Edge Computing Offloading at Middle-sea Scenario for Maritime Video Surveillances

Ziyang Gong[1], Ziyi Wang[2], Xin Su[2], Chang Choi[1*]

[1] Department of Computer Engineering, Gachon University, South Korea
[2] College of IoT Engineering, Hohai University, China
gongzy0123@163.com, wzy721617@163.com, leosu8622@163.com, enduranceaura@gmail.com

Abstract

Video surveillance has broad application prospects in maritime rescue, ship transportation and other fields. Mobile edge computing can effectively guarantee low-latency and highly reliable data transmission in maritime video surveillance. This paper comprehensively formulates the edge computing offloading schemes in the middle-sea scenario. The middle-sea scenario has sufficient edge computing nodes as in the offshore scenario and delay constraints due to limited network connectivity as in the far-sea scenario. Taking into account these characteristics, a single-user single-hop unicast offloading model is established and extends to a multi-user model. In addition, for the sufficient and limited edge computing nodes, the multi-user model is further divided into the multi-user single-hop unicast situation 1 and 2 models. We split the mixed-integer nonlinear programming problem and approximate the optimal transmission power by the binary search method. We use the offloading decision allocation algorithm based on alternating selection, offloading decision allocation algorithm based on multi-objective alternating selection, and offloading decision allocation algorithm based on node redistribution to optimize the offloading decisions of the above models. Subsequently, we analyze the simulation results from algorithm comparisons, changes in the number of subtasks, data, and oceanic user equipments. We verify the effectiveness of the proposed schemes and algorithms in saving delay.

Keywords: Maritime video surveillance, Maritime network, Mobile edge computing, Binary search algorithm, Multi-user offloading

1 Introduction

Advances in maritime information intelligent networks have been able to provide a variety of surveillance applications [1-3]. They can carry out all-weather, fully automatic maritime observation, target situational awareness, integrated maritime services, among other services. However, low-latency and high-reliability performance metrics are critical requirements for these services [4-5]. Mobile edge computing (MEC) technology can effectively fulfil these requirements [6-8]. Furthermore, it can offload the cloud data center to the edge of the networks and enable edge nodes to provide computing, storage, and communication capabilities closer to users [9-11].

Complex maritime surveillance applications can cause network data overload and sharp increases in network overhead. Thus, realizing flexible adaptation is a primary problem in maritime network (MN) resources. Compared with traditional terrestrial cellular and vehicle-mounted networks, MN has complex environmental factors and distinct node differences. In particular, the geographic location of the middle-sea scenario is between the offshore scenario and the far-sea scenario. The high density of offshore edge nodes, uneven distribution of edge computing capabilities, and complex resource scheduling need to be considered. In addition, it needs to be considered that the connectivity of the far-sea network is easily affected by factors such as weather and severe sea conditions. Therefore, it is difficult to guarantee the stability of maritime application services. How to propose the edge data offloading models and algorithms that adapt to the middle-sea scenario is also a problem that needs to be solved. In our study, we combine the previous works on MEC offloading technology [12-14] and the networking challenges of MN to meet the requirements of low-latency and high-reliability of maritime applications. This paper focuses on the research of MEC offloading technology in MN. This paper is organized as below:

In this study, we combined the characteristics of offshore and far-sea scenarios into MN. Furthermore, we considered the number of users and the adequacy of edge computing nodes. We introduce data splitting technology to establish the single-user single-hop unicast (SSU) and multi-user single-hop unicast models (MSUS1 and MSUS2) separately in the middle-sea scenario. The mixed-integer nonlinear programming (MINLP) problem proposed in this paper is split into two sub-problems. Furthermore, we used binary search to allocate transmission power effectively. However, the offloading decision optimization algorithm varies among the models. In the SSU model, we propose an offloading algorithm based on alternating selection (OAAS) to replace the subtask sets with strategies and compare in detail according to the corresponding priority. In the MSUS1 model, the OAAS is extended to include multi-objective alternating selection (OAMOAS) to the multi-user model. OAMOAS sets a priority function for multiple users to ensure objective selection.

The MSUS2 model considers the scenario where the subtask sets require nodes redistribution when the number of

*Corresponding Author: Chang Choi; E-mail: enduranceaura@gmail.com
DOI: 10.53106/160792642022112306021

edge computing nodes is limited. In addition, edge nodes with limited number of computing resources will cause considerable queuing delays if they are reused. Therefore, we propose a node redistribution offloading algorithm (OANR) that allocates edge nodes based on the proportion of user's data volume. The OANR makes the offloading decisions based on actual factors.

2 Related Works

MEC is the core technology of future mobile communications [15-18]. Combining the advantages of MEC and the edge computing features of MN, we propose a set of optimized offloading schemes for middle-sea scenario. However, the results of applying MEC to MNs to improve network performances are relatively scarce. Therefore, this paper focuses on analyzing the research results of the MEC offloading schemes based on the terrestrial scene.

2.1 Single-user Mobile Edge Computing

The single-user edge computing model can adapt well to the limitations of mobile terminals, bringing a better user experience by improving the utilization of computing and storage resources. [19] offloaded all the terminal data to an edge server and proposed offline strategies based on pre-calculations to optimize the scheduling and offloading of radio computing resources. The application of MEC technology can not only save energy but also has a considerable effect in reducing latency. [20] proposed an adaptive algorithm based on Lyapunov theory to optimize the weighted sum of delay and energy consumption. Their results showed that offloading schemes have improved latency and energy consumption compared to local processing. However, the above schemes treat the terminal data as a whole and can achieve either local processing or complete offloading. The overall efficiency of the system is still limited.

One data splitting and offloading scheme is dividing the data into multiple subtasks. Each subtask determines whether to offload. To meet the future 5G demands for low-latency processing, [21] studied the offloading decisions of each subtask and their computing resources. They proposed delay-optimal task scheduling (DOTS) algorithm for optimization. [22] further introduced reinforcement learning (RL) to optimize the execution time and energy consumption. However, the optimization goal should also include scheduling offloading, resource allocation, and other computational metrics along with the weighted delay and energy consumption. [23] studied the problem of minimizing the weighted sum of delay and energy consumption in a multi-subtasks MEC system. They proposed partial offloading decision and scheduling based on Johnson (POJ) to optimize offloading scheduling and resource allocation. However, for diversified maritime application requirements, the single-user model is not suitable for the envisioned smart oceans of the future. In the increasingly complex data migration environment, the rational allocation of resources needs to be further studied to improve user experience of the multi-user model.

2.2 Multi-user Mobile Edge Computing

[24] studied the joint optimization for the MEC networks. To solve the MINLP, they adopted the modified branch and bound (MBB) method to obtain accurate solutions, thereby reducing the delay. Regarding energy savings, [25] studied the offloading model of a multi-user single-server. They proposed select maximum saved energy first (SMSEF) and greedy selection algorithms to optimize channel and resource allocation. [26] applied time-division multiple access technology to a mobile edge computing multi-user offloading model. They proposed an optimization algorithm for communication and resource allocation to reduce delay. [27] introduced orthogonal frequency division multiple access (OFDMA) technology to optimize system overhead. However, with the surge in maritime applications, the multi-user single-server offloading models cannot meet the needs of users. [28] studied the offloading model between multiple users and servers. They proposed the Lagrange dual decomposition algorithm to optimize offload decision-making and computing resources. However, scaling offloading decisions lacks practical significance. [29] adopted the multiple-input multiple-output (MIMO) multi-cell system to minimize the overall energy consumption of the user while meeting the delay constraint. They proposed an iterative algorithm based on a continuous convex approximation to obtain sub-optimal solutions for offloading decision-making and resource allocation. [30] studied the offloading model between multiple users and servers. They introduced a new big data reinforcement learning method to optimize resource allocation and offloading strategies jointly; thus, improving the quality of experience for delay-sensitive users. [31] analyzed the problem of minimizing the weighted sum of delay and energy consumption, giving priority to user groups who could not complete tasks locally. Furthermore, they proposed the user equipment (UE) with largest saved energy consumption accepted first (CAR-E) and smallest required data rate accepted first (CAR-D) algorithms. However, user data in these models are inseparable, and the flexibility of offloading is poor. Subsequently, [32] divided the tasks and calculated the waiting delay of each subtask on the server. They proposed the Lyapunov algorithm to optimize dynamic multi-objectives, thereby reducing the overall delay of users. The offloading model and research content of [33] were the same as those of [32]. [33] proposed a resource scheduling algorithm that minimized the weighted sum of communication and calculation delays. [34] studied the problem of resource allocation based on quality of service (QoS). After reordering the tasks according to the delay tolerance, the reinforcement learning algorithm was used to allocate resources intelligently.

However, the existing multi-user edge computing offloading model constraints and specifications considerations are not comprehensive. An algorithm that relaxes the binary offloading decision variables to continuous variables is difficult to apply in practice. In addition, a heuristic algorithm ignores the calculation delay and energy consumption constraints of the servers are ignored. Furthermore, the multi-user, multi-server models can overlook the limited number of servers and their waiting delay. They cannot meet the needs of various applications of maritime observation monitoring sensory networks. Thus, this study applied the data division method of a single-user model to a multi-user model for the middle-sea scenario. Our results show that it is possible to develop a flexible partial offloading program. Furthermore, the designed models are suitable for MN with limited communication, computing resources, and network connection delays [35].

3 System Model

Figure 1 shows the edge computing-based offloading model on the maritime observation monitoring sensory networks in the middle-sea scenario. It is composed of oceanic user equipments (OUEs) and oceanic edge computing nodes (OECNs).

