IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 June 2022, accepted 6 July 2022, date of publication 11 July 2022, date of current version 15 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3189682

RESEARCH ARTICLE

Distributed Location-Aware Task Offloading in Multi-UAVs Enabled Edge Computing

JIANHUA LIU, ZIBO WU, JIAJIA LIU, AND XIAOGUANG TU
Institute of Electronic and Electrical Engineering, Civil Aviation Flight University of China, Guanghan, Deyang, Sichuan 618307, China

Corresponding author: Jianhua Liu (jianhuacafuc13@cafuc.edu.cn)

This work was supported in part by the Science and Technology Department in Sichuan Province of China under Grant 2022JDKP0093 and Grant 2022JDRC0076, in part by the Scientific Research Project of Civil Aviation Flight University of China under Grant ZX2021-03, and in part by the Project of Basic Scientific Research of Central Universities of China under Grant ZHMH2022-004 and Grant J2022-025.

ABSTRACT Edge computing is a new computing paradigm which distributes tasks to edge networks for processing, it provides effective support for various mobile applications to meet their rapid response requirement. Unmanned Aerial Vehicle (UAV) has been widely used in emergency rescue, mapping, etc., with the advantages of flexible deployment and rapid movement. However, on one hand, mobile applications will be terminated when the battery energy is exhausted. On the other hand, mobile applications will be out of service when mobile devices are out of radio coverage of UAVs. How to achieve low-cost task unloading in the resource limited and location sensitive multi-UAVs edge computing environment is rather challenging. In this paper, we propose a distributed location-aware task offloading scheme, aiming at the above issues. Specifically, we create a nonlinear task allocation problem by combining the limited energy constraints of edge nodes with the random movement of users, where the cost function is divided into static and dynamic costs, respectively. Then, we formulate this problem to a convex optimization one with linear constrains, based on regularization technology. The mathematical proof shows that the scheme can support a parameterized competitive ratio without requiring any prior knowledge of the input task. The simulation results show that the proposed scheme can achieve lower cost edge computing services.

INDEX TERMS Convex programming, edge computing, task allocation, UAV.

I. INTRODUCTION

Currently with the development of communication and chip technologies, the Internet of Things (IoT) is outstanding in improving humans' quality of life. Users can quickly obtain information and process various tasks through the IoT. However, many IoT devices are subject to restrictions, such as limited storage and computation capacity, and imbalanced distribution of the capacity among these devices, which can reduce the service performance of the IoT. Cloud computing can enhance the IoT's capability in terms of storage, computation and management of various resources [1]–[3]. Unfortunately, because the cloud is far away from users, the response time of cloud computing cannot meet real-time applications for IoT-based applications. Edge computing is a new paradigm that aims to reduce the burden on Cloud and improve the response delay of requirements. With edge computing, many tasks can be completed directly by these edge computing devices without the participation of the cloud [4]. To provide more flexible services, mobile edge computing (MEC) is proposed to enhance the information processing capabilities of mobile devices regardless of their high location sensitivity. However, these edge devices are often heterogeneous, their computing capacities are quite different, and there exists a competitive relationship among these edge devices, therefore, there are still some energy consumption and reliability problems with MEC that must be solved.

The MEC is vulnerable on reliability, especially high dynamics that frequently occur in UAV-enabled MEC. Unlike conventional Internet of Vehicles (IoV), the movement trajectory of a mobile node is predictable and always transforms

The associate editor coordinating the review of this manuscript and approving it for publication was Eyhab Al-Masri.

along a certain road. Although existing computation offloading schemes are effective in achieving low latency for mobile users, for battery-powered UAVs, the computation performance may be compromised due to limited battery energy and radio coverage for task offloading. On one hand, mobile applications will be terminated when the battery energy is exhausted [5]. On the other hand, mobile applications will be out of service when mobile devices are out of radio coverage of UAVs. It is difficult to guarantee a robust edge computing service with minimum latency during rapid movement, especially if loT devices produce a lot of data that require processing in real-time [6]. Besides the computation time cost, as discussed in [7]–[10], UAV-enabled MEC in multi-UAVs environments incurs an additional time overhead on task migration from one UAV to another UAV since the UAV has limited computation capability.

To solve the above problems, an multi-UAVs enabled edge computing scheme was proposed. Edge computing is performed at the Internet's edge with multiple UAVs. Edge nodes also refers to cloudlets, and fog nodes, which has advantages in the quick response to IoT applications [11], [12]. Motivated by [13], we divide the cost of task offloading into dynamic cost and static cost, these cost can be calculated by operating cost, quality of service (QoS) cost and migration cost. The position changing of UAVs is an important factor affecting the task offloading strategy. In summary, the main contributions of this work are presented as follows,

1. To solve the problem of dynamic task assignment in a UAV environment, a comprehensive optimization model is constructed. This model takes full account of the energy limit and location sensitivity of nodes, can comprehensively optimize the migration cost, operation cost, and QoS cost.

2. The above problem is converted into a convex optimization problem, and an efficient online algorithm is proposed. Through rigorous competitive analysis, it is proved that the proposed algorithm has a parameterized competitive ratio.

3. The simulation results show that the proposed scheme has reliable robustness regardless of various numbers of users and UAV nodes, and can save 20% of the total cost than other methods.

This paper is organized as follows. In Section II, related work is introduced. Section III describes our models and formulates the problem. Section IV focuses on the design details of online algorithm. Section V presents the formal competitive analysis. Experiment results and analyses are reported in Section VI. The final section concludes this paper.

II. RELATED WORK

Due to the development of chip and communication technology, the popularity of mobile devices is growing rapidly, such as smart phones, wearable devices, smart cars, robots and UAVs. Edge Computing integrates resources to provide users with real-time and feasible service on the side of the Internet. Edge Computing releases the pressure on the cloud effectively and realizes reliable utilization of optimized resource allocation on the edge Internet. A large number of location-sensitive mobile application services have grown rapidly because of the rapid development of Internet of Thing (IoT) and artificial intelligence (AI), such as intelligent maintenance, intelligent logistics, and so on. UAV has the advantages of flexible deployment and rapid movement. It is widely used in emergency rescue, aerial photography, logistics, and so on. In addition, the wireless communication channel between UAV and ground equipment has strong line-of-sight characteristics [14], which could provide high quantity communication services and edge computing services for users. UAV edge computing platform supplies fast and flexible information support for mobile services. Researchers pay more attention that Multi-UAVs could cover a wider testing scale in recent years [15]–[17].

The main problem for edge computing is how to provide a dynamic task assignment scheme for various mobile applications in a distributed environment [4], [18]–[22]. Because edge cloud is heterogeneous and dynamic, neither resource allocation scheme nor task offloading scheme is suitable for the large-scale central cloud. The user location is mobility-agnostic, and the task scheduling of edge cloud cannot be one-off, but an adaptive scheme that changes with the user's location. Therefore, the cost of task offloading is also an adaptive cost, including bandwidth cost, migration cost, latency cost, and so on [13].

A resource allocation scheme based on convex programming was proposed in [13] to decrease the service cost for edge computing, optimizing operation cost, migration cost, and reconfiguration cost. Unfortunately, this scheme ignored the energy limit of mobile edge nodes. A task assignment model based on hybrid nonlinear programming [23] was constructed to reduce service delay in the ultra-dense software defined network (SDN) and saved energy consumption of edge cloud users. Under the condition of time latency constraint, joint optimization was used to minimize the sum of energy consumption of local devices, to obtain a computing offloading scheme based on energy efficiency under a single MEC node. However, this scheme only considered a single edge node and did not discuss the mobility of nodes, so the scope of application was limited. A task assignment scheme was proposed in [24], which could reduce the task execution delay and improve the data transmission rate. The scheme reduced the delay of edge service and provided security protection for the information being processed. To enhance the privacy of users, Puthal *et al.* [25] propose a service allocation scheme based on trust awareness, which is a multi-objective optimization scheme based on privacy entropy, energy consumption, and delay. Unfortunately, the scheme ignores the user's energy consumption limitation and mobility. To guarantee the fairness of different edge devices in a completely distributed environment, Xu *et al.* [26] apply simulation learning to resource scheduling algorithm of edge computing, and realize the transformation from local optimal

solution to global optimal solution by guiding the online training of multi-agents with simulation learning results of multiple experts. But the convergence of the algorithm is not discussed in this work. To solve the problem of passive eavesdropping on the ground when ground users offload data to the edge computing network of multi-UAVs, Cui *et al.* [15] propose a safe task offloading strategy to minimize system energy consumption through joint optimization between user and resource, but this scheme ignores the position sensitivity of UAVs. By taking into consideration the inter-dependencies of the tasks, dynamic network states, and energy constraints of the UAVs, [27] formulates the average mission response time minimization problem and then model it as a Markov decision process. To achieve optimal average delay over trajectory, an edge computing framework [28] was developed, which allows enabling optimal offloading decisions as a function of network, computation load parameters and current state. The optimization is formulated as an optimal stopping time problem over a semi-Markov process. To maximize the secure computation efficiency, a two-stage alternative optimization algorithm [29] was proposed by jointly optimizing the transmission power and UAV's trajectory. To investigate the performance of intelligent reflecting surface (IRS)-aided wireless powered MEC systems, an offloading framework has been proposed in [30], in which three different levels of dynamic IRS beamforming schemes are considered. To minimize the total energy consumption of all TDs, a two-layer iterative algorithm is proposed to solve the non-convex and mixed-integer optimization problem in [31], which adopt a one-by-one access mechanism.

