi}$ of the DQN $\hat{Q}_{\mathrm{U}}$ by performing stochastic gradient descent (SGD) to minimize the loss function $L(\boldsymbol{\psi})$ in formula (18).

5) Every G learning steps, update the weights $\boldsymbol{\psi}_{-}$ of the target DQN: $\boldsymbol{\psi}_{-} \leftarrow \boldsymbol{\psi}$.

until convergence

D. Computational Complexity and Implementation of Proposed DRL Framework

According to analysis method in [25], the computational complexity of learning procedure for the lower-level control policy (i.e., **Algorithm 1**) can be expressed by

$$\mathcal{O}\left(T_{\mathrm{L}}\left(\sum\nolimits_{l=0}^{L_{\mathrm{actor}}} n_{\mathrm{actor}}^{(l)} n_{\mathrm{actor}}^{(l+1)} + \sum\nolimits_{l=0}^{L_{\mathrm{critic}}} n_{\mathrm{critic}}^{(l)} n_{\mathrm{critic}}^{(l+1)}\right)\right),$$

where T_{L} is the learning steps of lower-level policy training, $n_{\mathrm{actor}}^{(l)}$ is the number of neurons in the l-th layer of the actor neural network, i.e., the lower-level control policy π_{R}, $n_{\mathrm{critic}}^{(l)}$ is the number of neurons in the l-th layer of the critic neural network, i.e., the DQN $\hat{Q}_{\mathrm{L}}$, and L_{actor} (L_{critic}) denotes the number of the hidden layers in the actor (critic) network.

In detail, the lower-level control policy $\pi_{\mathrm{R}}(\boldsymbol{X}_t; \boldsymbol{C}_k | \boldsymbol{\varphi})$ is a multi-inputs neural network, which contains three hidden layers (i.e., $L_{\mathrm{actor}} = 3$). Each of the hidden layers has 128 neurons and applies the rectified linear units (ReLU) activation function. The output layer gives the scheduling priority factor of UEs $\boldsymbol{w}_t$. Furthermore, considering not all UEs are activated, the scheduling priority factor of UEs with zero queue length (i.e., $q_{i,t} = 0$) are set as 0. Meanwhile, the

TABLE I
DEFAULT PARAMETER SETTINGS FOR NETWORK SLICING

Parameter	**Assumption**			
Number of RBs F / Bandwidth of PRB	50/ 180 kHz			
Small scale fading	Rayleigh fading with zero mean and unit variance			
Total Transmit Power of BS	20 dBm			
The length of each epoch	100 ms (100 TTI)			
Traffic Generation Model	3GPP FTP model 1			
Service type	eMBB (slice #1)		uRLLC (slice #2)	
Average packet arrival rate of UE	0.03 packets/TTI		0.10 packets/TTI	
Weighting factors in utlity function (8)	$\boldsymbol{\alpha}_1 = [1, 0.5, 2 \times 10^{-4}]$		$\boldsymbol{\alpha}_2 = [2, 1, 4 \times 10^{-4}]$	
Packet size per UE	800 Byte		200 Byte	
Maximum tolerant packet delay	10 *ms*		5 *ms*	
Candidate network slice configurations	$R_1^{\min}$	$R_1^{\max}$	$R_2^{\min}$	$R_2^{\max}$
	2600 bits/TTI	3300 bits/TTI	1600 bits/TTI	2200 bits/TTI
	800 bits/TTI	2400 bits/TTI	800 bits/TTI	1600 bits/TTI
The number of candidate slice configurations	$2 \times 2 = 4$			

DQN $\hat{Q}_{\mathrm{L}}(\boldsymbol{X}_t, \boldsymbol{w}_t; \boldsymbol{C}_k | \boldsymbol{\theta})$ is a multi-inputs neural network with four hidden layers. Each hidden layer has 128 neurons and applies the ReLU activation function. With a sigmoid activation function, the output layer gives the estimated Q function value of the lower-level control. In **Algorithm 1**, the DQN $\hat{Q}_{\mathrm{L}}$ and π_{R} are updated by the Adam algorithm with a learning rate of 10^{-3} and the weights of target neural networks are updated by rate $\iota = 10^{-3}$.

On the other hand, the complexity of learning procedure for the upper-level control policy (i.e., **Algorithm 2**) is given by

$$\mathcal{O}\left(T_{\mathrm{U}}\left(\sum\nolimits_{l=0}^{L_{\mathrm{DQN}}} n_{\mathrm{DQN}}^{(l)} n_{\mathrm{DQN}}^{(l+1)}\right)\right),$$

where T_{U} is the learning steps of upper-level policy training, $n_{\mathrm{DQN}}^{(l)}$ is the number of neurons in the l-th layer of DQN $\hat{Q}_{\mathrm{U}}$, and L_{DQN} is the number of the hidden layers in the DQN $\hat{Q}_{\mathrm{U}}$.

