Two-Stage Offloading Optimization for Energy–Latency Tradeoff With Mobile Edge Computing in Maritime Internet of Things

Tingting Yang, *Member, IEEE*, Hailong Feng, Shan Gao, Zhi Jiang, Meng Qin, Nan Cheng, *Member, IEEE*, and Lin Bai, *Senior Member, IEEE*

***Abstract*—The ever-increasing growth in maritime activities with large amounts of Maritime Internet-of-Things (M-IoT) devices and the exploration of ocean network leads to a great challenge for dealing with a massive amount of maritime data in a cost-effective and energy-efficient way. However, the resources-constrained maritime users cannot meet the high requirements of transmission delay and energy consumption, due to the excessive traffic and limited resources in maritime networks. To solve this problem, mobile edge computing is taken as a promising paradigm to help mobile devices from edge servers via computation offloading considering the different quality of service (QoS) with the complex ocean environments, resulting in energy saving and increased transmission latency. To investigate the tradeoff between latency and energy consumption in low-cost large-scale maritime communication, we formulate the offloading optimization problem and propose a two-stage joint optimal offloading algorithm, optimizing computation and communication resource allocation under limited energy and sensitive latency. At the first stage, the maritime users make the decision on whether to offload a computation considering their demands and environments. Then, the channel allocation and power allocation problems were proposed to optimize the offloading policy which coordinates with the center cloud servers at the second stage, considering the dynamic tradeoff of latency and energy consumption. Finally, numerical simulation results show the effectiveness of the proposed algorithm.**

***Index Terms*—Energy–latency tradeoff, maritime Internet of Things (M-IoT), mobile edge computing (MEC), offloading strategies.**

Manuscript received September 1, 2019; revised October 22, 2019; accepted November 23, 2019. Date of publication December 10, 2019; date of current version July 10, 2020. This work was supported in part by the Natural Science Foundation of China under Grant 61771086, in part by the Dalian Outstanding Young Science and Technology Talents Foundation under Grant 2017RJ06, and in part by the Liaoning Province Xingliao Talents Program under Grant XLYC1807149. *(Corresponding authors: Hailong Feng and Meng Qin.)*

T. Yang is with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan 523000, China, and also with Quan Yu Lab, Shenzhen Pengcheng Laboratory, Shenzhen 518055, China (e-mail: yangtingting820523@163.com).

H. Feng, S. Gao, and Z. Jiang are with the Navigation College, Dalian Maritime University, Dalian 116026, China (e-mail: redshield@163.com; gshan08@163.com; 18900964321@163.com).

M. Qin is with the School of Electronics and Computer Engineering, Peking University, Beijing 518055, China, and also with Quan Yu Lab, Shenzhen Pengcheng Laboratory, Shenzhen 518055, China (e-mail: yaochnqm@gmail.com).

N. Cheng is with the School of Telecommunication, Xidian University, Xi'an 710071, China (e-mail: wmchengnan@gmail.com).

L. Bai is with the School of Cyber Science and Technology, Beihang University, Beijing 100191, China (e-mail: l.bai@buaa.edu.cn).

Digital Object Identifier 10.1109/JIOT.2019.2958662

I. Introduction

RECENTLY, with the advancement of global intelligence, large amounts of smart devices and the increasing smart applications (e.g., virtual reality VR, augmented reality AR, and intelligent transportation) have involved in various fields, which have deeply promoted the development of digital industry, assisting industrial production with information and communication technologies [1]. From steam engines to the Internet, the development of industry is inseparable from the technological revolution. In manufacturing, the Industrial Internet of Things (IIoT) has enormous potential in quality control, sustainability and green development, traceability, and overall efficiency of supply chain [2], [3]. The existing autonomous systems are getting more powerful and take advantage of the capabilities of several types of devices, which helps the smart IIoT to expand from the land to the vast ocean area. The rapid development of intelligent Internet creates new opportunities for maritime activities, in which the ships that sail in the ocean also generate a lot of data that needs to be transmitted and processed. Creating a low-cost and large-scale maritime communication network has recently attracted considerable attention. It has put forward a pivotal challenge for making full use of limited communication and computing resources in an energy-efficient way.

The Maritime Internet of Things (M-IoT) is designed to realize maritime applications, such as ocean monitoring and exploitation in recent decades, many countries have built maritime communication systems, achieving efficient communication for ships and users on the ocean. The construction of an air-space-sea integrated network, as well as the interconnection among satellites, airships, drones, coastal stations, and ships, can escort the safety of ships, which also puts forward high requirements for the ship system. With the increasing applications, a number of maritime mobile devices are largely deployed as well as a growing requirement for low cost and ultrareliable maritime communications. The mass data is provided to users on demand, and end users can also be the manufacturers of massive information.

In particular, data-driven maritime services have been much more popular and important, which has higher requirements for ocean big data. These requirements not only affect the capacity of the air interface but also the reconstruction of the transmission network and the cloud system to form a more decentralized topology, expanding into a converged mobile

core network, with storage and computing capacity dispersed in various forms to the wireless edge [4]. The dramatically increasing devices and data traffic in the M-IoT era are posing significant burdens on the capacity-limited Internet and uncontrollable service delay.

To tackle the challenges above, a number of studies have been done both in academia and industry [5]–[25]. Cloud computing is taken as a dominant paradigm for task computation by sending the data at end users to center cloud with powerful computing capability but at the cost of long latency. For the reason, the emerging of mobile edge computing (MEC), can provide the possibility of network latency, congestion, and capacity in the future, emphasizing closer to mobile users to make a reduction of delay in network operations and service delivery [5]–[18]. Zheng *et al.* [10] studied the dynamic computation offloading based on game theory for MEC-aided networks, and allocated wireless channels to users by achieving Nash equilibrium. Cheng *et al.* [13] studied the enhanced learning method based on the Markov decision process for SAGIN to achieve fast convergence and low energy consumption. He *et al.* [14] designed three transmission strategies in order to achieve low latency. Wei *et al.* [15] described the model of the offloading system and proposed an innovative architecture called MVR, which is helpful for computing offloading in MEC networks. Wang *et al.* [17] took advantage of the latest progress in convex optimization, fully considering the decision making of offloading, resource allocation, and cache-oriented MEC for cellular networks. Kim and Kim [18] developed and implemented a motor state estimation algorithm based on edge updating. Trinh *et al.* studied the potential of MEC in solving energy management related problems of limited power supply devices in the Internet of Things. Trinh *et al.* [19] provided low latency processing of high resolution visual data. A distributed joint computation offloading and resource allocation optimization scheme has been proposed by Zhang *et al.* for heterogeneous networks based on MEC capability. In order to jointly allocate the upstream subchannels, upstream transmission power and computing resources of mobile terminals, a cloud wireless resource allocation algorithm was designed [20]. A joint communication and computing resource allocation problem was proposed by Ren *et al.* to minimize the weighting and latency of all mobile devices [21]. Yang *et al.* [24] proposed a dynamic resource allocation architecture, which included a fast heuristic-based incremental allocation mechanism to dynamically execute resource allocation. Deng *et al.* [25] proposed an architecture based on fog-cloud computing, aiming at the research of delay and power balance, using approximate methods to solve the optimal workload allocation problem. These papers have contributed to the corresponding fields based on MEC, but there are few references for MEC based on M-IoT.

The collaborative and complementary relationship between MEC and cloud computing provides a strong guarantee for maritime wireless communication-oriented ship autopilot, ship distress search and rescue, etc. However, the MEC in the maritime networks brings the computing capability to the edge of the maritime networks, which can contribute to the reducing of energy cost, but resulting in an increased uncontrollable service delay. In addition, the highly dynamic of maritime network environments, such as the mobility of ship users and the reliability of information transmission, is an intractable issue that should be considered to guarantee the different quality of service (QoS) (data-sensitive service or latency-sensitive service) for the low-cost and energy-efficient maritime communications. All of the work brings the computing capability of MEC to the edge users, ignoring the cooperation between the edge computing and cloud servers. Furthermore, the highly dynamic of maritime network environments and the high energy consumption cost are intractable issues that should be considered in the resource-constrained maritime communication networks.