Unfortunately, the efficiency of these schemes is questionable for tasks with large volumes since they ignored the queuing time of tasks in a high dynamic environment. Moreover, few studies consider the rapid movement together with battery capacity limitation. Different from the above literatures, our paper proposes a dynamic task offloading scheme and exploits optimizing the online task offloading of IoT applications for the MEC network.

III. SYSTEM MODEL AND PROBLEM DESCRIPTION

A. SYSTEM MODEL

We consider a time-slotted system whose whole running time T is divided into h time slots, and denote $T = \{t_1, t_2, \ldots, t_h\}$, $(h \in Z^+)$. We envisage an edge computing system with m users, denoted by $U = \{u_1, u_2, \ldots, u_m\}$, $(m \in Z^+)$. Here, the users refer to various IoT applications. We consider an edge-compatible mobile service supported by a set of n UAVs, denoted by $S = \{s_1, s_2, \ldots, s_n\}$, $(n \in Z^+)$. In our model, a UAV is an edge node. All the UAVs constitutes an edge network, which can provide users with efficient edge computing services. The m users and n UAVs are distributed in the considered area and their locations changes randomly at time slot $t_i (1 \leq i \leq h)$. The maximum data processing power of edge cloud is denoted by $C_s, s \in S$. The internet transforming latency between two edge nodes s_1 and s_2 is denoted by $d(s_1, s_2) \geq 0, \forall s \in S, d(s, s) = 0$ is feasible. Each edge node can cover a certain area. Each user can choose the closest edge node with the best transmission effect to establish transmission link and upload workloads.

Since the position of UAVs and the users changes randomly at each time slot t_i, the links of the users in the system also change randomly. We assume that the transmission link established by the user is stable and remains unchanged at a time slot $t_i (1 \leq i \leq h)$. $l_{u,t}$ indicates the position of user u at time slot $t \in T$. $s^*_{u,t}$ indicates that user u establishes a transmission link with edge node s at time slot $t \in T$, and u offloads her tasks to the edge node s. $x_{u,s,t}$ represents the amount of tasks that user u offloads to edge node s at time slot t. We denote the total number of tasks by λ_u. The data processing capability of s at time slot t is denoted by $q_{s,t}$.

Table 1 shows the correspondent relationship between words and abbreviations.

The architecture of our proposed multi-UAVs enabled MEC system is shown in Figure 1. The system consists of a number of users (IoT applications), edge nodes (UAVs), base stations, cloud, wireless connections between UAV and IoT applications, and transmission links between UAVs, and transmission links between UAV and base station. IoT applications are usually pinned on some IoT nodes, such as smart phone, camera, Raspberry Pi and other sensors. The IoT applications can obtain a large amount of first-hand data from application environments such as earthquake, tsunami and smart farm, then transmit these data to the nearest UAV for further processing. When an UAV flies over these mobile applications, these applications will upload data processing tasks to the UAV. Based on the requirements of real-time and cost management, the UAV may divide the tasks into multiple subtasks and transmit the subtasks to other UAVs for processing. Therefore, task offloading is not a one-shot task and needs to be continuously adapted to accommodate UAVs' and applications' movements, incurring the ''adaptation cost'' over time. Every application and UAV can move arbitrarily in the system, and, from a time-slotted view, an application may connect to the UAV in one time slot and switch to another in the next. In each time slot the system can have its own optimal task allocation strategy, which may, however, become suboptimal if the adaptation cost during time-slot transition is considered.

When the service capability of the edge network cannot meet the needs of users, the task will be sent by the edge nodes to the cloud through the base station, because the users usually cannot establish direct links with the base station. In this paper, we mainly focus on the task allocation in the edge network equipped with multi-UAVs, ignore the situation that the UAV sends the task to cloud for processing. It is assumed that each UAV is equipped with a directional antenna to receive data from IoT applications and two transmit antenna systems for data transmission. Note that the UAV may suffer from the influences of aerodynamic forces. In order to guarante the stability of signals transmitted in our system, we assume that the antenna arrays at UAV are equipped on the airborne gimbals [32].

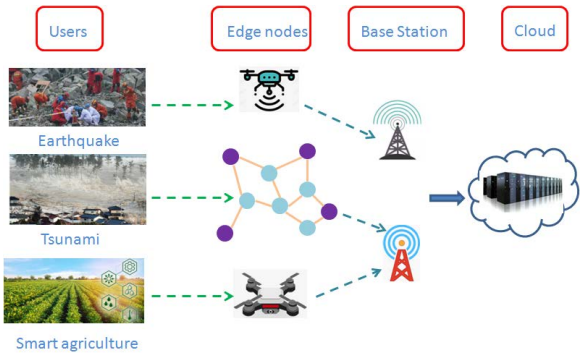


FIGURE 1. Network structure of edge computing with multi-UAVs.

TABLE 1. List of main notations.

Symbol	Meaning
T	running time of the edge computing system
t_h	time slot, $h \in Z^+$
U	the group of users
u_m	the mth user, $m \in Z^+$
S	the group of UAV edge nodes
s_n	the nth UAV edge nodes, $m \in Z^+$
C_s	the maximum data processing power of edge cloud
$d(s_1, s_2)$	The internet transforming latency between edge node s_1 and node s_2
$x_{u,s,t}$	the amount of tasks that user u offloads to edge node s at time slot t
λ_u	the total number of tasks
$q_{s,t}$	the data processing capability of s at time slot t
E_O	operating cost of the edge computing system for all tasks
E_Q	service quality cost of the edge computing system
E_M	operating cost of the edge computing system for all tasks
$T_w(u, s)$	the waiting time for task $x_{u,s,t}$ in the transmission queue from user u to edge node s

B. COST MODEL

The cost of the system is divided into three types: operation cost, QoS cost, and migration cost [13]. Among them, the first two are static costs and are generated independently at each time slot. The migration cost is a dynamic cost, which depends on the task transfer decision during the task execution.

Operating cost: This cost mainly includes employing and maintaining software, hardware resources, CPU and storage costs, electricity costs, and carbon emissions costs. The maintenance cost is proportional to the number of tasks handled by each edge cloud. Operation unit price represents the unit cost of edge node s processing unit task quantity at time slot t, which is denoted by $a_{s,t}$, In practical applications, the operation price $a_{s,t}$ for each edge node follows Gaussian distribution, where we set the mean value as the base price [13]. In reality, all equipment and energy are provided by the service provider, and the service provider only publishes a total operating price when charging users, and it is not necessary to distribute the detailed composition of the price. Thus, we use $a_{s,t}$ to represent the unit price of all operation costs. The operating cost of the edge computing system for all tasks is shown as formula (1):

$$E_O = \sum_{t\in T}\sum_{s\in S} a_{s,t} \sum_{u\in U} x_{u,s,t} \tag{1}$$

QoS cost: This cost is mainly used to represent the user quality of service. It is proportional to the network delay after a user submited her service request to the system to execute some tasks and return the associated result. It is assumed that all nodes in the system employ orthogonal frequency division multiple access (OFDMA) technology to communicate with other nodes. Transmission rules follows the first-in, first-out (FIFO) principle during transmission. The UAVs can leverage their flexible mobility to obtain line-of-sight (LoS) air-to-ground channel with a high probability [31]. In fact, the LoS link for ground-to-aerial (G2A) channel/aerialto-ground (A2G) channel offers a good approximation for the practical G2A/A2G model when the UAV is above a certain altitude, which has been verified by Qualcomm [33]. According to the work of [34], the uplink channel state information (CSI), such as delay and angle of arrival, can be obtained by pilot or blind estimation, while the downlink CSI is estimated by IoT nodes based on pilot training transmitted by downlink. Based on the M/G/1 queuing theory, the wait time of task $x_{u,s,t}$ in the transmission queue from user u to edge node s is calculated as follows [35], [36]:

$$T_w(u, s) = (\rho_{u,s,t}\overline{d}(u, s) + \rho_{u,s,t}\delta^2)/2(1 - \rho_{u,s,t}\overline{d}(u, s)),$$

where $\overline{d}(u, s)$ is the average time delay for a task to be transmited from u to s; δ^2 represents the variance of transmission delay; $\rho_{u,s,t}$ means transmission density. The user's access delay is denoted by $d(l_{u,t}, s^*_{u,t})$, here $l_{u,t}$ is the location of user u in time slot t. For simplicity, we denote $T_w(u, s)$ by T_w. The weighted sum of the delay between the access edge node and each of the edge nodes that host the workload of user u is denoted by $\sum_{s\in S} \frac{x_{u,s,t}}{\lambda_u}\frac{q_{s,t}}{C_s} d(s^*_{u,t}, s)$. Then, the total service quality cost of the edge computing system can be expressed as:

$$E_Q = \sum_{t\in T}\sum_{u\in U}(d(l_{u,t}, s^*_{u,t}) + T_w + \sum_{s\in S}\frac{x_{u,s,t}}{\lambda_u}\frac{q_{s,t}}{C_s} d(s^*_{u,t}, s)) \tag{2}$$