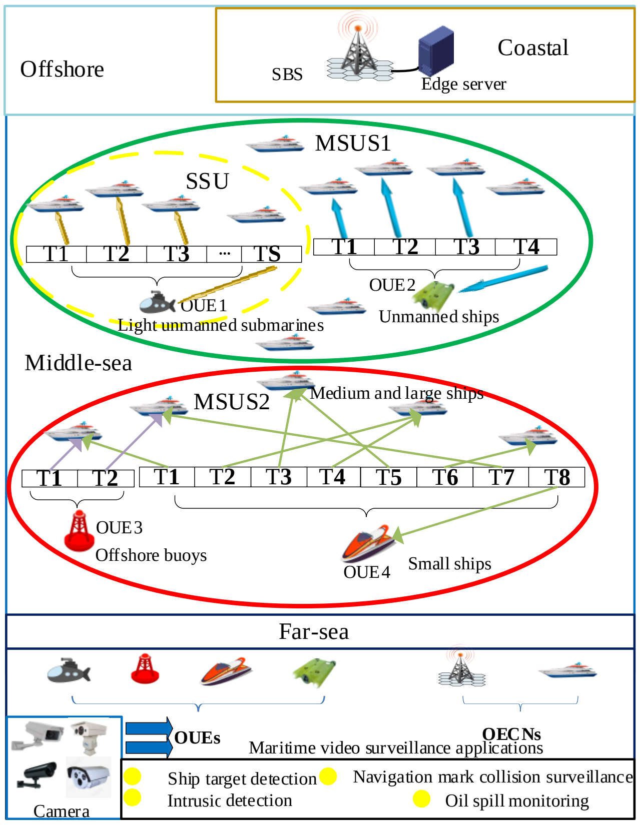


Figure 1. System model diagram of the middle-sea scenario

The OUEs have limited local computing power and signal coverage and are sensitive to energy consumption. They can perform maritime observation monitoring services in any weather and are fully automatic. OUEs include light unmanned submarines, offshore buoys, small ships, and unmanned ships. Light unmanned submarines can be used for port reconnaissance and surveillance, minefield detection, and seabed surveys of specific destinations. Offshore buoys can collect data for scientific research, offshore oil (gas) development, and port construction. Small ships have sensing and communication capabilities. They can perform hydrographic surveys and maritime environment monitoring and resource development. Unmanned ships help in shipwreck salvaging, deep-water exploration, and underwater cable laying operations. OUEs equipped with video surveillance cameras can execute the application in Figure 1. Ship target detection using video surveillance has attracted much attention in underwater cultural heritage protection and maritime aquaculture and transportation. Video surveillance of navigation marks ensures their safety. The safe and stable operation of navigation aids is the primary prerequisite for navigation aids to fully play the role of navigation aid, navigation, and warning. Maritime intrusion detection can be safely guaranteed for incidents such as small ships approaching oil platforms or illegally logging on to unguarded platforms. Oil spill monitoring is extremely important for preventing accidents in offshore oil and gas field exploitation. OECNs have sufficient internal energy, strong local computing capabilities, and extended signal coverage to perform maritime observation monitoring services effectively. OECNs include medium to large ships and seacoast base stations (SBS). Medium and large ships can be used as edge computing nodes to process real-time OUE node outsourcing tasks. They can also access the self-organized water surface networks to expand the coverage of the SBS networks. As an air interface, SBS can realize the link between the maritime stereoscopic monitoring equipment terminal and terrestrial cloud server. It can also be used as an edge server to support high-reliability and low-latency services effectively. The middle-sea scenario is far from the offshore scenarios. Therefore, OECNs include only medium and large ships. Figure 1 describes the real-time parallel execution of multi-task sets in the middle-sea scenario. A large amount of real-time data is generated in a local area of the maritime networks. These data can be processed locally or by fusion clustering and separately offloaded to nearby medium and large ships for simultaneous processing. Based on the above factors, such as the number of OUEs and the adequacy of OECNs, we establish SSU, MSUS1, and MSUS2 models. Compared to the far-sea scenario, the middle-sea scenario has a richer number of OECNs, considering the limited network connection delay. Therefore, the SSU model formulates a single-hop offloading mechanism between a single OUE and *M* OECNs, as shown in the yellow circle in Figure 1. We divide the to-be-processed data of a single OUE into *S* subtasks of varied sizes. The set of subtasks is denoted as $Task_i \in \{Task_1, Task_2, ..., Task_S\}$. Each subtask can be processed locally or offloaded to a nearby OECN for processing. To meet the network connection delay constraints and avoid resource competition, this paper stipulates that the subtasks have a one-to-one correspondence with OECNs. The demand of $Task_i$ can be represented by $J_{Task_i} = \left(D_{Task_i}, a_{Task_i}, E_{Task_i \max}\right), \forall Task_i$, where, D_{Task_i} is the data volume of $Task_i$, a is the average calculation density of $Task_i$, and $E_{Task_i \max}$ is the maximum energy consumption allowed by $Task_i$. We define variable $\boldsymbol{X} = \left\{x_{Task_i}, \forall Task_i\right\}$ as the pairing decision between the subtasks and OECNs. The MSUS1 model is based on the SSU model and considers the single-hop offloading mechanism between $\widehat{S}_K$ and $\widehat{M}$ OECNs among *K* OUEs. As shown in the green circle in Figure 1, any OUE offloading model can be regarded as an SSU model. In the communication range, multiple SSU models form a MSUS1 model. The above two offloading models assume that the number of OECNs is sufficient compared to the total number of subtasks to be processed. Multiple OECNs can complete the offloading of all subtasks in a single or multiple OUEs. Previous research on sub-task offloading decision assignment satisfies the condition that the number of edge computing nodes for selection is sufficient [36]. However, with the increasing OUE demands, this premise is difficult to achieve. Therefore, we establish an MSUS2 model suitable for the scenario where the number of OECN $\tilde{M}$ is limited compared with the number of $\tilde{K}$ OUE, where $\tilde{k} \in \{1,2,...,\tilde{K}\}$ and neutron tasks $\tilde{S}_{\tilde{k}} \in \{\tilde{S}_1, \tilde{S}_2, ..., \tilde{S}_{\tilde{K}}\}$, as shown in the red circle in Figure 1. We note that the number of OECNs composed of medium and large ships is significantly less than the total number of tasks divided by the two OUEs. The MSUS2 model stipulates that a sub-task can select only one OECN to offload at most. After the initial OECN assignment of the sub-task sets, all OECNs are assigned sub-tasks. There is no free OECN left. However, there could be unallocated subtasks that should be redistributed by OECNs. The optimal offloading decisions are determined by comparing the local processing and total offloading delays.

Based on the offloading decisions, we can obtain the local computing and computing offloading models for each subtask:

1) Local Computing Model: Taking the SSU model as an example, all other models are satisfied. $\forall Task_i$, $x_{Task_i} = 0$. In this step, energy consumption is given by $E^l_{Task_i} = l a f l_k^2 D_{Task_i}$ and delay is denoted as $t^l_{Task_i} = \frac{D_{Task_i} a}{fl_k}$. Here, fl_k is the local calculation frequency of OUE k and l depends on the coefficient of the processor chip structures.

2) Computing Offloading Model: Taking the SSU model as an example, if $\forall Task_i$ chooses to offload, the value of $Task_i$ is the corresponding OECN label, which is $x_{Task_i} = m, \forall m$. The uplink transmission rate of $Task_i$ on OECN m is: $r_{Task_i,m} = B\log_2\left(1 + \frac{p_{Task_i,m} g_m}{\sigma^2}\right)$, where B and $p_{Task_i,m}$ respectively represent the bandwidth and transmission power of $Task_i$ on OECN m, σ^2 represents Gaussian white noise on OECN m, and g_m represents the channel gain on OECN m. Therefore, the data transmission delay $t^r_{Task_i}$ and energy consumption $E^o_{Task_i,m}$ of $Task_i$ on OECN m can be expressed as:

$$t^r_{Task_i} = \frac{D_{Task_i}}{r_{Task_i,m}}, \forall Task_i, m, \quad \textbf{(1)}$$

$$E^o_{Task_i} = p_{Task_i,m} t^r_{Task_i}, \forall Task_i, m. \quad \textbf{(2)}$$

OECN m processing delay can be expressed as:

$$t^m_{Task_i} = \frac{D_{Task_i} a_{Task_i}}{f_m}, \forall Task_i, m, \quad \textbf{(3)}$$

where, f_m represents the calculation frequency of OECN m.

At time T, a subtask can only select at most one OECN offload. The offloading decision constraint condition is:

$$\sum_{m=1}^{M} I\left(x_{Task_i}, m\right) \le 1, \forall Task_i. \quad \textbf{(4)}$$

At time T, an OECN can only be selected by one sub-task. The offloading decision constraint condition is:

$$\sum_{Task_i=1}^{Task_S} I\left(x_{Task_i}, m\right) \le 1, \forall m. \quad \textbf{(5)}$$

Here I represents the XNOR function. In other words, the values of the two variables in the function are the same. The value of the function is either 1 or 0.

4 Problem Formulation and Analysis

For the SSU model, according to Formulas (1) and (3), the total delay of $Task_i$ offloading is calculated as $t^o_{Task_i} = t^r_{Task_i} + t^m_{Task_i}, \forall Task_i, m$. The total delay of the subtask is the smaller of the local processing and total offloading delays. In addition, there are M OECNs within the communication range that simultaneously process S subtasks in a single OUE. The overall OUE delay is the maximum of the total delays of all subtasks, that is, $t_{\max} = \max\left(\min\left(t^o_{Task_i}, t^l_{Task_i}\right) \forall Task_i, m\right)$.