In the upper-level control, DQN $\hat{Q}_{\mathrm{U}}(\boldsymbol{S}_k, \boldsymbol{C}_k | \boldsymbol{\psi})$ is a four layers neural network (i.e., $L_{\mathrm{DQN}} = 4$). Each hidden layer contains 48 units with the ReLU activation function. The output layer has the same size as the number of all candidate slicing configurations (i.e., $|\mathcal{C}|$), and each element of the output layer is mapping to an estimated value of $Q_{\mathrm{U}}(\boldsymbol{S}_k, \boldsymbol{C})$ $(\boldsymbol{C} \in \mathcal{C})$ under the upper-level state $\boldsymbol{S}_k$. The parameters of ϵ-greedy rule are set as $\epsilon_{\min} = 0.01$ and $\epsilon_{\mathrm{decay}} = 0.99$. In **Algorithm 2**, the weights $\boldsymbol{\psi}$ is updated by the Adam algorithm with a learning rate of 10^{-3}, and weights $\boldsymbol{\psi}_-$ are copied from $\boldsymbol{\psi}$ every 200 learning steps.

The computational complexity of **Algorithm 1** and **2** is linear time complexity with the number of learning steps and the number of layers in the neural network, quadratic time complexity with the number of neurons in the neural network's layer. Furthermore, the proposed DRL framework is only executing at the base station, which has sufficient computational resources available.

V. SIMULATION RESULTS AND ANALYSIS

In this section, a system-level simulation platform is implemented. Software including MATLAB 2019a and Keras 2.2.2 with Python 3.5.2. are utilized for simulations.

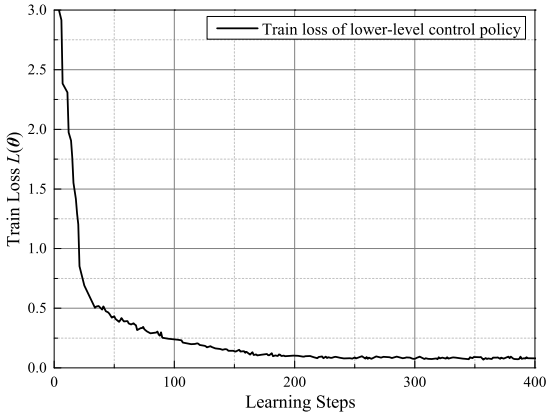
A. Simulation Setup

In this simulation, there are two types of services and two corresponding slices, that is, slice for ultra-reliable and low-latency communications (uRLLC) and slice for enhanced mobile broadband (eMBB) [33]: a). uRLLC for supporting mission-critical application, assuming that the BS needs to send control information to the related equipment in the cell; and b). eMBB slice for supporting high-definition video traffic, browsing, etc. Since the critical nature of the uRLLC service, it has a higher priority as compared to the eMBB service, that is, the uRLLC slice has a larger weighting factor in its utility function defined in formula (8). The discount rate λ in the target problem $\mathcal{P}$ is 0.9. Furthermore, all UEs are assumed to have i.i.d. Rayleigh fading wireless channel, and the mean SNR per PRB is set to 4 dB when the transmit power per PRB is 1 dBm. Detailed simulation parameters are in Table I.

