Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2179316, 10 pages
https://doi.org/10.1155/2018/2179316

Research Article

A Computing Offloading Game for Mobile Devices and Edge Cloud Servers

Meiwen Li, Qingtao Wu, Junlong Zhu, Ruijuan Zheng, and Mingchuan Zhang

College of Information Engineering, Henan University of Science and Technology, Luoyang 471000, China

Correspondence should be addressed to Qingtao Wu; wqt8921@haust.edu.cn

Received 25 September 2018; Accepted 12 November 2018; Published 2 December 2018

Academic Editor: Patrick Seeling

Computing offloading of mobile devices (MDs) through cloud is a greatly effective way to solve the problem of local resource constraints. However, cloud servers are usually located far away from MDs leading to a long response time. To this end, edge cloud servers (ECSs) provide a shorter response time due to being closer to MDs. In this paper, we propose a computing offloading game for MDs and ECSs. We prove the existence of a Stackelberg equilibrium in the game. In addition, we propose two algorithms, F-SGA and C-SGA, for delay-sensitive and compute-intensive applications, respectively. Moreover, the response time is reduced by F-SGA, which makes decisions quickly. An optimal decision is obtained by C-SGA, which achieves the equilibrium. Both algorithms above proposed can adjust the computing resource and utility of system users according to parameters control in computing offloading. The simulation results show that the game significantly saves the computing resources and response time of both the MD and the ECSs during the computing offloading process.

1. Introduction

The popularity of the Internet of Things (IoT) allows people to enjoy the convenience of the Internet in most scenarios of daily life. Especially for mobile devices (MDs), network services provide convenience and functional possibilities. However, functional and computationally intensive applications consume a large amount of energy and computing time on MDs, such as augmented reality [1] and face recognition [2]. Moreover, the MD is characterized by its mobility and portability with the poor CPU performance and limited battery power. To this end, the mobile cloud computing (MCC) is seen as an effective way to solve the problem of a shortage of local resources by offloading the computations to cloud infrastructure with remarkable computational power [3–5]. The popular approach is for offloading computing tasks to the public clouds such as Windows Azure and Amazon EC2. Although MCC provides considerable cloud resources, it cannot guarantee a low response time. Furthermore, the user experience is reduced due to the delay.

Edge cloud computing (ECC) is promising for mobile computing offloading, which also considers a promoter of 5G mobile networks because they are located near the edge of the network [6, 7], and has been extensively studied in recent years [8, 9]. As illustrated in Figure 1, edge cloud servers (ECSs) are closer to users, which greatly reduces the time for data transmission. Therefore, offloading computations to less resource ECSs is considered a more advantageous solution. Some previous works on ECC focus on reducing energy consumption such as [10, 11]. Although [12] considers the energy consumption of MDs and cloud servers in computing offloading, it does not analyze the computing performance of MDs and cloud servers. Refs. [13, 14] focus on computing performance enhancement. However, ECSs usually leased in real life scenarios rarely attract attention of researchers.

In this paper, we consider the equilibrium problem during the computing offloading process, which not only considers the needs of MDs, but also considers the maximum benefits of service providers. For this reason, we propose a strategy for computing offloading by computing the equilibrium between a MD and ECSs. Liu *et al.* [15] considered the computing offloading process between a remote cloud server and several ECSs, which concerned the benefits of different levels of service providers. However, they did not pay attention to the needs of mobile users, which also should be taken seriously

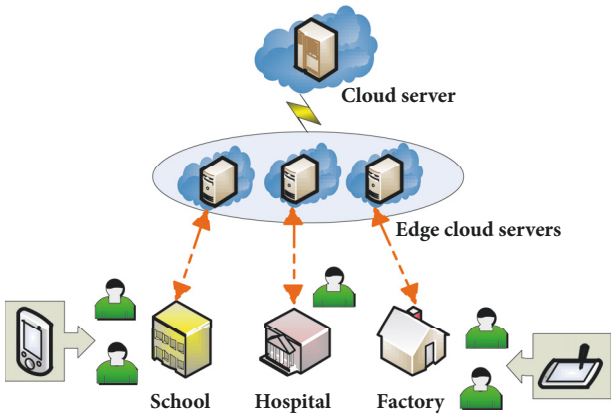

Figure 1: Edge cloud network.

because of its mobility and portability with limited resource [3]. Ref. [16] considered the mobile user that maximized their performance by choosing of the wireless access points for computing offloading. But they did not consider the benefit of mobile users. In this paper, we mainly focus on the features of mobile users, which concludes the issues of mobility, cost, and so on. We designed an efficient computing offloading strategy considering the scenario that a MD executes computing offloading through ECSs. Suppose a MD needs to offload its computing tasks to one of the sets of ECSs. The MD needs to negotiate an offloading policy with the ESCs to optimize the offloading efficiency of both the MD and the ECSs. For this reason, we propose a strategy among a MD and ECSs. Furthermore, we formulate the computing offloading process among the MD and the ECSs as a Stackelberg game. In particular, the MD increases computing efficiency by offloading complicated computation to the ECSs, and the ECSs obtain certain revenue by performing the computations which are offloaded by the MD. An equilibrium between the benefits of the MD and the ECSs is achieved through our proposed strategy. We make three important contributions in this paper.

(i) Realistically considering the real offloading scenario, we formulate the interaction of the MD and the ECSs during the computing offloading process as a Stackelberg game.

(ii) We prove the existence of equilibrium in the Stackelberg game. Furthermore, we propose the C-SGA for computing the equilibrium. And we also design the F-SGA for delay-sensitive applications, which greatly reduces response time.

(iii) We verify the performance of the proposed algorithms via the simulation experiments. The results show that the proposed algorithms increase the efficiency of computing offloading. In addition, we performed a detailed analysis of the performance of the strategy for the changes in the values of the different parameters.

The rest of the paper is organized as follows. We present the related work in Section 2. We present our system model and formulate the problem in Section 3. We analyze the Stackelberg game of our model in Section 4. We present our algorithms in Section 5. Performance evaluation is provided in Section 6. We conclude this paper in Section 7.

2. Related Work

Most previous works have been done on computing offloading [17–19]. The emergence of computing offloading techniques can be traced back to the concept of Cyber Foraging [20], which reduces the computation, storage, and energy of MDs by offloading tasks of MDs to nearby servers with sufficient resources. The main goal of computing offloading includes expanding CPU capacity, saving energy consumption of MDs, reducing service delays, and saving computational cost. Most of the early computing offloading techniques used static partitioning schemes, relying on programmers to statically divide the application into two parts: one part is executed on the MD; the other part is executed on the server. Li *et al.* [21] proposed a partitioning approach based on energy consumption. The communication energy consumption depends on the size of the transmitted data and the network bandwidth. The computational energy consumption depends on the number of instructions of the program. They obtain optimized program partitioning based on the consumption of computation and communication. Yang *et al.* [22] proposed a comprehensive consideration of the use of multiple resources, including CPU, memory, and communication costs. They offloaded some tasks on the MD to a nearby laptop with adequate resources.

