Partial Offloading Scheduling and Power Allocation for Mobile Edge Computing Systems

Zhufang Kuang, *Member, IEEE*, Linfeng Li, Jie Gao, *Member, IEEE*, Lian Zhao, *Senior Member, IEEE*, and Anfeng Liu

***Abstract*—Mobile edge computing (MEC) is a promising technique to enhance computation capacity at the edge of mobile networks. The joint problem of partial offloading decision, offloading scheduling, and resource allocation for MEC systems is a challenging issue. In this paper, we investigate the joint problem of partial offloading scheduling and resource allocation for MEC systems with multiple independent tasks. A partial offloading scheduling and power allocation (POSP) problem in single-user MEC systems is formulated. The goal is to minimize the weighted sum of the execution delay and energy consumption while guaranteeing the transmission power constraint of the tasks. The execution delay of tasks running at both MEC and mobile device is considered. The energy consumption of both the task computing and task data transmission is considered as well. The formulated problem is a nonconvex mixed-integer optimization problem. In order to solve the formulated problem, we propose a two-level alternation method framework based on Lagrangian dual decomposition. The task offloading decision and offloading scheduling problem, given the allocated transmission power, is solved in the upper level using flow shop scheduling theory or greedy strategy, and the suboptimal power allocation with the partial offloading decision is obtained in the lower level using convex optimization techniques. We propose iterative algorithms for the joint problem of POSP. Numerical results demonstrate that the proposed algorithms achieve near-optimal delay performance with a large energy consumption reduction.**

***Index Terms*—Convex optimization, energy efficiency, resource allocation, task offloading, task scheduling.**

Manuscript received February 1, 2019; revised March 31, 2019; accepted April 9, 2019. Date of publication April 16, 2019; date of current version July 31, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61309027, Grant 617725534, and Grant 61728108, in part by the Hunan Provincial Natural Science Foundation of China under Grant 2018JJ3888, in part by the Scientific Research Fund of Hunan Provincial Education Department under Grant 18B197, in part by the National Key Research and Development Program of China under Grant2018YFB1700200, in part by the Open Research Project of Key Laboratory of Intelligent Information Perception and Processing Technology (Hunan Province) under Grant 2017KF01, and in part by the Foundation Project of Hunan Internet of Things Society under Grant 2018-2. *(Corresponding author: Zhufang Kuang.)*

Z. Kuang is with the School of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China, and also with the Key Laboratory of Intelligent Information Perception and Processing Technology (Hunan Province), Zhuzhou 412008, China (e-mail: zfkuangcn@163.com).

L. Li is with the School of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China (e-mail: linfenglicn@163.com).

J. Gao and L. Zhao are with the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada (e-mail: j.gao@ryerson.ca; l5zhao@ryerson.ca).

A. Liu is with the School of Computer Science and Engineering, Central South University, Changsha 410010, China (e-mail: afengliu@mail.csu.edu.cn).

Digital Object Identifier 10.1109/JIOT.2019.2911455

I. Introduction

GLOBAL mobile data traffic is expected to reach 49EB by 2021, up sevenfold from 2016, with video traffic accounting for 78% [1]. Meanwhile, the emergence of low-latency services, such as face/fingerprint/iris recognition, augmented/virtual reality, natural language processing, interactive gaming, Internet of Things (IoT), and Internet of vehicles has brought great challenges to existing mobile communication networks [2]. In the existing network architecture, business traffic needs to flow through the entire access network and core network, and through multiple base stations and other key equipment, transport equipment, and so on. Even with improvement in the wireless aspect of transmission bandwidth, there is still an unpredictable end-to-end delay [3]. In order to effectively meet the high bandwidth and low latency required by the rapid development of mobile Internet and IoT, the European telecommunications standardization institute (ETSI) proposed mobile edge computing (MEC) in 2014 [4]. Through offloading the computation intensive tasks from mobile devices to MEC server, the energy consumption and latency can be reduced. In addition to MEC systems, the energy efficiency and delay are also very important optimization objectives for wireless networks in general [5]–[9].

The minimization of the system delay and energy consumption in the MEC offloading technique requires joint allocation of communication and computation resources between mobile devices and MEC servers. There are lots of recent works on this topic for single-user MEC systems, multiuser MEC systems, MEC systems with heterogeneous servers [10], [11], virtual machine migration [12]–[14], and computation offloading and caching [15]–[18].

For a single-user MEC system, the joint problem of task offloading scheduling and transmit power allocation is investigated in [19]. In [20], the resource allocation in ultralow power fog-computing switching simultaneous wireless information and power transfer (SWIPT)-based single-user MEC systems is investigated. In [21], considering the constrained power and unpredictable tasks, the power allocation problem with the goal of maximizing the processing capacity of the single-user MEC system is studied, a binary-search water-filling algorithm is proposed. In [22], the cooperation of cloud computing and MEC in IoT is investigated, and the single user computation offloading problem is solved by the branch and bound algorithm. The task offloading scheduling, and power allocation problem is studied in [19]. However, only the full offloading is considered in this paper. The task partial offloading decision and offloading scheduling are intercoupled with each other. In [20]–[22], task offloading scheduling is not considered, and joint task partial offloading decision, offloading scheduling and resource allocation in single-user MEC systems remains an open problem.

For multiuser MEC scenario, there are some recent studies. In [23], the joint problem of application partitioning and collaborative computation offloading are considered, so that mobile devices may help each other on job execution in order to meet the completion deadline of the applications while minimizing the overall energy consumption. In [24], the collaborative MEC offloading in multiuser multi-MEC is studied, and a game-theoretical computation offloading scheme is proposed. In [25], in order to reduce energy consumption, two kinds of time-to-live (TTL) in four cache replacement policies to cache data at the edge is proposed. In [26], this paper models the network where users share the communication channel to offload their computations as a competitive game. Optimum offloading decision for minimizing user energy consumption while satisfying the hard deadline of the applications is obtained. In [27], the joint communication and computation resource allocation problem in a multiuser time-division multiple access MEC system with the goal of minimizing latency is investigated. In [28], the problem of cloud-MEC collaborative computation offloading is investigated, and a game-theoretic collaborative computation offloading scheme is proposed. In [29], the task offloading problem in ultradense network is studied, and the goal is to minimize the delay while saving the battery life of users equipment. In [30], for a homogeneous fog network, the tradeoff between performance gain and energy cost of collaborative task offloading is investigated.

The above studies mainly consider the multiuser MEC offloading scenario with the objective of minimizing the energy consumption or latency. However, task offloading scheduling has not been considered in multiuser MEC offloading systems recently as well. Essentially, the joint offloading decision and offloading scheduling problem in multiuser MEC offloading scenario, which will be investigated in our future work, is a challenging issue.

