Deep Reinforcement Learning-based Online Resource Management for UAV-Assisted Edge Computing with Dual Connectivity

Linh T. Hoang, *Graduate Student Member, IEEE,* Chuyen T. Nguyen, and Anh T. Pham, *Senior Member, IEEE*

***Abstract*—Mobile Edge Computing (MEC) is a key technology towards delay-sensitive and computation-intensive applications in future cellular networks. In this paper, we consider a multi-user, multi-server system where the cellular base station is assisted by a UAV, both of which provide additional MEC services to the terrestrial users. Via dual connectivity (DC), each user can simultaneously offload tasks to the macro base station and the UAV-mounted MEC server for parallel computing, while also processing some tasks locally. We aim to propose an online resource management framework that minimizes the average power consumption of the whole system, considering long-term constraints on queue stability and computational delay of the queueing system. Due to the coexistence of two servers, the problem is highly complex and formulated as a multi-stage mixed integer non-linear programming (MINLP) problem. To solve the MINLP with reduced computational complexity, we first adopt Lyapunov optimization to transform the original multi-stage problem into deterministic problems that are manageable in each time slot. Afterward, the transformed problem is solved using an integrated learning-optimization approach, where model-free Deep Reinforcement Learning (DRL) is combined with model-based optimization. Via extensive simulation and theoretical analyses, we show that the proposed framework is guaranteed to converge and can produce nearly the same performance as the optimal solution obtained via an exhaustive search.**

***Index Terms*—Lyapunov Optimization, Mobile Edge Computing, Deep Reinforcement Learning (DRL), Queueing Networks.**

I. Introduction

Mobile Edge Computing (MEC) refers to an emerging distributed computing paradigm that brings cloud processing and storage capabilities closer to the end-user (i.e., to the network's edges) [1]–[3]. The technology has been widely recognized as a promising solution to solve the challenges aligned with the rapid growth of mobile applications and the Internet of Things (IoT). By offloading part of computational tasks to the MEC server, the end-users, especially ones that are battery-based and with limited hardware capabilities, can experience a better quality of service (QoS) in computation-intensive and latency-critical applications [4]–[7].

In support of the dynamic and rapid deployment of MEC networks, mounting the MEC server on Unmanned Aerial Vehicles (UAVs) has recently attracted attention in both industry and academia [2], [3], [8]–[10]. In such an approach, the UAV can act as a flying base station that can effectively complement existing cellular networks by providing additional computational services to the ground user. Due to the inherent mobility and flexibility of UAVs, this approach is an on-demand solution well-suited in hotspot areas or rural areas where network capacity is insufficient [3], [8], [9], [11]. In addition, the Dual Connectivity (DC) technology [12] can also be integrated into the network to further enhance the computational efficiency of the edge device. Using DC, edge devices are enabled to communicate simultaneously with several eNodeBs, which might significantly improve the network's throughput and mobility support [12]–[16]. Indeed, the concept of DC was introduced in the Third-Generation Partnership Project (3GPP) Release 12 and has recently been widely recognized as a promising approach for the deployment of ultra-dense 5G heterogeneous networks [12]–[16]. Associating users requesting computationally intensive services to a single MEC server might result in an overload and possible service denials of the server to other users. DC allows the users to offload tasks to two servers simultaneously for parallel computing, thus being a promising solution to balance the workload and avoid the service denial issue. Following these trends, MEC networks can be configured where the mBS and the UAV act as the Master eNodeB (MeNB) and the Second eNodeB (SeNB), respectively, both equipped with a MEC server. The edge user can then offload tasks to both the MEC servers simultaneously for parallel computing with abundant computational resources.

Besides mentioned advantages, integrating UAVs and DC into the MEC networks poses various challenges; one is the time complexity in resource management of the system. In general, the optimization in a multi-user, multi-server MEC network involves solving a mixed integer non-linear programming (MINLP) problem that jointly determines the channel assignment (i.e., the user association) for MEC servers and resource management for communication (e.g., bandwidth allocation) and computation (e.g., CPU frequency selection for local and remote computing) at the server and the user. Solving such a problem is computationally expensive, especially given the existence of a large number of users. Various solutions have been proposed to tackle the issue, such as metaheuristic methods [4], [17], [18], convex relaxation of binary variables [19], local search-based approaches [20], and decomposition-based methods [21]. Still, these conventional methods share a common of requiring a large number of iterations to bring out a good performance and are thus not suitable for real-time

Linh T. Hoang and Anh T. Pham are with the Computer Communications Laboratory, The University of Aizu, Aizuwakamatsu 965-8580, Japan. E-mail: d8232104@u-aizu.ac.jp, pham@u-aizu.ac.jp

Chuyen T. Nguyen is with Hanoi University of Science and Technology, Hanoi, Vietnam. E-mail: chuyen.nguyenthanh@hust.edu.vn

control of dynamic MEC systems.

To tackle the issue of time complexity, data-driven solutions such as deep reinforcement learning (DRL) are promising candidates that can perform very well while satisfying real-time control requirements [22]–[27]. The DRL framework harnesses deep neural networks (DNNs) to learn the optimal policy that directly maps the system state (e.g., the channel condition and the number of backlog tasks) to a proper action (i.e., resource management decision) in each time slot. Training is performed via continuous interaction between the optimization solver and the environment (i.e., the MEC network) to maximize the reward following the decision in each step (e.g., the system's energy efficiency and throughput). Indeed, using DNNs in optimization is a model-free approach in which the solver learns from experience (i.e., driven by the training data) to construct an optimal mapping policy, rather than relying on complex mathematical models that might not always be accurate and readily available. However, purely relying on the model-free solution has been reported to lead to unstable performance and suffer from slow convergence or even divergence [23], [25], [26]. A proper approach could be letting the DNN take part of the optimization (e.g., for optimizing binary variables) while still using conventional model-based methods for the rest. Indeed, the integration of data-driven and conventional model-based methods has improved the robustness and convergence of the DRL framework via online training [23]–[26].

The other challenge in optimizing MEC networks is the preference for long-term key performance indicators (KPIs) of dynamic queueing systems, which refer to time-evolving queueing networks where decisions made in a time slot affect the optimization in subsequent slots [26], [28]. A typical example could be minimizing the long-term average power consumption, subject to queue stability constraints and given the randomness of the environment (e.g., channel gains and task arrival). Despite its importance, most existing DRL-based solutions [23], [24], [29]–[31] do not focus on the long-term performance when solving resource management problems in MEC networks. A well-known approach to cope with the long-term KPIs of a dynamic system is Lyapunov optimization [28]. The framework can be used to transform a multi-state problem into deterministic per-time slot sub-problems while providing a theoretical guarantee to long-term system stability. The combination of the two robust tools, DRL and Lyapunov framework, thus is a promising approach to solving the MINLP problem of network resource management while monitoring the long-term KPIs, especially in large-scale multi-user, multi-server MEC systems [26].

In this paper, we consider a UAV-assisted MEC network with DC, where the UAV and the mBS act as the MeNB and the SeNB, respectively, to provide edge computing services to a set of ground users. Via DC, one mobile user can simultaneously obtain communication resources from both MEC servers in support of parallel computing. Under randomness of channel condition and task arrival, an MINLP is formulated to minimize the average power consumption of the whole system (including the user and the UAV, which are all battery-operated), given constraints on long-term queue stability and average task execution delay threshold. We aim to develop an online resource management algorithm with reduced computational complexity that produces system-wide energy efficiency while satisfying all QoS requirements for the user. We jointly optimize various system variables to achieve the goal, including channel assignment, local and remote computational resource scheduling and bandwidth allocation in each time slot. The Lyapunov framework is adopted to transform the original multi-stage problem into deterministic per-time slot problems. A hybrid scheme of combining the model-free DRL and model-based optimization is then proposed to solve resource management optimization in each time slot. To the authors' best knowledge, this is the first work considering a dynamic multi-user, multi-server MEC system with assistance from UAVs via DC and developing a DRL framework for resource management in such a system. The main contributions can be summarized as follows:

1) *Power Minimization for a multi-user, multi-server MEC System with dual connectivity:* In support of parallel computing, we propose to utilize a UAV to assist edge computing in a cellular network via DC. The problem of resource management is formulated as a multi-stage MINLP to minimize the long-term average of the weighted-sum power consumption, constrained on long-term queue stability and task execution delay.
2) *Lyapunov-guided DRL Approach:* we develop a Lyapunov-guided DRL framework that can efficiently produce sub-optimal solutions. DRL is to cope with the complexity of the problem with the coexistence of two servers. Meanwhile, Lyapunov optimization is to deal with the long-term constraints on queue stability and average task execution delay of the queueing system.
3) *Hybrid Approach for Actor-Critic Structure:* The proposed framework integrates conventional model-based optimization and a data-driven approach via model-free DRL. The actor module utilizes a DNN and an efficient action quantizer to balance exploration and exploitation in producing channel assignment decisions. To accurately evaluate decisions made by the actor, the critic module utilizes model-based optimization rather than using another DNN conventionally.
4) We provide theoretical analyses and numerical results via extensive simulation to demonstrate the efficiency of the proposed method.

The remainder is organized as follows. Related works are provided in Section II. Sections III and IV detail the system model and problem formulation, respectively. Sections V and VI provide the description and theoretical analyses of the proposed framework. The numerical results are provided in Section VII. Finally, Section VIII concludes the paper.

II. Related Works

User Association. In recent years, several works have studied the user association problem (i.e., channel assignment or server selection) in resource management for multi-user, multi-server MEC networks [5], [20], [21], [30], [31]. Dai *et al*. [5] formulated a problem of joint computation offloading

TABLE I: A comparison of our work with existing DRL-based resource management schemes

	Our work	[25]	[23]	[24]	[29]	[30]	[31]
Multiple users	✓	✓	✓	✓	-	-	-
Multiple servers	✓	-	-	✓	✓	✓	✓
Optimize network resources allocation (channel bandwidth or transmission time)	✓	✓	✓	✓	✓	✓	✓
Optimize task offloading of the user	✓	✓	✓	✓	✓	-	✓
Optimize computation on the server side	✓	-	-	✓	-	-	-
Towards long-term KPIs of queueing systems	✓	✓	-	-	-	-	-
Dual connectivity and parallel computation	✓	-	-	-	-	-	-

and user association to minimize overall power consumption of a MEC system where each user has multiple mutually dependent tasks. Tran *et al*. [20] studied the problem of joint sub-channel assignment and resource allocation of a multi-cell network to maximize a weighted sum of reduction in task completion time and energy consumption. Liu *et al*. [30] considered the switching cost when a mobile user migrates its service from one server to another and modeled the problem of continuous server selection as a Markov Decision Process (MDP). Guo *et al*. [31] proposed an online learning-based MEC server selection mechanism under incomplete network information to minimize the time average task execution delay. Hu *et al*. [21] proposed a submodular optimization-based server selection method for mobile users to optimize the long-term energy-delay tradeoff. However, the works mentioned above have yet to investigate the user association problem in a parallel-computing scenario with dual connectivity. Instead, users have been grouped into separate clusters and allowed to connect to only one server at a time. More investigation is thus needed to fill the gap.

DRL for Resource Management. The utilization of DRL-based methods in optimization for MEC networks has recently attracted lots of attention from research community [23]–[25], [29]–[31]. Min *et al*. [29] proposed a reinforcement learning-based offloading scheme for IoT devices with energy harvesting (EH) to select the MEC server according to the current battery level and the predicted amount of the harvested energy. Huang *et al*. [23] proposed a DRL-based offloading framework that utilizes a deep neural network to produce potential binary offloading solutions, while a model-based optimization module is responsible for evaluating candidate decisions and labeling training data samples. Wu *et al*. [24] developed a hybrid framework that combines a deep Q network and convex optimization for determining offloading strategies at the user side and allocating resources at the computational access point. However, the works [23], [24], [29]–[31] only considered quasi-static scenarios and failed to adequately address long-term performance requirements (such as queue stability and average energy consumption) of a time-evolving queueing system. Following the direction toward long-term KPIs, Bi *et al*. [25] recently proposed a hybrid optimization-learning framework called LyDROO. The framework combines Lyapunov optimization and DRL to optimize task offloading, where the user either computes tasks locally or offloads the tasks (i.e., binary offloading).

In this paper, we also integrate Lyapunov optimization and DRL into a resource management framework to cope with long-term KPIs of a queueing network. Compared to [25], our innovations are three-fold. First, we propose a new actor module that utilizes a DNN to optimize parallel computation between users and servers. The method in [25] allows the user to either process local computation or offload tasks to an edge serve; thus it dose not apply to the parallel paradigm. Second, our proposed framework jointly optimizes not only the user side but also the server side. Since the problem involves many interacting network entities, we propose a new model-based critic module, which is entirely different from [25]. Third, [25] considered a single powerful MEC server with no limit on the number of users served at a time. Thus, their algorithms cannot be directly applied in our study, where we consider a multi-server system with dual connectivity and specify constraints on the server's serving capability. A comprehensive comparison between our proposed scheme and other existing DRL-based methods is summarized in Table I. It is noteworthy that the critic module's algorithm in this paper is adopted in part from our previous work [32] to optimize local computation on the user side.