Thus, this study aimed to reduce the overall OUE delay as part of the optimization goal, that is, $\min(t_{\max})$. The optimization problem can be expressed as:

$$\begin{aligned}
&\min_{\{\boldsymbol{X},\boldsymbol{P}\}} \left(t_{\max}\right) \\
\text{s.t.}\quad & \mathrm{C}_1: \ x_{Task_i} \in \{0,1,...,M\}, \forall Task_i \\
& \mathrm{C}_2: \sum_{m=1}^{M} I\left(x_{Task_i}, m\right) \le 1, \forall Task_i \\
& \mathrm{C}_3: \sum_{Task_i=1}^{Task_S} I\left(x_{Task_i}, m\right) \le 1, \forall m \\
& \mathrm{C}_4: \ I\left(x_{Task_i}, 0\right) E^l_{Task_i} + I\left(x_{Task_i}, m\right) E^o_{Task_i,m} \\
& \qquad\qquad \le E_{Task_i \max}, \forall Task_i, m \\
& \mathrm{C}_5: \ 0 \le p_{Task_i,m} \le p_{Task_i \max}, \forall Task_i, m
\end{aligned} \quad , \textbf{(6)}$$

Where $\boldsymbol{P} = \left\{p_{Task_i, x_{Task_i}}, \forall Task_i\right\}$ is the transmission power allocated by $Task_i$, $E_{Task_i \max}$ is the maximum energy consumption allowed by $Task_i$, $p_{Task_i \max}$ is the rated transmission power allowed by $Task_i$, and constraint C_1 is the value range of the offloading decisions. Constraints C_2 and C_3 ensure a one-to-one correspondence between subtasks and OECNs. Constraint C_4 ensures that the actual energy consumption of each subtask does not exceed its maximum energy consumption. Constraint C_5 ensures that the transmission power of each subtask does not exceed its rated power.

Similarly, from Formula (6), we can directly obtain the optimization problem of the MSUS1 model:

$$
\begin{aligned}
&\min_{\{\hat{X},\hat{P}\}}\left(\hat{t}_{k,\max}\right),\forall k,\widehat{Task}_{\hat{i}}\\
\text{s.t. } &\mathrm{C}_6: \hat{x}_{k,\widehat{Task}_{\hat{i}}}\in\{0,1,\dots,\hat{M}\},\forall k,\widehat{Task}_{\hat{i}}\\
&\mathrm{C}_7: \sum_{k=1}^{K}\sum_{\hat{m}=1}^{\hat{M}} I\left(\hat{x}_{k,\widehat{Task}_{\hat{i}}},\hat{m}\right)\le 1,\forall k,\widehat{Task}_{\hat{i}}\\
&\mathrm{C}_8: \sum_{k=1}^{K}\sum_{\widehat{Task}_{\hat{i}}=1}^{\widehat{Task}_{\hat{S}_k}} I\left(\hat{x}_{k,\widehat{Task}_{\hat{i}}},\hat{m}\right)\le 1,\forall k,\hat{m}\\
&\mathrm{C}_9: I\left(\hat{x}_{k,\widehat{Task}_{\hat{i}}},0\right)\hat{E}^{l}_{k,\widehat{Task}_{\hat{i}}}+I\left(\hat{x}_{k,\widehat{Task}_{\hat{i}}},\hat{m}\right)\hat{E}^{o}_{k,\widehat{Task}_{\hat{i}},\hat{m}}\\
&\qquad\qquad\le \hat{E}_{k,\widehat{Task}_{\hat{i}}\max},\forall k,\widehat{Task}_{\hat{i}},\hat{m}\\
&\mathrm{C}_{10}: 0\le \hat{p}_{k,\widehat{Task}_{\hat{i}},\hat{m}}\le \hat{p}_{k,\widehat{Task}_{\hat{i}}\max},\forall k,\widehat{Task}_{\hat{i}},\hat{m}
\end{aligned}
\quad. \textbf{(7)}
$$

The definition of the related parameters is similar to Formula (6), and the only difference is the number of OUEs.

The above-mentioned SSU and MSUS1 models satisfy the condition that the number of OECNs is sufficient for processing the total number of subtasks. However, current research on subtask offloading does not consider the limited number of edge computing nodes [37]. The MSUS2 model established in this paper aims to solve the above problems. After the subtask sets are matched with the available OECNs for the first time, some subtasks might not get a match. These subtasks are redistributed to OECNs after further processing. They compare their local processing delay with the total redistribution delay, and make optimal offloading decisions. Due to the limited number of computing resources of OECNs, the total delay of redistribution of these subtasks have to consider the waiting delay $\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}$ on the corresponding OECNs.

$$
\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}=\begin{cases}0, & \widetilde{Task}_{\tilde{i}}=\widetilde{Task}_{1}\\ \max\left(0,\tilde{t}^{\tilde{m}}_{\tilde{k},\widetilde{Task}_{\tilde{i}-1}}+\tilde{t}^{r}_{\tilde{k},\widetilde{Task}_{\tilde{i}-1}}-\tilde{t}^{r}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}\right), & \widetilde{Task}_{\tilde{i}}\ne\widetilde{Task}_{1}\end{cases},\forall\tilde{k},\widetilde{Task}_{\tilde{i}}
\quad. \textbf{(8)}
$$

Formula (8) shows that $\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}$ of the $\widetilde{Task}_{\tilde{i}}$ includes two parts. One part is the total offloading delay $\tilde{t}^{\tilde{m}}_{\tilde{k},\widetilde{Task}_{\tilde{i}-1}}+\tilde{t}^{r}_{\tilde{k},\widetilde{Task}_{\tilde{i}-1}}$ for the selected OECN to complete the previous subtask. The other part is the offloading transmission delay $\tilde{t}^{r}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}$ of $\widetilde{Task}_{\tilde{i}}$. If $\widetilde{Task}_{\tilde{i}}$ has been offloaded through OECN during the initial allocation, then $\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}=0$. If $\tilde{t}^{\tilde{m}}_{\tilde{k},\widetilde{Task}_{\tilde{i}-1}}+\tilde{t}^{r}_{\tilde{k},\widetilde{Task}_{\tilde{i}-1}}\le\tilde{t}^{r}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}$, then $\widetilde{Task}_{\tilde{i}}$ does not need to wait in a queue on the OECN, the same $\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}=0$. Otherwise, $\widetilde{Task}_{\tilde{i}}$ exists in $\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}$.

From this analysis, we conclude that the total offloading delay of each subtask is $\tilde{t}^{o}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}=\tilde{t}^{w}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}+\tilde{t}^{\tilde{m}}_{\tilde{k},\widetilde{Task}_{\tilde{i}}},\forall\tilde{k},\widetilde{Task}_{\tilde{i}}$, and the corresponding optimization problem is as follows:

$$
\begin{aligned}
&\min_{\{\tilde{X},\tilde{P}\}}\left(\tilde{t}_{\tilde{k},\max}\right),\forall\tilde{k},\widetilde{Task}_{\tilde{i}}\\
\text{s.t. } &\mathrm{C}_{11}: \tilde{x}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}\in\{0,1,\dots,\tilde{M}\},\forall\tilde{k},\widetilde{Task}_{\tilde{i}}\\
&\mathrm{C}_{12}: \sum_{\tilde{k}=1}^{\tilde{K}}\sum_{\tilde{m}=1}^{\tilde{M}} I\left(\tilde{x}_{\tilde{k},\widetilde{Task}_{\tilde{i}}},\tilde{m}\right)\le 1,\forall\tilde{k},\widetilde{Task}_{\tilde{i}},\tilde{m}\\
&\mathrm{C}_{13}: \sum_{\tilde{k}=1}^{\tilde{K}}\sum_{\tilde{S}_{\tilde{k}}} I\left(\tilde{x}_{\tilde{k},\widetilde{Task}_{\tilde{i}}},\tilde{m}\right)\le \tilde{S}_{\tilde{k}},\forall\tilde{k},\widetilde{Task}_{\tilde{i}},\tilde{m}\\
&\mathrm{C}_{14}: I\left(\tilde{x}_{\tilde{k},\widetilde{Task}_{\tilde{i}}},0\right)\tilde{E}^{l}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}+I\left(\tilde{x}_{\tilde{k},\widetilde{Task}_{\tilde{i}}},\tilde{m}\right)\tilde{E}^{o}_{\tilde{k},\widetilde{Task}_{\tilde{i}},\tilde{m}}\\
&\qquad\qquad\le \tilde{E}_{\tilde{k},\widetilde{Task}_{\tilde{i}}\max},\forall\tilde{k},\widetilde{Task}_{\tilde{i}},\tilde{m}\\
&\mathrm{C}_{15}: 0\le \tilde{p}_{\tilde{k},\widetilde{Task}_{\tilde{i}},\tilde{m}}\le \tilde{p}_{\tilde{k},\widetilde{Task}_{\tilde{i}}\max},\forall\tilde{k},\widetilde{Task}_{\tilde{i}},\tilde{m}
\end{aligned}
\quad. \textbf{(9)}
$$

Constraint C11 is the value range of the offloading decisions. Constraint C12ensures that each subtask in an OUE can select only one OECN to complete the offloading. Constraint C13ensures that an OECN can serve multiple subtasks. Constraint C14 ensures that the actual energy consumption of each subtask does not exceed its maximum energy consumption. Constraint C15 ensures that the transmission power of each subtask does not exceed its rated power.

