Chinese Journal of Electronics
vol. 33, no. 4, pp. 875–885, July 2024
https://doi.org/10.23919/cje.2022.00.412

Special Focus on Multi-Dimensional QoS Provision of Intelligent Edge Computing for IoT

RESEARCH ARTICLE

QoS-Aware Computation Offloading in LEO Satellite Edge Computing for IoT: A Game-Theoretical Approach

Ying CHEN[1], Jintao HU[1], Jie ZHAO[1], and Geyong MIN[2]

1. *School of Computer Science, Beijing Information Science and Technology University, Beijing 100101, China*
2. *University of Exeter, Exeter EX4 4QF, UK*

Corresponding author: Ying CHEN, Email: chenying@bistu.edu.cn

Manuscript Received December 5, 2022; Accepted March 14, 2023

Abstract — Low earth orbit (LEO) satellite edge computing can overcome communication difficulties in harsh environments, which lack the support of terrestrial communication infrastructure. It is an indispensable option for achieving worldwide wireless communication coverage in the future. To improve the quality-of-service (QoS) for Internet-of-things (IoT) devices, we combine LEO satellite edge computing and ground communication systems to provide network services for IoT devices in harsh environments. We study the QoS-aware computation offloading (QCO) problem for IoT devices in LEO satellite edge computing. Then we investigate the computation offloading strategy for IoT devices that can minimize the total QoS cost of all devices while satisfying multiple constraints, such as the computing resource constraint, delay constraint, and energy consumption constraint. We formulate the QoS-aware computation offloading problem as a game model named QCO game based on the non-cooperative competition game among IoT devices. We analyze the finite improvement property of the QCO game and prove that there is a Nash equilibrium for the QCO game. We propose a distributed QoS-aware computation offloading (DQCO) algorithm for the QCO game. Experimental results show that the DQCO algorithm can effectively reduce the total QoS cost of IoT devices.

Keywords — Quality-of-service, Computation offloading, Low earth orbit satellite edge computing, Internet-of-things, Game theory.

Citation — Ying CHEN, Jintao HU, Jie ZHAO, *et al.*, “QoS-Aware Computation Offloading in LEO Satellite Edge Computing for IoT: A Game-Theoretical Approach,” *Chinese Journal of Electronics*, vol. 33, no. 4, pp. 875–885, 2024. doi: 10.23919/cje.2022.00.412.

I. Introduction

The worldwide mobile subscriptions are increasing rapidly with the unprecedented growth of the Internet-of-things (IoT) and emerging network services [1]. Common terrestrial networks have been difficult to meet the various complex needs in global communications. Terrestrial infrastructure in harsh environments such as oceans, deserts, and remote areas is vulnerable to damage from natural disasters such as earthquakes and flooding, which can interrupt the communication between devices and servers [2]. It is tricky to provide network services for IoT devices in harsh environments.

Recently, low earth orbit (LEO) satellite technology has made great strides in civil, commercial, and military services, enabling the economic miniaturization of LEO satellites. LEO satellites play an important role in the supply chain of communication technology-related equipment with the rapid development of LEO satellites [3]. Several global application providers have noted the rapid growth of the international satellite supply chain market, hoping to increase the market share of their LEO satellite networks in recent years. Many influential satellite projects, such as SpaceX StarLink and Amazon Kuiper, provide high-quality communication services for IoT devices around the world by launching and deploying hun-

Guest Editor: Prof. Feng LYU, Central South University.

dreds of LEO satellites [4]. It can be concluded that compared to terrestrial mobile communication systems, LEO satellite communication networks enable seamless global coverage and can better guarantee the quality-of-service (QoS) for IoT devices [5], [6].

The growing popularity of mobile devices for IoT has given rise to several emerging computationally intensive applications such as TikTok, 4K video, and unmanned vehicles. In many studies for terrestrial networks, computation tasks can be offloaded to nearby base stations or cloud servers for processing due to the limited computing and storage resources of IoT devices [7], [8]. However, computation tasks generated by devices in harsh environments not supported by terrestrial communication infrastructure can only be handled by servers deployed on LEO satellites. It is not negligible that the transmission delay of devices in LEO satellite networks increases accordingly due to the orbital altitude of LEO satellites, creating an obstacle to real-time QoS from IoT devices [9], [10]. In this paper, we combine LEO satellite networks and ground communication systems and then introduce mobile edge computing to provide computing resources to the edge of the LEO satellite network. We study the QoS-aware computation offloading (QCO) problem for IoT devices to efficiently reduce the total QoS cost of devices under LEO satellite coverage time and computing capability constraints.

Our main contributions are as follows:

- We study the QCO problem for IoT devices in LEO satellite edge computing, where the devices can offload their computation tasks to the LEO satellites via the channels. The computation offloading decision for each IoT device is local computing or LEO satellite edge computing. In the QoS-aware computation offloading problem, we aim to minimize the total QoS cost of IoT devices while satisfying the decision constraint, computing capability constraint, delay constraint, and energy consumption constraint.
- Based on the non-cooperative competition between IoT devices, we formulate this QCO problem as a game model named QCO game. We define a potential function for the QCO game and prove that the QCO game is an ordinal potential game. We then propose a distributed QoS-aware computation offloading (DQCO) algorithm for the QCO game, which can obtain at least one Nash equilibrium strategy for IoT devices after a finite number of iterations.
- To evaluate the performance of the DQCO algorithm, we conduct a series of parameter analysis experiments for the DQCO algorithm. We also perform extensive comparison experiments between DQCO and other benchmark algorithms such as the Random algorithm and the computing locally (CL) algorithm.

The rest of this paper is organized below. The system model and problem formulation are introduced in Section II. In Section III, we formulate the game model for game-theoretical computation offloading in LEO satellite edge computing and propose a DQCO algorithm. Then extensive parameter analysis and comparison experiments are conducted in Section IV to evaluate the performance. Section V discusses the related work. Finally, we conclude this study in Section VI.

II. System Model and Problem Formulation

In this section, we describe the system model in LEO satellite edge computing, communication time for LEO satellite, QoS model for computation offloading, and problem formulation in detail.

1. System model in LEO satellite edge computing

As shown in Figure 1, we study the scenario for an LEO satellite edge computing system, where the satellites operating in LEO are deployed with edge servers. IoT devices are in harsh environments, such as oceans, deserts, and remote areas, without the support of terrestrial communication infrastructure. The set of all n devices in the system is denoted as $\mathcal{D}=\{d_1,d_2,\dots,d_n\}$. We consider m LEO satellites with edge servers when the devices request IoT services, represented by the set $\mathcal{S}=\{s_1,s_2,\dots,s_m\}$. There are q channels for each LEO satellite $s_j\in\mathcal{S}$, the set of all the channels for satellite s_j is indicated by $\mathcal{H}_j=\{h_j^1,h_j^2,\dots,h_j^q\}$. The total bandwidth B_j for satellite s_j is divided evenly to each channel h_j^k, i.e., the channel bandwidth for h_j^k is $B_j^k=B_j/q$ [11]. In addition, the computing capability for LEO satellite s_j is F_j.