TABLE II: Summary of Key Notations

Symbol	**Definition**
$Q_i^l(t)$	Local queue length of the user i at the beginning of time slot t
$Q_i^s(t)$	Queue length of the UAV dedicated for the user i at time slot t
A_i^t	Task arrival of user i in time slot t
$f_{l,i}^t$	Local computation frequency of user i in time slot t
$f_{c,i}^t$	The UAV's CPU frequency dedicated for processing user i's tasks in time slot t
$p_{l,i}^t$	Power consumption for local task execution of user i in time slot t
$p_{\mathrm{Tx},i}(t)$	Transmit power for task offloading of user i in time slot t
$p_c(t)$	Power consumption for task execution of the UAV in time slot t
$\alpha_{i,j}^t$	The ratio of bandwidth allocated to user i on the server j's channel in time slot t
$r_{i,j}^t$	Offloading volume of user i on server j's link in time slot t
l_i^t	The amount of tasks processed locally by user i in time slot t
c_i^t	The amount of user i's tasks processed by the UAV in time slot t
W_j	The total bandwidth of server j
$h_{i,j}^t$	Channel gain of user i for the link to server j in time slot t
$x_{i,j}^t$	Link association for user i on the server j's channel in time slot t

III. System Model

As illustrated in Fig. 1, we consider a multi-server MEC system with Dual Connectivity (DC). There are two MEC servers, one located at a macro base station (i.e., the Master eNB, MeNB) and the other mounted on a UAV (i.e., the Secondary eNB, SeNB), provides additional edge computing services to a set of ground users. Each mobile user can simultaneously connect to the two MEC servers simultaneously. From now on, the UAV-mounted MEC sever and the SeNB,

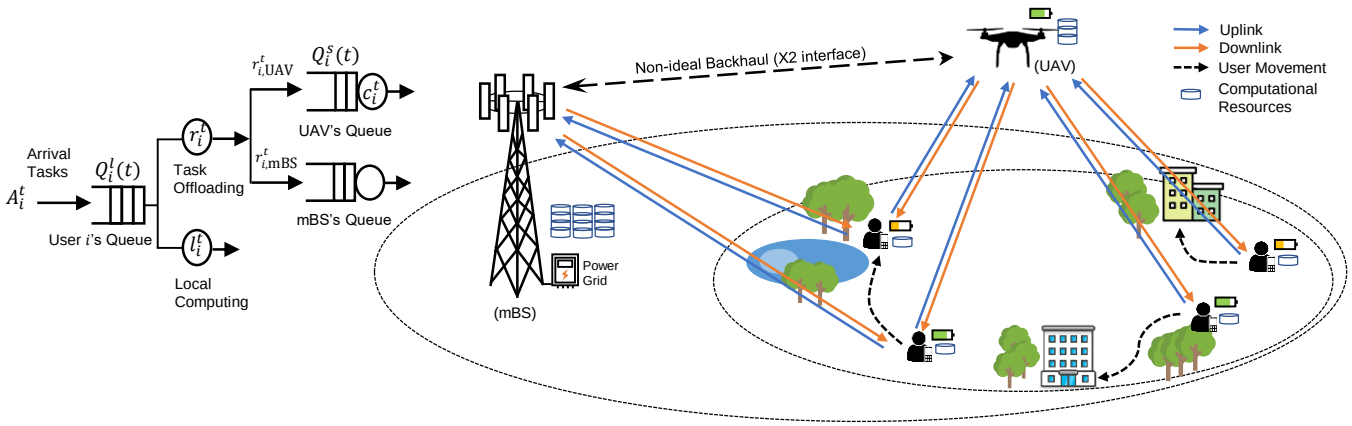


Fig. 1: Dual-Connectivity-supported UAV-assisted MEC system model

similarly to the macro base station and the MeNB, will be used interchangeably.

For convenience, we denote the index sets of the mobile devices, the MEC servers, and the time slots respectively as $\mathcal{N} \triangleq \{1, 2, ...N\}$, $\mathcal{S} \triangleq \{\mathrm{UAV}, \mathrm{mBS}\}$, and $\mathcal{T} \triangleq \{1, 2, ...\}$. It is noted that in the following, we index each user by the letter i and the MEC server by letter j, i.e., $i \in \mathcal{N}, j \in \mathcal{S}$.

The modeling of task computation, queueing system, task offloading, and power consumption are presented below. For ease of reference, key notations used in the article are summarized in Table II.

A. Task Queuing Model

We assume that the mobile devices are processing independent and fine-grained tasks [1]. Each task is represented by a volume of bits, which can be decomposed into several packets transmitted to nearby MEC servers and processed in parallel. At the beginning of each slot, a volume of A_i^t bits arrives at the user i and can be processed starting from the next slot. Without loss of generality, we assume that A_i^t is independent and identically distributed (i.i.d) over time slots with Poisson distribution and an average rate $\mathbb{E}[A_i^t] = \lambda_i$ (bits), $i \in \mathcal{N}$.

In the tth time slot, the user processes l_i^t bits locally and on opportunity can upload $r_{i,\mathrm{UAV}}^t$ and $r_{i,\mathrm{mBS}}^t$ bits to the UAV-mounted MEC and the macro base station, respectively. The newly arrived task volumes at the beginning of one slot will be buffered in the user queue before they can be processed in subsequent slots. Let $Q_i^l(t)$ denote the local queue length of user i at the beginning of time slot t; the queue update process can be expressed as

$$Q_i^l(t+1) = \max\left\{Q_i^l(t) - D_i^t, 0\right\} + A_i^t, t \in \mathcal{T}, \tag{1}$$

where $D_i^t \triangleq l_i^t + r_{i,\mathrm{UAV}}^t + r_{i,\mathrm{mBS}}^t$ denotes the amount of tasks departing from the user i's local queue in time slot t.

At the UAV side (i.e., the SeNB), we assume that the UAV maintains N dedicated queues, one for each user, to buffer tasks offloaded by users. Let c_i^t denote the amount of tasks from user i executed by the UAV in time slot t; the UAV's task queue dedicated for user i, denoted by $Q_i^s(t)$, can be derived similarly as

$$Q_i^s(t+1) = \max\left\{Q_i^s(t) - c_i^t, 0\right\} + r_{i,\mathrm{UAV}}^t, t \in \mathcal{T}. \tag{2}$$

In this paper, all user and UAV task queues are assumed with sufficiently large capacity. In addition, without loss of generalization, all task queues are empty initially, i.e., $Q_i^l(0) = Q_i^s(0) = 0, i \in \mathcal{N}$.

Regarding the macro BS (i.e., the MeNB), we assume that the server has redundant computational resources and is powered by an electrical grid; therefore, we do not consider the macro BS's queues and power consumption in optimization. In other words, tasks offloaded to the macro BS will not be buffered in queues but executed promptly, and the energy consumed is less important than other network entities.

According to *Little's Law* [33], the average delay experienced by one user is proportional to the long-term average number of tasks awaiting in the system. Thus, we exploit the average queue length at the user and the UAV, denoted by $\overline{Q}_i^l$ and $\overline{Q}_i^s$, as a measure of the task completion delay for local and remote task processing. Furthermore, the two thresholds $Q_{l,i}^{\mathrm{th}}$ and $Q_{s,i}^{\mathrm{th}}$ are defined as a Quality of Service (QoS) constraint for the ith user as

$$\overline{Q}_i^l = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[Q_i^l(t)\right] \le Q_{l,i}^{\mathrm{th}} \tag{3}$$

$$\overline{Q}_i^s = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[Q_i^s(t)\right] \le Q_{s,i}^{\mathrm{th}} \tag{4}$$

where the expected values of the queue length (i.e., $\mathbb{E}\left[Q_i^l(t)\right]$ and $\mathbb{E}\left[Q_i^s(t)\right]$) are taken over the randomness of the channel gain and task arrival in a time slot. It is worth noting that from (1) we have $Q_i^l(t+1) \ge A_i^t$, thus $\overline{Q}_i^l \ge \mathbb{E}[A_i^t]$ for all users. Therefore, $Q_{l,i}^{\mathrm{th}}$ should be selected such that $Q_{l,i}^{\mathrm{th}} \ge \lambda_i, i \in \mathcal{N}$.

B. Task Execution Model

To process tasks locally, the mobile user needs to assign a specific number of CPU frequencies for each task. Let $f_{l,i}^t$ denote the local CPU frequency of user i in time slot t; the amount of locally-computed tasks in time slot t can then be expressed as

$$l_i^t = \tau f_{l,i}^t / L_i. \tag{5}$$

Here, τ denotes the time slot length and L_i denotes the processing density, defined as the number of CPU cycles

required for user i to process one bit. According to circuit theory, the power consumption for local execution at the ith user is given by [34], [35]

$$p_{l,i}(t) = \kappa_i \left(f_{l,i}^t\right)^3, \tag{6}$$

where the parameter κ_i is the effective switched capacitance of the CPU at the ith device, and dependent on the hardware architecture.

Similarly, at the UAV, the amount of tasks computed by the MEC server for the ith user in time slot t can be expressed by

$$c_i^t = \tau f_{c,i}^t / L_s, \tag{7}$$

where $f_{c,i}^t$ denotes the CPU frequency resources that the UAV allocates for computing the ith user's tasks; L_s denotes the processing density for the UAV's CPU to process one bit. The power consumption of the UAV for computation can be defined similarly as

$$p_c(t) = \sum_{i \in \mathcal{N}} \kappa_s \left(f_{c,i}^t\right)^3, \tag{8}$$

where κ_s denotes effective switched capacitance of the UAV's CPU. We assume that the computational capability of the UAV is stronger than that of the mobile user, but limited by the maximum CPU frequency, i.e., $f_{c,i}^t \le f_c^{\max}, i \in \mathcal{N}$.

C. Task Offloading Model

Channel power gain from user i to the MEC server j:

$$h_{i,j}^t = \frac{\tilde{h}_{i,j}^t g_j}{\left(d_{i,j}\right)^{\gamma_j}}, \tag{9}$$

where $d_{i,j}$ denotes the distance from user i to the MEC server j, γ_j denotes the path loss exponent ($\gamma_j \ge 2$), g_j denotes the reference channel gain, and $\tilde{h}_{i,j}^t$denotes the small-scale fading channel power gain, which is assumed to have a finite mean value, $\mathbb{E}\left[\tilde{h}_{i,j}^t\right] < \infty$ [34], for the server j's link.

To allocate radio resources to the mobile device, each MEC server will first select a subset of users (assuming that one server cannot serve all the users at the same time), then allocate each user in that subset an appropriate bandwidth for communication offloading. Let $x_{i,j}^t$ denotes the link association for the user i on the server j's communication channel in time slot t: $x_{i,j}^t = 1$ indicates that the user i could utilize bandwidth allocated on the server j's channel; otherwise, no bandwidth is allocated and offloading is prohibited. $\mathcal{N}_j^t \triangleq \left\{i \in \mathcal{N} \middle| x_{i,j}^t = 1\right\} \subset \mathcal{N}$ then can be defined as the set of mobile devices associated with the MEC server j in time slot t. Similarly, the set of the MEC servers that associate with the ith user in time slot t can be defined as $\mathcal{S}_i^t \triangleq \left\{j \in \mathcal{S} \middle| x_{i,j}^t = 1\right\} \subset \mathcal{S}$.

Regarding the communication energy, according to the Shannon-Hartley formula, the transmit power for user i to offload r_i^t bits can be obtained as [36]

$$p_{\mathrm{Tx},i}(t) = \sum_{j \in \mathcal{S}_i^t} \left(2^{\frac{r_{i,j}^t}{W_j \alpha_{i,j}^t \tau}} - 1\right) \frac{N_0 W_j}{h_{i,j}^t}, \tag{10}$$

where W_j denotes the total bandwidth of the server j, $\alpha_{i,j}^t$ denotes the bandwidth ratio allocated to user i on the server j's channel, and N_0 denotes the background noise density. In (10), $r_{i,j}^t$ denotes the offloading volume of the user i on the server j's communication channel in time slot t, thus $r_i^t = \sum_{j \in \mathcal{S}_i^t} r_{i,j}^t$.

It is worth noting that since the mobile user is supported by dual connectivity, one user can connect to both the two MEC servers at the same time, i.e., $x_{i,\mathrm{UAV}}^t$ and $x_{i,\mathrm{mBS}}^t$ can be both equal to one in a time slot, thus

$$\left|\mathcal{S}_i^t\right| = \sum_{j \in \mathcal{S}} x_{i,j}^t \le 2, i \in \mathcal{N}, t \in \mathcal{T}, \tag{11}$$

where $|\mathcal{A}|$ denotes the number of elements in set $\mathcal{A}$. Due to signaling overhead for resource management, we assume that server j is able to serve at most $\chi_j^{\max}$ users in a time slot, i.e.,

$$\left|\mathcal{N}_j^t\right| = \sum_{i \in \mathcal{N}} x_{i,j}^t \le \chi_j^{\max}, j \in \mathcal{S}, t \in \mathcal{T}. \tag{12}$$

IV. The Multi-stage MINLP Problem of Power Consumption Minimization

A. Problem Formulation

We focus on the weighted-sum system power consumption, which consists of power consumed for task execution at the user device and the UAV-mounted MEC server, as well as the user's transmit power for task offloading. Energy consumed for other purposes, such as for maintaining the basic operations of the MEC system and for propulsion of the UAV, are omitted for simplicity. Accordingly, the system's power consumption at time slot t, denoted by $P_{\mathrm{sys}}(t)$, can be calculated as a weighted sum as

$$P_{\mathrm{sys}}(t) = \psi_c p_c(t) + \sum_{i \in \mathcal{N}} \psi_i \left(p_{l,i}(t) + p_{\mathrm{Tx},i}(t)\right), \tag{13}$$

where ψ_i and ψ_c are positive numbers denoting the weight factors for the power consumption of user i and the UAV, respectively. ψ_i and ψ_c can be adjusted to reflect the system's preference in optimizing the power consumption of different nodes, as well as to balance the impact of the UAV's and the mobile device's energy [34].

The ultimate goal of the optimization is to minimize the long-term average of the system power consumption, given constraints on the stability of task queues and the limit on the radio and computational resources. The optimization variables include the user's local computation, the volume of offloaded tasks, the UAV's remote processing scheduling, and the radio resource allocation.

Let $\mathbf{X} = \{\mathbf{X}^t\}_{t \in \mathcal{T}}$ denote the combination of all optimization variables over time. Additionally, let $\mathbf{X}^t = \left\{\mathbf{x}_j^t, \boldsymbol{\alpha}_j^t, \mathbf{r}_j^t, \mathbf{f}_l^t, \mathbf{f}_c^t\right\}_{j \in \mathcal{S}}$ denote the combined vector of optimization variables at time slot t for all server j in $\mathcal{S}$: $\mathbf{x}_j^t = \{x_{i,j}^t\}_{i \in \mathcal{N}}$, $\boldsymbol{\alpha}_j^t = \{\alpha_{i,j}^t\}_{i \in \mathcal{N}}$, $\mathbf{r}_j^t = \left\{r_{i,j}^t\right\}_{i \in \mathcal{N}}$, $\mathbf{f}_l^t = \{f_{l,i}^t\}_{i \in \mathcal{N}}$,

$\mathbf{f}_c^t = \{f_{c,i}^t\}_{i\in\mathcal{N}}$. Then, the problem can be formulated as a multi-stage MINLP problem as

$$\mathbf{P1}: \min_{\mathbf{X}} \quad \lim_{t\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[P_{\text{sys}}(t)\right] \tag{14a}$$

$$\text{s.t.} \quad x_{i,j}^t \in \{0,1\}, \left|\mathcal{N}_j^t\right| \le \chi_j^{\max}, i\in\mathcal{N}, j\in\mathcal{S}, \in\mathcal{T}, \tag{14b}$$

$$0 \le \alpha_{i,j}(t), \sum_{i\in\mathcal{N}_j} \alpha_{i,j}^t \le 1, i\in\mathcal{N}, j\in\mathcal{S}, t\in\mathcal{T}, \tag{14c}$$

$$0 \le f_{l,i}^t \le f_i^{\max}, i\in\mathcal{N}, t\in\mathcal{T}, \tag{14d}$$

$$0 \le f_{c,i}^t \le f_c^{\max}, i\in\mathcal{N}, t\in\mathcal{T}, \tag{14e}$$

$$0 \le p_{\text{Tx},i}(t) \le p_{\text{Tx},i}^{\max}, i\in\mathcal{N}, t\in\mathcal{T}, \tag{14f}$$

$$l_i^t + r_{i,\text{UAV}}^t + r_{i,\text{mBS}}^t \le Q_i^l(t), i\in\mathcal{N}, t\in\mathcal{T}, \tag{14g}$$

$$c_i^t \le Q_i^s(t), i\in\mathcal{N}, t\in\mathcal{T}, \tag{14h}$$

$$\lim_{t\to\infty} \frac{\mathbb{E}\left[Q_i^l(t)\right]}{t} = 0, i\in\mathcal{N}, \tag{14i}$$

$$\lim_{t\to\infty} \frac{\mathbb{E}\left[Q_i^s(t)\right]}{t} = 0, i\in\mathcal{N}, \tag{14j}$$

(3) and (4)

In **P1**, (14b) denotes that the server j can server at most $\chi_j^{\max}$ users at a time. (14c) ensures that the total bandwidth used for the server j's' uplink communication is bounded by W_j. (14d) and (14e) indicate the maximum CPU frequency of the user and the UAV, denoted by $f_i^{\max}$ and $f_c^{\max}$, respectively. (14f) denotes the maximum transmit power of the mobile device on each communication link. (14g) and (14h) guarantee that the amount of tasks processed in a time slot (i.e., tasks offloaded and computed by the user and tasks computed by the UAV) does not surpass the backlog of task queues in each time slot. Finally, (14i) and (14j) indicate the mean rate stability [28] for the local and remote queues. Note that (14i) and (14j) do not provide any guarantee of the time-average expected backlogs in queues (and thus the average computation delay). The QoS constraints (3) and (4), which is a stronger form of stability [28], are thus useful.

