Game-Based Task Offloading of Multiple Mobile Devices with QoS in Mobile Edge Computing Systems of Limited Computation Capacity

JUNYAN HU, KENLI LI, and CHUBO LIU, Hunan University, China
KEQIN LI, Hunan University and State University of New York

Mobile edge computing (MEC) is becoming a promising paradigm of providing computing servers, like cloud computing, to Edge node. Compared to cloud servers, MECs are deployed closer to mobile devices (MDs) and can provide high quality-of-service (QoS; including high bandwidth, low latency, etc) for MDs with computation-intensive and delay-sensitive tasks. Faced with many MDs with high QoS requirements, MEC with limited computation capacity should consider how to allocate the computing resources to MDs to maximize the number of served MDs. Besides, for each MD, he/she wants to minimize the energy consumption within an acceptance delay range. To solve these issues, we propose a Game-based Computation Offloading (GCO) algorithm including a task offloading profile of MEC and the transmission power controlling of each MD. Specifically, we propose a Greedy-Pruning algorithm to determine the MDs that can offload the tasks to MEC. Meanwhile, each MD competes the computing resources by using his/her transmission power-controlling strategy. We illustrate the problem of task offloading for multi-MD as a non-cooperative game model, in which the information of each player (MDs) is incomplete for others and each player wishes to maximize his/her own benefit. We prove the existence of the Nash equilibrium solution of our proposed game model. Then, it is proved that the transmission power solution sequence obtained from GCO algorithm converges to the Nash equilibrium solution. Extensive simulated experiments are shown and the comparison experiments with the state-of-the-art and benchmark solutions validate and show the feasibility of the proposed method.

CCS Concepts: • **Networks → Cloud computing**; • **Mathematics of computing → Mathematical analysis**; • **Theory of computation → Algorithmic game theory and mechanism design**;

Additional Key Words and Phrases: Mobile edge computing, Nash equilibrium, non-cooperative game theory, task offloading, power controlling

ACM Reference format:
Junyan Hu, Kenli Li, Chubo Liu, and Keqin Li. 2020. Game-Based Task Offloading of Multiple Mobile Devices with QoS in Mobile Edge Computing Systems of Limited Computation Capacity. *ACM Trans. Embed. Comput. Syst.* 19, 4, Article 29 (July 2020), 21 pages.
https://doi.org/10.1145/3398038

This research was partially funded by the National Key R&D Program of China (Grant No. 2018YFB1003401), the National Outstanding Youth Science Program of National Natural Science Foundation of China (Grant No. 61625202), the Program of National Natural Science Foundation of China (Grant No. 61876061 and Grant No. 61876191).
Authors' addresses: J. Hu, K. Li (corresponding author), and C. Liu, Hunan University, Lushan South Road 2, Changsha, China; emails: {junyanhu, lkl, liuchubo}@hnu.edu.cn; K. Li (corresponding author), Hunan University, Lushan South Road 2, Changsha, China; State University of New York, New Paltz, New York 12561, New York; email: lik@newpaltz.edu.

1539-9087/2020/07-ART29 $15.00
https://doi.org/10.1145/3398038

1 INTRODUCTION

1.1 Motivation

Presently, Mobile Devices (MDs) are an indispensable part in our daily life [1]. With the popularity of smart MDs and new electronic products, many new computation-intensive and delay-sensitive applications (e.g., face and voice recognition, virtual and augmented reality, human-computer interaction) require high demands on quality of service (QoS; bandwidth, energy consumption, latency, etc.) [24]. MDs have their own computation capacity (local computing), but sometimes the limited resources, e.g., battery and computation capacity, cannot meet their own high-quality requirements. In addition, MDs can also send requirements to cloud computing servers, but the long distances between MDs and cloud computing center will cause the high latency, which can decrease the QoS and the user experience. In order to meet the ever-increasing MDs with high QoS requirements, the concept of edge computing has been proposed, and it will gradually become a trend in service computing.

Mobile edge computing (MEC) provides high-bandwidth, high-computing resources for nearby MDs to meet the high QoS demands for computation-intensive and latency-sensitive applications via edge network [5, 8, 23, 34]. With the popularization of 5G technology, the connection between edge computing and 5G architecture can better realize the application of vehicle IoT, intelligent transportation, artificial intelligence, and so on, and can provide power support for the realization and popularization of driverless vehicles. Task computation requirements are as shown in Figure 1. MEC can be viewed as a small cloud with limited resources (processing speed, CPU cycle). Facing with resource requests from numerous devices, MEC should formulate a resource allocation mechanism to maximize the number of served MDs with QoS requirements. For each MD, he/she desires to have a minimum energy consumption with an expected delay value. Here, we consider that the transmission rate and the received computation resource of each MD are affected by others. That is to say, if too many MDs request computing resources from MEC, some MDs will not receive the computing resources or get relatively low transmission rates to result in high latency. Therefore, each MD can compete for the computing resources of MEC by dynamically adjusting the transmit power and the transmission rate to meet his/her own requirement.

As shown in Figure 1, we consider multiple mobile devices and multiple MEC systems. MDs can offload the computation-intensive and time-sensitive tasks to MEC servers through wireless channels, which can provide the computation resources to MDs and help them to improve the computation performance while satisfying the time constraints. With finite computational resources, MECs can not meet the requirements of all the MDs with QoS; therefore, MEC servers serve as many tasks as possible offloaded from a number of MDs within the coverage of the base station. Meanwhile, each MD minimizes his energy consumption under the conditions of competing for the mutual computation resources and satisfying the deadline constraint. In our article, we study the task offloading mechanism for MECs and multiple MDs with deadlines systems with the method of non-cooperative games. We propose agame-based computation offloading (GCO) algorithm to collectively optimize the task offloading scheduling and the transmission power controlling of the system from the perspective of both MECs and each MD, alternately optimizing the number of served MDs with deadlines and optimizing the energy consumption for MDs that their tasks are executed on MECs.

1.2 Our Contributions

According to the MEC's resource allocation rules, each MD selects an appropriate transmit power in order to compete for computing resources of MECs so that the energy consumption is as low as possible while meeting the deadline requirement. Since the transmission rate and the competing

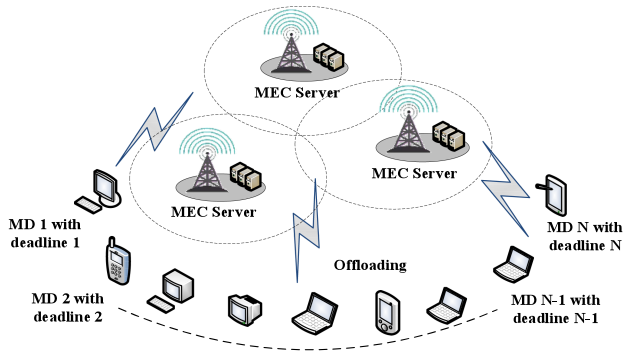


Fig. 1. A multi-device edge computing system.

resources between MDs are mutually influential, we establish a game model for task offloading of multiple MDs. Every MD is a player, and every player will choose an optimal transit power in the transmission power-controlling strategy set to maximize his own benefit. Eventually, the game system will reach a point of equilibrium, i.e., Nash equilibrium. Our main contributions are listed as follows:

- —With the perspective of a non-cooperative game, a mechanism of task offloading for the system of multiple MECs and multiple MDs with delay deadline constraints is constructed to optimize the benefits of both MECs and each MD.
- —Given the allocation of transmission power, a Greedy-Pruning algorithm is proposed to determine the number of tasks executed on MECs. Each MD controls his/her transmission power to complete the resource of MECs or minimize the energy consumption. We formulate a game model for illustrating the system and prove the existence of a Nash equilibrium solution.
- —A GCO algorithm is proposed to compute the Nash equilibrium solution, and the convergence of the solution sequence obtained by the GCO algorithm is analyzed.
- —Simulated experiments evaluate the proposed algorithm in terms of effectiveness evaluation. Besides, the comparison experiments with the method of random transmit power solution validate and show the feasibility of the proposed method.

The remainder of the article is organized as follows. In Section 2, we introduce the related work. Section 3 describes the system model and presents the problem that needs to be solved. In Section 4, we consider the problem as a non-cooperative game and propose Algorithm GCO to compute the Nash equilibrium solution. In Section 5, extensive experiments and the comparison experiments results indicate the feasibility of our algorithms. We conclude the works of this article in Section 6.

2 RELATED WORK

We present a review of the related work centered around the task offloading of MEC, controlling the transmission power, and a non-cooperative game.

Task offloading for user requirements in MEC has been studied by many scholars and most of the studies are analyzed from computational offloading, latency, storage, and energy efficiency [19, 37, 38]. Some scholars considered from the view of optimizing the energy consumption of users [3, 33, 35]. In Ref. [3], Chen et al. computed the green energy harvesting for MEC to solve the multi-task offloading problem by using the Lyaponuv Optimization method. In Ref. [35], Yang et al. proposed an artificial fish swarm algorithm (AFSA) algorithm to minimize the overall energy consumption

under the constraints of MEC's limited computation capability and each MD's deadline requirement. In Ref. [33], Wang et al. proposed a resource allocation mechanism to minimize the total energy consumption of MEC in the system that multiple users can offload all or a portion of tasks to MEC and each one has deadline constraint. Besides, some works and models were considered from guaranteeing the deadline or minimizing average delay [2, 6, 12, 18, 29]. In Ref. [2], Chen et al. transformed the task offloading issue as task placement and resource allocation sub-issues and proposed an efficient offloading mechanism to minimize the delay while decreasing user's energy consumption based on the software defined ultra-dense network. Fan et al. formulated an application-aware workload scheduling system for MEC-based Internet of Things to minimize the latency of each application request in Ref. [6]. In Ref. [18], Liu et al. viewed the user's task offloading process as the Markov decision process and proposed a one-dimensional search approach to solve the problem of power-constrained delay minimization. Besides, there are many works that were considered from the tradeoff overhead on energy consumption and delay [4, 13, 16, 36]. In Ref. [36], Zhang et al. combined with interior penalty function of difference programming (IPDC) to propose an iterative search method to solve the mixed integer nonlinear (MINL) problem of resource allocation and computation offloading, such that the entire system had a lower tradeoff overhead. In Ref. [4], Chen et al. proved the existence of the solution of the established multi-user non-cooperative game through the potential function game and the results obtained by his proposed algorithm can reduce the total cost. The difference between our work and others is that we first consider a more general multi-MD multi-MEC system, and all MDs dynamically adjust power-controlling strategies to compete for the limited computation capacity with the method of a non-cooperative game. The objective functions of optimizing average energy consumption, average latency, tradeoff overhead on energy consumption, or delay in other papers are globally optimized. However, for MDs with high requirements (such as deadline or energy consumption), the average latency or average energy consumption can no longer meet their requirements. Thus, different from all of them, we consider our problem from a more practical perspective to show the non-cooperative game between MDs for computing resources.