Apart from the computation offloading problem in the homogeneous MEC, macro base stations (MBSs) and small cells are considered in the heterogeneous MEC. The energy harvesting capability has also been considered. Considering the energy harvesting constraint and the UAV speed constraint, the partial and binary computation offloading problem in UAV-enabled MEC wireless powered system is investigated with the goal of computation rate maximization [31]. In [32], considering the scenario of including MBSs and small cells in ultradense IoT networks, a game-theoretic greedy offloading scheme is proposed to handle the computation offloading problem. In [33], considering the residual energy of smart device battery for the weighting factor of energy consumption, and latency in the single and multicell MEC network scenarios, energy and latency tradeoff energy-aware scheme is investigated. In [34], the problem of computation offloading in a network is studied, where both the mobile users and the vehicle-based cloudlet servers can be highly mobile. In [35], a joint radio and computational resource allocation problem to optimize the system performance and improve user satisfaction is studied, and a matching game framework is proposed.

However, these aforementioned studies in multiuser or heterogeneous MEC assume that MEC server can execute multiple tasks concurrently. Although such assumption is tractable, it may be not applicable for computation resource limited MEC server, e.g., for MEC server with a single-core CPU and a single transmission channel [19]. The joint problem of task offloading scheduling and resource allocation in single-user MEC has not been investigated.

In this paper, we consider a single mobile device with multiple independent computation tasks and parallel implementation of task offloading and execution. We design an energy efficient and low latency MEC offloading mechanism for a mobile device in a single-user MEC system. This mechanism minimizes the system energy consumption and concurrently reduces the system delay. The main contributions of this paper are as follows.

1) We model the task partial offloading scheduling and resource allocation problem for an MEC system with multiple independent tasks. The joint problem of the task offloading decision, offloading scheduling, and power allocation is considered. With the objective of minimizing the weighted sum of the execution delay and energy consumption, a nonlinear mixed-integer optimization problem is formulated.
2) We propose a two-level alternation method framework based on Lagrangian dual decomposition. In the upper level, the task offloading decision and offloading scheduling with given the allocated transmission power is solved using flow shop scheduling theory or greedy strategy; and in the lower level, the suboptimal power allocation with the partial offloading decision is obtained using convex optimization techniques.
3) We propose a partial offloading decision and scheduling based on Johnson (POJ) algorithm and a partial offloading decision and scheduling based on the number of CPU-cycles required (POC) algorithm in the upper level, and a partial offloading scheduling and power allocation (POSP) algorithm in the lower level, respectively. A joint POSP (JPOA) iterative algorithm is proposed based on Lagrange multiplier method and Johnson method.

The rest of this paper is organized as follows. The system model and problem formulation are presented in Section II. The two level alternation optimization algorithm is presented in Section III. Simulation results are discussed in Section IV. Finally, we conclude this paper in Section V.

II. System Model and Problem Formulation

In this section, the system model is introduced. The joint partial offloading scheduling and resource allocation problem with the goal of minimizing the weighted sum of the execution delay and energy consumption is formulated.

A. System Model

MEC system mainly comprises of mobile devices and an MEC server, which is a server with robust computing power

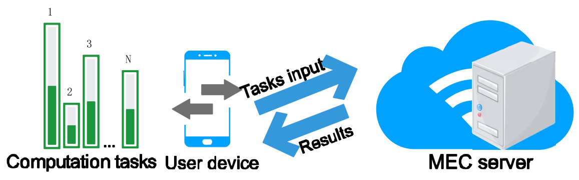


Fig. 1. MEC system with a mobile device and MEC server.

and can provide high strength computing services for mobile devices. Mobile devices offload the computing tasks to MEC server; then MEC server passes the computing result back to the corresponding device after processing. We consider one mobile device in the MEC system, as shown in Fig. 1. The mobile device saves energy and reduces latency by offloading tasks to MEC server.

We consider user device has N independent computing tasks to be processed. We assume that all tasks are available for offloading and scheduling. We denote $\Gamma = \{T_1, T_2, \ldots, T_N\}$ as the set of all tasks. Each task T_i is described by two parameters, i.e., D_i and C_i. For task T_i, D_i (in bits) is the amount of data to be processed; and C_i (in CPU cycles) is the number of CPU cycles for the processing. In this paper, we focus on the scenarios where the mobile device has limited computational resources. Each task is atomic and could not be further divided. In addition, we do not consider the individual task execution time constraint. The mobile device just outfits one antenna and can transmit only one task each time. Thus, we denote R_i as the transmission rate of the task T_i, which is given by

$$R_i = w \log_2\left(1 + \frac{g_0 (d_0/d)^\theta p_i}{N_0 w}\right) \tag{1}$$

where p_i is the transmission power of task T_i, w is the channel bandwidth, g_0 is the path loss constant, θ is the path loss exponent, d_0 is the reference distance, d is the distance from the mobile device to MEC server, and N_0 is the noise power spectral density between the mobile device and MEC.

For each task T_i, it can be either executed locally by the mobile device itself or by the MEC server via computation offloading. For this reason, an offloading decision should be made for each computing task. We denote $\mathbb{X} = \{x_i\}$ and $\mathbb{P} = [p_1, p_2, \ldots, p_N]$ as the offloading decision and transmission power vector. Let $x_i \in \{0, 1\}$ indicate whether the task T_i ($T_i \in \Gamma$) is offloaded to MEC server. Specifically, $x_i = 1$ if the task T_i is offloaded to MEC server; otherwise $x_i = 0$. We denote $\mathbb{S} = [S_1, S_2, \ldots, S_{Ns}]$ as the vector of offloaded tasks, in which the order of elements is the order of offloading. Essentially, $\mathbb{S}$ represents the task offloading scheduling decision, where S_i is the ith task being offloaded to the MEC server and executed at the MEC server. Hence, $\mathbb{S}$ satisfies $S_i \neq S_j, i \neq j, \forall S_i, S_j \in \Gamma$. We denote $\mathbb{L} = [L_1, L_2, \ldots, L_{Nl}]$ as the vector of local processing tasks, in which the order of elements is the order of execution. It follows that $\Gamma = \mathbb{L} \cup \mathbb{S}$, $\mathbb{L} \cap \mathbb{S} = \Phi$, and $N = Ns + Nl$, where Ns and Nl are the number of tasks executed by MEC remotely and mobile device locally, respectively.