Migration cost: To optimize the cost, tasks often need to be migrated from one edge node to other edge nodes during task execution. Migration cost includes network bandwidth cost and migration delay cost. It is noted that the difference between the tasks in the two time slots is not equal to the migrated tasks. Because the edge cloud is constantly processing tasks at each time slot, the number of tasks completed at a time slot should be added to the difference between two adjacent time slots to equal the amount of migrated tasks. We denoted the unit price of migrating unit task quantity by b_s. If function $(x)^+ = \max\{x, 0\}$ is defined, the total migration cost is calculated as follows:

$$E_M = \sum_{t\in T}\sum_{s\in S} b_s(\sum_{u\in U} x_{u,s,t} + q_{s,t-1} - \sum_{u\in U} x_{u,s,t-1})^+ \tag{3}$$

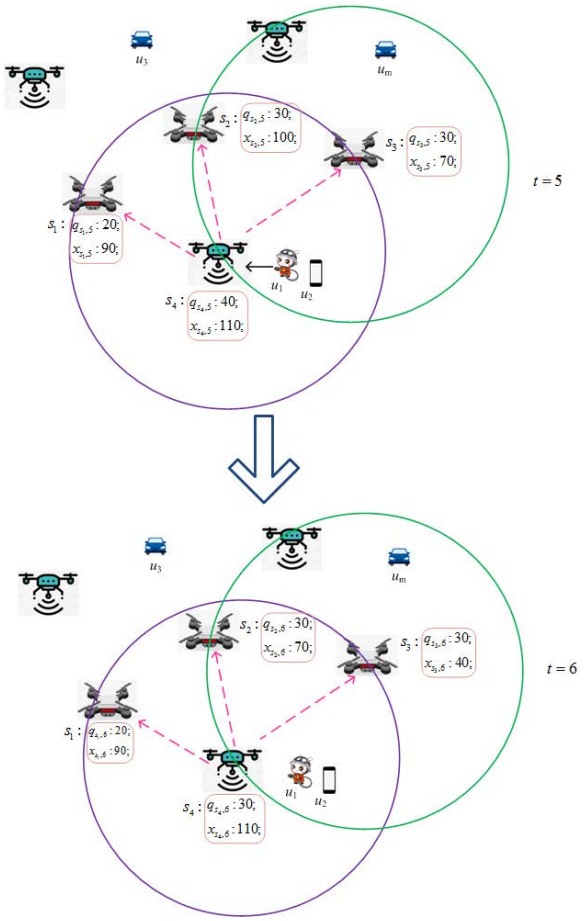


FIGURE 2. Task migration from time slot 5 to 6.

Here, $(\sum_{u\in U} x_{u,s,t} + q_{s,t-1} - \sum_{u\in U} x_{u,s,t-1})^+$ represents the amount of tasks transferred by edge node s in the process of transferring from time slot $t-1$ to time slot t. To simplify the expression, $\sum_{u\in U} x_{u,s,t}$ is denoted by $x_{s,t}$, that is, the sum of tasks assigned to edge cloud s by all users at time slot t. Although task migration itself is dynamic and complex, the migration cost of an edge node is always directly related to the change of task volume. However, the migration amount is not equal to the difference between the tasks in the current time slot minus the tasks in the previous time slot. An example of task migration is shown in Figure 2. The blue circle indicates the effective radio signal coverage of s_4, and the green circle indicates the radio coverage of s_3. In Figure 2, both the signals of edge node s_3 and s_4 can cover user u_1, who is closer to s_4. The signal of s_4 will be stronger than that of s_3. Therefore, u_1 takes s_4 as the preferred node for network service.

The total tasks of s_1 in the current slot 6 and the previous slot 5 are set to be 90. That is, $x_{s_1,6} = x_{s_1,5} = 90$, then, the task being migrited into edge node s_1 from time slot 5 to 6 cannot be calculated by $x_{s_1,6} - x_{s_1,5} = 90-90 = 0$ since the edge node s_1 is working all the time. Assuming $q_{s_1,5} = 20$, then the task being migrated into s_1 from time slot 5 to 6 can be calculated as $90+20-90 = 20$. The calculation process of task migration associated with Figure 2 is shown in Table 2.

TABLE 2. Calculation process of task migration associated with Fig 2.

Edge node	time slot 5		time slot 6		tasks moving into edge node
	$q_{s,5}$	$x_{s,5}$	$q_{s,6}$	$x_{s,6}$	$(x_{s,t} + q_{s,t-1} - x_{s,t-1})^+$
s_1	20	90	20	90	20
s_2	30	100	30	70	0
s_3	30	70	30	40	0
s_4	40	110	30	110	40

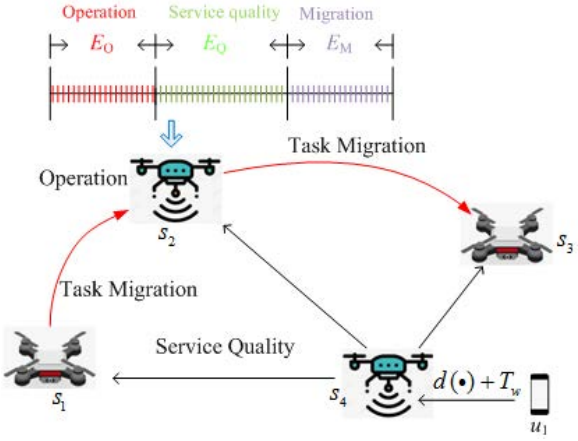


FIGURE 3. Overall cost structure of our multi-UAVs enabled edge computing system.

In actual operation, in order to reduce operation and maintenance costs and facilitate management, service providers may prefer to keep UAVs with the same model, which equipped with the same mission loads. Due to the change of working state, the available resources and real-time performance of the same UAV will also shift randomly. Therefore, it is reasonable and necessary to set different values of $q_{s,t}$ for UAVs in our experiment.

C. PROBLEM FORMULATION

The total cost of edge calculation can be expressed as the weighted sum of the three costs mentioned above, which can be expressed as:

$$P = E_O + E_Q + E_M \tag{4}$$

The overall cost structure is illustrated in Fig. 3. We believe that these cost models are general enough and can capture a wide range of practical performance measures in an multi-UAVs enabled edge computing system from the perspective of edge cloud users. Different from the stucture of [13], our cost function does not include reconstruction cost. Note that during the working process of an UAV, its software and hardware devices are always in a hot state. Thus, the hardware and software preparation time can be ignored.

The unit of operating cost is a price unit, because the coefficient $a_{s,t}$ is the price of a unit resource, and the product factor $x_{u,s,t}$ represents the quantity of resources. The QoS cost consists of three parts: the access delay, the wait time in the transmission queue, and the weighted sum of the delay between the access edge node and each of the edge nodes

that host the workload, thus, the unit of QoS cost is a time unit. Similar to the operating cost, the unit of migration cost is a price unit. The objective function is the sum of the three costs. In fact, the units of the three costs need to be unified before adding. Unity of units can be achieved by multiplying each part by a factor, i.e. a tradeoff weighting coefficient. Without loss of generality, we have omitted the weighting coefficient in the above equation for simplicity, since the weighting coefficient can be implied in the parameters of each cost model, such as $a_{s,t}$ and b_s in the expression form of Equation (4). To solve the optimization problem of the total cost, the following linear programming problem is generated:

$$\min \quad P_0(t) = E_O + E_Q + E_M \tag{5}$$

$$s.t. \quad \sum_u x_{u,s,t} \le C_s, \quad \forall s, \forall t \tag{5a}$$

$$\sum_s x_{u,s,t} \ge \lambda_u, \quad \forall u, \forall t \tag{5b}$$

$$\sum_u k x_{u,s,t} \le Q_s, \quad \forall u, \forall t \tag{5c}$$

$$x_{u,s,t} \ge 0, \quad \forall u, \quad \forall s, \quad \forall t \tag{5d}$$

Constraint condition (5a) ensures that the sum of all tasks assigned to the edge cloud does not exceed its computing and storage capacity. Constraint condition (5b) ensures that all tasks are assigned, and greater-than character ensures that the system remains redundant. k represents the power consumption coefficient of unit task quantity, and the constraint condition (5c) ensures that the sum of all tasks assigned to the edge cloud s will not exceed the battery limit of UAVs. Let $Z_s = min\{Q_S/k, C_s\}$, then constraint conditions (5a) and (5c) are combined into:

$$\sum_u x_{u,s,t} \le Z_s, \quad \forall s, \quad \forall t \tag{5e}$$

IV. ONLINE ALGORITHM DESIGN

Since each edge node cannot obtain the state information of the whole system in a distributed environment, we introduce a competitive ratio to estimate the efficiency of the proposed online algorithm solution. The competitive ratio is the optimization results of an online algorithm with limited information to an offline algorithm with all information.

A. ALGORITHM DESIGN

Next, we will solve the above problems based on regularization technique [13], [20]. The pseudo code of our proposed PSEC is listed in Table 3. At each time slot t, taking the user's current position $l_{u,t}$ and the task $x_{u,s,t-1}$ assigned at the previous time slot $t-1$ as input, the optimization problem $P_1(t)$ in the current time slot t can be obtained:

$$\min P_1(t) = \sum_{s\in S} a_{s,t} \sum_{u\in U} x_{u,s,t} + \sum_{u\in U} (d(l_{u,t}, s^*_{u,t}) + T_w + \sum_{s\in S} \frac{x_{u,s,t}}{\lambda_u} \frac{q_{s,t}}{C_s} d(s^*_{u,t}, s))$$

TABLE 3. Pseudo code of our proposed PSEC.

