Digital Communications and Networks 12 (2026) 662–676

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

Energy minimized joint resource allocation and task offloading with user association for URLLC and eMBB 5G multi-RAT systems ☆

Dongjun Jung [a], Jong-Moon Chung [a,*], Hea-Sook Park [b]

[a] School of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, South Korea
[b] Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea

ARTICLE INFO

Keywords:
Enhanced mobile broadband
Multi-agent deep reinforcement learning
Multi-access edge computing
Resource allocation
Ultra reliability low-latency communication

ABSTRACT

In this paper, a multi-agent deep reinforcement learning-based joint resource allocation and task offloading control technique is proposed to minimize the energy consumption of user equipment (UE) supporting 5G ultra reliability low latency communications (URLLC) and enhanced mobile broadband (eMBB) services while meeting strict quality of service (QoS) requirements in 5G multi-radio access technology (RAT) networks. An optimization problem involving transmission power, channel resource, user association, offloading rate, and central processing unit (CPU) frequency is formulated using a queueing system-based mathematical design to support services with different characteristics while minimizing the energy consumption. It is proven in this paper that this problem is nondeterministic polynomial (NP) hard, in which multi-agent deep reinforcement learning (DRL) is used to solve the problem. To increase the learning efficiency and stability of deep reinforcement learning, prioritized experience replay (PER) and delayed target network and policy updates are applied. Simulation results show that the proposed scheme provides an improved energy consumption performance compared to the benchmarked schemes.

1. Introduction

Due to the advancement of 5G communication technology, Internet of Things (IoT) networks are evolving to provide a variety of advanced real-time and broadband services. For these services, the Ultra-Reliable and Low Latency Communications (URLLC) or enhanced Mobile BroadBand (eMBB) mode is selected depending on the application requirements. URLLC is designed to provide high reliability and ultra-low latency for 5G applications, such as mission-critical IoT services, eXtended Reality (XR), factory automation, autonomous vehicles, and remote medical surgery [1,2]. On the other hand, eMBB is designed to support broadband wireless and large data processing services for 5G users, such as high-definition video and multimedia streaming [3]. As various heterogeneous services become more diverse, it becomes more difficult for 5G networks to meet different and stringent Quality of Service (QoS) requirements.

In addition, mobile User Equipment (UE) usually has a limited battery life, which directly affects whether users can use them for extended periods of time. Therefore, minimizing the UE's energy consumption is an important factor in providing a better user experience. However, executing latency sensitive and computationally intensive tasks while satisfying the QoS requirements is a burden for UEs with limited resources. As a result, new solutions are needed to address growing computational demands and provide fast service times. Multi-access Edge Computing (MEC) is a promising technology that can solve this heavy burden on the network by processing the UE's workload instead [4].

However, as the network becomes more complex to provide various services, there are more things to consider. The proposed method considers a multi-RAT based offloading method to achieve lower latency and higher energy efficiency of services in IoT networks. In addition, for optimal network resource control according to service type, the UE's Central Processing Unit (CPU) and transmission power are adjusted, and the peripheral MEC selection policy and channel resources for offloading are adjusted. Due to these various parameters and complex network situations, the energy minimization problem becomes a Nondeterministic Polynomial (NP) hard problem, which is very difficult to obtain an optimal solution in real-time.

NP-hard problems usually occur when modeling complex problems such as dynamic communications in the real-world, in which solving these problems requires highly complex environmental models. In or-

☆ Peer review under the responsibility of the Chongqing University of Posts and Telecommunications.

* Corresponding author.
E-mail addresses: dongjunjun@yonsei.ac.kr (D. Jung), jmc@yonsei.ac.kr (J.-M. Chung), parkhs@etri.re.kr (H.-S. Park).

https://doi.org/10.1016/j.dcan.2025.07.002
Received 24 February 2025; Received in revised form 30 June 2025; Accepted 1 July 2025

Available online 4 July 2025

2352-8648/© 2025 Chongqing University of Posts and Telecommunications. Publishing services by Elsevier B.V. on behalf of KeAi Communications Co. Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Table 1
Summary of differences between the proposed J-RATOA and existing studies.

Reference	Offloading Rate	Channel Resources	Transmission Power	CPU Frequency	User Association	Optimization Level
[7], [8]	✓	✗	✗	✗	✗	Single-level
[9]	✓	✗	✓	✗	✗	Multi-level
[10]	✓	✓	✓	✓	✗	Single-level
[11], [16]	✗	✗	✓	✓	✗	Single-level
[12], [17]	✓	✓	✗	✗	✗	Multi-level
[13], [18]	✓	✓	✓	✓	✗	Multi-level
[14], [15]	✓	✓	✗	✗	✓	Multi-level
[19]	✓	✓	✗	✗	✗	Single-level
[20]	✓	✓	✗	✓	✗	Single-level
[21]	✗	✗	✗	✓	✓	Single-level
J-RATOA	✓	✓	✓	✓	✓	Single-level

✓: Considered in optimization, ✗: Not considered in optimization.

der to solve these complex problems, multi-level optimization can be attempted by dividing complex optimization problems into several simple sub-optimization problems and solving them step by step. However, this is not an integrated optimization of all variables based on one objective (such as minimizing the energy consumption), but rather a sub-optimization of different variables at each stage, making it difficult to reach an optimized final result. Therefore, a single-level optimization process is needed to solve these complex problems. Heuristic algorithms such as genetic algorithms are insufficient to model and solve these problems because they require many iterations and have low stability in a dynamic environment. On the other hand, Deep Reinforcement Learning (DRL) can be a powerful tool to model and solve these complex problems [5,6]. The proposed Joint Resource Allocation and Task Offloading with user Association (J-RATOA) method aims to find the optimal offloading rate, UE's operational frequency and transmission power, channel resources allocation, and association relationship between the MEC and UE to minimize the normalized energy consumption through single-level optimization using deep reinforcement learning. Additionally, multi-agent deep reinforcement learning is used for efficient learning in complex real-world problems, where each UE can interact with each other, through cooperation or competition, to achieve their individual goals. The differences between the proposed J-RATOA and existing studies are summarized in Table 1, and more details are described in Section 2. Simulation results show that the proposed J-RATOA method outperforms the benchmarked methods in terms of normalized energy consumption. The main contributions of this article can be summarized as follows.

- To support both 5G URLLC and eMBB services, a mathematical design based on queueing systems is developed to analyze the latency, reliability, and energy consumption on a time slot basis. In order to simultaneously support services with different characteristics, a Processor Sharing (PS) queue is used to solve the problem of having URLLC packets wait unnecessary long times due to eMBB packets that arrived before them. Finally, a mathematical problem of minimizing the normalized energy consumption while satisfying the QoS for each service is formulated.
- This article presents a cross-layer optimization framework for the UE and MEC server to minimize the energy consumption through a single-level optimization process. To minimize the energy consumption, the UE's transmission power and computing resources, association between the UE and nearby MECs, task offloading rate, and resource allocation of the communication channel are adjusted. All these parameters are jointly adjusted to minimize the energy consumption of the UE in a single stage using DRL.
- The proposed scheme applies various enhancements to existing Multi-Agent Deep Deterministic Policy Gradient (MADDPG) methods. First, for learning stability in the network, the delayed target network and policy update method is applied. Not only is the target network update delayed to ensure the stability of main network training, but the actor network's policy update is also delayed to reduce errors that may occur due to the interaction between the actor and critic networks. Additionally, the Prioritized Experience Replay (PER) method is applied to improve the learning speed by sampling better-valued experiences more frequently.

The remainder of this article is organized as follows. Section 2 introduces the related work and Section 3 describes the system model. In Section 4, the analysis of latency, reliability, and energy consumption for offloading is described. Section 5 formulates the problem to minimize the energy consumption and presents how to solve it using multi-agent DRL with improvements. The performance analysis is presented in Section 6, followed by the conclusion in Section 7.

2. Related work

MECs are being widely studied as a technology that can relieve the UE's burden by offloading the UE's tasks and processing them to reduce the UE's energy consumption. The authors of [7] reduced the energy consumption of IoT devices and UEs by cooperatively processing tasks according to their popularity based on the Shapley value. However, this study optimized considering only task offloading between a single UE and a single MEC based on mostly fixed parameters. In [8], computation offloading was optimized to minimize the energy consumption by considering transmission and computation latency in an MEC-enabled Internet of Vehicles (IoV) network, but only offloading from a single vehicle was considered.

As networks become more complex, more factors must be considered to reduce the UE's energy consumption as much as possible, in which joint optimization of the MEC is being actively researched. In [9], transmission power and computation offloading were optimized, and in [10], the authors extended the joint optimization process to subcarrier allocation and computing resource allocation to minimize the UE's energy consumption. The authors of [11] minimized the energy consumption by jointly optimizing the computing resource and transmission power, but the offloading decision was only binary. To reduce the energy consumption of the mobile device, task offloading strategies and wireless resource allocation were jointly optimized in [12], and power allocation and computing resource allocation were further optimized in [13]. In addition to resource allocation, task offloading, and power optimization, the energy consumption also changes depending on which of the adjacent MECs the UE connects to and offloads to, but most studies do not consider this. To solve this problem, [14,15] optimized task offloading, resource allocation, and user association in multiple stages. Nevertheless, in order to reduce the UE's energy consumption as much as possible, all of these controllable parameters need to be optimized together in a single stage.

In order to solve complex problems such as joint optimization, studies that suggest use of Artificial Intelligence (AI), especially reinforcement learning, as a solution are prevalent. Among single-agent reinforcement learning methods, [16] adopted a Soft Actor-Critic (SAC) to

Table 2
Summary of Acronym.

Acronym	Description
AI	Artificial Intelligence
CCDF	Complementary Cumulative Distribution Function
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
CSI	Channel State Information
DU	Delayed Updates of the target network and policies
DQN	Deep Q Network
DRL	Deep Reinforcement Learning
eMBB	enhanced Mobile BroadBand
FCFS	First Come First Served
GA	Genetic Algorithm
IoT	Internet of Things
IoV	Internet of Vehicles
J-RATOA	Joint Resource Allocation and Task Offloading with user Association
mDQNR	multiactor Deep Q Network Resource allocation
MADDPG	Multi Agent Deep Deterministic Policy Gradient
MDP	Markov Decision Process
MEC	Multi-access Edge Computing
MEMP	Minimum Energy Multicast Problem
MRRME	Multi Radio Resource Management Entity
NP	Nondeterministic Polynomial
OFDMA	Orthogonal Frequency Division Multiple Access
PER	Prioritized Experience Replay
PS	Processor Sharing
QoS	Quality of Service
RAT	Radio Access Technology
SAC	Soft Actor Critic
TD	Temporal Difference
TD3	Twin Delayed Deep Deterministic Policy Gradient
UE	User Equipment
URLLC	Ultra Reliability Low Latency Communications
XR	eXtended Reality

Table 3
Summary of Notation.

Notation	Description
N, M, K	Total number of UEs, base station, and RATs
n, m, k	Index of UEs, base stations, and RATs
$\xi = \{u, e\}$	Service type indicator (u: URLLC service, e: eMBB service)
$x_{n,m}$	Association indicator
b_n^{ξ}	Bit size for each task of UE n
c_n^{ξ}	Number of CPU cycles required to process a task of UE n
λ_n^{ξ}	Average task arrival rate of UE n
$o_{n,m,0}^{\xi}$	Local processing rate of UE n
$o_{n,m,k}^{\xi}$	Offloading rate from UE n to MEC server m through RAT k
f_n^{ξ}	CPU frequency of UE n for processing
f_m	Service rate of MEC server m
ρ_m	Workload of the mth MEC server
$r_{n,m,k}^{\xi}$	Achievable data rate for transmission to the MEC server
$p_{n,m,k}^{\xi}$	Transmission power of UE n to MEC server m through RAT k
E_n^{ξ}	Expected normalized energy consumption per bit of UE n
$E_n^{\xi,\mathcal{L}}$	Energy consumption per packet of UE n in local processing
ϵ_n^{u}	Overall packet loss probability of UE n
$\epsilon_n^{u,D}$	Decoding error probability of URLLC packets for UE n
$\epsilon_n^{u,\mathcal{L}}$	Queueing delay violation probability of UE n in local processing case
$\epsilon_n^{u,\mathcal{M}}$	Processing delay violation probability of MEC server m
$\epsilon_n^{u,max}$	Maximum allowable reliability for URLLC service of UE n
$T_n^{\xi,max}$	Maximum allowable latency for each service of UE n
$T_{n,\mathcal{P}}^{\xi,\mathcal{L}}$	Processing delay of UE n in the local processing case
$T_n^{\xi,\mathcal{M}}$	Total latency of the nth UE's MEC server processing
$T_{n,\mathcal{P}}^{\xi,\mathcal{M}}$	Processing delay of MEC server m in the MEC server processing case
$T_n^{e,\mathcal{T}}$	Total multi-RAT transmission delay of UE n
T_n^{e}	Total latency of UE n for eMBB services
$T_n^{u,\mathcal{L}}$	Total latency of the nth UE's local processing for URLLC services
$T_{n,Q}^{u,\mathcal{L}}$	Queueing delay of UE n in local processing for URLLC services
$T_n^{u,\mathcal{T}}$	Transmission delay of UE n for URLLC services
T_s	Duration of one time slot

solve the joint transmit power, transmission time, and computing resource optimization problem with multiple deeply coupled variables, and [17] adopted Deep Q-Network (DQN) to optimize the offload scheduling and resource allocation problem in the discrete action space. Multi-agent reinforcement learning techniques are additionally being applied for efficient learning in complex real-world problems. Using multi-agent DQN, [18] proposed an optimized heterogeneous URLLC and eMBB task offloading and resource allocation scheme for multi-RAT systems, and [8] proposed an optimized computation offloading scheme for IoV networks to minimize the energy consumption. In addition to DQN, which has limitations in the discrete action space, continuous action space approaches are also being studied to solve complex problems in the real world. In [19], MADDPG based on a continuous action space was adopted as a method to optimize task offloading and resource allocation to minimize the energy consumption and revenue. Additionally, [20] proposed a method to optimize task offloading, resource allocation, and computing resource allocation to minimize the energy consumption of AR devices using a multi-MEC server based on MADDPG. In [21], a binary offloading and MEC server resource allocation scheme that uses a user association method to minimize the energy consumption based on an Independent Parameterized Deep Q-Network (I-PDQN) was proposed.