5 Heuristic Algorithm

5.1 Problem Analysis

Problems (6), (7), and (9), including the discrete offloading decision and continuous transmission power variables, are MINLP problems. In this paper, we divide these complex problems into two sub-problems: transmission power allocation and offloading decision optimizations, then solve them separately.

Optimizing transmission power allocation depends the offloading. Taking the SSU model as an example, we assume that all subtasks are offloaded, that is, $x_{Task_i}=m,\forall Task_i$. The optimal transmission power is only related to the transmission delay. Therefore, the simplified Formula (6) is:

$$
\begin{aligned}
&\min_{\{P\}} t^{r}_{Task_i},\forall Task_i\\
\text{s.t. } &\mathrm{C}_{16}\text{: } E^{o}_{Task_i,m}\le E_{Task_i\max},\forall Task_i,m \text{ and } \mathrm{C}_5
\end{aligned}
\quad. \textbf{(10)}
$$

Let $\lambda_{Task_i}=\dfrac{1}{t^{r}_{Task_i}},\forall Task_i$, then we get:

$$
\begin{aligned}
&\max\lambda_{Task_i},\forall Task_i\\
\text{s.t. } &\mathrm{C}_{17}\text{: } H_{Task_i}\left(\lambda_{Task_i}\right)\ge 0,\forall Task_i ,\\
&\mathrm{C}_{18}\text{: } \lambda_{Task_i}\le\lambda_{Task_i\max},\forall Task_i
\end{aligned}
\qquad \textbf{(11)}
$$

where $H_{Task_i}\left(\lambda_{Task_i}\right)=E_{Task_i\max}\lambda_{Task_i}-\dfrac{\sigma^2}{g_m}\left(2^{\frac{D_{Task_i}}{B}\lambda_{Task_i}}-1\right)$, $\lambda_{Task_i\max}=\dfrac{B}{D_{Task_i}}\log_2\left(1+\dfrac{p_{Task_i\max}g_m}{\sigma^2}\right),\forall Task_i,m$.

Taking the derivative of $H_{Task_i}\left(\lambda_{Task_i}\right)$, we have:

$$H_{Task_i}{}'\left(\lambda_{Task_i}\right) = E_{Task_i\max} - \frac{\sigma^2 2^{\frac{D_{Task_i}}{B}\lambda_{Task_i}} \ln 2 D_{Task_i}}{g_m B}, \forall Task_i, m. \quad \textbf{(12)}$$

Formula (12) shows that $H_{Task_i}\left(\lambda_{Task_i}\right)$ is a unimodal function with a maximum value. Then the optimal solution of the objective function of Formula (12) is the smaller value between the non-zero zero point of $H_{Task_i}\left(\lambda_{Task_i}\right)$ and $\lambda_{Task_i\max}$. $H_{Task_i}\left(\lambda_{Task_i}\right)=0$ is difficult to solve directly, and [11] used the extreme points of the equation to replace the above-mentioned non-zero zero, the optimization effect is not obvious. Therefore, this study used binary search to approximate the optimal value $\lambda^*_{Task_i}$ of $H_{Task_i}\left(\lambda_{Task_i}\right)=0$. Thus, the optimized transmission power $p^*_{Task_i,m}$ is obtained, and the expression is shown as:

$$p^*_{Task_i,m} = \frac{\sigma^2}{g_m}\left(2^{\frac{D_{Task_i}}{B}\lambda^*_{Task_i}} - 1\right), \forall Task_i, m. \quad \textbf{(13)}$$

The pseudocode of the transmission power allocation algorithm is shown in Algorithm 1.

Algorithm 1. Transmission power allocation algorithm based on binary search

```
Input: $S$, $M$, $D$, $p_{max}$, $E_{max}$, $f$, $\frac{g}{\sigma^2}$, $B$, $\lambda_{lower} = 0.001$,
       $e = 0.05$
Output: $p^*_{Task_i}$

1: for $i = 1: S$ for $j = 1: M$
2:  $\boldsymbol{\lambda}_{upper}(i,j) = \frac{B\log_2(1+\frac{p_{max}\boldsymbol{g}(j)}{\sigma^2})}{\boldsymbol{D}(i)}$
3:  $\boldsymbol{\lambda}_{best}(i,j) = erfensousuo(\lambda_{lower}, \lambda_{upper}(i,j), e)$
4: end end
5: $\boldsymbol{p}^* = \frac{\sigma^2(2^{\frac{D\lambda_{best}}{B}}-1)}{\boldsymbol{g}}$ Step 3 function specific operation
process:
6:  $\boldsymbol{\lambda}(i,j) = \frac{\lambda_{lower}+\boldsymbol{\lambda}_{upper}(i,j)}{2}$,
    $\boldsymbol{\vartheta}(i,j) = E_{max}\boldsymbol{\lambda}(i,j) - \frac{\sigma^2(2^{\frac{D(i)\lambda(i,j)}{B}}-1)}{\boldsymbol{g}(j)}$
7:  $\boldsymbol{\vartheta}1(i,j) = E_{max}\boldsymbol{\lambda}(i,j) - \frac{\sigma^2(2^{\frac{\boldsymbol{D}(i)\lambda_{lower}}{B}}-1)}{\boldsymbol{g}(j)}$
8: if $(\boldsymbol{\vartheta}(i,j) = \boldsymbol{\vartheta}1(i,j) < 0)$ $q = \boldsymbol{\lambda}(i,j) - \lambda_{lower}$
9:    if $(q > e)$ $\boldsymbol{\lambda}_{upper}(i,j) = \boldsymbol{\lambda}(i,j)$
10:  $\boldsymbol{\lambda}_{best}(i,j) = erfensousuo(\lambda_{lower}, \boldsymbol{\lambda}_{upper}(i,j), e)$
11: else $\boldsymbol{\lambda}_{best}(i,j) = \boldsymbol{\lambda}(i,j)$ end
12: else $q = \boldsymbol{\lambda}_{upper}(i,j) - \boldsymbol{\lambda}(i,j)$
13: if $(q > \hat{e})\lambda_{lower} = \boldsymbol{\lambda}(i,j)$
14:  $\boldsymbol{\lambda}_{best}(i,j) = erfensousuo(\lambda_{lower}, \boldsymbol{\lambda}_{upper}(i,j), e)$
15: else $\boldsymbol{\lambda}_{best}(i,j) = \boldsymbol{\lambda}(i,j)$ end end
```

5.2 Offloading Decision Allocation

5.2.1 Offloading Decision Allocation Algorithm Based on Alternating Selection (OAAS)

This section proposes OAAS to allocate OECNs to the SSU model effectively. The initial idea is to allocate the OECN offloading having the best communication and computing resources to the subtask with the highest priority in each iteration. This selection process is two-way. Subtask priority judgment rule: the greater the local processing delay of the subtask, the easier it is to reduce the delay through offloading, and the higher the offloading priority. OECN priority judgment rule: by comparing the total offloading delay of a fixed subtask on each OECN, the smaller the total offloading delay, the greater the OECN priority. First, we sort the subtask sets and set the OECNs based on the corresponding priority rules, specifically taking $x_{Task_i} = i, i \in [1,5]$ as an example. $Task_1$ with the highest priority is offloaded to OECN1 with the highest priority. This strategy saves delay for some high-priority subtasks. However, this does not guarantee reduction of the delay of the entire OUE. Therefore, we must analyze different scenarios, ignoring the local processing ability.

S1: $t^o_{Task_1}(1) \geq t^o_{Task_2}(2) \geq t^o_{Task_3}(3) \geq t^o_{Task_4}(4) \geq t^o_{Task_5}(5)$

The subscript of $Task$ represents the priority of the subtask after sorting. $(\ )$ is the priority of OECN after sorting. $t^o_{Task_1}(1)$ represents the total offloading delay of $Task_1$ processed by OECN1. This scenario is ideal. Each iteration can successfully match the current subtask based on its priority to the corresponding OECN having the best communication and computing resources. Compared with other allocation methods, the total offloading delay of OUE is small and only depends on $t^o_{Task_1}(1)$.

S2: $t^o_{Task_1}(1) \geq t^o_{Task_2}(2) \geq t^o_{Task_3}(3) \geq t^o_{Task_4}(4) \leq t^o_{Task_5}(5)$. As the size relationship between $t^o_{Task_1}(1)$ and $t^o_{Task_5}(5)$ cannot be determined, it is not conducive to optimizing the overall OUE delay.

The main idea of OAAS is to transform S2 into an ideal S1. We carry out strategy replacement and step-by-step comparison of $Task_5$ to solve the problem of not determining the relationship between $t^o_{Task_1}(1)$, $t^o_{Task_2}(2)$, $t^o_{Task_3}(3)$, and $t^o_{Task_5}(5)$ in S2. Specific improvement measures taken are as follows: first, we compare the size relationship between $\max\left(t^o_{Task_4}(5), t^o_{Task_5}(4)\right)$, and $t^o_{Task_5}(5)$ after the strategy change and judge whether the $t^o_{Task_5}(5)$ partial offloading delay can be reduced. $\max\left(t^o_{Task_4}(5), t^o_{Task_5}(4)\right) \leq t^o_{Task_5}(5)$ implies that the strategy transformation saves more local delay at current $t^o_{Task_5}(5)$. $\max\left(t^o_{Task_4}(5), t^o_{Task_5}(4)\right)$ continues to compare with $t^o_{Task_3}(3)$, $t^o_{Task_2}(2)$, and $t^o_{Task_1}(1)$ stepwise to determine whether a new round of strategy replacement can be performed. Consequently, we get the optimal offloading decision of $t^{o}_{Task_5}(5)$. In addition, $\max\left(t^{o}_{Task_4}(5), t^{o}_{Task_5}(4)\right) > t^{o}_{Task_5}(5)$ implies that the current $Task_5$ cannot be replaced by strategy with OECN4. $Task_5$ continues to conduct strategy replacement and step-by-step comparison of the subtasks that have made the offloading decision. Algorithm 2 shows the pseudo-code of OAAS.