Algorithm 1
1:caculate original total cost P_0
2:regularize the objective P_0 to P_1
3:P_1 is transformed into linear programming problem P_2
4:initialize parameters
5:for $u \in U$
6: for $s \in S$
7: for $t \in T$
8: update P_2 according to fomula (8)
9: solve P_2 via convex programming
10: $t = t + 1$
11: end for
12: end for
13: end for

$$+ \sum_{s\in S} \frac{b_s}{\eta_s} ((x_{s,t} + q_{s,t-1} + \varepsilon) \ln \frac{x_{s,t} + q_{s,t-1} + \varepsilon}{x^*_{s,t-1} + \varepsilon} - x_{s,t} - q_{s,t-1}) \tag{6}$$

s.t.

$$\sum_s x_{u,s,t} \ge \lambda_u, \forall u \tag{6a}$$

$$\sum_{f\in S\setminus s} x_{u,f,t} \ge \sum_{u\in U} \lambda_u - Z_s, \forall s \tag{6b}$$

$$x_{u,s,t} \ge 0, \forall u, \forall s \tag{6c}$$

Here, both $\eta_s = \ln(1 + Z_s/\varepsilon)$ and ε are parameters which are greater than 1. Obviously, the objective function $P_1(t)$ is a convex function with linear constraints, thus we can use any convex programming solution to search the solution of $P_1(t)$. By combining the optimization solution $x^*_{u,s,t}$ obtained by $P_1(t)$ at each time slot t, the approximate optimization solution $\{x^*_{u,s,t} | 0 \le t \le t_h\}$ of the original problem $P_0(t)$ is obtained, which is the task allocation strategy of edge computing. Next, we prove that the optimization solution $\{x^*_{u,s,t} | 0 \le t \le t_h\}$ is the solution of the optimization problem $P_0(t)$.

Theorem 1:(Validity). The optimal solution $\{x^*_{u,s,t} | 0 \le t \le t_h\}$ of $P_1(t)$ at each time slot constitutes a solution of $P_0(t)$.

Proof: It is proved that the optimal solution of $P_1(t)$ also satisfies the constraints (5b), (5d) and (5e) in $P_0(t)$. Since $x^*_{u,s,t}$ is the optimal solution of $P_1(t)$ at time slot t, $\forall t$, $x^*_{u,s,t}$ makes (6a) and (6c) valid, thus (5b) and (5d) are valid. Next, we prove constraint (5e) is also true:

Because

$$\frac{\partial P_1(t)}{\partial x_{u,s,t}} = a_{s,t} + \frac{q_{s,t}}{\lambda_u C_s} d(s^*_{u,t}, s) + \frac{b_s}{\eta_s} \ln \frac{x_{s,t} + q_{s,t-1} + \varepsilon}{x^*_{s,t-1} + \varepsilon - 1} + \frac{b_s}{\eta_s} (x^*_{s,t-1} + q_{s,t-1} + \varepsilon) > 0,$$

from the above equation, we can obtain that $\forall u, \forall s, \forall t$, the function $P_1(t)$ is monotonically increasing on the interval $[x^*_{u,s,t}, +\infty)$ with respect to $x_{u,s,t}$. Without losing generality, let's assume $x_{u,s,0} = 0$ and $x^*_{s,0} = \sum_{u\in U} x^*_{u,s,t} = 0$. The following is a proof by mathematical induction. When

$t = 1$, and $x_{u,s,1}$ represents an optimal solution of $P_1(t)$. Then, there must be $x^*_{s,1} = \sum_{u\in U} x^*_{u,s,1} \leq Z_s$. Otherwise, if $x^*_{u,s,1} > Z_s$ or $x^*_{s,1} > Z_s$, then, $\grave{x}_{u,s,1} = Z_s$ and $\grave{x}_{s,1} = Z_s$ is a new optimal solution for the problem $P_1(t)$. Obviously, this new optimal solution leads to a smaller target value for $P_1(t)$, which contradicts the hypothesis. Similarly, the same conclusion can be reached when $0 \leq t \leq t_h$.

B. COMPLEX ANALYSIS

At each time slot, the solution of $P_1(t)$ can be obtained by solving the convex programming problem. Therefore, the algorithm proposed by us is solvable in polynomial time [37]. Here, we can choose the interior point algorithm to solve our problem, then the time complexity of our algorithm is $O((mn))^{3.5}$, where both m and n are the numbers of variables in function $P_1(t)$, m is the number of edge nodes, and n is the number of users.

V. COMPETITION ANALYSIS

The efficiency analysis of the proposed scheme is given through competition analysis. Competition analysis mainly uses competitive ratio to analyze the efficiency of online algorithms. The method is to compare the efficiency of an online algorithm with that of an offline algorithm where all input conditions are known in advance. The basic steps of the analysis are as follows:

- The relaxation of programming problem $P_0(t)$ is transformed into linear programming problem $P_2(t)$ by auxiliary variables $y_{s,t}$ and $z_{s,t}$.
- Derive the duality problem D of $P_2(t)$.
- Construct an efficient solution of problem D by a solution of $P_1(t)$.

Our goal is to derive the inequality [13]:

$$P_0(t) \geq P_2(t) \geq D \geq \frac{1}{r} P_1(t) \tag{7}$$

Since $P_2(t)$ is a relaxation of the constraint on $P_0(t)$, $P_0(t) \geq P_2(t)$ is true. We know $P_2(t) \geq D$ from the weak duality theorem. The last inequality is deduced by comparing the solutions of D and $P_1(t)$. The existence of all the above inequalities guarantees that the proposed algorithm is r competition, where the competitive ratio r will be given later in Lemma 4. The specific analysis process is shown below.

A. AUXILIARY PLANNING PROBLEM

By introducing auxiliary variables $y_{s,t}$ and $z_{s,t}$, the nonlinear programming problem $P_0(t)$ is reconstructed to obtain the relaxed linear programming problem $P_2(t)$. We also give the lower bound of the relaxation variable, and the formula of $P_2(t)$ is shown as follow:

$$\begin{aligned} \min \quad & P_2(t) = \sum_{s\in S}\sum_{t\in T}\sum_{u\in U} a_{s,t} x_{u,s,t} \\ & + \sum_{s\in S}\sum_{t\in T}\sum_{u\in U} \frac{x_{s,u,t}}{\lambda_u}\frac{q_{s,t}}{C_s} + \sum_{s\in S}\sum_{t\in T} b_s y_{s,t} \end{aligned} \tag{8}$$

$$s.t. y_{s,t} \geq \sum_{u\in U} x_{u,s,t} + q_{s,t-1} - \sum_{u\in U} x_{u,s,t-1}, \forall s, \forall t \tag{8a}$$

$$\sum_{f\in S\setminus s} x_{u,f,t} \geq (\sum_{u\in U} \lambda_u - Z_s)^+, \forall s, \forall t \tag{8b}$$

$$y_{s,t} \geq 0, \forall s, \forall t \tag{8c}$$

$$(5b), (5d) \tag{8d}$$

When the connection between the user and the directly accessed UAV is established and the user queuing model is determined, $\sum_{t\in T}\sum_{u\in U}(d(l_{u,t}, s^*_{u,t}) + T_w)$ is independent of task allocation strategy, thus this part of cost is not considered in Equation (8).

Next, we give the Lagrangian dual D of P_2. Let $\alpha_{s,t}$, $\rho_{s,t}$,and $\theta_{u,t}$ be dual variables of equations (8a), (8b) and (5b) respectively. Let $h_{u,s}$ be the indicator variable associated with a user and edge cloud, namely, if user u has tasks assigned to s, $h_{u,s} = 1$; otherwise, $h_{u,s} = 0$. Duality programming problem D is as follows:

$$\max \quad D = \sum_{t\in T}\sum_{u\in U} \lambda_u h_{u,s}\theta_{u,t} + \sum_{t\in T}\sum_{s\in S}(\sum_{u\in U}\lambda_u - Z_s)^+ \rho_{s,t} \tag{9}$$

$$\begin{aligned} s.t. & -a_{s,t} - g_{s,u}\frac{q_{s,t}}{\lambda_u C_s} d(s^*_{u,t}, s) + \alpha_{s,t+1} - \alpha_{s,t} \\ & + \sum_{f\in S\setminus s} \rho_{f,t} + g_{s,u}\theta_{u,t} \leq 0, \forall u, \forall s, \forall t \end{aligned} \tag{9a}$$

$$-b_s + \alpha_{s,t} \leq 0, \forall s, \forall t \tag{9b}$$

$$\alpha_{s,t} \geq 0, \rho_{s,t} \geq 0, \forall s, \forall t \tag{9c}$$

$$\theta_{u,t} \geq 0, \forall u, \forall t \tag{9d}$$

At the same time, we can obtain the Karush-Kuhn-Tucker (KKT) condition of $P_1(t)$. Let the dual variables related to constraints (6a), (6b) and (6c) be $\grave{\theta}_{u,t}$, $\grave{\rho}_{f,t}$ and $\grave{\delta}_{u,s,t}$, respectively. The following formula we could obtain:

$$\begin{aligned} & a_{s,t} + g_{s,u}\frac{q_{s,t}}{\lambda_u C_s} d(s^*_{u,t}, s) + \frac{b_s}{\eta_u} \ln \frac{x_{s,t} + q_{s,t-1} + \varepsilon}{x^*_{s,t-1} + \varepsilon} \\ & - g_{s,u}\grave{\theta}_{u,t} - \sum_{f\in S\setminus s} \grave{\rho}_{f,t} - \grave{\delta}_{f,t} = 0, \forall s, \forall u \end{aligned} \tag{10}$$

$$s.t. \grave{\theta}_{u,t}(\lambda_u - \sum_{s} x_{u,s,t}) = 0, \forall u \tag{10a}$$

$$\grave{\rho}_{f,t}(\sum_{u\in U}\lambda_u - Z_s - \sum_{f\in S\setminus s} x_{u,f,t}) = 0, \forall s \tag{10b}$$

$$\grave{\delta}_{u,s,t} x_{u,s,t} = 0, \forall u, \forall s \tag{10c}$$

$$(6a)(6b)(6c), \grave{\theta}_{u,t} \geq 0, \grave{\rho}_{f,t} \geq 0 \forall s, \forall u \tag{10d}$$

where (10) is based on stability, and (10b), (10c), and (10d) are based on complementary relaxation conditions. Using an optimal solution $x^*_{u,s,t}$ of $P_2(t)$ and dual variables $\grave{\theta}_{u,t}$ and $\grave{\rho}_{u,t}$, an optimal solution S_D of the dual programming problem D can be constructed through following formulas:

$$\alpha_{s,t} = \frac{b_s}{\eta_s} \ln \frac{C_s + \varepsilon}{x^*_{s,t-1} + \varepsilon} \geq 0,$$

$$\theta_{u,t} = \grave{\theta}_{u,t},$$

$$\rho_{u,t} = \rho^{`}_{u,t}.$$

Lemma 1: S_D is a valid solution of the programming problem D.

Proof: We first derive some preliminaries. For $\alpha_{s,t}$ we have

$$\begin{aligned}-(\alpha_{s,t+1} - \alpha_{s,t}) &= \frac{b_s}{\eta_s}\ln\frac{C_s+\varepsilon}{x^*_{s,t-1}+\varepsilon} - \frac{b_s}{\eta_s}\ln\frac{C_s+\varepsilon}{x^*_{s,t}+\varepsilon}\\ &= \frac{b_s}{\eta_s}\ln\frac{x^*_{s,t}+\varepsilon}{x^*_{s,t-1}+\varepsilon}\end{aligned}$$

Based on the above equation, we have

$$\begin{aligned}&-a_{s,t} - g_{s,u}\frac{q_{s,t}}{\lambda_u C_s}d(s^*_{u,t}, s) + \alpha_{s,t+1} - \alpha_{s,t}\\ &\quad + \sum_{f\in S\setminus s}\rho_{f,t} + g_{s,u}\theta_{u,t}\\ &= -a_{s,t} - g_{s,u}\frac{q_{s,t}}{\lambda_u C_s}d(s^*_{u,t}, s) - \frac{b_s}{\eta_s}\ln\frac{x^*_{s,t}+\varepsilon}{x^*_{s,t-1}+\varepsilon}\\ &\quad + \sum_{f\in S\setminus s}\rho_{f,t} + g_{s,u}\theta_{u,t} \le 0,\end{aligned}$$

where the inequality in the last line follows from equation (10). The above inequality indicates that the constraint (9a) is satisfied by the solution S_D. Constraint (9b) is satisfied by the following inequality.

$$\begin{aligned}\alpha_{s,t} &= \frac{b_s}{\eta_s}\ln\frac{C_s+\varepsilon}{x^*_{s,t-1}+\varepsilon}\\ &= \frac{b_s}{\ln(1+C_s/\varepsilon)}(\ln(1+C_s/\varepsilon) - \ln(1+x^*_{s,t-1}/\varepsilon))\\ &= b_s(1 - \frac{\ln(1+x^*_{u,s,t}/\varepsilon)}{\ln(1+C_s/\varepsilon)}) \le b_s.\end{aligned}$$

That is to say, (9a) and (9b) are true. According to the definition and the construction of the optimal solution, (9c) and (9d) are established. Therefore, the solution S_D of D constructed based on the solution of $P_2(t)$ satisfies all constraints of D.

B. COMPETITIVE RATIO

Next, we will analyze the competitive ratio r. First we divide $P_1(t)$ into two parts, the static cost part $E_O + E_Q$ and the dynamic cost part E_M, and then prove that each part is bounded. Then we combine the two parts to prove that the population is bounded and deduce the competitive ratio $r = 1 + \gamma n_0$, where γ and n_0 will be defined later in Lemma 4.

Lemma 2: D is the upper bound of operation cost and QoS cost in problem $P_0(t)$.

We first show the proof of the following preliminary results that will be used in the proof of lemma 2.

Lemma 3: $\forall s, \sum_{t\in T} x^*_{s,t}\ln\frac{x^*_{s,t}+q_{s,t-1}+\varepsilon}{x^*_{s,t-1}+\varepsilon} \ge 0$.

Proof: We separate the $\sum_{t\in T} x^*_{s,t}\ln\frac{x^*_{s,t}+\varepsilon}{x^*_{s,t-1}+\varepsilon}$ into two parts $\sum_{t\in T}(x^*_{s,t}+\varepsilon)\ln\frac{x^*_{s,t}+\varepsilon}{x^*_{s,t-1}+\varepsilon}$ and $\sum_{t\in T}\varepsilon\ln\frac{x^*_{s,t-1}+\varepsilon}{x^*_{s,t}+\varepsilon}$. Then, we deduce the two lower bounds of both parts one by one, the two bounds together will form a lower bound for the added formula. The detailed proof process is as follows.

$$\begin{aligned}&\sum_{t\in T} x^*_{s,t}\ln\frac{x^*_{s,t}+q_{s,t-1}+\varepsilon}{x^*_{s,t-1}+\varepsilon}\\ &\ge \sum_{t\in T} x^*_{s,t}\ln\frac{x^*_{s,t}+\varepsilon}{x^*_{s,t-1}+\varepsilon}\\ &= \sum_{t\in T}(x^*_{s,t}+\varepsilon)\ln\frac{x^*_{s,t}+\varepsilon}{x^*_{s,t-1}+\varepsilon} + \sum_{t\in T}\varepsilon\ln\frac{x^*_{s,t-1}+\varepsilon}{x^*_{s,t}+\varepsilon}\\ &\ge (\sum_{t\in T}(x^*_{s,t}+\varepsilon))\ln\frac{\sum_{t\in T}x^*_{s,t}+\varepsilon}{\sum_{t\in T}x^*_{s,t-1}+\varepsilon} + \sum_{t\in T}\varepsilon\ln\frac{x^*_{s,t-1}+\varepsilon}{x^*_{s,t}+\varepsilon}\\ &\ge \sum_{t\in T}(x^*_{s,t}+\varepsilon) - \sum_{t\in T}(x^*_{s,t-1}+\varepsilon) + \sum_{t\in T}\varepsilon(\ln(x^*_{s,t-1}+\varepsilon)\\ &\quad - \ln(x^*_{s,t}+\varepsilon))\\ &= (x^*_{s,t_h}+\varepsilon) - (x^*_{s,t_0}+\varepsilon) + \varepsilon\ln((x^*_{s,t_0}+\varepsilon)/(x^*_{s,t_h}+\varepsilon))\\ &= x^*_{s,t_h} + \varepsilon\ln(\varepsilon/(x^*_{s,t_h}+\varepsilon))\\ &\ge x^*_{s,t_h} + \varepsilon - (x^*_{s,t_h}+\varepsilon) = 0,\end{aligned}$$

The second inequality above holds based on the following inequality:

$\sum_i p_i\ln p_i/q_i \ge \sum_i p_i\ln\sum_i p_i/\sum_i q_i, \forall p_i > 0, \forall q_i > 0,$

The third inequality above holds based on the following inequality:

$p\ln p/q \ge p - q, \forall p > 0, q > 0.$

Then, we can obtain the upper bound of $E_O + E_Q$ and give the detailed proof of Lemma 2 as follows:

Proof: The derivation is conducted by applying the equations (10) − (10d) obtained from the Karush-Kuhn-Tucker (KKT) condition $P(1)$. Note that $\sum_{t\in T}\sum_{u\in U}(d(l_{u,t}, s^*_{u,t}) + T_w)$ is omitted as in objective $P_2(t)$.

$$\begin{aligned}&\sum_{t\in T}\sum_{s\in S}a_{s,t}\sum_{u\in U}x_{u,s,t} + \sum_{t\in T}\sum_{s\in S}\sum_{u\in U}\frac{x_{u,s,t}}{\lambda_u}\frac{q_{s,t}}{C_s}d(s^*_{u,t}, s)\\ &= \sum_{t\in T}\sum_{s\in S}\sum_{u\in U}x_{u,s,t}(-\frac{b_s}{\eta_u}\ln\frac{x_{s,t}+q_{s,t-1}+\varepsilon}{x^*_{s,t-1}+\varepsilon} + g_{s,u}\theta^{`}_{u,t}\\ &\quad + \sum_{f\in S\setminus s}\rho^{`}_{f,t} + \delta^{`}_{u,s,t})\\ &\le \sum_{t\in T}\sum_{s\in S}\sum_{u\in U}x_{u,s,t}(g_{s,u}\theta_{u,t} + \sum_{f\in S\setminus s}\rho_{f,t} + \delta_{u,s,t})\\ &= \sum_{t\in T}\sum_{u\in U}\lambda_u g_{s,u}\theta_{u,t} + \sum_{t\in T}\sum_{s\in S}\rho_{s,t}(\sum_{u\in U}\lambda_u - Z_s)\\ &\le \sum_{t\in T}\sum_{u\in U}\lambda_u g_{s,u}\theta_{u,t} + \sum_{t\in T}\sum_{s\in S}\rho_{s,t}(\sum_{u\in U}\lambda_u - Z_s)^+ = D.\end{aligned}$$

where the equation in the first line is obtained according to (10), the inequality in the second line is obtained by applying Lemma 2 and Lemma 3, the equality in the third line follows by (10a) − (10d).

