Toward Scalable and Efficient Hierarchical Deep Reinforcement Learning for 5G RAN Slicing

Renlang Huang, *Student Member, IEEE*, Miao Guo, *Student Member, IEEE*, Chaojie Gu, *Member, IEEE*, Shibo He, *Senior Member, IEEE*, Jiming Chen, *Fellow, IEEE*, and Mingyang Sun, *Senior Member, IEEE*

***Abstract*—As an emerging and promising network paradigm, network slicing creates multiple logical networks on shared infrastructure to provide services with customized Quality-of-Service (QoS) for heterogeneous devices and applications. However, network complexity and service heterogeneity pose a huge challenge in achieving optimal performance and ensuring stringent QoS requirements. In this paper, we design a hierarchical deep reinforcement learning based 5G radio access network slicing framework to achieve scalable and efficient resource allocation. By decomposing the resource allocation problem into a slice-level task and several user-level tasks, the proposed framework tackles each task with an agent, thus conquering insufficient exploration and achieving scalable management. Knowledge transfer and progressive learning are employed to improve training efficiency and stability, respectively. We apply collaborative training to eliminate distribution mismatch by refining value approximators and policies of agents alternately. Extensive experiments show that the proposed framework can learn effective resource allocation policies stably and efficiently and outperform other methods in network utility maximization and QoS assurance, which improves the network utility by 25% and 8% compared with the random strategy and the ADMM strategy, respectively. Furthermore, we validate that our framework is more robust to changes in network traffic conditions including network congestion.**

***Index Terms*—Deep reinforcement learning (DRL), hierarchical reinforcement learning (HRL), network slicing (NS), radio access network (RAN), industrial Internet of Things (IIoT).**

I. Introduction

As an emerging and promising networking paradigm, Industrial Internet-of-Things (IIoT) has revolutionized industrial operation and empowered intelligent manufacturing via ubiquitous interconnection and interaction among massive machines, platforms, and people. Based on broad perception and service, IIoT enables industrial data modeling, automated analysis, and intelligent decision-making through deep cooperation with Artificial Intelligence (AI). With the rapid development of the fifth-generation (5G) mobile network, IIoT can support various emerging scenarios in vertical industries including intelligent manufacturing [1], smart cities [2], intelligent transportation [3], and collaborative sensing [4].

With the sustained growth of wireless devices and emerging applications, the 5G network needs to satisfy different service levels in diversified application scenarios, mainly including enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable and low-latency communications (URLLC). As a prospective solution, network slicing (NS) has been proposed to support increasing heterogeneous user devices and personalized service requirements. Network slicing system creates multiple isolated logical end-to-end networks on shared physical infrastructure, realizing independent management, diversified service support, and enhanced data privacy. Technically, network slicing is supported by Software Defined Networking (SDN) and Network Function Virtualization (NFV) [5], abstracting physical resources as virtual resources and packaging network functions as virtual machines or containers, i.e., virtual network functions (VNFs). Network slices chain multiple VNFs and configure virtual resources to provide customized services with personalized QoS requirements. 5G network slicing architecture mainly includes core network slicing, radio access network (RAN) slicing, and end-to-end slicing [6]. In this paper, we focus on RAN slicing, which can merge multi-dimensional cloud and network resources/functions provided by multi-access edge computing (MEC) at the network edge for customized service provision and personalized service-level assurance, alleviating the burden of backhaul and core networks [7]. Specifically, we aim to ensure QoS requirements and optimize network resource utilization of the 5G RAN slicing system through dynamic resource allocation.

Researchers have proposed optimization methodologies to conquer the resource allocation problem in network slicing, but conventional optimization algorithms, including convex optimization and nonlinear programming, fail to accommodate to high complexity and flexibility of 5G networks [8]. Additionally, these model-based optimization techniques are not applicable for time-varying communication networks either, which cannot be modeled precisely. In contrast, deep learning is regarded as a promising approach to satisfy the demands of network slicing management. Deep neural networks (DNNs) have drawn widespread attention in modeling network performance, traffic, and complex behavior

Manuscript received 13 July 2022; revised 9 March 2023 and 3 July 2023; accepted 9 July 2023. Date of publication 13 July 2023; date of current version 22 November 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1708700; in part by the National Science Foundation of China (NSFC) under Grant U1909207; in part by NSFC through the Autonomous Intelligent Unmanned Systems under Grant 62088101; and in part by the Fundamental Research Funds for the Central Universities under Grant 226-2022-00107 and Grant 226-2023-00111. The editor coordinating the review of this article was X. Ge. *(Corresponding author: Chaojie Gu.)*

The authors are with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, Zhejiang, China (e-mail: renlanghuang@zju.edu.cn; gm_oct@zju.edu.cn; gucj@zju.edu.cn; s18he@zju.edu.cn; cjm@zju.edu.cn; mingyangsun@zju.edu.cn).

patterns [9], while reinforcement learning (RL) turns out to be an efficient approach to optimal network management since it can learn smart and complicated multi-step decision policies during interaction with the network environment [10]. Deep reinforcement learning (DRL) integrates DNN into RL to approximate the action-value function and parameterize the sophisticated policy, thus implicitly modeling the dynamics of the environment and accommodating the large state space size and action space size. Deep reinforcement learning has showcased its strength in networking, which can automatically mine information from the actual performance of past decisions to optimize its control policy for the characteristics of the network, without using any pre-programmed control rules or mathematical models about the operating environment and the specific task at hand [11]. Recent works have exploited model-free DRL-based resource optimization in network slicing [12]. As the state dimension and action dimension increase due to the network complexity and resource diversity, it is difficult for single-agent DRL to explore the environment sufficiently and stably converge to a well-trained policy.

To this end, we decompose the resource allocation problem in network slicing as two-level sub-problems and further propose an efficient and scalable hierarchical deep reinforcement learning (HDRL) framework as a solution. Since each sub-problem corresponds to lower-dimensional state space and action space, hierarchically distributed agents of the proposed HDRL framework can explore the environment independently and sufficiently. Since the action space is continuous in our problem formulation, we adopt twin delayed deep deterministic policy gradient (TD3) [13] to handle each sub-problem. As an actor-critic architecture, TD3 evaluates the action value (i.e., Q value) by modeling network performance with the critic networks and guides the actor network to optimize the policy. Additionally, existing DRL-based solutions pay little attention to scalability and learning efficiency. As we relatively independently optimize the resource allocation policies for each network slice, our HDRL framework leads to more scalable and flexible network slicing management. In this paradigm, user-level allocation policies for single slice management can share knowledge with each other through knowledge transfer and parameter fine-tuning to greatly improve training efficiency, thus enabling rapid learning and deployment of new network slices. A series of numerical experiments validate the effectiveness and efficiency of our framework, which has improvements of 25% and 8% relative to the random feasible searching strategy and the *state-of-the-art* ADMM strategy [14] in network utility, respectively.