We observe that **P1** is a stochastic optimization problem of a time-evolving system. Indeed, radio and computation resource management decisions need to be made in each time slot under the randomness of the arrival task and the fading channel. Furthermore, optimal decisions are temporally correlated and should be adaptive to the time-varying system states, such as the current queue size at the mobile device and the UAV. Solving **P1** is challenging also because the optimization evolves a large number of interdependent optimization variables. Specifically, the radio resource management variables among different end users (i.e., $x_{i,j}^t$ and $\alpha_{i,j}^t, i\in\mathcal{N}$) are coupled and interdependent with the computational resource scheduling at both sides (i.e., $f_{l,i}^t$ and $f_{c,i}^t, i\in\mathcal{N}$), which suggests that a joint optimization approach is indeed needed. Later in the numerical results, we show that the over-aggressive approach (e.g., a greedy policy based on channel gain or queue length) could not solve the formulated problem effectively.

In the following, instead of solving **P1** directly, we consider its modified version, denoted by **P2**, to obtain an efficient asymptotically optimal online solution as follows. First, we can rewrite (14c) as $\boldsymbol{\alpha}_j^t \in \mathcal{A} \triangleq \left\{\boldsymbol{\alpha}_j^t \in \mathbb{R}_N^+ \middle| \sum_{i\in\mathcal{N}_j} \alpha_{i,j}^t \le 1\right\}, j\in\mathcal{S}, t\in\mathcal{T}$. To obtain **P2**, we replace (14c) by the following constraint,

$$\boldsymbol{\alpha}_j^t \in \tilde{\mathcal{A}} \triangleq \left\{\boldsymbol{\alpha}_j^t \in \mathbb{R}_+^N \middle| \sum_{i\in\mathcal{N}_j} \alpha_{i,j}^t \le 1 \text{ and } \alpha_{i,j}^t \ge \epsilon_A, i\in\mathcal{N}_j^t\right\} \tag{14c'}$$

where $\epsilon_A \in (0, 1/N)$ is a constant. Constraint (14c') causes the transmit power function in (10) continuous and differentiable with respect to $\alpha_{i,j}^t, j\in\mathcal{S}_i^t$. It is worth noting that although the optimal solution to **P2** is only a approximation of the optimal solution to **P1**, we can make them arbitrarily close by setting ϵ_A to be sufficiently small.

B. Lyapunov-guided Problem Transformation

We adopt Lyapunov optimization framework [28] to decouple the multi-stage problem **P2** into deterministic problems that can be solved in each time slot.

First, to cope with the QoS constraints (3) and (4) on the long-term average of the queue length, we introduce two virtual queues for each mobile,

$$Z_i^l(t+1) = \max\left\{Z_i^l(t) + Q_i^l(t+1) - Q_{l,i}^{\text{th}}, 0\right\}, \tag{15}$$

$$Z_i^s(t+1) = \max\left\{Z_i^s(t) + Q_i^s(t+1) - Q_{s,i}^{\text{th}}, 0\right\}, \tag{16}$$

for $i\in\mathcal{N}, t\in\mathcal{T}$, where $Z_i^l(0) = Z_i^s(0) = 0$. By the definition of the two virtual queues, it is proved in [28] that constraints (3) and (4) are satisfied if the two virtual queues are mean-rate stable, i.e., $\lim_{T\to\infty}\mathbb{E}\left[Z_i^l(t)\right]/T = 0$ and $\lim_{T\to\infty}\mathbb{E}\left[Z_i^s(t)\right]/T = 0$.

In support of the problem transformation, we define the system state at time slot t as $\boldsymbol{\Theta}(t) \triangleq \left\{Q_i^l(t), Q_i^s(t), Z_i^l(t), Z_i^s(t)\right\}_{i\in\mathcal{N}}$. The Lyapunov function is then defined as a measure of the total queue backlog at time slot t as

$$\mathcal{L}\left(\boldsymbol{\Theta}(t)\right) \triangleq \frac{1}{2}\sum_{i\in\mathcal{N}}\left(Q_i^l(t)^2 + Q_i^s(t)^2 + Z_i^l(t)^2 + Z_i^s(t)^2\right), \tag{17}$$

To keep all the queues stable, the Lyapunov drift function is introduced as

$$\Delta\left(\boldsymbol{\Theta}(t)\right) \triangleq \mathbb{E}\left[\mathcal{L}\left(\boldsymbol{\Theta}(t+1)\right) - \mathcal{L}\left(\boldsymbol{\Theta}(t)\right)|\boldsymbol{\Theta}(t)\right] \tag{18}$$

To minimize the long-term average power consumption while ensuring the queue stability constraint, we define the Lyapunov-drift-plus-penalty as

$$\Delta_V\left(\boldsymbol{\Theta}(t)\right) \triangleq \Delta\left(\boldsymbol{\Theta}(t)\right) + V\mathbb{E}\left[P_{\text{sys}}(t)|\boldsymbol{\Theta}(t)\right], \tag{19}$$

where V is a positive number denoting a control parameter for the trade-off between the system's power consumption and the average queueing delay. The following theorem provides an upper bound of $\Delta_V\left(\boldsymbol{\Theta}(t)\right)$, which is crucial to the transformation of the multi-stage problem **P2** into per-time slot deterministic problems.

Theorem 1. *The drift-plus-penalty* $\Delta_V(\mathbf{\Theta}(t))$ *is bounded as*

$$\begin{aligned}\Delta_V(\mathbf{\Theta}(t)) \leq & \hat{B} - \sum_{i\in\mathcal{N}} \mathbb{E}\left[\left(Q_i^l(t)+Z_i^l(t)\right)\left(D_i^t - A_i^t\right)\middle|\mathbf{\Theta}(t)\right] \\ & - \sum_{i\in\mathcal{N}} \mathbb{E}\left[\left(Q_i^s(t)+Z_i^s(t)\right)\left(c_i^t - r_{i,UAV}^t\right)\middle|\mathbf{\Theta}(t)\right] \\ & + V\mathbb{E}\left[P_{sys}(t)\middle|\mathbf{\Theta}(t)\right] \end{aligned} \tag{20}$$

where $D_i^t = l_i^t + r_{i,UAV}^t + r_{i,mBS}^t$*;* $\hat{B}$ *consists of constant terms from the observation at the beginning of time slot* t*, thus can be put aside from the optimization of the target variables* $\mathbf{X}^t$*.*

Proof. Please refer to Appendix A. □

With support from Theorem 1, we can transform the multi-stage problem **P2** into a deterministic problem that can be solved in each time slot using the *opportunistic expectation minimization* technique [28]. Specifically, the long-term goal of **P2** can be archived by minimizing the upper bound of $\Delta_V(\mathbf{\Theta}(t))$ in each time slot, taking the system state $\mathbf{\Theta}(t)$ observed at the beginning of a time slot as the input. By removing constant terms in (20), the deterministic per-time slot problem is formulated as

$$\mathbf{P3}: \min_{\mathbf{X}^t} \quad G\left(\mathbf{X}^t\right) \tag{21a}$$
$$\text{s.t.} \quad \text{(14b), (14c'), (14d) - (14h)}$$

with the objective function as follows

$$\begin{aligned} G(\mathbf{X}^t) = & -\sum_{i\in\mathcal{N}}\left[\left(Q_i^l(t)+Z_i^l(t)\right)\left(l_i^t + r_{i,\mathrm{UAV}}^t + r_{i,\mathrm{mBS}}^t\right)\right] \\ & -\sum_{i\in\mathcal{N}}\left[\left(Q_i^s(t)+Z_i^s(t)\right)\left(c_i^t - r_{i,j}^t\right)\right] \\ & + V\left(\psi_c p_c(t) + \sum_{i\in\mathcal{N}}\psi_i\left(p_{l,i}(t) + p_{\mathrm{Tx},i}(t)\right)\right) \end{aligned} \tag{22}$$

It is worth mentioning that solving **P3** does not require any future information about incoming tasks and wireless channel state other than the current state of the task queues, making the approach an online optimization design. In the next section, we will introduce a DRL-based online optimization algorithm to solve **P3** efficiently.

V. Deep Reinforcement Learning-based Online Resource Management

We propose a DRL-based resource management (DRLRM) scheme for solving the per-time slot problem **P3**. The proposed framework, as depicted in Fig.2, consists of three main modules, which are (1) the actor module, (2) the critic module, and (3) the policy update module. The actor module obtains necessary information for optimization from the system observer and adopts a DNN to output several potential decisions for channel assignment, $\tilde{\mathbf{x}}^t = \left\{x_{i,j}^t\right\}_{i\in\mathcal{N},j\in\mathcal{S}}$. The critic module evaluates each decision made by the actor module by solving remaining variables $\mathbf{y}^t = \left\{\alpha_{i,j}^t, r_{i,j}^t, f_{l,i}^t, f_{c,i}^t\right\}_{i\in\mathcal{N},j\in\mathcal{S}}$ using model-based optimization. The policy update module logs a history of the system state-optimal decision pairs on the fly and re-trains the actor module's DNN in a periodic manner so that the mapping policy will be updated and adaptive to the time-varying channel condition. In the following, we describe in detail the learning-based framework depicted in Fig. 2, followed by the model-based optimization of the critic module.

For ease of convenience, the input and output of the actor module are denoted as follows. The input is denoted by $\mathbf{\Xi}^t = \left\{h_{i,j}^t, Q_i^l(t), Q_i^s(t), Z_i^l(t), Z_i^s(t)\right\}_{i\in\mathcal{N},j\in\mathcal{S}}$, which includes the channel gain and the queue length of all physical and virtual queues for each user. The module's output is presented by $\left\{\tilde{\mathbf{x}}_m^t\right\}_{m=1}^k$, where k denotes the number of potential channel assignment decisions made.

Remark 1. *The above approach is backed by the Tammer decomposition technique [37], where combinatorial variables in* $\tilde{\mathbf{x}}^t$ *are decoupled from continuous variables in* $\mathbf{y}^t$*. Indeed, by temporarily fixing* $\tilde{\mathbf{x}}^t$*, we can further decompose* ***P3*** *into several sub-problems with separate objectives and constraints.*

Remark 2. *To obtain the optimal decision for* $\tilde{\mathbf{x}}^t$*, an exhaustive search requires evaluating*

$$k_{\max} = \binom{N}{\chi_{UAV}^{\max}}\binom{N}{\chi_{mBS}^{\max}} \tag{23}$$

possible channel assignment decisions, where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ *indicates the binomial coefficient of selecting an (unordered) subset of* k *elements from a fixed set of* n *elements. Since* $\binom{n}{k}$ *is in* $O(n^k)$*, an exhaustive search over all possible channel assignments is with complexity* $O\left(N^{\chi_{UAV}^{\max}+\chi_{mBS}^{\max}}\right)$*. In the following sub-section, we propose to use a DNN to find* $(\tilde{\mathbf{x}}^t)^\star$ *with much less computational complexity. The DNN will be trained periodically to dynamically adapt to the time-varying channel condition and approximate an optimal policy* $\Pi_t^\star$ *that maps the current system state to an optimal channel assignment decision in time slot* t*,* $\Pi_t^\star : \mathbf{\Xi}^t \mapsto (\tilde{\mathbf{x}}^t)^\star$*.*

A. Outline of the Proposed DRLRM Framework

1) Model-free Actor Module: The actor module consists of a DNN and an action quantizer. The DNN obtains $\mathbf{\Xi}^t$ and processes the forward propagation to output a relaxed channel assignment $\hat{\mathbf{x}}^t = \left\{\hat{x}_{i,j}^t \in [0,1]\right\}_{i\in\mathcal{N},j\in\mathcal{S}}$, which will be later quantized into a number of potential channel assignment decisions. We adopt a Deep Neural Network (DNN) to represent the channel assignment policy (an approximation to the optimal policy $\Pi_t^\star$) as $\Pi_{\Phi_t} : \mathbf{\Xi}^t \mapsto \hat{\mathbf{x}}^t$, where Φ_t denotes the DNN's parameters at time slot t. To ensure that $\hat{x}_{i,j}^t \in [0;1]$ for all $i \in \mathcal{N}, j \in \mathcal{S}$, we use the sigmoid activation function at the output layer of the DNN.

The action quantizer of the actor module will then obtain the relaxed channel assignment $\hat{\mathbf{x}}^t$ to generate a batch of k potential decisions. Let Γ denote the quantizer policy, we have $\Gamma : \hat{\mathbf{x}}^t \mapsto \tilde{\mathbf{x}}^t$, where $\tilde{\mathbf{x}}^t = \left\{\tilde{x}_{i,j}^t \in \{0,1\}\right\}_{i\in\mathcal{N},j\in\mathcal{S}}$ denotes the output channel assignment decision. Let ς_j^t denote the $\chi_j^{\max}$-th largest element of $\hat{\mathbf{x}}_j^t \triangleq \left\{\hat{x}_{i,j}^t \in \{0,1\}\right\}_{i\in\mathcal{N}}$ (the set of relaxed channel assignments corresponding to the server j).