Algorithm 2. Offloading decision allocation algorithm based on alternating selection (OAAS)

```
Input: $S, M, \boldsymbol{D}, \boldsymbol{p}_{max}, \mathrm{E}_{\max}, \boldsymbol{f}, \frac{g}{\sigma^2}, B, \boldsymbol{t^o}, \boldsymbol{t^l}$
Output: $\boldsymbol{X}$

1: $[\boldsymbol{t^l}, \boldsymbol{xuhao}]$=sort($\boldsymbol{t^l}$, 'descend')
2: The $\boldsymbol{p}^*$ obtained by Algorithm 1 is prioritized
$\boldsymbol{p}^* = \boldsymbol{p}^*(\boldsymbol{xuhao}, :)$
3: $\boldsymbol{t^o} = \frac{\boldsymbol{D}(\boldsymbol{xuhao})}{B\log_2\left(1+\boldsymbol{p}^*\frac{g}{\sigma^2}\right)} + \frac{\boldsymbol{D}(\boldsymbol{xuhao})\boldsymbol{a}(\boldsymbol{xuhao})}{\boldsymbol{f}}$
4: Sort OECN  $[\boldsymbol{aa}, \mathbf{iaa}]$=sort($\boldsymbol{t^o}(1, :)$)
5: if $\mathbf{aa}(1) > \boldsymbol{t^l}(1)$  $\boldsymbol{x}(1)$=0 else $\boldsymbol{x}(1) = \boldsymbol{iaa}(1)$ end
6: for $i$=2:$S$  $y$=sum($\boldsymbol{x}$~=0)+1  $\mathbf{x}(i) = \boldsymbol{iaa}(y)$
    Ideal situation:
7: if $\boldsymbol{t^o}(i, \boldsymbol{iaa}(y))$<=$\boldsymbol{t^l}(i)$ &&$\boldsymbol{t^o}(i, \boldsymbol{iaa}(y))$< $\boldsymbol{t^o}(i\text{-}1, \boldsymbol{x}(i\text{-}1))$
8: $\boldsymbol{x}(i) = \boldsymbol{iaa}(y)$  Non ideal situation:
9: else if $\boldsymbol{t^o}(i, \boldsymbol{iaa}(y))$<=$\boldsymbol{t^l}(i)$&&$\boldsymbol{t^o}(i, \boldsymbol{iaa}(y)) \geq \boldsymbol{t^o}(i\text{-}1, \boldsymbol{x}(i\text{-}1))$
Strategy transformation: $j = i$  $\boldsymbol{x}(j) = \boldsymbol{iaa}(y)$ $ii = i$
10: while ($ii$~=1) if $\boldsymbol{t^o}(ii\text{-}1, \boldsymbol{iaa}(y))$<= $\boldsymbol{t^o}(i, \boldsymbol{iaa}(y))$
11: while ($j$~ = 1) if $\boldsymbol{t^o}(j\text{-}1, \boldsymbol{x}(j))$< $\boldsymbol{t^o}(j, \boldsymbol{x}(j))$
    xx=$\boldsymbol{x}(j)$ $\boldsymbol{x}(j) = \boldsymbol{x}(j-1)$ $\boldsymbol{x}(j-1)$=xx $j = j-1$
12: else $j = j-1$ end  end break
13: else $ii = ii-1$ Try to find other tasks for policy change.  end end
14: else $\boldsymbol{x}(i)$=0  end end
```

5.2.2 Offloading Decision Allocation Algorithm Based on Multi-objective Alternative Selection (OAMOAS)

The SSU and MSUS1 models satisfy the condition that the total number of subtasks to be processed is less than the total number of assignable OECNs. Therefore, OAMOAS for the MSUS1 model is an extension of OAAS for the multi-user scenarios. The optimization goal of the MSUS1 is to reduce the overall delay of multiple OUEs. Due to the significant differences between OUEs, OUEs must be prioritized during OECN matching to meet optimization requirements. In this section, we determine the priority order of OUEs by comparing the total local processing delay of each OUE. If the OUE with high local processing delay selects the OECNs that have excellent computing and communication resources, the offloading effect of saving delay will be clearer. Consequently, once the priority is assigned, the real-time update of the relevant parameters is completed. Then, OAAS is performed on each OUE to obtain the optimal offloading decision of each subtask. The selected OECN must be removed to ensure the one-to-one correspondence between subtasks and OECNs. The pseudo-code of OAMOAS is shown in Algorithm 3.

Algorithm 3. Offloading decision allocation algorithm based on multi-objective alternative selection (OAMOAS)

```
Input: $K, \hat{M}, \hat{\boldsymbol{t}}^{\boldsymbol{o}}, \hat{\boldsymbol{t}}^{\boldsymbol{l}}$
Output: $\hat{\boldsymbol{X}}$

Determine the offloading priority of each OUE
1: [~, $\boldsymbol{por}$]= sort($\hat{\boldsymbol{t}}^{\boldsymbol{l}}$, 'descend')
2: Update relevant parameters
3: for $k$=1:$K$
4:   The $\boldsymbol{p}^*$ obtained by Algorithm 1
5: end
6:   Then we can get $\hat{\boldsymbol{t}}^{\boldsymbol{o}}$
7:   for $k$=1:$K$
8:   Complete the offloading decision allocation according to Algorithm 2
9:   Update OECN $\hat{M} = \hat{M} - \hat{\boldsymbol{x}}\{k\}$
10: end
```

5.2.3 Offloading Decision Allocation Algorithm Based on Node Redistribution (OANR)

Due to the limited number of edge computing nodes for selection, the MSUS2 model cannot process all tasks in OUEs simultaneously. Therefore, it is necessary to redistribute the OECNs, for which we propose OANR. The specific operation process is as follows:

First, we determine the offloading priority of each OUE by comparing the total local processing delay of each OUE. Simultaneously, we determine the OECN priority by comparing the total delay of the OUE offloaded by each OECN. Subsequently, based on the limited number of OECNs, OECN resources are divided so that a single OUE can obtain a certain number of OECNs to complete the offloading. The number of allocated OECNs is determined according to the proportion of the total amount of data to be processed in a single OUE in the total amount of all OUE data. This ensures that the OUEs do not interfere with each other and that fairness is maintained Let $m^{Number}_{\tilde{k}}$ denote the number of OECNs that OUE $\tilde{k}$ can use for offloading (Formula (14)). Subsequently, we can obtain the value range of the OUE's offloading decisions. Furthermore, there is a non-negligible queuing delay from the current subtask to the corresponding OECN, except for the first assignment of the OECN. Therefore, it is also necessary to obtain the number of times the OUEs need to perform nodes reallocation according to Formula (15). The OUE neutron subtask sets are prioritized according to the local processing delay. Subtasks with high priority should be offloaded by OECNs first. It minimizes the queuing delay of the subtasks that are strongly required to be offloaded on OECNs. Finally, the subtask and OECN sets selected by OUE in each iteration are offloaded and distributed through OAAS. The pseudo-code of OANR is shown in Algorithm 4.

Algorithm 4. Offloading decision allocation algorithm based on node redistribution (OANR)

```
Input: $\widetilde{K}$, $\tilde{S}$, $\widetilde{M}$, $\tilde{\boldsymbol{t}}^{\boldsymbol{o}}$, $\tilde{\boldsymbol{t}}^{\boldsymbol{l}}$
Output: $\widetilde{\boldsymbol{X}}$

1: Sort OUE[~, $\boldsymbol{p\tilde{o}r}$]= sort($\tilde{\boldsymbol{t}}^{\boldsymbol{l}}$, 'descend')
2: Sort OECN. Update relevant parameters.
3: Develop OECN allocation scheme according to priority
4: $m_{\tilde{k}}^{Number}$ is determined according to Formula (14)
5: Update OECN value range:
6: for $\tilde{k} - 1 : \widetilde{K}$ if $\tilde{k} \sim = 1$ $OECN_{\tilde{k}min} = OECN_{\tilde{k}-1min}$
   + $m_{\tilde{k}-1}^{Number}$ $OECN_{\tilde{k}min}$ is the lower limit of $OECN$
   obtain by OUE $\tilde{k}$
7: else $OECN_{\tilde{k}min}$ = 1 end
8: $OECN_{\tilde{k}max} = OECN_{\tilde{k}min} + m_{\tilde{k}}^{Number} - 1$ end
9: for $\tilde{k} = 1 : \widetilde{K}$
10: Sort each subtask in OUE $\tilde{k}$ and update data
11: According to Formula (15), $\boldsymbol{O}(\tilde{k})$ is obtained. end
12: for $\tilde{k} = 1 : \widetilde{K}$ for $\tilde{\jmath} = 1 : \boldsymbol{O}(\tilde{k})$ Execute Algorithm 2
end end
```

$$m_{\tilde{k}}^{Number} = \widetilde{\boldsymbol{M}} * \frac{\sum_{\widetilde{Task}_{\tilde{i}} = \widetilde{Task}_1}^{\widetilde{Task}_{\tilde{S}_{\tilde{k}}}} \tilde{D}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}}{\sum_{\tilde{k}=1}^{\widetilde{K}} \sum_{\widetilde{Task}_{\tilde{i}} = \widetilde{Task}_1}^{\widetilde{Task}_{\tilde{S}_{\tilde{k}}}} \tilde{D}_{\tilde{k},\widetilde{Task}_{\tilde{i}}}}, \forall \tilde{k} . \tag{14}$$

$$\boldsymbol{O}(\tilde{k}) = \frac{\tilde{\boldsymbol{S}}(\tilde{k})}{m_{\tilde{k}}^{Number}}, \forall \tilde{k} . \tag{15}$$

6 Simulation Results

This section focuses on the experimental simulations of the algorithms proposed to verify the effectiveness of the design schemes and algorithms in reducing delay.