Many works optimized the transmission to achieve the offloading balance in the MEC by controlling the transmission power [15, 20, 27, 30]. In Ref. [15], Li et al. proposed an adaptive transmission mechanism with a software defined network and edge computing for Internet of Things. In Ref. [27], Rodrigues et al. proposed an offloading-balance mechanism for cloudlets to minimize the overhead by using Virtual Machine Migration (VMM) and Transmission Power Control (TPC). In Ref. [20], Mao et al. alternately optimized the task offloading scheduling and transmission power allocation for MEC and the multi-task system, and then obtained an optional solution. In Ref. [30], Tao et al. applied KKT conditions to determine the optimal transmission rate, such that the entire energy consumption is minimized. Heuristically, we consider the mutual influence of the transmission rates of multiple MDs to achieve a balanced state for each MD.

Game theory plays an increasingly important method in MEC [4, 7, 9, 12–14, 17, 25, 26]. Since there is channel competition and resource competition of MEC between users, each user is independent of each other and affects each other. Therefore, it is impossible for each user to achieve his/her optimality. Multiple users reach an equilibrium state in the process of mutual constraint, i.e., a win-win situation. In Ref. [12], Li optimized the average response time of all tasks computation offloading by constructing a non-cooperative game mechanism for a multi-MEC and multi-UE environment, together with the M/G/1 queuing model. Then, in Ref. [13], Li expanded to comprehensively study the computation offloading optimization problems from the perspective of the average delay with the power consumption constraint, the average power consumption with the constraint of delay, the cost-performance ratio, and the power-time tradeof. In Ref. [4], Chen et al. considered the multi-task offloading mechanism for MEC in the multi-channel

Table 1. Comparisons between GCO and the State-of-the-Art Schemes Involving Service Delay and Power Control

Schemes	Environment Condition(s)	Delay Deadline	Power Control	Objective(s)	Main Method(s)
Kiani [11]	1. Multi-MD and One MEC 2. Multiple channels	No	Yes	Energy consumption	Heuristic algorithm
Liu [18]	1. One MD and one MEC 2. MD generates jobs with varying time slot	No	No	Average delay with power-constrained	Markov chain model
Zhang [36]	1.Multi-MD and multi-MEC 2. Every job can be performed on the MEC or local	Yes	Yes	Overhead of the weighted energy consumption and latency	Iterative search
Chen [4]	1.Multi-MD and one MEC 2. Every job can be transmitted from different channels to MEC	Yes	No	Number of jobs whose constraints (weighted sum of delay and energy) are satisfied	Non-cooperative game
Mao [20]	1. One MD and one MEC 2. MD associates with multiple independent tasks	No	Yes	Overhead of the weighted energy consumption and latency	Convex optimization
Tao [30]	1. Multi-MD and one MEC 2. Every job can be performed proportionally in the MEC and local	Yes	No	Energy consumption	Optimization (KKT conditions)
Our GCO	1. **Multi-MD and one MEC** 2. **Each MD has one computation-intensive and time-sensitive**	**Yes**	**Yes**	**1. Number of tasks with deadlines** **2. Energy consumption for each MD executed on MEC**	**Non-cooperative game**

wireless system from the perspective of multi-user game model. In Ref. [9], Hu et al. minimized the MEC's total transmit energy with the computational tasks constraints by MDs' cooperative communications. Heuristically, our work introduces an adaptive transmission power mechanism in the competing process for limited-computation resources provided by MECs. We formulate a non-cooperative game-based mechanism for MDs' power-controlling strategies based on MEC's offloading decision-making mechanism.

Table 1 shows the comparisons between our proposed GCO algorithm and the state-of-the-art schemes involving service delay and power control. These are all mechanisms from the perspective of MEC, e.g., number of tasks whose bounds are satisfied [4], or MDs, e.g., minimum average delay [18], minimum average energy consumption [11, 30], tradeoff energy-latency [20, 36]. Different from the others mentioned, our work considers not only from the view of serving the maximum number of MDs with deadline constraints to formulate an MEC's offloading decision-making mechanism, but also from the perspective of each MD's minimum energy consumption to establish a game system. In this system, each MD is a player. Each player has his strategy set and adjusts his optimal strategy based on the actions of others. The system reaches an equilibrium state in the process of mutual interaction between the players.

3 SYSTEM MODEL

We represent the M MECs as the set of $\mathcal{M} = \{1, 2, \ldots, M\}$ and the set of computational capacity of MECs $\mathcal{S}$ is denoted as $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$, where each C_m ($m \in \{1, 2, \ldots, M\}$) is limited. We denote $\mathcal{N} = \{1, 2, \ldots, N\}$ as the set of N MDs and each MD has a computation-intensive and time-sensitive task to be completed. Let τ_n be the task of n, and the requirement of MD τ_n can be denoted as a tuple (c_n, d_n, T_n), where c_n denotes the total number of required CPU cycles, d_n denotes the size

of the input task data, and T_n denotes the expected time required to complete task τ_n. The task can be computed either locally on the mobile device or remotely executed on MEC via computation offloading. Therefore, we denote the decision profile $\mathcal{X} = \{x_{n,m}\}$ $(n \in \mathcal{N}, m \in \mathcal{M})$ as the set of indicator function for N MDs, where $x_{n,m} \in \{0, 1\}$. If the task of MD n is computed on MEC $\mathcal{M}$, $x_{n,m} = 1$; otherwise, $x_{n,m} = 0$. Besides, we denote $\mathcal{J} = \{J_1, J_2, \ldots, J_M\}$ as the set of MD groups, where J_m is the MD group of offloading their tasks to MEC m and $J_m = \{n | x_{n,m} = 1\}$. If an MD prepares to offload his task to MEC $\mathcal{S}$, the energy and time consumption of communication and computation are considered.

3.1 Communication Model

Since non-orthogonal multiple access (NOMA) [10, 11] can effectively meet the requirements of 5G wireless networks to provide massive connectivity of mobile devices and meet the demand for low latency, we consider the communication process for wireless access with the NOMA technique in the MEC system. The NOMA technique allows multiple MDs to share the same resources in one channel condition. Power-domain NOMA utilizes superposition coding (SC) at the transmitter and successive interference cancellation (SIC) at the receiver to detect the desired signals. If MD n is selected to offload his task τ_n to edge computing m, the input data d_n should be transmitted to MEC servers of m. For every MEC, the channel bandwidth is B. $G_{n,m}$ is the channel gain in the process of transmission from MD n to MEC m. In the channel of MEC m, we define the order set $\mathcal{O} = \{O_1, O_2, \ldots, O_M\}$, where $O_m = \{1, 2, \ldots, |J_m|\}$ is the set of orders in set J_m. Here, $|J_m|$ is the number of J_m. $G_{n,m}$ is the channel gain in the process of transmission, and it is related to the environment and the distance between MD n and MEC m. η_0 is the background noise power. We represent the signal noise ratio of MD n in the wireless channel of MEC m as $S_{n,m}$. Let mapping function be $a(i, m) \rightarrow (n, m) : \mathcal{O} \otimes \mathcal{M} \rightarrow \mathcal{N} \otimes \mathcal{M}$, which represents that the ith MD in set J_m is MD n. Besides, for each MEC m, the channel gains satisfy the condition $0 < S_{a(1,m)} \leq S_{a(2,m)} \leq \cdots S_{a(|J_m|,m)}$, which indicates that mobile device $a(i, m)$ holds the ith weakest instantaneous channel. Besides, we denote $\beta^i_{n,m}$ as the indicator variable to indicate the ith order in set J_m to MD n. Therefore, the communication rate of MD n $(n \in \mathcal{N})$ via the wireless channel can be denoted as

$$r_n = \sum_{m \in \mathcal{M}} x_{n,m} \sum_{i \in O_m} \beta^i_{n,m} B \log_2 \left(1 + \frac{p_n G_{n,m}}{\eta_0 + \sum_{k \in \mathcal{N} \setminus \{n\}} \sum_{l=i+1}^{|J_m|} \beta^l_{k,m} p_k G_{k,m}} \right). \tag{1}$$

$\mathcal{P} = \{p_1, p_2, \ldots, p_N\}$ is the transmission power profile of all MDs, and each p_n can be chosen from the internal $[\underline{p_n}, \overline{p}_n]$, in which $\underline{p_n}$ and $\overline{p}_n$ are the minimum and maximum powers that MD n can accept, respectively. η_0 is the background noise power. In Equation (1), we denote $I_{n,m} = \sum_{k \in \mathcal{N} \setminus \{n\}} \sum_{l=i+1}^{|J_m|} \beta^l_{k,m} p_k G_{k,m}$ as the sum of interference from other MDs in set J_m.

Here, we focus on exploring the computation offloading and power control problems, which enable MECs $\mathcal{S}$ to serve as many MDs as possible under limited resource conditions and minimize the energy consumption for the MDs in $\mathcal{J}$. Note that the transmission rate of MD n in J_m is not only related to his own transmit power but also related on the transmit powers of other MDs in J_m, i.e., the transmission rate of each MD in J_m is mutually influential. Therefore, we consider the distributed power control of the interference-aware multi-device MEC system from the perspective of the non-cooperative game theory.