In addition to the traditional task offloading optimization, recent research has also focused on integrating green energy harvesting mechanisms into MEC systems. Although this paper does not directly model energy harvesting, the proposed approach fits this trend by promoting efficient resource utilization and reducing energy consumption at the UEs through MEC operations. The authors of [22] proposed a minimized energy consumption scheme by jointly adjusting the offloading rate, transmission power, and CPU frequency through fairness-aware computation offloading optimization in an energy harvesting MEC environment. In addition, in [23], a scheme that optimizes task offloading decisions and CPU frequency control based on Twin Delayed Deep Deterministic policy gradient (TD3) to maximize system utility in an MEC system that applied hybrid energy harvesting was proposed.

Despite the existence of these various studies, the studies so far have not been able to perform optimal user association, task offloading, and resource allocation to minimize the energy consumption of the UEs by considering the different characteristics of URLLC and eMBB services. Even in most cases, it is difficult to achieve the goal of minimizing the energy consumption due to the multi-level joint optimization tasks. To solve these problems, it is necessary to perform single-level joint optimization to minimize the energy consumption through appropriate control of channel resources, offloading rate, computing resources, transmission power, and user association. To solve the above challenging issues, this study simultaneously optimizes the transmission power, channel resources, user association, offloading rate, and the computing resources in a single stage with a single-level joint optimization process and makes improvements to the existing reinforcement learning architecture to enable it to solve these complex problems. (See Tables 2 and 3.)

3. System model

3.1. Network model

Fig. 1 shows an overview of the proposed MEC system model. The computation tasks generated by the UE with eMBB service or UE with URLLC service can be offloaded to one of the BSs with a MEC server for processing. There are N UEs and M base stations in the network. Each UE is indexed by n, where $n \in \boldsymbol{N} = \{1, \ldots, N\}$. The BSs are connected through the Xn interface and are assumed to know the state of the entire network required for learning [24]. Each base station is indexed by m, where $m \in \boldsymbol{M} = \{1, \ldots, M\}$. Each base station is assumed to be a 5G gNB functional base station equipped with a MEC server for analytical simplicity [25]. Since each MEC server is connected 1:1 to each base station, in this paper, both the base station and the MEC server are equally

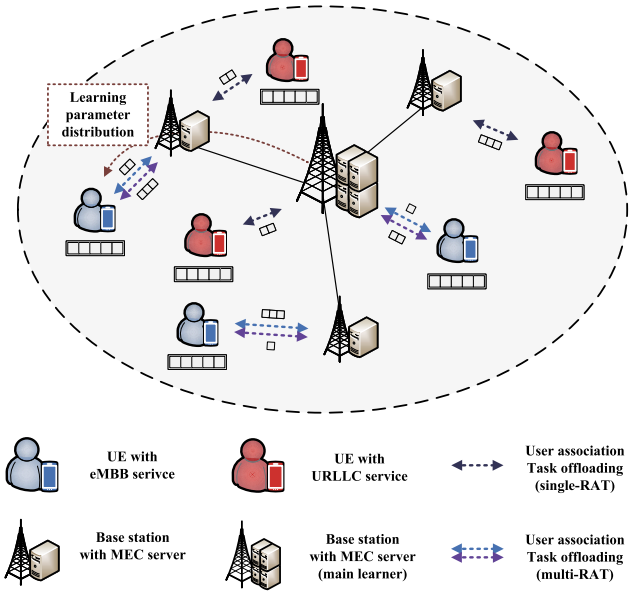


Fig. 1. Overview of the system model.

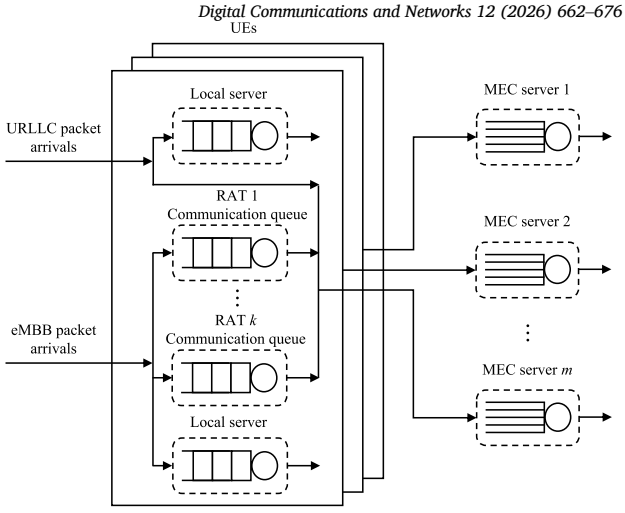


Fig. 2. Queueing model for UEs and MEC servers.

expressed as m. Each UE is equipped with a multi-RAT system, where each RAT is indexed by k and $k \in \boldsymbol{K} = \{1, \ldots, K\}$. For decentralized execution of deep reinforcement learning, the learning parameters trained from the MEC server, which acts as the main learner, are distributed to each UE.

All RATs are connected to a Multi-Radio Resource Management Entity (MRRME) through low latency fronthaul links [26]. Since different RATs operate in different frequency spectrum bands (e.g., 2.4 GHz for Wi-Fi, 1.8 to 2.3 GHz for LTE with larger coverage, and 3.7 to 4.2 GHz for 5G midband), it is assumed that there is no cross-RAT interference in this system [27].

Each UE makes offloading decisions by adjusting the resource parameters (e.g., transmission power, local CPU frequency, and offloading ratio) and the association status with the BS, and determines the resource allocation policy by adjusting the transmission bandwidth to minimize the energy consumption. In this paper, time is divided into slots according to the 5G NR architecture, and the time slot duration is denoted as T_s [28]. For each time slot duration, the UE determines the resource allocation and task offloading policies. It is assumed that downlink transmission takes 1 slot time as in [14,28].

3.2. Task models

Task models are divided into two categories, which are URLLC and eMBB services. The service type indicator $\xi = \{u, e\}$ represents u for URLLC and e for eMBB. For URLLC services, it is assumed that each task can be included in one packet [14]. For eMBB services, the task can be split into multiple packets and partially offloaded through multi-RAT.

The computational task of the nth UE is characterized by $\mathbb{I}_n^{\xi} = (\mathbb{Q}_n^{\xi}, b_n^{\xi}, c_n^{\xi}, \lambda_n^{\xi})$, where $\mathbb{Q}_n^{\xi}$ is the QoS constraint of the service, b_n^{ξ} (bits/task) is the bit size for each task, c_n^{ξ} (cycles/task) is the number of CPU cycles required to process each task, and λ_n^{ξ} (tasks/slot) is the average task arrival rate. QoS constraints $\mathbb{Q}_n^{\xi} = \{T_n^{\xi,max}, \varepsilon_n^{\xi,max}\}$ represent the maximum allowable latency and reliability for each service, respectively. In addition, c_n^{ξ} is expressed as $c_n^{\xi} = b_n^{\xi} a_n$, where a_n (cycles/bit) is the number of CPU cycles required to process one bit, which depends on the computational complexity [29]. It is assumed that the task arrival rate of URLLC follows the Bernoulli process. Unlike URLLC services with constant task sizes and required CPU cycles, the task sizes, required CPU cycles, and inter-arrival time between packets of eMBB services can follow any general distribution.

3.3. User association and offloading policy

3.3.1. User association

Each UE can be associated with one of the base stations. The association indicator $x_{n,m}$ represents a binary variable indicating the associating relationship between the UE and the base station as shown in the following equation.

$$x_{n,m} = \begin{cases} 1, & \text{if UE } n \text{ is associated with BS } m \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Each UE can process tasks while being connected to one base station, or process them on a local server, so $\sum_{m \in \boldsymbol{M}} x_{n,m} \leq 1$, $n \in \boldsymbol{N}$. The association scheme of UE n is expressed as $\boldsymbol{x}_n = [x_{n,1}, \ldots, x_{n,M}]^T$.

3.3.2. Offloading policy

Offloading policies for URLLC and eMBB services are applied differently. Because URLLC packets are very short (e.g., 32 bytes [30]), single RAT offloading over a wireless link with the best condition would be appropriate, since partial offloading may incur additional overhead due to packet splitting and reordering, which is generally not beneficial for short URLLC packets. On the other hand, eMBB packets with long packet sizes (e.g., video images [3]) are split into multiple packets and offloaded through multi-RAT parallel transmission to reduce the service delay time. For complete processing of a packet, the total sum of the offload splits must be 1, so if UE n offloads tasks to the associated MEC server m, the following equation must be satisfied

$$o_{n,m,0}^{\xi} + \sum_{k \in \boldsymbol{K}} o_{n,m,k}^{\xi} = 1 \tag{2}$$

where $o_{n,m,0}^{\xi}$ is the local processing rate of UE n, and $o_{n,m,k}^{\xi}$ is the offloading rate from UE n to MEC server m connected through the kth RAT.

3.4. Queueing model

Fig. 2 represents the queueing model of the proposed system. Each UE can generate URLLC or eMBB packets, and each packet will be processed based on First-Come-First-Served (FCFS) scheduling in the local server or offloaded to the MEC server. For URLLC services, it is assumed that each packet is transmitted within 1 slot [14], [28]. The packet arrival rate of URLLC follows a Bernoulli distribution, so the peak arrival rate is 1 slot. Since the peak arrival rate is equal to the transmission

rate, there is no queue to wait before uplink transmission. For eMBB services, the peak arrival rate may be greater than the transmission rate, so some packets may have to wait in the communication queue before uplink transmission. Nevertheless, for eMBB traffic, the communication queue is modeled, but due to the relatively relaxed latency and reliability requirements compared to URLLC traffic, the analysis is simplified by only imposing a queue stability constraint. Specifically, keeping the queue utilization less than 1, which will be presented in Section 4.2, is sufficient to ensure a bounded average queuing delay without detailed delay distribution analysis.

Both URLLC and eMBB packets may exist in the MEC server. If the MEC server processes packets using the FCFS method, a short URLLC packet following a long eMBB packet must wait until the long eMBB packet is processed. To avoid the unnecessary long queueing delays, the MEC server adopts a PS server. If there are i packets in MEC server m with service rate f_m, the service rate of each packet becomes f_m/i.

3.5. Communication model

Packets from each UE are offloaded to the MEC server through a wireless link based on an Orthogonal Frequency Division Multiple Access (OFDMA) system. The total bandwidth is equally allocated to N_{max} subcarriers, each with a bandwidth of W. The number of resource blocks allocated for transmission to MEC server m through RAT k for each UE is $N^{\xi}_{n,m,k}$.

3.5.1. Achievable data rate for URLLC service

In the case of URLLC, because the packet size is very small, the bandwidth $N^{u}_{n,m,k}W$ and transmission time are assumed to be smaller than the coherence bandwidth and channel coherence time, respectively. Therefore, each packet is transmitted through a quasi-static flat fading channel [31].

According to [31], if UE n transmits a task to MEC server m through RAT k, the achievable data rate in short block length over a quasi-static flat fading channel can be approximated by

$$r^{u}_{n,m,k} \approx \frac{N^{u}_{n,m,k}W}{\ln 2}\left[\ln\left(1+\frac{\alpha^{u}_{n,m,k}g^{u}_{n,m,k}p^{u}_{n,m,k}}{N^{u}_{n,m,k}WN_0}\right) - \sqrt{\frac{V^{u}_{n,m,k}}{T_s N^{u}_{n,m,k}W}} f_Q^{-1}(\varepsilon^{u,\mathcal{D}}_n)\right] \tag{3}$$

where $N^{u}_{n,m,k}$ is the number of resource blocks in the frequency domain, W is each resource block's size in the frequency domain, N_0 is the noise spectral density, $\alpha^{u}_{n,m,k}$ and $g^{u}_{n,m,k}$ are large-scale and small-scale channel gains, respectively, $p^{u}_{n,m,k}$ is the transmission power for URLLC services, f_Q^{-1} is the inverse of the Q-function, $\varepsilon^{u,D}_n$ is the decoding error probability of URLLC packets, and the channel dispersion is $V^{u}_{n,m,k} = 1 - \frac{1}{\left(1+\frac{\alpha^{u}_{n,m,k}g^{u}_{n,m,k}p^{u}_{n,m,k}}{N_0W}\right)^2}$.