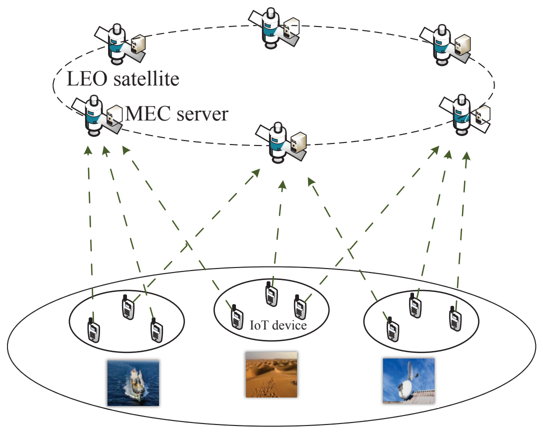


Figure 1 The system scenario in LEO satellite edge computing for IoT.

The computation task for each IoT device d_i when computation offloading is expressed as $K_i=\{Z_i,X_i\}$, Z_i is the data size of task K_i during task transmission, and X_i is the number of CPU cycles required for computing the task. To request computation offloading services, the devices can either compute their tasks locally or offload their computation tasks to the edge server of the LEO satellites for computation. Then the definition of computation offloading decision and strategy regarding the device is denoted below.

Definition 1 (Computation offloading decision) Let a_i denote the binary variable for computation offloading decision of each IoT device d_i, where $a_i = (j, k)$ indicates that device d_i offloads its computation task K_i to LEO satellite s_j through the channel h_j^k, $a_i = (0, 0)$ when device d_i computes its task locally.

Definition 2 (Computation offloading strategy) The computation offloading strategy for all IoT devices is represented as $\boldsymbol{a} = \{a_1, a_2, \ldots, a_n\}$.

Then the offloading decisions for all devices other than device d_i are represented as $a_{-i} = \{a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n\}$. According to Definition 1, the set of users whose task is offloaded to the channel h_j^k of satellite s_j is

$$\mathcal{D}_j^k = \{d_i \in \mathcal{D} \mid a_i = (j,k)\}, s_j \in \mathcal{S}, h_j^k \in \mathcal{H}_j \tag{1}$$

2. Coverage time model for LEO satellite

To comprehend the movement trend and communication time of the LEO satellite, we plot a spatial geometry relation between IoT devices and LEO satellites. Figure 2 shows the communication state of LEO satellites for computation offloading. The elevation angle between the IoT device on the ground and the LEO satellite can be regarded as angle α by geometric transformation. Based on the law of sines, it can be further found that

$$\frac{r}{\sin\beta} = \frac{r+h}{\sin\left(\dfrac{\pi}{2}+\alpha\right)} = \frac{r+h}{\cos\alpha} \tag{2}$$

where r is the average radius of the earth, and h represents the distance between the substantial point and the LEO satellite. The space length from the LEO satellite to the IoT device is l, and the angle among l with h is denoted as β.

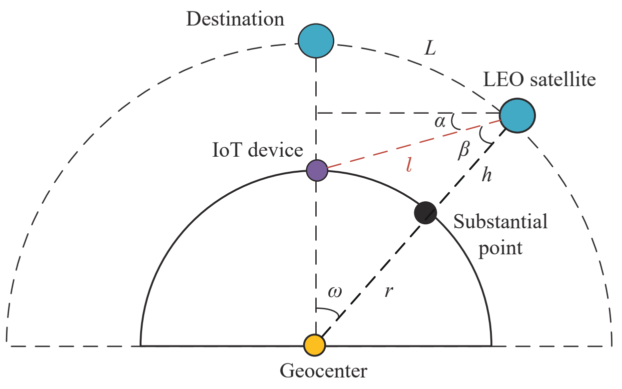


Figure 2 The spatial geometry relation between IoT devices and LEO satellites.

We assume that the line between the geocenter and the LEO satellite is y_1, the line between the geocenter and the destination of the LEO satellite is y_2, and ω is the angle among y_1 with y_2. It is straightforward to get $\omega + \alpha = \frac{\pi}{2} - \beta$, and the relationship between the three angles α, β, and ω can be expressed as

$$\cos(\omega+\alpha) = \cos\left(\frac{\pi}{2}-\beta\right) = \sin\beta \tag{3}$$

Combining (2) and (3), we can derive that the association between ω and α should be indicated as

$$\cos(\omega+\alpha) = \frac{r}{r+h}\cdot\cos\alpha \tag{4}$$

According to (4), we can further obtain the value of ω, which is denoted as

$$\omega = \arccos\left(\frac{r}{r+h}\cdot\cos\alpha\right) - \alpha \tag{5}$$

For each IoT device d_i in the LEO satellite edge computing, according to the Pythagorean theorem, the linear distance between device d_i and LEO satellite s_j is indicated by

$$l_i^j = \sqrt{r^2 + (r+h)^2 - 2\cdot r\cdot(r+h)\cdot\cos\omega} \tag{6}$$

The distance from the LEO satellite to its destination is regarded as an arc length, which is expressed as

$$L = 2\cdot\omega\cdot(r+h) \tag{7}$$

From the above, the maximum coverage time for the LEO satellite when an IoT device offloads its computation task to the satellite can be represented by

$$T_{\text{sat}}^{\max} = \frac{L}{v_{\text{sat}}} \tag{8}$$

where the velocity of the LEO satellite operating in the orbit is v_{sat}.

3. QoS model for computation offloading

Based on Section II.1, there are two ways to offload its computation task for the IoT device, namely LEO satellite computing and local computing. Then the QoS cost of the IoT device will be obtained. All details are listed below.

1) LEO satellite computing: For the LEO satellite computing, the computation task K_i of each IoT device d_i is offloaded to channel h_j^k of LEO satellite s_j, i.e., the computation offloading decision of device d_i is $a_i = (j, k)$. Then the data rate for computation task transmission from IoT device d_i to LEO satellite s_j through channel h_j^k is

$$R_{i,j}^k = B_j^k \log_2\left(1 + \frac{g_{i,j}^k p_{i,j}^k}{\sum_{d_t\in\mathcal{D}_j^k\setminus\{d_i\}} g_{t,j}^k p_{t,j}^k + N_0}\right) \tag{9}$$

where $g_{i,j}^k$ represents the channel gain between IoT device d_i and LEO satellite s_j, $p_{i,j}^k$ is the transmission power of device d_i for computation offloading, and N_0 indicates the background noise power. $\sum_{d_t\in\mathcal{D}_j^k\setminus\{d_i\}} g_{t,j}^k p_{t,j}^k$ denotes the intra-cell interference received by device d_i, that is, the sum of power interference from devices except for device d_i that offload their computation tasks to LEO satellite s_j through channel h_j^k [12].