6.1 Setting SSU Model Simulation Parameters and Analyzing Results

Setting S = 6, M = [10, 15, 20, 25, 30, 35, 40]. In other words, there are six simultaneous subtasks sending offloading requests to various MEC servers in OECNs. SSU model parameter values are shown in Table 1.

We compare the performance of OAAS with the following proposed algorithms. Algorithm 1 assigns the OECN with the best channel condition and computing resources to the subtask with the highest priority in the current iteration for processing [18]. Algorithm 2 ignores the priorities of subtasks and executes Algorithm 1 according to the initial order of each subtask. In addition, we compare our results with artificial fish swarm algorithm (AFSA) [38], particle swarm optimization algorithm (PSO) [39], and ISS-AFSA [18]. Figure 2 describes the delay comparison of a single OUE on different OECNs. The delay of a single OUE decreases with the increase of the number of OECNs. Results show that the greater the number of OECNs, the greater is the chance that the subtask selects OECN with excellent channel conditions and computing resources for offloading. Thus, it is easier to save delays. In addition, Figure 2 compares the delay of a single OUE using different algorithms. Algorithm 1 can effectively reduce the offloading delay of subtasks in poor conditions. However, the overall delay of OUE is not considered. Compared with Algorithm 1, OAAS considers the delay of each subtask and the overall delay of OUE. Executing strategy replacement between the current and the high priority subtasks can further reduce the overall delay of OUE. Algorithm 2 ignores the priorities between subtasks. Therefore, subtasks with high priorities are offloaded to OECNs with low priorities. The overall delay of OUE is the highest compared with other algorithms.

Table 1. SSU model simulation parameters

Parameter	Value
Average calculated density a	[500, 1000] cycles/bit
Mobile nodes computing frequency f	[10, 20] GHZ
Maximum energy consumption of subtasks E_{max}	4 J
Rated power of subtasks $p_{\max}$	0.4 W
Local computing frequency fl	0.01 GHZ
Channel bandwidth B	1 MHZ
Channel gain to noise ratio g/σ^2	[0.01, 0.2] W^{-1}
Task data volume D	[2, 8] KB

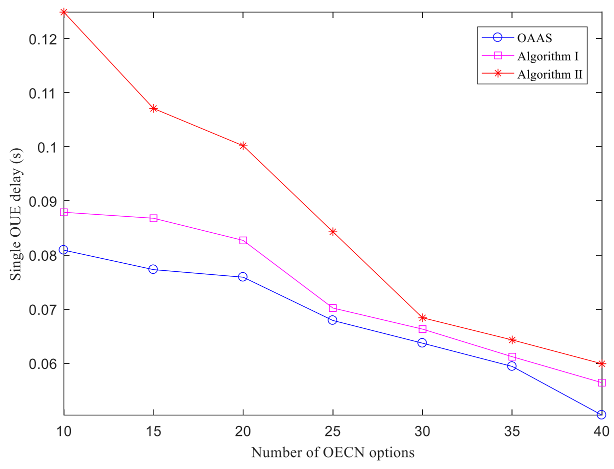


Figure 2. Comparison of OUE latency for the selected OECNs and algorithms

We take M =30, subtask data volume $\boldsymbol{D}$ (KB) = [2,8], [9, 15], [16, 22], [23, 29], [30, 36], S = 6 as an example. Figure 3 describes the OUE delay comparison for different subtask data volumes. When the number of OECNs is sufficient, the OUE delay increases with the increase in the volume of subtask data. Figure 3 compares OAAS with ISS-AFSA, AFSA, and PSO algorithms. OAAS can find the optimal offloading decision allocation by comparing each subtask in stages and replacing strategies. We note that to improve the performance of other algorithms, they have to randomize the offloading strategies through continuous iterations. As there is no optimal resolution for the iterative convergence of parameter settings, the optimal solution is difficult to achieve. Among the discussed algorithms, AFSA and PSO have relatively constant parameter settings, limiting their abilities to optimize locally. However, we note that using step-size randomization process, ISS-AFSA has marginally improved its local optimization. In addition, because OAAS does not need to perform iterative convergence, it can reduce the program execution delay.

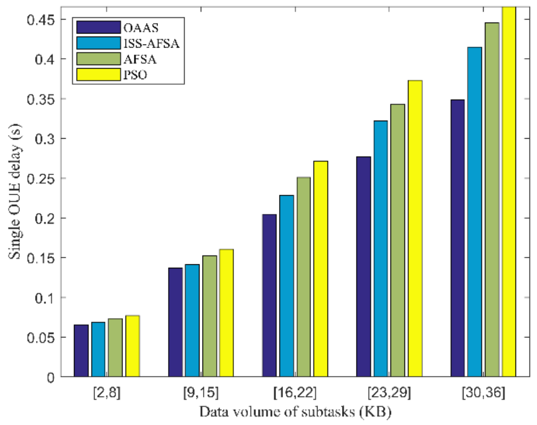


Figure 3. Comparison of OUE latency for different OECN data volumes and algorithms

We take $M = 30$, $S = [6, 8, 10, 12]$ as an example. Figure 4 depicts the comparison of OUE delay for different numbers of subtasks. Results show that the OUE delay increases as the number of subtasks increases because, with every iteration, the unassigned subtasks can be offloaded to an OECN with a relatively low priority. Thus, the offloading delay of each of these subtasks contributes to the total OUE delay. Therefore, the overall OUE delay continues to increase. Figure 4 compares the OUE delays for different algorithms. Results show that OAAS saves more time when dealing with large number of subtasks than other algorithms. We note that other algorithms may require many iterations to converge. Thus, the more the subtasks, the easier to fall into an optimal local situation. Therefore, saving delay can be difficult.

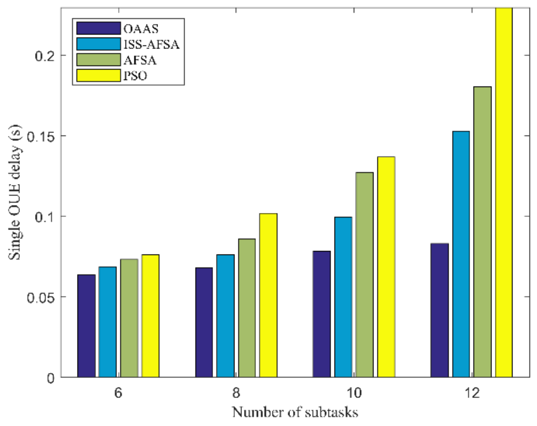


Figure 4. Comparison of OUE delays for different number of subtasks and algorithms

6.2 MSUS1 Model Experiment Parameter Setting and Result Analysis

Setting $K = [1, 2, 3, 4, 5], \widehat{M} = 30$. In other words, the subtasks in K OUEs simultaneously send offloading requests to the MEC servers in thirty OECNs. The parameter values of the MSUS1 model are shown in Table 2.

Table 2. MSUS1 model simulation parameters

Parameter	Value
Average calculated density $\widehat{a}$	[500, 1000] cycles/bit
Mobile nodes computing frequency $\widehat{f}$	[10, 20] GHZ
Maximum energy consumption of subtasks $\widehat{E}_{max}$	4 J
Rated power of subtasks $\widehat{p}_{\max}$	[0.4, 0.6] W
Number of subtasks $\widehat{S}$	[2, 5]
Local computing frequency $\widehat{fl}$	[0.01, 0.1] GHZ
Channel bandwidth $\widehat{B}$	1 MHZ
Channel gain to noise ratio $\widehat{g}/\widehat{\sigma}^2$	[0.01, 0.2] W^{-1}
Task data volume $\widehat{D}$	[1, 10] KB

Figure 5 compares the average delay of different numbers of OUEs. Figure 5 shows that the average delay of multiple OUEs increases as the number of OUEs increases. When the number of OECNs is sufficient, the more the OUEs to be offloaded, the greater the demand for OECNs. Thus, it is easier for subtasks to choose OECNs with relatively poor communication and computing resources for offloading. Therefore, the average delay of multiple OUEs continues to rise. In addition, Figure 5 compares the average delay of multiple OUEs using different algorithms. Experimental results show that OAMOAS saves more node allocation delay as compared to the AFSA and PSO algorithms that require considerable iterative convergence. When $K = 1$, $K = 2$, $K = 3$ with sufficient number of OECNs, the difference in OUE offloading delay with and without priorities is marginal. As the number of OUEs increases, the number and quality of the selected OECNs continue to decrease. The OUE priority assignment is particularly important in cases where $K = 4$, $K = 5$. We conclude that OAMOAS can cope well with a large number of OUEs.