To summarize, the main contributions of this paper are:

- We formulate the dynamic resource allocation problem in RAN slicing as maximizing network utility while ensuring customized QoS requirements, which is decomposed as slice-level and user-level resource allocation problems.
- We propose a scalable hierarchical deep reinforcement learning-based network slicing framework to solve the two-level constrained optimization problems in the highly dynamic 5G RAN environment, overcoming insufficient exploration and brittle convergence in high-dimensional continuous action space.
- We design a four-stage training pipeline for our framework. The convergence and efficiency are validated by numerical experiments. Compared with existing methods, our approach obtains higher utility in various network traffic scenarios while satisfying QoS requirements.

The rest of this paper is organized as follows. Section II reviews the related works. Section III introduces the RAN slicing architecture and formulates the resource allocation problem. Section IV decomposes the problem into slice-level and user-level resource optimization and proposes an HDRL framework and a four-stage training pipeline. Section V evaluates the performance of our framework. Finally, Section VI concludes the paper.

II. Related Work

In this section, we introduce optimal resource allocation in network slicing in Section II-A and deep reinforcement learning techniques in Section II-B.

A. Optimal Resource Allocation in Network Slicing

Currently, existing works for network slicing management are dedicated to resource allocation problems. Shi et al. [10] applied Q-learning to maximize network slicing utility in terms of diversified user requests via dynamic resource assignment. Liu et al. [14] proposed an alternating direction method of multipliers (ADMM) solution to solve a series of sub-problems in optimal resource allocation, integrating convex optimization and deep reinforcement learning. Hua et al. [15] proposed a Wasserstein generative adversarial network (WGAN) empowered dueling deep distributional Q-network to estimate the Q value distribution, which is efficient in managing radio resources in RAN slicing. Wang et al. [6] proposed a novel DRL approach named twin-actor Deep Deterministic Policy Gradient (DDPG) to jointly optimize communication, computing, and caching resources allocation in multi-access edge network slicing at the slice level and the user level. The proposed approach can maximize network utility while ensuring QoS. Dong et al. [16] designed a novel DRL-based framework for joint optimization of network routing control and dynamic resource management. They integrated a graph convolutional network (GCN) with differentiable pooling into DRL to capture non-Euclidean topology information and graph-structured network status. Wu et al. [17] proposed a DRL-based constrained learning-driven dynamic RAN slicing framework for service-oriented vehicular networks. This hierarchical optimization framework formulates the RAN slicing as a constrained stochastic optimization problem, where DRL is applied to optimize resource allocation at the outer layer while convex optimization is applied for workload distribution optimization at the inner layer. Naeem et al. [12] proposed a digital twin-empowered network slicing framework with a graph neural network (GNN) and distributional deep Q-networks integrated for feature embedding and resource allocation, respectively. Their simulation experiments illustrate that representation learning with GNN can promote DRL.

Most of these existing studies focus on the dynamic resource allocation problem in network slicing and investigate the application and expansion of deep reinforcement learning solutions in network management. In this paper, we further extend these ideas and introduce multi-agent hierarchical deep reinforcement learning as a novel framework for 5G RAN slicing, which is efficient in solving model-free decision-making problems with high-dimensional continuous state space and action space. An independent and scalable slice management strategy is also enabled under this framework.

B. Deep Reinforcement Learning Techniques

Model-free DRL has gained great popularity ever since it achieves excellent performance on sequential decision-making tasks, e.g., game playing [18], [19] and robotics [20], [21]. Mnih et al. [18] proposed Deep Q-Network (DQN) integrating DNN into Q-learning as action-value (i.e., Q value) approximation adapted to larger action space and state space. This off-policy algorithm also introduced experience replay to achieve greater data efficiency and reduce sample correlations, as well as a target network to enhance stability. However, DQN suffers from the overestimation of Q value since greedy actions are chosen for target Q value estimation. Van Hasselt et al. [19] proposed double DQN to address the overestimation by decoupling action selection and value estimation with double Q-networks. Wang et al. [22] proposed dueling DQN to decompose Q value estimation into separate estimates of state value and action advantages. These value-based algorithms specialize in high-dimensional state space and discrete action space.

Unlike value-based algorithms, policy-based algorithms can handle continuous action space. Vanilla policy gradient algorithm [23] directly updates a policy network via Monte-Carlo sampling, suffering from sampling inefficiency, high variance and noisy gradients. Actor-critic algorithms integrate critic networks for value estimation and actor networks for action selection. Mnih et al. [24] proposed Asynchronous Advantage Actor-Critic (A3C) that estimated action advantage and replaced experience replay with parallel multi-agent exploration to improve stability and efficiency while involving far less computational resource. Schulman et al. [25] proposed Trust Region Policy Optimization (TRPO) to improve the policy monotonically by constraining the Kullback-Leibler (KL) divergence of the new policy from the old one in a reliable range. To reduce the computational complexity of TRPO, Schulman et al. [26] proposed Proximal Policy Optimization (PPO) to optimize a clipped surrogate objective or an adaptive KL-penalized objective instead. Lillicrap et al. [27] proposed DDPG by integrating DQN [18] and policy gradient [23] to learn a deterministic policy. However, DDPG inherits overestimation bias from Q-learning and brittle convergence from policy gradient. Fujimoto et al. [13] proposed an improved version named TD3, which designed a clipped double Q-learning variant to address the overestimation of Q value and introduced delayed policy updates to address the coupling of value and policy. Besides, TD3 proposed a novel regularization for target policy smoothing to alleviate overfitting.

Recently, deep reinforcement learning has drawn much attention in networking. Many recent DRL-based techniques are now among the state of the arts for a variety of networking and systems adaptation problems, including congestion control [28], adaptive bitrate streaming [11], computation offloading [29], wireless resource scheduling [30], and cloud scheduling [31]. This paper aims to leverage the emerging DRL techniques to optimize resource allocation for network slicing. Besides, we further improve the scalability and efficiency of our HDRL framework by analyzing the training details and designing a novel four-stage training process including knowledge transfer and collaborative training.