We investigate the dynamic offloading problem with the energy–latency tradeoff in maritime communication networks and propose a two-stage joint optimal offloading algorithm (JOOA) for ship users in MEC, considering the resource limitation and latency requirements in this article. Specifically, we consider that mobile ship users are oriented to the sea and constitute the communications topologies of MEC. To investigate the tradeoff between energy and latency for computation-intensive tasks, we present a two-stage JOOA, jointly optimizing communication and computation resource allocation under the constrained energy and sensitive delay. At the first stage, the maritime users make the decision on whether to offload a computation considering their demands and environments. Then, the channel allocation and power allocation problem is proposed to optimize the offloading policy which coordinates with the center cloud server at the second stage, considering the dynamic tradeoff of latency and energy consumption. In particular, the edge-cloud coordination is taken as a complementary offloading scheme, which can better support the real-time intelligent processing and execution of maritime business. The contributions are summarized as follows.

1) A novel maritime communication network framework is proposed, which combines with MEC to provide efficient computing capability for developing and utilizing the ocean with different QoS requirements.
2) A two-stage JOOA for computation-intensive tasks is proposed, which makes the task more personalized with differentiated constrained resources and limited energy.
3) A fast and efficient channel allocation and power allocation optimal method for ship users'offloading decision is proposed, with full consideration of energy consumption and delay, which can provide a guideline for task offloading in maritime networks.

The remainder of this article is arranged as follows. The system model is described in Section II. We put forward the problem formulation and the proposed two-stage offloading optimization solutions for energy–latency tradeoff in Section III. The simulation results are given in Section IV. In Section V, we summarize this article.

II. System Model

The space–air–sea-integrated communication network is constructed in this section. We mainly focus on the task offloading between the ship users and MEC servers or center

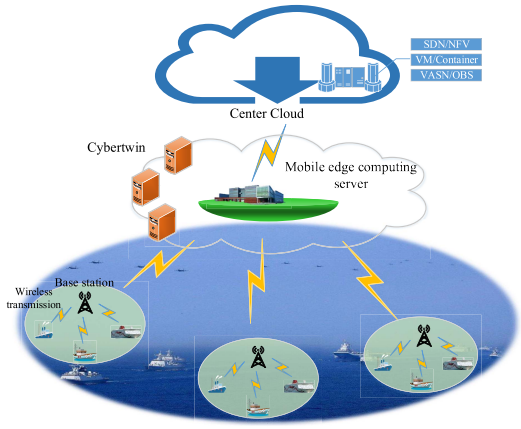


Fig. 1. Maritime network system model.

cloud servers, as shown in Fig. 1. We deploy MEC servers on the edge of the network (e.g., coastal base stations, etc) and the center cloud servers on the remote cloud network. A critical function network named cybertwin is located in the edge cloud which is an intelligent agent of maritime smart devices or ship users, and cybertwin is also deployed on the edge side in virtual cyberspace [26]. There is no need to establish a connection between the end users and servers first, instead, cybertwin establishes a connection with the end users, and according to the needs of the end users, it requests to offload to a better server on behalf of end users, so as to provide efficient and high-quality services. In short, cybertwin is a service agent for end users. In the center cloud server, SND/NFV and other technologies are used for global control, SDN separates the control layer from the data layer, while NFV decouples the functions of network devices from the network hardware, they are supplementary to each other. At the same time, it is easy for the network services to make deployment, and becomes more elastic and powerful. Compared with the hardware-based implementation, it can achieve more complex network topologies and abundant network resident functions [27]. The network consists of several ship communication units. Each ship can be seen as an access user, and a ship can access multiple MEC servers. However, a MEC server can only serve a ship user at the same time. It uses the FIFO service mode to process local task requests [29]. The center cloud server has a powerful processing power, which can handle the arrived tasks immediately. Ships move relatively slow to ensure that each ship is in a communicable range. During the voyage of a ship, a large amount of data (such as navigation monitoring data, etc.) are generated. There are many tasks for transmission calculation and the MEC server can handle more sensitive tasks. Therefore, ship users can break through the limitations of ship users' equipment by offloading these tasks or data onto appropriate MEC servers. Navigating ship users can be represented by set $S = \{s_1, s_2, \ldots, s_m\}$. The access channel of ships can be represented by set $C = \{c_1, c_2, \ldots, c_n\}$.

A. Offloading Strategy

We suppose that each ship user s_m has a computationally intensive task. We define a triple $\xi_m = \{\phi_m, \chi_m, \delta_m\}$. Among them, ϕ_m represents the workload of the ship, χ_m represents the input data, and δ_m represents that how many CPU cycles are needed to compute the intensive task. For each ship user, its compute-intensive tasks can be performed on the local CPU, or offloaded to the MEC server, or the center cloud server farther away. Therefore, the selection of the appropriate task offloading strategy has a vital impact on the function of the system. We define I_{s_m} as an offloading decision. If $I_{s_m} = 1$, the ship user offloads the task to the mobile edge server or the center cloud server for execution. If $I_{s_m} = 0$, the ship user offloads the task to local execution, the ship user's offloading decision can be written as follows:

$$I_{s_m} = \begin{cases} \text{MEC server or Center cloud} \\ 1, \ \text{ship user offloads its task to} \\ 0, \ \text{otherwise.} \end{cases} \tag{1}$$

B. Local Computational Offloading Model

When Ship User s_m decides to offload locally to perform computational task ξ_m, the computational process is only related to the CPU computational capability of the ship user. δ_m represents the number of CPU cycles needed to compute tasks, and $f_{s_m}^{\text{loc}}$ represents the local computing power of the ship user (CPU revolution per second). Each ship's user equipment is different with different computing power. However, the ship user equipment can dynamically adjust the frequency and voltage of the chip according to the different requirements of various application programs running on the chip (with regard to same chip, the lower the frequency, the lower the voltage required). The dynamic voltage frequency scaling (DVFS) technology is taken to achieve the purpose of energy saving [30]. The time spent locally for performing computational tasks can be expressed as

$$T_{s_m}^{\text{loc}} = \delta_m \left(f_{s_m}^{\text{loc}} \right)^{-1}. \tag{2}$$

Then, the energy consumed by performing the computing task locally of the ship user is given as follows:

$$E_{s_m}^{\text{loc}} = \kappa_m \left(f_{s_m}^{\text{loc}} \right)^2 \delta_m. \tag{3}$$

Among them, κ depends on the chip structure of the device. In this article, the value $\kappa = 10^{-11}$ is taken [31]. Based on the above formulas, we can conclude that the total cost of local execution of computational tasks $Q_{s_m}^{\text{loc}}$ is given as follows:

$$Q_{s_m}^{\text{loc}} = \mu_{s_m}^{\text{loc}} T_{s_m}^{\text{loc}} + \nu_{s_m}^{\text{loc}} E_{s_m}^{\text{loc}} \tag{4}$$

where $\mu_{s_m}^{\text{loc}}$ and $\nu_{s_m}^{\text{loc}}$ are the time weight and energy consumption weight coefficients for local execution of computational tasks. Generally speaking, the relationship between the two coefficients is given as follows:

$$\begin{aligned} \mu_{s_m}^{\text{loc}} + \nu_{s_m}^{\text{loc}} &= 1 \\ 0 \le \mu_{s_m}^{\text{loc}} &\le 1 \\ 0 \le \nu_{s_m}^{\text{loc}} &\le 1. \end{aligned} \tag{5}$$

According to the observation, we can clearly see that $\mu_{s_m}^{\text{loc}}$ and $\nu_{s_m}^{\text{loc}}$ can adjust the total consumption of local computing tasks. When $\mu_{s_m}^{\text{loc}}$ is larger, it means that ship user s_m pays

more attention to the delay, while $\nu_{s_m}^{\text{loc}}$ is larger, it pays more attention to energy consumption [32]. Therefore, balancing the two coefficients can reduce total consumption.