Lemma 4: $\gamma n_0 D$ is an upper bound on migration cost in problem $P_0(t)$, where $\gamma = \max\{(Z_s+\varepsilon)\eta_s\}$, n_0 is the number of edge nodes.

Proof: For facilitate narration, set
$S_t^+ = \{s | s \in S \cap (x^*_{s,t} + q_{s,t-1}) > x^*_{s,t-1}\}$,
according to the definition $(x)^+ = max\{x, 0\}$, we have

$$\begin{aligned}
&\sum_{t\in T}\sum_{s\in S} b_s(\sum_{u\in U} x_{u,s,t} + q_{s,t-1} - \sum_{u\in U} x_{u,s,t-1})^+ \\
&= \sum_{t\in T}\sum_{s\in S_t^+} b_s(x^*_{s,t} + q_{s,t-1} - x^*_{s,t-1}) \\
&\le max\{(Z_s+\varepsilon)\eta_s\}\sum_{t\in T}\sum_{s\in S_t^+}\frac{b_s}{\eta_s}\ln\frac{x^*_{s,t}+q_{s,t-1}+\varepsilon}{x^*_{s,t-1}+\varepsilon} \\
&\le \gamma\sum_{t\in T}\sum_{s\in S}\sum_{u\in U}(g_{s,u}\theta_{u,t} + \sum_{f\in S\setminus s}\rho_{f,t}) \\
&\le \gamma\sum_{t\in T}\sum_{s\in S}\sum_{u\in U}(g_{s,u}\theta_{u,t} + \sum_{f\in S\setminus s}\rho_{f,t})\lambda_u \\
&\le \gamma n_0(\sum_{t\in T}\sum_{u\in U}\lambda_u g_{s,u}\theta_{u,t} + \sum_{t\in T}\sum_{s\in S}\rho_{s,t}(\sum_{u\in U}\lambda_u - Z_s)^+) \\
&= \gamma n_0\, D.
\end{aligned}$$

where, $\lambda_u \ge 1$. The equation in the first line is obtained by applying the definition $(x)^+ = max\{x, 0\}$, the inequality in the second line follows by the $p-q \le ln\frac{p}{q}$ for any $p > 0$, $q > 0$. Therefore, the proof is completed.

Combining all the results in Theorem 1, Lemma 3, and Lemma 4, the following theorem on the competitive ratio can be obtained for our proposed PSEC algorithm.

Theorem 2: PSEC generates feasible solutions to P_0 with a competitive ratio $r = 1 + \gamma n_0$.

VI. SIMULATED ANALYSIS

In this section, the performance of the PSEC algorithm is simulated in PC that has disposition with the Intel Core I5-9300H processor, 8 GB RAM and 512GB solid drive. The simulation platform uses MATLAB 2020a. The simulation scene is set within $100m * 100m$. The users and edge nodes move randomly within the range. The capacity C_s of each edge node is set to be [6000, 8000, 10000, 12000, 14000, 16000]Mb. The essential parameters and the scope of values are indicated in Table 4. Using the method of randomly generating workload, the distribution of task quantity satisfies the constraint condition (6a-6c), and the influence of different users and nodes on communication price is studied in the delay scenes. For further analysis, we compare our proposed PSEC algorithm with MOERA [13] algorithm and MEH [38] algorithm, respectively. The MEH is based on M/M/c queue to capture the execution process of tasks in MEC server with the optimal offloading probability ($p = 0.35$). Where, the task arrival rate is denoted by $\lambda = 0.3$, and task service rate is denoted by $\mu = 0.5$.

Figure 4 depicts the interrelationship between the operating cost, QoS cost and migration cost with different users. The three costs E_Q, E_M and E_O, are defined by the formulas (1), (2) and (3), respectively. It is obvious that the QoS cost increases rapidly with the increase of the number of users. The reason is that the QoS cost is affected by the communication distance. For mobile edge computing, the user is moving all the time, and the distance from each user to the associated edge node will varies randomly, which will significantly affect the user's QoS cost. The E_O hardly changes with the number of users since it only depends on the number of tasks processed in each time slot. When the number of users increases from 5 to 10, the migration cost E_M increases rapidly. However, when the number of users changes between 10 and 30, E_M tends to be stable. This is because when the number of users is small, migrating tasks from busy nodes to idle nodes will significantly reduce the total time cost, and the migration volume is large. When the number of users is greater than 10, all edge nodes enter busy state, thus, the migration of tasks has little effect on the reduction of the total cost.

TABLE 4. Experiment parameters.

Parameters	Default values
$T(s)$	3600
$\|U\|$	$[5, 10, 15, 20, 25, 30]$
$\|S\|$	$[5, 10, 15, 20, 25, 30]$
$q_{s,t}(Mb)$	$[64, 128, 256, 512, 1024, 2048]$
$C_s(Mb)$	$[6000, 8000, 10000, 12000, 14000, 16000]$
Qs/k	100000
ε	$[0.001, 0.01, 0.1, 1, 10, 100, 1000]$
$a_{u,s,t}$	$\sim$ U(0,1) ($a_{u,s,t}$ follows the uniform distribution)
b_s	$\sim$ U(0,1) (b_s follows the uniform distribution)
$\rho_{u,s,t}$	$\sim P(0.5)$ ($\rho_{u,s,t}$ follows the poisson distribution)

The performance affected by different number of edge nodes is shown in Figure 5. When number of edge nodes becomes larger, the operation cost increase lareger. This is because when a UAV is deployed to the edge network, even if it does not undertake any edge computing task, it still needs to consume operation cost. The number of UAVs is not the more the better, it should be deployed according to actual needs. The migration cost does not increase with the increase of the number of edge nodes. This is mainly because as the task volume remains unchanged, too much migration may lead to the increase of the total cost. In terms of the E_Q, it decreases as the number of edge nodes increases. It can be seen that when the number of UAVs increases, the operation cost of UAVs will increase, but the QoS cost will decrease. Therefore, the number of UAVs is not the more the better, it should be determined according to the total cost requirements.

Figure 6 illustrates the performance comparison with randomly moving edge nodes under various number of users. When the number of users increases, the total costs of the three schemes are increasing, respectively. When the number of users is less than 10, the cost of our PSEC is slightly higher than that of the MEH. When the number of users is greater than 10, the cost of the MEH increases rapidly, and the cost growth of our consumption is not obvious. For different number of users, the cost of our PSEC always accounts for about half of the MOERA's cost. From the perspective of the number of users, our proposed PSEC has achieved a more gentle growth. Futhermore, when the number of users

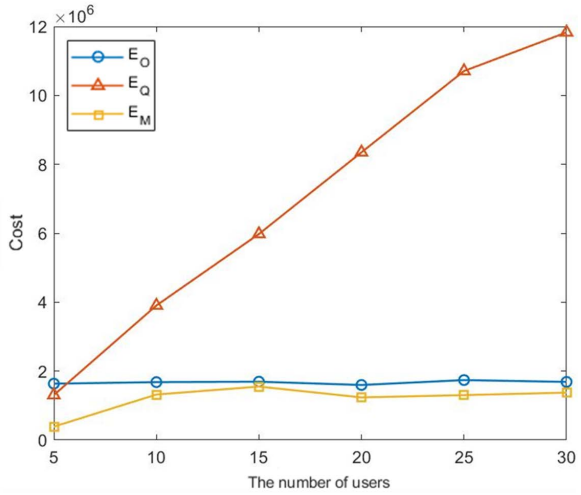


FIGURE 4. Impact of different users' number on three costs.

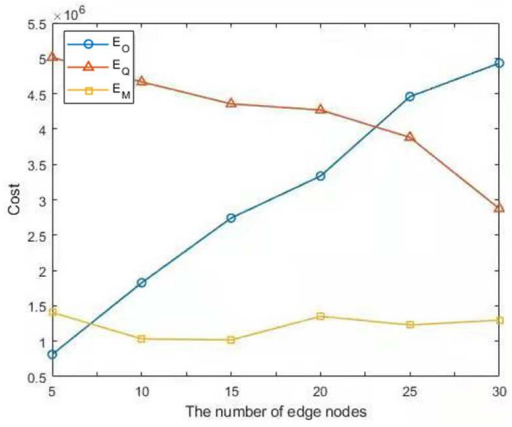


FIGURE 5. Impact of different edge nodes' number on three costs.