3.2 Computation Model

If task τ_n of mobile device n is offloaded to MEC m to execute and the order in J_m is i, i.e., $x_{n,m} = 1$, $\beta^i_{n.m} = 1$, the completion time will contain communication time and computation time. We define

the completion time as

$$t_{n,off} = \frac{d_n}{r_n} + \frac{c_n}{f_n} = \frac{d_n}{B\log_2\left(1 + \frac{p_n G_{n,m}}{\eta_0 + I_{n,m}}\right)} + \frac{c_n}{f_n}, \tag{2}$$

where f_n is the computation capability (i.e., CPU cycles per second) assigned to MD n by the MEC m. Therefore, the energy consumption can be denoted as

$$\begin{aligned} E_{n,off} &= \frac{p_n d_n}{r_n}, \\ &= \frac{p_n d_n}{B\log_2\left(1 + \frac{p_n G_{n,m}}{\eta_0 + I_{n,m}}\right)}. \end{aligned} \tag{3}$$

We extend function $E_{n,off}$ to the interval $p_n \geq 0$. The function can be written as

$$E_{n,off} = \begin{cases} \frac{p_n d_n}{B\log_2\left(1 + \frac{p_n G_{n,m}}{\eta_0 + I_{n,m}}\right)} & \beta^i_{n,m} = 1, \\ M & \beta^i_{n,m} = 0, \end{cases} \tag{4}$$

where M is assumed to be an infinite value.

We then introduce the local computation model. The completion time of MD n is defined as

$$t_{n,loc} = \frac{c_n}{f_n^{loc}}, \tag{5}$$

where f_n^{loc} is the local computation capability of MD n. Let l_n be the local power consumption per CPU cycle for MD n. If task τ_n is executed locally, the energy consumption can be denoted as

$$E_{n,loc}(X, \mathcal{P}) = l_n c_n. \tag{6}$$

We extend function $E_{n,loc}(X, \mathcal{P})$ to the interval $p_n \geq 0$. The function can be written as

$$E_{n,loc}(X, \mathcal{P}) = \begin{cases} M & \sum_{m \in \mathcal{M}} x_{n,m} = 1, \\ l_n c_n & x_n = 0. \end{cases} \tag{7}$$

According to Equations (5) and (6), if task τ_n is executed locally, the energy consumption is a fixed value. Generally speaking, each MD possesses relatively few local computation resources (f_n^{loc} is small), and $t_{n,loc}$ is not likely to meet his expected deadline T_n.

- For MDs whose local execution time $t_{n,loc}$ can not satisfy $t_{n,loc} \leq T_n$, they are more willing to choose MEC to meet their expected deadlines.
- If $t_{n,loc} \leq T_n$ and $E_{n,off} < E_{n,loc}$, MD n may also choose MEC to minimize the energy consumption.

Therefore, if task τ_n of MD is offloaded on MEC and one of the above cases is satisfied, we call MD n a beneficial MD.

3.3 MEC's Resource Allocation Strategy

Since we consider that the computation capacity of each MEC is limited, the objective function for all MECs is to provide the service computing for MDs as much as possible when the computation capacity can not meet the demands of all MDs. We denote $|J|$ as the number of elements in set J. Thus, the objective function can be represented as maximizing $\sum_{m \in \mathcal{M}} |J_m|$. For each MEC m, the QoS requirements of each MD in J_m need to be met, that is, $x_{n,m} t_n \leq T_n$ ($x_{n,m} = 1$). With the limit of computation capacity of each MEC, the sum of computation capacity assigned to MD n

in J_m can not be greater than C_m, that is, $\sum_{n\in J_m} x_{n,m} f_n \le C_m$. We propose a distributed resource allocation problem with constraints and model it as follows

$$\begin{aligned} &\max_{\mathcal{X}} \quad \sum_{m\in\mathcal{M}} |J_m|, \\ &\text{s.t.} \quad x_{n,m} t_n \le T_n, n \in J_m, \\ &\quad\quad \textstyle\sum_{n\in J_m} x_{n,m} f_n \le C_m, \end{aligned} \tag{8}$$

Theorem 1. *The problem* $\max_{\mathcal{X}} \sum_{m\in\mathcal{M}} |J_m|$ *that maximizes the number of tasks with QoS executed on MECs is NP-hard.*

Proof. Given the power profile $\mathcal{P}$, a decision profile $\mathcal{X}$ is available if the condition $t_n \le T_n$ for each MD n in set J_m ($x_{n,m} = 1, n \in J$ and $x_{n,m} = 0, n \notin J_m$) is satisfied. So, given the $\mathcal{P}$, the available $\mathcal{X}$, and based on $t_n \le T_n$, the computation capability $f_n(J_m)$ ($n \in J_m$) assigned to MD n should satisfy the constraint

$$f_n(J_m) \ge \frac{c_n}{T_n - \frac{d_n}{\gamma_n}} = f'_n(J_m). \tag{9}$$

Let $f'_n(J_m)$ be the critical point of computation capability that MD n needs. The sum of critical points of all MDs in J_m should satisfy the constraint of $\sum_{n\in J_m} f'_n \le C_m$.

Therefore, the problem $\max_{\mathcal{X}} \sum_{m\in\mathcal{M}} |J_m|$ can be viewed as the maximum cardinality bin packing problem, which is NP-hard. Thus, the problem $\max_{\mathcal{X}} \sum_{m\in\mathcal{M}} |J_m|$ that serves the maximal number of MDs under constraints is also NP-hard. The proof has been completed. □

Since Theorem 1 proves that the problem is NP-hard, we intend to propose a heuristic algorithm to solve our problem. Each addition or deletion of a mobile device will affect the transmission rate of other devices. The best case is that the computation capacity of MECs can meet the requirements of all mobile devices. Because the considered computation capacity of the MECs are limited, we choose to delete some mobile devices until the computation capacity of the MECs can meet all the requirements of the remaining mobile devices. Every time we delete a mobile device, we will show that the deleted mobile device is optimal under the current situation. The greedy pruning algorithm can solve our problem most directly.

The code for mobile device grouping and the computation resource allocation is summarized in Algorithm 1. As shown in this algorithm, we carry out the mobile device grouping, computing resource allocation for each group, and checking the remaining computing resources in three separated phases. In the grouping phase (lines 2–6), we follow a grouping method based on the channel conditions for each mobile device. That is to say, each mobile device chooses the MEC that will make him receive the maximum channel gain. Then, we sort each grouping in ascending order based on computing $S_{n,m}$. In the second phase (lines 8–17), we allocate computing resources to mobile devices in each grouping. For each MEC m, assuming $\sum_{n\in \mathcal{G}roup_m} f'_n(\mathcal{G}roup_m) \le C_m$, then there is $J_m = \mathcal{G}roup_m$. Otherwise, MEC m needs to filter out some MDs to maximize the number of beneficial MDs with QoS. In Algorithm 1, J_m is the set of MDs to be selected, and J_1 is the set of MDs in $\mathcal{G}roup_m$ to be filtered out. In the outer *while* loop of lines 11–17, once $\sum_{k\in J_m} f'_k(J_m) > C_m$, an appropriate MD will be added to J_1 to check whether the condition $\sum_{k\in\widetilde{J}} f'_k(\widetilde{J}) \le C_m$ is satisfied, where $\widetilde{J}$ is the updated J_m. In each round of preparation to remove a MD to J_1, we use $\min \sum_{j\in\widetilde{J}} f'_j(\widetilde{J})$ as the objective function. But removing MD i in J_m that minimizes $\sum_{j\in J_m\setminus\{i\}} f'_j(J_m\setminus\{i\})$ directly does not guarantee that updated J_m is globally optimal. If there is always

$$l = \arg\min_i \sum_{j\in(J_m\cup\{l\})\setminus\{i\}} f'_j((J_m\cup\{l\})\setminus\{i\}), \tag{10}$$

ALGORITHM 1: Greedy-Pruning algorithm

Require: $\mathcal{N}, \mathcal{P}, G, B, C$.
Ensure: $\mathcal{J}, f_{\mathrm{n}}(\mathcal{J})$.

1: Mobile Device Group
2: $\mathcal{G}roup \leftarrow \{\emptyset, \ldots, \emptyset\}$;
3: **for** $n \in \mathcal{N}$ **do**
4: $k \leftarrow \arg\max_{m \in \mathcal{M}} G_{n,m}$;
5: $Group_k \leftarrow Group_k \bigcup \{n\}$;
6: Sort the MDs in $\mathcal{G}roup_m$ such that $0 < S_{a(1,m)} \leq S_{a(2,m)} \leq \cdots S_{a(|J_m|,m)}$;
7: Resource allocation
8: **for** $m \in \mathcal{M}$ **do**
9: $J_m \leftarrow Group_m$, $J_1 \leftarrow \emptyset$, $J_2 \leftarrow \emptyset$;
10: Calculate each $f'_n(J_m)$ $(n \in J_m)$ based on, Equation (9);
11: **while** $(\sum_{k \in J_m} f'_k(J_m) > C_m)$ **do**
12: $J_1 \leftarrow J_1 \bigcup \{\arg\min_{i \in J_m} \sum_{j \in J_m \backslash \{i\}} f'_j(J_m \backslash \{i\})\}$;
13: **while** $(J_1 \neq J_2)$ **do**
14: $J_2 \leftarrow J_1$;
15: **for** $(l \in J_2)$ **do**
16: $J_1 \leftarrow (J_1 \backslash \{l\}) \bigcup \{\arg\min_{i \in \complement_{\mathcal{G}roup_m}^{J_1 \backslash \{l\}}} \sum_{j \in \complement_{\mathcal{G}roup_m}^{J_1 \backslash \{l\}} \backslash \{i\}} f'_j(\complement_{\mathcal{G}roup_m}^{J_1 \backslash \{l\}} \backslash \{l\})\}$;
17: $J_m \leftarrow \mathcal{G}roup_m \backslash J_1$;
18: Check Remaining Computing Resources
19: **for** $m \in \{k | J_k = Group_k\}$ **do**
20: **while** $(\sum_{j \in J_m} f'_j(J_m) < C_m)$ **do**
21: $l \leftarrow \arg\min_{n \in \mathcal{N} \backslash \{\bigcup J\}} \sum_{j \in J_m \bigcup \{n\}} f'_j(J_m \bigcup \{n\})$;
22: **if** $(\sum_{j \in J_m \bigcup \{n\}} f'_j(J_m \bigcup \{n\}) \leq C_m)$ **then**
23: $J_m \leftarrow J_m \bigcup \{l\}$,
24: **return** $\mathcal{J}, f_{\mathrm{n}}(\mathcal{J})$.

for any MD $(l \in \mathcal{G}roup_m \backslash J_m)$, J_m is optimal. The *while* loop of lines 13–16 guarantees that J_m is the optimal in every round of filtering out a MD. In the third phase (lines 19–23), we check if there is any MEC that has redundant computing resources. For an MEC m with $J_k = \mathcal{G}roup_m$, it may have computation capacity $(\sum_{j \in J_m} f'_j(J_m) < C_m)$ to others' mobile devices who are not assigned to computing resources. In order to add more mobile devices, we choose the mobile device n that minimizes $\sum_{j \in J_m \bigcup \{n\}} f'_j(J_m \bigcup \{n\})$ every time, where $n \in \mathcal{N} \backslash \{\bigcup J\}$. MEC m can keep adding the mobile devices until $\sum_{j \in J_m} f'_j(J_m) \geq C_m$.