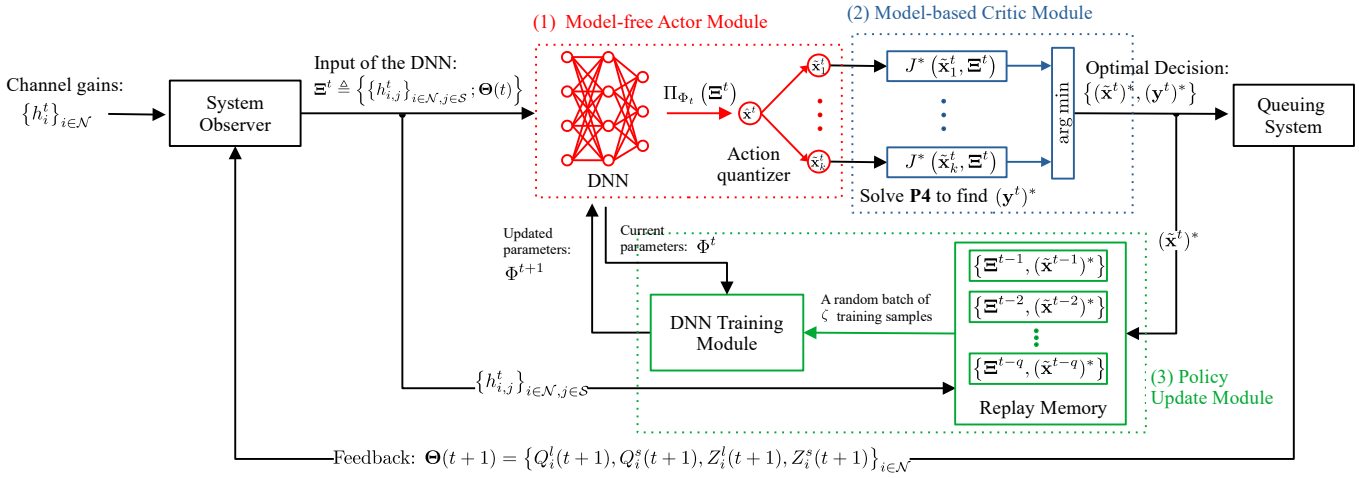


Fig. 2: Schematic of the proposed DRLRM framework

Then, for each $\hat{x}_{i,j}^t$ in $\hat{\mathbf{x}}^t$, the quantization policy Γ generates a corresponding value of $\tilde{x}_{i,j}^t$ in $\tilde{\mathbf{x}}^t$ as

$$\tilde{x}_{i,j}^t = \begin{cases} 1, & \text{if } \hat{x}_{i,j}^t \geq \varsigma_j^t, \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

To generate the first decision $(\tilde{\mathbf{x}}_1^t)$, we apply the policy Γ directly to the output of the DNN, i.e., $\hat{\mathbf{x}}^t$. The remaining $(k-1)$ decisions are generated by applying the policy Γ to noise-added versions of $\hat{\mathbf{x}}^t$, denoted as Sigmoid $(\hat{\mathbf{x}}^t + \mathbf{n})$, where Sigmoid$(\cdot)$ denotes the element-wise sigmoid function to ensure each element of the output vector falls within the range $(0, 1)$. Here, $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$ is a $2N$-dimensional zero-mean random vector following the normal (Gaussian) distribution with a diagonal covariance matrix $\sigma_n^2 \mathbf{I}$; $\mathbf{I}$ denotes the identity matrix. The variable σ_n is a hyper-parameter responsible for the balance between exploration and exploitation of the action quantizer. Too relaxed values of σ_n might not take advantage of the DNN output to predict the optimal channel assignment. In contrast, too strict values of σ_n might hamper the critic module from extracting good approximations of the per-time slot problem's global minimum in each time slot. In the long term, this phenomenon causes the DNN to experience difficulties in learning the optimal channel assignment policy due to the noisy labels extracted by the critic module.

2) Model-based Critic Module: The model-based critic module obtains the set of potential channel assignment decisions from the actor module to select the best decision among them and solve the optimization problem for the remaining variables. Unlike the conventional approach that adopts another DNN for the critic module, our approach leverages the model information on the user-server communication and power consumption to evaluate each channel assignment decision analytically. Indeed, by fixing the setting for $\mathbf{x}_j^t, j \in \mathcal{S}$, it is feasible to find optimal settings for the remaining variables in $\mathbf{y}^t$, which are all continuous. Specifically, let $(\mathbf{y}^t)^\star$ denote the optimal decision for $\mathbf{y}^t$ and $J^\star\left(\tilde{\mathbf{x}}^t, \Xi^t\right)$ denote the optimal value of the objective function (21a) given $\tilde{\mathbf{x}}^t$ and Ξ^t, **P3** is equivalent to the problem, denoted as **P4**, of finding

$$(\tilde{\mathbf{x}}^t)^\star \triangleq \arg\min J^\star\left(\tilde{\mathbf{x}}^t, \Xi^t\right), \tag{25}$$

where $\tilde{\mathbf{x}}^t$ is one among k channel assignment decisions given by the actor module. We will introduce in detail the algorithm to obtain $J^\star\left(\tilde{\mathbf{x}}^t, \Xi^t\right)$ in Section V-B.

Using a model-based critic module brings the advantage of having an accurate evaluation for each decision on the channel assignment, thus improving the convergence of training. Besides, it is worth noting that to obtain $(\tilde{\mathbf{x}}^t)^\star$ and $(\mathbf{y}^t)^\star$, we need to evaluate k times the function $J^\star\left(\tilde{\mathbf{x}}^t, \Xi^t\right)$. Thus, k is another hyper-parameter of the system that will affect the trade-off between performance and computational complexity. In general, larger values of k result in a better performance in terms of convergence time but require more computational resources.

3) Policy Update Module: The policy update module exploits training samples labeled by the critic module (i.e., the pair $\left\{\Xi^t, (\tilde{\mathbf{x}}^t)^\star\right\}$) to update the parameters of the actor module's DNN. A replay memory of size q is adopted to record the training samples. Only the most recent data samples are kept, i.e., new data will continuously replace the old ones to avoid memory bloat. Beginning with an empty memory, we start training the DNN only when at least $q/2$ data samples are available. Afterward, the DNN is trained periodically once every δ_T time slots. Such a training scheme helps prevent the DNN from overfitting with noise in the input and enables the neural network to adapt dynamically to the time-varying channel condition.

Specifically, a batch of training samples is randomly selected from the replay memory when $\mod(t, \delta_T) = 0$ ($\mod$ indicates the modulo operation). We then use these samples to train the DNN by using the Adam algorithm [38] to minimize the cross-entropy cost function $L(\Phi_t)$, given as

$$L(\Phi_t) = \frac{-1}{|\mathcal{S}^t|} \sum_{\tau \in \mathcal{S}^t} \Big[(\tilde{\mathbf{x}}^\tau)^\mathsf{T} \log\left(\Pi_{\Phi^t}\left(\Xi^\tau\right)\right) + (1 - \tilde{\mathbf{x}}^\tau)^\mathsf{T} \log\left(1 - \Pi_{\Phi^t}\left(\Xi^\tau\right)\right) \Big]. \tag{26}$$

In (26), $\mathcal{S}^t$ denotes the set of time indices of data samples selected for training at time t; $|\mathcal{S}^t|$ denotes the size of $\mathcal{S}^t$; $(\cdot)^\mathsf{T}$ denotes the transpose operator; and $\log(\cdot)$ denotes the element-wise logarithm operation of a vector. The cost function in (26) measures the goodness of the relaxed channel assignment (i.e., the output of the DNN) compared to the best decision selected by the critic module. Lower values of $L(\Phi_t)$ indicate good performance and that Φ_t is appropriate in generalizing the mapping rule for channel assignment.

B. Model-based Optimization of the Critic Module

In this section, we present in detail the optimization algorithm used by the critic module to obtain $J^\star\left(\tilde{\mathbf{x}}^t, \Xi^t\right)$ in (25). For a given the channel assignment decision $\tilde{\mathbf{x}}^t = \left\{x^t_{i,j}\right\}_{i\in\mathcal{N}, j\in\mathcal{S}}$, we can obtain the optimal solution for $\mathbf{y}^t = \left\{\alpha^t_{i,j}, r^t_{i,j}, f^t_{l,i}, f^t_{c,i}\right\}_{i\in\mathcal{N}, j\in\mathcal{S}}$ by decomposing the per-time slot problem **P3** into four sub-problems, including optimization for offloading volume and bandwidth allocation for the UAV and the mBS links, optimization for local computation, and optimization for UAV's computational resource scheduling.

1) Optimization on Offloading Volume and Bandwidth Allocation for the UAV link: Given a feasible channel assignment decision, the optimization variables related to the UAV includes the bandwidth allocation, $\boldsymbol{\alpha}^t_{\text{UAV}} \triangleq \left\{\alpha^t_{i,j} | i \in \mathcal{N}^t_{\text{UAV}}, j = \text{UAV}\right\}$, and the offloading volume on the UAV link, $\mathbf{r}^t_{\text{UAV}} \triangleq \left\{r^t_{i,j} | i \in \mathcal{N}^t_{\text{UAV}}, j = \text{UAV}\right\}$. Let $\mathbf{x}^t_{\text{UAV}} \triangleq \left\{\boldsymbol{\alpha}^t_{\text{UAV}}, \mathbf{r}^t_{\text{UAV}}\right\}$ denote the combination of these variables. The optimal decision for $\mathbf{x}^t_{\text{UAV}}$ can be obtained by solving the following problem, hereinafter referred to as **P3.1**:

$$\begin{aligned}
\min_{\mathbf{x}^t_{\text{UAV}}} \quad & -\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \left[\left(Q^l_i(t) + Z^l_i(t) - Q^s_i(t) - Z^s_i(t)\right) r^t_{i,\text{UAV}}\right] \\
& + V \sum_{i\in\mathcal{N}^t_{\text{UAV}}} \psi_i \left[\left(2^{\frac{r^t_{i,\text{UAV}}}{W_{\text{UAV}}\alpha^t_{i,\text{UAV}}\tau}} - 1\right)\frac{N_0 W_{\text{UAV}}}{h^t_{i,\text{UAV}}}\right] \quad (27a)\\
\text{s.t.} \quad & \epsilon_A \le \alpha^t_{i,\text{UAV}}, \sum_{i\in\mathcal{N}^t_{\text{UAV}}} \alpha^t_{i,\text{UAV}} \le 1, i \in \mathcal{N}^t_{\text{UAV}}, \quad (27b)\\
& 0 \le r^t_{i,\text{UAV}} \le \min\left\{Q^l_i(t), r^{t,\max}_{i,\text{UAV}}\right\}, i \in \mathcal{N}^t_{\text{UAV}} \quad (27c)
\end{aligned}$$

where $r^{t,\max}_{i,\text{UAV}} = W_{\text{UAV}}\alpha^t_{i,\text{UAV}}\tau\log_2\left(1 + \frac{p^{\max}_{\text{Tx},i}h^t_{i,\text{UAV}}}{N_0 W_{\text{UAV}}}\right)$ denotes the upper bound of $r^t_{i,\text{UAV}}$ using the maximum transmit power. Note that for all users that are not associated with the UAV in time slot t, the bandwidth and offloading volume assigned to them equal zero, i.e., $r^{t,\max}_{i,\text{UAV}} = 0, \alpha^t_{i,\text{UAV}} = 0, \forall i \notin \mathcal{N}^t_{\text{UAV}}$.

To solve **P3.1**, we adopt the Gauss-Seidel approach [39] to optimize the offloading volume and the bandwidth allocation in an alternating manner. Specifically, in each iteration, the optimal offloading volume decision is obtained in closed forms, and the optimal bandwidth allocation is determined by the *Lagrangian method*. The alternating approach is guaranteed to converge to the optimal solution since **P3.1** is convex, and the feasible region is a Cartesian product [39] of $\mathbf{r}^t_{\text{UAV}}$ and $\boldsymbol{\alpha}^t_{\text{UAV}}$.

a) Optimal offloading volume: For a feasible bandwidth allocation $\boldsymbol{\alpha}^t_{\text{UAV}}$, the optimal offloading volume for mobile devices in $\mathcal{N}^t_{\text{UAV}}$ can be obtained by solving

$$\begin{aligned}
\textbf{P3.1.1}: \quad & \underset{\mathbf{r}^t_{\text{UAV}}}{\text{minimize}} \quad (27a)\\
& \text{subject to} \quad (27c)
\end{aligned}$$

The optimal solution to the above problem is either the stationary point of (27a) or one of the boundary points. Specifically, for $i \in \mathcal{N}^t_{\text{UAV}}$, $(r^t_{i,\text{UAV}})^\star = 0$ if $Q^l_i(t) + Z^l_i(t) \le Q^s_i(t) + Z^s_i(t)$, otherwise, $(r^t_{i,\text{UAV}})^\star = \max\left\{\min\left\{\hat{r}^t_{i,\text{UAV}}, r^{t,\max}_{i,\text{UAV}}\right\}, 0\right\}$, where $\hat{r}^t_{i,\text{UAV}} = W_{\text{UAV}}\alpha^t_{i,\text{UAV}}\tau \times \log_2\left(\frac{\left(Q^l_i(t)+Z^l_i(t)-Q^s_i(t)-Z^s_i(t)\right)\alpha^t_{i,\text{UAV}}\tau h^t_{i,\text{UAV}}}{V\psi_i\ln(2)N_0}\right)$.

b) Optimal bandwidth allocation: For a feasible offloading volume decision $\mathbf{r}^t_i$, the optimal bandwidth allocation can be obtained by solving

$$\begin{aligned}
\textbf{P3.1.2}: \quad & \min_{\boldsymbol{\alpha}^t_{\text{UAV}}} \quad \sum_{i\in\mathcal{N}^t_{\text{UAV}}} \psi_i p^{\text{UAV}}_{\text{Tx},i}(t) \quad (29a)\\
& \text{s.t.} \quad \epsilon_A \le \alpha^t_{i,\text{UAV}}, \sum_{i\in\mathcal{N}^t_{\text{UAV}}} \alpha^t_{i,\text{UAV}} \le 1, \quad (29b)\\
& \quad p^{\text{UAV}}_{\text{Tx},i}(t) \le p^{\max}_{\text{Tx},i}, i \in \mathcal{N}^t_{\text{UAV}} \quad (29c)
\end{aligned}$$

where $p^{\text{UAV}}_{\text{Tx},i}(t) = \left(2^{\frac{r^t_{i,\text{UAV}}}{W_{\text{UAV}}\alpha^t_{i,\text{UAV}}\tau}} - 1\right)\frac{N_0 W_{\text{UAV}}}{h^t_{i,\text{UAV}}}$ denotes the transmit power of user i on the UAV link. First, we observe that (29c) is equivalent to $\alpha^t_{i,\text{UAV}} \ge \alpha^{t,\min}_{i,\text{UAV}}$, where

$$\alpha^{t,\min}_{i,\text{UAV}} \triangleq \frac{r^t_{i,\text{UAV}}}{W_{\text{UAV}}\alpha^t_{i,\text{UAV}}\tau\log_2\left(1 + \frac{p^{\max}_{\text{Tx},i}h^t_{i,\text{UAV}}}{N_0 W_{\text{UAV}}}\right)} \quad (30)$$

Then, the partial Lagrangian function associated with the above problem can be written as

$$\begin{aligned}
\mathcal{L}(\alpha^t_{i,\text{UAV}}, \lambda^t_{\text{UAV}}) \triangleq & \sum_{i\in\mathcal{N}^t_{\text{UAV}}} \psi_i \left[\left(2^{\frac{r^t_{i,\text{UAV}}}{W_{\text{UAV}}\alpha^t_{i,\text{UAV}}\tau}} - 1\right)\frac{N_0 W_{\text{UAV}}}{h^t_{i,\text{UAV}}}\right] \\
& + \lambda^t_{\text{UAV}}\left(\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \alpha^t_{i,\text{UAV}} - 1\right), \quad (31)
\end{aligned}$$

where $\lambda^t_{\text{UAV}} \ge 0$ is the Lagrangian multiplier associated with the constraint $\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \alpha^t_{i,\text{UAV}} \le 1$. Based on the *Karush-Kuhn-Tucker* (KKT) condition, the optimal bandwidth allocation $(\boldsymbol{\alpha}^t_{\text{UAV}})^\star$ and the optimal Lagrangian multiplier $(\lambda^t_{\text{UAV}})^\star$ should satisfy the following equation set

$$\begin{cases}
(\alpha^t_{i,\text{UAV}})^\star = \max\left\{\hat{\epsilon}_A, \mathcal{R}_{i,\text{UAV}}((\lambda^t_{\text{UAV}})^\star)\right\}, i \in \mathcal{N}^t_{\text{UAV}}, \\
\sum_{i\in\mathcal{N}^t_{\text{UAV}}} (\alpha^t_{i,\text{UAV}})^\star = 1,
\end{cases} \quad (32)$$

where $\hat{\epsilon}_A = \max\left\{\epsilon_A, \alpha^{t,\min}_{i,\text{UAV}}\right\}$ and $\mathcal{R}_{i,\text{UAV}}(\lambda^t_{\text{UAV}})$ denotes the root of $\frac{\delta}{\delta\alpha^t_{i,\text{UAV}}}\mathcal{L}(\alpha^t_{i,\text{UAV}}, \lambda^t_{\text{UAV}}) = 0$. The following proposition provides a close-form expression for $\mathcal{R}_{i,\text{UAV}}(\lambda^t_{\text{UAV}})$.