III. System Model

We focus on the dynamic resource allocation problem for 5G RAN with a base station and multiple network slices. The users physically associate with the physical network, while logically connect to virtual slices. The problem can be described as network utility maximization (NUM) while ensuring the minimum QoS requirements. Utility is a quantitative description of network utilization and user satisfaction, whose different definitions would lead to different resource allocation strategies. We use α-fairness [32], a widely adopted scheme, to quantitatively represent the utility. Let $\mathcal{I}$ and $\mathcal{E}_i$ be the set of network slices and the user equipments (UEs) of the *i*th network slice, respectively. Denote $y_{i,k}^{(t)}$ as the wireless data rate of the *k*th user served by the *i*th network slice at the *t*th time slot, then the network slice utility can be defined as a non-decreasing function of data rate according to the α-fairness scheme as follow:

$$U_i^{(t)} = \sum_{k\in\mathcal{E}_i} w_{i,k}\, U_{i,k}\left(y_{i,k}^{(t)}\right) = \sum_{k\in\mathcal{E}_i} \frac{w_{i,k}\left[y_{i,k}^{(t)}\right]^{1-\alpha}}{1-\alpha}, \quad (1)$$

where $U_i^{(t)}$ and $U_{i,k}(y_{i,k}^{(t)})$ denote the utility of the *i*th network slice at the *t*th time slot and the utility of the *k*th equipment served by the *i*th network slice, and $w_{i,k}$ denotes the priority weight of the *k*th equipment served by the *i*th network slice. When $\alpha \to 0$, α-fairness maximizes network throughput. When $\alpha \to 1$, α-fairness approximates proportional fairness which tends to configure the data rate of each equipment to be the best. When $\alpha = 2$, α-fairness is equivalent to potential delay minimization and optimizes the network latency. When $\alpha \to \infty$, α-fairness is equivalent to max-min fairness which tends to optimize the lowest data rate of the network [33]. In our system model, different network slices use different α to achieve diversified QoS requirements.

In network slicing, there is a complex relationship between data rate and network status including network configuration, topology, traffic and resource allocation, etc. Network traffic would flow through predefined ordered VNFs deployed on various equipment, then the network queuing delay, processing delay, packet loss rate and other network key performance indicators (KPIs) would be greatly influenced especially in network congestion, and the performance degradation of any VNF can affect the network slicing performance. Network resources including computation, communication and cache resources can also influence multiple network KPIs. For simplification, we mainly consider the impact of network traffic and resource allocation on network data rate, since the impacts of other factors can be regarded as constant in our network system. Denote the network resource allocated to the *k*th equipment served by the *i*th slice at time slot t as $x_{i,k}^{(t)}$ and the network traffic at time slot t as $\theta^{(t)}$, data rate can be modeled as:

$$y_{i,k}^{(t)} = g_{i,k}\left(x_{i,k}^{(t)}; \theta^{(t)}\right). \tag{2}$$

Actually, wireless data rate has no closed-form expression but it can be measured in real time. Denote *R* and $U_{i,k}^{min}$ as the network resource capacity of the whole system and the minimum QoS requirement of the *k*th UE served by the *i*th slice. The optimization objective is to maximize the weighted sum of network utility of slices in a long period $\mathcal{T}$ with limited resources while fulfilling personalized minimum QoS requirements of various users. Therefore, we formulate the dynamic resource allocation problem for multiple network slices as:

$$\mathcal{P}_1 : \max_{\{x_{i,k}^{(t)}\}} \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} U_i^{(t)}(x;\theta,w,\alpha)$$
$$s.t. \begin{cases} C_1 : U_{i,k}^{(t)} \geq U_{i,k}^{min}, \forall t \in \mathcal{T}, i \in \mathcal{I}, k \in \mathcal{E}_i \\ C_2 : 0 \leq x_{i,k}^{(t)} \leq R, \forall t \in \mathcal{T}, i \in \mathcal{I}, k \in \mathcal{E}_i \\ C_3 : 0 \leq \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{E}_i} x_{i,k}^{(t)} \leq R, \forall t \in \mathcal{T} \end{cases} \tag{3}$$

In the formulation above, constraints C_1 ensure that the resource allocation can meet the personalized QoS requirements, constraints C_2 and C_3 restrict the resource allocated to all equipment should not surplus the total amount of resources.

IV. Design

In this section, we propose an HDRL-based dynamic resource allocation algorithm to maximize the weighted sum of the network utility of multiple slices while satisfying the personalized QoS requirements. However, the optimization problem $\mathcal{P}_1$ corresponds to high-dimensional state space and action space, due to network complexity and resource diversity, respectively. Hence it is difficult for single-agent deep reinforcement learning to explore the environment sufficiently and converge to a well-trained policy stably. To deal with insufficient exploration in a complicated environment [34], we propose to use hierarchical deep reinforcement learning, which divides the task into several sub-problems (Section IV-A). Each sub-problem is modeled as a Markov decision process (Section IV-B) and conquered by TD3 algorithm (Section IV-C). A series of hierarchically distributed agents of HDRL can learn atomic policies towards easier goals by sufficiently exploring the lower-dimensional action spaces and state spaces corresponding to the sub-problems (Section IV-D). The intrinsic behavior patterns of hierarchically distributed agents eventually make up an effective policy to tackle the original complicated task.

A. Problem Decomposition

As for resource allocation to multiple network slices, $\mathcal{P}_1$ can be naturally divided into a slice-level problem and several

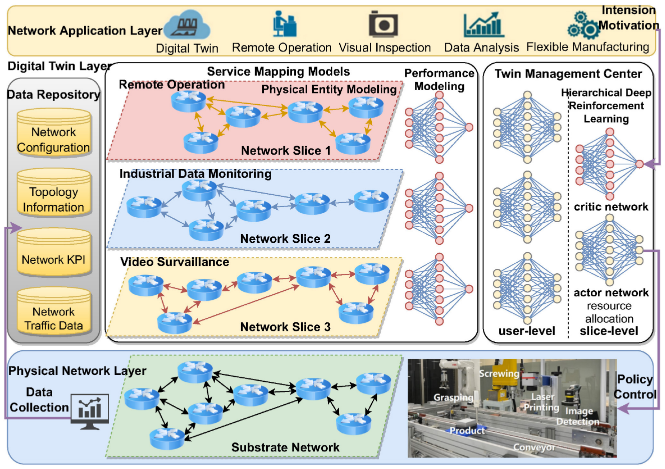


Fig. 1. The 5G radio access network slicing orchestration system empowered by hierarchical deep reinforcement learning, consisting of three layers: physical network layer, digital twin layer and network application layer, and three domains in digital twin layer: data repository, service mapping models and twin management center. The data repository collects and stores data from the substrate network, while the service mapping models establish physical entity representations and performance models. Motivated by the network application layer, the twin management center optimizes policies for physical network management with experience dataset and performance prediction models, satisfying customized QoS requirements of heterogeneous industrial applications and devices.

user-level problems. The slice-level problem is to dynamically allocate virtual resources to network slices by SDN controller. The user-level problems are handled by each slice controller, allocating network resources to user equipment served by each slice. Each sub-problem can be solved with a deep reinforcement learning agent individually, while knowledge transfer and collaborative training can further improve training efficiency and refine the policy, respectively. The system model and HDRL framework are shown in Fig. 1.