3.4 Power Control Strategy of Mobile Device

If the task of an MD is executed locally, the energy supply of MDs may not meet his energy consumption. Besides, if the task is uploaded to the cloud computing for execution, there will be a high delay due to the distance. Compared with the above two, MEC makes up for the shortcomings, which can reduce MDs' energy consumption and enable MDs to get faster feedback. Therefore, MDs are more willing to upload tasks to edge computing to execute. From the MDs' perspective, MDs expect lower latency and low power consumption simultaneously. However, it is known from

the literature that MDs cannot guarantee the lowest energy consumption with the lowest delay. Here, we consider that how each MD minimizes his own energy consumption within the expected delay range based on the computation resource allocation in Algorithm 1.

From Equation (2), we can know that $t_{n,off}$ decreases as p_n increases. Given $\mathcal{F}$ and the deadline constraint T_n, $t_{n,off} \leq T_n$ can be further written as follows

$$p_n \geq \left(2^{\frac{d_n}{(T_n - \frac{c_n}{f_n})B}} - 1\right)\left(\frac{\eta_0 + I_{n,m}}{G_{n,m}}\right) = p'_n. \tag{11}$$

Let p'_n be the critical power of MD n. If $p'_n > \overline{p}_n$, MD n will not want to transmit his task τ_n to MEC m. We consider the case $p'_n \leq \overline{p}_n$ and analyze the energy consumption of MD n in the internal $[\max\{p'_n, \underline{p_n}\}, \overline{p}_n]$.

In each round, for MD n in $\mathcal{G}roup_m$ who doesn't belong to J_m, if he wants to enhance his own competitiveness, he can increase p_n. This leads to two outcomes—removing one of the other MDs in J_m or adding to the set J_m directly.

Removing one of the other MDs in J_m: Increasing p_n to p_n^1 and satisfying the conditions

$$\begin{aligned} &\arg\min_k \sum_{j \in J_3} f'_j(J_3 \backslash \{k\}) \neq n, (J_3 = J_m \cup \{n\}), \\ &\min_k \sum_{j \in J_3} f'_j(J_3 \backslash \{k\}) \leq C_m. \end{aligned} \tag{12}$$

Adding to the set J_m directly: Increasing p_n to p_n^2 and satisfying the condition

$$\sum_{j \in J_m \cup \{n\}} f'_j(J_m \cup \{n\}) = C_m. \tag{13}$$

Considering the energy consumption and $p_n \leq \overline{p}_n$, we define $\widetilde{p}_n = \min\{p_n^1, p_n^2, \overline{p}_n\}$, where $\widetilde{p}_n$ is the updated p_n in next round. Let $\widetilde{\mathcal{P}} = (\widetilde{p}_1, \widetilde{p}_2, \ldots, \widetilde{p}_N)$ be the updated power profile of all MDs. We propose a binary search algorithm (Calculate_$\widetilde{\mathcal{P}}(\cdot)$, Algorithm 2) to update p_n.

4 GAME FORMULATION AND ANALYSES

4.1 Game Formulation

In this article, we consider the distributed computation offloading decision-making problem and power control problem among the MDs. We also propose a non-cooperative game-based mechanism for MDs' power-controlling decision-making based on the resource allocation mechanism of MEC. The MDs submit their requirements of tasks, and MEC $\mathcal{S}$ maximizes the number of beneficial MDs with QoS by allocating the computation capacity to MDs. The MDs repetitively submit their transmission powers to MECs $\mathcal{S}$, which arrange the computation offloading profile. After a series of interaction iterations, it reaches a steady state. Namely, it reaches a Nash equilibrium.

Let $\mathcal{P}_{-\mathbf{n}} = (p_1, \ldots, p_{n-1}, p_{n+1}, \ldots, p_{|J_m|})$ be the transmission power profile of all MDs in J_m except MD n, where $\mathcal{P}_n$ is the set of power decision-making strategies of MD n, i.e., $p_n \in \mathcal{P}_n$. Given other MDs' transmission power profile $\mathcal{P}_{-\mathbf{n}}$, MD n would like to select a proper decision p_n to compete the computation resource of MEC m and minimize his own energy consumption, under the condition of deadline. The objective function of MD n can be written as follows

$$\min\ E_n = \begin{cases} M & \sum_{m \in \mathcal{M}} x_{n,m} = 0, t_{n,loc} \leq T_n, \\ E_{n,off}(X, \mathcal{P}) & \sum_{m \in \mathcal{M}} x_{n,m} = 1, t_{n,off} \leq T_n, \end{cases} \tag{14}$$

ALGORITHM 2: Calculate_$\widetilde{\mathcal{P}}(\cdot)$

Require: $\mathcal{N}, \mathcal{P}, G, B, C, J, \varepsilon$.
Ensure: $\widetilde{\mathcal{P}}$.

for $m \in \mathcal{M}$ **do**
  for $n \in J_m$ **do**
    $\widetilde{p}_n = p_n$;
  for $n \in \mathcal{G}roup_m \backslash J_m$ **do**
    $l_1 p_n \leftarrow p_n, r_1 p_n \leftarrow \overline{p}_n$;
    $l_2 p_n \leftarrow p_n, r_2 p_n \leftarrow \overline{p}_n$;
    while $(|r_1 p_n - l_1 p_n > \varepsilon|)$ **do**
      $mid_1 \leftarrow \frac{l_1 p_n + r_1 p_n}{2}$;
      if Conditions in Equation (12) are satisfied **then**
        $r_1 p_n \leftarrow mid_1$;
      else
        $l_1 p_n \leftarrow mid_1$;
    $p_n^1 \leftarrow r_1 p_n$;
    while $(|r_2 p_n - l_2 p_n > \varepsilon|)$ **do**
      $mid_2 \leftarrow \frac{l_2 p_n + r_2 p_n}{2}$;
      if Condition in Equation (13) is satisfied **then**
        $r_2 p_n \leftarrow mid_2$;
      else
        $l_2 p_n \leftarrow mid_2$;
    $p_n^2 \leftarrow r_2 p_n$;
    $\widetilde{p}_n = \min\{p_n^1, p_n^2, \overline{p}_n\}$;
return $\widetilde{\mathcal{P}}$.

where M is an infinite constant. The strategy set of MECs $\mathcal{M}$ is $\mathcal{X}$ and the objective function is maximizing the beneficial number of MDs $\sum_{m \in \mathcal{M}} |J_m|$. Then, the multi-device computation offloading game can be represented as G, where $G = \{(\mathcal{P}_n)_{n \in \mathcal{N}}, \mathcal{X}; (E_n)_{n \in \mathcal{N}}, \sum_{m \in \mathcal{M}} |J_m|\}$.

Definition 1 (Nash equilibrium of computation offloading). A Nash equilibrium strategy profile $\mathcal{P}_m^* = \{p_1^*, \ldots, p_{|\mathcal{G}roup_m|}^*\}$, $X_m^* = \{x_{1,m}^*, x_{2,m}^*, \ldots, x_{|\mathcal{G}roup_m|,m}^*\}$ of game $G_m = \{(\mathcal{P}_n)_{n \in \mathcal{G}roup_m}, \mathcal{X}_m; (E_n)_{n \in \mathcal{G}roup_m}, |J_m|\}$ satisfies

$$\mathcal{P}_m^* = \arg\min_{p_{\mathbf{n}} \in \mathcal{P}_n} E_n, p_{\mathbf{n}}^* \in \mathcal{P}_n, \tag{15}$$

$$X_m^* = \arg\max_{X_m \in \mathcal{X}_m} |J_m|, X_m^* \in \mathcal{X}_m, \tag{16}$$

for the MEC m and each MD in J_m.

For all MDs in J_m, $\mathcal{P}_m^* = \{p_1^*, \ldots, p_{|\mathcal{G}roup_m|}^*\}$ is the optimal countermeasure strategy. That is to say, for MD n in J_m and any $p_n \in \mathcal{P}_n$, there is $E_n(p_n, \mathcal{P}_{-\mathbf{n}}^*) \geq E_n(p_n^*, \mathcal{P}_{-\mathbf{n}}^*)$. For MEC m and any $X_m = (x_1, x_2, \ldots, x_{|\mathcal{G}roup_m|})$, $|J_m(X_m^*)| \geq |J_m(X_m)|$.

4.2 Nash Equilibrium Existence Analysis

There are many studies of equilibrium solution existence analysis [21, 22, 28, 31, 32]. In Ref. [22], John used the fixed point theorem to prove the existence of equilibrium points and expanded the

two-person games to n-person games. In Ref. [28], Scutari considered a generic optimization problem with the minimization $f(x)$ form under the constraint of $x \in \mathcal{K}$, where $\mathcal{K}$ is a constraint set. There is an optimal solution $x^* \in \mathcal{K}$ if and only if $(y - x^*)^T \bigtriangledown f(x^*) \geq 0 \ \forall y \in \mathcal{K}$. In Ref. [21], Musku provided the efficient use of system resources in next generation systems that require control of both data transmission rate and power for mobile terminals. Tsiropoulou [31] solved the problem of efficient distributed power control in the uplink of CDMA wireless networks supporting multiple services to find and determine the Nash equilibrium point via convex pricing of users' transmission power. In Ref. [32], Tsiropoulou further considered heterogeneous services with various transmission rates and requirements in the efficient use of wireless system resources. The similarity between our article and these research works on game-theoretic analysis is that we analyze the objective function and variable of each player. Then, by proving that the objective function is a convex function and the interaction between players, it can be proved that the proposed Game exists Nash equilibrium solution. Finally, we prove the convergence of the solution sequence obtained by the proposed algorithm. The following is the existence of a Nash equilibrium about our problem.