Algorithm 1 Lagrangian method for **P3.1.2**

Initialization : $\xi = 10^{-7}$, $\tilde{\lambda}_L = \lambda^L_{\text{UAV}}(t)$, $\tilde{\lambda}_U = \lambda^U_{\text{UAV}}(t)$, $l = 0$, $I_{\max} = 200$, $\epsilon_A = 10^{-4}$, $\alpha^t_{i,\text{UAV}} = \hat{\epsilon}_A, i \in \mathcal{N}^t_{\text{UAV}}$
while $\left|\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \max\left\{\hat{\epsilon}_A, \mathcal{R}_{i,\text{UAV}}(\lambda^U_{\text{UAV}}(t))\right\} - 1\right| > \xi$ and $i \le I_{\max}$ **do**
  $\tilde{\lambda} = \frac{1}{2}\left(\tilde{\lambda}_L + \tilde{\lambda}_R\right)$ and $l = l + 1$
  Set $\alpha^t_{i,\text{UAV}} = \max\left\{\hat{\epsilon}_A, \mathcal{R}_{i,\text{UAV}}(\tilde{\lambda})\right\}, i \in \mathcal{N}^t_{\text{UAV}}$
  if $\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \alpha^t_{i,\text{UAV}} \le 1$ **then**
    $\tilde{\lambda}_L = \tilde{\lambda}$
  else
    $\tilde{\lambda}_U = \tilde{\lambda}$
  end if
end while

Proposition 1. *Given* $\lambda^t_{UAV} > 0$, *the root of* $\frac{\delta}{\delta\alpha^t_{i,UAV}}\mathcal{L}(\alpha^t_{i,UAV}, \lambda^t_{UAV}) = 0$ *is positive and unique as*

$$\mathcal{R}_{i,UAV}(\lambda^t_{UAV}) = \frac{r^t_{i,UAV}\ln(2)}{2W_{UAV}\tau\mathcal{W}\left(\frac{r^t_{i,UAV}\ln(2)}{2W_{UAV}\tau}\sqrt{\frac{\lambda^t_{UAV}h^t_{i,UAV}W_{UAV}\tau}{\psi_i\sigma^2_{UAV}r^t_{i,UAV}\ln(2)}}\right)}, \tag{33}$$

where $\mathcal{W}(\cdot)$ *denotes the Lambert-W function.*

Proof. Please refer to Appendix B □

It is observed that $\frac{\delta}{\delta\alpha^t_{i,\text{UAV}}}\mathcal{L}(\alpha^t_{i,\text{UAV}}, \lambda^t_{\text{UAV}})$ is a monotonic function with response to $\alpha^t_{i,\text{UAV}}$. Thus, the optimal Lagrangian multiplier $\left(\lambda^t_{\text{UAV}}\right)^\star$ can be found using a bisection search over $\left[\lambda^L_{\text{UAV}}(t); \lambda^U_{\text{UAV}}(t)\right]$ in which $\lambda^L_{\text{UAV}}(t)$ and $\lambda^U_{\text{UAV}}(t)$ are selected so that $\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \max\left\{\hat{\epsilon}_A, \mathcal{R}_{i,\text{UAV}}(\lambda^L_{\text{UAV}}(t))\right\} > 1$ and $\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \max\left\{\hat{\epsilon}_A, \mathcal{R}_{i,\text{UAV}}(\lambda^U_{\text{UAV}}(t))\right\} < 1$. Specifically, the two boundaries can be selected as follows

$$\begin{cases} \lambda^L_{\text{UAV}}(t) = \max\limits_{i\in\mathcal{N}^t_{\text{UAV}}} \left.\frac{-\delta p^{\text{UAV}}_{\text{Tx},i}(t)}{\delta\alpha^t_{i,\text{UAV}}}\right|_{\alpha^t_{i,\text{UAV}}=1}, \\ \lambda^U_{\text{UAV}}(t) = \min\limits_{i\in\mathcal{N}^t_{\text{UAV}}} \left.\frac{-\delta p^{\text{UAV}}_{\text{Tx},i}(t)}{\delta\alpha^t_{i,\text{UAV}}}\right|_{\alpha^t_{i,\text{UAV}}=\hat{\epsilon}_A}, \end{cases} \tag{34}$$

The searching process for $\left(\lambda^t_{\text{UAV}}\right)^\star$ can terminate when $|\sum_{i\in\mathcal{N}^t_{\text{UAV}}} \max\left\{\hat{\epsilon}_A, \mathcal{R}_{i,\text{UAV}}(\lambda^U_{\text{UAV}}(t))\right\} - 1| \le \xi$, where ξ is the accuracy of the algorithm. Details of the the Lagrangian method for solving **P3.1.2** are summarized in Algorithm 1.

2) Optimization on Offloading Volume and Bandwidth Allocation for the mBS link: Similar to the UAV link, given a feasible channel assignment decision, the optimization variables related to the mBS include the bandwidth allocation, $\boldsymbol{\alpha}^t_{\text{mBS}} \triangleq \left\{\alpha^t_{i,j} | i \in \mathcal{N}^t_{\text{mBS}}, j = \text{mBS}\right\}$, and the offloading volume on the mBS link, $\mathbf{r}^t_{\text{mBS}} \triangleq \left\{r^t_{i,j} | i \in \mathcal{N}^t_{\text{mBS}}, j = \text{mBS}\right\}$. Denoted by $\mathbf{x}^t_{\text{mBS}} \triangleq \left\{\boldsymbol{\alpha}^t_{\text{mBS}}, \mathbf{r}^t_{\text{mBS}}\right\}$ the combination of these variables, the optimal decisions for the mBS link can be obtained by solving

$$\begin{aligned} \textbf{P3.2}: \min_{\mathbf{x}^t_{\text{mBS}}} \quad & -\sum_{i\in\mathcal{N}^t_{\text{mBS}}} \left[\left(Q^l_i(t) + Z^l_i(t) - \left(r^t_{i,\text{UAV}}\right)^\star\right) r^t_{i,\text{mBS}}\right] \\ & + V\sum_{i\in\mathcal{N}^t_{\text{mBS}}} \psi_i \left[\left(2^{\frac{r^t_{i,\text{mBS}}}{W_{\text{mBS}}\alpha^t_{i,\text{mBS}}\tau}} - 1\right)\frac{N_0 W_{\text{mBS}}}{h^t_{i,\text{mBS}}}\right] \quad (35a) \\ \text{s.t.} \quad & \epsilon_A \le \alpha^t_{i,\text{mBS}}, \sum_{i\in\mathcal{N}^t_{\text{mBS}}} \alpha^t_{i,\text{mBS}} \le 1, \quad (35b) \\ & 0 \le r^t_{i,\text{mBS}} \le \min\left\{Q^l_i(t) - \left(r^t_{i,\text{UAV}}\right)^\star, r^{t,\max}_{i,\text{mBS}}\right\}, \\ & i \in \mathcal{N}^t_{\text{mBS}} \quad (35c) \end{aligned}$$

where $r^{t,\max}_{i,\text{mBS}} = W_{\text{mBS}}\alpha^t_{i,\text{mBS}}\tau\log_2\left(1 + \frac{p^{\max}_{\text{Tx},i}h^t_{i,\text{mBS}}}{N_0 W_{\text{mBS}}}\right)$ and the right-hand side (RHS) of (35c) denotes the upper bound of $r^t_{i,\text{mBS}}$ at time slot t. Similar to **P3.1**, we adopt the Gauss-Seidel method to solve $\boldsymbol{\alpha}^t_{\text{mBS}}$ and $\mathbf{r}^t_{\text{mBS}}$ of **P3.2** in an alternating manner, in which the optimal offloading volume is given in a close-form expression and the bandwidth allocation is determined by the Lagrangian method.

a) Optimal offloading volume: For a feasible bandwidth allocation $\boldsymbol{\alpha}^t_{\text{mBS}}$, the optimal offloading volume for mobile devices in $\mathcal{N}^t_{\text{mBS}}$ can be obtained by solving

$$\begin{aligned} \textbf{P3.2.1}: \min_{\mathbf{r}^t_{\text{mBS}}} \quad & (35a) \\ \text{subject to} \quad & (35c) \end{aligned}$$

The optimal solution to the above problem is either the stationary point of (35a) or one of the boundary points. Specifically, for $i \in \mathcal{N}^t_{\text{mBS}}$, $(r^t_{i,\text{mBS}})^\star = \max\left\{\min\left\{\hat{r}^t_{i,\text{mBS}}, \text{RHS of (35c)}\right\}, 0\right\}$, where $\hat{r}^t_{i,\text{mBS}} = W_{\text{mBS}}\alpha^t_{i,\text{mBS}}\tau \times \log_2\left(\frac{\left(Q^l_i(t) + Z^l_i(t) - \left(r^t_{i,\text{mBS}}\right)^\star\right)\alpha^t_{i,\text{mBS}}\tau h^t_{i,\text{mBS}}}{V\psi_i\ln(2)N_0}\right)$.

b) Optimal bandwidth allocation: For a feasible offloading volume decision, the optimal bandwidth allocation on the mBS link can be obtained by solving the problem

$$\begin{aligned} \textbf{P3.2.2}: \min_{\boldsymbol{\alpha}^t_{\text{mBS}}} \quad & \sum_{i\in\mathcal{N}^t_{\text{mBS}}} \psi_i p^{\text{mBS}}_{\text{Tx},i}(t) \quad (37a) \\ \text{s.t.} \quad & \epsilon_A \le \alpha^t_{i,\text{mBS}}, \sum_{i\in\mathcal{N}^t_{\text{mBS}}} \alpha^t_{i,\text{mBS}} \le 1, \quad (37b) \\ & p^{\text{mBS}}_{\text{Tx},i}(t) \le p^{\max}_{\text{Tx},i}, i \in \mathcal{N}^t_{\text{mBS}} \quad (37c) \end{aligned}$$

We observe that **P3.2.2** is similar to **P3.1.2**, in which one is for the SeNB, the other is for the MeNB. Thus, we can adopt Algorithm 1 to solve **P3.2.2**; the procedure is exactly the same by replacing $j = \text{UAV}$ with $j = \text{mBS}$.

3) Optimization on User's Local Computation: Given the optimal decision on offloading volume of each user, the problem of resource allocation for local computation can be decomposed for each individual $f^t_{l,i}$:

$$\begin{aligned} \textbf{P3.3}: \min_{\mathbf{f}^t_l} \quad & \sum_{i\in\mathcal{N}}\left(-Q^l_i(t)\tau f^t_{l,i}L^{-1}_i + V\psi_i\kappa_i\left(f^t_{l,i}\right)^3\right) \quad (38a) \\ \text{s.t.} \quad & 0 \le f^t_{l,i} \le f^{\max}_i, i \in \mathcal{N}, \quad (38b) \\ & l^t_i \le Q^l_i(t) - (r^t_{i,\text{UAV}})^\star - (r^t_{i,\text{mBS}})^\star, i \in \mathcal{N} \\ & \quad (38c) \end{aligned}$$

in which constraint (38c) is added to ensure (14g) of the original problem **P1**. First, we observe that **P3.3** is a convex

problem since its objective function (38a) is convex and all constraints are linear. Furthermore, since both the objective function and constraints of **P3.3** can be decomposed for each $f_{l,i}^t$, the optimization for $\mathbf{f}_l^t$ can be done by solving each $f_{l,i}^t$ separately. Specifically, the optimal solution to **P3.3** is either at the stationary point of the objective function (38a) or one of the boundary points as $(f_{l,i}^t)^\star = \min\left\{F_i, \sqrt{\frac{Q_i^l(t)\tau}{3\kappa_i V \psi_i L_i}}\right\}, i \in \mathcal{N}$, where $F_i = \min\left\{f_i^{\max}, \left(Q_i^l(t) - (r_{i,\text{UAV}}^t)^\star - (r_{i,\text{mBS}}^t)^\star\right) L_i/\tau\right\}$.

4) Optimization on UAV's computational resource scheduling: The optimal $\mathbf{f_c}^\star(t)$ can be obtained by solving

$$\mathbf{P3.4}: \min_{\mathbf{f_c}(t)} \quad -\sum_{i\in\mathcal{N}} Q_i^s(t)\tau f_{c,i}^t/L_s + V\psi_c \sum_{i\in\mathcal{N}} \kappa_s \left(f_{c,i}^t\right)^3 \tag{39a}$$

$$\text{s.t.} \quad 0 \le f_{c,i}^t \le f_c^{\max}, i \in \mathcal{N}, \tag{39b}$$

$$c_i^t = \tau f_{c,i}^t/L_s \le Q_i^s(t), i \in \mathcal{N} \tag{39c}$$

We observe that similar to **P3.3**, **P3.4** is convex and the optimization can be decomposed into sub-problems that involve $f_{c,i}^t$ separately. Specifically, the optimal solution for each $f_{c,i}^t$ is either at the stationary point of the objective function or one of the boundary points as $(f_{c,i}^t)^\star = \min\left\{f_{c,i}^{\max}, \sqrt{\frac{Q_i^s(t)\tau}{3\kappa_s V \psi_c L_s}}\right\}, i \in \mathcal{N}$, where $f_{c,i}^{\max} = \min\left\{f_c^{\max}, Q_i^s(t)L_s/\tau\right\}$.