Denote the amount of resources allocated to the *i*th slice as z_i, then $z_i = \sum_{k\in\mathcal{E}_i} x_{i,k}$, hence the slice-level problem can be written as:

$$\mathcal{P}_2 : \max_{\{z_i^{(\tau)}\}} \sum_{\tau\subset\mathcal{T}} \sum_{i\in\mathcal{I}} U_{\pi_i}^{(\tau)}(z_i;\theta_i,w_i,\alpha_i)$$
$$s.t. \begin{cases} C_4 : 0 \leq z_i^{(\tau)} \leq R, \forall \tau \subset \mathcal{T}, i \in \mathcal{I} \\ C_5 : 0 \leq \sum_{i\in\mathcal{I}} z_i^{(\tau)} \leq R, \forall \tau \subset \mathcal{T} \end{cases} \tag{4}$$

where π_i is the resource allocation policy of the *i*th slice and $U_{\pi_i}^{(\tau)}$ is the accumulated network utility with penalty of the *i*th slice in a time period τ. Based on $\mathcal{P}_1$ and $\mathcal{P}_2$, the user-level problem to be solved by the *i*th slice can be formulated as:

$$\mathcal{P}_3 : \max_{\{x_{i,k}^{(t)}\}} \sum_{t\in\tau} \sum_{k\in\mathcal{E}_i} w_{i,k} U_{i,k}^{(t)}(x_{i,k};\theta_i,\alpha_i)$$
$$s.t. \begin{cases} C_6 : U_{i,k}^{(t)} \geq U_{i,k}^{min}, \forall t \in \tau, k \in \mathcal{E}_i \\ C_7 : 0 \leq x_{i,k}^{(t)} \leq R, \forall t \in \tau, k \in \mathcal{E}_i \\ C_8 : \sum_{k\in\mathcal{E}_i} x_{i,k}^{(t)} = z_i^{(\tau)}, \forall t \in \tau \end{cases} \tag{5}$$

Actually, it is challenging for deep reinforcement learning to solve this constrained optimization problem directly. As for $\mathcal{P}_3$, we can rewrite constraints C_6 and C_8 as penalty terms added to the objective function, then the optimization problem can be rewritten as:

$$\begin{aligned}\mathcal{P}_4:\ &\max_{\{x_{i,k}^{(t)}\}} \sum_{t\in\tau} U_{\pi_i}^{(t)}\left(x_{i,k}^{(t)}; z_i^{(\tau)}, \theta_i, w_i, \alpha_i\right)\\ &U_{\pi_i}^{(t)} = \sum_{k\in\mathcal{E}_i}\left(w_{i,k}U_{i,k}^{(t)} - H\left(U_{i,k}^{min} - U_{i,k}^{(t)}\right)\right)\\ &\qquad - \beta\left\|\sum_{k\in\mathcal{E}_i} x_{i,k}^{(t)} - z_i^{(\tau)}\right\|_2^2\\ s.t.\ \ &C_7: 0 \le x_{i,k}^{(t)} \le R, \forall t\in\tau, k\in\mathcal{E}_i \end{aligned} \tag{6}$$

where $H(\cdot)$ for QoS penalty is a leaky rectified linear unit (ReLU). Practically, we set the QoS penalty as $H(1.1U_{i,k}^{min} - U_{i,k}^{(t)})$ rather than $H(U_{i,k}^{min} - U_{i,k}^{(t)})$ to reserve a performance margin so that agents can satisfy minimum QoS requirements.

Since constraints C_8 have been changed into a penalty item, the sum of resources allocated to UEs by a network slice would be not equal to the resources allocated by the slice-level resource coordinator strictly. Therefore, the resource allocation may violate constraints C_5 of $\mathcal{P}_2$. We define a proportional reallocation strategy as Eq. (7) to handle this violation so that constraints C_5 of $\mathcal{P}_2$ can be always satisfied.

$$x'_{i,k} = \begin{cases} x_{i,k}, \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{E}_i} x_{i,k}^{(t)} \le R\\ x_{i,k}\cdot\dfrac{R}{\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{E}_i} x_{i,k}}, \ \ else\end{cases} \tag{7}$$

B. Markov Decision Process Modeling

As the complicated constraints of the optimization problems have been tackled by surrogate penalized objectives and the reallocation strategy, we can easily apply DRL to solve these problems. Each sub-problem can be considered as a Markov decision process (MDP): at a given state $s_t \in \mathcal{S}$, the agent chooses an action $a_t \in \mathcal{A}$ to perform according to its policy $\pi: \mathcal{S} \to \mathcal{A}$, then the environment would transfer to a new state s_{t+1} and the agent would gain an immediate reward $r_t = R_{a_t}(s_t, s_{t+1})$. As for a user-level problem $\mathcal{P}_4$, the states correspond to the present network traffic and utility of all user equipment served by the corresponding slice as well as the total available amount of network resources. The actions are the assignments of resources to user equipment. The reward is the slice utility with the penalty, i.e., $U_{\pi_i}^{(t)}$. As for the slice-level problem $\mathcal{P}_2$, the states correspond to the average network traffic and utility of all network slices in a short period τ. The actions are the assignments of resources to network slices. The reward is the sum of accumulated slice rewards in τ with a penalty of re-allocation. DRL can train a multi-step decision policy to maximize the γ-discounted long-term reward of MDP, given by $R_t = \sum_{k=0}^{\infty}\gamma^k r_{k+t}$, where γ is a discount factor determining the priority of short-term rewards.

C. Twin Delayed Deep Deterministic Policy Gradient

Considering the action space is continuous, we employ TD3 algorithm [13] to train a deterministic policy, which considers the interplay between function approximation error in both policy and value updates. TD3 consists of two critic networks $Q_{\theta_1}, Q_{\theta_2}$ with weights θ_1, θ_2, and an actor network π_ϕ with weights ϕ, as well as their target networks $Q_{\theta'_1}, Q_{\theta'_2}$, and $\pi_{\phi'}$ with weights θ'_1, θ'_2 and ϕ'. The critic network is an estimation of Q value while the actor network would choose actions to maximize Q value. As an off-policy algorithm, TD3 stores transitions (s_t, a_t, r_t, s_{t+1}) in the replay buffer $\mathcal{B}$ while exploring the environment. According to Bellman equation, the loss function of the critic network Q_{θ_i} is defined as the mean squared temporal difference error:

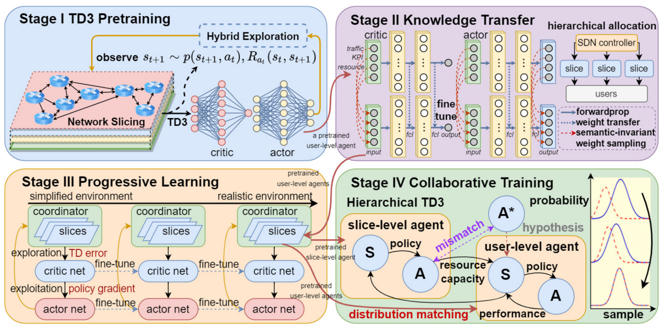


Fig. 2. The four-stage training pipeline of hierarchical TD3 framework.

$$L_i = \left(R_{a_t}(s_t, s_{t+1}) + \gamma \min_{i=1,2} Q_{\theta'_i}(s_{t+1}, \tilde{a}_{t+1}) - Q_{\theta_i}(s_t, a_t)\right)^2, \tag{8}$$

where $\tilde{a}$ is an action chosen by the target actor network, added with clipped noise as regularization:

$$\tilde{a}_{t+1} = \pi_{\phi'}(s_{t+1}) + \epsilon, \epsilon \sim clip(\mathcal{N}(0,\sigma), -c, c). \tag{9}$$

The actor network is updated at a lower frequency than the critic networks according to the following deterministic policy gradient:

$$\nabla_\phi J(\phi) = \nabla_a Q_{\theta_1}(s,a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s). \tag{10}$$

Meanwhile, the target networks update their weights at a lower frequency than the evaluation network:

$$\phi' = \tau\phi + (1-\tau)\phi', \theta'_i = \tau\theta_i + (1-\tau)\theta'_i, i = 1,2, \tag{11}$$

where τ should be a very small positive value. The twin delayed updates of the actor network and target networks according to Eq. (10) and Eq. (11) can improve training stability, overcoming brittle convergence property of DDPG.

D. Hierarchical Reinforcement Learning

We can adopt TD3 algorithm to solve the slice-level problem and user-level problems independently, thus realizing sufficient exploration and scalable management. To improve efficiency and enhance stability, knowledge transfer, and progressive learning are employed for user-level agents and the slice-level agent, respectively. To eliminate distribution mismatching caused by different training settings of hierarchical agents, a collaborative training stage is added to refine value approximation and policy. Hence, we propose a four-stage training pipeline as is presented in Fig. 2.

Stage I (TD3 Pretraining): In the first stage, we train a user-level agent with TD3 algorithm via interaction with the network slicing environment. Different from common problems solved by DRL, resource allocation problems $\mathcal{P}_2$ and $\mathcal{P}_3$ are nonlinear optimization problems with sophisticated constraints. Hence, the action space $\mathcal{A}$ is restricted and changeable with states. The original exploration method of TD3 is to select an action by the actor network with Gaussian noise at each step:

$$a_t = \pi_\phi(s_t) + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma). \tag{12}$$

As for $\mathcal{P}_3$, constraints C_8 would be hardly satisfied with this exploration method, so the immediate reward is always negative and instructive actions from feasible regions are hardly sampled, leading to a bad policy. To handle this problem, we propose a novel hybrid exploration method integrating ϵ-greedy exploration and Gaussian noise exploration: sample an action according to Eq. (12) with probability $1-\varepsilon$, otherwise, select a random action a_t satisfying constraints C_7 and C_8 of $\mathcal{P}_3$. This hybrid exploration method is used for $\mathcal{P}_2$ as well.

Stage II (Knowledge Transfer): In this stage, the other untrained user-level agents are pretrained with prior knowledge transferred from the user-level TD3 agent pretrained in Stage I. Since single slice resource allocation tasks have high similarity, knowledge transfer among user-level TD3 agents can benefit training efficiency and convergence. Knowledge transfer among different user-level neural networks can be implemented via parameter transfer and fine-tuning. Since multi-layer perceptron (MLP) networks are leveraged as critic networks and actor networks, different numbers of users served by different slices correspond to different input and output dimensions, i.e., heterogeneous neural network architectures. Therefore, only partial parameters can be directly copied among user-level agents of different network slices. Considering the task similarity, the weights of connections between neurons with identical semantics (e.g., traffic and identical performance indicator) and other layers are likely to follow similar probability distributions, so pretrained weights can provide a probability distribution prior for untrained weights initialization via semantic-variant weight sampling.

Stage III (Progressive Learning): In this stage, the slice-level TD3 agent would be pretrained through hybrid exploration with the parameters of the pretrained user-level agents fixed. However, the system model is more complicated than a single network slice model, so it is difficult to stably converge to a well-trained policy. We propose a progressive training approach to handle these problems and enhance the training stationarity. We first train the slice-level agent with a simplified reward function and simulation environment, i.e., the heavier penalty for re-allocation and no re-allocation strategy, thus the slice-level agent can learn a vanilla resource allocation policy and adapt to the resource capability. Afterward, the agent will adapt to the realistic reward mechanism and reallocation strategy stably via step-by-step transfer learning.

Stage IV (Collaborative Training): In this stage, collaborative training would eliminate the distribution mismatch between user-level agents and the slice-level agent [20] caused by different pretraining settings and refine their value approximation and policy. At the first two training stages, we randomize the slice resource amount z_i in a much wider range than realistic allocation by the slice-level agent so that pretrained user-level agents can adapt to arbitrary z_i. However, the sampling is sparse in such a large state space, leading to insufficient exploration of user-level agents. This unrealistic hypothesis of the slice-level action's marginal probability distribution leads to distribution mismatch of user-level state space between exploration and exploitation, indirectly resulting in distribution mismatch of slice-level state space between exploration and exploitation (see Fig. 2). A main solution is to replace the slice-level action distribution hypothesis with realistic sampling by slice-level policy, so we propose collaborative training to train the hierarchical distributed agents alternately: the slice-level and user-level agents are trained alternately through TD3 algorithm while other agents fix their network parameters and select greedy actions via actor networks, while user-level agents can be trained in parallel to improve efficiency. Since the realistic standard deviation of slice-level action distribution is smaller, user-level agents can explore the realistic state space more sufficiently and converge to better policies.

V. Evaluation

This section provides detailed simulation results of the proposed hierarchical deep reinforcement learning based dynamic resource allocation framework for 5G RAN slicing orchestration system. The goals of our experiments are three-fold: 1) to validate the effectiveness of deep reinforcement learning in solving non-convex optimization problems for network slicing, 2) to verify the effectiveness and efficiency of the four-stage training pipeline for the proposed HDRL framework, and 3) to demonstrate that our framework is capable of optimizing network utility and fulfilling various personalized QoS requirements via dynamic resource allocation. The rest of this section is organized as follows. In Section V-A, we describe the simulation experiment setup and neural network designation. Finally, the training convergence and performance are validated in Section V-B.