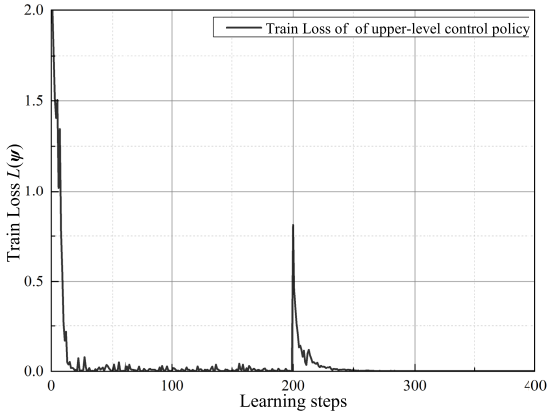
B. Experiment Results

1) Training Performance: The first experiment aims to examine the convergence of the proposed hierarchical DRL framework. The metric can measure the convergence speed of the loss function during the training procedure in **Algorithm 1** and **Algorithm 2**. Under the default simulation setup, in Figure 2, we plot the variations in the training loss of upper-level and lower-level control policy during the training procedure, in which both control policy reaches a state of convergence. It confirms that the proposed DRL framework can avoid the mis-convergence and unstable issues in the learning of the control strategy π as mentioned in ***Challenge 1***. Additionally, in Figure 2(b), a peak is suddenly appeared at around 200 learning steps during the training of the upper-level control policy. It can be explained as follows: As stated in Section IV-D, the weights of target DQN $\hat{Q}_{\mathrm{U}}(\cdot | \boldsymbol{\psi}^-)$ are copied from the weights of DQN $\hat{Q}_{\mathrm{U}}(\cdot | \boldsymbol{\psi})$ every 200 steps. When DQN $\hat{Q}_{\mathrm{U}}(\cdot | \boldsymbol{\psi})$ is not completely converged, this operation will induce increased volatility in the target Q-value of upper-level in (18), making the train loss increase suddenly.

On the other hand, Figure 3 shows the utility function (i.e. $\sum_{n=1}^{N} U_{n,k}$) of the target problem $\mathcal{P}$ during the training process. The values of utility function vary dramatically between 2.4 and 3.45 during the first 600 epochs. The main reason is that the parameter of ϵ-greedy in the upper-level controlling does not decay to the minimal value of $\epsilon_{\min}$, and the upper-level controller takes more exploration in these epochs, which may deteriorate the controlling performance.



(a) Training of the lower-level control policy



(b) Training of the upper-level control policy

Fig. 2. Illustration of the convergence of the proposed hierarchical DRL framework in the training procedure: We follow the proposed two-stage training procedure in Section IV-A: 1) In stage I, we train the lower-level control policy π_{R}, in which the train loss is measured by loss function $L(\boldsymbol{\theta})$ defined in formula (15); 2) In stage II, upon the converged lower control policy, we train the DQN $\hat{Q}_{\mathrm{U}}$ of upper-level control is trained, where its train loss is measured by $L(\boldsymbol{\varphi})$ in formula (18).

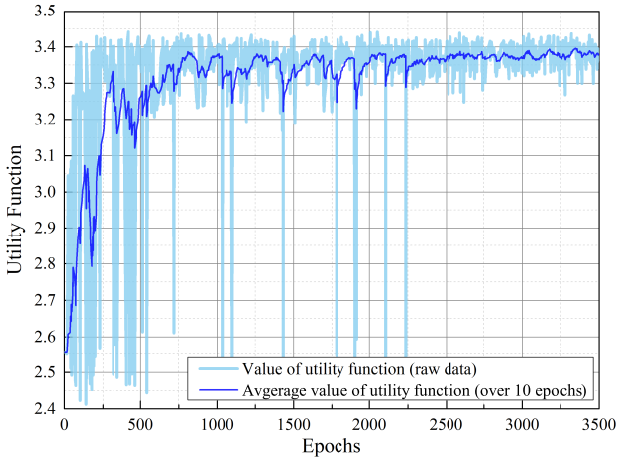


Fig. 3. The utility function of all network slices, i.e. $\sum_{n=1}^{N} U_{n,k}$ defined in (8), with the proposed scheme during the training procedure. The utility function can reflect the performance of RAN slicing control, that is, the higher value of the utility function, the better the QoS performance of network slices.

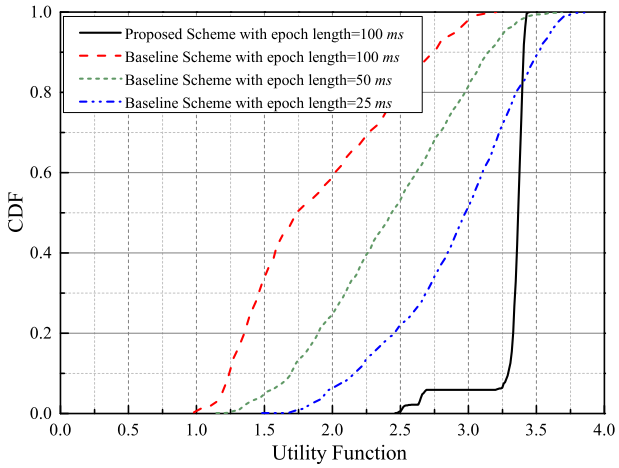


Fig. 4. CDF of the utility function of all network slices, i.e. $\sum_{n\in\mathcal{N}} U_{n,k}$ defined in (8), with different schemes.

After the first 600 epochs, the values of the utility function become more stable. The average utility function value curve is convergent to the value of about 3.36, which implies the control strategy has reached a stable performance.