The total delay for IoT device d_i offloading its computation task to LEO satellite s_j is

$$T_i^{\text{sat}} = \frac{l_i^j}{c} + \frac{Z_i}{R_{i,j}^k} + \frac{X_i}{f_i^j} \tag{10}$$

where l_i^j/c represents the propagation delay of computation task K_i in vacuum medium, and c is the speed of light. $Z_i/R_{i,j}^k$ indicates the task transmission delay for computation offloading. Then X_i/f_i^j denotes the computing delay of computation task K_i in LEO satellite s_j, where f_i^j is the computing capability allocated to IoT device d_i by LEO satellite s_j. In the satellite edge computing system, when a terminal has a data transmission demand, it needs to send a scheduling request before the edge server can allocate resources for the terminal to offload the computation task. This process causes the transmission delay for computation offloading to be significantly larger than the downlink transmission delay. Therefore, the downlink data transmission delay is basically negligible. Additionally, the energy consumption of device d_i for its computation task transmission is

$$E_i^{\text{sat}} = p_i \frac{Z_i}{R_{i,j}^k} \tag{11}$$

2) Local computing: For the local computing, the computation task K_i of each IoT device d_i is computed locally, i.e., the computation offloading decision of d_i is $a_i = (0,0)$. There is the computing capability f_i^{loc} of each IoT device d_i, which is related to its chip architecture. The delay of device d_i for computing its task K_i is

$$T_i^{\text{loc}} = \frac{X_i}{f_i^{\text{loc}}} \tag{12}$$

Let θ_i represent the energy consumption factor of IoT device d_i for task computing, the energy consumption of device d_i for local computing is

$$E_i^{\text{loc}} = \theta_i (f_i^{\text{loc}})^2 X_i \tag{13}$$

3) QoS cost: In order to emphasize the QoS of computation service, the QoS delay for IoT device d_i is

$$Q_i(a_i, a_{-i}) = T_i^{\text{sat}} I_{\{a_i\}} + T_i^{\text{loc}} (1 - I_{\{a_i\}}) \tag{14}$$

where $I_{\{a_i\}}$ is denoted by

$$I_{\{a_i\}} = \begin{cases} 1, & a_i \neq (0,0) \\ 0, & a_i = (0,0) \end{cases} \tag{15}$$

Considering the impact of device d_i on other devices, the sum of QoS delay for all devices except device d_i that does not ignore the computation offloading decision a_i of device d_i is M_i^{pre}, which is denoted as

$$M_i^{\text{pre}} = \sum_{d_t \in \mathcal{D} \setminus \{d_i\}} Q_t(a_t, a_{-t}) \tag{16}$$

While the sum of QoS delay for the devices except for device d_i overlooking decision a_i is M_i^{pos} [13], which is expressed as

$$M_i^{\text{pos}} = \sum_{d_t \in \mathcal{D} \setminus \{d_i\}} Q_t(a_t, a_{-t\setminus i}) \tag{17}$$

where $a_{-t\setminus i}$ indicates that the computation offloading decisions for other devices do not contain the offloading decision a_i for device d_i, i.e., $a_i \notin a_{-t}$. Equation (17) is actually the sum of QoS delay for all devices except for device d_i, which are supposed to consider the decisions of other devices when making its decision but ignore the decision a_i of device d_i. Therefore, the difference between (16) and (17) represents the impact of device d_i on the QoS delay of other devices. Therefore, the impact of device d_i on other devices is reflected as the difference between M_i^{pre} and M_i^{pos}, which is indicated by

$$\Delta M_i = M_i^{\text{pre}} - M_i^{\text{pos}} \tag{18}$$

Given the computation offloading decisions a_{-i} for other devices, we take the sum of QoS delay $Q_i(a_i, a_{-i})$ and the impact ΔM_i as the QoS cost of device d_i, which is expressed as

$$C_i(a_i, a_{-i}) = Q_i(a_i, a_{-i}) + \Delta M_i \tag{19}$$

4. Problem formulation

In the LEO satellite edge computing system, we study the QCO problem for providing services to IoT devices in harsh environments. To improve the computing offloading efficiency and minimize the total QoS cost of all devices, the QCO problem is modeled as a constrained optimization problem for each device $d_i \in \mathcal{D}$, which is expressed as

$$\min \sum_{d_i \in \mathcal{D}} C_i(a_i, a_{-i}) \tag{20}$$

$$\text{s.t. } I_{\{a_i=(j,k)\}} + I_{\{a_i=(0,0)\}} = 1, \forall d_i \in \mathcal{D} \tag{20a}$$

$$\sum_{k=1}^{q} \sum_{d_i \in \mathcal{D}_j^k} f_i^j \leq F_j, \forall d_i \in \mathcal{D}, \forall s_j \in \mathcal{S}, \forall h_j^k \in \mathcal{H}_j \tag{20b}$$

$$T_i^{\text{sat}} I_{\{a_i=(j,k)\}} \leq T_{\text{sat}}^{\max}, \forall d_i \in \mathcal{D} \tag{20c}$$

$$E_i^{\text{sat}} I_{\{a_i=(j,k)\}} + E_i^{\text{loc}} I_{\{a_i=(0,0)\}} \leq E_i^{\text{loc}}, \forall d_i \in \mathcal{D} \tag{20d}$$

where (20a) indicates the decision constraint that only one computation offloading decision is made for device d_i, and (20b) denotes the resource constraint that the sum of computing capability allocated to IoT devices by LEO satellite s_j is not greater than the computing capability of satellite s_j, which is related to the physical characteristics of the edge server. Equation (20c) denotes that the total delay for device d_i that offloads its computation task to LEO satellite s_j is not supposed to exceed the coverage time of the LEO satellite. The reason is that if the total delay of device d_i exceeds the coverage communication time of LEO satellite s_j, the computation offloading service between device d_i and satellite s_j will be interrupted. Equation (20d) guarantees that the energy consumption for local computing is the maximum energy consumption of device d_i. The CPU architecture of the device for local computing causes more energy consumption than the energy consumption generated by satellite edge computing.

The optimization problem in (20) is an NP-hard problem due to the discrete nature of computation offloading and the limited computing resources of LEO satellites [12]. Since the computation offloading decision of the device while satisfying the constraints is a binary variable, there is a huge challenge to find the centralized optimal solution of the QCO problem within polynomial time complexity.

III. Game-Theoretical Computation Offloading in LEO Satellite Edge Computing

To solve the QCO problem, the QCO game is presented through our game-theoretical approach in this section. Specifically, we formulate the QoS-aware offloading problem as the model of a non-cooperative game and propose a distributed algorithm that can obtain the Nash equilibrium solution [14], [15].