A. Environment Setting

In the simulation environment, we create three network slices and each slice serves five users. To simulate various personalized QoS requirements, we randomize service priority weights $w_{i,k}$, fairness parameters $\alpha_{i,k}$ and minimum QoS requirements $U_{i,k}^{min}$ for each UE, as is defined in α-fairness utility optimization model (see Section III). The network traffic and network performance can be measured in real-time in our simulation environment. According to the problem formulation, the reward function is defined as the sum of network utility, QoS penalty and resource limitation penalty. Hence, we can use episode accumulative reward, episode accumulative utility and QoS penalty as evaluation metrics to evaluate both user-level and slice-level resource allocation strategies.

DNNs and DRL are implemented by Pytorch. Each critic network or actor network is an MLP of 4 fully connected layers, with 256 neurons and 128 neurons in two hidden layers. ReLU is used as an activation function for the hidden layers of all DNNs, while $\tanh$ is used to activate the output layers of

TABLE I
THE AVERAGE PERFORMANCE ON A SINGLE SLICE

Method	Reward	Utility
TD3 with hybrid exploration	**416.52**	**448.94**
DDPG with hybrid exploration	379.67	393.12
DDPG with Gaussian noise	238.45	359.61
Random feasible solution	-95.50	351.38

actor networks. We set the discounted factor γ for cumulative reward to be 0.99, so as to optimize the dynamic resource allocation over a long period.

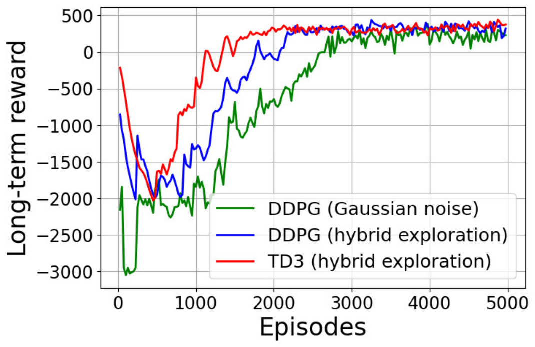
We use an NVIDIA GeForce RTX 3090 GPU to train and test our algorithms. We train all of our TD3 networks by Adam optimizer, using the learning rates of 6e-4 and 3e-4 for critic networks and actor networks, respectively, batch size of 256, target network soft update rate of 0.002, and policy update delay of 5 steps. As for our hybrid exploration method, when training the user-level TD3 agents for slice management, the standard deviation σ of Gaussian noise is 0.02 and the probability ε to sample a non-greedy action is 0.3; when training the slice-level TD3 agent, σ is 0.05 and ε is 0.4.

B. Performance Evaluation

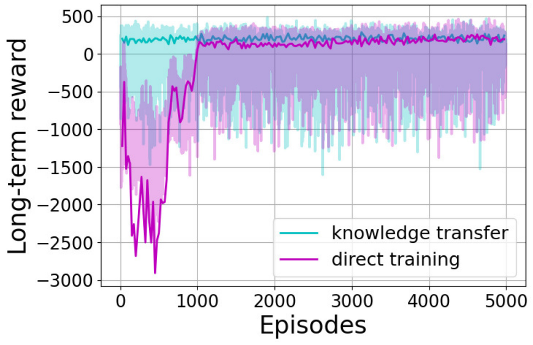
1) Performance Evaluation of Single Slice Management: The hierarchical TD3 framework for dynamic resource allocation is trained through the four-stage pipeline as proposed in Section IV-D. User-level agents for single slice management are pretrained in the first two stages. At TD3 pretraining stage, user-level agents are trained through TD3 algorithm and hybrid exploration strategy individually, given a randomized amount of resources in a single slice. In comparison, We trained a DDPG version without hybrid exploration (with only Gaussian noise) and a DDPG version with hybrid exploration on the same network slice, while the random feasible solution samples actions satisfying constraints C_7 and C_8 of $\mathcal{P}_3$.

The average long-term rewards versus training episodes are shown in Fig. 3. The comparison between the DDPG versions with and without hybrid exploration indicates that hybrid exploration can improve performance at a faster pace and achieve higher training efficiency since the constraints of action selection can be quickly explored with the guidance of random feasible solutions. Compared with DDPG versions, TD3 algorithm further improves training efficiency. The average performance of different pretraining algorithms for single slice management is evaluated in Table I, indicating that deep reinforcement learning is effective in optimizing resource allocation for network slicing, and our training approach leads to better performance.

At the knowledge transfer stage, we directly utilize semantic-variant weight sampling and partial parameter transfer to initialize the untrained user-level agents with the pretrained ones. The untrained user-level agents are fine-tuned with TD3 algorithms and hybrid exploration afterward. The average long-term rewards versus training episodes are recorded in Fig. 3, indicating the stability and efficiency of transfer learning. The performance evaluated by episode cumulative reward is shown in Table II. After parameter transfer,



(a) Pretraining stage



(b) Knowledge transfer stage

Fig. 3. The pretraining and knowledge transfer process of the user-level agents. The pre-training process of user-level TD3 agents is compared with a DDPG version without hybrid exploration and a DDPG version with hybrid exploration. The transfer learning process of user-level TD3 agents is compared with direct training.

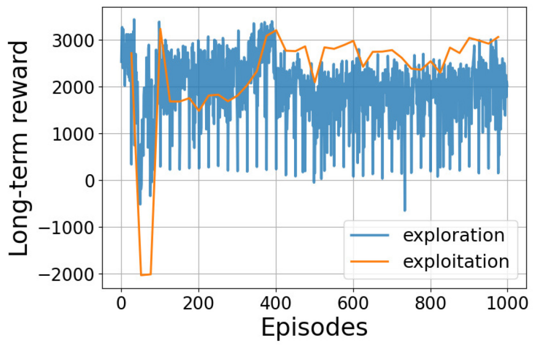
TABLE II
LONG-TERM REWARDS OF KNOWLEDGE TRANSFER AMONG SLICES

Stage	Slice 2	Slice 3
Before transfer learning	167.47	211.05
After transfer learning	**285.74**	**335.70**
Direct training	281.29	327.38
Random feasible solution	-95.50	-99.33

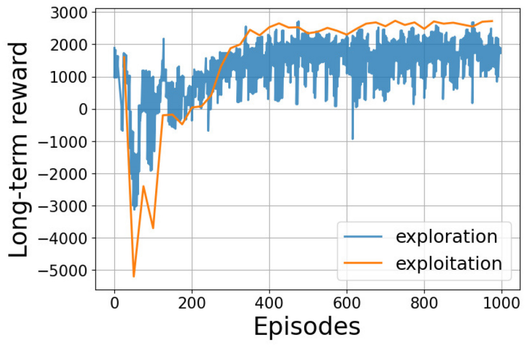
an untrained agent can achieve much better performance than a random strategy, indicating the effectiveness of semantic-variant weight sampling and partial parameter transfer. Fine-tuning achieves good performance similar to direct training but at a faster pace.