The MEC server has a single core CPU and can only perform tasks in the first in first out (FIFO) mode. Additionally, in order to execute the tasks in the order given by $\mathbb{S}$, we assume that the MEC server deploys a task cache with enough memory to store tasks that the CPU has not yet processed. In our scenario, we assume that the computation results of the tasks are small in terms of data size, so that the feedback delay can be ignored. The CPU cycle frequency of the MEC server is represented as f_{ser} (in Hz). The CPU cycle frequency of the mobile device is represented as f_{user} (in Hz). Thus, we denote $t_{L_i}^{\text{dispose}}$ and $t_{L_i}^{\text{comp}}$ as the time duration of the local execution and the time from $t = 0$ to the completion of task L_i, respectively, which is given by

$$t_{L_i}^{\text{dispose}} = \frac{C_{L_i}}{f_{\text{user}}} \tag{2}$$

$$t_{L_i}^{\text{comp}} = \sum_{j=1}^{j=i} t_{L_j}^{\text{dispose}}. \tag{3}$$

Denote $t_{S_i}^{\text{dispose}}$ as the MEC execution duration of task S_i, which can be found using the following equation:

$$t_{S_i}^{\text{dispose}} = \frac{C_{S_i}}{f_{\text{ser}}}. \tag{4}$$

Denote $t_{S_i}^{\text{trans}}$ as the transmission time of task S_i, which is given by

$$t_{S_i}^{\text{trans}} = \frac{D_{S_i}}{R_{S_i}}. \tag{5}$$

Denote $e_{i,s}$ and $e_{i,l}$ as the energy consumption when the task S_i is executed by MEC and when the task L_i is executed locally, respectively, which are given by

$$e_{i,s} = \delta_S \frac{C_{S_i}}{f_{\text{ser}}} \tag{6}$$

$$e_{i,l} = \delta_L \frac{C_{L_i}}{f_{\text{user}}} \tag{7}$$

where δ_S (in W) and δ_L (in W) are energy consumption per time unit (one CPU cycle) of the MEC and the mobile device, respectively.

Denote E as the energy consumption of all the tasks, which can be written as

$$E = \sum_{i=1}^{i=Ns} e_{i,s} + \sum_{i=1}^{i=Nl} e_{i,l}. \tag{8}$$

B. Problem Formulation

We investigate the joint problem of the computation task partial offloading, offloading scheduling, and resource allocation while satisfying the transmission power constraint of the tasks. We aim to minimize the weighted sum of the execution delay and energy consumption.

If the task is offloaded to MEC for execution, the following two conditions must be met. First, all previous and current task data transmissions have been completed. Second, the CPU at the MEC is available for performing a new task. Thus, we denote $t_{S_i}^{\text{ready}}$ and $t_{S_i}^{\text{comp}}$ as the ready time and the completion time of the offloaded task S_i, respectively, which are given by

$$t_{S_i}^{\text{ready}} = \sum_{k=1}^{k=i} \frac{D_{S_k}}{R_{S_k}}, \quad i = 1, \ldots, N \tag{9}$$

$$t_{S_i}^{\text{comp}} = \begin{cases} t_{S_i}^{\text{ready}} + t_{S_i}^{\text{dispose}}, & i = 1 \quad (10) \\ \max\{t_{S_i}^{\text{ready}}, t_{S_{i-1}}^{\text{comp}}\} + t_{S_i}^{\text{dispose}}, & i > 1 \quad (11) \end{cases}$$

where $t_{S_i}^{\text{dispose}}$ is the mobile edge execution time of task S_i, which is given by (4), and $t_{S_{i-1}}^{\text{comp}}$ is the completion time of the previous being executed task S_{i-1} before the task S_i.

Consequently, the joint optimization of computation task partial offloading, offloading scheduling, and resource allocation problem are formulated as follows:

$$P1: \min_{\mathbb{X},\mathbb{S},\mathbb{L},\mathbb{P}} \max\left\{t_{S_{Ns}}^{\text{comp}}, t_{L_{Nl}}^{\text{comp}}\right\} + \eta \times \sum_{i=1}^{i=Ns} t_{S_i}^{\text{trans}} \times p_i + \eta_1 \times E \quad (12a)$$

$$\text{subject to: } 0 \le p_i \le p_{\max}, \quad 1 \le i \le N \quad (12b)$$

$$x_i \in \{0, 1\} \quad \forall T_i \in \Gamma \quad (12c)$$

$$\mathbb{S} = [S_1, S_2, \ldots S_{Ns}], S_i \neq S_j, i \neq j \quad \forall S_i, S_j \in \Gamma \quad (12d)$$

where $t_{S_{Ns}}^{\text{comp}}$ and $t_{L_{Nl}}^{\text{comp}}$ are the completion time for the task sets $\mathbb{S}$ and $\mathbb{L}$ being executed in MEC and mobile device, respectively, given by (3), (10), and (11). We denote η and η_1 as the energy consumption weight factor of transmission energy consumption and system executing energy consumption, respectively.

The objective function in (12a) is the weighted sum of the system execution delay and the energy consumption. We can treat η as a user experience parameter and can adjust it to pursue the balance of delay and energy saving. Equation (12b) defines the constraint of transmission power for the mobile device. Equation (12b) states that the task T_i is either offloaded to MEC server or executed locally. Equation (12d) is the scheduling sequence of the offloaded tasks, which means that S_i is the ith task being offloaded to the MEC server.

III. Two-Level Alternation Optimization Algorithm

The formulated problem $P1$ is a nonconvex mixed-integer optimization problem. In order to solve this problem, a two-level alternation method framework based on decomposition is proposed. The upper level is to solve the task offloading decision and offloading scheduling problem given the allocated transmission power using flow shop scheduling theory or greedy strategy [36]. The lower level is to obtain the sub-optimal power allocation with the partial offloading decision using convex optimization techniques [37].

We investigate the task offloading decision and offloading scheduling subproblem given the transmission power allocation vector $\mathbb{P}$. In the upper level, we propose the POJ algorithm in Section III-A, and the POC algorithm in Section III-B, respectively. In the lower level, the POSP algorithm is proposed in Section III-C. Finally, in Section III-D, we propose two JPOA iterative algorithms (JPOA-J and JPOA-C), which is based on POJ and POC to solve the task offloading decision and offloading scheduling in the upper level, respectively. The relationship of the proposed algorithms are shown in Fig. 2.

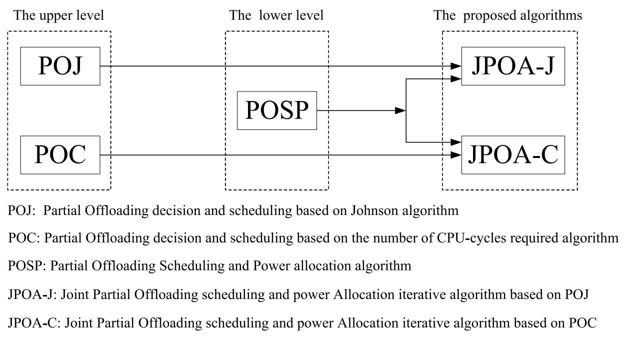

The upper level
The lower level
The proposed algorithms
POJ
POC
POSP
JPOA-J
JPOA-C
POJ: Partial Offloading decision and scheduling based on Johnson algorithm
POC: Partial Offloading decision and scheduling based on the number of CPU-cycles required algorithm
POSP: Partial Offloading Scheduling and Power allocation algorithm
JPOA-J: Joint Partial Offloading scheduling and power Allocation iterative algorithm based on POJ
JPOA-C: Joint Partial Offloading scheduling and power Allocation iterative algorithm based on POC

Fig. 2. Relationship of the proposed algorithms.