C. Edge Computing Offloading Model

When ship user s_m decides to offload computing task ξ_m to the MEC server, there will be interference between ship users. The channel set is represented by C. In order to facilitate calculation, we divide the bandwidth B into c_n parts, and the bandwidth of each subchannel is B/c_n. According to Shannon's theorem, the task upload rate is obtained as follows:

$$r_{s_m}^{ch} = \varphi_{s_m}^{ch} \frac{B}{c_n} \log_2\left(1 + \text{SINR}_{s_m}^{ch}\right). \tag{6}$$

$\text{SINR}_{s_m}^{ch}$ is the signal-to-noise ratio of data transmission

$$\text{SINR}_{s_m}^{ch} = \frac{p_{s_m}^{ch} g_{s_m{}^{ch}}}{N_0 + \sum_{s_m=1, s_m \neq s_n}^{S} p_{s_n}^{ch} g_{s_m, s_n}^{ch}} \tag{7}$$

where N_0 is the background noise, $p_{s_m}^{ch}$ and $p_{s_n}^{ch}$ are the transmission power of ship users s_m and s_n on the channel, respectively. $g_{s_m}^{ch}$ is the channel gain between ship users s_m and MEC server. g_{s_m, s_n}^{ch} is the channel gain of ship user s_m and ship user s_n on the channel [33]. $\varphi_{s_m}^{ch}$ indicates whether channel c_n is allocated to ship user s_m, and if $\varphi_{s_m}^{ch} = 1$, the subchannel is allocated to ship user, *vice versa*, $\varphi_{s_m}^{ch} = 0$. Then when the ship user s_m offloads the task, the total uplink transmission rate is

$$R_{s_m}^{ch} = \sum_{c_n=1}^{C} r_{s_m}^{ch}. \tag{8}$$

According to the above formulas, the sum of the time when a ship user s_m offloads a computing task to an MEC server and executes the task is given as follows:

$$T_{s_m}^{\text{edge}} = \frac{\chi_m}{R_{s_m}^{ch}} + T_{s_m}^{\text{wait}} + \frac{\delta^{\text{edge}}}{f^{\text{edge}}}. \tag{9}$$

The energy consumed by ship users s_m in offloading computational tasks to the MEC servers and performing computations is obtained as follows:

$$E_{s_m}^{\text{edge}} = p_{s_m}^{ch}\left(\frac{\chi_m}{R_{s_m}^{ch}} + T_{s_m}^{\text{wait}}\right) + E_{ex} + \kappa_{\text{edge}}\left(f^{\text{edge}}\right)^2 \delta_{\text{edge}} \tag{10}$$

where $T_{s_m}^{\text{wait}}$ is the waiting time generated by ship user s_m on the channel [34]. We predicts the waiting time according to the queuing theory and multitask computing requirements. E_{ex} indicates tail energy that there will be some additional consumption for ship users s_m after transmitting the tasks, because of the need to maintain channel connectivity to wait for tasks to enter the edge servers or center cloud servers. The additional energy consumption accounts for a small proportion of the whole energy consumption, but it cannot be ignored. The main energy consumption still comes from transmission, edge computing, and cloud computing [28]. According to most literatures, if the data that needs to be returned are relatively small, the energy consumption and latency of the results that needs to be returned are neglected. The total cost of computing tasks in MEC servers is

$$Q_{s_m}^{\text{edge}} = \mu_{s_m}^{\text{edge}} T_{s_m}^{\text{edge}} + \nu_{s_m}^{\text{edge}} E_{s_m}^{\text{edge}}. \tag{11}$$

Here, we assume that the processing capacity of MEC servers is limited, while the processing capacity of center cloud servers is infinite. Only by making rational use of MEC servers and center cloud servers, can the optimal resource allocation be achieved. When the MEC server cannot deal with the intensive tasks of current ship user s_m, it can offload the tasks to the center cloud server through wireless network for computing. At this point, ship user s_m offloads the intensive task to the center cloud server and the total execution time is given as

$$T_{s_m}^{\text{cloud}} = \frac{\chi_m}{R_{s_m}^{ch}} + T_{s_m}^{\text{wait}} + \frac{\delta^{\text{cloud}}}{f^{\text{cloud}}}. \tag{12}$$

The energy consumed by ship user s_m to offload computing tasks to the center cloud server is given as

$$E_{s_m}^{\text{cloud}} = p_{s_m}^{ch}\left(\frac{\chi_m}{R_{s_m}^{ch}} + T_{s_m}^{\text{wait}}\right) + E_{ex}. \tag{13}$$

In the above formulas, there is no energy consumption of the center cloud server because it is assumed that the processing capacity of the center cloud server is infinite. The total cost of computing tasks on center cloud servers is given as

$$Q_{s_m}^{\text{cloud}} = \mu_{s_m}^{\text{cloud}} T_{s_m}^{\text{cloud}} + \nu_{s_m}^{\text{cloud}} E_{s_m}^{\text{cloud}}. \tag{14}$$

D. Problem Formulation

We describe a situation of ship user s_m performing the intensive tasks in the MEC server and in center cloud server, respectively. We can give the total consumption as follows:

$$Q_{ec} = \alpha_{s_m} Q_{s_m}^{\text{edge}} + \left(1 - \alpha_{s_m}\right) Q_{s_m}^{\text{cloud}} \tag{15}$$

where α_{s_m} represents whether it is executed on the MEC server or in the center cloud

$$\alpha_{s_m} = \begin{cases} 1, & \text{MEC server} \\ 0, & \text{cloud center.} \end{cases} \tag{16}$$

Based on the above analysis, it can be concluded that the total consumption of ship user s_m from the start of the task is given as follows:

$$Q_{\text{total}} = I_{s_m} Q_{ec} + \left(1 - I_{s_m}\right) Q_{s_m}^{\text{loc}} \tag{17}$$

where I_{s_m} represents offloading decision making parameters, 1 represents offloading, and 0 represents not offloading. A ship user can only offload a computing task to a destination for execution, so we can get the user's offloading decision matrix

$$H_{\text{off}} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{18}$$

Based on above all, we take the total consumption of ship user s_m in offloading and executing tasks as the optimization objective, with the aim of balancing transmission delay and energy consumption, the offloading optimization problem can be expressed as follows:

$$\min \sum_{s_m=1}^{S} I_{s_m}\left(\alpha_{s_m} Q_{s_m}^{\text{edge}} + \left(1 - \alpha_{s_m}\right) Q_{s_m}^{\text{cloud}}\right) + \min \sum_{s_m=1}^{S} \left(1 - I_{s_m}\right) Q_{s_m}^{\text{loc}} \tag{19}$$

Subject to

$$\begin{aligned}
&C1\text{: } I_{s_m}\left(\alpha_{s_m}T_{s_m}^{\text{edge}} + \left(1-\alpha_{s_m}\right)T_{s_m}^{\text{cloud}}\right) + \left(1-I_{s_m}\right)T_{s_m}^{\text{loc}} \\
&\quad \le T_{\max} \\
&C2\text{: } I_{s_m}\left(\alpha_{s_m}E_{s_m}^{\text{edge}} + \left(1-\alpha_{s_m}\right)E_{s_m}^{\text{cloud}}\right) + \left(1-I_{s_m}\right)E_{s_m}^{\text{loc}} \\
&\quad \le E_{\max} \\
&C3\text{: } 0 \le \sum_{c_m=1}^{C} \varphi_{s_m}^{ch} p_{s_m}^{ch} \le P_{\max} \\
&C4\text{: } I_{s_m} \in \{0,1\}, \alpha_{s_m} \in \{0,1\}, \varphi_{s_m}^{ch} \in \{0,1\} \quad \forall s_m \in S. \quad (20)
\end{aligned}$$

$C1$ represents the maximum delay that the system can tolerate, $C2$ represents that the energy consumption cannot exceed $E_{\max}$, $C3$ represents the power limitations, $C4$ represents the offloading decision of computing tasks that the tasks whether offload by MEC servers or center cloud. According to the above conditions, it can be concluded that the total consumption minimization problem is a nonconvex and mixed integer nonlinear programming problem, which is an NP-hard problem [35].