2) Overall Performance of the Network Slicing System: Given all of the pretrained user-level agents responsible for resource allocation on a single network slice, we can train a slice-level agent for hierarchical resource allocation. In this part, we will evaluate the overall performance of the HDRL framework for resource allocation in RAN slicing.

The slice-level agent can be trained via either direct training or progressive training with the parameters of user-level agents fixed. The average long-term rewards versus episodes of direct training and progressive training are shown in Fig. 4, which indicate that it is not stable to train the slice-level agent directly in a sophisticated simulation environment. Instead, we can stably pretrain the slice-level agent with a simplified reward mechanism, then we adapt the slice-level policy to the realistic reward mechanism through fine-tuning.



(a) Direct training



(b) Progressive training

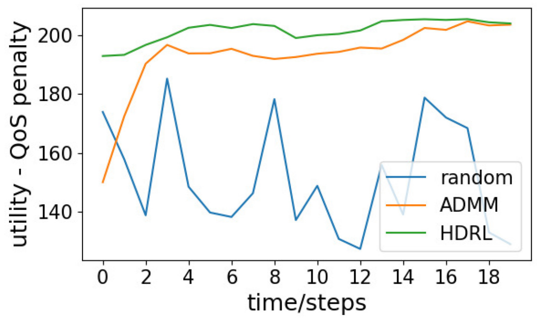
Fig. 4. The progressive learning process of the slice-level agent, compared with direct training.

Given the slice-level agent pretrained in stage III and the user-level agents pretrained in stage I and II, we further train these hierarchically distributed agents collaboratively to eliminate the distribution mismatch. The performance of this HDRL-based network slicing orchestration system is evaluated by episode accumulative reward, episode accumulative utility and QoS penalty, as is shown in Table III. In this table, H-TD3 and H-DDPG refer to hierarchical TD3 framework and hierarchical DDPG framework where the slice-level agent is directly trained without the progressive scheme, respectively, while (III) and (IV) refer to training through the first three stages and training through the four-stage pipeline, respectively. In comparison to the proposed H-TD3 framework, random allocation strategy, ADMM algorithm [14], and H-DDPG are evaluated in the simulation environment as baseline algorithms, with our pretrained user-level agents settling the user-level resource allocation problems.

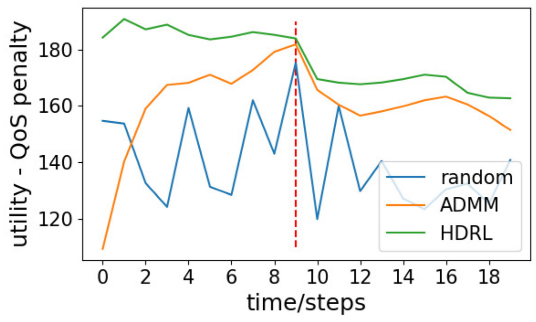
The performance evaluation indicates that the proposed H-TD3 framework with a four-stage training pipeline outperforms all the baseline algorithms in network utility and QoS assurance, which has significant improvements of 25% and 8% relative to random strategy and ADMM strategy in network utility, respectively. Additionally, we observe that the proposed HDRL framework and four-stage training pipeline are suitable for different state-of-the-art DRL algorithms (TD3 and DDPG). In our network slicing orchestration system, H-TD3 can achieve better performance than H-DDPG. By comparing the performance among direct training (H-DDPG and H-TD3), progressive training (H-DDPG (III) and H-TD3 (III)), and collaborative training (H-DDPG (IV) and H-TD3 (IV)), we conclude that progressive training is more effective than direct training, and the collaborative training can significantly improve the overall performance of HDRL-based network slicing system. The QoS penalty is still negative because we reserve a performance margin when training user-level agents, actually the proposed hierarchical TD3 framework can completely fulfill personalized QoS requirements under arbitrary network traffic circumstances.

TABLE III
PERFORMANCE EVALUATION ON THE WHOLE NETWORK SLICING SYSTEM

Method	Reward	Utility	QoS penalty
Random	1110.32	3033.05	-425.66
ADMM [14]	327.14	3510.12	-97.89
H-DDPG	2816.15	3269.66	-77.21
H-DDPG (III)	3026.92	3521.34	-75.52
H-DDPG (IV)	3470.23	3618.59	-61.33
H-TD3	2830.30	3285.36	-76.74
H-TD3 (III)	3112.06	3634.94	-72.91
H-TD3 (IV)	**3561.67**	**3792.30**	**-53.12**



(a) Ordinary case



(b) Congestion case

Fig. 5. Performance of network slicing system in two network traffic cases. In the congestion case, the network traffic increases rapidly from time step 9 (the red line in (b)), resulting in performance degradation. The proposed HDRL strategy outperforms other baseline strategies while it also suffers less performance degradation in network congestion.

3) Adaptability to Various Network Traffic Scenarios: We monitor the performance of network slicing system in two classic network traffic cases: ordinary case and congestion case, as is shown in Fig. 5. Under ordinary network traffic circumstances, our hierarchical TD3 framework can maximize network utility and satisfy minimum QoS requirements in dynamic resource allocation tasks, outperforming ADMM framework and random strategy. Furthermore, our framework can still satisfy the minimum QoS requirements and suffer less performance degradation than ADMM framework under network congestion as the network traffic increases rapidly (after step 9 in the congestion case). Compared with the random strategy and ADMM strategy, our framework can better guarantee minimum QoS requirements and network performance in network congestion.

VI. Conclusion

In this work, we study the task of dynamic resource allocation for 5G radio access network slicing to maximize the network utility while ensuring the customized QoS, which is formulated as a hierarchical constrained optimization problem. We propose a novel hierarchical deep reinforcement learning framework to solve the slice-level and user-level allocation problems efficiently with hierarchical distributed TD3 agents. The framework is trained with hybrid exploration through four stages. Simulation experiments demonstrate that our training approach can enhance convergence and improve efficiency, and the policy is further improved with the distribution mismatch eliminated via collaborative training, enabling scalable management and rapid deployment. Extensive experiments show that the proposed framework outperforms existing methods in network utility maximization and QoS assurance, which improves the network utility by 25% and 8% compared with the random strategy and the ADMM strategy, respectively. Compared with baseline algorithms, our framework achieves better performance in network slicing while fulfilling personalized QoS requirements in diversified network traffic cases.

References

[1] L. Ji, S. He, W. Wu, C. Gu, J. Bi, and Z. Shi, "Dynamic network slicing orchestration for remote adaptation and configuration in Industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 6, pp. 4297–4307, Jun. 2022.