VI. Algorithm Analysis

In this section, we provide the computational complexity of the proposed scheme in solving the per-time slot problem, followed by an analysis of the scheme's optimality.

A. Computational Complexity

The calculation mainly comes from the model-based critic module because, in each time slot, this module must examine k potential channel assignment decisions made by the actor module to select the best one. Importantly, examining one decision involves solving the optimization of computation and communication of all users and servers. The time complexity in each problem is as follows.

Complexity of joint optimization on offloading volume and bandwidth allocation. Given feasible offloading volumes, Algorithm 1 requires $O\left(\log_2(N/\xi)\right)$ iterations via a bi-section search to find the optimal bandwidth allocation, where N denotes the number of users and ξ specifies the algorithm accuracy (e.g., $\xi = 10^{-4}$). Given a feasible bandwidth allocation, the optimization on the offloading volume for each user can be obtained in closed forms; thus, the complexity is in $O(N)$ considering N users. Consequently, the Gauss-Seidel joint optimization with $I_{\max}$ iterations maximum is with complexity $O\left(I_{\max}\left(N + \log_2(N/\xi)\right)\right)$.

Complexity of optimization on the computation of the user and the UAV. Since solutions to **P3.3** and **P3.4** can both be obtained in closed forms, the optimization is with complexity $O(1)$ for each user.

In summary, the computational complexity for evaluating one channel assignment decision is $O\left(I_{\max}\left(N + \log_2(N/\xi)\right)\right)$. Since the critic module examines k potential decisions in each time slot, the overall computational complexity of the proposed scheme is in $O\left(kI_{\max}\left(N + \log_2(N/\xi)\right)\right)$ with fast execution.[1]

B. Optimality Analysis

We assume that the traffic arrival and channel gain fluctuation for each user is an independent and identically distributed (i.i.d) random process, denoted as $\omega(t) = \left\{h_{i,j}^t, A_i^t\right\}_{i\in\mathcal{N}, j\in\mathcal{S}}$. A policy that observes $\omega(t)$ in each time slot and makes control decisions independent of the queue backlog is referred to as an ω-only policy.

To ensure the strong stability of queues, we assume that the following assumption holds, which is the Slater condition for Lyapunov optimization [28]. The asymptotic optimality of the proposed DRLRM scheme is then provided in Theorem 2.

Assumption 1. *(Slater Condition) There are values $\epsilon > 0$ and $\Phi(\epsilon)$ (where $0 \le \Phi(\epsilon) \le P_{sys}^{\max}$) and an ω-only policy Π making control decision $\alpha^{\Pi,t}$ in time slot t that satisfies*

$$\begin{aligned}\mathbb{E}\left[P_{sys}(t)\middle|\alpha^{\Pi,t}\right] &= \Phi(\epsilon),\\ \mathbb{E}\left[A_i^t\right] &\le \mathbb{E}\left[D_i^t\middle|\alpha^{\Pi,t}\right] - \epsilon, \forall i \in \mathcal{N}.\\ \mathbb{E}\left[r_{i,UAV}^t\middle|\alpha^{\Pi,t}\right] &\le \mathbb{E}\left[c_i^t\middle|\alpha^{\Pi,t}\right] - \epsilon, \forall i \in \mathcal{N}\end{aligned} \tag{40}$$

Theorem 2. *Suppose that the $\omega(t)$ process is i.i.d over time slots, **P1** is feasible, and the Slater condition holds for some ϵ and $\Phi(\epsilon)$. Suppose that the proposed DRLRM algorithm produces a C-additive approximation ($C \ge 0$) of the minimum of* (22) *every time slot, then the following statements hold.*

(a) The average system power consumption satisfies

$$\lim_{t\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[P_{sys}(t)\right] \le P_{sys}^* + \frac{\hat{B}+C}{V}, \tag{41}$$

where P_{sys}^* is the minimum average power cost achievable by any policy that meets the required constraints.

(b) All queues $Q_i^l(t)$, $Q_i^s(t)$, $Z_i^l(t)$, $Z_i^s(t)$ are mean rate stable and QoE constraints (3) *and* (4) *are satisfied.*

Proof. Please refer to Appendix C □

VII. Numerical Results and Discussion

In this section, simulation results are provided to evaluate the proposed scheme's performance. In simulation, we consider $N = 8$ mobile devices placed randomly around a hot spot within a radius of 100 meters. The MeNB (i.e., the mBS) is 500 meters away from the users while the SeNB (i.e., the UAV) flies above them at a fixed altitude of 50 meters. Each MEC server serves at most two users at a time. The small-scale fading channel power gains in dB, $\tilde{h}_{i,j}^t$

[1] Given settings in Section VII, it is demonstrated that the running time of the proposed DRLRM scheme for one-shot decision is milliseconds, using Python and Tensorflow on a PC with Intel Core i7 2.9 GHz CPU and Nvidia GeForce RTX 3070 GPU. In this regard, the MEC server can output the one-slot decision within the time frame duration (e.g., 10 ms in LTE).

TABLE III: Details of two scenarios for performance evaluation

	Scenario I (Energy Efficiency Comparison)	Scenario II (Stress Test for Queue Stability)
Objective	Investigate **energy efficiency**, given a strict constraint on thresholds at which the average queue length must be less than or equal to.	Relax constraints on the queue length threshold, investigate the **ability to stabilize queues** under very high computational traffic.
Arrival rate	**Reasonable**	**Very high**
Focused KPI	**Power consumption** How much energy is consumed to satisfy the predefined queue length threshold constraint?	**Queue stability** Given that the arrival rate approximates the maximum processing capability, at which level are queues stable?

[dB], are normally distributed with mean $\mathbb{E}[\tilde{h}^t_{i,j}[\text{dB}]] = 0$ and standard variance $\sigma_{\tilde{h}^t_{i,j}[\text{dB}]} = 4$. The task arrival in each time slot follows the Poisson distribution with the same mean for all users $\mathbb{E}[A^t_i] = \lambda = 15$ kbits, unless otherwise stated. Other parameters include $\kappa = 10^{-28}$ Ws^3/cycle^3, $f^{\max}_c = 1$ GHz, $\psi_c = 0,1$, $N_0 = -174$ dBm/Hz, $g_0 = -50$ dB, $\gamma_{\text{UAV}} = \gamma_{\text{mBS}} = 2.7601$. $L_i = L_s = 737.5$ cycles/bit, $\psi_i = 1$, $p^{\max}_{\text{Tx},i} = 20$ dBm, $f^{\max}_i = 0.5$ GHz, $i \in \mathcal{N}$. Each simulation is conducted over a time duration of $T = 10000$ with slot length $\tau = 10$ milliseconds.

For the proposed method, we set $\sigma_n = 0.25$ and $k = 16$ to balance the exploration and exploitation for the actor module. The DNN consists of three 1-D convolutional layers, followed by a Flatten layer and three Dense layers; all are implemented using Tensorflow. The ReLU (rectified linear unit) activation is used for all layers except the last one with the sigmoid function.

Benchmarking schemes: To evaluate the performance, we use the following four benchmark schemes:

- *Exhaustive Channel Assignment and Joint Resource Allocation (**Exhaustive**)*: The network investigate all possible decisions to select the best channel assignment.
- *Queue Length-based Greedy Channel Assignment and Joint Resource Allocation (**QL-JRA**)*: The network prioritizes users having longest virtual queues $Z^l_i(t)$ to assign communication channels.
- *Channel Gain-based Greedy Channel Assignment and Joint Resource Allocation (**CG-JRA**)*: Users with highest channel gains are prioritized to obtain communications resources.
- *Random Channel Assignment Policy and Joint Resource Allocation (**RA-JRA**)*: The network randomly assigns users to the mBS and the UAV.

These benchmark schemes differentiate in the approach to solving $\mathbf{x}^t$ for channel assignment. However, all apply the critic module's optimization procedure specified in section V-B to optimize the remaining variables in $\mathbf{y}^t$.

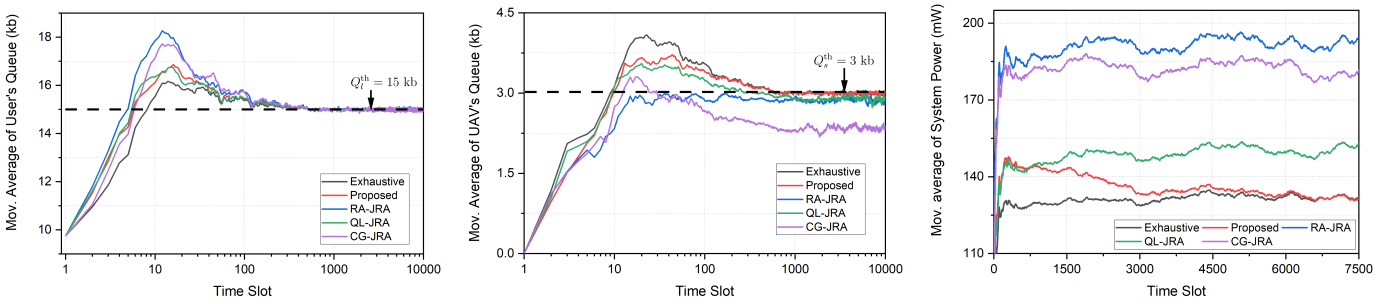
It is noteworthy that the exhaustive method is considered the optimal solution for benchmarking. However, finding the optimal solution via exhaustive search is not feasible if we consider a system with many users.

Performance evaluation scenarios: To evaluate the degree of suboptimality and convergence, we compare the proposed DRLRM method with the benchmark schemes in the two scenarios described in Table III.

- *Scenario I*: Power consumption is considered the main KPI for performance evaluation since all methods can satisfy the queue stability constraints given a reasonable computational load. The arrival rate is set at a reasonable level, $\lambda = 10$ kb. Accordingly, the queue length thresholds are $Q^{\text{th}}_l = 15$ kb and $Q^{\text{th}}_s = 3$ kb. For the proposed DRLRM method, the number of generated actions is set at $k = 16$ (i.e., 2% of the search space).
- *Scenario II*: A very high computational load with $\lambda = 30$ kb is set to demonstrate high-traffic periods. All methods are expected to run at the highest energy level virtually all the time to stabilize queues. Thus, power consumption is not our focus. The queue length threshold is relaxed (i.e., $Q^{\text{th}}_l = Q^{\text{th}}_s = \infty$) to facilitate the optimization. For the proposed method, $k = 80$ is set (approximately 10% of the search space).

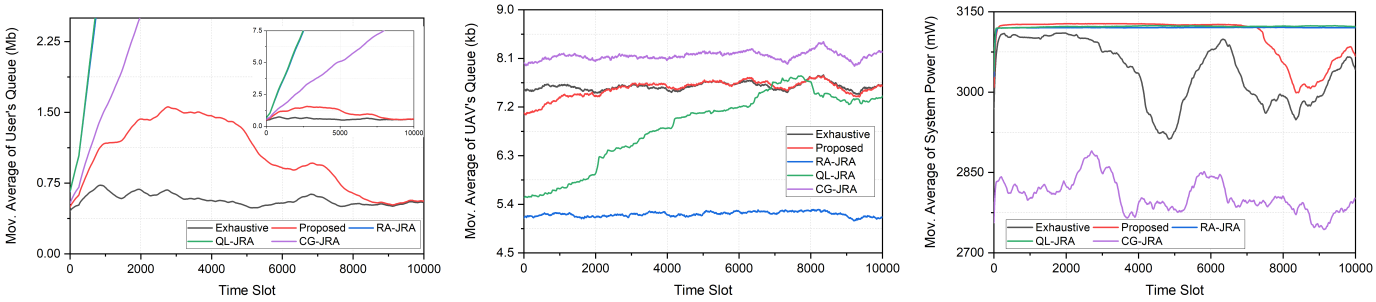
Performance in Scenario I: In Fig. 3, we compare all methods in Scenario I, where we focus on the power consumption given predefined queue threshold constraints. Each point in the figure is a moving average of 1500 time slots. From Fig. 3a and Fig. 3b, we observe that thanks to the critic module's optimization, all methods can keep the user's and the UAV's queues stable at levels lower than or equal to the predefined thresholds (i.e., queue threshold constraints are satisfied). From Fig. 3c, we observe a power consumption reduction for the proposed scheme over time. In the early stage, the proposed method consumes much more power than the optimal channel assignment (as much as the QL-JRA method). In the later phase, the scheme's power consumption gently decreases over time and eventually converges to the same level as if the optimal decision was selected. This result proves the effectiveness of the training when the DNN gradually learns from experience to mimic the optimal policy. With the considered settings, our proposed method provides a reduction of 13.4%, 26.6%, and 30.5% compared to the QL-JRA, CH-JRA, and RA-JRA schemes, respectively.

Performance in Scenario II: Figure 4 illustrates the performance of all schemes in Scenario II, where we focus on the queue stability KPI under very high traffic. Each point in the figure is a moving average of 1500 time slots. We observe that while the UAV's queue is kept stable by all methods, only the optimal channel assignment and the proposed scheme can cause local queues of users to be stable. Specifically, the user's backlog queue of the QL-JRA, CH-JRA, and RA-JRA schemes increases almost linearly over time, indicating that the queue is not stable. This is because, without a proper policy for channel assignment, the achievable task processing capability is very limited and cannot afford the given task arrival rate. In contrast, the proposed method's user queue length also increases rapidly in the early stage but decreases gradually in the later phase. In the end, the scheme's



(a) User's queue length (b) UAV's queue length (c) Weighted-sum system power

Fig. 3: Performance in Scenario I (energy efficiency examination with constraints on the stable level of queues), $\lambda = 10$ kb, $Q_l^{\text{th}} = 15$ kb, $Q_s^{\text{th}} = 3$ kb, $k = 16$, $V = 10^{10}$.



(a) User's queue length (b) UAV's queue length (c) Weighted-sum system power

Fig. 4: Performance in Scenario II (Stress test for queue stability with relaxed queue stable thresholds), $\lambda = 30$ kb, $Q_l^{\text{th}} = Q_s^{\text{th}} = \infty$, $k = 80$, $V = 10^{10}$.

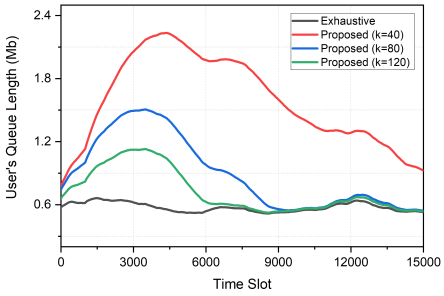
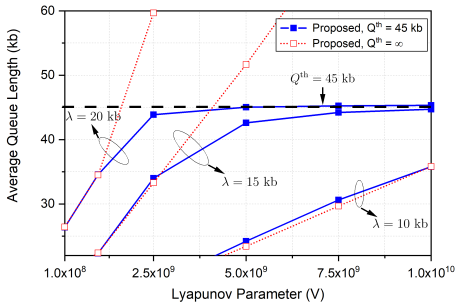


Fig. 5: Convergence behavior of the proposed scheme in Scenario II, $\lambda = 30$ kb, $Q_l^{\text{th}} = Q_s^{\text{th}} = \infty$, $V = 10^{10}$.