III. Problem Formulation

Whether the ship user s_m will offload the intensive task to the edge network or the center cloud network is the key to this joint offloading strategy. The purpose of offloading computational tasks is to obtain lower latency and minimum energy consumption. If the total consumption of offloading intensive tasks to the edge network or the center cloud network by ship user s_m is greater than the total consumption of local execution of computational tasks, then the computation offloading will be meaningless. Hence, we begin to study the conditions of local computing strategies, which can be expressed as

$$\begin{cases} Q_{ec} \le Q_{s_m}^{\text{loc}}, & I_{s_m} = 1 \\ Q_{ec} \ge Q_{s_m}^{\text{loc}}, & I_{s_m} = 0. \end{cases} \quad (21)$$

When $I_{s_m} = 1$, the task is offloaded, then $Q_{ec} \le Q_{s_m}^{\text{loc}}$, and *vice versa*, $Q_{ec} \ge Q_{s_m}^{\text{loc}}$.

A. Local Computing Consumption

When ship user s_m chooses to compute intensive tasks locally, the total consumption is given as

$$\min \quad \sum_{s_m=1}^{S} \mu_{s_m}^{\text{loc}}\left(\delta_m {f_{s_m}^{\text{loc}}}^{-1}\right) + \nu_{s_m}^{\text{loc}}\left(\kappa_m {f_{s_m}^{\text{loc}}}^{2}\delta_m\right) \quad (22)$$

$$\begin{aligned}
\text{Subject to: } &C1\text{: } \delta_m\left(f_{s_m}^{\text{loc}}\right)^{-1} \le T_{\max} \\
&C2\text{: } \kappa_m\left(f_{s_m}^{\text{loc}}\right)^{2}\delta_m \le E_{\max} \\
&C3\text{: } f_{s_m}^{\text{loc}} \le f_{\max}^{\text{loc}}. \quad (23)
\end{aligned}$$

From the optimization objective, the total cost of local computing for intensive tasks is only related to the processing capacity of ship user s_m equipment [36]. Therefore, we take $f_{s_m}^{\text{loc}}$ as the first element to calculate the optimal equipment handling capacity of ship user s_m, $f_{s_m}^{\text{loc}}$. We derive the $f_{s_m}^{\text{loc}}$ of (22) and

$$\frac{\partial Q_{s_m}^{\text{loc}}\left(f_{s_m}^{\text{loc}}\right)}{\partial f_{s_m}^{\text{loc}}} = -\mu_{s_m}^{\text{loc}}\frac{\delta_m}{{f_{s_m}^{\text{loc}}}^2} + 2\nu_{s_m}^{\text{loc}}\kappa_m f_{s_m}^{\text{loc}}\delta_m = 0 \quad (24)$$

where $(f_{s_m}^{\text{loc}})^* = \sqrt[3]{[(\mu_{s_m}^{\text{loc}}\delta_m)/(2\nu_{s_m}^{\text{loc}}\kappa_m)]}$ can be obtained to prove the existence of extreme value. According to the constraints of the above conditions, $(\delta_m/T_{\max}) \le f_{s_m}^{\text{loc}} \le \sqrt{[E_{\max}/(\kappa_m\delta_m)]}$ can be obtained. Different computing tasks are sensitive to time delay and energy consumption according to the particularity of tasks. Some tasks require higher time delay and some tasks require higher energy consumption. Therefore, the optimal value $(f_{s_m}^{\text{loc}})^*$ can be obtained by balancing the value of $[(\mu_{s_m}^{\text{loc}})/(\nu_{s_m}^{\text{loc}})]$. Then, we can get the optimal expression of local total consumption is $(Q_{s_m}^{\text{loc}})^*(f_{s_m}^{\text{loc}})$.

B. Offload Model Channel Allocation

In the previous section, the optimal total local execution consumption has been obtained. But when $I_{s_m} = 1$, ship user s_m chooses appropriate channel c_n to offload to servers. At this time, the channel capacity transmitted by the ship user should be less than the total channel capacity, which can be understood as the matching problem between the navigation ship user $S = \{s_1, s_2, \ldots, s_m\}$ and channel $C = \{c_1, c_2, \ldots, c_n\}$. At the same time, it should not exceed the maximum capacity, which can be expressed as the maximum packing problem. We choose the appropriate subchannel c_n for the ship user s_m to offload. The channel state can be shown as

$$\begin{aligned}
&\max \quad \sum_{c_n=1}^{C} \varphi_{s_m}^{ch}\frac{B}{c_n}\log_2\left(1+\text{SINR}_{s_m}^{ch}\right) \le R_{\max} \\
&\text{Subject to: } C1\text{: } 0 < \sum_{c_n=1}^{C} \varphi_{s_m}^{ch} \le c_n \quad \forall C, \varphi_{s_m}^{ch} \in \{0,1\}. \quad (25)
\end{aligned}$$

If too many ship users choose to offload intensive tasks, it will cause channel congestion, and users will enter the waiting state. Moreover, in the current state, the MEC server may not be able to meet the needs of multiple users. So some ship users offload the task to a farther center cloud network. Then, the system tends to be stable [37]. Edge servers or cloud-based processing capabilities are greater than the rate at which the ship users offload. According to the queuing theory of first-in–first-out queuing principle, the time of multitask computing needs is predicted. Using the queuing model of $M/M/1$ with limited capacity, the concept of task average waiting time is introduced. We use $T_{s_m}^{\text{wait}}$ to represent the average waiting time. When the number of intensive tasks exceeds the maximum capacity of the system, the average number of ship users on the mobile edge is S_t, $S_t > R_{\max}$. The entry rate of the system's tasks is λ_{in}. According to little's law, it can be obtained $T_{s_m}^{\text{wait}} = (S_t/\lambda_{in})$. When the system reaches a stable state, the intensive tasks waiting in the MEC server choose whether to continue or offload to the center cloud server according to their own needs. When the task begins to offload, the channel capacity is larger than the number of tasks calculated to offload, i.e., $S_t \le R_{\max}$, and the average waiting time is $T_{s_m}^{\text{wait}} = 0$.

C. Power Allocation for Offloading Tasks

When the ship user s_m chooses to offload, the minimum target consumption is given as

$$\min \quad \sum_{s_m=1}^{S}\left(\alpha_{s_m}Q_{s_m}^{\text{edge}}+\left(1-\alpha_{s_m}\right)Q_{s_m}^{\text{cloud}}\right) \tag{26}$$

$$\begin{aligned}\text{Subject to: } & C1\text{: } \alpha_{s_m}\mu_{s_m}^{en}T^{\text{edge}}+\left(1-\alpha_{s_m}\right)\nu_{s_m}^{en}T^{\text{cloud}}\le T_{\max}\\ & C2\text{: } \alpha_{s_m}\mu_{s_m}^{en}E^{\text{edge}}+\left(1-\alpha_{s_m}\right)\nu_{s_m}^{en}E^{\text{cloud}}\le E_{\max}\\ & C3\text{: } 0<f^{\text{edge}}<f^{\text{cloud}}\\ & C4\text{: } \alpha_{s_m}\in\{0,1\}\end{aligned} \tag{27}$$

where α_{s_m} represents whether the ship user offloads to MEC server or center cloud server when the ship user decides to offload. If $\alpha_{s_m}=1$, the task is offloaded to the MEC server, $\alpha_{s_m}=0$ means to choose the center cloud server. For better understanding, we will discuss the situations separately. When $Q_{s_m}^{\text{edge}}<Q_{s_m}^{\text{cloud}}$, we choose to offload it to the MEC server, and when $Q_{s_m}^{\text{edge}}\ge Q_{s_m}^{\text{cloud}}$, we prefer to offload it to the center cloud server. It can be written as

$$\begin{cases}Q_{s_m}^{\text{edge}}<Q_{s_m}^{\text{cloud}}, & \text{MEC server}\\ Q_{s_m}^{\text{edge}}\ge Q_{s_m}^{\text{cloud}}, & \text{center cloud.}\end{cases} \tag{28}$$