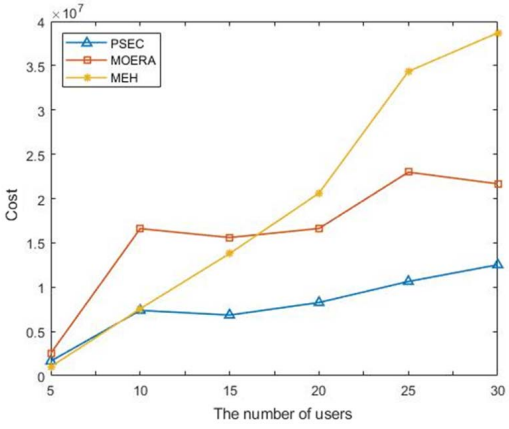


FIGURE 6. Impact of number of users on total cost.

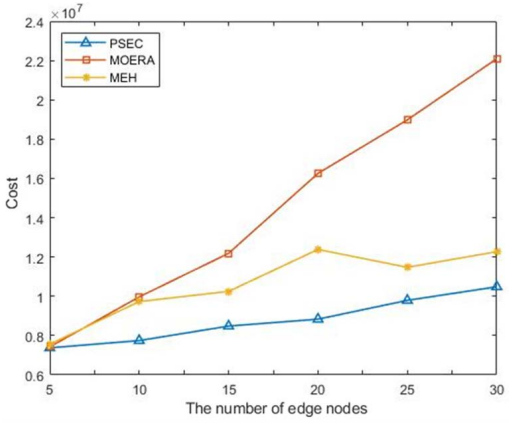


FIGURE 7. Impact of number of edge nodes on total cost.

is greater than 10, our PSEC can save about 50% of the total cost.

Figure 7 shows that the PSEC method can provide users with the lowest cost edge computing services under the different number of edge nodes. When the number of edge nodes is 5, the cost of the three methods is almost the same. But when the number of nodes increases, the cost of the three methods increases significantly to provide better service and faster response. However, the PSEC method can always require the lowest total cost, which is at least 20% less than the other two schemes. Through comprehensive comparison in Figure 6 and Figure 7, the PSEC method is less sensitive to the number of users and nodes. Thus, our PSEC can be applied to complex scenes and has strong robustness.

In Figure 8 and Figure 9, we reveal the relationship between the system performance and the parameter factor ε. According to the parameter setting in [13], we set $\varepsilon_1 = \varepsilon_2 = \varepsilon > 0$ for the MOERA and vary ε from 0.001 to 1000. In Figure 8, it can be seen that when $\varepsilon < 100$, the difference between the two methods on the total cost is not obvious. When ε is greater than 100, the cost gap between the two schemes has become very large. Therefore, the PSEC algorithm has stronger adaptability and robustness. In order to achieve a better performance on system cost, ε should preferably be a small positive number less than 10. In Figure 9, It is interesting to notice that with the increase of ε, the empirical competitive ratio of our algorithm declines sharply at the beginning and then decreases to a stable level. It can be proved that the competitive ratio is a monotonically decreasing function of ε. Please refer to the Appendix A for the detailed proof process. We observe that when $0.1 < \varepsilon < 10$, our algorithm can roughly achieve a stable yet reasonably good competitive ratio.

Figure 10 plots the total cost of different schemes with the increasing task size. Obviously, the proposed scheme is superior to other schemes and the gain is up to 40%. When the task size increases from 64 to 2048Mb, the cost of PSEC is always less than $2.5 * 10^6$, while the cost of MOERA increased to more than $16 * 10^7$. When the amount of data is greater than 100, the cost of MOERA shows a very obvious increase. PSEC can always maintain a relatively stable state in terms of cost with the increasing task size. This is because the task can be migrated from one edge node to another. Our

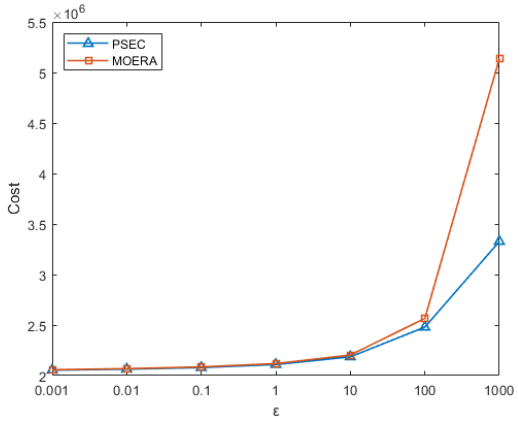


FIGURE 8. System cost versus ε.

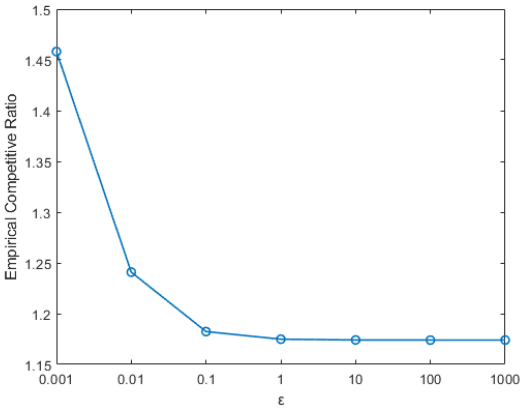


FIGURE 9. Competitive ratio versus ε of PSEC.

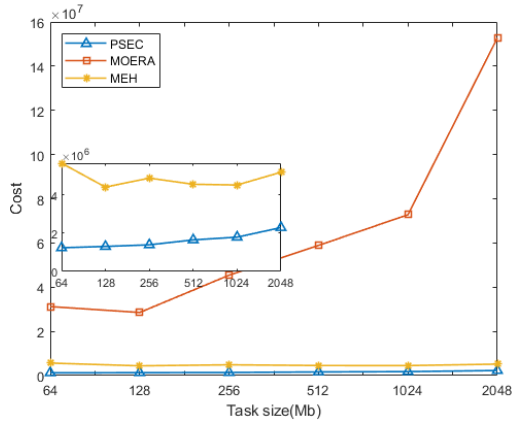


FIGURE 10. Total cost versus the task size under different schemes.

PSEC can always help users find the most suitable edge node for offloading and obtain a convergent and stable cost.

Figure 11 plots the total cost of different schemes with the increasing capacities of edge nodes. In this simulation, the number of users is fixed at 20. It can be seen that when the capacity is less than 10000Mb, the cost of our PSEC scheme is much less than that of MOERA, but slightly greater than

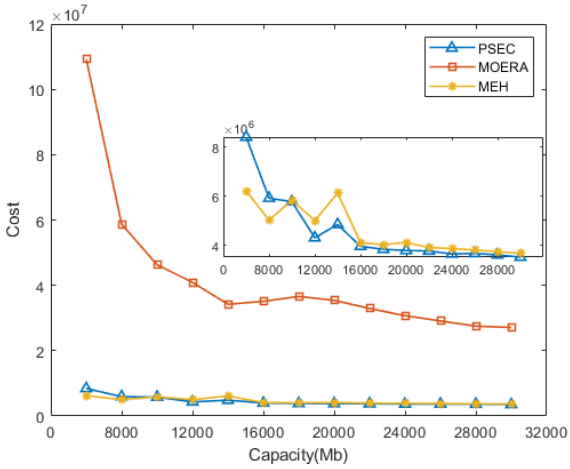


FIGURE 11. Total cost versus capacities under different schemes.

that of MEH, when the capacity is greater than 10000Mb, our scheme requires the lowest cost. Although the cost of MEH scheme is close to that of our scheme, it is always higher than that of our scheme. In fact, the capacity of many edge devices is more than 10000MB, so our PSEC scheme can achieve the best performance in actual operation.

VII. CONCLUSION

To solve the problem of task offloading in the resource-limited and position-sensitive multi-UAVs edge environment, a distributed location-aware task offloading scheme based on convex optimal analysis is proposed. Considering the limited energy of edge nodes and the random movement of users, the total cost is divided into operation cost, quality of service cost, and migration cost. A nonlinear cost optimization problem is constructed, and then the problem is transformed into a convex optimization problem with linear constraints based on regularization technology. The mathematical proof shows that the scheme can support a parameterized competitive ratio without prior knowledge of the input task. Experimental results show that the proposed scheme can achieve better performance with strong robustness.

APPENDIX A

This appendix illustrates the monotonicity proof of the competitive ratio.