Proposition of MAUI [23] is to provide a common dynamic computing offloading solution and minimize the burden on developers. The programmer only needs to divide the application into local methods and remote methods without having to make an offloading decision set for each program. In order to solve the problem of excessive transmission delay in wide area network (WAN), researchers have considered offloading tasks of MDs to infrastructure that are closer to the information source. Then, Satyanarayanan *et al.* [24] first proposed the concept of Cloudlet, which is defined as a trusted, resource-rich computing device or a group of computing devices to provide computing to nearby mobile terminals. Patel *et al.* [25] proposed the concept of MEC, which provides powerful computing capability in the wireless access network close to mobile users. MEC runs at the edge of the network and is logically independent of the rest of the network, which is important for applications with high security requirements [26, 27]. In addition, ECSs are particularly suited for dealing with massive analyses and mass data. At the same time, since the ECSs are geographically close to users, the delay of the network responding to the user request is greatly reduced, and the possibility of network congestion in the transmission network and the core network portion is also reduced [28]. There have been some previous studies on computing offloading using ECSs [29–31]. Wang *et al.* [32] propose a MEC-WPT design for computing offloading by considering multiuser mobile edge cloud system. Ref. [33] explored how infrastructure of edge computing can improve latency and energy consumption relative to the

cloud by analyzing multiple types of mobile applications. They demonstrate that the use of edge computing platforms in WIFI and LTE networks can significantly improve the latency of interactive and intensive computing applications. Sardellitti *et al.* [34] proposed a QoS-based incentive mechanism for mobile data offloading. The incentive mechanism is used in the quality-aware system to stimulate the users' participation and enhance the robustness of the system [35]. Neto *et al.* [36] designed a mobile offloading framework with an original decision engine, which can significantly reduce energy consumption.

Some of previous work conducted extensive research on game theory. There has been some research on the game that can be used in computing offloading [37, 38]. Chen *et al.* [39] proposed an approach for computing offloading for mobile cloud computing. In addition, they designed a computation offloading mechanism to achieve a Nash equilibrium of the game. Wang *et al.* [40] proposed an analysis framework based on evolutionary game theory. Xu *et al.* [41] designed a security-aware incentive mechanism for computation offloading by using game theory and epidemic theory.

3. System Model

As the primary deployment method for MEC [25], we consider the edge system in this paper consisting of a set of ECSs and several MDs. We assume that a MD decides to offload its computations to a set $\mathcal{S} = \{1, 2, \ldots, S\}$ of ECSs. X_i denotes the total computation of ECS i for all $\forall i \in \mathcal{S}$. x_i denotes the computation offloaded from the MD to ECS i, $x_i \in [0, X_i]$.

3.1. Mobile Device. We assume that computation offloaded profile is $\mathbf{x} = (x_1, \ldots, x_S)$; given $\mathbf{x}$, the local remaining unoffloaded computation is given by

$$\varphi(\mathbf{x}) = \sum_{i=1}^{S} (X_i - x_i), \tag{1}$$

and computations performed locally completely without offloading is given by

$$\varphi(\mathbf{0}) = \sum_{i=1}^{S} X_i. \tag{2}$$

The cost of local computing for the mobile device is given by

$$F(\mathbf{x}) = e^{\alpha(\varphi(\mathbf{x}))}, \tag{3}$$

where α is the modeling parameter. The payment profile is $\mathbf{m} = (m_1, \ldots, m_S)$. Given $\mathbf{m}$, the payment for offloading is given by

$$M(\mathbf{x}) = \sum_{i=1}^{S} m_i, \tag{4}$$

where the payment m_i is determined by offloading unit price p_i and computation amount x_i, i.e.,

$$m_i = p_i \cdot x_i. \tag{5}$$

We consider the consumption of performing computing offloading time as part of the offloading cost, which includes the time to compute local remaining computations and the time to transfer the data needed by computing offloading. K^d denotes the computational capability of the MD, and L denotes the transmission capacity of the MD. The total consumption of performing offloading time is given by

$$T(\mathbf{x}) = \sum_{i=1}^{S} \left(T_i^{exe} + T_i^{off}\right), \tag{6}$$

where T_i^{exe} represents the time of computing local remaining computations and is given by

$$T_i^{exe} = \beta \cdot \frac{x_i}{K^d}, \tag{7}$$

where β represents the complexity of the computation. T_i^{off} represents the time taken to transfer the data needed by computing offloading and is given by

$$T_i^{off} = \gamma \cdot \frac{x_i}{L}, \tag{8}$$

where γ represents the coefficient of the amount of data required by transferring computations.

Therefore, the cost of the MD for performing computing offloading is determined by the computing time, the transmission time, and the payment for the ECSs, i.e.,

$$\begin{aligned} C(\mathbf{x}) &= F(\mathbf{x}) + M(\mathbf{x}) + T(\mathbf{x}) \\ &= e^{\alpha(\varphi(\mathbf{x}))} + \sum_{i=1}^{S} m_i + \sum_{i=1}^{S} \left(T_i^{exe} + T_i^{off}\right). \end{aligned} \tag{9}$$

We define the local utility Γ as the cost reduction from performing offloading, i.e.,

$$\Gamma = C(\mathbf{0}) - C(\mathbf{x}) = C(\mathbf{0}) - F(\mathbf{x}) - M(\mathbf{x}) - T(\mathbf{x}). \tag{10}$$

3.2. Edge Servers. ECSs usually have their own computations to compute as shown in Figure 1. When the ECSs decide whether to perform the computing offloading for the MD, they must take their own computations into account. w_i denotes the revenue for one unit of ECS i to perform its own computations, and K_i^s the computational capability of the ECS i. Similar to previous work [42], we ignore the time for transmitting the computation results.

The profit of the ECS i is given by

$$\begin{aligned} G_i(x_i) = w_i \cdot (X_i - x_i) + m_i - e^{\alpha(X_i - x_i)} - e^{\alpha x_i} - \beta \\ \cdot \frac{x_i}{K_i^s}, \end{aligned} \tag{11}$$

where $X_i - x_i$ is the computation for its own users, and $e^{\alpha x_i}$ denotes the cost of computing x_i.