A. POJ Algorithm

The offloading decision and scheduling is handled based on flow shop scheduling theory in this section. The flow shop scheduling problem is a class of scheduling problems with a workshop or group shop and some independent jobs. The task offloading scheduling problem is a type of flow shop scheduling problem [19]. With a given transmission power vector $\mathbb{P}$, the optimal task POJ algorithm is described in Algorithm 1. Johnson algorithm is widely used to solve the flow shop scheduling problem. According to the relationship between the task transmission time and processing time, two disjoint subsets W and Q are obtained. The tasks in set W will be scheduled before those in Q, and tasks in W are scheduled in the ascending order of their transmission time. The tasks in set Q will be scheduled in the descending order of their processing time at the MEC server.

The basic idea of POJ is to maximize the CPU utilization of the MEC server. According to Johnson algorithm, all tasks are assigned to the MEC server or local mobile device based on their task transmission time and execution time.

In line 2 of Algorithm 1, two disjoint subsets W and Q are obtained based on Johnson algorithm and given by

$$W = \left\{T_i | \frac{D_i}{R_i} < \frac{C_i}{f_{\text{ser}}}, T_i \in \Gamma\right\} \quad (13)$$

$$Q = \left\{T_i | \frac{D_i}{R_i} \ge \frac{C_i}{f_{\text{ser}}}, T_i \in \Gamma\right\}. \quad (14)$$

In line 3, we sort W and Q according to the ascending order of the transmission time and the descending order of the executing time at the MEC server, respectively. Lines 4 and 5 mean that the respective first tasks from the set W and Q are taken out, and then add them to the set $\mathbb{S}^*$ and $\mathbb{L}^*$, respectively. The completion time for the tasks L_1^* and S_1^* is calculated according to (3) and (10) in line 7. Lines 9–17 show the steps to chose the tasks from the set W and Q, and add to the set $\mathbb{S}^*$ and $\mathbb{L}^*$. If the completion time of all the tasks in set $\mathbb{L}^*$ is larger than that of $\mathbb{S}^*$, then a new task is taken away from the Q and added to $\mathbb{S}^*$. Similarly, if the completion time of all the tasks in set $\mathbb{S}^*$ is larger than that of $\mathbb{L}^*$, then a new task is taken out from the W, and added to $\mathbb{L}^*$. Lines 18–22 are to take the remaining tasks in W or Q into the set M. Lines 24–33 illustrate that the tasks in the set M are added to the $\mathbb{S}^*$ or $\mathbb{L}^*$.

Algorithm 1 POJ Algorithm

```
Input: $\Gamma, f_{ser}, f_{user}, p_{max}, N$
Output: $\mathbb{S}^*$, $\mathbb{L}^*$, $X = \{x_i\}$
1: $\mathbb{S}^* = \varnothing$, $\mathbb{L}^* = \varnothing$
2: Obtain $W$ and $Q$ based on Johnson Algorithm
3: Sort $W$ and $Q$ according to the ascending ordering of the transmission time and the descending order of the executing time at the MEC server, respectively.
4: Set the index $t0 = 1$ and $t1 = 1$ for $W$ and $Q$
5: $\mathbb{S}^* = \mathbb{S}^* \cup W_{t0}$, $\mathbb{L}^* = \mathbb{L}^* \cup Q_{t1}$
6: Set the latest index $k0 = 1$ and $k1 = 1$ for $\mathbb{S}^*$ and $\mathbb{L}^*$
7: Calculate $t_{L_1^*}^{comp}$ and $t_{S_1^*}^{comp}$ according to (10) and (3)
8: Do
9: while $t_{L_{k1}^*}^{comp} \geq t_{S_{k0}^*}^{comp}$ and $t0 \leq |W|$
10:     $t0 = t0 + 1$, $k0 = k0 + 1$, $\mathbb{S}^* = \mathbb{S}^* \cup W_{t0}$, $x_{k0} = 1$
11:     Calculate $t_{S_{k0}^*}^{comp}$ according to (11)
12: end while
13: while $t_{S_{k0}^*}^{comp} \geq t_{L_{k1}^*}^{comp}$ and $t1 \leq |Q|$
14:     $t1 = t1 + 1$, $k1 = k1 + 1$, $\mathbb{L}^* = \mathbb{L}^* \cup Q_{t1}$, $x_{k1} = 0$
15:     Calculate $t_{L_{k1}^*}^{comp}$ according to (3)
16: end while
17: Until $t0 = |W|$ or $t1 = |Q|$
18: if $t0 = |W|$ then
19:     $M = \{w_i | w_i \in W, t0 \leq i \leq |W|\}$
20: else
21:     $M = \{q_i | q_i \in Q, t1 \leq i \leq |Q|\}$
22: end if
23: $mt = 1$
24: Do
25: if $t_{S_{k1}^*}^{comp} \geq t_{L_{k0}^*}^{comp}$ then
26:     $mt = mt + 1$, $k0 = k0 + 1$, $\mathbb{S}^* = \mathbb{S}^* \cup M_{mt}$, $x_{k0} = 1$
27:     Calculate $t_{L_{k1}^*}^{comp}$ according to (3)
28: end if
29: if $t_{L_{k0}^*}^{comp} \geq t_{S_{k1}^*}^{comp}$ then
30:     $mt = mt + 1$, $k1 = k1 + 1$, $\mathbb{L}^* = \mathbb{L}^* \cup M_{mt}$, $x_{k1} = 1$
31:     Calculate $t_{S_{k0}^*}^{comp}$ according to (11)
32: end if
33: Until $M = \varnothing$
```

B. POC Algorithm

In this section, the offloading decision and scheduling is handled based on greedy strategy. With a given transmission power vector $\mathbb{P}$, the algorithm for POC is described in Algorithm 2.

The key point of POC is to maximize the CPU utilization of the MEC server and edge mobile device simultaneously. All tasks are allocated to the local mobile device and the MEC server according to the required CPU cycles and scheduled based on Johnson algorithm. The difference between the completion time at mobile device and the MEC server can be minimized by using POC.

Line 2 in Algorithm 2 means that the computation tasks in set Γ are sorted according to the descending order of C_i. Line 4 shows that the completion time of the tasks L_1^* and S_1^* is calculated according to (3) and (10), respectively. Lines 5–14 show the procedure to chose the tasks from the set Γ and add to the set $\mathbb{S}^*$ and $\mathbb{L}^*$. If the task T_t is added to the $\mathbb{L}^*$, and the completion time of all the tasks in set $\mathbb{L}^*$ is less than that of $\mathbb{S}^*$, then the task T_t is added to $\mathbb{L}^*$ in lines 5–9. Similarly, if the task T_t is added to the S^*, and the completion time of all the tasks in set S^* is larger than that of L^*, then the task T_t is added to $\mathbb{S}^*$ in lines 10–14. Lines 15–24 mean that the task T_t is added to the $\mathbb{S}^*$ or $\mathbb{L}^*$ according to the comparison of the value of $t_{S_{k1}^*}^{\text{comp}}$ and $t_{L_{k0}^*}^{\text{comp}}$. Line 25 shows that two disjoint subsets F and J are obtained based on Johnson algorithm for the offloading set $\mathbb{S}^*$, $\mathbb{S}^* = F \cup J$, $F \cap J = \Phi$. The sets F and J are given by

$$F = \left\{ S_i | \frac{D_{S_i}}{R_{S_i}} < \frac{C_{S_i}}{f_{\text{ser}}}, S_i \in \mathbb{S}^* \right\} \tag{15}$$

$$J = \left\{ S_i | \frac{D_{S_i}}{R_{S_i}} \geq \frac{C_{S_i}}{f_{\text{ser}}}, S_i \in \mathbb{S}^* \right\}. \tag{16}$$

In line 26, the sets F and J are sorted according to the ascending ordering of the transmission time and the descending order of the executing time at the MEC server, respectively.