3.5.2. Data rate for eMBB service

For eMBB services with large packet sizes, the achievable data rate based on Shannon's capacity formula can be expressed as

$$r^{e}_{n,m,k} = N^{e}_{n,m,k}W\log_2\left(1+\frac{\alpha^{e}_{n,m,k}g^{e}_{n,m,k}p^{e}_{n,m,k}}{N^{e}_{n,m,k}WN_0}\right) \tag{4}$$

where $N^{e}_{n,m,k}$ is the number of resource blocks in the frequency domain, $\alpha^{e}_{n,m,k}$ and $g^{e}_{n,m,k}$ are large-scale and small-scale channel gains, respectively, and $p^{e}_{n,m,k}$ is the transmission power for eMBB services.

4. Analysis of latency, reliability, and energy consumption

4.1. URLLC service analysis

URLLC packets can be processed locally at the UE or at the MEC server (i.e., $o^{u}_{n,m,0}$, $o^{u}_{n,m,k} \in \{0,1\}$, $\forall k \in \boldsymbol{K}$ and Eq. (2)). The latency, reliability, energy consumption, and QoS constraint model for each case are depicted in the following sections.

4.1.1. Local processing

If the packet is processed locally at the UE, the processing delay of UE n is

$$T^{u,\mathcal{L}}_{n,\mathcal{P}} = \left\lceil \frac{c^{u}_{n}}{f^{u}_{n}T_s} \right\rceil \text{ (slots)} \tag{5}$$

where f^{u}_{n} is the CPU frequency of UE n for processing the URLLC task. The notation $\mathcal{L}$ indicates the case of a local UE and $\mathcal{P}$ indicates the case of processing.

The local queueing model for URLLC service can be expressed using a Geo/D/1/FCFS model, where "Geo" refers to a geometric distributed interarrival time between packets and "D" refers to a deterministic service rate. The total latency of the UE's local processing is obtained from the sum of processing delay and queueing delay $T^{u,\mathcal{L}}_{n,Q}$, and is expressed as

$$T^{u,\mathcal{L}}_{n} = T^{u,\mathcal{L}}_{n,\mathcal{P}} + T^{u,\mathcal{L}}_{n,Q} \tag{6}$$

where the notation Q indicates the case of a queue. For the UE's local processing, the QoS constraint for delay is expressed as

$$T^{u,\mathcal{L}}_{n} \le T^{u,max}_{n} \tag{7}$$

where $T^{u,max}_{n}$ is the latency requirement of the URLLC service.

The Complementary Cumulative Distribution Function (CCDF) of queueing delay in the Geo/D/1/FCFS model has been obtained in [32] as presented in Eq. (8).

$$\Pr\left\{T^{u,\mathcal{L}}_{n,Q} > i\right\} = 1-(1-\lambda)^{-i-1}\left(1-\lambda T^{u,\mathcal{L}}_{n,\mathcal{P}}\right) \tag{8}$$

Given the latency requirement $T^{u,max}_{n}$, the delay violation probability at the local UE $\varepsilon^{u,L}_{n}$ can be obtained by substituting $i = T^{max}_{n} - T^{u,\mathcal{L}}_{n,\mathcal{P}}$ and $\lambda = \lambda^{u}_{n}T_s$ into Eq. (8) [28]. Therefore, the queueing delay violation probability at the local UE is as follows.

$$\begin{aligned}\varepsilon^{u,\mathcal{L}}_{n} &= \Pr\left\{T^{u,\mathcal{L}}_{n,Q} > \left(T^{u,max}_{n} - T^{u,\mathcal{L}}_{n,\mathcal{P}}\right)\right\} \\ &= 1-\left(1-\lambda^{u}_{n}T_s\right)^{-\left(T^{max}_{n}-T^{u,\mathcal{L}}_{n,\mathcal{P}}\right)-1}\left(1-\lambda^{u}_{n}T_sT^{u,\mathcal{L}}_{n,\mathcal{P}}\right)\end{aligned} \tag{9}$$

For the UE's local processing, the QoS constraint for reliability is expressed as Eq. (10).

$$\varepsilon^{u,\mathcal{L}}_{n} \le \varepsilon^{u,max}_{n} \tag{10}$$

Let $e^{u,\mathcal{L}}_{n}$ be the energy consumption per CPU cycle of UE n. According to [33], $e^{u,L}_{n} = \kappa\left(f^{u}_{n}\right)^2$ (J/cycle), where κ is the energy coefficient that depends on the processing capacity of the CPU. Therefore, the energy consumption per packet in the case of local processing of UE n can be obtained from Eq. (11).

$$E^{u,\mathcal{L}}_{n} = e^{u,\mathcal{L}}_{n}c^{u}_{n} = \kappa\left(f^{u}_{n}\right)^2 c^{u}_{n} \text{ (J/packet)} \tag{11}$$

4.1.2. MEC processing

The transmission delay when UE n transmits a URLLC task to MEC server m through RAT k is defined as

$$T^{u,\mathcal{T}}_{n} = T^{u,\mathcal{T}_{ul}}_{n} + T^{u,\mathcal{T}_{dl}}_{n} = \left\lceil \frac{b^{u}_{n}}{r^{u}_{n,m,k}T_s} \right\rceil + 1 \text{ (slots)} \tag{12}$$

where the notation $\mathcal{T}_{ul}$ and $\mathcal{T}_{dl}$ indicate the uplink transmission and downlink transmission, respectively.

When the packet is processed by the MEC server, the processing delay of MEC server m is defined as

$$T_{n,\mathcal{P}}^{u,\mathcal{M}} = \left\lceil \frac{c_n^u}{(f_m/i)\,T_s} \right\rceil \text{ (slots)} \tag{13}$$

where the notation $\mathcal{M}$ indicates the case of the MEC server. The total latency of the MEC server processing is obtained from the sum of the transmission delay and processing delay in the MEC server, and is expressed as follows.

$$T_n^{u,\mathcal{M}} = T_n^{u,\mathcal{T}} + T_{n,\mathcal{P}}^{u,\mathcal{M}} \tag{14}$$

When packets are processed on the MEC server, the QoS constraint for delay can be expressed as follows.

$$T_n^{u,\mathcal{M}} \le T_n^{u,max} \tag{15}$$

Each MEC server may serve multiple tasks. The aggregation of the multiple independent Bernoulli processes can be modeled as a Poisson process [34]. Therefore, the MEC server queueing model can be expressed as an M/G/1/PS model, where "M" stands for a memoryless Poisson distributed interarrival time between packets and "G" represents a general distribution service rate.

Lemma 1. *The CCDF of the processing delay of the URLLC packets in the M/G/1/PS model is* $\Pr\left\{W_m > \left\lceil \frac{c_n^u(q+1)}{f_m T_s} \right\rceil\right\} \approx \Pr\{Q_m > q\} = \rho_m^{q+1}$. ■

Proof. The MEC server may have short URLLC and long eMBB packets together. Because short URLLC packets are much shorter than the eMBB packets, the processing delay is also much shorter. When a URLLC packet arrives at server m, let Q_m be the number of packets that already exist in the server. If $Q_m = q$, then the service rate allocated for URLLC packet processing can be approximated as $f_m/(q+1)$, where $q = 0, 1, \ldots$. Therefore, the processing delay of the URLLC packet at this time is approximated as Eq. (16).

$$W_m|_{Q_m=q} \approx \left\lceil \frac{c_n^u(q+1)}{f_m T_s} \right\rceil \text{ (slots)} \tag{16}$$

According to [35], in the M/G/1/PS model, the distribution of the number of packets Q_m in the server can be expressed as

$$\Pr\{Q_m = q\} = \rho_m^q(1-\rho_m) \tag{17}$$

and the workload of the MEC server m ρ_m is

$$\rho_m = \frac{\sum_{n\in\boldsymbol{N}^u} x_{n,m}\left(1-o_{n,m,0}^u\right)\lambda_n^u c_n^u + \sum_{n\in\boldsymbol{N}^e} x_{n,m}\left(1-o_{n,m,0}^e\right)\lambda_n^e \overline{c_n^e}}{f_m} \tag{18}$$

where $\boldsymbol{N}^u$ and $\boldsymbol{N}^e$ represent the set of URLLC service users and the set of eMBB service users. From Eq. (16) and Eq. (17), the following equation can be derived.

$$\Pr\left\{W_m = \frac{c_n^u(q+1)}{f_m T_s}\right\} \approx \Pr\{Q_m = q\} = \rho_m^q(1-\rho_m) \tag{19}$$

From Eq. (19), the Cumulative Distribution Function (CDF) of Q_m is derived as

$$\begin{aligned}\Pr\{Q_m \le q\} &= \sum_{i=0}^{q} \Pr\{Q_m = i\} \\ &= \sum_{i=0}^{q} \rho_m^i(1-\rho_m) \\ &= \frac{(1-\rho_m)(1-\rho_m^{q+1})}{1-\rho_m} \\ &= 1-\rho_m^{q+1}\end{aligned}$$

Therefore, the CCDF of the processing delay of URLLC packets in the M/G/1/PS model can be derived as

$$\Pr\left\{W_m > \frac{c_n^u(q+1)}{f_m T_s}\right\} \approx \Pr\{Q_m > q\} = 1 - \Pr\{Q_m \le q\} = \rho_m^{q+1} \tag{20}$$

and the approximation in Eq. (20) is accurate when $c_n^u \ll c_n^e$ [28]. ■

From the CCDF in Eq. (20), the processing delay violation probability at the MEC server can be derived as Eq. (21).

$$\begin{aligned}\varepsilon_n^{u,\mathcal{M}} &= \Pr\left\{T_{n,\mathcal{P}}^{u,\mathcal{M}} > \left(T_n^{u,max} - T_n^{u,\mathcal{T}}\right)\right\} \\ &= \rho_m^{\frac{f_m\left(T_n^{u,max}-T_n^{u,\mathcal{T}}\right)T_s}{c_n^u}}\end{aligned} \tag{21}$$

The decoding error probability can be obtained from Eq. (3) and can be expressed as follows.

$$\varepsilon_n^{u,\mathcal{D}} \approx f_Q\left(\sqrt{\frac{T_s N_{n,m,k}^u W}{V_{n,m,k}^u}}\left[\ln\left(1+\frac{\alpha_{n,m,k}^u g_{n,m,k}^u p_{n,m,k}^u}{N_{n,m,k}^u W N_0}\right) - \frac{b_n^u \ln 2}{T_s N_{n,m,k}^u W}\right]\right) \tag{22}$$

Due to the processing delay violation and decoding error, the overall packet loss probability can be expressed as

$$\varepsilon_n^u = 1 - \left(1-\varepsilon_n^{u,\mathcal{M}}\right)\left(1-\varepsilon_n^{u,\mathcal{D}}\right) \approx \varepsilon_n^{u,\mathcal{M}} + \varepsilon_n^{u,\mathcal{D}} \tag{23}$$

where, in the URLLC service, $\varepsilon_n^{u,\mathcal{M}}$ and $\varepsilon_n^{u,\mathcal{D}}$ are very small, so the approximation is accurate. In the case of the MEC server processing, the QoS constraint for reliability is expressed as Eq. (24).

$$\varepsilon_n^u \le \varepsilon_n^{u,max} \tag{24}$$

The upper bound of the processing delay violation and decoding error probability are set equal as follows for minor power loss, as shown in [28].

$$\varepsilon_n^{u,\mathcal{M}} \le 0.5\varepsilon_n^{u,max}, \quad \varepsilon_n^{u,\mathcal{D}} \le 0.5\varepsilon_n^{u,max} \tag{25}$$

From Eq. (21) and Eq. (25), the workload constraint of the MEC server can be derived as follows.

$$\rho_m \le \left(0.5\varepsilon_n^{u,max}\right)^{\left\lceil \frac{c_n^u}{f_m\left(T_n^{u,max}-T_n^{u,\mathcal{T}}\right)T_s}\right\rceil} \triangleq \rho_{th} \tag{26}$$

The total energy consumption for URLLC services can be formulated from the UE's local processing energy and transmission energy. Considering the offloading rate and user association, the expected normalized energy consumption per bit for URLLC services can be expressed as Eq. (27).

$$E_n^u = \frac{o_{n,m,0}^u E_n^{u,\mathcal{L}} + \sum_{k\in\boldsymbol{K}}\sum_{m\in\boldsymbol{M}} x_{n,m} o_{n,m,k}^u p_{n,m,k}^u T_n^{u,\mathcal{T}} T_s}{b_n^u} \text{ (J/bit)} \tag{27}$$

Since the object of this paper is to minimize the energy consumption of battery-operated UEs, the energy consumption of the MEC server is not considered. It is assumed that the MEC servers are connected to the gNBs and receive energy through a wired electrical power line [36].

4.2. eMBB service analysis

eMBB packets can be partially processed by the UE locally and partially processed by the MEC server through partial offloading (i.e., $o_{n,m,0}^e$, $o_{n,m,k}^e \in [0,1]$, $\forall k \in \boldsymbol{K}$ and Eq. (2)). To ensure the stability of the queueing system for eMBB services, the following utilization constraints need to be satisfied

$$f_n^e \ge o_{n,m,0}^e \lambda_n^e \overline{c_n^e} \tag{28}$$

$$\overline{r_{n,m,k}^e} \ge o_{n,m,k}^e \lambda_n^e \overline{b_n^e} \tag{29}$$

$$\rho_m = \frac{\sum_{n\in\boldsymbol{N}^e}\left(1-o^e_{n,m,0}\right)\lambda^e_n\overline{c^e_n}}{f_m} \le 1 \tag{30}$$

where $\overline{r^e_{n,m,k}}$, $\overline{b^e_n}$, and $\overline{c^e_n}$ are the average transmission rate, average number of bits, and average number of CPU cycles required, respectively. Inequality Eq. (28) represents the stability of the local server queue, Eq. (29) represents the stability of the communication queue, and Eq. (30) represents the stability of the PS server. In the following sections, latency and energy consumption analysis, and QoS constraint models for each case are depicted.