The utility of each ECS i, which is the profit improvement by performing computing offloading, is defined as

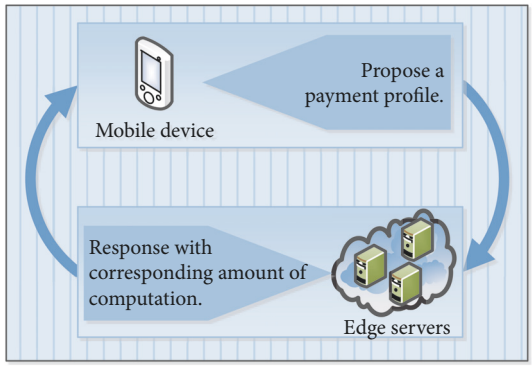

Figure 2: Two-step Stackelberg game.

$$\begin{aligned}\Psi_i(x_i) &= G_i(x_i) - G_i(0)\\ &= w_i \cdot (X_i - x_i) + m_i - e^{\alpha\cdot(X_i - x_i)} - e^{\alpha x_i} - \beta\\ &\quad \cdot \frac{x_i}{K_i^s} - w_i \cdot X_i + e^{\alpha X_i} + 1.\end{aligned} \tag{12}$$

3.3. Problem Formulation. The strategy among the MD and the ECSs is formulated as a Stackelberg game. Our strategy has two steps as shown in Figure 2. In the first step, the MD proposes a payment profile, denoted by $\mathbf{p} = (p_1, \ldots, p_S)$. In the second step, the ECSs respond with corresponding amount of computation for offloading, denoted by $\mathbf{x} = (x_1(p_1), \ldots, x_S(p_S))$.

We assume that the MD gives a first initial payment; then the optimal decision of each ECS i is obtained by solving the following optimization problem:

$$\begin{aligned}\max \quad & \Psi_i(x_i)\\ s.t. \quad & x_i \in [0, X_i].\end{aligned} \tag{13}$$

Next, we obtain the optimal strategy by maximizing the utility of the MD after receiving the decision of the ECS, i.e.,

$$\begin{aligned}\max \quad & \Gamma(\mathbf{x}, \mathbf{p})\\ s.t. \quad & p_i \geq 0, \quad \forall i = 1, \ldots, S, x_i(p_i) \leq X_i,\end{aligned} \tag{14}$$

where $\Gamma(\mathbf{x}, \mathbf{p})$ is the utility of the MD in (10).

4. Stackelberg Game Analysis

As discussed above, the MD determines the payment profile of computation offloading to the ECSs, while the ECSs provide corresponding amount of computation. We model the problem as a two-step Stackelberg game among the MD and the ECSs based on the noncooperative game theory. Explicitly, as shown in Figure 2, the MD acts as a leader proposing the payment profile. As followers of the MD, the ECSs can adjust the corresponding decisions.

4.1. Stackelberg Game Design. In this work, we denote by $x_i^\star$ the optimal strategy of the ECS i, which is the optimal problem of (13), and $\mathbf{x}^\star$ the optimal strategy profile. In addition, $p_i^\star$ denotes the optimal payment for the ECS i. A Nash equilibrium of the Stackelberg game is defined as $(\mathbf{x}^\star, \mathbf{p}^\star)$, in which none of the ECSs can further improve its profit by changing strategy.

Definition 1. A Nash equilibrium of the Stackelberg game is a strategy profile $(\mathbf{x}^\star, \mathbf{p}^\star)$, if the strategy satisfies the following conditions:

$$\begin{aligned}&\Psi(\mathbf{x}^\star) \geq \Psi(\mathbf{x}')\\ &\Gamma(\mathbf{x}^\star, \mathbf{p}^\star) \geq \Gamma(\mathbf{x}^\star, \mathbf{p}')\\ s.t. \quad & x_i \in [0, X_i], \quad \forall i = 1, \ldots, S,\\ & p_i \in [\check{p}_i, \widehat{p}_i], \quad \forall i = 1, \ldots, S,\end{aligned} \tag{15}$$

where $\widehat{p}_i$ is the upper bound of the payment p_i, and $\check{p}_i$ is the lower bound.

4.2. Nash Equilibrium Analysis. In this part, we will analyze the equilibrium in the Stackelberg game that we proposed and prove the existence of Nash equilibrium in the Stackelberg game. In order to confirm the existence of Nash equilibrium in our Stackelberg game, we have made the following proofs. We first discuss the existence and uniqueness of Nash equilibrium of the Stackelberg game.

Lemma 2. *The strategy profile set of the ECSs is nonempty, convex, and compact.*

Proof. First, according to the characteristics of ECSs, we calculate the first-order derivative of Ψ_i to x_i that is

$$\frac{\partial \Psi_i}{\partial x_i} = \alpha \cdot \left(e^{(X_i - x_i)} - e^{\alpha x_i}\right) - w_i - \frac{\beta}{K_i^s} + p_i. \tag{16}$$

According to the first-order derivative, we calculate the second-order derivative of Ψ_i to x_i that is

$$\frac{\partial^2 \Psi_i}{\partial x_i^2} = -\alpha^2 \cdot \left(e^{\alpha(X_i - x_i)} + e^{\alpha x_i}\right). \tag{17}$$

We can obtain the optimal strategy of ECS i by taking (16) to zero

$$x_i^\star(p_i) = \frac{1}{2} \cdot \left(X_i - \frac{1}{\alpha} \cdot \ln \frac{1}{\alpha}\left(w_i + \frac{\beta}{K_i^s} - p_i\right)\right). \tag{18}$$

Then we can obtain the payment's lower bound of ECS i by taking x_i to zero,

$$\check{p}_i = w_i + \frac{\beta}{K_i^s} - \alpha \cdot e^{\alpha X_i}, \tag{19}$$

and we can obtain the payment's upper bound of ECS i by taking x_i to X_i,

$$\widehat{p}_i = w_i + \frac{\beta}{K_i^s} - \alpha \cdot e^{-\alpha X_i}. \tag{20}$$

Therefore, Lemma 2 is proved completely. □

Lemma 3. *The ECS i has a unique optimal strategy after receiving the mobile's strategy.*

Proof. Obviously, if the MD gives a p_i that $p_i > \widehat{p}_i$, the ECS i will perform all of its computation for the MD. However, if $p_i < \check{p}_i$, the ECS i does not perform the offloading.