1) *Case 1:* When $Q_{s_m}^{\text{edge}}<Q_{s_m}^{\text{cloud}}$, the MEC service is chosen to offload, and the minimum target consumption is

$$\min \quad \sum_{s_m=1}^{S}\mu_{s_m}^{\text{edge}}T_{s_m}^{\text{edge}}+\nu_{s_m}^{\text{edge}}E_{s_m}^{\text{edge}} \tag{29}$$

$$\begin{aligned}& C1\text{: } \frac{\chi_m}{R_{s_m}^{ch}}+T_{s_m}^{\text{wait}}+\frac{\delta^{\text{edge}}}{f^{\text{edge}}}\le T_{\max}\\ & C2\text{: } p_{s_m}^{ch}\left(\frac{\chi_m}{R_{s_m}^{ch}}+T_{s_m}^{\text{wait}}\right)+E_{ex}+\kappa_{\text{edge}}\left(f^{\text{edge}}\right)^2\delta_{\text{edge}}\\ & \qquad\le E_{\max}\\ & C3\text{: } 0\le p_{s_m}^{ch}\le P_{\max}.\end{aligned} \tag{30}$$

Let $Q_{s_m}^{\text{edge}}(p_{s_m}^{ch})=\mu_{s_m}^{\text{edge}}([\chi_m/R_{s_m}^{ch}]+T_{s_m}^{\text{wait}}+[\delta^{\text{edge}}/f^{\text{edge}}])+\nu_{s_m}^{\text{edge}}(p_{s_m}^{ch}([\chi_m/R_{s_m}^{ch}]+T_{s_m}^{\text{wait}})+E_{ex}+\kappa_{\text{edge}}(f^{\text{edge}})^2\delta_{\text{edge}})$. By sorting out the equation, the following equation can be obtained:

$$\begin{aligned}&Q_{s_m}^{\text{edge}}\left(p_{s_m}^{ch}\right)\\ &=\frac{\left(\mu_{s_m}^{\text{edge}}+\nu_{s_m}^{\text{edge}}p_{s_m}^{ch}\right)\chi_m}{\sum_{c_n=1}^{C}\varphi_{s_m}^{ch}\frac{B}{c_n}\log_2\left(1+\frac{p_{s_m}^{ch}g_{s_m}^{ch}}{N_0+\sum_{s_m=1,s_m\ne s_n}^{S}p_{s_n}^{ch}g_{s_m,s_n}^{ch}}\right)}\\ &\quad+\nu_{s_m}^{\text{edge}}p_{s_m}^{ch}T_{s_m}^{\text{wait}}+Z\end{aligned} \tag{31}$$

where $Z=\mu_{s_m}^{\text{edge}}(T_{s_m}^{\text{wait}}+[\delta^{\text{edge}}/f^{\text{edge}}])+\nu_{s_m}^{\text{edge}}(E_{ex}+\kappa_{\text{edge}}(f^{\text{edge}})^2\delta_{\text{edge}})$. It can be seen that $Q_{s_m}^{\text{edge}}(p_{s_m}^{ch})$ is only related to $p_{s_m}^{ch}$ and is monotonic, which can be obtained by restricting $C1$ according to conditions

$$\begin{aligned}p_{s_m}^{ch}\ge&\left(2^{\frac{\chi_m f^{\text{edge}}}{f^{\text{edge}}\left(T_{\max}-T_{s_m}^{\text{wait}}\right)-\delta^{\text{edge}}}\varphi_{s_m}^{ch}\frac{c_n}{B}}-1\right)\\ &\times\frac{N_0+\sum_{s_m=1,s_m\ne s_n}^{S}p_{s_n}^{ch}g_{s_m,s_n}^{ch}}{g_{s_m}^{ch}}.\end{aligned} \tag{32}$$

According to the condition, $C2$ can be obtained

$$\begin{aligned}&p_{s_m}^{ch}\left(\frac{\chi_m}{\sum_{c_n=1}^{C}\varphi_{s_m}^{ch}\frac{B}{c_n}\log_2\left(1+\frac{p_{s_m}^{ch}g_{s_m}^{ch}}{N_0+\sum_{s_m=1,s_m\ne s_n}^{S}p_{s_n}^{ch}g_{s_m,s_n}^{ch}}\right)}\right)\\ &+p_{s_m}^{ch}T_{s_m}^{\text{wait}}\le E_{\max}-E_{ex}-\kappa_{\text{edge}}\left(f^{\text{edge}}\right)^2\delta_{\text{edge}}.\end{aligned} \tag{33}$$

It is also monotonous [33], [38]. The interval of $p_{s_m}^{ch}$ is $[p_a,p_b]$. Among them, $p_a{}^*=\max\{P_{\min},p_a\}$,$p_b{}^*=\min\{P_{\max},p_b\}$. According to the above two formulas, it can be concluded that the minimum total consumption $(Q_{s_m}^{\text{edge}})^*(p_{s_m}^{ch})$ is

$$\begin{aligned}&\left(Q_{s_m}^{\text{edge}}\right)^*\left(p_{s_m}^{ch}\right)\\ &=\begin{cases}Q_{s_m}^{\text{edge}}(p_a)^*, & \left(p_{s_m}^{ch}\right)^*\le(p_a)^*\\ Q_{s_m}^{\text{edge}}\left(p_{s_m}^{ch}\right)^*, & p_a{}^*<\left(p_{s_m}^{ch}\right)^*<(p_b)^*\\ Q_{s_m}^{\text{edge}}(p_b)^*, & \left(p_{s_m}^{ch}\right)^*\ge(p_b)^*.\end{cases}\end{aligned} \tag{34}$$

Among them, $(p_{s_m}^{ch})^*$ represents the optimal transmission power, making $\nabla Q_{s_m}^{\text{edge}}(p_{s_m}^{ch})|_{p_{s_m}^{ch}=(p_{s_m}^{ch})^*}=0$.

2) *Case 2:* When $Q_{s_m}^{\text{edge}}\ge Q_{s_m}^{\text{cloud}}$, the computing task is chosen to offload to the center cloud server

$$\min \quad \sum_{s_m=1}^{S}\mu_{s_m}^{\text{cloud}}T_{s_m}^{\text{cloud}}+\nu_{s_m}^{\text{cloud}}E_{s_m}^{\text{cloud}} \tag{35}$$

$$\begin{aligned}& C1\text{: } \frac{\chi_m}{R_{s_m}^{ch}}+T_{s_m}^{\text{wait}}+\frac{\delta^{\text{cloud}}}{f^{\text{cloud}}}\le T_{\max}\\ & C2\text{: } p_{s_m}^{ch}\left(\frac{\chi_m}{R_{s_m}^{ch}}+T_{s_m}^{\text{wait}}\right)+E_{ex}\le E_{\max}\\ & C3\text{: } 0\le p_{s_m}^{ch}\le P_{\max}.\end{aligned} \tag{36}$$