4.2.1. Local processing

When an eMBB packet is partially processed locally in the UE, the processing delay of UE n and energy consumption per packet depending on the partial offloading rate can be represented as follows.

$$T^{e,\mathcal{L}}_{n,\mathcal{P}} = \left\lceil \frac{o^e_{n,m,0}c^e_n}{f^e_n T_s} \right\rceil \text{ (slots)} \tag{31}$$

$$E^{e,\mathcal{L}}_n = e^{e,\mathcal{L}}_n o^e_{n,m,0}c^e_n = \kappa\left(f^e_n\right)^2 o^e_{n,m,0}c^e_n \text{ (J/packet)} \tag{32}$$

4.2.2. MEC processing

Partial offloading through other RATs is carried out in parallel, and the overall transmission delay is determined by the delay of the longest RAT transmission, so when UE n transmits a task to the MEC server m through RAT k, the transmission delay $T^{e,\mathcal{T}}_{n,m,k}$ and total multi-RAT transmission delay $T^{e,\mathcal{T}}_n$ can be calculated as follows.

$$T^{e,\mathcal{T}}_{n,m,k} = \left\lceil \frac{o^e_{n,m,k}b^e_n}{r^e_{n,m,k}T_s} \right\rceil \text{ (slots)} \tag{33}$$

$$T^{e,\mathcal{T}}_n = \max_{k\in\boldsymbol{K}} T^{e,\mathcal{T}}_{n,m,k} = \max_{k\in\boldsymbol{K}} \left\lceil \frac{o^e_{n,m,k}b^e_n}{r^e_{n,m,k}T_s} \right\rceil \text{ (slots)} \tag{34}$$

When packets offloaded through each RAT k are processed by the MEC server, the processing delay of MEC server m can be represented as in Eq. (35).

$$T^{e,\mathcal{M}}_{n,\mathcal{P}} = \sum_{k\in\boldsymbol{K}} \left\lceil \frac{o^e_{n,m,k}c^e_n}{\left(f_m/i\right)T_s} \right\rceil \text{ (slots)} \tag{35}$$

The total latency of the MEC server processing can be obtained from the sum of the transmission delay and processing delay in the MEC server, which is expressed as Eq. (36).

$$T^{e,\mathcal{M}}_n = T^{e,\mathcal{T}}_n + T^{e,\mathcal{M}}_{n,\mathcal{P}} \tag{36}$$

The total latency for eMBB service is determined by the larger value of the delay when processed locally in the UE and the delay when processed by the MEC server, so it can be expressed as Eq. (37).

$$T^e_n = \max\left\{T^{e,\mathcal{L}}_{n,\mathcal{P}}, T^{e,\mathcal{M}}_n\right\} \tag{37}$$

The QoS constraint for delay of eMBB service is expressed as Eq. (38).

$$T^e_n \le T^{e,max}_n \tag{38}$$

The total energy consumption for eMBB service can be formulated from the local partial processing energy and partial transmission energy at the UE. Considering the offloading rate and user association, the expected normalized energy consumption per bit for eMBB services can be expressed as Eq. (39).

$$E^e_n = \frac{E^{e,\mathcal{L}}_n + \sum_{k\in\boldsymbol{K}}\sum_{m\in\boldsymbol{M}} x_{n,m}p^e_{n,m,k}T^{e,\mathcal{T}}_{n,m,k}T_s}{b^e_n} \text{ (J/bit)} \tag{39}$$

5. Problem formulation and deep reinforcement learning framework

In this section, the joint optimization problem of user association, task offloading, as well as the power and resource allocation is formulated. A deep reinforcement learning algorithm is proposed to solve this problem. The details of the problem formulation and optimization algorithms are described as follows.

5.1. Problem formulation

The energy consumption of each UE varies depending on the transmission power, channel resources, user association, offloading rate, and the CPU frequency. Therefore, to minimize the normalized energy consumption of each UE, the optimization problem can be formulated as follows.

$$\min_{p^\xi_{n,m,k}, N^\xi_{n,m,k}, x_{n,m}, o^\xi_{n,m,k}, f^\xi_n} E^\xi_n$$

$$\begin{aligned}
\text{s.t.}\quad & C1: o^u_{n,m,0}, o^u_{n,m,k} \in \{0,1\} \\
& C2: o^e_{n,m,0}, o^e_{n,m,k} \in [0,1] \\
& C3: o^\xi_{n,m,0} + \sum_{k\in\boldsymbol{K}} o^\xi_{n,m,k} = 1 \\
& C4: \sum_{m\in\boldsymbol{M}}\sum_{n\in\boldsymbol{N}^u} x_{n,m}N^u_{n,m,k} + \sum_{m\in\boldsymbol{M}}\sum_{n\in\boldsymbol{N}^e} x_{n,m}N^e_{n,m,k} \le N^{max}_k \\
& C5: \sum_{m\in\boldsymbol{M}} x_{n,m} \le 1 \\
& C6: f^\xi_n \le f^{max}_n \\
& C7: p^\xi_{n,m,k} \le p^{max}_{n,m,k} \\
& C8: \varepsilon^{u,\mathcal{L}}_n \le \varepsilon^{u,max}_n \\
& C9: \varepsilon^u_n \le \varepsilon^{u,max}_n \\
& C10: \varepsilon^{u,\mathcal{M}}_n \le 0.5\varepsilon^{u,max}_n, \quad \varepsilon^{u,\mathcal{D}}_n \le 0.5\varepsilon^{u,max}_n \\
& C11: T^{u,\mathcal{L}}_n \le T^{u,max}_n \\
& C12: T^{u,\mathcal{M}}_n \le T^{u,max}_n \\
& C13: T^e_n \le T^{e,max}_n \\
& C14: f^e_n \ge o^e_{n,m,0}\lambda^e_n\overline{c^e_n} \\
& C15: \overline{r^e_{n,m,k}} \ge o^e_{n,m,k}\lambda^e_n\overline{b^e_n} \\
& C16: \rho_m \le \begin{cases} 1, & \text{if } o^u_{n,m,0}=1,\ \forall n\in\boldsymbol{N}^u \\ \rho_{th}, & \text{otherwise} \end{cases}
\end{aligned} \tag{40}$$

In Eq. (40), $m \in \boldsymbol{M} = \{1,\dots,M\}$, $k \in \boldsymbol{K} = \{1,\dots,K\}$, and C1-C3 denote the offloading policy constraints. Constraint C4 ensures that the sum of the number of subcarriers allocated to the users associated to each specific gNB must be less than or equal to the maximum number of subcarriers N^{max}_k of that gNB. Constraint C5 denotes the user association constraint. C6 and C7 are constraints of maximum CPU frequency and transmission power. C8-C10 denote the QoS constraints for reliability. C11-C13 denote the QoS constraints for latency. C14-C16 are the utilization constraints to ensure the stability of the queueing system.

Lemma 2. *The optimization problem Eq. (40) is an NP-hard problem.* ■

Proof. Consider a special case in which only one eMBB UE performs offloading using only RAT 1 (i.e., $n = 1$ and $k = 1$), and the offloading ratio, CPU frequency, transmission power, and channel bandwidth are fixed and equal to o, f, p, N, respectively. In this case, problem Eq. (40) is equivalent to

$$\max \sum_{m\in\boldsymbol{M}} (-opT_m)x_m$$
$$\text{s.t.} \sum_{m\in\boldsymbol{M}} N_m x_m \le N^{max} \text{ and } x_m \in \{0,1\}$$

which is a 0-1 knapsack problem that determines which MEC server to associate with, in order to minimize the energy consumption without exceeding the total amount of bandwidth resources. The 0-1 knapsack problem is a well-known and proven NP-hard problem [37]. Since the special case is NP-hard, problem Eq. (40) is also NP-hard [38]. ■

5.2. Deep reinforcement learning-based solution

According to Lemma 2, the optimization problem Eq. (40) is an NP-hard problem. Because NP-hard problems have high complexity, they are difficult to solve with traditional solutions and usually solved by dividing them into multi-level optimization subproblems. However, deep reinforcement learning can be an effective solution to solve this problem without dividing Eq. (40) into subproblems. Moreover, deep reinforcement learning can overcome the challenges posed by the large number of constraints in the formulated optimization problem Eq. (40). By embedding the constraint structure into the state, action, and reward spaces, the agent learns feasible and optimal solutions without explicit enumeration of all constraints, thereby significantly reducing the computational complexity compared to traditional solvers. In this section, a joint optimization algorithm of user association, task offloading, as well as power and resource allocation based on deep reinforcement learning is proposed. The goal of the deep reinforcement learning system is to maximize the long-term rewards by optimizing offloading decisions and the resource allocation for the UEs. Each agent corresponds to a UE device and interacts with the environment based on a policy that specifies what actions the agent will perform in each state. Multi-agent DRL is utilized to reduce the instability and complexity in the multi-UE environment. Details of the state, action, reward, training, and execution strategy of the proposed multi-agent DRL model are described in the following sections.

5.2.1. State space

The state space $\mathcal{S}$ includes a set of task information $\mathbb{I}^{\xi}$ and a Channel State Information (CSI) matrix $\mathbb{H}$. The state $s_n(t) \in \mathcal{S}$ is defined to describe the environment state for the task and channel information of UE n at time slot t, which is presented as

$$\begin{aligned} s_n(t) &= \left\{ \mathbb{I}_n^{\xi}(t), \mathbb{H}_n^{\xi}(t) \right\} \\ &= \left\{ \left(\lambda_n^{\xi}(t), b_n^{\xi}(t), c_n^{\xi}(t), \mathbb{Q}_n^{\xi}(t) \right), \mathbb{H}_n^{\xi}(t) \right\} \end{aligned} \tag{41}$$

where $\mathbb{H}$ represents a matrix of $1 \times K$ dimension containing the CSI information of each RAT, and task information $\mathbb{I}$ consists of the average task arrival rate, size of task, required CPU cycles for processing, and QoS constraints. For the stability of learning, state values are normalized to the same order of magnitude.

5.2.2. Action space

The action space $\mathcal{A}$ is determined by the user association, CPU resource allocation, multi-RAT offloading rate decision, transmission power allocation, and resource block allocation. At time slot t, UE n takes an action $a_n(t)$ according to the current state $s_n(t)$ based on the decision policy π_n. Action $a_n(t)$ is presented in Eq. (42).

$$a_n(t) = \left\{ x_{n,m}(t), f_n^{\xi}(t), o_{n,m,k}^{\xi}(t), p_{n,m,k}^{\xi}(t), N_{n,m,k}^{\xi}(t) \right\} \tag{42}$$

The range of values for each action is $[-1, 1]$.

5.2.3. Reward

According to state $s_n(t)$, UE n takes an action $a_n(t)$ and obtains an immediate reward $r_n(t)$, which is related to the amount of energy consumption. The objective of this problem is to minimize the energy consumption of each UE. Therefore, the reward is set negatively to the energy consumption and can be expressed as Eq. (43).

$$r_n(t) = -E_n^{\xi} \tag{43}$$

Each UE attempts to maximize its expected discounted reward through its interactions with the environment. The action-value function $Q_n(s_n, a_n)$ based on the Bellman equation in the deterministic target policy can be expressed as

$$Q_n(s_n, a_n) = \mathbb{E}\left[r_n + \gamma Q_n(s_n', a_n') \right] \tag{44}$$

where γ is the discount factor, s_n and a_n respectively represent the current state and action, and s_n' and a_n' represent the next state and action in the Markov Decision Process (MDP) at the next time slot.

5.3. Proposed multi-agent DRL framework

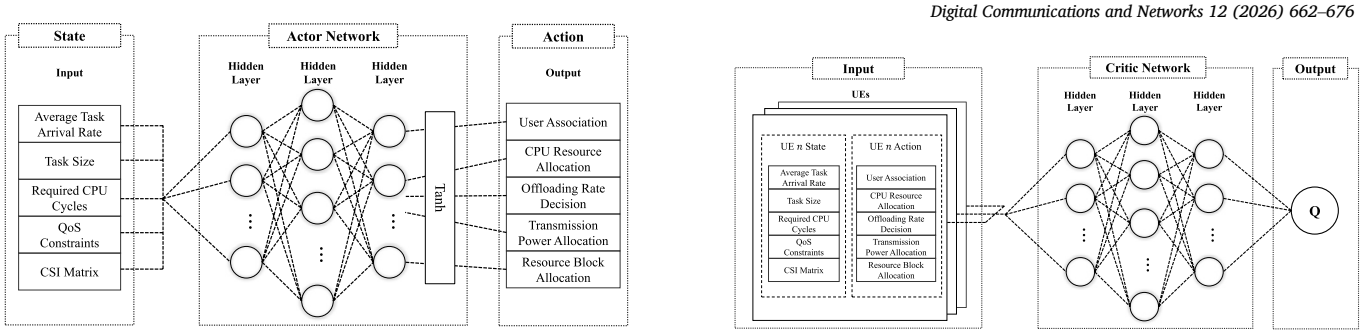
Assume that each UE has an intelligent agent module to perform reinforcement learning. Each agent module has two parts: 1) actor network and 2) critic network. The actor network is used to generate the agent's action. The actor network consists of the main actor network μ parameterized by θ and the target actor network μ', which is parameterized by θ'. The actor network takes the state as input and outputs the action through multiple fully connected layers. The critic network is used to criticize the actor network's actions. The critic network consists of the main critic network Q parameterized by ω and target actor network Q' parameterized by ω'. The critic network takes the state and action as inputs, and will output the Q value through multiple fully connected layers.