When $\check{p}_i < p_i < \widehat{p}_i$, the expression of the strategy is given by (18), and we have

$$\frac{\partial x_i^*}{\partial p_i} = \frac{1}{2\alpha \cdot (w_i + \beta/K_i^s - p_i)}, \tag{21}$$

$$\frac{\partial^2 x_i^*}{\partial p_i^2} = \frac{1}{2 \cdot \alpha \left(w_i + \beta/K_i^s - p_i\right)^2}. \tag{22}$$

Evidently, the relation (21) implies that x_i^* is an increasing function of p_i, which means that the higher payment the MD offers, the more computation the ECS i provides. Equation (22) implies that x_i^* is a concave function of p_i. Since $x_i \in [0, X_i]$, $\Psi_i(x_i)$ is strictly concave in $\mathbf{x}^*$, and thus the strategy is unique and optimal. Lemma 3 is proved. □

Lemma 4. *For the optimal strategies of ECSs, the MD has a unique optimal strategy.*

Proof. The first-order partial derivative of Γ to p_i is

$$\begin{aligned}\frac{\partial \Gamma}{\partial p_i} &= C'\left(\varphi\left(\mathbf{x}^*\right)\right) \cdot \frac{\partial x_i^*}{\partial p_i} - x_i^* - p_i \cdot \left(\frac{\beta}{K^d} + \frac{\gamma}{L}\right) \cdot \frac{\partial x_i^*}{\partial p_i} \\ &= C'\left(\varphi\left(\mathbf{x}^*\right)\right) \cdot \frac{\partial x_i^*}{\partial p_i} - x_i^* - \left(p_i + \frac{\beta}{K^d} + \frac{\gamma}{L}\right) \\ &\quad \cdot \frac{\partial x_i^*}{\partial p_i}.\end{aligned} \tag{23}$$

The second-order partial derivative of Γ to p_i is

$$\begin{aligned}\frac{\partial^2 \Gamma}{\partial p_i^2} &= -C''\left(\varphi\left(\mathbf{x}^*\right)\right) \cdot \left(\frac{\partial x_i^*}{\partial p_i}\right)^2 + C'\left(\varphi\left(\mathbf{x}^*\right)\right) \\ &\quad \cdot \frac{\partial^2 x_i^*}{\partial p_i^2} - 2 \cdot \frac{\partial x_i^*}{\partial p_i} - \left(p_i + \frac{\beta}{K^d} + \frac{\gamma}{L}\right) \\ &\quad \cdot \frac{\partial^2 x_i^*}{\partial p_i^2},\end{aligned} \tag{24}$$

$$\frac{\partial^2 \Gamma}{\partial p_i \partial p_j} = -C''\left(\varphi\left(\mathbf{x}^*\right)\right) \cdot \frac{\partial x_i^*}{\partial p_i} \cdot \frac{\partial x_j^*}{\partial p_j}. \tag{25}$$

We define two auxiliary matrices as follows,

$$\begin{aligned}H_i = \operatorname{diag}&\left[C'\left(\varphi\left(\mathbf{x}^*\right)\right) - \left(p_1 + \frac{\beta}{K^d} + \frac{\gamma}{L}\right)\right] \cdot \frac{\partial^2 x_1^*}{\partial p_1^2} \\ &- 2 \cdot \frac{\partial x_1^*}{\partial p_1}, \ldots, \left[C'\left(\varphi\left(\mathbf{x}^*\right)\right) - \left(p_S + \frac{\beta}{K^d} + \frac{\gamma}{L}\right)\right] \\ &\cdot \frac{\partial^2 x_S^*}{\partial p_S^2} - 2 \cdot \frac{\partial x_S^*}{\partial p_S},\end{aligned} \tag{26}$$

$$H_2 = -C''\left(\varphi\left(\mathbf{x}^*\right)\right) \begin{bmatrix} \frac{\partial x_1^*}{\partial p_1} \\ \frac{\partial x_2^*}{\partial p_2} \\ \vdots \\ \frac{\partial x_S^*}{\partial p_S} \end{bmatrix} \left[\frac{\partial x_1^*}{\partial p_1} \frac{\partial x_2^*}{\partial p_2} \cdots \frac{\partial x_S^*}{\partial p_S}\right], \tag{27}$$

and thus, the Hessian matrix of $\Gamma(\mathbf{x}^*, \mathbf{p}^*)$ is given by

$$H = H_1 + H_2. \tag{28}$$

Furthermore, for any nonzero column vector z, we have

$$z^T H z = z^T H_1 z + z^T H_2 z. \tag{29}$$

From (23), we have $p_i < C'(\varphi(\mathbf{x}^*))$. Since $\partial x_i^*/\partial p_i \geq 0$, $\partial^2 x_S^*/\partial p_S^2 \leq 0$, $C''(\varphi(\mathbf{x}^*)) \geq 0$, we have

$$z^T H_1 z \leq 0, \tag{30}$$

$$z^T H_2 z \leq 0. \tag{31}$$

Thus, we obtain

$$z^T H z \leq 0. \tag{32}$$

From the inequality (32), $\Gamma(\mathbf{x}, \mathbf{p})$ is strictly concave, and then (15) is a convex optimization problem. Since H is strictly negative, $\mathbf{p}^*$ is unique, which is the optimal strategy of MD. Lemma 4 is proved. □

With Lemmas 2, 3, and 4 in place, we prove the following theorem.

Theorem 5. *A unique Nash equilibrium exists among the MD and the ECSs in our proposed Stackelberg game.*

5. Algorithm Design

In this section, the process of strategy is shown in Figure 3. We propose two algorithms, F-SGA and C-SGA, for delay-sensitive services and compute-intensive services, respectively. Generally, delay-sensitive services require strict response time, which requires the system to make decisions quickly. Compute-intensive services require a large amount of computing by the servers. F-SGA, a fast Stackelberg game algorithm, can make decisions quickly due to its simple decision mechanism. C-SGA, a complex Stackelberg game algorithm, is slower than F-SGA in decision making speed, but it provided more accurate price and computation to maximize the benefit of MD and ECSs. F-SGA and C-SGA are described in detail as follows.

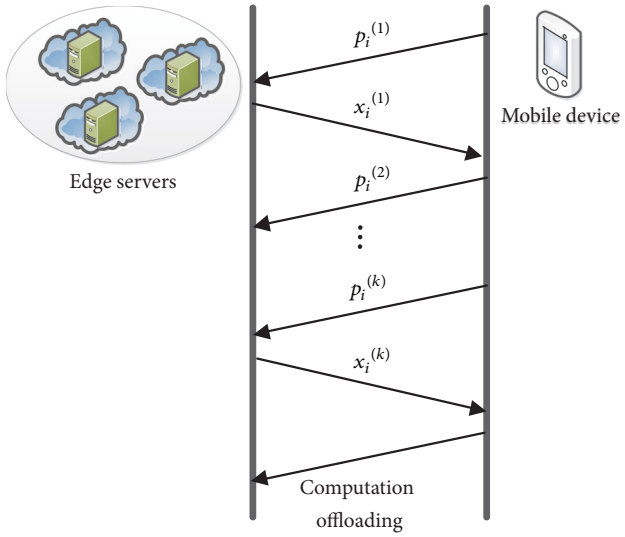

Figure 3: The game process of our Stackelberg game.