Let $Q_{s_m}^{\text{cloud}}(p_{s_m}^{ch})=\mu_{s_m}^{\text{cloud}}([\chi_m/R_{s_m}^{ch}]+T_{s_m}^{\text{wait}}+[\delta^{\text{cloud}}/f^{\text{cloud}}])+\nu_{s_m}^{\text{cloud}}(p_{s_m}^{ch}([\chi_m/R_{s_m}^{ch}]+T_{s_m}^{\text{wait}})+E_{ex})$. By sorting it out

$$\begin{aligned}&Q_{s_m}^{\text{edge}}\left(p_{s_m}^{ch}\right)\\ &=\frac{\left(\mu_{s_m}^{\text{cloud}}+\nu_{s_m}^{\text{cloud}}p_{s_m}^{ch}\right)\chi_m}{\sum_{c_n=1}^{C}\varphi_{s_m}^{ch}\frac{B}{c_n}\log_2\left(1+\frac{p_{s_m}^{ch}g_{s_m}^{ch}}{N_0+\sum_{s_m=1,s_m\ne s_n}^{S}p_{s_n}^{ch}g_{s_m,s_n}^{ch}}\right)}\\ &\quad+\nu_{s_m}^{\text{cloud}}(p_{s_m}^{ch}T_{s_m}^{\text{wait}}+E_{ex})\\ &\quad+\mu_{s_m}^{\text{cloud}}\left(T_{s_m}^{\text{wait}}+\frac{\delta^{\text{cloud}}}{f^{\text{cloud}}}\right)\end{aligned} \tag{37}$$

$$p_{s_m}^{ch}\ge\left(2^{\frac{\chi_m f^{\text{cloud}}}{f^{\text{cloud}}\left(T_{\max}-T_{s_m}^{\text{wait}}\right)-\delta^{\text{edge}}}\varphi_{s_m}^{ch}\frac{c_n}{B}}-1\right)$$

$$\times \frac{N_0 + \sum_{s_m=1, s_m \neq s_n}^{S} p_{s_n}^{ch} g_{s_m, s_n}^{ch}}{g_{s_m}^{ch}}. \quad (38)$$

As in case 1, it can be concluded that the range of $p_{s_m}^{ch}$ is $[p_c, p_d]$. Among them, $p_c{}^* = \max\{P_{\min}, p_c\}$, $p_d{}^* = \min\{P_{\max}, p_d\}$. According to the above two formulas, it can be concluded that the minimum total consumption of $(Q_{s_m}^{\text{cloud}})^*(p_{s_m}^{ch})$ is

$$\left(Q_{s_m}^{\text{cloud}}\right)^*\left(p_{s_m}^{ch}\right) = \begin{cases} Q_{s_m}^{\text{cloud}}(p_c)^*, & \left(p_{s_m}^{ch}\right)^* \leq (p_c)^* \\ Q_{s_m}^{\text{cloud}}\left(p_{s_m}^{ch}\right)^*, & p_c{}^* < \left(p_{s_m}^{ch}\right)^* < (p_d)^* \\ Q_{s_m}^{\text{cloud}}(p_d)^*, & \left(p_{s_m}^{ch}\right)^* \geq (p_d)^*. \end{cases} \quad (39)$$

Among them, $(p_{s_m}^{ch})^*$ represents the optimal transmission power, making $\nabla Q_{s_m}^{\text{cloud}}(p_{s_m}^{ch})|_{p_{s_m}^{ch}=(p_{s_m}^{ch})^*} = 0$.

We can see that only $p_{s_m}^{ch}$ has an impact on $Q_{ec}(p_{s_m}^{ch})$. Therefore, it can be concluded that $(p_{s_m}^{ch})^*$ represents the optimal transmission power so that $\nabla Q_{ec}(p_{s_m}^{ch})|_{p_{s_m}^{ch}=(p_{s_m}^{ch})^*} = 0$.

The detailed optimal offloading algorithm is presented in pseudocode form, as shown in Algorithm 1.

Algorithm 1 JOOA for Ship Users in MEC

Input: the intensive-task $\xi_m = \{\phi_m, \chi_m, \delta_m\}$ of ship user s_m and related parameters.

Output: offloading strategy I_{s_m}, decision factor α_{s_m} and total consumption of ship user s_m.

Initialize channel state, system state, input parameters.

for $S = 1, \cdots, s_m$ **do**

Each task has its own characteristics, time sensitive or energy sensitive. According to the feedback information of cyberwin and the unloading matrix, we can decide whether to unload the related tasks or not.

if $I_{s_m} = 0$ **then**

The intensive-task ξ_m is executed locally.

Calculate $T_{s_m}^{loc}$, $E_{s_m}^{loc}$.

Calculate the optimal CPU-cycle frequency $\left(f_{s_m}^{loc}\right)^*$.

Calculate the optimal $Q_{s_m}^{loc}$.

else

$I_{s_m} = 1$, Ship user s_m chooses to offload intensive-tasks to MEC server or center cloud.

end if

while $I_{s_m} = 1$ **do**

if $Q_{s_m}^{edge} < Q_{s_m}^{cloud}$ **then**

Ship user s_m chooses to offload tasks to MEC server $\alpha_{s_m} = 1$.

Detecting CSI, the system reaches a stable state, and the ship user enters the waiting sequence.

Allocation of appropriate power, calculate the optimal $Q_{s_m}^{edge}$.

else

$Q_{s_m}^{edge} \geq Q_{s_m}^{cloud}$, ship user s_m chooses to offload tasks to center cloud $\alpha_{s_m} = 0$.

Ship users choose center cloud according to their own tasks, calculate the optimal $Q_{s_m}^{cloud}$

end if

Calculate the optimal power $\left(p_{s_m}^{ch}\right)^*$ to make $\nabla Q_{ec}\left(p_{s_m}^{ch}\right)\Big|_{p_{s_m}^{ch}=(p_{s_m}^{ch})^*} = 0$.

end while

end for

Calculate minimum consumption Q_{total} of ship users.

IV. Simulation and Analysis

In this part, we will simulate and analyze the proposed algorithm, compare the impact of multiple ship users on the offloading strategy of computing tasks, and analyze the total consumption and workload of various situations. Ship users are all within the coverage of the base station, and their slow movement has no effect on the offloading of tasks. We set the experimental conditions as follows: the channel bandwidth is 4 MHz, there are ten channels, the bandwidth of each channel is -60 dBm, and each ship user has a computationally intensive task to deal with. The size of the computational task is 1–4 MB. The range of operands required by the task is 500–2000 megacycle, the CPU cycle is 0.1–1 GHz, and the clock frequency of the MEC server is 10 GHz. The clock frequency of the center cloud is 100 GHz. The values of parameters μ_{s_m} and ν_{s_m} are adjusted according to the different types of tasks. Due to the characteristics of different intensive tasks, in the simulation, weight coefficients and between 0-1 are randomly generated to represent the corresponding time sensitive and energy sensitive of different tasks ($\mu_{s_m} + \nu_{s_m} = 1$). In this way, it can show the randomness of the type of tasks that the ship users want to offload.

We compare three strategies of "execute locally," "MEC offloading," and "JOOA," which represent ship users to perform computing tasks locally, offload computing tasks to MEC server, and the proposed JOOA algorithm, respectively.

Fig. 2 shows the change of time delay under the different number of ship users. We can clearly see that the time delay becomes larger and larger as the number of tasks accessing the network increases. By comparison, when the number of computing tasks is small, the local execution latency is less than the other two strategies. It shows that the execute locally strategy has its advantages when the number of tasks of the ship user is within the acceptable computational range. However, with the increase of the number of tasks accessed, the other two strategies are more and more sensitive to low latency. When the maximum number of tasks can be accommodated is reached, the MEC offloading strategy will appear the phenomenon of task queuing, which greatly increases the waiting delay of the characters. The JOOA strategy mentioned in this article is in the process of waiting, some users choose to offload to the center cloud according to their own needs, which greatly reduces the delay, and proves the effectiveness of the proposed strategy.

Fig. 3 shows the change of energy consumption under the different number of ship users. We can clearly see that as the number of tasks in the access network increases, the energy consumption also increases. Through the analysis, it can be concluded that when the ship user carries out computational tasks locally, the energy consumption is much higher than the other two strategies when the number of tasks is small due

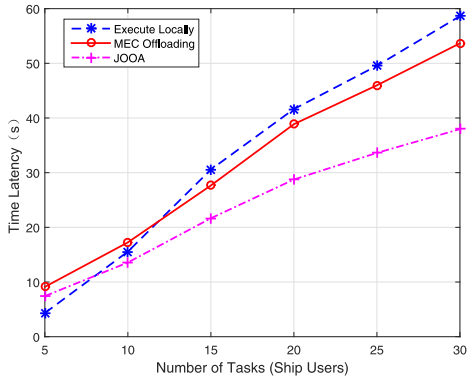


Fig. 2. Time latency versus different tasks.