1. Computation offloading game model

At first, we consider the computation offloading decision problem among the IoT devices in this system. Given the offloading decision set a_{-i} for IoT devices other than device d_i, device d_i would like to make an appropriate decision a_i to minimize its own QoS cost, which can not only offload its computation task to LEO satellite $s_j \in \mathcal{S}$ through channel $h_j^k \in \mathcal{H}_j$, but compute the task locally, i.e.,

$$\min_{a_i \in \{(j,k) \cup (0,0) | s_j \in \mathcal{S}, h_j^k \in \mathcal{H}_j\}} C_i(a_i, a_{-i}) \tag{21}$$

According to (21), we formulate the QCO problem as a non-cooperative game:

$$\mathcal{G} = \{\mathcal{D}, \{\mathcal{A}_i\}_{d_i \in \mathcal{D}}, \{C_i(a_i, a_{-i})\}_{d_i \in \mathcal{D}}\}$$

where $\mathcal{D}$ denotes the set of IoT devices, $\mathcal{A}_i$ indicates the set of finite computation offloading decisions for device d_i, and $C_i(a_i, a_{-i})$ represents the QoS cost of device d_i produced by its decision $a_i \in \mathcal{A}_i$. In this game, there is non-cooperative competition among IoT devices. For example, several IoT devices select to offload their computation tasks through a specific channel to a specific LEO satellite, which may affect the offloading decisions of other IoT devices for the same satellite channel or any satellite channel [16]. For investigating this non-cooperative competition, we need to introduce the concept of Nash equilibrium in Definition 3.

Definition 3 (Nash equilibrium) A computation offloading strategy $a^* = \{a_1^*, a_2^*, \ldots, a_n^*\}$ of the QCO game is a Nash equilibrium if no device can further reduce its QoS cost by unilaterally changing its decision, i.e.,

$$C_i(a_i^*, a_{-i}^*) \le C_i(a_i, a_{-i}^*), \forall d_i \in \mathcal{D}, \forall a_i \in \mathcal{A}_i \tag{22}$$

Based on Definition 3, the Nash equilibrium for the QCO game is a stable offloading strategy. Given the offloading decisions of other IoT devices, the Nash equilibrium decision for each device is the best decision for itself, which is defined in Lemma 1.

Lemma 1 Any computation offloading decision a_i^* in the Nash equilibrium a^* is the best offloading decision of device d_i for $\mathcal{A}_i$ in response to a_{-i}.

Proof According to the disproof method, if the computation offloading decision $a_i^* \in \boldsymbol{a}^*$ is not the best selection for device d_i, there must be a preferable decision $a_i \in \mathcal{A}_i$ that will decrease its QoS cost, i.e.,

$$C_i(a_i, a_{-i}^*) < C_i(a_i^*, a_{-i}^*)$$

It is contradictory to (22) that no device in the Nash equilibrium can unilaterally reduce its QoS cost by changing its decision.

Lemma 1 shows that if the QCO game admits a Nash equilibrium, the QCO game will eventually reach the Nash equilibrium after a limited number of iterations. There must exist at least one Nash equilibrium in the potential game owing to its FIP, which is the finite improvement property of a potential game, indicating that the potential game can reach a Nash equilibrium within a finite improvement step. The reason is that the computation offloading decisions of the game model in this paper are a limited set. Each decision change based on the best response principle will reduce the QoS cost of IoT devices, so the improvement steps are finite. To verify the Nash equilibrium in the QCO game, we should demonstrate that the QCO game is a potential game, where the definition of a potential game is shown below.

Definition 4 (Ordinal potential game) A game is an ordinal potential game if there exists a potential function $\Phi(a_i, a_{-i})$ satisfying

$$C_i(a_i, a_{-i}) < C_i(a_i', a_{-i}) \Rightarrow \Phi(a_i, a_{-i}) < \Phi(a_i', a_{-i}) \tag{23}$$

According to Definition 4, we are supposed to prove that our QCO game $\mathcal{G}$ is an ordinal potential game, which is denoted in Theorem 1.

Theorem 1 The QCO game is an ordinal potential game, its potential function $\Phi(a_i, a_{-i})$ is expressed as

$$\Phi(a_i, a_{-i}) = \sum_{d_i \in \mathcal{D}} Q_i(a_i, a_{-i}) \tag{24}$$

Proof Given the computation offloading decisions a_{-i} of IoT devices except device d_i, the potential function may also be indicated as

$$\Phi(a_i, a_{-i}) = Q_i(a_i, a_{-i}) + \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} Q_t(a_t, a_{-t}) \tag{25}$$

For two different computation offloading decisions $a_i, a_i' \in \mathcal{A}_i$, the difference between $C_i(a_i, a_{-i})$ produced by decision a_i and $C_i(a_i', a_{-i})$ produced by decision a_i' is denoted by

$$\begin{aligned}
&C_i(a_i, a_{-i}) - C_i(a_i', a_{-i}) \\
&\quad = Q_i(a_i, a_{-i}) + \Delta M_i - (Q_i(a_i', a_{-i}) + \Delta M_i') \\
&\quad = Q_i(a_i, a_{-i}) + \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} \left(Q_t(a_t, a_{-t}) - Q_t(a_t, a_{-t\setminus i})\right) \\
&\qquad - Q_i(a_i', a_{-i}) - \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} \left(Q_t(a_t, a_{-t}') - Q_t(a_t, a_{-t\setminus i}')\right) \\
&\quad = Q_i(a_i, a_{-i}) - Q_i(a_i', a_{-i}) + \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} Q_t(a_t, a_{-t}) \\
&\qquad - \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} Q_t(a_t, a_{-t}') + M_{i'}^{\text{pos}} - M_i^{\text{pos}}
\end{aligned} \tag{26}$$

According to (17), M_i^{pos} denotes the sum of the QoS delay for all devices except device d_i and the decision a_i of device d_i will be ignored, i.e., M_i^{pos} has nothing to do with a_i. In other words, $M_{i'}^{\text{pos}}$ does not produce any change compared to M_i^{pos} when the decision is a_i', so we can get that $M_i^{\text{pos}} = M_{i'}^{\text{pos}} = \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} Q_t(a_t, a_{-t\setminus i}')$. Then the difference between $C_i(a_i, a_{-i})$ and $C_i(a_i', a_{-i})$ can further be denoted as

$$\begin{aligned}
&C_i(a_i, a_{-i}) - C_i(a_i', a_{-i}) \\
&\quad = Q_i(a_i, a_{-i}) - Q_i(a_i', a_{-i}) + \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} Q_t(a_t, a_{-t}) \\
&\qquad - \sum_{d_t \in \mathcal{D}\setminus\{d_i\}} Q_t(a_t, a_{-t}') \\
&\quad = \Phi(a_i, a_{-i}) - \Phi(a_i', a_{-i})
\end{aligned} \tag{27}$$

According to (24)–(27), we can deduce that the QCO game $\mathcal{G}$ is an ordinal potential game.