2) Performance Evaluation: To match our framework for the performance comparisons, we modify the work in [25] and design the baseline scheme as follows,

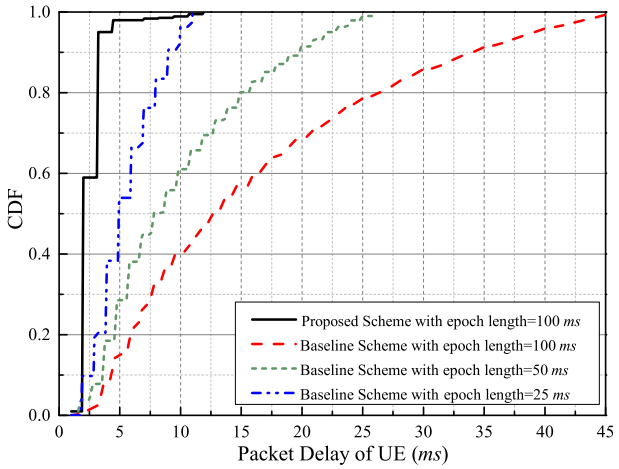
Baseline Scheme (iRSS): According to the RAN slicing control strategy proposed by article [25], the BS assigns dedicated PRBs to each slice in a large time-scale (referred to as epoch). Meanwhile, in each slice, the assigned PRBs are dynamically scheduled to UEs in the small time-scale (i.e., each TTI). Specifically, following the paradigm of our previous work [7], we use Deep Recurrent Q Network (DRQN) to assign a number of PRBs for each network slice in the next epoch. During each TTI, the radio resource management aims at maximizing the weighted sum data rate of UEs in slice [19].

From a statistical point of view, we evaluate the utility function defined in (8), which is used to indicate the performance of slicing control. Figure 4 presents the cumulative distribution functions (CDFs) of the utility function with the proposed scheme and the baseline scheme under different settings of epoch length. It shows that the CDF curve of our proposed scheme significantly outperforms the baseline scheme under the default epoch length (i.e., 100 ms). The main reason is that our proposed scheme enables dynamic sharing of the radio resources among slices in every TTI, which realizes a flexible radio resource management. In contrast, in the baseline scheme, the radio resources are dedicated to each slice in a large time scale (i.e., each epoch), in which the number of PRBs in each slice is fixed in every TTI. This drawback makes the baseline scheme cannot efficiently deal with the network dynamics at the small timescale. For instance, when the channel condition of UEs in one slice deteriorate sharply, the demand of PRBs may exceed the fixed number of PRBs in this slice. In this case, the UEs of slice will suffer from the deteriorated QoS performance due to the insufficient transmission opportunities.

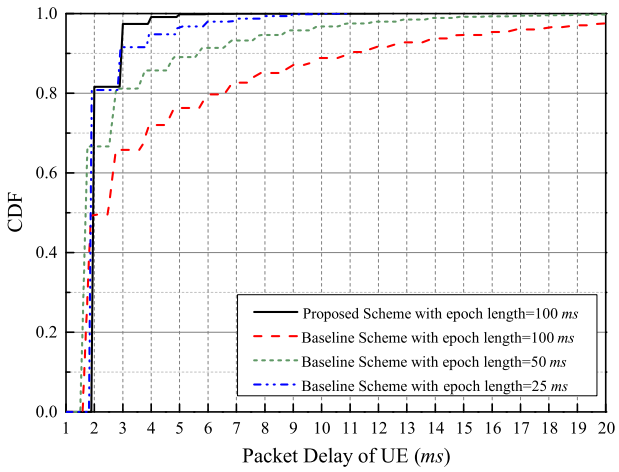
TABLE II

PERFORMANCE OF THE PROPOSED INTELLIGENT RAN SLICING STRATEGY

	Average Utility Function	Average Packet Delay		Average Packet Drop Rate	
		eMBB slice	uRLLC slice	eMBB slice	uRLLC slice
Baseline Scheme (epoch length=100 *ms*)	1.884	17.42 *ms*	4.92 *ms*	0.4060	0.1086
Baseline Scheme (epoch length=50 *ms*)	2.456	11.09 *ms*	3.16 *ms*	0.2578	0.0363
Baseline Scheme (epoch length=25 *ms*)	2.955	6.64 *ms*	3.48 *ms*	0.0669	0.0143
Proposed Scheme (epoch length=100 *ms*)	3.330	4.49 *ms*	3.23 *ms*	0.0357	0.0039