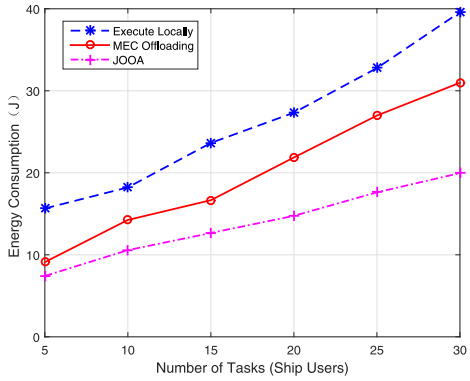


Fig. 3. Energy consumption under different tasks (ship users).

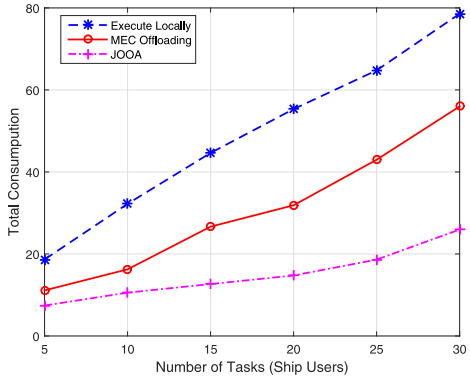


Fig. 4. Total consumption versus different tasks.

to the limited computational capacity. While offloading computing tasks to MEC servers breaks through the limitation of computing power of their own devices, greatly reducing energy consumption, but MEC servers handle too many tasks, and their computing power is limited. Therefore, the proposed JOOA strategy has greatly widened the gap of energy consumption when the number of tasks reaches a certain level.

Fig. 4 shows the change of total consumption under the different number of ship users. We can see that with the increase of the number of tasks accessing the network, the total consumption is also increasing. If the ship user chooses the execute locally strategy, its total consumption is much higher than the other two strategies, because of its relatively poor computing ability. Compared with JOOA strategy, the

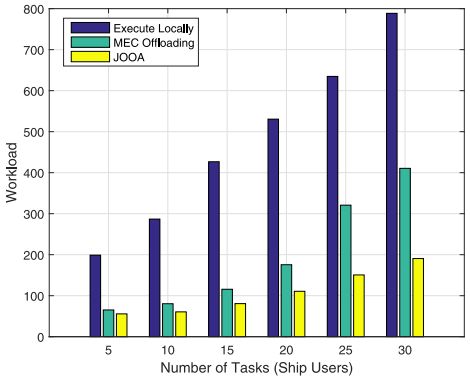


Fig. 5. Workload versus different tasks.

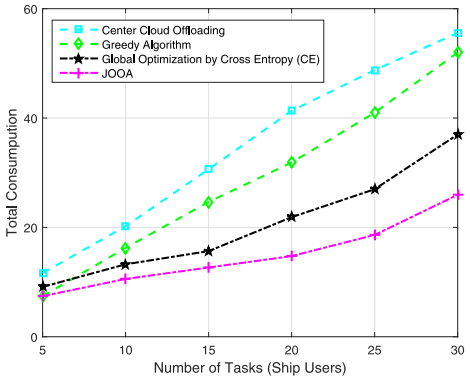


Fig. 6. Total consumption versus different tasks.

overall consumption of the MEC server is close when the number of tasks is small. With the increase of the number of tasks, the MEC server reaches its limits and runs overloaded. Therefore, the proposed JOOA strategy, which coordinates the MEC server with the center cloud server, can greatly reduce the total consumption.

Fig. 5 shows how the workload varies with the number of ship users. We can see that the workload increases with the number of tasks accessing the network. Because of the limited computing power of the ship itself, if the execute locally strategy is chosen, the load of the ship user's own equipment will be greatly increased, even exceeding its load capacity. For the other two strategies, when the number of tasks is small, the workload of the MEC server is almost the same as that of JOOA strategy. It shows that the ship users can offload MEC according to their own needs and judging the status of the MEC servers. However, with the increase of computing tasks, the proposed JOOA strategy obviously has higher performance.

Fig. 6 presents the comparisons with algorithms in [39] and [40]. First, we can see that the ship users offload all tasks to the center cloud server, resulting in a large total consumption. Compared with the other two algorithms, the greedy algorithm has almost the same consumption as JOOA in the case of fewer ship users, and with the increase of the number of ship users accessing, the disadvantages of the greedy algorithm are obvious. Another algorithm is a global optimization by the cross entropy (CE) method. All the ship users need

to report information to the center cloud, which may cause information blocking. At the same time, facing the personalized needs of ship users, this method may appear inadequate. From the data point of view, when the number of ship users reach to 30, the total consumption is reduced by about 24.5%. The validity of the proposed algorithm is proved.

V. Conclusion

In this article, we investigated the dynamic offloading problem considering the energy–latency tradeoff in maritime communication networks. A novel two-stage JOOA for the ship user in MEC is proposed, considering the resource limitation and latency requirements under the constrained energy and sensitive delay. It can provide an efficient guideline for optimizing the maritime communication networks. In the future work, we will consider introducing artificial intelligence (AI) [41] into the ocean network to better explore the ocean and serve the ocean with smart management.

References

[1] J. Wu, S. Guo, H. Huang, W. Liu, and Y. Xiang, "Information and communications technologies for sustainable development goals: State-of-the-art, needs and perspectives," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2389–2406, 3rd Quart., 2018.

[2] S. Vitturi, C. Zunino, and T. Sauter, "Industrial communication systems and their future challenges: Next-generation Ethernet, IIoT, and 5G," *Proc. IEEE*, vol. 107, no. 6, pp. 944–961, Jun. 2019.

[3] L. D. Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[4] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu, "Real-time tasks oriented energy-aware scheduling in virtualized clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 168–180, Apr.–Jun. 2014.

[5] H. Li, G. Shou, Y. Hu, and Z. Guo, "Mobile edge computing: Progress and challenges," in *Proc. 4th IEEE Int. Conf. Mobile Cloud Comput. Service Eng. (MobileCloud)*, Oxford, U.K., 2016, pp. 83–84.

[6] Y. Jin, I. Jung, K. Kim, and S. Choi, "ETSI reconfigurable radio system ął standard architecture and radio application," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, 2016, pp. 1094–1097.

[7] N. Cheng *et al.*, "Big data driven vehicular networks," *IEEE Netw.*, vol. 32, no. 6, pp. 160–167, Nov./Dec. 2018.

[8] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[9] R. Deng, S. He, P. Cheng, and Y. Sun, "Towards balanced energy charging and transmission collision in wireless rechargeable sensor networks," *J. Commun. Netw.*, vol. 19, no. 4, pp. 341–350, Aug. 2017.

[10] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 18, no. 4, pp. 771–786, Apr. 2019.

[11] J. Xu, K. Ota, and M. Dong, "Saving energy on the edge: In-memory caching for multi-tier heterogeneous networks," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 102–107, May 2018.

[12] C. Lai, M. Zhang, J. Cao, and D. Zheng, "SPIR: A secure and privacy-preserving incentive scheme for reliable real-time map updates," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2019.2953188.

[13] N. Cheng *et al.*, "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, May 2019.

[14] S. He *et al.*, "Cloud-edge coordinated processing: Low-latency multicasting transmission," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1144–1158, Nov. 2019.

[15] X. Wei *et al.*, "MVR: An architecture for computation offloading in mobile edge computing," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, Honolulu, HI, USA, 2017, pp. 232–235.