Algorithm 2 POC Algorithm

```
Input: $\Gamma, f_{ser}, f_{user}, p_{max}, N$
Output: $\mathbb{S}^*$ $\mathbb{L}^*$, $X = \{x_i\}$
1: $\mathbb{S}^* = \varnothing$ $\mathbb{L}^* = \varnothing$
2: sort the computation tasks in set $\Gamma$ according to the descending order of the CPU-cycles required for execution.
3: $\mathbb{S}^* = \mathbb{S}^* \cup T_1$, $\mathbb{L}^* = \mathbb{L}^* \cup T_1$, $t = 1$, $k1 = 1$, $k0 = 1$
4: Calculate $t_{L_1^*}^{comp}$ and $t_{S_1^*}^{comp}$ according to (10) and (3)
5: while $t_{S_{k1}^*}^{comp} \geq t_{L_{k0}^*}^{comp}$ and $t < N$
6: $x_t = 0$, $\mathbb{S}^* = \mathbb{S}^* - T_t$, $k0 = k0 + 1$, $t = t + 1$
7: Calculate $t_{L_{k0}^*}^{comp}$ according to (3)
8: $\mathbb{S}^* = \mathbb{S}^* \cup T_t$, $\mathbb{L}^* = \mathbb{L}^* \cup T_t$
9: end while
10: while $t_{S_{k1}^*}^{comp} < t_{L_{k0}^*}^{comp}$ and $t < N$
11: $x_t = 1$, $\mathbb{L}^* = \mathbb{L}^* - T_t$, $k1 = k1 + 1$, $t = t + 1$
12: Calculate $t_{S_{k1}^*}^{comp}$ according to (11)
13: $\mathbb{S}^* = \mathbb{S}^* \cup T_t$, $\mathbb{L}^* = \mathbb{L}^* \cup T_t$
14: end while
15: while $t < N$
16: $\mathbb{S}^* = \mathbb{S}^* \cup T_t$, $\mathbb{L}^* = \mathbb{L}^* \cup T_t$
17: if $t_{S_{k1}^*}^{comp} \geq t_{L_{k0}^*}^{comp}$, then
18:     $x_t = 0$, $\mathbb{S}^* = \mathbb{S}^* - T_t$, $k0 = k0 + 1$, $t = t + 1$
19:     Calculate $t_{L_{k0}^*}^{comp}$ according to (3)
20: else
21:     $x_t = 1$, $\mathbb{L}^* = \mathbb{L}^* - T_t$, $k1 = k1 + 1$, $t = t + 1$
22:     Calculate $t_{S_{k1}^*}^{comp}$ according to (11)
23: end if
24: end while
25: Obtain $F$ and $J$ based on Johnson Algorithm for set $\mathbb{S}^*$
26: Sort $F$ and $J$ according to the ascending ordering of the transmission time and the descending order of the executing time at the MEC server, respectively.
27: Obtain the new offloading scheduling $\mathbb{S}^* = [FJ]$.
```

C. Partial Offloading Scheduling and Power Allocation

In this section, the lower level problem is studied. We investigate the optimal transmit power allocation given the task scheduling and decision result. Both local and offloaded tasks vary with the transmission power p_i for each task. If the transmit power is determined, the offloading decision and scheduling can be obtained. If the offloading decision, scheduling, and transmission power are determined, the local execution delay and MEC completion time are all obtained. In this step, we solve the suboptimal transmit power problem given the offloading decision and scheduling. However, the objective function in $P1$ given $\mathbb{X}$ and $\mathbb{S}$ is nondifferentiable, which makes it difficult to solve. In order to overcome this issue, a set of auxiliary variables $t_{\widetilde{\mathbb{S}}} \triangleq [t_{\widetilde{S}_1}, \ldots, t_{\widetilde{S}_{Ns}}]$ is introduced. We denote $\widetilde{\mathbb{X}}$ as the offloading decision, $\widetilde{\mathbb{S}}$ as the execution sequence of the offloaded task vector $\mathbb{S}$, $t_{\widetilde{S}_i}$ as the completion time of the task $\widetilde{S}_i$, respectively, given the offloading decision and offloading scheduling in the upper level. We denote $\widetilde{\mathbb{L}}$ as the local processing task set. A modified version of the transmit power allocation problem is formulated as

$$P2: \min_{\mathbb{P}} \max\left\{t_{\widetilde{S}_{Ns}}, t_{\widetilde{L}_{Nl}}\right\} + \eta \times \sum_{k=1}^{k=Ns} p_{\widetilde{S}_k} \frac{D_{\widetilde{S}_k}}{R_{\widetilde{S}_k}} + \eta_1 \times \left(\sum_{k=1}^{k=Ns} \delta_{\widetilde{S}} \frac{C_{\widetilde{S}_k}}{f_{\text{ser}}} + \sum_{k=1}^{k=Nl} \delta_{\widetilde{L}} \frac{C_{\widetilde{L}_k}}{f_{\text{user}}}\right) \quad (17a)$$

$$\text{subject to: } 0 \le p_i \le p_{\max}, \quad 1 \le i \le N \quad (17b)$$

$$t_{\widetilde{S}_k} \ge \sum_{j=1}^{j=k} \frac{D_{\widetilde{S}_j}}{R_{\widetilde{S}_j}} + \frac{C_{\widetilde{S}_j}}{f_{\text{ser}}}, \quad k = 1, \ldots, Ns \quad (17c)$$

$$t_{\widetilde{S}_k} \ge t_{\widetilde{S}_{k-1}} + \frac{C_{\widetilde{S}_k}}{f_{\text{ser}}}, \quad k = 2, \ldots, Ns. \quad (17d)$$

We can see that $P2$ is a relaxed version of $P1$ given $\widetilde{\mathbb{X}}$, $\widetilde{\mathbb{S}}$, and $\widetilde{\mathbb{L}}$. In next lemma, we show why $P2$ is bound.

Lemma 1: If $(\mathbb{P}^*, t_{\widetilde{\mathbb{S}}}{}^*)$ is the optimal solution for $P2$, then $\mathbb{P}^*$ is optimal for $P1$ given scheduling decision $\widetilde{\mathbb{S}}$, and $t_{\widetilde{\mathbb{S}}}{}^*$ is the optimal execution delay for the offloading tasks.