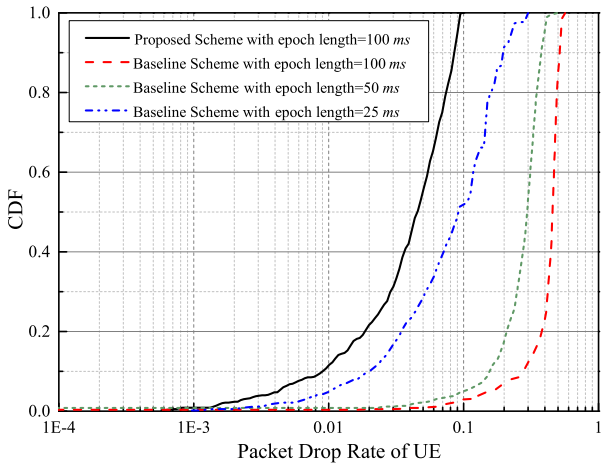


(a) Packet latency of UE in eMBB slice

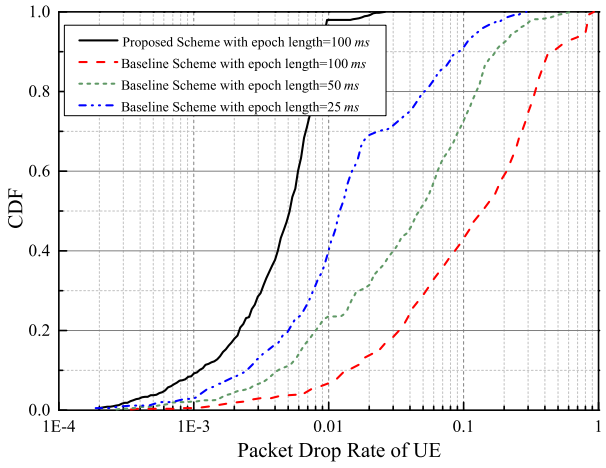


(b) Packet latency of UE in uRLLC slice

Fig. 5. CDF of packet delay of UEs in each slice under different schemes.



(a) Packet drop rate (PDR) of UE in eMBB slice



(b) Packet drop rate (PDR) of UE in uRLLC slice

Fig. 6. CDF of packet drop rate (PDR) of UE in each slice under different schemes.

Furthermore, in Figure 4, we notice that the CDF curve of the baseline scheme with 50 ms epoch length is better than the baseline scheme with default epoch length (100 ms). Meanwhile, the baseline scheme with 25 ms epoch length is close to the control performance of the proposed scheme. More Specifically, the average utility function of the proposed scheme is 3.329 and its standard deviation (STD), denoting how much the utility function fluctuates around its average value, is 0.184. On the other hand, the average utility function of the baseline scheme with 25 ms is 3.055, where its STD is 0.545. These result confirms that, to achieve reach the same level of performance as the proposed scheme, the baseline scheme needs to reduce the epoch length to keep up with the fluctuations of wireless networks at the small timescale. This disadvantage makes the baseline scheme inevitably consumes much more signalling overheads as compared to the proposed scheme. Furthermore, the detailed QoS performance is summarized in Table II. Figure 5 depicts the CDF curve of packet delay of UE under different slicing schemes. Compared to the baseline scheme, the proposed scheme can better reduce the packet delay of UE due to the flexible radio resource management paradigm. In the proposed scheme, the average packet delay of UE in eMBB slice is 4.49 ms and the average delay in uRLLC slice is 2.23 ms. On the other hand, in the

baseline scheme with default epoch length, the average delay in eMBB slice is 17.42 ms and the average delay in uRLLC slice is 5.92 ms. Furthermore, only when the epoch length is equal to 25 ms, can the baseline scheme achieve the same performance level of the proposed scheme. On the other hand, Figure 6 plots the CDF curve of the packet drop rate (PDR) of UE under different schemes. It is noteworthy to mention that baseline scheme with default epoch length has the worst PDR performance. It means that the baseline scheme cannot guarantee even worsen the PDR performance of some UEs in the slice.

From these results, we can conclude that the proposed scheme can better guarantee the QoS performance of each slice under a satisfying range. In the baseline scheme, it assigns fixed number of PRBs to each slice with epoch, which hinders the slice to share the idle PRBs with the other overloaded slice. Different from the baseline scheme, the proposed scheme only restricts the maximum/minimum data rate of UE in the slice in the large time scale. Meanwhile, in every TTI, the proposed scheme can precisely and intelligently manage radio resources and better deal with all kinds of network situations.