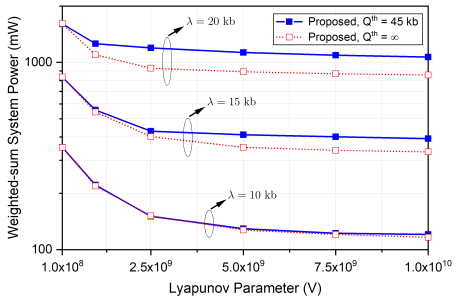
user queue length is almost the same as that of the optimal channel assignment. This result once again demonstrates the effectiveness and convergence performance of the proposed DRLRM framework, even in unfavorable circumstances with a very high computational workload.

Effect of the parameter k: In Fig. 5, we investigate the convergence behavior of the proposed DRLRM method under different settings of the hyper-parameter k in Scenario II. The moving average rolling over 2000 time slots of the user queue length is plotted. The number of potential actions of the actor module (k) is set at 40, 80, and 120 for evaluation. The figure clearly shows that the hyperparameter plays a critical role in the convergence speed of the proposed method. Higher values of k help the DNN learn the optimal channel assignment policy faster. They speed up the convergence at the cost of higher computational resources required for the critic module to investigate the generated potential decisions. Specifically, the time duration until convergence of the proposed method (within a 1% gap compared to the optimal decision) for $k = 120$ and $k = 80$ are 6000 and 8000 time slots, respectively. This is because, with more decisions generated, the critic module has more chance to extract good approximations of the global minimum of the per-time slot problem. In other words, by investigating more potential decisions, the critic module helps improve the quality of the training data. In the long term, this advantage speeds up the learning process.

Effect of the queue length threshold: Fig. 6a illustrates the impact of the queue threshold on the average system queue length when changing V. It is noteworthy that the queue length threshold can be referred to as a means of controlling the service delay since the average task execution delay is proportional to the average queue length. We observe that if the queue stability level is not constrained, the average queue length increases almost linearly with the Lyapunov parameter. In addition, a higher task arrival rate leads to an increase in the average queue length. The interesting point is when we add additional constraints (3) and (4) to enforce the stability level for the queue length. We notice that with small values of V, the queue length threshold constraint does not affect the optimization. When V increases, the average queue length is effectively controlled so that the constraint is satisfied for all considered task arrival rates. Note that in case $\lambda = 10$ kb, the



(a) Queue length



(b) System power

Fig. 6: Effect of the queue length threshold constraints

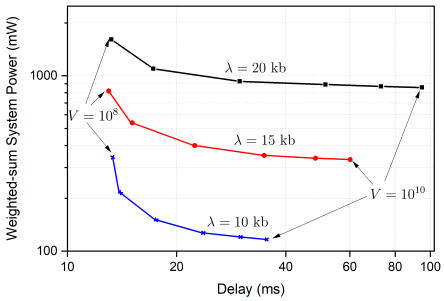


Fig. 7: Power consumption vs. task completion delay, $Q_l^{\text{th}} = Q_s^{\text{th}} = \infty$, V=10^8, 10^9, $\{2.5, 5, 7.5\} \times 10^9$, 10^{10}.

queue stability level with and without constraints are almost the same since the setting of V is not large enough to make the average queue length surpass the threshold.

Fig. 6b investigates the impact of the queue length threshold on the system power consumption when changing V. We observe that the power consumption of all settings decreases, corresponding to the increase of the parameter V, as expected. This is because increasing V puts more emphasis on the system power consumption than stabilizing the queue length in the per-time slot problem. In addition, we notice that as the arrival rate increases, the power consumption increases accordingly. The queue threshold constraint also significantly impacts power consumption. The gap between the two scenarios (with and without the constraint) tends to enlarge with increasing the arrival rate and control parameter V. This is because the network has no choice but to consume more energy to maintain queues stable at a satisfactory level (as depicted in Fig. 6a), especially in cases with high arrival rates.

Power-delay trade-off In Fig. 7, we investigate the trade-off between the average weighted sum system power consumption and the average task computation delay by varying the Lyapunov control parameter V with unconstrained queue thresholds. As can be observed, the average task computation delay increases as the power consumption decreases, indicating that a proper setting of V is critical to balance the two objectives. Besides, we observe that given a specific execution delay level, the average weighted sum power consumption increases with the task arrival rate. The result is logical since more power consumption is required to keep the queue stable when the workload grows.

VIII. Conclusion

This paper proposes a hybrid method that combines conventional model-based optimization and model-free DRL to minimize the power consumption of a multi-user, multi-server MEC network. Via DC, one user can offload tasks to the macro base station and the UAV-mounted MEC server simultaneously. The power minimization is formulated as a multi-stage MINLP problem with long-term constraints of queue stability and average task computation delay. Lyapunov optimization is exploited to transform the original multi-stage problem into a per-time slot problem, which is then solved using a DRL framework. Theoretical analyses are provided to demonstrate the proposed method's optimality and computational complexity. Extensive simulations show that the proposed framework can produce nearly the same performance as the optimal solution obtained via an exhaustive search. In future research, we will investigate the impact on system performance of the cooperation between the UAV-mounted edge server and the macro base station and the ability to deploy multiple UAVs.

Appendix A
Proof of Theorem 1

To begin with, let $\mathbf{Q}^l(t) \triangleq \left\{Q_i^l(t)\right\}_{i\in\mathcal{N}}, \mathbf{Q}^s(t) \triangleq \left\{Q_i^s(t)\right\}_{i\in\mathcal{N}}, \mathbf{Z}^l(t) \triangleq \left\{Z_i^l(t)\right\}_{i\in\mathcal{N}}$, and $\mathbf{Z}^s(t) \triangleq \left\{Z_i^s(t)\right\}_{i\in\mathcal{N}}$ denote the system state variables. We then define the Lyapunov function $\mathcal{L}(\cdot)$ and the drift function $\Delta(\cdot)$ for $\mathbf{Q}^l(t)$, $\mathbf{Q}^s(t)$, $\mathbf{Z}^l(t)$, and $\mathbf{Z}^s(t)$ in a similar way as for $\mathbf{\Theta}(t)$ in (17) and (18), respectively, where the conditional expectation is taken given $\mathbf{\Theta}(t)$. The following lemmas give the upper bound of the drift function for each of the above system state. Note that in what follows, $[x]^+$ denotes the function $\max\{x, 0\}$. Additionally, since the communication and computation resources at the user and the UAV are limited, we denote the upper bound of D_i^t, r_i^t and c_i^t as $D_{i,\max}$, $r_{i,\max}$ and $c_{i,\max}$, respectively. Similarly, $A_{i,\max}$ denotes the maximal task arrival for user i in a time slot.

Lemma 1. *The drift function for $\mathbf{Q}^l(t)$ is bounded as*

$$\Delta\left(\mathbf{Q}^l(t)\right) \le B_1 - \sum_{i\in\mathcal{N}} \mathbb{E}\left[Q_i^l(t)\left(l_i^t + r_i^t - A_i^t\right)\middle|\mathbf{\Theta}(t)\right], \quad (42)$$

where $B_1 = \frac{1}{2}\sum_{i\in\mathcal{N}}\left(D_{i,\max}^2 + A_{i,\max}^2\right)$, $r_i^t = r_{i,UAV}^t + r_{i,mBS}^t$.

Proof. We have $Q_i^l(t+1)^2 = \left(\left[Q_i^l(t) - D_i^t\right]^+ + A_i^t\right)^2 \overset{(\dagger)}{=} \left(Q_i^l(t) - D_i^t + A_i^t\right)^2 = Q_i^l(t)^2 - 2Q_i^l(t)(D_i^t - A_i^t) + (D_i^t -$

$A_i^t)^2$, where (†) is obtained since $D_i^t = l_i^t + r_i^t \le Q_i^l(t)$ as in (14g). By some algebraic transformations, we have $\frac{1}{2}\left(Q_i^l(t+1)^2 - Q_i^l(t)^2\right) \le -Q_i^l(t)(D_i^t - A_i^t) + \frac{1}{2}(D_i^t - A_i^t)^2$ $\le -Q_i^l(t)(D_i^t - A_i^t) + \frac{1}{2}\left(D_{i,\max}^2 + A_{i,\max}^2\right)$ By taking the conditional expectation on both sides of the inequation and summing up over all users $i \in \mathcal{N}$, we obtain (42). □

Lemma 2. *The drift function $\Delta\left(\mathbf{Z}^l(t)\right)$ is upper bounded by*

$$B_2 - \sum_{i\in\mathcal{N}} \mathbb{E}\left[Z_i^l(t)\left(l_i^t + r_i^t - Q_i^l(t) - A_i^t + Q_{l,i}^{th}\right)\middle|\mathbf{\Theta}(t)\right], \quad (43)$$

where $B_2 = \frac{1}{2}\sum_{i\in\mathcal{N}}\left[D_{i,\max}^2 + Q_i^l(t)^2 + (A_i^t)^2 + (Q_{l,i}^{th})^2 + D_{i,\max}Q_{l,i}^{th} + Q_i^l(t)A_i^t\right]$ and $r_i^t = r_{i,UAV}^t + r_{i,mBS}^t$.

Proof. From (15), we have

$$\begin{aligned} Z_i^l(t+1)^2 &\overset{(\dagger)}{\le} \left(Z_i^l(t) + Q_i^l(t+1) - Q_{l,i}^{\text{th}}\right)^2 \\ &\overset{(\ddagger)}{=} \left(Z_i^l(t) + Q_i^l(t) - D_i^t + A_i^t - Q_{l,i}^{\text{th}}\right)^2 \\ &= Z_i^l(t)^2 - 2Z_i^l(t)\left(D_i^t - Q_i^l(t) - A_i^t + Q_{l,i}^{\text{th}}\right) \\ &\quad + \left(D_i^t - Q_i^l(t) - A_i^t + Q_{l,i}^{\text{th}}\right)^2 \end{aligned} \quad (44)$$

where (†) is due to the fact that $\left(\max\{a-b,0\}\right)^2 \le (a-b)^2$; and (‡) is with condition (14g) of **P1** that $D_i^l = l_i^t + r_i^t \le Q_i^l(t)$. Thus, we have $\frac{1}{2}\left(Z_i^l(t+1)^2 - Z_i^l(t)^2\right) \overset{(\dagger)}{\le} -Z_i^l(t)\left(D_i^t - Q_i^l(t) - A_i^t + Q_{l,i}^{\text{th}}\right) + \frac{1}{2}\Big[D_{i,\max}^2 + Q_i^l(t)^2 + (A_i^t)^2 + (Q_{l,i}^{\text{th}})^2 + D_{i,\max}Q_{l,i}^{\text{th}} + Q_i^l(t)A_i^t\Big]$ where (†) is obtained since $(a-b)^2 \le a^2 + b^2$ for $ab \ge 0$ and $0 \le D_i^t \le D_{i,\max}$. By replacing D_i^t by $l_i^t + r_i^t$, taking the conditional expectation on both sides of the inequation and summing up over all users $i \in \mathcal{N}$, we obtain (43). □

Lemma 3. *The drift function for $\mathbf{Q}^s(t)$ is bounded as*

$$\Delta\left(\mathbf{Q}^s(t)\right) \le B_3 - \sum_{i\in\mathcal{N}} \mathbb{E}\left[Q_i^s(t)\left(c_i^t - r_{i,UAV}^t\right)\middle|\mathbf{\Theta}(t)\right] \quad (45)$$

where $B_3 = \frac{1}{2}\sum_{i\in\mathcal{N}}\left(c_{i,\max}^2 + r_{i,\max}^2\right)$.

Proof. The proof is similar to that of Lemma 1. □

Lemma 4. *The drift function $\Delta\left(\mathbf{Z}^s(t)\right)$ is upper bounded by*

$$B_4 - \sum_{i\in\mathcal{N}} \mathbb{E}\left[Z_i^s(t)\left(c_i^t - Q_i^s(t) - r_{i,UAV}^t + Q_{s,i}^{th}\right)\middle|\mathbf{\Theta}(t)\right] \quad (46)$$

where $B_4 = \frac{1}{2}\sum_{i\in\mathcal{N}}\left[c_{i,\max}^2 + Q_i^s(t)^2 + r_{i,\max}^2 + (Q_{s,i}^{th})^2 + c_{i,\max}Q_{s,i}^{th} + Q_i^s(t)r_{i,\max}\right]$.

Proof. The proof is similar to that of Lemma 2. □

It is straightforward that $\Delta\left(\mathbf{\Theta}(t)\right) = \Delta\left(\mathbf{Q}^l(t)\right) + \Delta\left(\mathbf{Q}^s(t)\right) + \Delta\left(\mathbf{Z}^l(t)\right) + \Delta\left(\mathbf{Z}^s(t)\right)$. Thus, by summing up the left hand sides of (42), (43), (45), (46) we obtain the upper bound of the drift-plus-penalty as in (20), where $\hat{B} = B_1 + B_2 + B_3 + B_4 + \sum_{i\in\mathcal{N}}\left[Z_i^l(t)\left(Q_i^l(t) - Q_{l,i}^{\text{th}}\right) + Z_i^s(t)\left(Q_i^s(t) - Q_{s,i}^{\text{th}}\right)\right]$. We observe that $\hat{B}$ consists of constant terms from the observation at the beginning of time slot t, thus can be put aside from the optimization of the target control variables $\mathbf{X}^t$.