Proof: Please refer to Appendix A. ■

$P2$ is still difficult to solve because it is not convex. However, we can transform it into a convex optimization problem using the change-of-variable technique. By introducing a set of new variables $\gamma_{\widetilde{S}_k} = 1/R_{\widetilde{S}_k}, k = 1, \ldots, Ns$, $P2$ can be rewritten as

$$P3: \min_{\gamma_{\widetilde{S}_k}} \max\left\{t_{\widetilde{S}_{Ns}}, t_{\widetilde{L}_{Nl}}\right\} + \sum_{k=1}^{k=Ns} \eta_1 \delta_{\widetilde{S}} \frac{C_{\widetilde{S}_k}}{f_{\text{ser}}} + \sum_{k=1}^{k=Nl} \eta_1 \delta_{\widetilde{L}} \frac{C_{\widetilde{L}_k}}{f_{\text{user}}} + \eta \times M \times \sum_{j=1}^{k=Ns} d_{\widetilde{S}_j} \gamma_{\widetilde{S}_j} \left(2^{\frac{1}{\omega \gamma_{\widetilde{S}_j}}} - 1\right) \quad (18a)$$

$$\text{subject to: } t_{\widetilde{S}_k} \ge t_{\widetilde{S}_{k-1}} + \frac{C_{\widetilde{S}_j}}{f_{\text{ser}}}, \quad k = 2, \ldots, Ns \quad (18b)$$

$$\gamma_{\widetilde{S}_k} \ge B, \quad k = 1, \ldots, Ns \quad (18c)$$

$$t_{\widetilde{S}_k} \ge \sum_{j=1}^{j=k} \frac{D_{\widetilde{S}_j}}{R_{\widetilde{S}_j}} + \frac{C_{\widetilde{S}_j}}{f_{\text{ser}}}, \quad k = 1, \ldots, Ns \quad (18d)$$

Algorithm 3 POSP Algorithm

```
Input: $\mathbb{S}^*, \mathbb{L}^*, f_{ser}, f_{user}, iter_{max}=50, I = 0, \epsilon = 10^{-7}, V_{new} = max\{t^{comp}_{S_{Ns}}, t^{comp}_{L_{Nl}}\} + \eta \times \sum_{i=1}^{i=Ns} t^{trans}_{S_i} \times p_i + \eta_1 \times E, V_{old} = V_{new} + 10$
Output: $\mathbb{S}^*, V_{new}$
1: while $V_{old} - V_{new} \ge \epsilon$ and $I \le iter_{max}$
2:     $I = I + 1$
3:     $V_{old} = V_{new}$
4:     Update $\mathbb{S}^*$ using Johnson Algorithm.
5:     Update $p_i$ with the task offloading scheduling decision $\mathbb{S}^*$ by solving $P3$.
6: end while
```

Algorithm 4 JPOA Algorithm

```
Input: $\Gamma, f_{ser}, f_{user}, \epsilon = 10^{-4}$
Output: $\mathbb{S}^*, \mathbb{L}^*, x$
1: $\Delta = 1$
2: while $\Delta \ge \epsilon$
3:     Update $\mathbb{S}^*$, $\mathbb{L}^*$ and $x_i$ using Algorithm 1 or Algorithm 2.
4:     $V_{old} = max\{t^{comp}_{S_{Ns}}, t^{comp}_{L_{Nl}}\} + \eta \times \sum_{i=1}^{i=Ns} t^{trans}_{S_i} * p_i + \eta_1 \times E$
5:     Sequential scheduling and power allocation are performed on $\mathbb{S}^*$ by calling Algorithm 3 to update $\mathbb{S}^*$ and obtain $V_{new}$.
6:     $\Delta = V_{old} - V_{new}$
7: end while
```

where M and B are given by

$$M = \frac{\eta N_0 \omega}{g_0 (L_0/L)^\theta} \quad (19)$$

$$B = \frac{1}{R(P_{\max})} = \frac{1}{\left(w \log_2\left(1 + \frac{g_0(L_0/L)^\theta p_{\max}}{N_0 w}\right)\right)}. \quad (20)$$

In the following lemma, we show $P3$ is a convex problem.

Lemma 2: $P3$ is a convex optimization problem.

Proof: Please refer to Appendix B. ■

The POSP algorithm is presented in Algorithm 3. The task offloading scheduling and the transmit power allocation vector will be updated in an alternating manner. The Johnson algorithm determines the optimal task offloading scheduling with a given transmit power allocation vector. The solution for $P3$ determines the optimal transmit power allocation with given task offloading scheduling. The value of the objective function in $P1$ decreases after each update.

D. Energy-Efficient and Low-Delay Optimization Algorithm

In this section, we focus on solving the formulated optimization problem $P3$. The JPOA iterative algorithm is proposed, which is described in Algorithm 4. There are two levels in the algorithm. The upper level is to solve the task offloading decision and offloading scheduling problem. The lower level is to obtain the suboptimal power allocation. In the computation offloading decision scheduling process, the mobile device chooses tasks to be offloaded through the binary decision variable x_i. The decision of the x_i depends on not only the delay of the tasks but also the transmission power and system energy consumption. In order to obtain the suboptimal solution of $P1$ in the process of solving the transmission power, we design the energy-efficient and low-delay offloading scheduling and power allocation algorithm to decide the offloading and execution order and transmit power allocation of the tasks.

JPOA algorithm is a two-layer iterative algorithm. The offloading decision and offloading scheduling are obtained in the upper layer using POC or POJ algorithm. The offloading task scheduling and transmission power allocation is solved in the lower layer using POSP algorithm. Through repeated offloading decision, offloading scheduling, and transmission power optimization, the minimum objective value is obtained.

In line 3 of Algorithm 4, the upper level problem is handled. The offloaded tasks set and its offloading order, the local processing tasks set, the binary decision variable are obtained by calling Algorithms 1 or 2. We denote JPOA-J and JPOA-C as JPOA using Algorithms 1 and 2 to update $\mathbb{S}^*$, $\mathbb{L}^*$, and x_i, respectively. Line 4 shows that the objective value of the last iteration is recorded. In line 5, the sequential scheduling order and transmission power allocation of the task are performed for $\mathbb{S}^*$ by calling Algorithm 3, and the new objective value is obtained. The objective value decreases after each iteration.

IV. Numerical Results

The proposed two-layer iterative approximate suboptimization algorithm is evaluated in this section, and compared with the following schemes.

1) *FOSA:* The full offloading algorithm, which is proposed in [19]. The N independent computing tasks are all offloaded to MEC execution.
2) *JPOA-J:* The proposed JPOA (JPOA-J) algorithm in this paper, which uses the POJ to solve the upper-level task offloading decision and scheduling. The joint tasks of offloading decision, offloading scheduling, and power allocation are considered. The objectives of energy efficiency and system execution delay are considered as well.
3) *JPOA-C:* The proposed JPOA (JPOA-C) algorithm in this paper, which uses the POC to solve the upper-level task offloading decision and scheduling. The joint tasks of offloading decision, offloading scheduling, and power allocation are considered. The objectives of energy efficiency and system execution delay are considered as well.
4) *JPOA-J-S:* A simple version of JPOA-J. JPOA-J runs just one time. It solves the task offloading decision and offloading scheduling problem using Algorithm 1, and obtains the suboptimal power allocation using Algorithm 3.
5) *JPOA-J-S-p_{max}:* A special form of JPOA-J-S. The transmission power of all being offloaded tasks is allocated with $p_{\max}$, and does not change at all.
6) *JPOA-C-S:* A simple version of JPOA-C. JPOA-C runs just once. It solves the task offloading decision and offloading scheduling problem using Algorithm 2, and obtains the suboptimal power allocation using Algorithm 3.
7) *JPOA-C-S-p_{max}:* A special form of JPOA-C-S. The transmission power of all being offloaded tasks is allocated with $p_{\max}$, and does not change at all.