2. Distributed QoS-aware computation offloading algorithm design

We propose a distributed QoS-aware computation offloading (DQCO) algorithm to minimize the total QoS cost for IoT devices. In the DQCO algorithm, the primary experiment parameters are defined first. Then the computation offloading strategy for IoT devices reaches the Nash equilibrium after a limited number of iterations based on the FIP [17]. Specially, we denote $\sum_{d_i \in \mathcal{D}} C_i(a_i, a_{-i})$ as $V(a_i, a_{-i})$ to simplify English writing. The DQCO algorithm is defined in Algorithm 1, the main procedures are as follows:

1) Initialization: Given $\mathcal{D}$, $\mathcal{S}$, $\mathcal{H}_j$ for each LEO satellite s_j, and other required parameters, we initialize the offloading decision of each device d_i to $a_i = (0, 0)$ (lines 1–3).

2) Iteration process: In each iteration (line 4), the DQCO algorithm first calculates the current total QoS cost of all IoT devices denoted by $V(a_i, a_{-i})$ (line 5). For each IoT device $d_i \in \mathcal{D}$ in parallel, we traverse each LEO satellite $s_j \in \mathcal{S}$ and each of its channel $h_j^k \in \mathcal{H}_j$. Then, we calculate the total QoS cost $V(a_i', a_{-i})$ assuming that the decision of d_i is $a_i' = (j, k)$ (lines 6–11). We then need to find the decision $a_i^o \in \mathcal{A}_i$ that obtains the minimum total QoS cost $V(a_i^o, a_{-i})$ (line 12). When the decision a_i^o for each device d_i meets that $a_i^o \neq a_i$ and the state that $V(a_i^o, a_{-i}) < V(a_i, a_{-i})$ is satisfied, d_i competes for the opportunity to update its decision, and d_i can update its decision from a_i to a_i^o if it wins the competition (lines 13–19).

3) Ending of iteration: No device would like to update its decision (line 20).

Algorithm 1 Distributed QoS-aware computation offloading (DQCO) algorithm

```
Input: $\mathcal{D}, \mathcal{S}, \mathcal{H}_j$, and other primary parameters.
Output: device computation offloading strategy $a$.
 1: Initialization:
 2: The offloading decision of each device $d_i$ is $a_i = (0, 0)$;
 3: End of initialization.
 4: repeat
 5:   Compute the current total QoS cost $V(a_i, a_{-i})$;
 6:   for each device $d_i \in \mathcal{D}$ in parallel do
 7:     for each LEO satellite $s_j \in \mathcal{S}$ do
 8:       for each channel $h_j^k \in \mathcal{H}_j$ do
 9:         Compute the total QoS cost $V(a_i', a_{-i})$ assuming that the decision of $d_i$ is $a_i' = (j, k)$;
10:       end for
11:     end for
12:     Find the offloading decision $a_i^o \in \mathcal{A}_i'$ that obtains the minimum total QoS cost $V(a_i^o, a_{-i})$;
13:     if $a_i^o \neq a_i$ and $V(a_i^o, a_{-i}) < V(a_i, a_{-i})$ then
14:       Contend for the opportunity to update its decision;
15:       if device $d_i$ wins the competition then
16:         Update its decision from $a_i$ to $a_i^o$;
17:       end if
18:     end if
19:   end for
20: until no device would like to update its decision.
```

IV. Performance Evaluation

In this section, we perform extensive parameter analysis and comparison experiments to evaluate the simulation performance of the DQCO algorithm. The details include experiment settings, parameter analysis, and comparison experiments.

1. Experiment settings

To precisely evaluate the performance of the DQCO algorithm, the parameters involved in the experiment are given in Table 1 [17]–[19].

Table 1 Experiment parameters

Parameter	Value
Number of LEO satellites	3
Number of IoT devices	40
Channel bandwidth	10 MHz
Transmission power of each IoT device	[200, 300] mW
Data size of each computation task	[0.10, 0.45] Mb
Background noise power	−100 dBm
Computing capability of each IoT device	1 GHz
Computing capability of each LEO satellite	30 GHz
Number of CPU cycles required for each bit of data	1000
Altitude of LEO satellite	784 km
Elevation angle between IoT devices and LEO satellites	20°

2. Parameter analysis

In the satellite edge computing system, IoT devices request computation offloading services from satellite edge servers, and there is competition among devices for server resources, channel resources, and computing resources. The computation offloading decision of each device d_i will not only affect the QoS delay of d_i itself but also may affect the QoS delay of other devices $d_t \in \mathcal{D}\backslash\{d_i\}$. Therefore, it is necessary to consider not only the QoS delay of each device d_i but the impact of device d_i on other devices when performing optimization. In other words, the smaller the QoS delay of the device and the impact of device d_i on other devices, the smaller the QoS cost of device d_i until it reaches an equilibrium state of the system.

Figure 3 shows the total QoS cost of IoT devices with a different number of IoT devices. As IoT devices grow from 5 to 40, the total QoS cost gradually increases from 0.18 to 6.05. When other parameters are not changed in the experiment, it can be seen from (9) that the increase in the IoT devices enhances the interference between devices, which not only increases the QoS delay of this device but makes the impact among devices more obvious. In other words, greater interference reduces the data rate at which the device offloads its computing task through one channel to the LEO satellite, and for each device, the QoS delay impact of that device on other devices is more pronounced. The local computing way for IoT devices is not affected by the above. Therefore, even

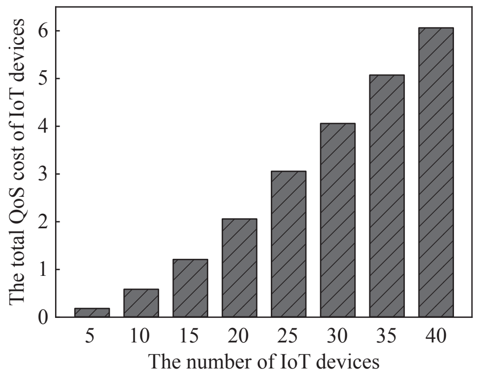


Figure 3 The total QoS cost of IoT devices with different numbers of IoT devices.

if many devices compute their tasks locally, the increase in the number of IoT devices leads to an increase in the total QoS cost of IoT devices.

Figure 4 shows the number of iterations in the DQCO algorithm with different numbers of IoT devices. It is obvious that the number of iterations keeps increasing from 11 to 38 as the number of IoT devices grows from 5 to 40. Because the increase in the number of devices makes the original solution space of the algorithm larger and the decision update process more complex. The competition among devices regarding the server and channel resources becomes more intense with the increase in the number of devices. In addition, when the number of IoT devices increases from 30 to 40, the number of iterations remains unchanged. The reason is that the available decision space for devices reaches a threshold value with limited server and channel resources.

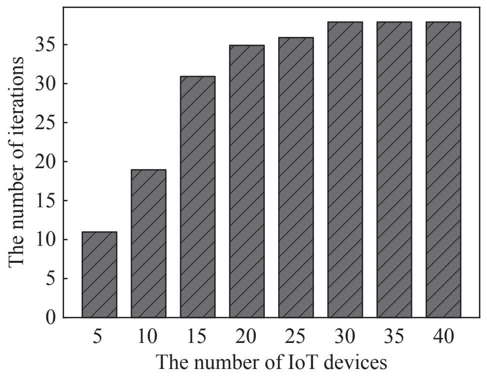


Figure 4 The number of iterations with different numbers of IoT devices.