Input: $p_{min} = \min_{i \in S} \check{p}_i$, $p_{max} = \max_{i \in S} \widehat{p}_i$.
Output: $\mathbf{x}, \mathbf{p}$.
1: **if** $p_{max} - p_{min} \geq \xi$ **then**
2: Let $p_d = 2p_{min}/3 + p_{max}/3$;
3: Let $p_u = p_{min}/3 + 2p_{max}/3$;
4: Let $p_i = (p_d + p_u)/2$;
5: **else**
6: Let $p_i = (p_{max} + p_{min})/2$;
7: **end if**
8: **for all** i such that $i \in S$ **do**
9: Calculate $x_i(p_i)$ according to (18);
10: **end for**
11: Calculate $\Gamma(\mathbf{x}, \mathbf{p})$ according to (10);

Algorithm 1: F-SGA.

5.1. F-SGA. We propose F-SGA to quickly achieve equilibrium as shown in Algorithm 1. Because the algorithm can quickly obtain the optimal objective for the MD and the ECSs, it is suitable for computing offloading of latency-sensitives applications.

As discussed before, we set two points, denoted as p_{min} and p_{max}. Initially, we set $p_{min} = 0$, $p_{max} = \max_{i \in S} \widehat{p}_i$. Then we describe F-SGA in detail. The ECSs first send their payment to the MD. After receiving the payment of ECSs, F-SGA proceeds to the next step by comparing the difference between p_{min} and p_{max}. It means that when the difference of payment from the ECSs is small, we simply compromise the payment of computing offloading. But when the different of payment is large, F-SGA will also quickly determine the payment of computing offloading, but more detailed.

5.2. C-SGA. The algorithm proposed in Algorithm 1 needs less time and has a faster response speed, but it directly determines the payment, which leads to an increase in the cost of computing offloading. We next propose C-SGA, which uses the fine iteration to obtain the optimal strategy. C-SGA are shown in Algorithm 2.

Input: $p_{min} = 0$, $p_{max} = \max_{i \in S} \widehat{p}_i$.
Output: $\mathbf{x}$,$\mathbf{p}$.
1: Let $p_d = 2p_{min}/3 + p_{max}/3$;
2: Let $p_u = p_{min}/3 + 2p_{max}/3$;
3: **while** $p_{max} - p_{min} \geq \xi$ **do**
4: **for all** i such that $i \in S$ **do**
5: Calculate $x_i(p_d)$ according to (18);
6: **end for**
7: Calculate $\Gamma(\mathbf{x}, \mathbf{p}_d)$ according to (10);
8: **for all** i such that $i \in S$ **do**
9: Calculate $x_i(p_u)$ according to (18);
10: **end for**
11: Calculate $\Gamma(\mathbf{x}, \mathbf{p}_u)$ according to (10);
12: **if** $\Gamma(\mathbf{x}, \mathbf{p}_d) < \Gamma(\mathbf{x}, \mathbf{p}_u)$ **then**
13: Let $p_{min} = p_d$;
14: **else**
15: Let $p_{max} = p_u$;
16: **end if**
17: **end while**
18: Let $p_i = p_{min}$;
19: **for all** i such that $i \in S$ **do**
20: Calculate $x_i(p_i)$ according to (18);
21: **end for**
22: Calculate $\Gamma(\mathbf{x}, \mathbf{p})$ according to (10);

Algorithm 2: C-SGA.

In contrast, C-SGA only needs the ECSs to compare the the bound of payment, which not only greatly reduces the computation of the ECSs, but also obtains more optimized payment. Therefore, C-SGA is suitable for situations where the ECSs have less computation resources. The process of C-SGA is summarized as follows.

(1) The MD first sends the initial value of payment. With the strategy $(\mathbf{x}^\star, \mathbf{p}^\star)$, the ECSs compute the optimal computation for offloading based on (18) and send the computing results to the MD.

(2) After receiving the results, the MD calculates the utility function $\Gamma(\mathbf{x}, \mathbf{p_d})$ and $\Gamma(\mathbf{x}, \mathbf{p}_u)$ of the strategy. Then we compare $\Gamma(\mathbf{x}, \mathbf{p}_d)$ and $\Gamma(\mathbf{x}, \mathbf{p}_u)$. If $\Gamma(\mathbf{x}, \mathbf{p}_d) < \Gamma(\mathbf{x}, \mathbf{p}_u)$, this implies that the optimal value must be located between p_d and p_u, and thus we set $p_{min} = p_d$, $p_{max} = p_u$.

(3) Then the MD and the ECSs continue to perform procedures (1) and (2) until $p_{max} - p_{min} < \xi$, where ξ is supposed to make our algorithm more accurate.

Theorem 6. *The proposed Algorithm 2 can reach a unique Nash equilibrium.*

Proof. Since Γ is strict convex, the optimal strategy $\mathbf{p}^\star$ is unique. We can obtain $\mathbf{p}^\star$ by Algorithm 2 to maximize Γ. Then, ECS i can determine corresponding $\mathbf{x}^\star$ to maximize its utility. Consequently, the MD can also obtain a definitive

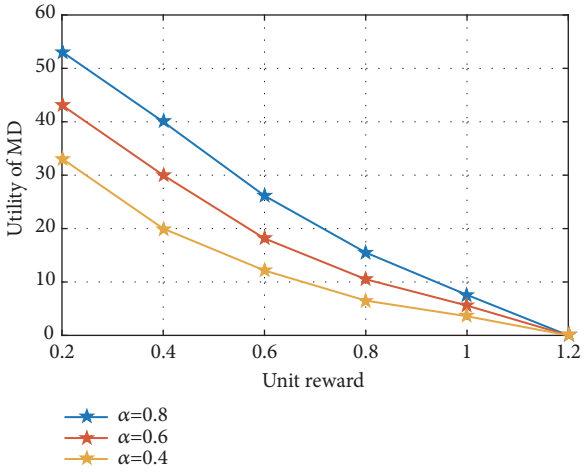

Figure 4: The utility of the MD with different unit reward.