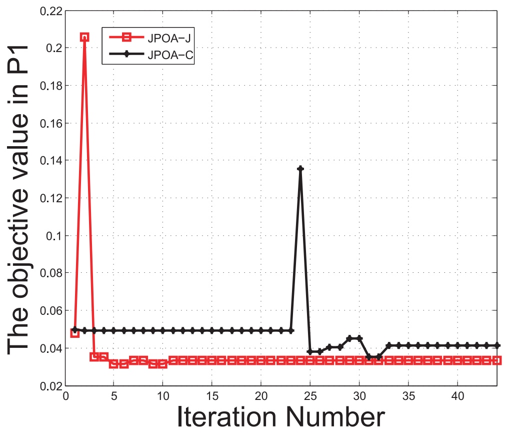


Fig. 3. Convergence of JPOA-J and JPOA-C.

In this section, the numerical parameters are the same as [19]. However, similar results can be observed if a different parameter configuration is used. We set $g_0 = -40$ dB, $L_0 = 1$ m, $L = 100$ m, $\theta = 4$, $w = 1$ MHz, $N_0 = -174$ dBm/Hz, $P_{\max} = 100$ mW, and $N = 40$. We assume that the D_i and C_i are uniformly distributed, i.e., $D_i \sim \text{Unif}[0, 2d_{\text{avg}}]$ and $C_i \sim \text{Unif}[0, 2c_{\text{avg}}]$, where $d_{\text{avg}} = 1000$ bits and $c_{\text{avg}} = 797.5$ cycles/bit. The CPU cycle frequency of the MEC server and mobile device are $f_{\text{ser}} = 3.3$ GHz and $f_{\text{user}} = 1.33$ GHz, respectively. Energy consumption within one CPU-cycle of the mobile device and MEC server are $\delta_L = 1.6541 \times 10^{-9}$ W and $\delta_S = 2.8788 \times 10^{-8}$ W, and the energy weighting factor system executes $\eta_1 = \eta/250$.

Fig. 3 shows the convergence of the JPOA-J and JPOA-C algorithms with $N = 40$ and $\eta = 30$. As can be seen from Fig. 3, the JPOA-J algorithm can obtain lower objective value than JPOA-C algorithm, and the convergence iterations number of JPOA-J is less than that of JPOA-C. The reason is that the number of tasks offloaded by POJ algorithm is more than POC algorithm. The peak of JPOA-C and JPOA-J is due to the change of offloading decision. The number of increasing offloading tasks leads to an increase in the objective value. Then the objective value is reduced by transmission power allocation and offloading scheduling algorithm.

Fig. 4 compares the task execution delay difference between MEC and local device of JPOA-J and JPOA-C algorithms, respectively. We can see from Fig. 4 that the task execution delay difference between MEC and local device of JPOA-J is less than that of JPOA-C. The reason is that the offloading decision algorithm POJ can obtain the optimal task distribution balancing. The number of convergence iterations of algorithm JPOA-J is less than that of algorithm JPOA-C because the number of tasks scheduled in each iteration of JPOA-J is more than that of JPOA-C, which makes the JPOA-J achieve convergence quickly.

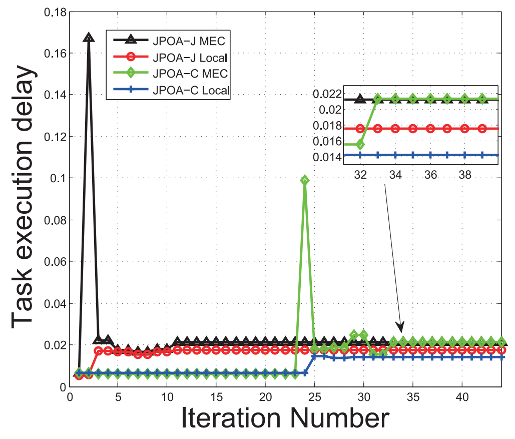


Fig. 4. Task execution delay in MEC and local device.

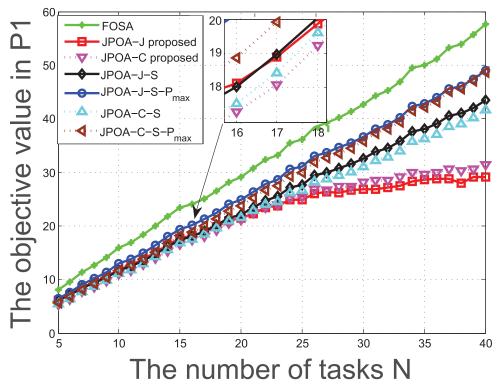


Fig. 6. Objective value with different N, $\eta = 15\,000$.

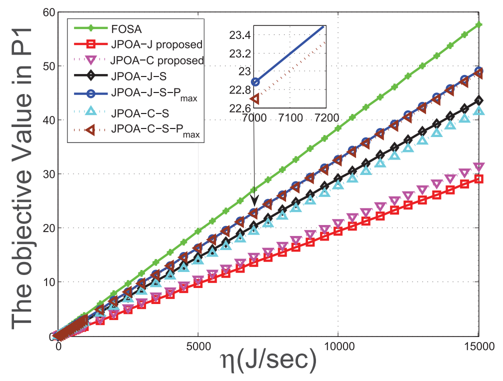


Fig. 5. Objective value with different η, $N = 40$.