Proof. We first review the following definitions:

$$\begin{aligned} r &= 1 + \gamma n_0, \\ \gamma &= \max\{(z_s + \varepsilon)\,\eta_s\}, \\ \eta_s &= \ln(1 + z_s/\varepsilon), \\ z_s &= \min\{Q_s/k, C_s\} \end{aligned}$$

To analysis the monotonicity competitive ratio r, we need to analyze γ which is related to the ε as well as z_s. For above equations, z_s is the minimum value of the array $\{Q_s/k, C_s\}$. Thus, a constant z_0 is existed which is defined by the

following formula:

$$z_0 = \arg\max_{z_s} \gamma.$$

Then, we have

$$\gamma = (z_s + \varepsilon)\,\eta_s = (z_0 + \varepsilon)\ln(1 + z_0/\varepsilon),$$

and

$$\begin{aligned}\gamma^{`} &= \ln(1 + z_0/\varepsilon) + (z_0 + \varepsilon)\frac{-z_0\frac{1}{\varepsilon^2}}{1 + \frac{z_0}{\varepsilon}} \\ &= \ln(1 + z_0/\varepsilon) - \frac{z_0}{\varepsilon} = \ln\left(1 + z_0'\right) - z_0',\end{aligned}$$

where, both $z_0' = \frac{z_0}{\varepsilon}$ and ε are positive numbers. According to the Taylor formula: $\ln(1 + x) = x - \frac{x^2}{2} + \frac{x^3}{3} \ldots$, we have

$$\ln\left(1 + z_0'\right) - z_0' \le 0, \quad z_0' > 0.$$

Therefore, we can get

$$\gamma^{`} = \ln(1 + z_0/\varepsilon) - \frac{z_0}{\varepsilon} < 0.$$

That is to say γ is a subtractive function of ε, and we can get that the competitive ratio r decreases with the increase of ε.

REFERENCES

[1] S. Shen, Y. Han, X. Wang, and Y. Wang, "Computation offloading with multiple agents in edge-computing–supported IoT," *ACM Trans. Sensor Netw.*, vol. 16, no. 1, pp. 1–27, Feb. 2020.

[2] F. A. Salaht, F. Desprez, and A. Lebre, "An overview of service placement problem in fog and edge computing," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–35, May 2021.

[3] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[4] Y. Liu, M. Peng, G. Shou, Y. Chen, and S. Chen, "Toward edge intelligence: Multiaccess edge computing for 5G and Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6722–6747, Aug. 2020.

[5] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Sep. 2016.

[6] G. Faraci, C. Grasso, and G. Schembra, "Fog in the clouds: UAVs to provide edge computing to IoT devices," *ACM Trans. Internet Technol.*, vol. 3, pp. 1–26, Aug. 2020.

[7] Y. Xu, T. Zhang, J. Loo, D. Yang, and L. Xiao, "Completion time minimization for UAV-assisted mobile-edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 12253–12259, Nov. 2021.

[8] H. Wang, J. Wang, G. Ding, J. Chen, F. Gao, and Z. Han, "Completion time minimization with path planning for fixed-wing UAV communications," *IEEE Trans. Wireless Commun.*, vol. 18, no. 7, pp. 3485–3499, Jul. 2019.

[9] C. Zhan and Y. Zeng, "Completion time minimization for multi-UAV-enabled data collection," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4859–4872, Oct. 2019.

[10] Y. Zeng, X. Xu, and R. Zhang, "Trajectory design for completion time minimization in UAV-enabled multicasting," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2233–2246, Apr. 2018.

[11] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, "A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4831–4843, Jun. 2019.

[12] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 1, no. 1, pp. 30–39, Jan. 2017.

[13] L. Wang, J. Lei, and J. Li, "MOERA: Mobility-agnostic online resource allocation for edge computing," *IEEE Trans. Mobile Comput.*, vol. 9, no. 8, pp. 1843–1856, Aug. 2019.

[14] Q. Wu, W. Me, and R. and Zhang, "Safeguarding wireless network with UAVs: A physical layer security perspective," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 12–18, Oct. 2019.

[15] G. Cui, Y. Xu, S. Zhang, and W. Wang, "Secure data offloading strategy for multi-UAV wireless networks based on minimum energy consumption," *J. Commun.*, vol. 42, no. 2, pp. 51–62, May 2021.

[16] J. Ji, K. Zhu, and C. and Yi, "Energy consumption minimization in UAV-assisted mobile-edge computing systems: Joint resource allocation and trajectory design," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8570–8584, Dec. 2021.

[17] Y. Wang, W. Fang, Y. Ding, and N. Xiong, "Computation offloading optimization for UAV-assisted mobile edge computing: A deep deterministic policy gradient approach," *Wireless Netw.*, vol. 27, no. 4, pp. 2991–3006, May 2021.

[18] P. Gopika, D. Mario, and T. Tarik, "Edge computing for the Internet of Things: A case study," *IEEE Internet Things J.*, vol. 5, no. 12, pp. 1275–1284, Feb. 2018.

[19] J. Pan and J. Mcelhannon, "Future edge cloud and edge computing for Internet of Things applications," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 439–449, Oct. 2018.

[20] K. Zhang, G. Lin, R. Wang, L. Jing, and R. Sheng, "Survey on computation offloading and content caching in mobile edge networks," *J. Softw.*, vol. 30, no. 8, pp. 2491–2516, 2019.

[21] N. Zhang, S. Guo, Y. Dong, and D. Liu, "Joint task offloading and data caching in mobile edge computing networks," *Comput. Netw.*, vol. 182, Dec. 2020, Art. no. 107446.

[22] S. Wang, M. Chen, X. Liu, C. Yin, S. Cui, and H. V. Poor, "A machine learning approach for task and resource allocation in mobile-edge computing-based networks," *IEEE Internet Things J.*, vol. 182, no. 3, pp. 6824–6836, Jul. 2020.

[23] E. el Haber, T. M. Nguyen, C. Assi, and W. Ajib, "An energy-efficient task offloading solution for MEC-based IoT in ultra-dense networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2019, pp. 1–7.

[24] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[25] D. Puthal, M. Obaidat, P. Nanda, M. Prasad, S. P. Mohanty, and A. Y. Zomaya, "Secure and sustainable load balancing of edge data centers in fog computing," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 60–65, May 2018.

[26] X. Xu, Q. Wu, W. Dou, and S. Tsai, "Trust-aware service offloading for video surveillance in edge computing enabled internet of vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1787–1796, Jun. 2021.

[27] S. Zhu, L. Gui, N. Cheng, Q. Zhang, F. Sun, and X. Lang, "UAV-enabled computation migration for complex missions: A reinforcement learning approach," *IET Commun.*, vol. 14, no. 15, pp. 2472–2480, Sep. 2020.

[28] D. Callegaro and M. Levorato, "Optimal edge computing for infrastructure-assisted UAV systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1782–1792, Feb. 2021.

[29] P. Amos, P. Li, W. Wu, and B. Wang, "Computation efficiency maximization for secure UAV-enabled mobile edge computing networks," *Phys. Commun.*, vol. 46, Jun. 2021, Art. no. 101284.

[30] G. Chen, Q. Wu, W. Chen, D. W. K. Ng, and L. Hanzo, "IRS-aided wireless powered MEC systems: TDMA or NOMA for computation offloading," 2021, *arXiv:2108.06120*.

[31] M. Hua, Y. Wang, C. Li, Y. Huang, and L. Yang, "UAV-aided mobile edge computing systems with one by one access scheme," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 3, pp. 664–678, Sep. 2019.

[32] R. Han, Y. Wen, L. Bai, J. Liu, and J. Choi, "Rate splitting on mobile edge computing for UAV-aided IoT systems," *IEEE Trans. Cognit. Commun. Netw.*, vol. 6, no. 4, pp. 1193–1203, Dec. 2020.

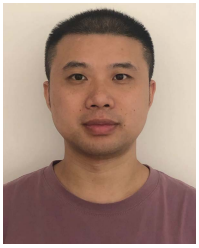
[33] *LTE Unmanned Aircraft Systems Trial Report*, Qualcomm, San Diego, CA, USA, 2017.

[34] U. Ugurlu, R. Wichman, C. B. Ribeiro, and C. Wijting, "A multipath extraction-based CSI acquisition method for FDD cellular networks with massive antenna arrays," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2940–2953, Apr. 2015.

[35] X. Wang, Z. Ning, and S. Guo, "Multi-agent imitation learning for pervasive edge computing: A decentralized computation offloading algorithm," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 2, pp. 411–425, Feb. 2021.

[36] M. U. Thomas, "Queueing systems," *SIAM Rev.*, vol. 3, no. 5, pp. 512–514, 1977.

[37] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn*, vol. 8, pp. 231–257, Nov. 2015.
[38] W. Li and S. Jin, "Performance evaluation and optimization of a task offloading strategy on the mobile edge computing with edge heterogeneity," *J. Supercomput.*, vol. 77, no. 11, pp. 12486–12507, Nov. 2021.



JIANHUA LIU received the Ph.D. degree from Beihang University, Beijing, China, in 2013. He is currently an Associate Professor with the School of Avionics and Electronics Engineering, Civil Aviation Flight University of China, Guanghan, China. His research interests include information security, the Internet of Things, and edge computing.

ZIBO WU was born in Greenwich, New York, NY, USA, in 1977. He received the B.S. degree in communication engineering from the Tianjin University of Technology, China, in 2019. He is currently pursuing the master's degree with the School of Avionics and Electronics Engineering, Civil Aviation Flight University of China. His research interests include mobile edge computing and cloud computing in wireless communications.

JIAJIA LIU received the master's degree in communication and information system from Sichuan University, in 2011. She is currently an Associate Professor with the Civil Aviation Flight Institute of China. Her research interests include edge computing and image processing.

XIAOGUANG TU received the Ph.D. degree from the University of Electronic Science and Technology of China (UESTC). He is currently an Associate Professor with the Civil Aviation Flight Institute of China. His research interests include convex optimization, computer vision, and deep learning.