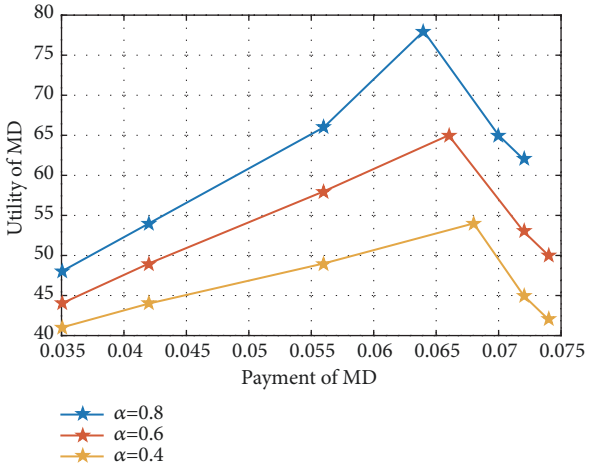

Figure 5: The utility of the MD with different payment.

optimal utility with $\mathbf{p}^{\star}$. Therefore, Algorithm 2 can reach a unique Nash equilibrium. □

6. Performance Evaluation

We use extensive simulations to verify our proposed strategy and algorithms. We set up 3 ECSs as the initial default settings for our simulations. The total computation X_i of ECSs is 100. We consider that the MD and the ECSs are placed over a $1km \times 1km$ region. The ECSs are located at grid points in the region, while the MDs are placed uniformly at random. In the simulations, we set the modeling parameter $\alpha = 1.5$ and the computation complexity parameter $\beta = 1$.

6.1. The Utility of the Mobile Device. In this part, we analyze the changes in the utility of MD as a function of different conditions. First we analyze the relationship between the utility of a MD and the revenue of the ECSs. Figure 4 shows the utility of the MD by varying the value of revenue of the ECSs from their own users. As shown in the figure, regardless of the parameter α being 0.4, 0.6, or 0.8, with the increase of the unit revenue, the ECSs prefer providing the computation to their own users. As a result, the amount of computation provided to computing offloading is reduced. Therefore, the utility of the MD decreases.

Then we analyze how the payment of a MD affects the utility of the MD. Figure 5 shows the correlation between the payment form the MD and the utility of the MD under different value of parameter α. With the increase of the payment from the MD, the utility of it increases at first. But after reaching a peak, the utility of the MD has dropped. This is because as the payment increases, the ECS provides more computation. However, the payment continues to increase, resulting in an increase in the cost of MD, so the utility is reduced.

6.2. The Utility of the Edge Servers. Figure 6 shows the average utility of the ECSs by varying the unit revenue. With the increase of the unit revenue, the utility of the ECSs increases.

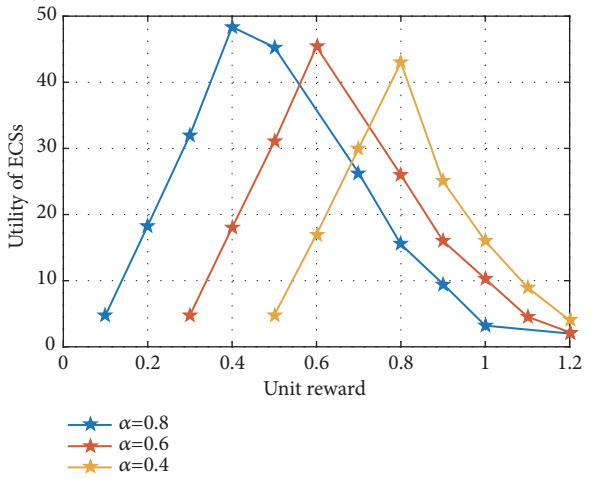

Figure 6: The utility of the ECSs with different unit reward.

The high utility of ECSs is attributed to the fact that ECSs are willing to provide computation to their own users, leading to efficient computation offloading. Interestingly, the ECSs achieve less utility when the revenue is large enough. As we know, ECSs are profit driven to provide computation for their users. However, large unit revenue is a double-edged sword for ECSs. Although edge servers are more willing to serve their users, it may result in fewer computing offloading.

Figure 7 shows the trend of ESCs' utility as a function of payment for a MD. We set $\alpha = 0.4$, $\alpha = 0.6$, and $\alpha = 0.8$, respectively. As the figure shows, the utility of ECSs begins to increase as the payment of the MD increases, and the utility of ECSs begins to decrease after reaching a peak. At the beginning, due to the increase in the payment, the utility of ECSs increases, but this will increase the cost of the MD, so it will not continue to increase.

6.3. Offloading Computations. In order to evaluate how the unit revenue impacts computation for offloading, we set three different unit revenues for simulation. We set w =

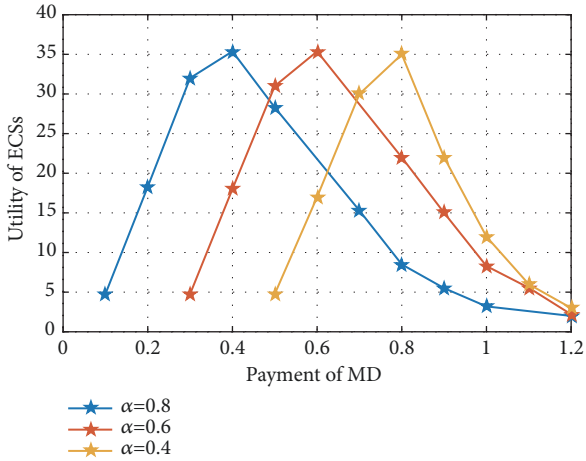

Figure 7: The utility of the ECSs with different revenue.

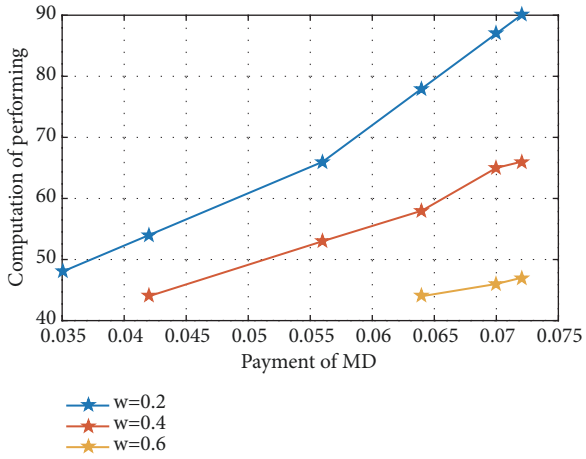

Figure 8: Payment of MD.

0.2, $w = 0.4$, and $w = 0.6$, respectively. As can be seen from Figure 8, the trends of different w are similar, and their computation for offloading increases as the payment increases. In other words, in the case of the above three unit revenues, the computation increases as the payment from the MD increases. Note that when the unit revenue is greater than the MD's given payment, the ECSs will not perform any offloading.