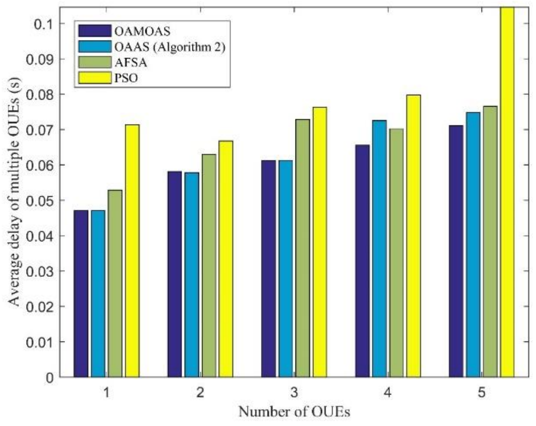


Figure 5. Comparison of OUE delays for different numbers and algorithms

6.3 Setting MSUS2 Model Simulation Parameters and Analyzing Results

Setting $\widetilde{K} = [4, 5, 6, 7]$, $\widetilde{M} = 10$. In other words, different numbers of subtasks in the $\widetilde{K}$ OUEs simultaneously send offloading requests to the MEC servers in ten OECNs. The parameter values of the MSUS2 model are shown in Table 3.

To verify the effectiveness of OANR in the MSUS2 model, we compare its performance with the following proposed algorithms. A limited number of OECNs are allocated to the subtask sets in each OUE based on Formulas (14) and (15). Subsequently, Algorithm 1 is executed in each node allocation iteration of Algorithm 3. Algorithm 4 ignores the priorities between multiple OUEs when executing OANR. Algorithm 3 randomizes the number of OECNs obtained by each OUE and implements OANR on this basis. In Algorithm 4, when all OECNs are assigned, the remaining subtasks are processed locally, that is, the OECNs are not reused.

Table 3. MSUS2 model simulation parameters

Parameter	Value
Average calculated density $\tilde{a}$	[500, 1000] cycles/bit
Mobile nodes computing frequency $\tilde{f}$	[5, 10] GHZ
Maximum energy consumption of subtasks $\tilde{E}_{max}$	4 J
Rated power of subtasks $\tilde{p}_{\max}$	[0.4, 1] W
Number of subtasks $\tilde{S}$	[2, 5]
Local computing frequency $\tilde{fl}$	[0.05, 0.1] GHZ
Channel bandwidth $\tilde{B}$	1 MHZ
Channel gain to noise ratio $\tilde{g}/\tilde{\sigma}^2$	[0.1, 2] W^{-1}
Task data volume $\tilde{D}$	[1, 10] KB

Figure 6 compares the average delay of each OUE for a limited number of OECNs. It shows that the average delay of OUEs increases as the number of OUEs increases. The more the OUEs to be offloaded, the greater the demand for OECNs. Thus, it is easier for subtasks to choose OECNs with relatively poor communication and computing resources for offloading. Due to the limited number of OECNs, subtasks have to compare their local processing delay with the total offloading delay when OECNs are redistributed. Therefore, the average OUE delay is larger than the sufficient-OECN scenario. Figure 6 also compares the average OUE delay for different algorithms. Compared with Algorithm 3, OANR considers the overall OUE delay and introduces strategy replacement. Thus, the effect of saving OUE delay is more obvious. We note that when $\tilde{K}=7$ is due to $\tilde{M}=10$, there are fewer OECNs compared to the number of OUEs to be processed. Each OUE can only be assigned a maximum of two OECNs. The selection of OECNs for OUE neutron tasks are also very limited. Only the best and worst cases are included. Therefore, OANR and Algorithm 3 have approximately the same delay-saving effect owing to insufficient local computing power. In addition, in Algorithm 4, poor OUEs will select OECNs with relatively poor communication and computing resources for offloading. In contrast, a good OUE allocates OECNs with excellent communication and computing resources for offloading, resulting in the wastage of OECNs. Both are not conducive to saving delay. Algorithm 3 randomly allocates OECNs to each OUE, resulting in the assignment of OUEs with a small number of subtasks to considerable OECNs. However, OUEs with a large number of subtasks should use a limited number of OECNs for offloading. It wastes OECN's communication and computing resources. The effect of saving delay is not good.

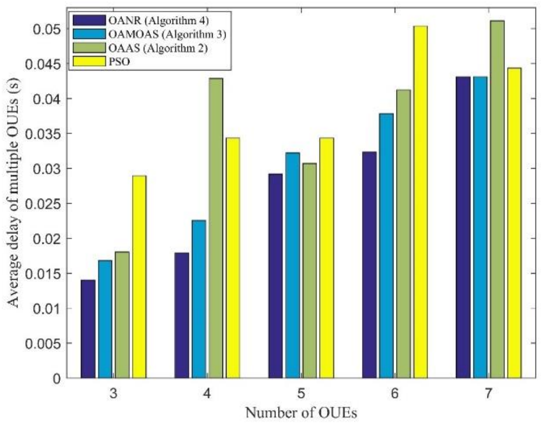


Figure 6. Comparison of OUE delays for different numbers and algorithms

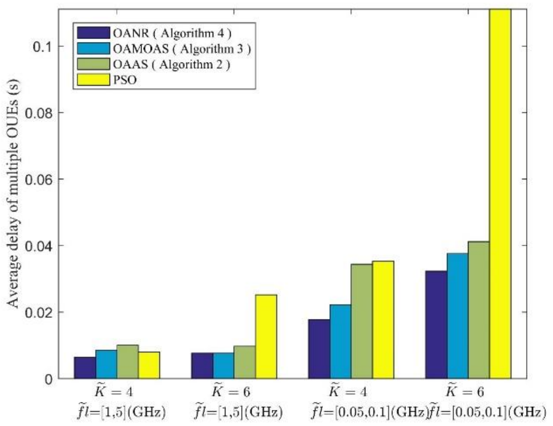


Figure 7. Comparison of OUE delays for different local calculation frequencies and algorithms

Figure 7 compares the average OUE delay for different numbers of OUEs and local computing frequencies. It also shows that the average OUE delay increases as the number of OUEs increases. However, it decreases with the increase in OUE's local calculation frequency. The greater the frequency of OUE's local calculation, the easier it is for subtasks to choose local processing. Thus, the unselected OECNs can process other subtasks, saving considerable delay. In addition, Figure 7 compares various algorithms. Unlike in the OANR, Algorithm 3 does not consider the delay of a single OUE. Algorithm 4 ignores the differences in offloading requirements between multiple OUEs. Algorithm 4 only performs OECN allocation once. Owing to the limited frequency of OUE's local calculation, the effect of saving delay is not ideal. In particular, when $\tilde{K}=6$, $\tilde{fl}=[0.05,0.1](\text{GHZ})$, OUEs are overburdened and the number of OECNs is limited. When each OUE undergoes an OECN allocation, the remaining subtasks are processed locally. In addition, the limited local calculation frequency of OUEs leads to a significant average OUE using Algorithm 3. However, the aim of Algorithm 3 is to save delay when the local calculation frequency of OUEs is sufficient. When $\tilde{K}=6$, $\tilde{fl}=[1,5](\text{GHZ})$, because the OUE's local calculation frequency is sufficient, OANR chooses local processing when comparing offloading strategies. The effect is consistent with Algorithm 4.

7 Conclusions

Faced with massive maritime video surveillance information, we combine MEC with abundant node resources, and the network connectivity delay characteristics. We then establish SSU, MSUS1, and MSUS2 models based on the middle-sea scenario. The MSUS1 model is an extension of the SSU model in the multi-user scenario. Compared with the MSUS1 model, the MSUS2 model considers a limited number of OECNs. We divide the optimization problems into two sub-problems. A binary search method is proposed to optimize the transmission power allocation. We propose OAAS, OAMOAS, and OANR to optimize the offloading decision allocation. We analyze the simulation results from algorithm comparisons and changes in the number of subtasks, data, and OUEs. Therefore, we can verify the effectiveness of the design schemes and algorithms in saving delay. We believe that further research is needed to establish the trust between OECNs in specific maritime video surveillance applications and reduce the dependency between subtasks.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government. (MSIT) (2021R1A2B5B02087169) and the Gachon University research fund of 2021 (GCU-202103440001).

References

[1] H. Gao, J. Xiao, Y. Yin, T. Liu, J. Shi, A Mutually Supervised Graph Attention Network for Few-Shot Segmentation: The Perspective of Fully Utilizing Limited Samples, *IEEE Transactions on Neural Networks and Learning Systems*, March, 2022, doi: 10.1109/TNNLS.2022.3155486.

[2] H. Gao, B. Qiu, R. J. D. Barroso, W. Hussain, Y. Xu, X. Wang, TSMAE: A Novel Anomaly Detection Approach for Internet of Things Time Series Data Using Memory-Augmented Autoencoder, *IEEE Transactions on Network Science and Engineering*, March, 2022, doi:10.1109/TNSE.2022.3163144.

[3] Y. Xu, Y. Wu, H. Gao, S. Song, Y. Yin, X. Xiao, Collaborative APIs recommendation for Artificial Intelligence of Things with information fusion, *Future Generation Computer Systems*, Vol. 125, pp. 471-479, December, 2021.

[4] P. Liang, Z. Yao, J. Li, Marine Meteorological Observation Technology and Application Based On Large Floating Platform, *2019 International Conference on Meteorology Observations (ICMO)*, Chengdu, China, 2019, pp. 1-4.

[5] X. Yang, H. Xing, A data complementary method for thunderstorm point charge localization based on atmospheric electric field apparatus array group, *Digital Communications and Networks*, Vol. 7, No. 2, pp. 170-177, May, 2021.

[6] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, T. Taleb, Survey on multi-access edge computing for Internet of things realization, *IEEE Communications Surveys & Tutorials*, Vol. 20, No. 4, pp. 2961-2991, Fourth Quarter, 2018.