VI. Conclusion

In this paper, we propose an intelligent RAN slicing strategy with multiple control granularity, which aims at maximizing the long-term QoS of services and SE of network slices. Specifically, the proposed strategy consists of an upper-level controller and a lower-level controller. The upper-level controller adapts the slice configuration to improve QoS performance at a coarse granularity. The lower-level controller schedules PRB and power allocation to active UEs in each network slice at a fine granularity. Then, based on multi time-scale MDP, we propose a novel hierarchical DRL framework, which is an integration of modified DDPG and double-DQN algorithm, for learning the optimal RAN slicing control strategy. Furthermore, we also propose an action space reducing approach to address the extremely high dimensionality of action space in lower-level control. Simulation results show that the proposed scheme has stable convergence control performance and achieves higher QoS performance as compared to the baseline scheme.

However, two open issues still need to be carefully addressed before practical implementation of intelligent RAN slicing. Firstly, for multi-cell scenarios, since frequency reuse takes place among different cells, each cell will suffer from the dynamic interference from neighboring cells, which makes PRBs in-homogeneous in communication capacity. This characteristic calls for inter-cell coordination techniques to ensure the performance of multi-cell RAN slicing. Secondly, this paper does not quantify the signaling overhead generated by RAN slicing control. When a UE concurrently associates multiple slices, this UE will incur excessive signaling overhead. Since the limited radio resources for transmitting the signaling overhead, we need to consider the cost of signaling overhead for attaching multiple slices, e.g., its impact on the UE that only associating with one slice, especially in high traffic load. How to balance the trade-off between the signaling overhead for the UEs attaching multiple slices and QoS fairness among all UEs is an essential issue in the design of RAN slicing schemes. These issues need to be further studied.

Appendix

A. Proof of **Proposition 1**

Proof: The main barrier faced by solving problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ is non-convexity of the data rate formulation (2) for the short packet transmission. However, when the SNR of UE in the high region (greater than 10 dB), the channel dispersion V of formulation (2) can be approximated by $V = 1$ [34]. By setting γ_{TH} larger than 10 dB, C3 can ensure the SNR of UE larger than 10 dB. Under this condition, formulation (2) can be approximated as a convex function with respect to $p_{i,j}$. Then, we can treat problem $\mathcal{P}_1^{\mathrm{R}}(\boldsymbol{w}_t)$ as a convex problem, which can be solved by the Dual Lagrangian decomposition method and KKT conditions. ■

References

[1] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6G networks: Use cases and technologies," *IEEE Commun. Mag.*, vol. 58, no. 3, pp. 55–61, Mar. 2020.

[2] B. Zong, C. Fan, X. Wang, X. Duan, B. Wang, and J. Wang, "6G technologies: Key drivers, core requirements, system architectures, and enabling technologies," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 18–27, Sep. 2019.

[3] W. Guan, H. Zhang, and V. C. M. Leung, "Customized slicing for 6G: Enforcing artificial intelligence on resource management," 2021, *arXiv:2102.10498*. [Online]. Available: http://arxiv.org/abs/2102.10498

[4] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[5] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J.-A. Zhang, "The roadmap to 6G: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, Aug. 2019.

[6] X. Shen *et al.*, "AI-assisted network-slicing based next-generation wireless networks," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 45–66, 2020.

[7] J. Mei, X. Wang, and K. Zheng, "Intelligent network slicing for V2X services toward 5G," *IEEE Netw.*, vol. 33, no. 6, pp. 196–204, Nov. 2019.

[8] Y. Liu, X. Wang, G. Boudreau, A. B. Sediq, and H. Abou-Zeid, "Deep learning based hotspot prediction and beam management for adaptive virtual small cell in 5G networks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 1, pp. 83–94, Feb. 2020.

[9] *Study on Management and Orchestration of Network Slicing for Next Generation Network (Release 15)*, Standard 3GPP TS 28.801, Jan. 2018.

[10] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2429–2453, 3rd Quart., 2018.

[11] H. Halabian, "Distributed resource allocation optimization in 5G virtualized networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 627–642, Mar. 2019.

[12] R. Wen *et al.*, "On robustness of network slicing for next-generation mobile networks," *IEEE Trans. Commun.*, vol. 67, no. 1, pp. 430–444, Jan. 2019.

[13] R. Ferrus, O. Sallent, J. Perez-Romero, and R. Agusti, "On 5G radio access network slicing: Radio interface protocol features and configuration," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 184–192, May 2018.

[14] T. Guo and A. Suárez, "Enabling 5G RAN slicing with EDF slice scheduling," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2865–2877, Mar. 2019.