Figure 9 shows the relationship between the computation of offloading and the utility of MD. It also shows the relationship between the computation and the utility of ECSs. Through the simulation results, we see that, within a certain range, the greater the computation of offloading, the higher the utility of the MD. At the same time, the utility of ECSs increases. It is well understood that, due to the strategy, when the computation of offloading increases, the utility of the MD will increase accordingly. Of course ECSs will also benefit. We can say with certainty that, within a certain range bound, the more computation performed, the more beneficial to both sides.

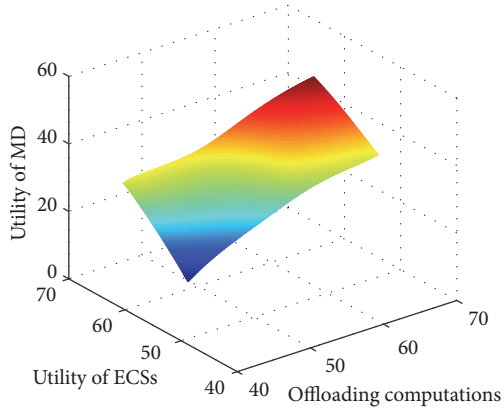

Figure 9: Offloading quantity.

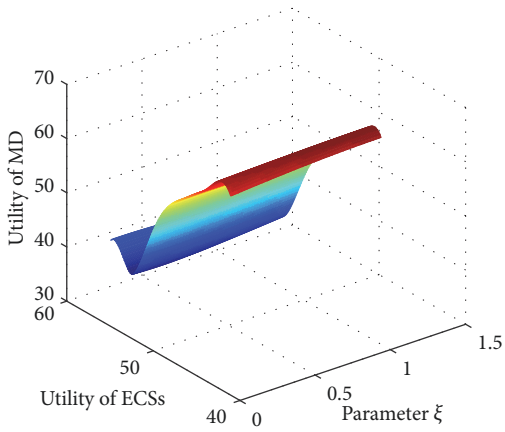

Figure 10: The impact of computational accuracy.

6.4. The Impact of the Utility and Response Time. We are interested in the impact of ξ that represents the value of accuracy for the proposed strategy. So we set a series of ξ values in the range of 0.2 to 1.5. Figure 10 shows the simulation results. As the value of ξ increases, the accuracy of our strategy increases, and the utility of MD and the profit of ECSs increase. It will be more beneficial to both the MD and the ECSs to calculate the benefits of offloading.

Figure 11 shows the response time of F-SGA and C-SGA using the same conditions during the computing offloading. As can be seen from the figure, as the parameter ξ increases, the response time of C-SGA decreases. However, the response time of the F-SGA remains the same and has been at a low level. This is because as the parameter ξ increases, the C-SGA needs to make more rounds of judgment and iteration, resulting in an increase in response time. But F-SGA can always make decisions quickly, regardless of the value of the parameter ξ.

7. Conclusion

In this paper, we proposed a game for the computing offloading between a MD and ECSs. We provided a Stackelberg game theoretic analysis and proved the existence of

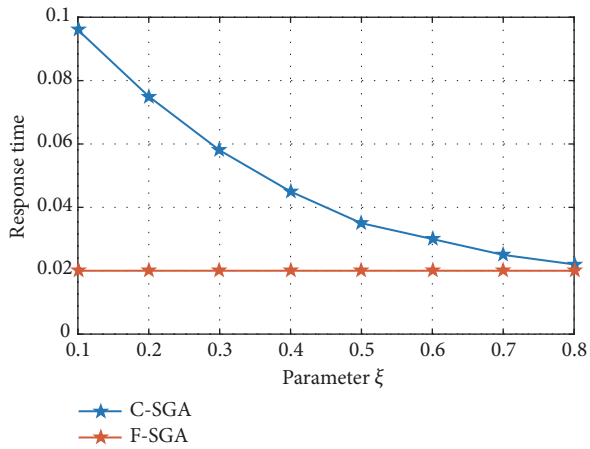

Figure 11: Response time of F-SGA and C-SGA.

the equilibrium in the Stackelberg game. Furthermore, we proposed two algorithms for different scenarios and provided the upper and lower boundary of the payment. Moreover, multiple optimization results were obtained by regulating model parameters. The simulation results showed that the game is effective in improving the utility of both the MD and the ECSs.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants no. 61602155, no. 61871430, no. U1604155, and no. 61370221 and in part by Henan Science and Technology Innovation Project under Grant no. 174100510010 and in part by the industry university research project of Henan Province under Grant No. 172107000005 and in part by the basic research projects in the University of Henan Province under Grant No. 19zx010.

References

[1] W. Zhang, B. Han, and P. Hui, "On the networking challenges of mobile Augmented Reality," in *Proceedings of the 2017 ACM SIGCOMM Workshop on Virtual Reality and Augmented Reality Network, VR/AR Network 2017*, pp. 24–29, USA.

[2] Y. Shen, M. Yang, B. Wei, C. T. Chou, and W. Hu, "Learn to Recognise: Exploring Priors of Sparse Face Recognition on Smartphones," *IEEE Transactions on Mobile Computing*, vol. 16, no. 6, pp. 1705–1717, 2017.

[3] D. Liu, L. Khoukhi, and A. Hafid, "Prediction-Based Mobile Data Offloading in Mobile Cloud Computing," *IEEE Transactions on Wireless Communications*, vol. 17, no. 7, pp. 4660–4673, 2018.

[4] Qingtao Wu, Xulong Zhang, Mingchuan Zhang, Ying Lou, Ruijuan Zheng, and Wangyang Wei, "Reputation Revision Method for Selecting Cloud Services Based on Prior Knowledge and a Market Mechanism," *The Scientific World Journal*, vol. 2014, Article ID 617087, 9 pages, 2014.

[5] Q. Wu, M. Zhang, R. Zheng, and W. Wei, "A QoS-satisfied Prediction Model for Cloud-service Composition Based on Hidden Markov Model," *International Journal of Online Engineering (iJOE)*, vol. 9, no. 3, p. 67, 2013.

[6] M. Olsson, C. Cavdar, P. Frenger, S. Tombaz, D. Sabella, and R. Jantti, "5GrEEn: Towards Green 5G mobile networks," in *Proceedings of the 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 212–216, Lyon, France, October 2013.

[7] N. Cheng, W. Xu, W. Shi et al., "Air-Ground Integrated Mobile Edge Networks: Architecture, Challenges, and Opportunities," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 26–32, 2018.

[8] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[9] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications, IEEE INFOCOM 2016*, USA, April 2016.

[10] F. Liu, P. Shu, H. Jin et al., "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wireless Communications Magazine*, vol. 20, no. 3, pp. 14–21, 2013.