[16] H. Li, K. Ota, and M. Dong, "ECCN: Orchestration of edge-centric computing and content-centric networking in the 5G radio access network," *IEEE Wireless Commun.*, vol. 25, no. 3, pp. 88–93, Jun. 2018.

[17] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 4924–4938, Aug. 2017

[18] J. Kim and B. K. Kim, "Development of precise encoder edge-based state estimation for motors," *IEEE Trans. Ind. Electron.*, vol. 63, no. 6, pp. 3648–3655, Jun. 2016.

[19] H. Trinh *et al.*, "Energy-aware mobile edge computing and routing for low-latency visual data processing," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2562–2577, Oct. 2018.

[20] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing," *IEEE Access*, vol. 6, pp. 19324–19337, 2018.

[21] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.

[22] X. Tao, K. Ota, M. Dong, H. Qi, and K. Li, "Performance guaranteed computation offloading for mobile-edge cloud computing," *IEEE Wireless Commun. Lett.*, vol. 6, no. 6, pp. 774–777, Dec. 2017.

[23] F. Lyu *et al.*, "Characterizing urban vehicle-to-vehicle communications for reliable safety applications," *IEEE Trans. Intell. Transp. Syst.*, early access, 2019, doi: 10.1109/TITS.2019.2920813.

[24] B. Yang, W. K. Chai, Z. Xu, K. V. Katsaros, and G. Pavlou, "Cost-efficient NFV-enabled mobile edge-cloud for low latency mobile applications," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 475–488, Mar. 2018.

[25] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

[26] Q. Yu, J. Ren, Y. Fu, Y. Li, and W. Zhang, "CybertWin: An origin of next generation network architecture," *IEEE Wireless Commun. Mag.*, to be published. [Online]. Available: https://arxiv.org/pdf/1904.11313.pdf

[27] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1994–2008, Dec. 2017.

[28] W. Hu and G. Cao, "Quality-aware traffic offloading in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3182–3195, Nov. 2017.

[29] A. Muhammed, K. M. Saleh, and A. Abdullah, "Optimum packet size of voice packet in the FIFO adversarial queuing model," in *Proc. Asia–Pac. Conf. Appl. Electromagn.*, Melaka, Malaysia, 2007, pp. 1–6.

[30] X. Lin, Y. Wang, Q. Xie, and M. Pedram, "Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment," *IEEE Trans. Service Comput.*, vol. 8, no. 2, pp. 175–186, Mar./Apr. 2015.

[31] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. Usenix Conf. Hot Topics Cloud Comput.*, Boston, MA, USA, 2010, p. 4.

[32] L. Liu, R. Zhang, and K. Chua, "Wireless information transfer with opportunistic energy harvesting," *IEEE Trans. Wireless Commun.*, vol. 12, no. 1, pp. 288–300, Jan. 2013.

[33] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[34] Q. Wang, S. Guo, J. Liu, and Y. Yang, "Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing," *Sustain. Comput. Inf. Syst.*, vol. 21, pp. 154–164, Mar. 2019,

[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[36] J. Zhang *et al.*, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

[37] M. Masoudi, B. Khamidehi, and C. Cavdar, "Green cloud computing for multi cell networks," in *Proc. Wireless Commun. Netw. Conf.*, San Francisco, CA, USA, 2017, pp. 1–6.

[38] 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, Dec. 2016.

[39] M. Ra *et al.*, "Odessa: Enabling interactive perception applications on mobile devices categories and subject descriptors," in *Proc. 9th Int. Conf. Mobile Syst. Appl. Security*, New York, NY, USA, 2011, pp. 43–56.

[40] R. Y. Rubinstein and D. P. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. New York, NY, USA: Springer, 2004.

[41] X. Wang, X. Li, and V. C. M. Leung, "Artificial intelligence-based techniques for emerging heterogeneous network: State of the arts, opportunities, and challenges," *IEEE Access*, vol. 3, pp. 1379–1391, 2015.

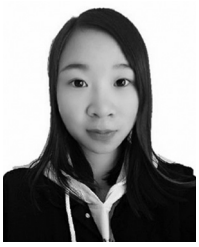
Tingting Yang (M'13) received the B.Sc. and Ph.D. degrees from Dalian Maritime University, Dalian, China, in 2004 and 2010, respectively.

She is currently a Professor with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan, China. From September 2012 to August 2013, she was a Visiting Scholar with the Broadband Communications Research Lab, Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her research interests are maritime wideband communication networks, DTN networks, and green wireless communications.

Prof. Yang serves as an Associate Editor-in-Chief for *IET Communications*, as well as an Advisory Editor for *SpringerPlus*. She also served as a TPC Member for IEEE ICC'14 and ICC'15.

Hailong Feng received the B.S. and M.S. degrees from Dalian Maritime University, Dalian, China, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree with Navigation College.

His research interests include maritime wideband communication networks, DTN networks, wireless sensor network, and mobile edge computing.



Shan Gao received the B.E. degree in electronic and information engineering from Dalian Minzu University, Dalian, China, in 2018. She is currently pursuing the M.S. degree with Navigation College, Dalian Maritime University, Dalian.

Her research interests include artificial intelligence, deep reinforcement learning, and maritime wideband communication networks.

Zhi Jiang received the B.S. degree from Dalian Maritime University, Dalian, China, in 2016, where he is currently pursuing the M.S. degree with Navigation College.

His research interests include maritime broadband communication networks, DTN networks, wireless sensor networks, and intelligent scheduling algorithms.

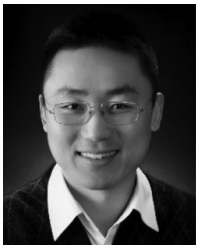
Mr. Jiang served as a TPC Member for the 2019 IEEE/CIC International Conference on Communications in China.

Meng Qin received the B.S. degree in communication engineering from the Taiyuan University of Technology, Taiyuan, China, in 2012, and the M.S. and Ph.D. degrees in information and communication systems from Xidian University, Xi'an, China, in 2015 and 2018, respectively.

He is currently working as a Joint Postdoctoral Fellow with Peng Cheng Laboratory, Shenzhen, China, and Peking University Shenzhen Graduate School, Pengcheng Laboratory, Beijing, China. His research interests include AI-aided wireless network operation and management, machine learning, self-organized network, statistical quality-of-service provisioning and applications of stochastic optimization in intelligent wireless networks, and green cloud storage.

Nan Cheng (S'12–M'16) received the B.E. and M.S. degrees from the Department of Electronics and Information Engineering, Tongji University, Shanghai, China, in 2009 and 2012, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2016.

He is currently a Professor with the School of Telecommunication Engineering, State Key Laboratory of ISN, Xidian University, Xi'an, China. He worked as a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada, from 2017 to 2018. His current research focuses on space-air-ground integrated system, big data in vehicular networks, self-driving system, performance analysis, MAC, opportunistic communication, application of AI for vehicular networks, and space-air-ground integrated networks.



Lin Bai (M'13–SM'17) received the B.Sc. degree in electronic and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2004, the M.Sc. degree (Hons.) in communication systems from the University of Wales, Swansea, U.K., in 2007, and the Ph.D. degree in advanced telecommunications from the School of Engineering, Swansea University, Swansea, in 2010.

Since 2011, he has been with Beihang University (Beijing University of Aeronautics and Astronautics), Beijing, China, as an Associate Professor, where he is currently with the School of Cyber Science and Technology. His research interests include multiple-input–multiple-output, Internet of Things, and unmanned aerial vehicle communications.

Dr. Bai received the IEEE Communications Letters Exemplary Reviewers Certificate for 2012 and the IEEE Wireless Communications Letters Top Editor Award for 2018. He is currently an Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and IEEE WIRELESS COMMUNICATIONS LETTERS, and a Managing Editor of the *Journal of Communications and Information Networks*. He also has served as a Lead Guest Editor for IEEE WIRELESS COMMUNICATIONS and a Guest Editor for the IEEE INTERNET OF THINGS JOURNAL.