Appendix B
Proof of Proposition 1

Let $\mathcal{L}'$ denote the derivative of $\mathcal{L}(\alpha_{i,\text{UAV}}^t, \lambda_{\text{UAV}}^t)$ with response to $\alpha_{i,\text{UAV}}^t$. we have $\mathcal{L}' = \frac{\delta}{\delta\alpha_{i,\text{UAV}}^t}\mathcal{L}(\alpha_{i,\text{UAV}}^t, \lambda_{\text{UAV}}^t) = -\frac{\psi_i\sigma_j^2 r_{i,\text{UAV}}^t \ln(2)}{h_{i,j}^t W_j \tau (\alpha_{i,\text{UAV}}^t)^2}\exp\left(\frac{r_{i,\text{UAV}}^t\ln(2)}{W_j\alpha_{i,\text{UAV}}^t\tau}\right) + \lambda_{\text{UAV}}^t$. The first term of $\mathcal{L}'$, denoted as $F(\alpha_{i,\text{UAV}}^t)$, is an increasing function with response to $\alpha_{i,\text{UAV}}^t$. Furthermore, we have $\lim_{\alpha_{i,\text{UAV}}^t\to 0^+} F(\alpha_{i,\text{UAV}}^t) = -\infty$ and $\lim_{\alpha_{i,\text{UAV}}^t\to+\infty} F(\alpha_{i,\text{UAV}}^t) = 0$. Thus, given $\lambda_{\text{UAV}}^t > 0$, the root of equation $\mathcal{L}' = 0$ is positive and unique. Solving $\mathcal{L}' = 0$, we have $\frac{1}{\left(\alpha_{i,\text{UAV}}^t\right)^2}\exp\left(\frac{r_{i,\text{UAV}}^t\ln(2)}{W_{\text{UAV}}\tau}\frac{1}{\alpha_{i,\text{UAV}}^t}\right) = \frac{\lambda_{\text{UAV}}^t h_{i,j}^t W_{\text{UAV}}\tau}{\psi_i\sigma_j^2 r_{i,\text{UAV}}^t\ln(2)}$. We recognize that the equation has the form $x^2\exp(ax) = c$, where $x = \frac{1}{\alpha_{i,\text{UAV}}^t}$, $a = \frac{r_{i,\text{UAV}}^t\ln(2)}{W_j\tau}$, $c = \frac{\lambda_{\text{UAV}}^t h_{i,j}^t W_j\tau}{\psi_i\sigma_j^2 r_{i,\text{UAV}}^t\ln(2)}$. Solving the equation using the Lambert-W function yields $\alpha_{i,\text{UAV}}^t = \frac{a}{2\mathcal{W}\left(\frac{a\sqrt{c}}{2}\right)}$. Substituting a and c, we obtain (33).

Appendix C
Proof of Theorem 2

To begin with, we introduce the following lemma for feasibility of problem **P1**.

Lemma 5. *Suppose that $\omega(t)$ is stationary. If **P1** is feasible, then for any $\delta > 0$, there is an ω-only policy Π that satisfies*

$$\begin{aligned} \mathbb{E}\left[P_{sys}(t)\middle|\alpha^{\Pi,t}\right] &\le P_{sys}^* + \delta \\ \mathbb{E}\left[A_i^t\right] &\le \mathbb{E}\left[D_i^t\middle|\alpha^{\Pi,t}\right] + \delta, \forall i \in \mathcal{N} \\ \mathbb{E}\left[r_{i,UAV}^t\middle|\alpha^{\Pi,t}\right] &\le \mathbb{E}\left[c_i^t\middle|\alpha^{\Pi,t}\right] + \delta, \forall i \in \mathcal{N} \end{aligned} \quad (47)$$

Proof. Please see Theorem 4.5 of [28] for detailed proof. □

Proof of statement (a). Considering an ω-only policy Π with a corresponding value $\delta > 0$, applying Lemma 5 into the right-hand side (RHS) of (20) yields

$$\begin{aligned} &\Delta\left(\mathbf{\Theta}(t)\right) + V\mathbb{E}\left[P_{\text{sys}}(t)\right] \\ &\overset{(\dagger)}{\le} \hat{B} + C - \sum_{i\in\mathcal{N}}\mathbb{E}\left[\left(Q_i^l(t) + Z_i^l(t)\right)\left(D_i^t - A_i^t\right)\right] \\ &\quad - \sum_{i\in\mathcal{N}}\mathbb{E}\left[\left(Q_i^s(t) + Z_i^s(t)\right)\left(c_i^t - r_{i,\text{UAV}}^t\right)\right] + V\mathbb{E}\left[P_{\text{sys}}(t)\right] \\ &\overset{(\ddagger)}{\le} \hat{B} + C + \delta\sum_{i\in\mathcal{N}}\left(Q_i^l(t) + Z_i^l(t) + Q_i^s(t) + Z_i^s(t)\right) + V\left(P_{\text{sys}}^* + \delta\right), \end{aligned} \quad (48)$$

where (†) is because the ω-only policy Π is independent of the queue backlog $\mathbf{\Theta}(t)$ and $\omega(t)$ is i.i.d over time; (‡) is obtained by plugging in (47). By letting $\delta \to 0$, we obtain $\Delta\left(\mathbf{\Theta}(t)\right) + V\mathbb{E}\left[P_{\text{sys}}(t)\right] \le \hat{B} + C + VP_{\text{sys}}^*$. By summing up both sides from $t = 0$ to $T-1$, taking iterated expectation and telescoping sums, then dividing both sides by TV, we obtain $\frac{1}{TV}\mathbb{E}\left[\mathcal{L}(\mathbf{\Theta}(T))\right] - \frac{1}{TV}\mathbb{E}\left[\mathcal{L}(\mathbf{\Theta}(0))\right] + \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[P_{\text{sys}}(t)\right]$ $\le (\hat{B} + C)/V + P_{\text{sys}}^*$. Taking the limit on both sides of the inequation as $T \to \infty$, we obtain (41). This concludes the proof of statement (a).

Proof of statement (b). We consider an ω-only policy Π that satisfies the Slater condition in Assumption 1. Plugging (40) into (†) of (48), we obtain $\Delta\left(\mathbf{\Theta}(t)\right) + V\mathbb{E}\left[P_{\text{sys}}(t)\right] \le$

$\hat{B} + C - \epsilon \sum_{i\in\mathcal{N}} \left(Q_i^l(t) + Z_i^l(t) + Q_i^s(t) + Z_i^s(t)\right) + V\Phi(\epsilon)$. By taking integrated expectations, summing the telescoping series, and rearranging terms, we obtain $\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}\left(Q_i^l(t) + Z_i^l(t) + Q_i^s(t) + Z_i^s(t)\right) \leq \frac{\hat{B}+C}{\epsilon} - \frac{V}{\epsilon}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[P_{\text{sys}}(t)\right] - \Phi(\epsilon)\right) + \frac{\mathbb{E}[\mathcal{L}(\Phi(0)]}{\epsilon T}$. By taking the limit of both sides of the inequation as $T \to \infty$ and considering the fact $\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[P_{\text{sys}}(t)\right] \leq P_{\text{sys}}^*$, we obtain $\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}\left(Q_i^l(t) + Z_i^l(t) + Q_i^s(t) + Z_i^s(t)\right) \leq \frac{1}{\epsilon}\left(\hat{B} + C - V(P_{\text{sys}}^* - \Phi(\epsilon))\right) < \infty$. The inequation indicates that all queues $Q_i^l(t)$, $Z_i^l(t)$, $Q_i^s(t)$, $Z_i^s(t)$ are strongly stable, which also implies mean rate stability (Theorem 2.8 of [28]). In addition, since the two virtual queues $Z_i^l(t)$, $Z_i^s(t)$ are mean-rate stable, the QoE constraints (3) and (4) are satisfied. This conludes the proof of statement (b).

REFERENCES

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tutor.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[2] Q.-V. Pham *et al.*, "A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art," *IEEE Access*, vol. 8, pp. 116 974–117 017, 2020.

[3] ——, "Aerial Computing: A New Computing Paradigm, Applications, and Challenges," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8339–8363, 2022.

[4] F. Guo *et al.*, "An Efficient Computation Offloading Management Scheme in the Densely Deployed Small Cell Networks With Mobile Edge Computing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2651–2664, 2018.

[5] Y. Dai *et al.*, "Joint Computation Offloading and User Association in Multi-Task Mobile Edge Computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12 313–12 325, 2018.

[6] S. Jošilo and G. Dán, "Computation Offloading Scheduling for Periodic Tasks in Mobile Edge Computing," *IEEE/ACM Trans. Netw.*, vol. 28, no. 2, pp. 667–680, 2020.

[7] H. Tout *et al.*, "Multi-Persona Mobility: Joint Cost-Effective and Resource-Aware Mobile-Edge Computation Offloading," *IEEE/ACM Trans. Netw.*, vol. 29, no. 3, pp. 1408–1421, 2021.

[8] M. Mozaffari *et al.*, "A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 3, pp. 2334–2360, 2019.

[9] B. Li, Z. Fei, and Y. Zhang, "UAV Communications for 5G and Beyond: Recent Advances and Future Trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, 2019.

[10] X. Pang *et al.*, "When UAV Meets IRS: Expanding Air-Ground Networks via Passive Reflection," *IEEE Wirel. Commun.*, vol. 28, no. 5, pp. 164–170, 2021.

[11] W. Xu, Y. Sun, R. Zou, W. Liang, Q. Xia, F. Shan, T. Wang, X. Jia, and Z. Li, "Throughput Maximization of UAV Networks," *IEEE/ACM Trans. Netw.*, vol. 30, no. 2, pp. 881–895, 2022.

[12] M. Agiwal *et al.*, "A Survey on 4G-5G Dual Connectivity: Road to 5G Implementation," *IEEE Access*, vol. 9, pp. 16 193–16 210, 2021.

[13] Y. Li *et al.*, "Hybrid NOMA-FDMA Assisted Dual Computation Offloading: A Latency Minimization Approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3345–3360, 2022.

[14] C. Li, H. Wang, and R. Song, "Intelligent Offloading for NOMA-Assisted MEC via Dual Connectivity," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2802–2813, 2021.

[15] H. Guo, J. Liu, J. Zhang, W. Sun, and N. Kato, "Mobile-Edge Computation Offloading for Ultradense IoT Networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4977–4988, 2018.

[16] H. Guo, J. Liu, and J. Zhang, "Computation Offloading for Multi-Access Mobile Edge Computing in Ultra-Dense Networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 14–19, 2018.

[17] Q.-V. Pham *et al.*, "Whale Optimization Algorithm With Applications to Resource Allocation in Wireless Networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4285–4297, 2020.

[18] H.-G. T. Pham *et al.*, "Joint Task Offloading and Resource Management in NOMA-Based MEC Systems: A Swarm Intelligence Approach," *IEEE Access*, vol. 8, pp. 190 463–190 474, 2020.

[19] T. Q. Dinh *et al.*, "Offloading in Mobile Edge Computing: Task Allocation and Computational Frequency Scaling," *IEEE Trans. Commun*, vol. 65, no. 8, pp. 3571–3584, 2017.

[20] T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, 2019.

[21] H. Hu, W. Song, Q. Wang, R. Q. Hu, and H. Zhu, "Energy Efficiency and Delay Tradeoff in an MEC-Enabled Mobile IoT Network," *IEEE Internet Things J.*, pp. 1–1, 2022.

[22] P. Wei *et al.*, "Reinforcement Learning-Empowered Mobile Edge Computing for 6G Edge Intelligence," *IEEE Access*, vol. 10, pp. 65 156–65 192, 2022.

[23] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, 2020.

[24] Y.-C. Wu *et al.*, "A Hybrid DQN and Optimization Approach for Strategy and Resource Allocation in MEC Networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 7, pp. 4282–4295, 2021.

[25] S. Bi, L. Huang, H. Wang, and Y.-J. A. Zhang, "Lyapunov-Guided Deep Reinforcement Learning for Stable Online Computation Offloading in Mobile-Edge Computing Networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7519–7537, 2021.

[26] X. Li *et al.*, "An Integrated Optimization-Learning Framework for Online Combinatorial Computation Offloading in MEC Networks," *IEEE Wirel. Commun.*, vol. 29, no. 1, pp. 170–177, 2022.

[27] Y. He *et al.*, "Deep-Reinforcement-Learning-Based Optimization for Cache-Enabled Opportunistic Interference Alignment Wireless Networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10 433–10 445, 2017.

[28] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

[29] M. Min *et al.*, "Learning-Based Computation Offloading for IoT Devices With Energy Harvesting," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, 2019.

[30] H. Liu and G. Cao, "Deep Reinforcement Learning-Based Server Selection for Mobile Edge Computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13 351–13 363, 2021.

[31] K. Guo *et al.*, "Online Learning Based Computation Offloading in MEC Systems With Communication and Computation Dynamics," *IEEE Trans. Commun*, vol. 69, no. 2, pp. 1147–1162, 2021.

[32] L. T. Hoang, C. T. Nguyen, P. Li, and A. T. Pham, "Joint Uplink and Downlink Resource Allocation for UAV-enabled MEC Networks under User Mobility," in *2022 IEEE Int. Conf. Commun. Workshops (ICC Workshops 2022) Proc.*, 2022, pp. 1059–1064.

[33] S. M. Ross, *Introduction to Probability Models*, 12th ed. Academic Press, 2019.

[34] Y. Mao *et al.*, "Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, 2017.

[35] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 13, no. 2, pp. 203–221, Aug 1996, 00477.

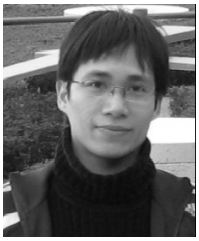
[36] J. Zhang *et al.*, "Stochastic Computation Offloading and Trajectory Scheduling for UAV-Assisted Mobile Edge Computing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3688–3699, 2019.

[37] K. Tammer, "The application of parametric optimization and imbedding to the foundation and realization of a generalized primal decomposition approach," 1987.

[38] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[39] L. Grippo and M. Sciandrone, "On the convergence of the block nonlinear Gauss–Seidel method under convex constraints," *Operations Research Letters*, vol. 26, no. 3, pp. 127–136, 2000.

Linh T. Hoang (S'20) received the B.E. degree in electronics and telecommunications engineering from Hanoi University of Science and Technology, Vietnam, in 2018 and the M.S. degree in computer science and engineering from the University of Aizu, Japan, in 2021, where he is working toward the Ph.D. degree. His study in Japan is funded by the Japanese Government Scholarship (MonbuKagakusho). His research interests include unmanned aerial vehicle-aided networks, medium access control techniques, resource management in mobile edge computing, and machine learning in wireless communications.



Chuyen T. Nguyen received the B.E. degree in electronics and telecommunications from Hanoi University of Science and Technology (HUST), Vietnam, in 2006, the M.S. degree in communications engineering from the National Tsing Hua University, Taiwan, in 2008, and the Ph.D. degree in informatics from Kyoto University, Japan, in 2013. From September to November 2014, he was a Visiting Researcher with The University of Aizu, Japan. He is currently an Associate Professor with the School of Electrical and Electronic Engineering (SEEE), HUST. His current research interests include in areas of MAC protocol design and reliable transmission in wireless/optical networks. He received the Fellow Award from Hitachi Global Foundation, in August 2016; the First Best Paper Award in 2019 IEEE ICT; and the Best Paper Awards in 2018 KICS/IEEE ICTC and 2019 IEEE ICC.

Anh T. Pham received the B.E. and M.E. degrees in electronics engineering from Hanoi University of Technology, Vietnam, in 1997 and 2000, respectively, and the Ph.D. degree in information and mathematical sciences from Saitama University, Japan, in 2005. From 1998 to 2002, he was with NTT Corporation, Vietnam. Since 2005, he has been a Faculty Member with The University of Aizu, where he is currently a Professor and the Head of the Computer Communications Laboratory, Division of Computer Engineering. He has authored/coauthored over 200 peer-reviewed articles on these topics. His research interests include the broad areas of communication theory and networking with a particular emphasis on modeling, design, and performance evaluation of wired/wireless communication systems and networks. He is a member of IEICE and OSA.