Fig. 3(a) shows the actor network structure in J-RATOA. The main network and the target network have different parameters but the same network architecture. For UE n, the actor network takes the current observed state s_n and outputs action a_n through its multiple fully connected hidden layers using the tanh function. The critic network structure of J-RATOA is presented in Fig. 3(b). For UE n, the critic network takes joint states $\boldsymbol{s} = (s_1, \ldots, s_N)$ and joint actions $\boldsymbol{a} = (a_1, \ldots, a_N)$ of the agents in the system as inputs, and outputs the Q value of UE n. These procedures are applied independently to all UEs.

Fig. 4 shows an overview of the training strategy applied in the J-RATOA system. In the training step, each agent n receives the state s_n, takes the action a_n decided through the main actor network, and receives the reward r_n accordingly. Joint state $\boldsymbol{s}$, joint action $\boldsymbol{a}$, joint reward $\boldsymbol{r}$, and joint next state $\boldsymbol{s}'$ are all stored in the experience replay buffer R in the form of a transition set $(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{s}')$. Because the training process requires enormous computational power and a long time, the training takes place in the main learner. The proposed network is trained in three stages of network updates: 1) main critic network update, 2) main actor network update, and 3) target networks update. After the training process is completed and in the distributed execution process, the networks' learning parameters are transferred from the main learner to the agents, and each agent can perform optimal actions (which are user association, CPU resource allocation, offloading rate decision, transmission power allocation, and resource block allocation) according to the state through the main actor network. While the training phase requires significant computational resources and is performed on a centralized learner, the proposed J-RATOA framework supports real-time inference during the execution phase. Each agent independently executes the actor network to compute offloading, resource allocation, and association decisions based on local state observations. Given the compact structure of the actor network and the low-dimensional input space, the inference process incurs low computational overhead. This design makes the framework practically feasible for latency-sensitive 5G systems.

5.3.1. Main critic network update

Each agent n derives the Q value through the main critic network as feedback to evaluate the actions of all agents decided through the



(a) actor network **(b)** critic network

Fig. 3. Structure of the J-RATOA actor network and critic network.

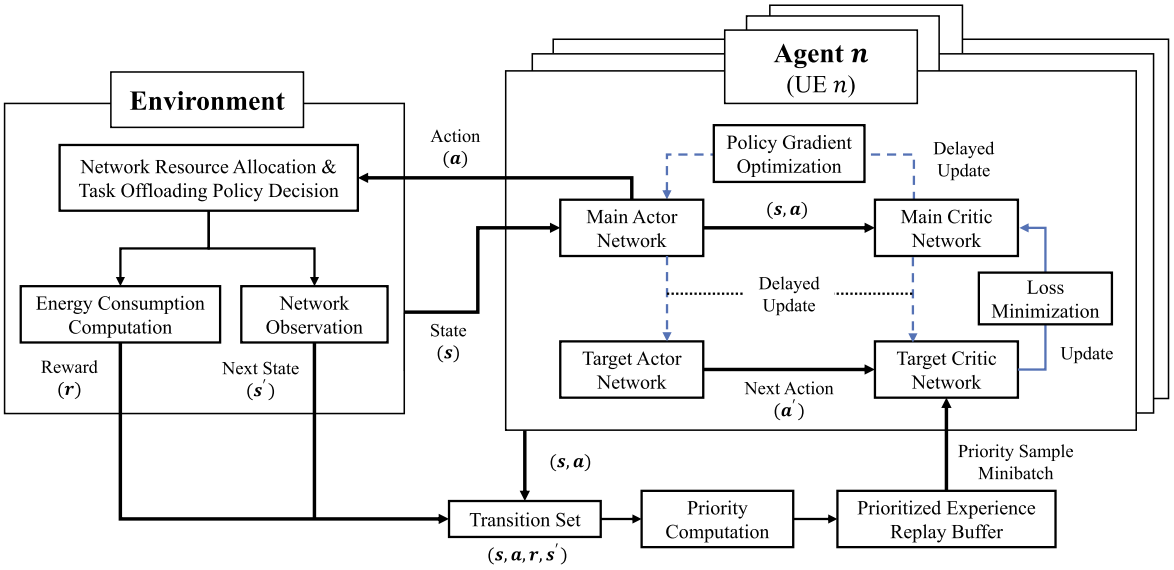


Fig. 4. Overview of the training strategy of the J-RATOA scheme.

actor network with states. The main critic network is updated through minimizing the loss function $\mathcal{L}(\omega_n)$ defined as

$$\mathcal{L}(\omega_n) = \mathbb{E}_{\boldsymbol{s},\boldsymbol{a},\boldsymbol{r},\boldsymbol{s}'}\left[\left(Q_n(\boldsymbol{s}, a_1, \ldots, a_N) - y_n\right)^2\right] \tag{45}$$

where $y_n = r_n + \gamma Q_n'(\boldsymbol{s}', a_1', \ldots, a_N')|_{a_l' = \boldsymbol{\mu}'(s_l)}$ is the target Q value and $\boldsymbol{\mu}' = \{\boldsymbol{\mu}_{\theta_1'}', \ldots, \boldsymbol{\mu}_{\theta_N'}'\}$ is the set of target policies with delayed parameter θ_n'. The parameter ω_n is updated through the gradient descent method in the loss minimization process.

5.3.2. Main actor network update

The parameter of the main actor network θ_n can be updated according to the feedback of the main critic network through the deterministic policy gradient descent method for each agent n. The gradient of the expected return can be represented as

$$\nabla_{\theta_n} \boldsymbol{J} = \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}\sim R}\left[\nabla_{\theta_n}\mu_{\theta_n}(a_n|s_n)\nabla_{a_n}Q_n(\boldsymbol{s}, a_1, \ldots, a_N)|_{a_i=\boldsymbol{\mu}(s_l)}\right] \tag{46}$$

5.3.3. Target networks update

The parameter of the target actor network θ_n' and the parameter of the target critic network ω_n' are updated for each agent n as follows.

$$\theta_n' = \tau\theta_n + (1-\tau)\theta_n' \tag{47}$$

$$\omega_n' = \tau\omega_n + (1-\tau)\omega_n' \tag{48}$$

where τ is the soft update rate for the target network.

The overall training process of the proposed J-RATOA scheme is presented in Algorithm 1. Each agent initializes its parameters for training the network (lines 1-4). At each time slot t, each agent computes its action in its state with policies based on a main actor network (lines 9-11). After each agent executes its actions, each agent obtains the reward and the next state (line 12). The states, actions, rewards, and the next states are stored in the replay buffer as a transition set with maximal priority (line 13). Each agent samples a minibatch based on the priority at every time slot t and updates the main critic networks and the priorities of the transition sets (lines 15-19). Finally, the policies and target network parameters are renewed every delayed update cycle (lines 20-26).

Lemma 3. *The overall learning complexity of J-RATOA can be expressed as* $O\left(N\left(\frac{\mathcal{E}_{num}}{D}|A|P_{actor} + \mathcal{E}_{num}N(|S|+|A|)P_{critic} + \log R\right)\right) \approx O(N^2)$. ■

Proof. The computational complexity of the proposed J-RATOA scheme in Algorithm 1 is determined by the number of agents, size of state space and action space, neural network structure, number of learning episodes, delayed update cycle, priority updating, and sampling. Learning of the proposed J-RATOA consists of two parts: actor network learning and critic network learning. The actor network is trained using policy gradient descent to maximize the expected value of the Q function and receives the state as input, and outputs actions. At this time, the actor network learning complexity of each agent is $O(|A|P_{actor})$,

Algorithm 1 J-RATOA Framework for Training.

Input: Number of UEs N, system parameters $\lambda_n, b_n, c_n, \mathbb{Q}_n$ for all UEs $\forall n$, training episodes $\mathcal{E}_{num}$, learning hyperparameters $\alpha, \gamma, \tau, D, R, \mathcal{N}, X$
Output: Trained main actor networks $\boldsymbol{\mu}_n$ for each UE n

Initialize the replay buffer R and the priority of the replay buffer
Initialize the main actor network μ with weights θ and the main critic network Q with weights ω
Initialize the target actor network μ' with weights θ' and the target critic network Q' with weights ω'
Initialize the delayed update cycle D
for episode $e = 1, 2, \ldots, \mathcal{E}_{num}$, **do**
  Receive the initial state $\boldsymbol{s}$
  Initialize a random process $\mathcal{N}$ for action exploration
  for time slot $t = 1, 2, \ldots, T$, **do**
    for UE $n = 1, 2, \ldots, N$, **do**
      Compute action $a_n = \boldsymbol{\mu}_{\theta_n}(s_n) + \mathcal{N}$
    end for
    Execute action $\boldsymbol{a} = (a_1, \ldots, a_N)$ and observe $\boldsymbol{r}$ and next state $\boldsymbol{s}'$
    Store $(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{s}')$ in the replay buffer R with maximal priority
    Set the state $\boldsymbol{s} \leftarrow \boldsymbol{s}'$
    for UE $n = 1, 2, \ldots, N$, **do**
      Sample a minibatch of X samples $(\boldsymbol{s}^j, \boldsymbol{a}^j, \boldsymbol{r}^j, \boldsymbol{s}'^j)$ from the replay buffer R based on sampling probability $\mathbf{P}_n^j = \left(|\mathbb{P}_n^j|\right)^\alpha / \sum_{x \in X} \left(|\mathbb{P}_n^x|\right)^\alpha$
      Set $y_n^j = r_n^j + \gamma Q_n^{\mu'}(\boldsymbol{s}'^j, a_1', \ldots, a_N')|_{a_k' = \mu_k'(s_k^j)}$
      Update main critic network by minimizing the loss: $\mathcal{L}(\omega_n) = \frac{1}{X} \sum_{j \in X} (Q_n^{\boldsymbol{\mu}}(\boldsymbol{s}^j, a_1^j, \ldots, a_N^j) - y_n^j)^2$
      Update priority: $\mathbb{P}_n^j = r_n^j + \gamma Q_n^{j+1}(\boldsymbol{s}^{j+1}, a_1^{j+1}, \ldots, a_N^{j+1}) - Q_n(\boldsymbol{s}^j, a_1^j, \ldots, a_N^j)$
      if $t \mod D = 0$, **then**
        Update main actor network by using sampled policy gradient: $\nabla_{\theta_n} \boldsymbol{J} = \frac{1}{X} \sum_{j \in X} \nabla_{\theta_n} \boldsymbol{\mu}_n(a_n | s_n) \nabla_{a_n} Q_n^{\boldsymbol{\mu}}(\boldsymbol{s}^j, a_1^j, \ldots, a_N^j)$
      end if
    end for
    if $t \mod D = 0$, **then**
      Update the target network parameters for each UE n: $\theta_n' = \tau\theta_n + (1-\tau)\theta_n'$, $\omega_n' = \tau\omega_n + (1-\tau)\omega_n'$
    end if
  end for
end for

and the number of actor network parameters P_{actor} is determined by the depth and width of the actor network, that is, the number of layers and the number of nodes per layer. Due to delayed policy updates, learning is performed for $\mathcal{E}_{num}/D$ episodes rather than the total number of episodes $\mathcal{E}_{num}$, so the learning complexity of the entire actor network becomes $O(\frac{\mathcal{E}_{num}}{D}|A|P_{actor})$. In addition, the critic network receives the state and action of all agents as input, calculates the Q value, and is trained to minimize the loss function Eq. (45) based on the Bellman equation. At this time, the complexity of learning the critic network for each agent is $O(N(|S| + |A|)P_{critic})$, and the number of critic network parameters, P_{critic}, is determined by the depth and width of the critic network, that is, the number of layers and the number of nodes per layer. Since learning is performed for the entire number of episodes $\mathcal{E}_{num}$, the learning complexity of the entire critic network becomes $O(\mathcal{E}_{num}N(|S| + |A|)P_{critic})$. If the size of the replay buffer is given as R, the complexity of priority updating and sampling is given as $O(\log R)$ [40]. Since learning of each network is performed for all agents, and the number of episodes, delayed update cycle, size of action space and state space, number of layers of each network, number of nodes per layer, and size of the replay buffer are all constants, the overall learning complexity of J-RATOA can be expressed as $O\left(N\left(\frac{\mathcal{E}_{num}}{D}|A|P_{actor} + \mathcal{E}_{num}N(|S| + |A|)P_{critic} + \log R\right)\right) \approx O(N^2)$. ■

5.4. Improvements

In the proposed multi-agent DRL architecture, further improvements of the MADDPG system were applied as follows.