[2] Z. Xia, S. Xue, J. Wu, Y. Chen, J. Chen, and L. Wu, "Deep reinforcement learning for smart city communication networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 6, pp. 4188–4196, Jun. 2021.

[3] K. Zhang, J. Cao, and Y. Zhang, "Adaptive digital twin and multiagent deep reinforcement learning for vehicular edge computing and networks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1405–1413, Feb. 2022.

[4] S. He, K. Shi, C. Liu, B. Guo, J. Chen, and Z. Shi, "Collaborative sensing in Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 3, pp. 1435–1474, 3rd Quart., 2022.

[5] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Comput. Netw.*, vol. 167, Feb. 2020, Art. no. 106984.

[6] Z. Wang, Y. Wei, F. R. Yu, and Z. Han, "Utility optimization for resource allocation in multi-access edge network slicing: A twin-actor deep deterministic policy gradient approach," *IEEE Trans. Wireless Commun.*, vol. 21, no. 8, pp. 5842–5856, Aug. 2022.

[7] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[8] F. Naeem, G. Srivastava, and M. Tariq, "A software defined network based fuzzy normalized neural adaptive multipath congestion control for the Internet of Things," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2155–2164, Oct.–Dec. 2020.

[9] H. Wang, Y. Wu, G. Min, and W. Miao, "A graph neural network-based digital twin for network slicing management," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1367–1376, Feb. 2022.

[10] Y. Shi, Y. E. Sagduyu, and T. Erpek, "Reinforcement learning for dynamic resource optimization in 5G radio access network slicing," in *Proc. IEEE 25th Int. Workshop Comput.-Aided Model. Design Commun. Links Netw. (CAMAD)*, 2020, pp. 1–6.

[11] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proc. Conf. ACM Special Interest Group Data Commun.*, 2017, pp. 197–210.

[12] F. Naeem, G. Kaddoum, and M. Tariq, "Digital twin-empowered network slicing in B5G networks: Experience-driven approach," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, 2021, pp. 1–5.

[13] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor–critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.

[14] Q. Liu, T. Han, N. Zhang, and Y. Wang, "DeepSlicing: Deep reinforcement learning assisted resource allocation for network slicing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2020, pp. 1–6.

[15] 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.

[16] T. Dong et al., "Intelligent joint network slicing and routing via GCN-powered multi-task deep reinforcement learning," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 2, pp. 1269–1286, Jun. 2022.

[17] W. Wu et al., "Dynamic RAN slicing for service-oriented vehicular networks via constrained learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2076–2089, Jul. 2020.

[18] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[19] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double *Q*-learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, 2016, pp. 2094–2100.

[20] K. Xu, H. Yu, Q. Lai, Y. Wang, and R. Xiong, "Efficient learning of goal-oriented push-grasping synergy in clutter," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6337–6344, Oct. 2021.

[21] K. Xu, H. Yu, R. Huang, D. Guo, Y. Wang, and R. Xiong, "Efficient object manipulation to an arbitrary goal pose: Learning-based anytime prioritized planning," 2021, *arXiv:2109.10583*.

[22] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.

[23] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.

[24] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[25] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[27] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[28] N. Jay, N. Rotman, B. Godfrey, M. Schapira, and A. Tamar, "A deep reinforcement learning perspective on Internet congestion control," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3050–3059.

[29] Y. Dai, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning for stochastic computation offloading in digital twin networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 4968–4977, Jul. 2021.

[30] S. Chinchali et al., "Cellular network traffic scheduling with deep reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, 2018, pp. 766–774.

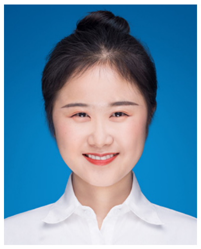
[31] H. Mao, M. Schwarzkopf, S. B. Venkatakrishnan, Z. Meng, and M. Alizadeh, "Learning scheduling algorithms for data processing clusters," in *Proc. ACM Special Interest Group Data Commun.*, 2019, pp. 270–288.

[32] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

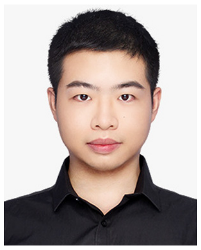
[33] M. Uchida and J. Kurose, "An information-theoretic characterization of weighted alpha-proportional fairness," in *Proc. IEEE INFOCOM*, 2009, pp. 1053–1061.

[34] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 3675–3683.

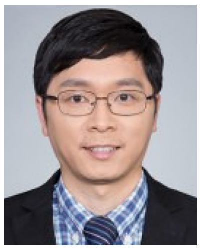
Renlang Huang (Student Member, IEEE) received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 2022, where he is currently pursuing the Ph.D. degree with the College of Control Science and Engineering. His research interests include visual and multi-modal perception, deep learning, and localization and mapping for autonomous mobile robots.



Miao Guo (Student Member, IEEE) received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 2020, where she is currently pursuing the Ph.D. degree with the College of Control Science and Engineering. Her research interests include industrial IoT, deterministic network, and time-sensitive networking.



Chaojie Gu (Member, IEEE) received the B.Eng. degree from the Harbin Institute of Technology, Weihai, China, in 2016, and the Ph.D. degree in computer science and engineering from Nanyang Technological University, Singapore, in 2020. He was a Research Fellow with Singtel Cognitive and Artificial Intelligence Lab for Enterprise in 2021. He is an Assistant Professor with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China. His research interests include IoT, industrial IoT, edge computing, and low power wide area network.



Shibo He (Senior Member, IEEE) received the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2012, where he is currently a Professor. He was an Associate Research Scientist from March 2014 to May 2014 and a Postdoctoral Scholar from May 2012 to February 2014 with Arizona State University, Tempe, AZ, USA. From November 2010 to November 2011, he was a Visiting Scholar with the University of Waterloo, Waterloo, ON, Canada. His research interests include Internet of Things, crowdsensing, and big data analysis.

Jiming Chen (Fellow, IEEE) received the B.Sc. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2000 and 2005, respectively, where he is currently a Professor with the College of Control Science and Engineering and the Deputy Director of the State Key Laboratory of Industrial Control Technology. His research interests include the Internet of Things, sensor networks, networked control, and control system security.

Mingyang Sun (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., in 2017. From 2017 to 2019, he was a Research Associate and a DSI Affiliate Fellow with Imperial College London. He is currently a Professor of Control Science and Engineering under the Hundred Talents Program with Zhejiang University, Hangzhou, China. Also, he is an Honorary Lecturer with Imperial College London. His research interests include AI in energy systems and cyber-physical energy system security and control.