Theorem 1. *Given $\mathcal{G}roup_m$, G_m, B, C_m, and $p_n \geq \max\{p'_n, \underline{p_n}\}$, non-cooperative game strategies for $|\mathcal{G}roup_m|$ MDs and MEC m $\mathcal{H} = (\mathcal{G}roup_m, \{\mathcal{P}_n\}_{n \in \mathcal{G}roup_m}, \{E_{n,off}\}; MEC\ m, \mathcal{X}_m, |J_m|)$ have a Nash equilibrium $\langle \mathcal{P}^*_m, X^*_m \rangle$, $(p^*_n \in \mathcal{P}_n, X^*_m \in \mathcal{X}_m)$.*

Proof. Based on Equation (1), we do the derivation of transmission rate of MD n r_n to p_n, and the result is Equation (17):

$$\frac{\partial r_n}{\partial p_n} = \frac{BG_{n,m}}{(\eta_0 + I_{n,m} + p_n G_{n,m}) \ln 2}. \tag{17}$$

According to Equation (3), the energy consumption of MD n E_n takes a derivative with respect to p_n, and the result is Equation (18):

$$\begin{aligned} \frac{\partial E_{n,off}}{\partial p_n} =& \frac{d_n r_n - p_n d_n \frac{\partial r_n}{\partial p_n}}{r_n^2}, \\ =& \frac{d_n B(\log_2 \frac{\eta_0 + I_{n,m} + p_n G_{n,m}}{\eta_0 + I_{n,m}} - \frac{p_n G_{n,m}}{(\eta_0 + I_{n,m} + p_n G_{n,m}) \ln 2})}{r_n^2}. \end{aligned} \tag{18}$$

Let

$$h = \log_2 \frac{\eta_0 + I_{n,m} + p_n G_n}{\eta_0 + I_{n,m}} + \frac{\eta_0 + I_{n,m}}{(\eta_0 + I_{n,m} + p_n G_n) \ln 2} - \frac{1}{\ln 2}. \tag{19}$$

We consider the function $g(x) = \log_2 x + \frac{1}{x \ln 2}$, and its derivative with respect to x is $g'(x) = \frac{1}{x \ln 2} - \frac{1}{x^2 \ln 2} = \frac{x-1}{x^2 \ln 2}$. We can know that when $x > 1$, $g'(x) > 0$, and it shows that function $g(x)$ is monotonically increasing with x $(x > 1)$. Because $\frac{\eta_0 + I_{n,m} + p_n G_{n,m}}{\eta_0 + I_{n,m}} \geq 1$ $(p_n \geq 0)$, we can know that $h(X, \mathcal{P})$ is monotonically increasing with p_n $(p_n \geq 0)$. Therefore, when $p_n = 0$, h has the minimal value, and $\min h = 0$ $(p_n \geq 0)$. Further, when $p_n \in [\max\{p'_n, \underline{p_n}\}, \overline{p}_n]$, $\frac{\partial E_{n,off}}{\partial p_n} > 0$. It shows that $E_{n,off}$ increases as p_n increases in the internal $[\max\{p'_n, \underline{p_n}\}, \overline{p}_n]$.

Then, we consider the second derivative of $E_{n,off}$. Based on Equations (1) and (17), we can obtain that

$$\frac{\partial^2 r_n}{\partial p_n^2} = -\frac{BG_{n,m}^2}{(\eta_0 + I_{n,m} + p_n G_{n,m})^2 \ln 2}. \tag{20}$$

$E_{n,off}$ has taken the second derivative with respect to p_n, and it yields that

$$\begin{aligned}\frac{\partial^2 E_{n,off}}{\partial p_n^2} &= \frac{d_n}{r_n^3}\left(-p_n\frac{\partial^2 r_n}{\partial p_n^2}r_n - 2r_n\frac{\partial r_n}{\partial p_n} + 2p_n\left(\frac{\partial r_n}{\partial p_n}\right)^2\right),\\ &= \frac{d_n B^2 G_{n,m}}{(\eta_0 + I_{n,m})\mu r_n^3 \ln 2}\left[\left(-1-\frac{1}{\mu}\right)\log_2\mu + \frac{2}{\ln 2}\left(1-\frac{1}{\mu}\right)\right],\end{aligned} \tag{21}$$

where $\mu = \frac{\eta_0 + I_n + p_n G_{n,m}}{\eta_0 + I_n}$ and $\mu > 1$.

Let function $g(x) = (-1-\frac{1}{x})\log_2 x + \frac{2}{\ln 2}(1-\frac{1}{x})$. We analyze function $g(x)$, and its derivative for x is

$$g'(x) = \frac{-x + \ln 2 \log_2 x + 1}{x^2 \ln 2}. \tag{22}$$

Let function $s(x) = -x + \ln 2 \log_2 x$. When $x \geq 1$, $s(x)$ is monotonically decreasing, and $s(x) \leq s(1) = 0$. Therefore, when $x \geq 1$, $g'(x) < 0$, $g(x)$ is monotonically decreasing, and $g(x) \leq g(1) = 0$. Because $\mu > 1$, the second derivative of $E_{n,off}$ with respect to p_n is always less than 0, i.e., $\frac{\partial^2 E_{n,off}}{\partial p_n^2} \leq 0$ ($p_n \geq \max\{p_n', \underline{p_n}\}$). Based on $\frac{\partial E_{n,off}}{\partial p_n} > 0$ and the power variable of each MD that is a closed interval, $E_{n,off}$ takes the minimal value when $p_n = \max\{p_n', \underline{p_n}\}$. Thus, $p_n^* = \max\{p_n', \underline{p_n}\}$, and for any $p_n \geq p_n^*$, there always is $E_n(p_n, \mathcal{P}_{-\mathbf{n}}^*) \geq E_n(p_n^*, \mathcal{P}_{-\mathbf{n}}^*)$.

For MEC m, J_m^* is the first set in Algorithm Greedy-pruning that satisfies the following conditions: (1) $\sum_{k\in J_m^*} f_k'(J_m^*) \leq C_m$; (2) for any MD $l \in \mathcal{G}roup_m \backslash J_m^*$, there is always

$$l = \arg\min_i \sum_{j\in(J_m^*\cup\{l\})\backslash\{i\}} f_j'((J_m^*\cup\{l\})\backslash\{i\}).$$

Then, the maximum number of beneficial MDs with QoS will no longer decrease. Therefore, for any offloading scheduling profile $X_m \in \mathcal{X}_m$ satisfying the conditions $t_n \leq T_n, n \in J_m$ and $\sum_{n\in J_m} f_n \leq C_m$, there always will be $|J_m(X_m)| = \sum_{n\in\mathcal{G}roup_m} x_{n,m} \leq |J_m(X_m^*)| = \sum_{n\in\mathcal{G}roup_m} x_{n,m}^*$. □

Theorem 2. *Given $\mathcal{N}$, G, B, C, and the $p_n (p_n \in \mathcal{P}_n)$, there exists a Nash equilibrium solution set for the formulated game $G = \{(\mathcal{P}_n)_{n\in\mathcal{N}}, \mathcal{X}; (E_n)_{n\in\mathcal{N}}, \sum_{m\in\mathcal{M}} |J_m|\}$.*

Proof. First, the initial value of p_n is set as $\underline{p_n}$. Based on Algorithm 1 (Greedy-pruning), Algorithm 2 ($\mathcal{C}$alculate_$\widetilde{\mathcal{P}}(\cdot)$), and Theorem 1, there exists a Nash equilibrium $\langle \mathcal{P}^*, X^*\rangle$ for $\mathcal{H} = (\mathcal{N}, \{\mathcal{P}_n\}_{n\in\mathcal{N}}, \{E_{n,off}\}; \mathcal{S}, \mathcal{X}, |J|)$. If the transmission power p_n^* of each MD n satisfies $\underline{p_n} \leq p_n^* \leq \overline{p}_n$ and $E_{\{n,off\}}^* \leq E_{\{n,loc\}}$, $\langle \mathcal{P}^*, X^*\rangle$ is the Nash equilibrium solution of game $G = \{(\mathcal{P}_n)_{n\in\mathcal{N}}, \mathcal{X}; (E_n)_{n\in\mathcal{N}}, |J|\}$. That is, game G has reached the Nash equilibrium. Otherwise, the MD will not choose MEC m to execute his task τ_n; meanwhile, $p_n^* = 0$ and $\mathcal{G}roup_m = \mathcal{G}roup_m \backslash \{n\}$. Based on Theorem 1, each MEC m and each MD n update J_m and p_n, respectively, until J_m and each p_n don't change simultaneously. Then, we can find the Nash equilibrium for game $G = \{(\mathcal{P}_n)_{n\in\mathcal{N}}, \mathcal{X}; (E_n)_{n\in\mathcal{N}}, \sum_{m\in\mathcal{M}} |J_m|\}$. □

4.3 Nash Equilibrium Solution Computation

We propose a GCO algorithm to find the equilibrium solution. The initial transmission power value p_n of each MD's power strategy is equal to $\underline{p_n}$, respectively. After the Greedy-Pruning algorithm, each MEC m can achieve an optimal X_m and profile $f_{\mathrm{n}}(J_m)$. Then, each MD n ($n \in \mathcal{G}roup_m$) updates his transmission power $\widetilde{p}_{\mathrm{n}}$ by Algorithm 2 (Calculate_$\widetilde{\mathcal{P}}(\cdot)$). Each player (MEC m and MD) adjusts his strategy continuously to optimal. For each MD n ($n \in J_m$), if $E_{n,off} > E_{n,loc}$ and $t_{n,loc} \leq T_n$, $\mathcal{N}$

will be updated to $\widetilde{\mathcal{G}roup}_m = \mathcal{G}roup_m \backslash \{n\}$. Then, we update the set of MD (players $\mathcal{G}roup_m$) until it does not change anymore. The detailed steps of the GCO algorithm are described in Algorithm 3.