[15] J. Li *et al.*, "A hierarchical soft RAN slicing framework for differentiated service provisioning," *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 90–97, Dec. 2020.

[16] M. Zambianco and G. Verticale, "Interference minimization in 5G physical-layer network slicing," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4554–4564, Jul. 2020.
[17] V. Sciancalepore, M. Di Renzo, and X. Costa-Perez, "STORNS: Stochastic radio access network slicing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–7.
[18] P. Caballero, A. Banchs, G. de Veciana, X. Costa-Perez, and A. Azcorra, "Network slicing for guaranteed rate services: Admission control and resource allocation games," *IEEE Trans. Wireless Commun.*, vol. 17, no. 10, pp. 6419–6432, Oct. 2018.
[19] Y. L. Lee, J. Loo, T. C. Chuah, and L.-C. Wang, "Dynamic network slicing for multitenant heterogeneous cloud radio access networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2146–2161, Apr. 2018.
[20] G. Ghatak, A. De Domenico, and M. Coupechoux, "Small cell deployment along roads: Coverage analysis and slice-aware RAT selection," *IEEE Trans. Commun.*, vol. 67, no. 8, pp. 5875–5891, Aug. 2019.
[21] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3039–3071, Jul. 2019.
[22] Y. Liu, X. Wang, J. Mei, G. Boudreau, H. Abou-Zeid, and A. B. Sediq, "Situation-aware resource allocation for multi-dimensional intelligent multiple access: A proactive deep learning framework," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 116–130, Jan. 2021.
[23] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "GAN-powered deep distributional reinforcement learning for resource management in network slicing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 334–349, Feb. 2020.
[24] R. Li *et al.*, "Deep reinforcement learning for resource management in network slicing," *IEEE Access*, vol. 6, pp. 74429–74441, 2018.
[25] M. Yan, G. Feng, J. Zhou, Y. Sun, and Y.-C. Liang, "Intelligent resource scheduling for 5G radio access network slicing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7691–7703, Aug. 2019.
[26] X. Chen *et al.*, "Multi-tenant cross-slice resource orchestration: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2377–2392, Oct. 2019.
[27] *Study on New Radio Access Technology Physical Layer Aspects*, Standard 3GPP RT 38.802, Jun. 2017.
[28] H. Yang, K. Zheng, K. Zhang, J. Mei, and Y. Qian, "Ultra-reliable and low-latency communications for connected vehicles: Challenges and solutions," *IEEE Netw.*, vol. 34, no. 3, pp. 92–100, May 2020.
[29] H. S. Chang, P. J. Fard, S. I. Marcus, and M. Shayman, "Multitime scale Markov decision processes," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 976–987, Jun. 2003.
[30] A. T. Z. Kasgari, W. Saad, M. Mozaffari, and H. V. Poor, "Experienced deep reinforcement learning with generative adversarial networks (GANs) for model-free ultra reliable low latency communication," Nov. 2019, *arXiv:1911.03264*. [Online]. Available: http://arxiv.org/abs/1911.03264
[31] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[32] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: http://arxiv.org/abs/1509.02971
[33] I.-P. Belikaidis *et al.*, "5G radio access network slicing: System-level evaluation and management," *IEEE Veh. Technol. Mag.*, vol. 14, no. 4, pp. 49–55, Dec. 2019.
[34] H. Yang, K. Zhang, K. Zheng, and Y. Qian, "Joint frame design and resource allocation for ultra-reliable and low-latency vehicular networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3607–3622, May 2020.

Jie Mei (Member, IEEE) received the B.S. degree from the Nanjing University of Posts and Telecommunications (NJUPT), China, in 2013, and the Ph.D. degree in information and communication engineering from the Beijing University of Posts and Telecommunications (BUPT) in June 2019. He is currently a Post-Doctoral Associate with the Department of Electrical and Computer Engineering, Western University, Canada. His research interests include intelligent communications, multi-dimensional intelligent multiple access, and vehicle-to-everything (V2X) communication. He was a TPC member of IEEE GLOBECOM 2020.

Xianbin Wang (Fellow, IEEE) is a Professor and the Tier-1 Canada Research Chair of Western University, Canada. He received the Ph.D. degree in electrical and computer engineering from the National University of Singapore in 2001.

Prior to joining Western, he was with the Communications Research Centre Canada (CRC), as a Research Scientist/Senior Research Scientist from July 2002 to December 2007. From January 2001 to July 2002, he was a System Designer with STMicroelectronics. His current research interests include 5G/6G technologies, the Internet-of-Things, communications security, machine learning, and intelligent communications. He has over 450 highly cited journal and conference articles, in addition to 30 granted and pending patents and several standard contributions.