[11] S. Tayade, P. Rost, A. Maeder, and H. D. Schotten, "Device-centric energy optimization for edge cloud offloading," in *Proceedings of the 2017 IEEE Global Communications Conference (GLOBECOM 2017)*, pp. 1–7, Singapore, December 2017.

[12] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: can offloading computation save energy?" *The Computer Journal*, vol. 43, no. 4, Article ID 5445167, pp. 51–56, 2010.

[13] C. Wu, B. Yang, W. Zhu, and Y. Zhang, "Toward High Mobile GPU Performance Through Collaborative Workload Offloading," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 2, pp. 435–449, 2018.

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

[15] Y. Liu, C. Xu, Y. Zhan, Z. Liu, J. Guan, and H. Zhang, "Incentive mechanism for computation offloading using edge computing: a Stackelberg game approach," *Computer Networks*, vol. 129, pp. 399–409, 2017.

[16] S. Josilo and G. Dan, "A game theoretic analysis of selfish mobile computation offloading," in *Proceedings of the IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, GA, USA, May 2017.

[17] S. Yang, D. Kwon, H. Yi, Y. Cho, Y. Kwon, and Y. Paek, "Techniques to minimize state transfer costs for dynamic execution offloading in mobile cloud computing," *IEEE Transactions on Mobile Computing*, vol. 13, no. 11, pp. 2648–2660, 2014.

[18] M. Chen, Y. Hao, M. Qiu, J. Song, D. Wu, and I. Humar, "Mobility-aware caching and computation offloading in 5G ultra-dense cellular networks," *Sensors*, vol. 16, no. 7, pp. 974–987, 2016.

[19] K. Sucipto, D. Chatzopoulos, S. Kostap, and P. Hui, "Keep your nice friends close, but your rich friends closer - Computation

offloading using NFC," in *Proceedings of the 2017 IEEE Conference on Computer Communications, INFOCOM 2017*, USA, May 2017.

[20] M. Satyanarayanan, "Pervasive computing: vision and challenges," *IEEE Personal Communications*, vol. 8, no. 4, pp. 10–17, 2001.

[21] Z. Li, C. Wang, and R. Xu, "Computation offloading to save energy on handheld devices: A partition scheme," in *Proceedings of the 2nd International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, CASES 2001*, pp. 238–246, USA, November 2001.

[22] K. Yang, S. Ou, and H.-H. Chen, "On effective offloading services for resource-constrained mobile devices running heavier mobile internet applications," *IEEE Communications Magazine*, vol. 46, no. 1, pp. 56–63, 2008.

[23] E. Cuervoy, A. Balasubramanian, D.-K. Cho et al., "MAUI: making smartphones last longer with code offload," in *Proceedings of the 8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys '10)*, pp. 49–62, New York, NY, USA, June 2010.

[24] M. Satyanarayanan, V. Bahl, R. Caceres, and N. Davies, "The Case for VM-based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, 2011.

[25] M. Patel, B. Naughton, C. Chan et al., "Mobile-edge computing introductory technical white paper," White Paper, Mobile-edge Computing (MEC) industry initiative, 2014.

[26] J. Ni, K. Zhang, X. Lin, and X. S. Shen, "Securing Fog Computing for Internet of Things Applications: Challenges and Solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 601–628, 2018.

[27] S. N. Shirazi, A. Gouglidis, A. Farshad, and D. Hutchison, "The extended cloud: Review and analysis of mobile edge computing and fog from a security and resilience perspective," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2586–2595, 2017.

[28] Z. Chen, R. Klatzky, D. Siewiorek et al., "An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance," in *Proceedings of the the Second ACM/IEEE Symposium*, pp. 1–14, San Jose, California, October 2017.

[29] F. Wang and X. Zhang, "Dynamic Computation Offloading and Resource Allocation over Mobile Edge Computing Networks with Energy Harvesting Capability," in *Proceedings of the 2018 IEEE International Conference on Communications (ICC 2018)*, pp. 1–6, Kansas City, MO, May 2018.

[30] L. Tang and S. He, "Multi-User Computation Offloading in Mobile Edge Computing: A Behavioral Perspective," *IEEE Network*, vol. 32, no. 1, pp. 48–53, 2018.

[31] K. Guo, M. Yang, Y. Zhang, and Y. Ji, "An Efficient Dynamic Offloading Approach Based on Optimization Technique for Mobile Edge Computing," in *Proceedings of the 2018 6th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, pp. 29–36, Bamberg, March 2018.

[32] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint Offloading and Computing Optimization in Wireless Powered Mobile-Edge Computing Systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 1784–1797, 2018.

[33] W. Hu, Y. Gao, K. Ha et al., "Quantifying the Impact of Edge Computing on Mobile Applications," in *Proceedings of the the 7th ACM SIGOPS Asia-Pacific Workshop*, pp. 1–8, Hong Kong, Hong Kong, August 2016.

[34] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, no. 2, pp. 89–103, 2015.

[35] C. H. Liu, J. Fan, P. Hui, J. Crowcroft, and G. Ding, "QoI-aware energy-efficient participatory crowdsourcing," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3742–3753, 2013.

[36] J. L. Neto, S. Yu, D. F. Macedo, J. M. Nogueira, R. Langar, and S. Secci, "ULOOF: A User Level Online Offloading Framework for Mobile Edge Computing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 11, pp. 2660–2674, 2018.

[37] V. Cardellini, V. De Nitto Persone, V. Di Valerio et al., "A game-theoretic approach to computation offloading in mobile cloud computing," *Mathematical Programming*, vol. 157, no. 2, Ser. B, pp. 421–449, 2016.

[38] D. Nowak, T. Mahn, H. Al-Shatri, A. Schwartz, and A. Klein, "A Generalized Nash Game for Mobile Edge Computation Offloading," in *Proceedings of the 2018 6th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, pp. 95–102, Bamberg, March 2018.

[39] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 5, pp. 1045–1057, 2014.

[40] Y. Wang, A. Nakao, and A. V. Vasilakos, "Heterogeneity playing key role: Modeling and analyzing the dynamics of incentive mechanisms in autonomous networks," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 7, no. 3, 2012.

[41] J. Xu, L. Chen, K. Liu, and C. Shen, "Designing Security-Aware Incentives for Computation Offloading via Device-to-Device Communication," *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 6053–6066, 2018.

[42] J. Guo, Z. Song, Y. Cui, Z. Liu, and Y. Ji, "Energy-Efficient Resource Allocation for Multi-User Mobile Edge Computing," in *Proceedings of the GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–7, Singapore, December 2017.