ALGORITHM 3: Game-based Computation Offloading (GCO)

Require: $\mathcal{N}, \underline{\mathcal{P}}, \overline{\mathcal{P}}, G, B, \mathcal{C}, \varepsilon, \delta$.
Ensure: $\mathcal{P}, J, X$.
1: $\mathcal{N}(0) \leftarrow \mathcal{N}$;
2: $s \leftarrow 1$;
3: $p_{\mathbf{n}}(0) \leftarrow \underline{\mathcal{P}}(\mathcal{N}(s-1))$;
4: $\langle J(0), f_{\mathbf{n}}(J(0))\rangle \leftarrow \mathrm{GP}(\mathcal{N}(0), p_{\mathbf{n}}(0), G, B, \mathcal{C})$;
5: $t \leftarrow 0$;
6: **while** $|P(t+1) - P(t)| < \delta$ **do**
7: $\quad p_{\mathbf{n}}(t+1) \leftarrow \mathrm{Calculate\_}\widetilde{\mathcal{P}}(\mathcal{N}(t), p_{\mathbf{n}}(t), G, B, \mathcal{C}, J(t), \varepsilon)$;
8: $\quad \langle J(t+1), f_{\mathbf{n}}(J(t+1))\rangle \leftarrow \mathrm{GP}(\mathcal{N}(s), p_{\mathbf{n}}(t+1), G, B, \mathcal{C})$;
9: $\quad t \leftarrow t+1$;
10: $J \leftarrow J(t)$;
11: $\mathcal{N}(s) \leftarrow \mathcal{N}(s-1)$;
12: **for** $(n \in J)$ **do**
13: $\quad$ **if** $(E_{n,off}(X, \mathcal{P}(t)) > E_{n,loc}(X, \mathcal{P}(t))$ and $t_{n,loc} \le T_n)$ **then**
14: $\quad\quad \mathcal{N}(s) \leftarrow \mathcal{N}(s) \backslash \{n\}$;
15: **while** $(\mathcal{N}(s) \neq \mathcal{N}(s-1))$ **do**
16: $\quad s \leftarrow s+1$;
17: $\quad$ *Repeat* steps 3 to 14;
18: **return** $\mathcal{P}, J, X$.

We know that M and N are the number of MECs and mobile devices, respectively. We set $a_{max}N = \max_{m\in\mathcal{M}}\{\mathcal{G}roup_m\}$, where $a_{max} \le 1$. Assuming $h(n), g(n)$ are the number of *while* loops and computation overhead of every *while* loop required for a group with n MDs. Then, the computational overheads of Algorithm 1 is $O(\sum_{m\in\mathcal{M}} h(|\mathcal{G}roup_m|)g(|\mathcal{G}roup_m|))$. The worst case is $O(Mh(a_{max}N)g(a_{max}N))$. The computation overhead of Algorithm 2 is $O(Ma_{max}N \lg P)$. Assuming the number of convergence times of Algorithm GCO is $O(k)$, then the computational overhead of Algorithm GCO in the worst case is $O(kMh(a_{max}N)g(a_{max}N) + kMa_{max}N \lg P)$. From the analysis of Section 5.2.2, we can know that $h(a_{max}N) = O(N)$ and $g(a_{max}N) = O(N^2)$. Therefore, the computational overhead of Algorithm GCO is $O(kMN^3 + kMN \lg P)$.

4.4 Convergence of the GCO Algorithm

We should verify that whether the obtained solution sequence from the Algorithm GCO converges to the Nash equilibrium. If a sequence is monotonic and bounded, then the sequence is convergent.

Theorem 3. *Assuming the Nash equilibrium solution of non-cooperative game strategies for N MDs and M MECs* $\mathcal{H} = (\mathcal{N}, \{\mathcal{P}_n\}_{n\in\mathcal{N}}, \{E_{n,off}\}; \mathcal{S}, \mathcal{X}, |J|)$ *as* $\langle \mathcal{P}^*, X^*\rangle$, *sequence solution* $\boldsymbol{p}(t)$ *obtained by the proposed GCO algorithm converges to* $\langle \mathcal{P}^*, X^*\rangle$.

Proof. For MD n, the transmission power sequence is bounded because $\underline{p_n} \le p_n \le \overline{p}_n$. Then, we prove the transmission power sequence obtained from the *while* loop (lines 6–9) in algorithm GCO is monotonicity.

It is assumed that the loop has iterated t times, that is, $P_{\mathbf{n}}(t)$, $J(t)$, and $f_{\mathbf{n}}(t)$ have been calculated. $J(t)$, $f_{\mathbf{n}}(t)$ are updated based on $P_{\mathbf{n}}(t)$. For MD n $(n \in \mathcal{N}(s))$, $n \in J(t)$ or $n \notin J(t)$.

Table 2. System Parameters

System parameters	Variable range
Number of MDs (N)	[15–70]
Number of MECs (M)	[1–3]
Input task data size (d_n)	(0,2]MB
Workload requirement (w_n)	[100,500]cycles/bit
Expected Time (T_n)	(0,3]s
Transmission power range (p_n)	100-[1000,3000]mWatts
Computational capacity of $\mathcal{S}$ (C)	1GHz
Local computation capability (f_n^{loc})	$\{0.05, 0.1, 0.15, 0.2\}$GHz
Channel bandwidth (B)	10MHz
Background noise power (η_0)	−100dBm
Distance between n to $\mathcal{S}$ (dis_n)	(0,50]m
Path loss factor (α)	−4

(1) $n \notin J(t)$: MD n can increase p_n to provide his own competitiveness. Then, $p_n(t+1) = \min\{p_n^1, p_n^2, \overline{p}_n\} \geq p_n(t)$.
(2) $n \in J(t)$: The computation capability $f_n(t)$ assigned to MD n by the MEC $\mathcal{S}$ satisfies $t_n \leq T_n$, that is, $p_n(t) \geq p'_n$. Based on Algorithm 2, $p_n(t+1) = p_n(t)$.

In summary, each MD's transmission power sequence satisfies $\underline{p_n} \leq \cdots \leq p_n(t-1) \leq p_n(t) \leq \overline{p}_n$. □

5 SIMULATIONS

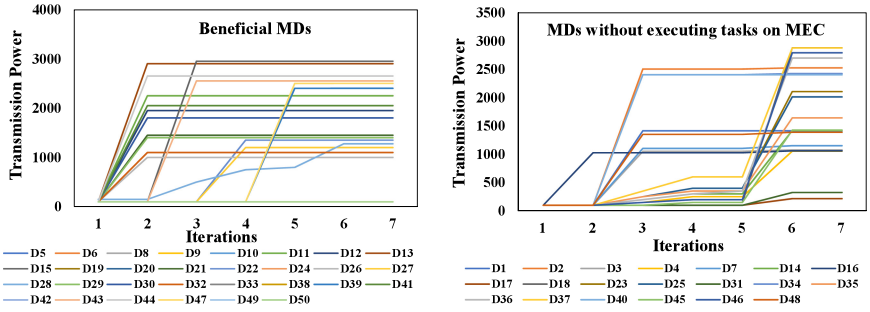
5.1 Simulation Settings

We evaluate the system performance of the proposed GCO algorithm. As shown in Table 2, we consider N MDs and M MECs in this system, where N and M are randomly selected from the interval [15−70] and [1−3], respectively. The size of the input task data d_n of each MD n is randomly selected from the interval $(0, 2]$MB and the total number of required CPU cycles $c_n = d_n \cdot w_n$, where w_n is the workload requirements of task τ_n ($w_n \in [100, 500]$ cycles/bit). Similarly, the expected Time T_n of MD n also follows a uniform distribution with $(0, 3]$s. The minimum transmission power p_n is 100mWatts, and the maximum value is randomly selected from the interval $[1000, 3000]$mWatts. The local computation capacity f_n^{loc} of MD n is randomly selected from the set $\{0.05, 0.1, 0.15, 0.2\}$GHz. We consider MEC $\mathcal{S}$ has a coverage range of 50m. The computational capacity C of MEC $\mathcal{S}$ is 1GHz. The bandwidth $B = 10$MHz and the background noise power $\eta_0 = -100$dBm. Based on the wireless interference model for urban cellular radio environment, the channel gain $G_{n,m} = dis_{n,m}^{\alpha}$, where $dis_{n,m}$ is the distance between MD n and the MEC m, and $\alpha = -4$ is the path loss factor. Our experiments are performed on i5-6200U 2cores CPU@2.3GHz2.4GHz platform.

5.2 Simulation Results

5.2.1 Convergence of Algorithm GCO. We first consider 50 MDs and one MEC system. The remaining parameters are shown in Table 2.

Figure 2(a) and (b) illustrate the convergence process of transmission power for each MD by executing our proposed GCO algorithm. With the number of iterations increasing, the transmission power of each MD is increasing, and then the curve reaches to a stable value. During the



(a) (b)

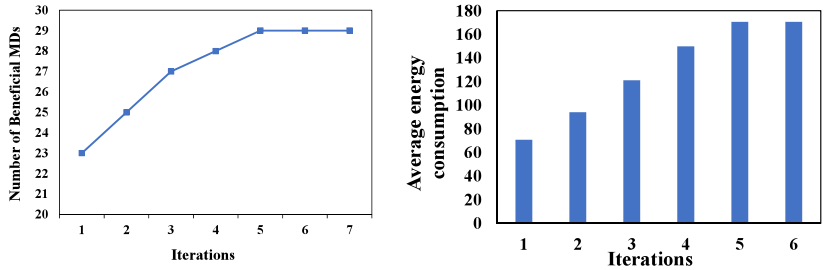
Fig. 2. Change of transmission power in the iterative process.