Dr. Wang is a fellow of the Canadian Academy of Engineering, a fellow of the Engineering Institute of Canada, and an IEEE Distinguished Lecturer. He has received many awards and recognitions, including the Canada Research Chair, the CRC President's Excellence Award, the Canadian Federal Government Public Service Award, the Ontario Early Researcher Award, and six IEEE best paper awards. He currently serves/has served as the Editor-in-Chief, the Associate Editor-in-Chief, an Editor, or an Associate Editor for over ten journals. He was involved in many IEEE conferences, including GLOBECOM, ICC, VTC, PIMRC, WCNC, CCECE, and CWIT, in different roles such as the General Chair, the Symposium Chair, a Tutorial Instructor, the Track Chair, the Session Chair, the TPC Co-Chair, and a Keynote Speaker. He has been nominated as an IEEE Distinguished Lecturer several times during the last ten years. He is currently serving as the Chair of IEEE London Section and the Chair of the ComSoc Signal Processing and Computing for Communications (SPCC) Technical Committee.

Kan Zheng (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees from the Beijing University of Posts and Telecommunications (BUPT), China, in 1996, 2000, and 2005, respectively. He is currently a Professor with BUPT. He has rich experiences on the research and standardization of the new emerging technologies. He is the author of more than 200 journal articles and conference papers in the field of wireless networks, IoT, and vehicular communication. He is on Editorial Board for several journals and has organized several Special Issues, including IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, *IEEE Communication Magazine*, and IEEE SYSTEMS JOURNAL.

Gary Boudreau (Senior Member, IEEE) received the B.A.Sc. degree in electrical engineering from the University of Ottawa in 1983, the M.A.Sc. degree in electrical engineering from Queens University in 1984, and the Ph.D. degree in electrical engineering from Carleton University in 1989.

From 1984 to 1989, he was employed as a Communications Systems Engineer with Canadian Astronautics Ltd. From 1990 to 1993, he worked as a Satellite Systems Engineer at MPR Teltech Ltd. From 1993 to 2009, he was employed by Nortel Networks in a variety of wireless systems and management roles within the CDMA and LTE base station product groups. In 2010, he joined Ericsson Canada, where he is currently employed with the 5G Systems Architecture Group. His interests include digital and wireless communications and digital signal processing.

Akram Bin Sediq (Member, IEEE) received the B.Sc. degree in electrical engineering degree *(summa cum laude)* from the American University of Sharjah (AUS), United Arab Emirates, in 2006, and the M.A.Sc. degree in electrical engineering and the Ph.D. degree in electrical and computer engineering from Carleton University, Ottawa, Canada, in 2008 and 2013, respectively.

In Winter 2012, he was a Course Instructor at Carleton University. From 2013 to 2014, he was a System Engineer at BLiNQ Networks, Ottawa, where he was mainly responsible for contributing to the company's patent portfolio and developing novel algorithms for non-line-of-sight small-cell backhaul networks. He joined the Research and Development Group, Ericsson Canada, Ottawa, in November 2014, where he is currently a Senior Specialist in radio access network (RAN) edge analytics and his responsibilities include securing embedded machine intelligence in RAN to facilitate automated and adaptive RAN solutions, driving early-phase system designs of 4G/5G wireless systems, contributing to the company's patent portfolio, and managing university collaborations. His work resulted in more than 25 granted and filed patents and more than 30 peer-reviewed publications. His research interests include radio resource management (RRM), machine-learning-based RRM, data analytics, optimization, dual connectivity, control channel design, energy-saving, and constellation design.

Hatem Abou-Zeid (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Queen's University in 2014. He is currently a Senior 5G Systems Designer at Ericsson, Canada, where he drives system designs, algorithm development, and patents for next generation radio access networks (RAN). Prior to joining Ericsson, he was at Cisco Systems designing scalable traffic engineering and routing solutions for service provider and data-center infrastructure. He has coauthored over 50 patents and peer-reviewed publications. His research focus is on RAN intelligence and network innovations to support new 5G services, such as uRLLC and extended reality. His research draws on methods from stochastic optimization, Bayesian estimation, reinforcement learning, time series analysis, and deep learning. He highly values university-industry collaborations and leads multiple academic partnerships on various topics, including autonomous networking, edge intelligence, and critical communications infrastructure. He was awarded a DAAD RISE Fellowship at Bell Labs and nominated for an Outstanding Thesis Medal.