Figure 5 shows the total QoS cost of IoT devices with different computing capabilities of each IoT device. It describes that the total QoS cost decreases from 2.05 to 0.92 when the computing capability of each device increases from 0.5 GHz to 4.0 GHz. There is almost no reduction in total QoS cost as the computing capability of each device increases from 0.5 GHz to 1.0 GHz, while the maximum decrease in total QoS cost as the computing capability increases from 1.5 GHz to 4.0 GHz is 16.6%. It is because the increase in device computing capability re-

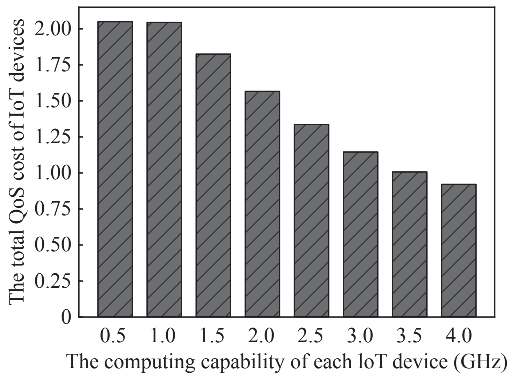


Figure 5 The total QoS cost of IoT devices with different computing capabilities of each IoT device.

sults in a significant reduction in the QoS delay of devices for local computing and a reduction in the number of devices competing for server and channel resources in this system. Thus the increase in computing capability reduces the QoS cost of each device for local computing, and the total QoS cost of all devices is lowered.

As shown in Figure 6, which shows the effect of different elevation angles between IoT devices and LEO satellites on the total QoS cost. It should be emphasized that elevation angle α in this experiment is the value in radians. The total QoS cost decreases from 2.85 to 1.73 when the elevation angle increases from 0.0 to 1.4 in Figure 6. We can infer that the total QoS cost decreases from 2.85 to 1.73 when the elevation angle increases from 0.0 to 1.4. The distance between IoT devices and LEO satellites is reduced when the elevation angle raises from 0.0 to 0.8, which will decrease the channel loss between devices and LEO satellites. It reduces the QoS delay of the device that offloads its computation task to the LEO satellites, and the total QoS cost will gradually decrease to 1.73. And after the elevation angle increases to 1.0, the channel loss and QoS delay are less than enough to make the number of devices that selects satellite computing essentially constant.

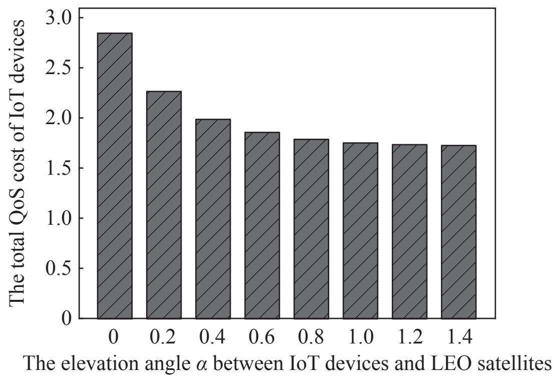


Figure 6 The total QoS cost of IoT devices with different elevation angle α (unit: radian) between IoT devices and LEO satellites.

3. Comparison experiments

The proposed DQCO algorithm is supposed to be compared with other computation offloading algorithms, which are as follows:

- Computing locally (CL): The computation task of each IoT device is computed by itself [20].
- Random: A computation offloading decision is randomly selected for each IoT device, i.e., local computing or offloading computation task to LEO satellite. If the device chooses to offload its task to the LEO satellite through the channel, which channel and which satellite for computation offloading is random. The total QoS cost of all devices is then calculated, and the final experiment result is the average value that repeats the above process two hundred times.

Figure 7 shows the total QoS cost of IoT devices in various algorithms with different numbers of IoT devices. Although the total QoS cost of the CL algorithm is lower than the total QoS cost of the Random algorithm only when the quantity of IoT devices is 40, the total QoS cost of the DQCO algorithm with a different number of devices is always smaller than that of the CL and Random algorithms. When the quantity of IoT devices raises from 5 to 40, the total QoS cost rises from 1.03 to 7.98 for the CL algorithm, from 0.63 to 8.67 for the Random algorithm, and from 0.18 to 6.05 for our proposed DQCO algorithm. Each device in the CL algorithm computes the task by itself, and the total QoS cost of all devices keeps increasing at a steady rate as the IoT devices increase. In the Random algorithm, the computation offloading way of each device is stochastic, which leads to a stochastic number of devices in different channels of different satellites, i.e., the channel interference to the devices is random. When the number of devices keeps increasing to 40, the total interference calculated by the satellites will keep increasing, and the impact on the total QoS cost will be more and more, so the total QoS cost of the Random algorithm is higher than the total QoS cost of CL algorithm when the number of devices is 40.

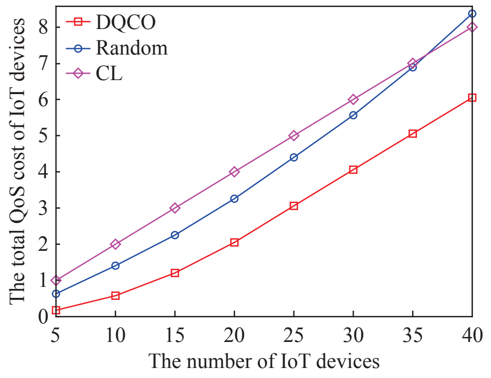


Figure 7 The total QoS cost of IoT devices for various algorithms with different numbers of IoT devices.

Figure 8 shows the total QoS cost of IoT devices for various algorithms with different data sizes of each computation task. When the data size of the computation task increases from 0.1 Mb to 0.45 Mb, the total QoS cost rises from 1.98 to 9.03 for the CL algorithm, from 1.64 to 7.17 for the Random algorithm, and from 1.07 to 4.48 for the proposed DQCO algorithm. As the amount of data for each computation task becomes larger, the device QoS delay for local computing and LEO satellite computing continues to increase, and the total QoS cost increases with it. But the DQCO algorithm finds a reasonable computation offloading decision for each device in parallel, updating the decision for the winner of the competition between devices in each iteration until Nash equilibrium is reached [21]. So the total QoS cost in the DQCO algorithm is definitely lower than that of the CL and Random algorithms, with a maximum difference of 4.52.

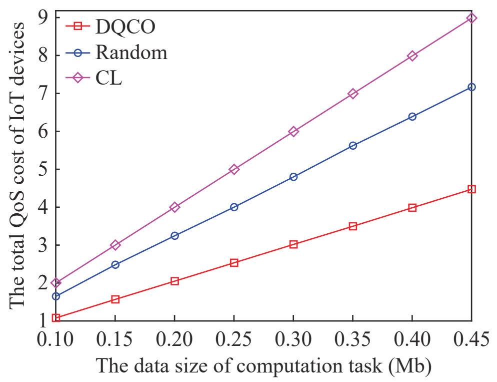


Figure 8 The total QoS cost of IoT devices for various algorithms with different data sizes of computation task.