Table 3. Changes in Beneficial MDs during Each Round of Changes in the Transmission Power of MDs

D2	D3	D10	D11	D12	D13	D15	D16	D17	D21	D22	D26	D27
1	1	0	0	0	0	1	0	1	0	1	0	1
0	0	1	1	1	1	0	1	1	1	1	1	1
0	0	1	1	1	0	1	1	0	1	0	1	0
0	0	1	1	1	1	1	0	0	1	1	1	1
0	0	1	1	1	1	1	1	0	1	1	1	1
0	0	1	1	1	1	1	1	0	1	1	1	1
D29	D30	D31	D32	D34	D39	D40	D41	D43	D44	D47	D48	
0	0	1	0	1	1	1	0	1	0	1	1	
1	1	0	1	0	1	0	1	0	0	1	0	
1	1	0	1	0	1	0	1	1	1	1	0	
1	1	0	1	0	0	1	1	1	1	0	0	
1	1	0	1	0	1	0	1	1	1	1	0	
1	1	0	1	0	1	0	1	1	1	1	0	

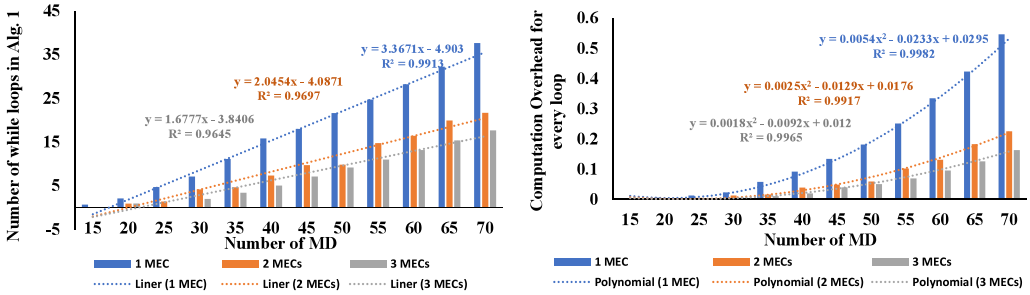
process of computing, some MDs will withdraw the resource competition if the transmission power is higher than their accepted maximum value, i.e., $p_i > \overline{p}_i$. Figure 2(b) is the transmission power curve of MDs who don't offload their tasks to MECs. Table 3 illustrates the detailed changes in beneficial MDs during each round of changes in the transmission power of MDs. The beneficial MD and the MD without executing his task on the MEC are represented by 1 and 0, respectively. In addition, the specific change between 0 and 1 is marked in blue. Moreover, we can know that the transmission power can be obtained after six iterations, which shows high efficiency of our proposed algorithm.

Figure 3(a) and (b) are the curve of the number of MDs who obtain computing resources provided by MEC and the bar graph of the average energy consumption during the iterative process, respectively. At the beginning, each MD's transmission power is set as the initial value, i.e., the minimum value. The number of MDs with QoS served by the MEC with limited computing resources is 23, and the average energy consumption is about 70. Each MD increases its transmission power to compete for the computing resources of MEC, which causes the average energy consumption to rise during the iteration, as shown in Figure 3(b). In Figure 3(a), after several rounds of mutual



(a) The process change of number of beneficial MDs (b) The process change of average energy consumption

Fig. 3. The changes during the iterative process.



(a) The number of *while* loops of number of beneficial MDs in Algorithm. 1 (b) The computation overhead for every loop

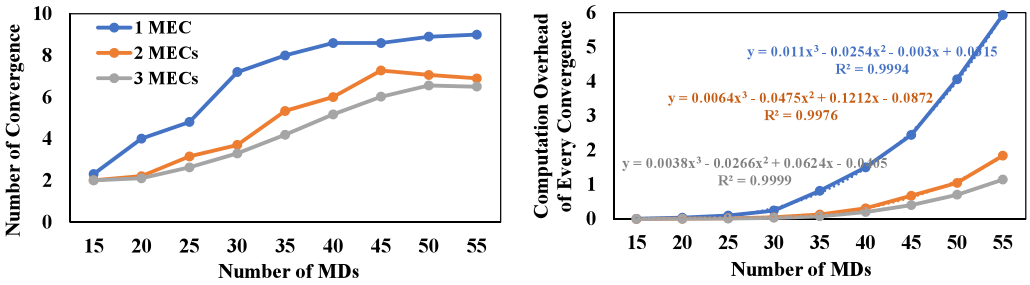
Fig. 4. The analysis of Algorithm 1.

negotiation between MDs, the number of MDs who can use the computing resources provided by MEC gradually increases and reaches a stable value 29 as the number of iterations increases.

5.2.2 Performance Evaluation. We analyze the performance of Algorithm 1 from the perspectives of the number of both *while* loops and computation overhead of every loop. The number of MDs N is selected from the interval [15–70] with the increment of 5. The number of MECs M is the element of $\{1, 2, 3\}$. For each N and M, we repeat the experiment 20 times. The experimental results are shown in Figure 4(a) and (b).

Figure 4(a) and (b) shows the number of *while* loops and computation overhead of every loop in Algorithm 1 as the number of MDs increases, respectively. With the number of MDs increasing, green, orange, and gray bars represent the analysis of 1 MEC, 2 MECs, and 3 MECs systems, respectively. Each group bar has further added the corresponding trend line. From Figure 4(a), for each M MECs system, the number of *while* loops increase linearly as the scale of MDs increases. Besides, for each N MDs, the greater the number of MECs M, the fewer number of the *while* loops. From Figure 4(b), for each M MECs system, computation overhead of every loop increases in a second polynomial. Fitting degrees between the bars and their trend lines are greater than 0.99.

Then, we analyze the performance of the GCO algorithm, which is evaluated from two respects: the number of convergence and the computation overhead of every convergence. Like Figure 4, three colors represent 1 MEC, 2 MECs, and 3 MECs, respectively. From Figure 5(a), the general trend of every curve for each M MECs system increases and reaches a stable value. In addition, even if the number of MDs is 55, the average number of convergence time is very small, just 9.1. In



(a) Average iterative times of different scales of MDs (b) The computation overhead of every convergence

Fig. 5. The analysis of Algorithm GCO.

Figure 5(b), as the scale of MDs increases, the computation overhead curve of each M MECs system increases in a polynomial. The dashed line is the trend line of the corresponding computation overhead curve of every convergence, which is a third-order polynomial. Fitting degrees of the trend lines and the corresponding dashed curve are more than 0.99. Besides, with the number of MDs N increasing, the difference between each M MECs system regarding the computation overhead of every convergence is increasing.

5.2.3 Comparisons with RPCO and DCOA. We compare our GCO algorithm to other algorithms (RPCO and DCOA [4]) to show our high efficiency and feasibility. We briefly introduce the Random Power Computation Offloading (RPCO) algorithm and Distributed Computation Offloading Algorithm (DCOA).

RPCO: The transmission power of each MD is randomly selected from the interval [100 − 3000]mWatts. Other relevant conditions and parameters are the same as in Table 2. MEC $\mathcal{S}$ selects the offloading strategy by optimizing the number of tasks with delay deadline. We scale the number of MD from [15−55] with the increase of 5. We compare our GCO algorithm and RPCO from the number of MDs selected by MEC $\mathcal{S}$.

DCOA: It is assumed that MEC satisfies the deadline requirements of each offloaded MD. The CPU computational capability f_n^m of MD n is randomly assigned from the set $\{0.5, 0.8, 1.0\}$GHz to account for the heterogenous computing capability of mobile devices. Each MD optimizes his objective function $Z_n(a_n, a_{-n})$ by selecting the appropriate channel, where Z_n is the system-wide computation overhead and a_n is the selected channel of MD n. Besides, the transmission rates of MDs in the same channel affect each other.

Figure 6 is the comparison between our GCO algorithm and RPCO on the number of selected MDs. For each N value, we do many times of experiments and get the average value. As the number of MDs increases, the overall trend in these four-type bars is rising. Compared with the RPCO algorithm, the number of beneficial MDs of three systems (1 MEC, 2 MECs, and 3 MECs) improves 4%−17%, 11%−40%, and 11%−56%, respectively. As the number of MDs increases, the greater the number of MECs, the more the number of beneficial MDs increases. From Figure 6, we can see that our GCO algorithm selects more MDs under the same limited-resource of MECs, which shows the superiority and feasibility of our GCO algorithm.

We compare the number of changing strategies between MECs and MDs with Algorithm DCOA, which is shown Figure 7(a). As the strategies of MDs change, the computation resource allocation strategy of MECs in Algorithm 1 will change correspondingly. We use the number of changing strategies (MECs, MDs) to represent the total changing number between MECs and MDs, which is the same as "decision slots" in Ref. [4]. Note that the system in the DCOA algorithm considers

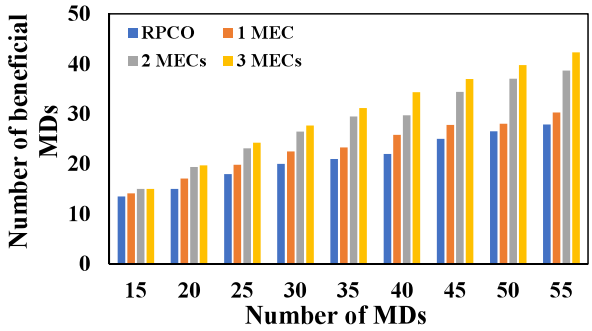
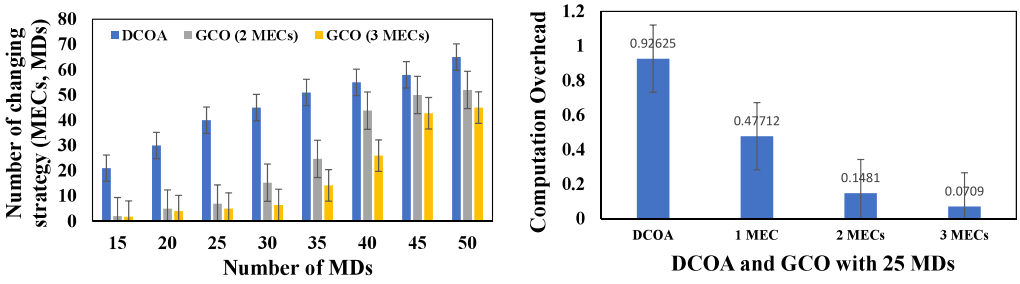


Fig. 6. Comparison of the number of selected MDs.