5.4.1. Delayed target network and policy updates

The target network is used in maintaining the stability in the deep reinforcement learning process [39]. Since the deep function approximations require multiple gradient updates to converge, the target network provides a stable objective for the learning procedure. In traditional MADDPG, the target network and policy are updated every time the main networks are updated. However, in this case, the interaction between the actor updates and the critic updates may cause a failure. The residual Temporal Difference (TD) error $\delta_n(s_n, a_n)$ is due to the errors that may result from the neural network approximation in reinforcement learning, and the Q value becomes $Q_n(s_n, a_n) = \mathbb{E}\left[r_n + \gamma Q_n(s_n', a_n')\right] - \delta_n(s_n, a_n)$. With the error accumulated at each update, the Q value approximates the expected return minus the expected discounted sum of future TD-errors and becomes $Q_n(s_n(t), a_n(t)) = r_n(t) + \gamma\mathbb{E}[Q_n(s_n(t+1), a_n(t+1))] - \delta_n(s_n(t), a_n(t)) = r_n(t) + \gamma\mathbb{E}[r_n(t+1) + \gamma\mathbb{E}[Q_n(s_n(t+2), a_n(t+2)) - \delta_n(s_n(t+1), a_n(t+1))] - \delta_n(s_n(t), a_n(t)) \approx \mathbb{E}\left[\sum_{i=t}^{T} \gamma^{i-t}(r_n(i) - \delta_n(i))\right]$, so the variance of the Q value is proportional to the variance of the future reward and estimation error. If the policy is poor, the Q value estimate diverges due to an overestimation, and if the Q value estimate itself is inaccurate, the policy also becomes poor. Therefore, in the proposed method, the policy network update cycle is changed to increase the stability of learning by reducing the variance of the Q value estimation. If the target networks can be used to reduce the errors through multiple updates, and policy updates on the high-error states will result in divergent behaviors, then the policy network needs to be first updated at a lower frequency than the main network to minimize errors before introducing a policy update.

To further explain the impact of the delayed target network updates on Q value stability, consider that the Q value at each step incorporates the expected future rewards minus accumulated TD-errors. When the target network is updated at every step, it rapidly adapts to the noise in the parameters of the main network, causing large variations in TD-targets and leading to higher variance in Q value estimates. In contrast, delaying the target network update through a soft update mechanism $\theta' = \tau\theta + (1-\tau)\theta'$ smooths out the rapid fluctuations of the main network by averaging the updates over multiple steps. This process significantly reduces the variance of the TD-targets because the target values are less sensitive to temporary noise in the main network. As a result, the critic receives a more stable learning signal, which in turn produces more accurate Q value estimates. This variance reduction ultimately stabilizes the actor updates, preventing poor policy improvements based on inaccurate Q value estimates, and thus improves the overall stability of the training.

5.4.2. Prioritized experience replay

In RL, for each time slot, each agent calculates its actions in its state based on a policy. After each agent performs its action, each agent gets a reward and the next state. At this point, the state, action, reward, and next state are stored in the experience replay buffer as a transition set. In the existing MADDPG, transition sets are randomly sampled. However, sampling better-valued experiences more frequently can have a learning advantage [40]. From the loss function form in Eq. (45), the TD-error of transition $(\mathbb{P}_n^j)$ is computed by $\mathbb{P}_n^j = r_n^j + \gamma Q_n^{j+1}(\boldsymbol{s}^{j+1}, a_1^{j+1}, \ldots, a_N^{j+1})|_{a_l^{j+1} = \boldsymbol{\mu}^{j+1}\left(s_l^j\right)} - Q_n(\boldsymbol{s}^j, a_1^j, \ldots, a_N^j)$. The transition sampling probability based on proportional prioritization is defined as $\mathbf{P}_n^j = \left(|\mathbb{P}_n^j|\right)^\alpha / \sum_{x \in X} \left(|\mathbb{P}_n^x|\right)^\alpha$, where α is the prioritization coefficient which determines how much prioritization is used, with $\alpha = 0$ corresponding to the uniform random sampling case and $\alpha = 1$ corresponding to the greedy prioritization case. The priority-based sampling

Table 4
Simulation Parameters.

Symbol	Parameters	Value
$T_n^{u,max}$	Latency requirement of URLLC service	1 ms [41]
$T_n^{e,max}$	Latency requirement of eMBB service	1 s [42]
$\varepsilon_n^{u,max}$	Reliability requirement of URLLC service	10^{-6} [41]
b_n^u	Task size of URLLC service	32 bytes [30]
b_n^e	Task size of eMBB service	[50, 100] KB [42]
λ^u	Arrival rate of URLLC tasks	500 tasks/sec [14]
λ^e	Arrival rate of eMBB tasks	5 tasks/sec [14]
a_n	Number of required CPU cycles per byte	[300, 400] cycles/byte
N_0	Noise spectral density	-174 dBm/Hz
W	Bandwidth of each resource block n	180 kHz
T_s	Duration of one time slot	0.125 ms
$p_{n,m,k}^s$	Transmission power of UE n	[1, 23] dBm
f_n	Computation resource of UE n	[0.1, 2] GHz
f_m	Computation resource of MEC server m	5 GHz
M	Total number of MEC servers	4
N_1^{max}	Maximum number of RBs in RAT 1	50
N_2^{max}	Maximum number of RBs in RAT 2	200
α	Prioritization coefficient	0.8

Table 5
Learning Parameters.

Parameters	Value
Number of hidden layers in actor network	2
Number of hidden layers in critic network	2
Number of neurons in hidden layer of actor network	64
Number of neurons in hidden layer of critic network	64
Optimizer	Adam
Learning rate	0.001
Discount rate	0.95
Soft update rate	0.01
Replay buffer size	10^7
Batch size	100
Total number of training episodes	20000

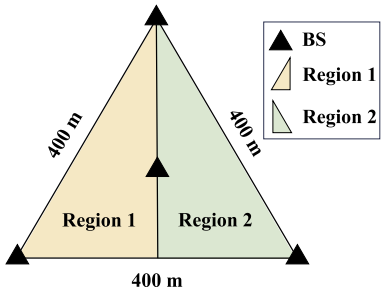


Fig. 5. Network topology used in the simulation.

method has advantages in terms of the convergence speed of learning over random sampling used in the conventional MADDPG method.

6. Performance analysis

This section describes the simulation results and the analysis of the proposed J-RATOA scheme, which is compared to the mDQNR shceme of [18], the multi-MEC scheme of [20], and the I-PDQN scheme of [21]. The purpose of the experimental setup is to validate the effectiveness and practicality of the proposed J-RATOA framework in realistic MEC environments. The scenarios are designed to reflect heterogeneous services (eMBB and URLLC) and multi-RAT conditions commonly observed in 5G networks. The experiments specifically aim to (i) assess the reduction in energy consumption of J-RATOA compared to the existing schemes, (ii) evaluate the learning stability improvement enabled by PER and DU methods, and (iii) demonstrate the advantage of joint optimization across communication and computation resources with user association. All system parameters are selected to align with values

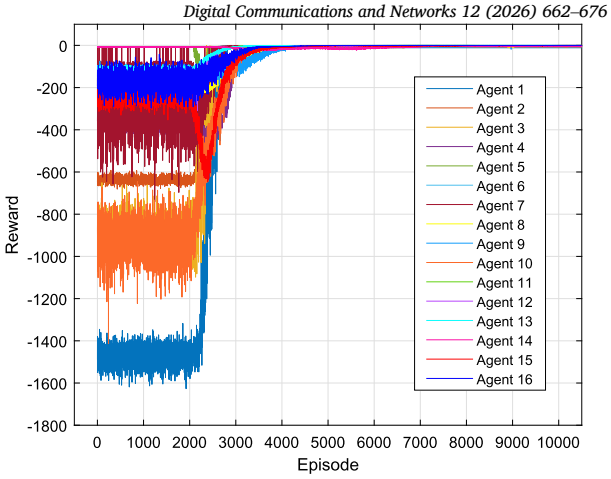


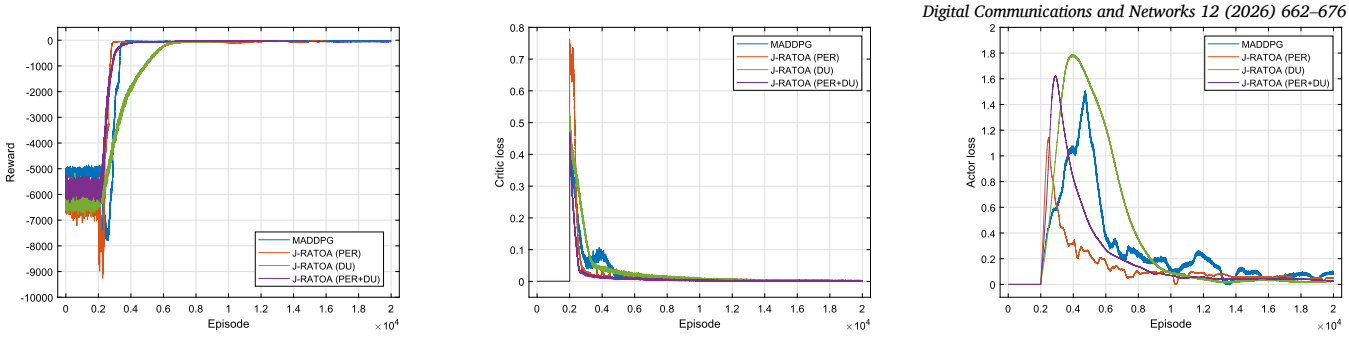
Fig. 6. Reward of each agent on based on the training episodes.

suggested in the 3GPP specifications and recent studies to ensure the applicability of the results. The simulations were implemented in PyTorch using the DRL functionality. The system parameters used in the simulations are summarized in Table 4. The performance was analyzed in terms of the convergence, normalized energy consumption, and latency, based on the system environment of the heavy workload scenario [14]. The network topology to be used in the simulation is shown in Fig. 5. Basically, the UE distribution in region 1 and region 2 is the same ratio (i.e., 5:5), but additionally, it is confirmed how robust the proposed and benchmark schemes are in optimizing the normalized energy consumption of the UEs as the UE distribution ratio changes. The path loss model is given by $35.3 + 37.6 \log_{10} d$, where d is the distance between the BS and UE in meters. The shadowing model follows a lognormal distribution with a standard deviation of 8 dB. The simulation was based on 8 URLLC and eMBB service UEs each. The reliability and latency requirement of URLLC service is set to 10^{-6} and 1 ms, respectively, according to AR/VR and factory automation system requirements [41]. The latency requirement of eMBB service is set to 1 s and the reliability of eMBB service is not considered, according to the video surveillance and smart factory system requirements [42]. According to the maximum transmission bandwidth configuration in [43], the LTE-based RAT 1 is set to a maximum of 50 RBs, and the 5G midband-based RAT 2 is set to a maximum of 200 RBs.

The learning parameters are shown in Table 5. The actor network and critic network both have two hidden layers, with the hidden layers having 64 neurons. The Adam optimizer with a learning rate 0.001 is used to update the network. The discount rate γ is set to 0.95 and the soft update rate τ is set to 0.01. The prioritization coefficient α is set to 0.8. The size of the replay buffer is 10^7 and the batch size is 100. Training is conducted for a total of 20000 episodes, and the networks are not learned until 2000 episodes and proceed in a random exploration phase to fill the replay buffer. Therefore, the loss values appear after 2000 episodes.

6.1. Convergence of the proposed J-RATOA algorithm

To analyze the optimality of the proposed J-RATOA scheme, the convergence of learning needs to be verified. The J-RATOA scheme is designed so that each UE agent optimizes its own action while taking into account the state and action of other agents using a centralized critic network. The proposed J-RATOA scheme is based on Q-learning, and the convergence of learning has been proven in [44]. According to the convergence theorem of single-agent Q-learning, the Q function can converge to the optimal value given the bounded reward and learning rate under the MDP. The reward function in the proposed system is bounded because the reward is the negative value of energy consumption and has



(a) reward **(b)** critic loss **(c)** actor loss

Fig. 7. Reward, critic loss, and actor loss variation trends based on the training episodes.

a maximum value of 0. In the proposed environment, the task information and CSI information of each UE agent become the state, and each agent takes actions in these states to perform user association, resource allocation, and task offloading. At this time, the UE's energy consumption is immediately calculated as a reward, the state of the environment changes according to the newly implemented policies, and new actions and rewards are recalculated accordingly. The continuous process of each of these agents creates a sequence of state, action, reward, and next state for each agent, which all satisfy the MDP. In a situation where the policy is fixed, the Q function receives the action and state information of other agents as input and learns the optimal state-action value function through iterative updates of the Bellman equation. If the policy of each agent is fixed, the centralized critic of J-RATOA could be expected to converge by reliably learning the Q value for each state-action pair.

Additionally, each UE agent continuously updates its policy through the policy gradient descent in Eq. (46). During this process, if updates to the policy gradually improve or maintain the expected value of the Q function, the policy has monotonicity and the probability of convergence increases. In other words, if each agent's policy continues to reduce the agent's energy consumption over time, or at least maintains it at the current level, as shown in Fig. 6, the policy will follow a stable convergence path. A multi-agent system can reach a Nash Equilibrium through the interaction of each agent [45]. Because the J-RATOA is designed so that each agent optimizes its policy while considering the state and action of other agents through a centralized critic, the policy of all UEs reaches a Nash equilibrium, and the system can converge to an optimal state of energy consumption. In addition, delayed target network and policy updates reduce the variance of the Q value estimation, and the PER reduces the learning step to converge to the optimal value of the Q function, which does not violate the convergence theory because it does not deviate from the MDP.