Figure 9 shows the total QoS cost of the three algorithms with different computing capabilities of each IoT device. The total QoS cost of the DQCO algorithm is the smallest compared to the Random and CL algorithms. Specifically, the total QoS cost is reduced from 2.05 to 1.37 for the DQCO algorithm, from 3.23 to 2.06 for the Random algorithm, and from 4.01 to 1.66 for the CL algorithm. Because the devices in the CL algorithm choose local computing, the total QoS cost decreases with the increase in computing capability. The total QoS cost of the Random algorithm decreases with the increase in computing capability, the devices in the Random algorithm may choose satellite edge computing, and the total QoS cost of the Random algorithm will be higher than CL when the computing capability increases to a certain value such as 1.8 GHz. The DQCO algorithm reaches Nash equilibrium due to its best response principle and FIP [22]. The DQCO algorithm enormously reduces the cost of devices for computation offloading compared to the Random and CL algorithms.

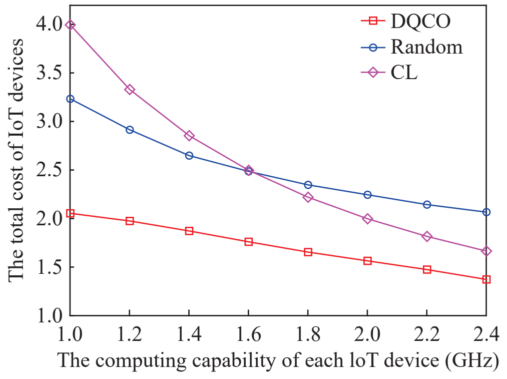


Figure 9 The total QoS cost of IoT devices for various algorithms with different computing capabilities of each IoT device.

As shown in Figure 10, the elevation angle α between IoT devices and LEO satellites is the independent variable for the comparison experiment, which results in the total QoS cost of IoT devices. The total QoS cost of the DQCO algorithm is always smaller than the total QoS cost of Random and CL algorithms as the elevation angle increases from 0.0 to 1.4. The total QoS cost of the DQCO algorithm is reduced from 2.85 to 1.73, the total QoS cost of the Random algorithm is reduced from 3.71 to 3.08, and the total QoS cost of the CL algorithm remains unchanged. Because the local computing method of the CL algorithm is not affected by the elevation angle α, the computation offloading decision of each device in the Random algorithm is random, while the DQCO algorithm gets the computation offloading strategy that each device is not willing to change its decision. Our proposed DQCO algorithm has a significant advantage over other offloading algorithms in solving the computation offloading problem in satellite edge computing.

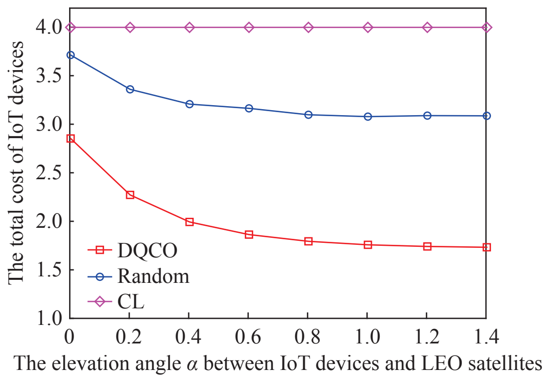


Figure 10 The total QoS cost of IoT devices for various algorithms with different elevation angle α (unit: radian) between IoT devices and LEO satellites.

V. Related Work

Satellite edge computing is playing an increasingly significant role to provide emerging application services for ground devices. Zhang *et al.* [23] proposed satellite mobile edge computing, in which IoT devices without near-end edge computing servers can request edge computing services via satellite links, consolidating network resources through a dynamic network virtualization technique. Li *et al.* [24] investigated how to efficiently deploy services on edge nodes in satellite edge computing to achieve robust perceptual service coverage with resource constraints, and introduced an online service deployment algorithm to face the challenges of dynamic systems and conflicting service coverage. Ding *et al.* [25]

used satellite edge computing to combine multi-user multi-computing tasks, computation task allocation, and resource allocation to minimize the weighted total energy consumption.

Recently, there has been large amounts of work for computation offloading in IoT systems. Mao *et al.* [26] investigated effective computation offloading strategies in green mobile edge computing systems with multiple devices, and execution cost is formulated as the performance metric, including execution delay and task failure. In order to solve the problem of computation offloading in communication networks with edge computing, Wang *et al.* [27] formulated the computation offloading decision and resource allocation as a convex optimization problem. Kong *et al.* [28] constructed a task offloading model to ensure the quality of user experience and promote system efficiency, and also performed resource preprocessing trust evaluation, and resource clustering before task processing. However, the computation offloading with edge computing in these studies unreasonably considers channel interference during task transmission or the devices in harsh environments.

The computation offloading problem of ground devices with LEO satellite edge computing in harsh environments and unexpected situations acquired considerable importance. Wang *et al.* [19] considered the intermittency of earth-satellite communication produced by the orbit operation of the satellite, and developed a computation offloading model in satellite edge computing. It calculated the response delay and energy consumption of the computation task based on queuing theory. Tang *et al.* [20] considered LEO satellite networks integrated with edge computing and investigated computation offloading decisions aimed at minimizing the total energy overhead of users while satisfying multiple constraints such as execution delay and computing capacity. At last, a double edge computation offloading algorithm is proposed in [29] to efficiently provide computing resources in satellites for edge users.

VI. Conclusion

In this paper, we investigate the QoS-aware computation offloading (QCO) problem in LEO satellite edge computing for Internet-of-things (IoT). Each IoT device can not only offload the computation task to low earth orbit (LEO) satellites by the channels but also compute its task locally. We aim to minimize the total QoS cost for IoT devices to heighten the service quality for IoT devices. The problem for QCO with multiple decision variables is NP-hard. We formulate this offloading problem as a game model of non-cooperative competition among devices named QCO game. Proposing a potential function for the QCO game, we demonstrate that the QCO game is an ordinal potential game. We then present the distributed QoS-aware computation offloading (DQCO) algorithm based on updated competition among devices, which can obtain the Nash equilibrium offloading strategy at the end of the iteration process. We conduct extensive parameter analysis experiments and comparison experiments to testify the practicality of the DQCO algorithm. In future work, we would like to consider the problem of computation offloading in networks combining LEO satellites and unmanned aerial vehicles with edge computing.

Acknowledgement

This work was supported by the Project of Cultivation for Young Top-motch Talents of Beijing Municipal Institutions (Grant No. BPHR202203225), the Young Elite Scientists Sponsorship Program by BAST (Grant No. BYESS2023031), and the National Key Research and Development Program of China (Grant No. 2022YFF06 04502).