(a) (b)

Fig. 7. Comparison between GCO and DCOA.

five channels, so we choose two MECs and three MECs system in our GCO algorithm to compare. From Figure 7(a), we can know that the trends of DCOA and GCO algorithms increase with the number of MDs increasing. However, compared to DCOA, the number of iterations in our GCO algorithm is reduced by at least 15%. Besides, according to Ref. [4], we can know that the number of MDs is 25 when the iteration time of calculating the computation overhead is 38. Therefore, we compare the GCO and DCOA algorithm with the computation overhead when the number of MDs is 25. Experiment results in Figure 7(b) show that the overall computation overhead is reduced by at least 50%.

6 CONCLUSIONS

Our study focuses on the task offloading problem of one MEC and multiple MDs with delay deadlines. From the perspective of MEC $\mathcal{S}$, we propose a greedy algorithm to optimize the number of served MDs with delay deadline. We analyze the energy consumption of all tasks executed on MEC $\mathcal{S}$ based on the non-cooperative game theoretical method. We prove the existence of Nash equilibrium solution and propose GCO algorithm to solve it. Besides, the convergence of the algorithm is also analyzed. Extensive simulated experiments results and the comparison experiments with the RPCO solution validate and show the feasibility of our proposed method. In future work, we will consider the multiple MDs mobility and the task migration in a real-time multiple edge computing system based on our existing work to explore the impact of task migration on the latency performance of multiple MECs.

ACKNOWLEDGMENTS

The authors are grateful to the anonymous reviewers for their constructive comments. A preliminary version of the article was presented at the 16th Annual IFIP International Conference on Network and Parallel Computing (NPC 2019), Hohhot, Inner Mongolia, China, August 23–24, 2019. The research was supported by the National Natural Science Foundation of China under Grant No. 61876061.

REFERENCES

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie. 2018. Mobile edge computing: A survey. *IEEE Internet of Things Journal* 5, 1 (2018), 450–465.

[2] M. Chen and Y. Hao. 2018. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE Journal on Selected Areas in Communications* 36, 3 (2018), 587–597.

[3] W. Chen, D. Wang, and K. Li. 2019. Multi-user multi-task computation offloading in green mobile edge cloud computing. *IEEE Transactions on Services Computing* 12, 5 (2019), 726–738.

[4] X. Chen, L. Jiao, W. Li, and X. Fu. 2016. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Transactions on Networking* 24, 5 (2016), 2795–2808.

[5] B. Du, R. Huang, Z. Xie, J. Ma, and W. Lv. 2018. KID model-driven things-edge-cloud computing paradigm for traffic data as a service. *IEEE Network* 32, 1 (2018), 34–41.

[6] Q. Fan and N. Ansari. 2018. Application aware workload allocation for edge computing-based IoT. *IEEE Internet of Things Journal* 5, 3 (2018), 2146–2153.

[7] Daniel Grosu and Anthony T. Chronopoulos. 2005. Noncooperative load balancing in distributed systems. *Journal of Parallel and Distributed Computing* 65, 9 (2005), 1022–1034.

[8] Hongzhi Guo and Jiajia Liu. 2018. Collaborative computation offloading for multiaccess edge computing over fiberÍcwireless networks. *IEEE Transactions on Vehicular Technology* 67, 5 (May 2018).

[9] X. Hu, K. Wong, and K. Yang. 2018. Wireless powered cooperation-assisted mobile edge computing. *IEEE Transactions on Wireless Communications* 17, 4 (2018), 2375–2388.

[10] S. M. R. Islam, N. Avazov, O. A. Dobre, and K. Kwak. 2017. Power-domain non-orthogonal multiple access (NOMA) in 5G systems: Potentials and challenges. *IEEE Communications Surveys and Tutorials* 19, 2 (2017), 721–742.

[11] A. Kiani and N. Ansari. 2018. Edge computing aware NOMA for 5G networks. *IEEE Internet of Things Journal* 5, 2 (2018), 1299–1306.

[12] K. Li. 2018. A game theoretic approach to computation offloading strategy optimization for non-cooperative users in mobile edge computing. *IEEE Transactions on Sustainable Computing* (Sept. 2018). DOI: https://doi.org/10.1109/TSUSC.2018.2868655

[13] K. Li. 2019. Computation offloading strategy optimization with multiple heterogeneous servers in mobile edge computing. *IEEE Transactions on Sustainable Computing* (March 2019). DOI: https://doi.org/10.1109/TSUSC.2019.2904680

[14] K. Li, C. Liu, K. Li, and A. Y. Zomaya. 2016. A framework of price bidding configurations for resource usage in cloud computing. *IEEE Transactions on Parallel and Distributed Systems* 27, 8 (2016), 2168–2181.

[15] X. Li, D. Li, J. Wan, C. Liu, and M. Imran. 2018. Adaptive transmission optimization in SDN-based industrial Internet of Things with edge computing. *IEEE Internet of Things Journal* 5, 3 (2018), 1351–1360.

[16] C. Liu, K. Li, J. Liang, and K. Li. 2019. COOPER-SCHED: A cooperative scheduling framework for mobile edge computing with expected deadline guarantee. *IEEE Transactions on Parallel and Distributed Systems* (2019), 1–1.

[17] C. Liu, K. Li, C. Xu, and K. Li. 2016. Strategy configurations of multiple users competition for cloud service reservation. *IEEE Transactions on Parallel and Distributed Systems* 27, 2 (2016), 508–520.

[18] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief. 2016. Delay-optimal computation task scheduling for mobile-edge computing systems. In *2016 IEEE International Symposium on Information Theory (ISIT)*. 1451–1455.

[19] L. Ma, X. Liu, Q. Pei, and Y. Xiang. 2019. Privacy-preserving reputation management for edge computing enhanced mobile crowdsensing. *IEEE Transactions on Services Computing* 12, 5 (2019), 786–799.

[20] Y. Mao, J. Zhang, and K. B. Letaief. 2017. Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. 1–6.

[21] M. R. Musku, A. T. Chronopoulos, D. C. Popescu, and A. Stefanescu. 2010. A game-theoretic approach to joint rate and power control for uplink CDMA communications. *IEEE Transactions on Communications* 58, 3 (2010), 923–932.

[22] John F. Nash. 1950. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences* 36, 1 (1950), 48–49. DOI: https://doi.org/10.1073/pnas.36.1.48 arXiv:https://www.pnas.org/content/36/1/48.full.pdf

[23] J. L. D. Neto, S. Yu, D. F. Macedo, J. M. S. Nogueira, R. Langar, and S. Secci. 2018. ULOOF: A user level online offloading framework for mobile edge computing. *IEEE Transactions on Mobile Computing* 17, 11 (2018), 2660–2674.

[24] Z. Ning, X. Wang, and J. Huang. 2019. Mobile edge computing-enabled 5g vehicular networks: Toward the integration of communication and computing. *IEEE Vehicular Technology Magazine* 14, 1 (2019), 54–61.
[25] Satish Penmatsa and Anthony T. Chronopoulos. 2011. Game-theoretic static load balancing for distributed systems. *Journal of Parallel and Distributed Computing* 71, 4 (2011), 537–555.
[26] S. Ranadheera, S. Maghsudi, and E. Hossain. 2018. Computation offloading and activation of mobile edge computing servers: A minority game. *IEEE Wireless Communications Letters* 7, 5 (2018), 688–691.
[27] T. G. Rodrigues, K. Suto, H. Nishiyama, N. Kato, and K. Temma. 2018. Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration. *IEEE Transactions on Computers* 67, 9 (2018), 1287–1300.
[28] G. Scutari, D. P. Palomar, F. Facchinei, and J. Pang. 2010. Convex optimization, game theory, and variational inequality theory. *IEEE Signal Processing Magazine* 27, 3 (2010), 35–49.
[29] X. Sun and N. Ansari. 2017. Latency aware workload offloading in the cloudlet network. *IEEE Communications Letters* 21, 7 (2017), 1481–1484.
[30] X. Tao, K. Ota, M. Dong, H. Qi, and K. Li. 2017. Performance guaranteed computation offloading for mobile-edge cloud computing. *IEEE Wireless Communications Letters* 6, 6 (2017), 774–777.
[31] E. E. Tsiropoulou, G. K. Katsinis, and S. Papavassiliou. 2012. Distributed uplink power control in multiservice wireless networks via a game theoretic approach with convex pricing. *IEEE Transactions on Parallel and Distributed Systems* 23, 1 (2012), 61–68.
[32] E. E. Tsiropoulou, P. Vamvakas, and S. Papavassiliou. 2012. Energy efficient uplink joint resource allocation non-cooperative game with pricing. In *2012 IEEE Wireless Communications and Networking Conference (WCNC).* 2352–2356.
[33] F. Wang, J. Xu, X. Wang, and S. Cui. 2018. Joint offloading and computing optimization in wireless powered mobile-edge computing systems. *IEEE Transactions on Wireless Communications* 17, 3 (2018), 1784–1797.
[34] K. Wang, H. Yin, W. Quan, and G. Min. 2018. Enabling collaborative edge computing for software defined vehicular networks. *IEEE Network* 32, 5 (2018), 112–117.
[35] L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji. 2018. Mobile edge computing empowered energy efficient task offloading in 5G. *IEEE Transactions on Vehicular Technology* 67, 7 (2018), 6398–6409.
[36] J. Zhang, X. Hu, Z. Ning, E. C. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu. 2018. Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks. *IEEE Internet of Things Journal* 5, 4 (2018), 2633–2645.
[37] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang. 2018. Cooperative content caching in 5G networks with mobile edge computing. *IEEE Wireless Communications* 25, 3 (2018), 80–87.
[38] Z. Zhao, G. Min, W. Gao, Y. Wu, H. Duan, and Q. Ni. 2018. Deploying edge computing nodes for large-scale IoT: A diversity aware approach. *IEEE Internet of Things Journal* 5, 5 (2018), 3606–3614.

Received September 2019; revised April 2020; accepted May 2020