[7] C. Chen, C. Wang, T. Qiu, M. Atiquzzaman, D. Wu, Caching in vehicular named data networking: architecture, schemes and future directions, *IEEE Communications Surveys & Tutorials*, Vol. 22, No. 4, pp. 2378-2407, Fourth Quarter, 2020.

[8] S. Safavat, N. N. Sapavath, D. Rawat, Recent advances in mobile edge computing and content caching, *Digital Communications and Networks*, Vol. 6, No. 2, pp. 189-194, May, 2020.

[9] L. Huang, X. Feng, C. Zhang, L. Qian, Y. Wu, Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing, *Digital Communications and Networks*, Vol. 5, No. 1, pp. 10-17, February, 2019.

[10] J. Zhang, K. Letaief, Mobile edge intelligence and computing for the Internet of vehicles, *Proceedings of the IEEE*, Vol. 108, No. 2, pp. 246- 261, February, 2020.

[11] Y. Wu, L. P. Qian, J. Zheng, H. Zhou, X. S. Shen, Green-Oriented Traffic Offloading through Dual Connectivity in Future Heterogeneous Small Cell Networks, *IEEE Communications Magazine*, Vol. 56, No. 5, pp. 140-147, May, 2018.

[12] Y. Li, H. Ma, L. Wang, S. Mao, G. Wang, Optimized Content Caching and User Association for Edge Computing in Densely Deployed Heterogeneous Networks, *IEEE Transactions on Mobile Computing*, Vol. 21, No. 6, pp. 2130-2142, June, 2022.

[13] S. Xia, Z. Yao, Y. Li, S. Mao, Online Distributed Offloading and Computing Resource Management With Energy Harvesting for Heterogeneous MEC-Enabled IoT, *IEEE Transactions on Wireless Communications*, Vol. 20, No. 10, pp. 6743-6757, October, 2021.

[14] Y. Li, C. Liao, Y. Wang, C. Wang, Energy-Efficient Optimal Relay Selection in Cooperative Cellular Networks Based on Double Auction, *IEEE Transactions on Wireless Communications*, Vol. 14, No. 8, pp. 4093-4104, August, 2015.

[15] N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, Mobile Edge Computing: A Survey, *IEEE Internet of Things Journal*, Vol. 5, No. 1, pp. 450-465, February, 2018.

[16] Y. Zhou, L. Liu, L. Wang, N. Hui, X. Cui, J. Wu, Y. Peng, Y. Qi, C. Xing, Service-aware 6G: an intelligent and open network based on the convergence of communication, computing and caching, *Digital Communications and Networks*, Vol. 6, No. 3, pp. 253-260, August, 2020.

[17] K. Sha, T. Yang, W. Wei, S. Davari, A survey of edge computing-based designs for IoT security, *Digital Communications and Networks*, Vol. 6, No. 2, pp. 195-202, May, 2020.

[18] X. Su, Z. Wang, Y. Wang, S. Zhou, Multi-access edge computing offloading in maritime monitoring sensor networks, *Chinese Journal on Internet of Things*, Vol. 5, No. 1, pp. 36-52, March, 2021.

[19] M. Kamoun, W. Labidi, M. Sarkiss, Joint resource allocation and offloading strategies in cloud enabled cellular networks, *2015 IEEE International Conference on Communications (ICC)*, London, UK, 2015, pp.

5529-5534.

[20] J. Wang, J. Peng, Y. Wei, D. Liu, J. Fu, Adaptive application offloading decision and transmission scheduling for mobile cloud computing, *China Communications*, Vol. 14, No. 3, pp. 169-181, March, 2017.

[21] G. Zhang, F. Shen, N. Chen, P. Zhu, X. Dai, Y. Yang, DOTS: Delay-Optimal Task Scheduling Among Voluntary Nodes in Fog Networks, *IEEE Internet of Things Journal*, Vol. 6, No. 2, pp. 3533-3544, April, 2019.

[22] S. Pan, Z. Zhang, Z. Zhang, D. Zeng, Dependency-Aware Computation Offloading in Mobile Edge Computing: A Reinforcement Learning Approach, *IEEE Access*, Vol. 7, pp. 134742-134753, September, 2019.

[23] Z. Kuang, L. Li, J. Gao, L. Zhao, A. Liu, Partial Offloading Scheduling and Power Allocation for Mobile Edge Computing Systems, *IEEE Internet of Things Journal*, Vol. 6, No. 4, pp. 6774-6785, August, 2019.

[24] J. Zhang, X. Hu, Z. Ning, E. C. H. Ngai, L. Zhou, J. Wei, J. Cheng, B. Hu, V. C. M. Leung, Joint Resource Allocation for Latency-Sensitive Services Over Mobile Edge Computing Networks with Caching, *IEEE Internet of Things Journal*, Vol. 6, No. 3, pp. 4283-4294, June, 2019.

[25] Y. Mao, J. Zhang, K. Letaief, Joint Task Offloading Scheduling and Transmit Power Allocation for Mobile-Edge Computing Systems, *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, USA, 2017, pp. 1-6.

[26] J. Ren, G. Yu, Y. Cai, Y. He, Latency Optimization for Resource Allocation in Mobile-Edge Computation Offloading, *IEEE Transactions on Wireless Communications*, Vol. 17, No. 8, pp. 5506-5519, August, 2018.

[27] Q. Li, J. Zhao, Y. Gong, Q. Zhang, Energy-efficient computation offloading and resource allocation in fog computing for Internet of everything, *China Communications*, Vol. 16, No. 3, pp. 32-41, March, 2019.

[28] J. Feng, F. Yu, Q. Pei, J. Du, L. Zhu, Joint Optimization of Radio and Computational Resources Allocation in Blockchain-Enabled Mobile Edge Computing Systems, *IEEE Transactions on Wireless Communications*, Vol. 19, No. 6, pp. 4321-4334, June, 2020.

[29] Y. D. Lin, Y. Lai, J. X. Huang, H. T. Chien, Three-Tier Capacity and Traffic Allocation for Core, Edges, and Devices for Mobile Edge Computing, *IEEE Transactions on Network and Service Management*, Vol. 15, No. 3, pp. 923-933, September, 2018.

[30] Y. Cui, Y. Liang, R. Wang, Resource Allocation Algorithm with Multi-Platform Intelligent Offloading in D2D-Enabled Vehicular Networks, *IEEE Access*, Vol. 7, pp. 21246-21253, February, 2019.

[31] K. Wang, P. Huang, K. Yang, C. Pan, J. Wang, Unified Offloading Decision Making and Resource Allocation in ME-RAN, *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 8, pp. 8159-8172, August, 2019.

[32] W. Sun, H. Zhang, R. Wang, Y. Zhang, Reducing Offloading Latency for Digital Twin Edge Networks in 6G, *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 10, pp. 12240-12251, October, 2020.

[33] Y. Zhang, P. Du, J. Wang, T. Ba, R. Ding, N. Xin, Resource Scheduling for Delay Minimization in Multi-Server Cellular Edge Computing Systems, *IEEE Access*, Vol. 7, pp. 86265-86273, June, 2019.

[34] G. Wang, F. Xu, C. Zhao, QoS-Enabled Resource Allocation Algorithm in Internet of Vehicles with Mobile Edge Computing, *IET Communications*, Vol. 14, No. 14, pp. 2326-2333, August, 2020.

[35] X. Su, L. Meng, J. Huang, Intelligent maritime networking with edge services and computing capability, *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 11, pp. 13606-13620, November, 2020.

[36] H. Zheng, K. Xiong, P. Fan, Z. Zhong, Z. Ding, K. B. Letaief, Achievable Computation Rate in NOMA-Based Wireless-Powered Networks Assisted by Multiple Fog Servers, *IEEE Internet of Things Journal*, Vol. 8, No. 6, pp. 4802-4815, March, 2021.

[37] Q. Fan, H. Yin, L. Jiao, Y. Lyu, H. Huang, X. Zhang, Towards Optimal Request Mapping and Response Routing for Content Delivery Networks, *IEEE Transactions on Services Computing*, Vol. 14, No. 2, pp. 606-613, March-April, 2021.

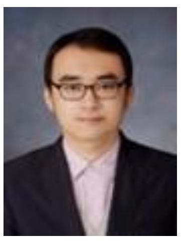
[38] L. Yang, H. Zhang, M. Li, J. Guo, H. Ji, Mobile Edge Computing Empowered Energy Efficient Task Offloading in 5G, *IEEE Transactions on Vehicular Technology*, Vol. 67, No. 7, pp. 6398-6409, July, 2018.

[39] X. Diao, J. Zheng, Y. Wu, Y. Cai, Joint Computing Resource, Power, and Channel Allocations for D2D-Assisted and NOMA-Based Mobile Edge Computing, *IEEE Access*, Vol. 7, pp. 9243-9257, January, 2019.

Biographies

Ziyang Gong is studying for a master’s degree at Gachon University from 2022. Her main research interests include intelligent information processing and intelligent system security.

Ziyi Wang graduated from Hohai University with a master’s degree. Her main research interests include Ocean networks modeling, Edge/Fog Computing, and Computing offloading.



Xin Su received his.D. degree in the Ph Program in IT & Media Convergence Studies, Inha University, in 2015. He is with the College of IoT Engineering, Hohai University. His research interests include 5G systems, Edge/Fog Computing, wireless backhaul solutions, and mobile ad-hoc networks.

Chang Choi received his Ph.D. degree in computer engineering from Chosun University, South Korea, in 2012, and is now working as an assistant professor at Gachon University. His research interests include intelligent information processing, semantic web, smart IoT systems, and intelligent system security.