References

[1] W. Obile, "Ericsson mobility report," https://www.ericsson.com/4ae28d/assets/local/reports-papers/mobility-report/documents/2022/ericsson-mobility-report-november-2022.pdf, 2022-11.

[2] Y. Chen, W. Gu, J. J. Xu, *et al.*, "Dynamic task offloading for digital twin-empowered mobile edge computing via deep reinforcement learning," *China Communications*, vol. 20, no. 11, pp. 164–175, 2023.

[3] J. W. Huang, J. Y. Wan, B. F. Lv, *et al.*, "Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning," *IEEE Systems Journal*, vol. 17, no. 2, pp. 2500–2511, 2023.

[4] P. Mróz, A. Otarola, T. A. Prince, *et al.*, "Impact of the spacex starlink satellites on the zwicky transient facility survey observations," *The Astrophysical Journal Letters*, vol. 924, no. 2, article no. L30, 2022.

[5] K. X. Li, J. Zhao, J. T. Hu, *et al.*, "Dynamic energy efficient task offloading and resource allocation for NOMA-enabled IoT in smart buildings and environment," *Building and Environment*, vol. 226, article no. 109513, 2022.

[6] Y. Chen, H. Xing, Z. Ma, *et al.*, "Cost-efficient edge caching for Noma-enabled IoT services," *China Communications*, in press, 2022.

[7] C. W. Wang, Y. L. Cui, D. H. Deng, *et al.*, "Trajectory optimization and power allocation scheme based on DRL in energy efficient UAV-aided communication networks," *Chinese Journal of Electronics*, vol. 31, no. 3, pp. 397–407, 2022.

[8] J. W. Huang, H. Gao, S. H. Wan, *et al.*, "AoI-aware energy control and computation offloading for industrial IoT," *Future Generation Computer Systems*, vol. 139, pp. 29–37, 2023.

[9] Y. H. Deng, F. Lyu, J. Ren, *et al.*, "AUCTION: Automated and quality-aware client selection framework for efficient federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1996–2009, 2022.

[10] Y. Chen, J. Zhao, Y. Wu, *et al.*, "QoE-aware decentralized task offloading and resource allocation for end-edge-cloud systems: A game-theoretical approach," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 769–784, 2024.

[11] Y. X. Zhang, F. Lyu, P. Yang, *et al.*, "IoT intelligence empowered by end-edge-cloud orchestration," *China Communications*, vol. 19, no. 7, pp. 152–156, 2022.

[12] J. J. Yu and Y. Wei, "Digital signal processing for high-speed THz communications," *Chinese Journal of Electronics*, vol. 31, no. 3, pp. 534–546, 2022.

[13] T. Fang, F. Yuan, L. Ao, *et al.*, "Joint task offloading, D2D pairing, and resource allocation in device-enhanced MEC: A potential game approach," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3226–3237, 2021.

[14] L. Chen, D. D. Jiang, R. Bao, *et al.*, "MIMO scheduling effectiveness analysis for bursty data service from view of QoE," *Chinese Journal of Electronics*, vol. 26, no. 5, pp. 1079–1085, 2017.

[15] Y. Chen, N. Zhang, Y. C. Zhang, *et al.*, "TOFFEE: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing," *IEEE Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1634–1644, 2021.

[16] F. Lyu, P. Yang, H. Q. Wu, *et al.*, "Service-oriented dynamic resource slicing and optimization for space-air-ground integrated vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7469–7483, 2022.

[17] C. S. Yang, X. L. Tao, F. Zhao, *et al.*, "Secure data transfer and deletion from counting bloom filter in cloud computing," *Chinese Journal of Electronics*, vol. 29, no. 2, pp. 273–280, 2020.

[18] X. Chen, L. Jiao, W. Z. Li, *et al.*, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[19] Y. X. Wang, J. Yang, X. Y. Guo, *et al.*, "A game-theoretic approach to computation offloading in satellite edge computing," *IEEE Access*, vol. 8, pp. 12510–12520, 2019.

[20] Q. Q. Tang, Z. S. Fei, B. Li, *et al.*, "Computation offloading in LEO satellite networks with hybrid cloud and edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 9164–9176, 2021.

[21] Y. X. Zhang, J. Ren, J. G. Liu, *et al.*, "A survey on emerging computing paradigms for big data," *Chinese Journal of Electronics*, vol. 26, no. 1, pp. 1–12, 2017.

[22] Y. Chen, Z. Y. Liu, Y. C. Zhang, *et al.*, "Deep reinforcement learning-based dynamic resource management for mobile edge computing in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4925–4934, 2021.

[23] Z. J. Zhang, W. Y. Zhang, and F. H. Tseng, "Satellite mobile edge computing: Improving QoS of high-speed satellite-terrestrial networks using edge computing techniques," *IEEE Network*, vol. 33, no. 1, pp. 70–76, 2019.

[24] Q. Li, S. G. Wang, X. Ma, *et al.*, "Service coverage for satellite edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 695–705, 2022.

[25] C. F. Ding, J. B. Wang, H. Zhang, *et al.*, "Joint optimization of transmission and computation resources for satellite and high altitude platform assisted edge computing," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1362–1377, 2022.

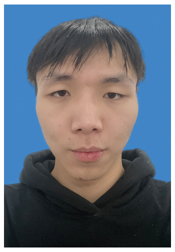
[26] Y. Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[27] C. M. Wang, C. C. Liang, F. R. Yu, *et al.*, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924–4938, 2017.

[28] W. P. Kong, X. Y. Li, L. Y. Hou, *et al.*, "A reliable and efficient task offloading strategy based on multifeedback trust mechanism for IoT edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13927–13941, 2022.

[29] Y. J. Wang, J. X. Zhang, X. Zhang, *et al.*, "A computation offloading strategy in satellite terrestrial networks with double edge computing," in *2018 IEEE International Conference on Communication Systems (ICCS)*, Chengdu, China, pp. 450–455, 2018.

Ying CHEN received the Ph.D. degree in 2017 from Tsinghua University, Beijing, China, and was a joint Ph.D. student with University of Waterloo, Waterloo, Canada, from 2016 to 2017. She is currently a Professor with Beijing Information Science and Technology University, Beijing, China. Her current research interests include Internet-of-things, mobile edge computing, wireless networks and communications, machine learning, etc.
(Email: chenying@bistu.edu.cn)



Jintao HU is currently pursuing the M.E. degree with Beijing Information Science and Technology University, Beijing, China. His current research interests include mobile edge computing and game theory.

Jie ZHAO is currently pursuing the M.E. degree with Beijing Information Science and Technology University, Beijing, China. His current research interests include mobile edge computing and wireless networks

Geyong MIN received the Ph.D. degree from University of Glasgow, Glasgow, UK, in 2003. He is currently a Professor with University of Exeter, Exeter, UK. His research interests include future Internet, computer networks, and wireless communications.