In order to show the convergence of the deep reinforcement learning algorithm, the proposed J-RATOA schemes' reward, training loss in the critical network (i.e., critic loss), and training loss in the actor network (i.e., actor loss) are presented in reference to the number of learning episodes in Fig. 7. In Fig. 7, "PER" and "DU" refer to the improved points of the proposed J-RATOA scheme, Prioritized Experience Replay (PER) and Delayed Updates (DU) of the target network and policies, respectively. MADDPG shows that the critic loss and actor loss values fluctuate as the training proceeds. J-RATOA applying the DU method (represented as "J-RATOA (DU)" in Fig. 7) helps to reduce this fluctuation and improve the stability of training, which results in a slower convergence speed than the existing J-RATOA as shown in (a), but the training loss is stably reduced as shown in (b) and (c). The PER method is applied to compensate for the slow convergence speed (represented as "J-RATOA (PER)" in Fig. 7), which shows a faster training convergence speed than the conventional MADDPG in each figure. Combining the PER and DU advantages (represented as "J-RATOA (PER+DU)" in Fig. 7) results in a similar training convergence speed to that of the conventional MADDPG, as seen in (a), but has the effect of improving the stability of the learning process, as can be observed in (b) and (c).

Table 6
Comparison of normalized RSD of critic loss and actor loss for each scheme.

Scheme	Critic loss (normalized RSD)	Actor loss (normalized RSD)
MADDPG	2.094	1.616
J-RATOA (PER)	2.098	1.682
J-RATOA (DU)	**1.483**	**1.123**
J-RATOA (PER+DU)	1.561	1.247

To further quantify the training stability, the normalized Residual Standard Deviation (RSD) of the critic loss and actor loss for each scheme is measured after 2000 episodes of random exploration. The normalized RSD fairly evaluates the stability regardless of the absolute loss magnitude by assessing the level of variation of the training loss with respect to the average loss magnitude. Given a residual sequence r_t defined as the difference between the actual loss value and the moving average at time step t, the normalized RSD is calculated as

$$\text{Normalized RSD} = \frac{\text{std}(r_t)}{\mathbb{E}[|r_t|]}$$

where $\text{std}(r_t)$ denotes the standard deviation of the residual sequence, and $\mathbb{E}[|r_t|$ represents its mean absolute value. The residual r_t captures how much the instantaneous loss deviates from the smoothed trend, and thus directly reflects the training instability.

The application of the delayed target network and policy updates significantly improves the training stability in comparison to the conventional MADDPG. As presented in Table 6, the J-RATOA (DU) scheme achieves a 29.18% reduction in the normalized RSD of critic loss and a 30.48% reduction in actor loss relative to the conventional MADDPG. This substantial decrease in residual fluctuations validates the effectiveness of DU in stabilizing the training process. Meanwhile, PER aims to accelerate the training convergence speed. However, PER slightly deteriorates the training stability, as it prioritizes sampling based on the magnitude of TD-errors, which can introduce bias and amplify variability in the sampled mini-batches. As a result, J-RATOA (PER) exhibits slightly higher normalized RSD values compared to MADDPG. By combining PER and DU, the proposed J-RATOA (PER+DU) successfully maintains a fast convergence speed due to the PER mechanism, while preserving a significantly improved training stability due to the DU mechanism. Although J-RATOA (PER+DU) shows slightly higher instability compared to DU alone due to the inclusion of PER, it still outperforms MADDPG by a considerable margin in both critic and actor loss fluctuations.

6.2. Normalized energy consumption

Fig. 8 shows the total normalized energy consumption simulation result for a different number of UEs. For all numbers of UEs, the multi-MEC scheme has a much worse energy consumption performance than the other schemes. The multi-MEC scheme distributes tasks by using the MECs around the UE together, but in situations where the available resources in the MEC are limited (i.e., as the number of UEs increases),

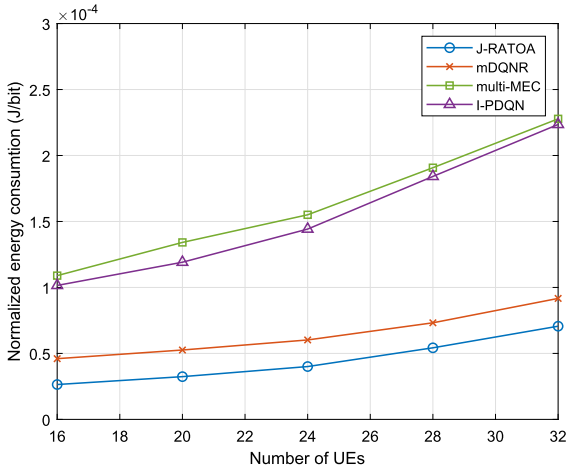


Fig. 8. Total normalized energy consumption based on the number of UEs.

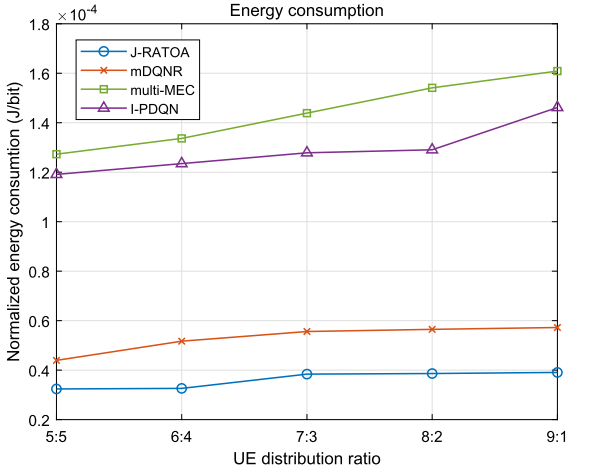


Fig. 9. Total normalized energy consumption based on the UE distribution ratio.

there is a limit to the inability to use the MEC resources optimally. Additionally, the multi-MEC scheme does not consider transmission power and user association optimization for energy minimization. The I-PDQN scheme, like the multi-MEC scheme, also has a poor energy consumption performance. The I-PDQN scheme adjusts CPU resources and user association, but only binary offloads tasks and does not control channel resources and transmission power. In addition, because the latency penalty was taken into consideration when setting the reward in reinforcement learning to minimize energy consumption, it cannot be considered as an optimized energy consumption result. In terms of normalized energy consumption, the mDQNR scheme is not as good as J-RATOA, but shows a better performance than the multi-MEC scheme and the I-PDQN shceme. The mDQNR scheme carries out bilevel optimization to minimize the UE's energy consumption in two stages. The first stage is task offloading by adjusting the transmission power and CPU resources with a Genetic Algorithm (GA) to minimize the energy consumption in the UE-side, and in the second step, resource allocation is performed by maximizing the MEC server's system utility through DQN. In addition, the mDQNR scheme also does not take into account user association optimization. Therefore, from the overall system perspective, optimization for energy minimization is lacking. Because it does not minimize the UE's energy consumption in the entire system through joint optimization of task offloading and resource allocation, the energy consumption performance is not as good as J-RATOA. The proposed J-RATOA scheme has the best energy consumption performance for all numbers of UEs. The J-RATOA scheme jointly optimizes the transmission power, channel resources, offloading rate, and CPU resources, as well as user association, unlike the other schemes that attempt to minimize the UE's energy consumption. These points indicate that the inability to adjust the UE's transmission power and user association in situations where network resources are limited can result in a dominant defect in terms of energy consumption when trying to support heterogeneous services.

The simulation results of the total normalized energy consumption according to the change of UE distribution ratio are shown in Fig. 9. In order to efficiently use network resources to reduce the energy consumption, it is necessary to properly associate UEs to MEC servers with appropriate resources according to the UE distribution. As the UE distribution changes from 5:5 (uniform) to 9:1 (i.e., as the number of UEs becomes more skewed to URLLC), the energy consumption increases due to changes in factors such as a decrease in the quality of the channel state and the resources of the MEC server available for each UE, and an increase in the distance between the UE and the gNB. Although the multi-MEC scheme can handle tasks using multiple MEC servers at the same time, it can be seen that the energy consumption increases rapidly as the UE distribution ratio is biased to one side (i.e., as there are fewer available network resources). In order to minimize the UE's energy consumption in a situation where the UE distribution is biased and resources are insufficient, it is necessary to adjust the transmission power and deliver the tasks to an appropriate MEC server according to the network situation, including the channel state. The I-PDQN scheme shows a somewhat stable overall energy consumption performance in terms of UE distribution ratio through appropriate association with the MEC based on the network situation and adjustments in CPU frequency. Nevertheless, in the I-PDQN scheme, which does not perform channel resources control, partial offloading, and transmission power control, its energy consumption can be seen to increase rapidly in the extreme UE distribution ratio (9:1) situation where resources are extremely insufficient. The mDQNR scheme, which considers transmission power adjustment and allocates channel resources by adjusting the resources of the MEC servers, is somewhat robust to changes in UE distribution ratios. The proposed J-RATOA scheme, which selects the optimal MEC to perform task offloading and resource allocation as the distribution of UEs changes in the network, shows an almost constant level of normalized energy consumption performance according to the change in the UE distribution ratio. These results indicate that simultaneous use of all MECs may not necessarily be good for resource-poor network situations, and above all, the modulation of the transmission power and the method of choosing the optimal MEC can dramatically reduce the energy consumption of the UE.

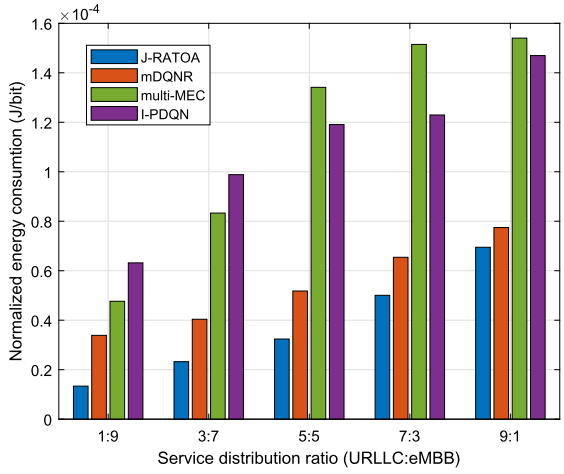


Fig. 10. Total normalized energy consumption based on the service distribution ratio.

Fig. 10 represents the normalized energy consumption according to the distribution of each service. Overall, as the proportion of URLLC services increases, the energy consumption per unit bit of the UEs increases

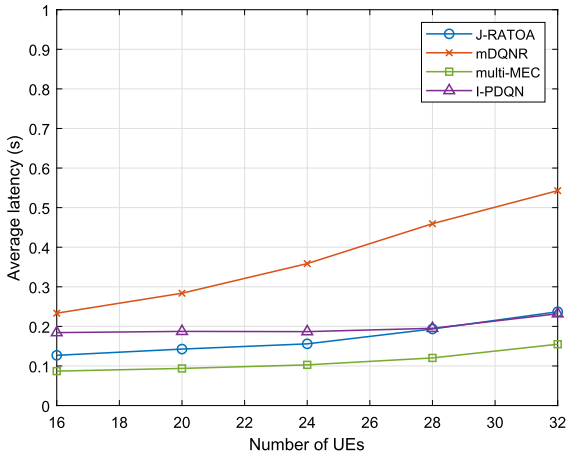


Fig. 11. Average eMBB service latency based on the number of UEs.

due to task offloading, resource allocation, and user association policies to satisfy strict QoS compared to eMBB services. The proposed J-RATOA scheme shows the best performance in terms of normalized energy consumption according to the service distribution ratio through appropriate partial offloading, channel resources allocation, transmission power and CPU frequency adjustment, and user association to minimize the energy consumption. Despite partial offloading, channel resources allocation, and transmission power and CPU frequency adjustment, mDQNR fails to perform appropriate user association according to the service distribution as the UE only communicates with the nearest gNB, resulting in a worse energy consumption performance than the J-RATOA scheme. The multi-MEC scheme and the I-PDQN scheme consume much more energy than the other two schemes as can be seen in Fig. 8. The presence or absence of a process to ensure the latency in the process of minimizing the energy consumption reveals the difference in normalized energy consumption between the two schemes according to the service distribution ratio. The I-PDQN scheme, which also guarantees latency reduction in the energy consumption minimization process, shows a better performance in terms of normalized energy consumption than the multi-MEC scheme as the ratio of URLLC services requiring strict latency QoS increases and becomes more than that of eMBB services.

6.3. Latency

Fig. 11 shows the average service latency of eMBB for a different number of UEs. The mDQNR scheme is the worst in terms of latency, and the higher the number of UEs, the faster the latency increases. Unlike other schemes that use central training and distributed execution, the mDQNR scheme uses reinforcement learning based on central training and central execution. Therefore, the network latency increases because the policy decided by a central learner must be delivered to each UE for execution, rather than conducting a decentralized policy decision in each UE. Additionally, since DQN is based on a discrete action space, as the number of UEs increases, it becomes difficult to find the optimal offloading ratio, transmission power, and resource allocation. In addition, the mDQNR scheme minimizes the energy consumption through a bilevel optimization of GA and DQN, so additional latency occurs compared to other schemes that perform single-level optimization. As the latency will increase rapidly as the number of UEs increases, the mDQNR scheme will experience a difficulty in satisfying the service QoS requirement (i.e., 1 second), which will have a significant impact on the energy consumption from then on. The I-PDQN scheme shows a slightly poor latency performance compared to J-RATOA and multi-MEC due to its inability to perform appropriate partial offloading and channel resources control. However, by adding the latency to the reward in reinforcement learning and applying compensation for the latency in the energy consumption optimization process, the latency tends to be the most consistent depending on the number of users. The multi-MEC scheme has the best performance in terms of latency. The multi-MEC scheme can use multiple adjacent MEC resources at once through a wired network, thus reducing the latency compared to the other schemes that only use wireless communication. The proposed J-RATOA scheme shows a higher latency than the multi-MEC scheme, but when compared to the multi-MEC scheme, even if the number of UEs increases, it can sufficiently satisfy the service QoS requirement of 1 second. In particular, considering the research to minimize the energy consumption of the UE, the proposed J-RATOA scheme is superior to the multi-MEC scheme considering the energy saving benefits that are obtained by sacrificing a little latency (which can be considered small compared to the gain in terms of energy), considering that this difference in latency performance is acceptable. These results indicate that a partially decentralized method may be better in terms of latency than a fully centralized method, and that even without using multi-MECs through a wired network, there is no problem in providing heterogenous services while satisfying the QoS through optimal adjustment of controllable network variables, such as, transmission power, channel resources, user association, offloading rate, and the CPU frequency.

7. Conclusion

In this paper, a multi-agent deep reinforcement learning-based joint resource allocation and task offloading cross-layer control technique is proposed to minimize the energy consumption of UEs that supports 5G URLLC and eMBB services while meeting stringent QoS requirements in 5G multi-RAT networks. The proposed J-RATOA scheme jointly optimizes the UE's transmission power and computing resources, association between the UE and nearby MECs, task offloading rate, and resource allocation of the communication channel to minimize the UE's energy consumption. Additionally, PER and delayed target network and policy updates are applied to increase the training efficiency and stability of the deep reinforcement learning process for the complex and high-dimensional optimization problem. The simulation results show that the proposed J-RATOA method can dramatically reduce the energy consumption compared to the benchmarked methods not only when UEs and the services are distributed evenly, but also when they are skewed, and the same is true depending on the service distribution. In particular, J-RATOA reduces the normalized energy consumption by up to 69% compared to existing optimization schemes, and improves training stability by up to 30% while maintaining the training speed by integrating PER and DU techniques. In future research, more effective AI-based offloading and resource allocation technologies are recommended for more advanced Beyond 5G (B5G) or 6G standards. The source code used in this paper is available at https://github.com/dongjunjun5/J-RATOA.

CRediT authorship contribution statement

Dongjun Jung: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Jong-Moon Chung:** Supervision, Project administration. **Hea-Sook Park:** Project administration, Funding acquisition.

Declaration of competing interest

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

Acknowledgement

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant funded by

the Republic of Korea Government (MSIT, Development of Trust Inter-Networking Technology of Defense Mobile Environment for Real-Time Information Sharing) under Grant RS-2022-II220030.

References

[1] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S.A. Ashraf, B. Almeroth, J. Voigt, I. Riedel, A. Puschmann, A. M.-Thiel, M. Muller, T. Elste, M. Windisch, Latency critical IoT applications in 5G: perspective on the design of radio interface and network architecture, IEEE Commun. Mag. 55 (2) (2017) 70–78.

[2] J.-M. Chung, Emerging Metaverse XR and Video Multimedia Technologies, Springer Nature, Apress, USA, 2023.

[3] A.M. Nor, O. Fratu, S. Halunga, Quality of service based radio resources scheduling for 5G eMBB use case, Symmetry 14 (10) (2022) 2193–2207.

[4] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, On multi-access edge computing: a survey of the emerging 5G network edge cloud architecture and orchestration, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1657–1681.

[5] K. Arulkumaran, M.P. Deisenroth, M. Brundage, A.A. Bharath, Deep reinforcement learning: a brief survey, IEEE Signal Process. Mag. 34 (6) (2017) 26–38.

[6] J.-M. Chung, Emerging Secure Networks, Blockchains & Smart Contract Technologies, Springer Nature, Springer, Switzerland, 2024.

[7] D. Jung, J. Kim, J.-M. Chung, Energy minimized computation offloading with popularity-based cooperation in 5G mMTC networks, IEEE Internet Things J. 9 (5) (2022) 3238–3250.

[8] T.Z.H. Ernest, A.S. Madhukumar, Computation offloading in MEC-enabled IoV networks: average energy efficiency analysis and learning-based maximization, IEEE Trans. Mob. Comput. 23 (5) (2024) 6074–6087.

[9] J. Lu, W. Feng, D. Pu, Resource allocation and offloading decisions of D2D collaborative UAV-assisted MEC systems, KSII Trans. Int. Inf. Syst. 18 (1) (2024) 211–232.

[10] X. Yang, X. Yu, H. Huang, H. Zhu, Energy efficiency based joint computation offloading and resource allocation in multi-access MEC systems, IEEE Access 7 (2019) 117054–117062.

[11] Y. Pan, C. Pan, K. Wang, H. Zhu, J. Wang, Cost minimization for cooperative computation framework in MEC networks, IEEE Trans. Wirel. Commun. 20 (6) (2021) 3670–3684.

[12] K. Cheng, Y. Teng, W. Sun, A. Liu, X. Wang, Energy-efficient joint offloading and wireless resource allocation strategy in multi-MEC server systems, in: Proceedings of the 2018 IEEE International Conference on Communications, IEEE, 2018, pp. 1–6.

[13] L. Tan, Z. Kuang, L. Zhao, A. Liu, Energy-efficient joint task offloading and resource allocation in OFDMA-based collaborative edge computing, IEEE Trans. Wirel. Commun. 21 (3) (2022) 1960–1972.

[14] R. Dong, C. She, W. Hardjawana, Y. Li, B. Vucetic, Deep learning for hybrid 5G services in mobile edge computing systems: learn from a digital twin, IEEE Trans. Wirel. Commun. 18 (10) (2019) 4692–4707.

[15] J. Bi, Z. Wang, H. Yuan, J. Zhang, M. Zhou, Cost-minimized computation offloading and user association in hybrid cloud and edge computing, IEEE Internet Things J. 11 (9) (2024) 16672–16683.

[16] S. Han, B. Lu, S. Lin, X. Hong, J. Shi, Learning-assisted energy minimization for MEC systems with noncompletely overlapping NOMA, IEEE Syst. J. 17 (4) (2023) 6126–6137.

[17] J. Gao, Z. Kuang, J. Gao, L. Zhao, Joint offloading scheduling and resource allocation in vehicular edge computing: a two layer solution, IEEE Trans. Veh. Technol. 72 (3) (2023) 3999–4009.

[18] J. Yun, Y. Goh, W. Yoo, J.-M. Chung, 5G multi-RAT URLLC and eMBB dynamic task offloading with MEC resource allocation using distributed deep reinforcement learning, IEEE Internet Things J. 9 (20) (2022) 20733–20749.

[19] X. Huang, L. He, X. Chen, L. Wang, F. Li, Revenue and energy efficiency-driven delay-constrained computing task offloading and resource allocation in a vehicular edge computing network: a deep reinforcement learning approach, IEEE Internet Things J. 9 (11) (2022) 8852–8868.

[20] X. Chen, G. Liu, Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks, IEEE Internet Things J. 8 (13) (2021) 10843–10856.

[21] B. Shi, Y. Wu, Task offloading and resource allocation strategies among multiple edge servers, IEEE Internet Things J. 11 (8) (2024) 14647–14656.

[22] D. Triyanto, I.W. Mustika, Widyawan, P. Pavarangkoon, Fairness-aware computation offloading for mobile edge computing with energy harvesting, IEEE Access 13 (2025) 55345–55357.

[23] P. Ge, J. Zhao, H. Zhang, D. Zou, M. Wang, Green hybrid energy harvesting for intelligent mobile edge computing in Internet of things, Phys. Commun. 61 (2023) 102171–102180.

[24] NR and NG-RAN Overall Description, V18.2.0, ETSI Standard TS 138 300, 2024.

[25] MEC Framework and Reference Architecture, V3.2.1, ETSI GS MEC 003, 2024.

[26] M. Qin, N. Cheng, T. Yang, W. Xu, Q. Yang, R.R. Rao, Service oriented energy-latency tradeoff for IoT task partial offloading in MEC-enhanced multi-RAT networks, IEEE Internet Things J. 8 (3) (2021) 1896–1907.

[27] W. Wu, Q. Yang, P. Gong, K.S. Kwak, Energy-efficient resource optimization for OFDMA-based multi-homing heterogenous wireless networks, IEEE Trans. Signal Process. 64 (22) (2016) 5901–5913.

[28] C. She, Y. Duan, G. Zhao, T.Q.S. Quek, Y. Li, B. Vucetic, Cross-layer design for mission-critical IoT in mobile edge computing systems, IEEE Internet Things J. 6 (6) (2019) 9360–9374.

[29] A.P. Miettinen, J.K. Nurminen, Energy efficiency of mobile clients in cloud computing, in: Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing, USENIX, 2010, pp. 1–7.

[30] Study on scenarios and requirements for next generation access technologies, 3GPP, Sophia Antipolis, France, Rep. TSG RAN TR38.913 V14, 2017.

[31] W. Yang, G. Durisi, T. Koch, Y. Polyanskiy, Quasi-static multiple antenna fading channels at finite blocklength, IEEE Trans. Inf. Theory 60 (7) (2014) 4232–4264.

[32] A. Gravey, J.-R. Louvion, P. Boyer, On the Geo/D/1 and Geo/D/1/n queues, Perform. Eval. 11 (2) (1990) 117–125.

[33] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, D.O. Wu, Energy-optimal mobile cloud computing under stochastic wireless channel, IEEE Trans. Wirel. Commun. 12 (9) (2013) 4569–4581.

[34] S.-W. Ko, K. Han, K. Huang, Wireless networks for mobile edge computing: spatial modeling and latency analysis, IEEE Trans. Wirel. Commun. 17 (8) (2018) 5225–5240.

[35] M. Harchol-Balter, Performance Modeling and Design of Computer Systems: Queueing Theory in Action, Cambridge Univ. Press, New York, 2013.

[36] Multi-Access Edge Computing (MEC) MEC 5G Integration, V2.1.1, document ETSI GR MEC 031, ETSI, Sophia Antipolis, France, 2020.

[37] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher, J.D. Bohlinger (Eds.), Complexity of Computer Computations, Plenum Press, Berlin, 1972, pp. 85–103.

[38] M.R. Garey, D.S. Johnson, 'Strong' NP-completeness results: motivation, examples, and implications, J. ACM 25 (3) (1978) 499–508.

[39] S. Fujimoto, H. van Hoof, D. Meger, Addressing function approximation error in actor-critic methods, in: Proceedings of the 2018 International Conference on Machine Learning, IMLS, 2018, pp. 1587–1596.

[40] T. Schaul, J. Quan, I. Antonoglou, D. Silver, Prioritized experience replay, in: Proceedings of the 2016 International Conference on Learning Representations, CBLS, 2016, pp. 1–21.

[41] Study on Physical Layer Enhancements for NR Ultra-Reliable and Low Latency Case (URLLC), 3GPP, Sophia Antipolis, France, Rep. TSG RAN TR38.824 V16.0.0, 2019.

[42] Study on Communication for Automation in Vertical Domains, 3GPP, Sophia Antipolis, France, Rep. TSG SA TR22.804 V16.3.0, 2020.

[43] 5G NR-User Equipment (UE) Radio Transmission and Reception Part 1: Range 1 Standalone, V15.2.0, ETSI Standard TS 138 101-1, 2018.

[44] C. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (1992) 279–292.

[45] J. Hu, M.P. Wellman, Multiagent reinforcement learning: theoretical framework and an algorithm, in: Proceedings of the 1998 International Conference on Machine Learning, IMLS, 1998, pp. 242–250.

Dongjun Jung received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, Republic of Korea, in 2019, and the Ph.D degree in electrical and electronic engineering from Yonsei University in 2025. From 2019 to 2025, he was a research member with the Communications and Networking Laboratory (CNL), Yonsei University. He is currently a staff engineer with Samsung Electronics. His research interests include MECs, IoT, Wi-Fi, 5G & 6G mobile systems, augmented reality (AR) systems, military networks, reinforcement learning, game theory, large language model (LLM), and deep learning based optimization.

Jong-Moon Chung received B.S. and M.S. degrees in electronic engineering from Yonsei University and Ph.D. in electrical engineering from the Pennsylvania State University. Since 2005, he has been a professor in the School of Electrical and Electronic Engineering at Yonsei University, where he is also Associate Dean of the College of Engineering and professor of the Department of Emergency Medicine in the College of Medicine and Director of CNL. From 1997 to 1999, he was an assistant professor and instructor at the Pennsylvania State University. From 2000 to 2005, he was with the Oklahoma State University as a tenured associate professor. He is an IEEE Fellow, Vice President of the IEEE Consumer Technology Society (CTSoc) and Product Safety Engineering Society (PSES), Senior Editor of the *IEEE Transactions on Consumer Electronics*, Section Editor of the *Wiley ETRI Journal*, Chair Editor-in-Chief of the *KSII Transactions on Internet and Information Systems*, and served as an Editor of the *IEEE Transactions on Vehicular Technology* from 2011 to 2021.

Hea-Sook Park received the Ph.D. degree in computer science from the Department of Computer Engineering, Chungnam National University, Daejeon, Republic of Korea, in 2005. Since 1994, she has been a Principal Member of the Research Engineering Staff with the Electronics and Telecommunications Research Institute, Daejeon, where she is currently working on intelligent stealth networks in the Defense ICT Convergence Research Section. Her significant areas of research interests include network QoS and trustworthy IP networks.