Fig. 5 compares the weighted sum of the system execution delay and energy consumption performance of the different optimization algorithms with different η. The energy consumption includes both the energy cost spending on computation and data transmission. In this figure, the objective function in $P1$ of all seven algorithms increases with η. The system execution delay and energy consumption of FOSA are higher than that of the other six algorithms in all cases, especially with large values of η. This is because the MEC server is with large δ_S. It is worth noting that the partial offloading can save more energy for any case compared with full offloading. From the two curves of JPOA-J-S and JPOA-J-S-$p_{\max}$, we see a significant performance improvement achieved by JPOA-J-S increasing with η, since the suboptimal transmit power can be obtained. From the two curves of JPOA-J and JPOA-J-S, we see a significant performance improvement achieved by JPOA-J since the dynamic iteration has been used, and the optimal offloading decision and scheduling and power allocation are obtained by JPOA-J. With the increase of η, JPOA-C achieves better performance than JPOA-C-S since the dynamic iteration strategy has been used by JPOA-C. JPOA-C-S obtains better performance than JPOA-C-S-$p_{\max}$ since the suboptimal transmit power can be obtained by JPOA-C-S. The performance of JPOA-J is better than JPOA-C because the offloading decision and scheduling algorithm POJ called in JPOA-J and POC algorithm used in JPOA-C, respectively, is used to obtain the optimal offloading scheduling. JPOA-C-S performs better than JPOA-J-S because JPOA-J-S has more offloading tasks after the first offloading decision. JPOA-J-S-P$_{\max}$ and JPOA-C-S-P$_{\max}$ are two kinds of partial offloading algorithms, both of which have an offloading decision. But the transmission power is not optimized, and thus the optimization performance is limited. The optimization effect of JPOA-J-S-$P_{\max}$ is slightly better than that of JPOA-C-S-$P_{\max}$ because POJ classifies all tasks according to offloading transmission time and offloading execution time, and divides all tasks into two sets suitable for local execution and suitable for offloading execution. Therefore, the number of offloading tasks of POJ algorithm is greater than that of POC algorithm. Then the energy consumption of JPOA-J-S-$P_{\max}$ is large, so the performance of JPOA-C-S-$P_{\max}$ is superior to that of JPOA-J-S-$P_{\max}$.

Fig. 6 shows the influence of different number of tasks on the objective function in $P1$. In this figure, the objective function in $P1$ of all seven algorithms increases with N. The weighted sum of FOSA is higher than that of the other six algorithms in all cases. When $N \leq 17$, JPOA-J-S performs better than JPOA-J due to fewer available tasks for being scheduled in JPOA-J. With the increasing of N, the JPOA-J gradually outperforms JPOA-J-S. JPOA-J-S-$P_{\max}$ has worse performance than JPOA-J and JPOA-J-S since the optimal transmission power is not obtained. The objective value of JPOA-J-S-$P_{\max}$ and JPOA-C-S-$P_{\max}$ are lower than FOSA due to offloading decision. The objective value of JPOA-J-S and JPOA-C-S is further reduced due to transmission power optimization and offloading scheduling. When the number of tasks is small, the performance of JPOA-C-S-$P_{\max}$, JPOA-C-S, and JPOA-C is better than that of JPOA-J-S-$P_{\max}$, JPOA-J-S, and JPOA-J, respectively. Because there are not enough tasks for POJ to handle. The performance of JPOA-J is better than JPOA-C with the increasing of task number N. But the performance of JPOA-J-S is still lower than JPOA-C-S with fewer number of tasks, because POJ will offload more tasks at the first offloading decision and need to make multiple decisions to select the optimal offloading strategy. There are fewer tasks leading to fewer offloading decisions for JPOA-J, and the number of offloaded tasks of POJ is higher than that of POC.

Fig. 7 compares the system delay of the seven algorithms with different η. When the energy weighting factor $\eta = 0$,

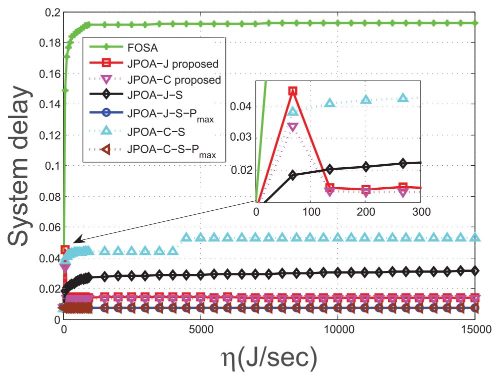


Fig. 7. System delay with different η, $N = 40$.

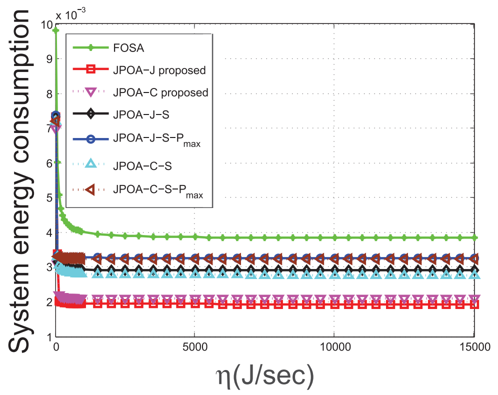


Fig. 9. System energy consumption with different η, $N = 40$.

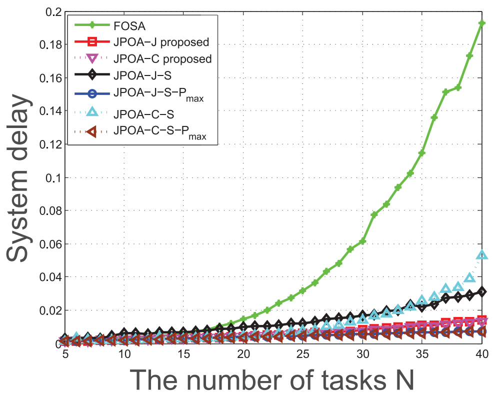


Fig. 8. System delay with different N, $\eta = 15\,000$.

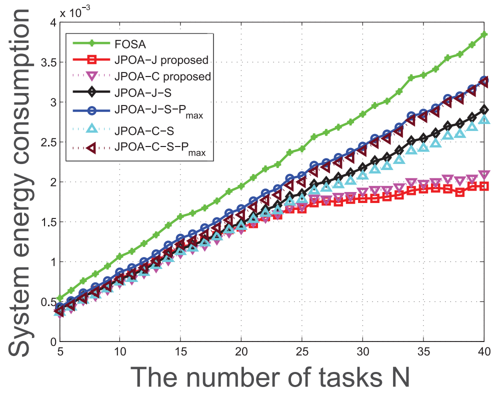


Fig. 10. System energy consumption with different N, $\eta = 15\,000$.

without considering energy optimization, all tasks are with the maximum transmission power. The minimum system delay is obtained by those seven algorithms. With the increase of η, the system delay of JPOA-J and JPOA-C increases first, then decreases and becomes be stable. The reason for the decreasing phase is that the offloading decision changes the offloading execution tasks set. When the energy weighting factor η is small, the main goal is to reduce system delay, and the system delay increases with η. When η is large, the main goal is to reduce energy consumption, and then the offloading scheduling is not changed at all. In this figure, the system delay of FOSA is higher than other six algorithms in all cases. It shows that partial offloading is superior to full offloading. The delay of JPOA-J and JPOA-C is slightly higher than that of JPOA-J-S-$p_{\max}$ and JPOA-C-S-$p_{\max}$, and the delay of JPOA-J and JPOA-J-S-$p_{\max}$, JPOA-C and JPOA-C-S-$p_{\max}$ is lower than that of JPOA-J-S and JPOA-C-S. This shows the superiority of JPOA-J and JPOA-C. The minimum delay is obtained by JPOA-J-S-$p_{\max}$ and JPOA-C-S-$p_{\max}$ since the maximum transmission power is allocated. When $\eta = 4500$, JPOA-C-S system delay increases with different η, the reason is that the η increases and transmission power continues to optimize. This leads to the order of task execution in the MEC server changing and the system delay increasing.

Fig. 8 indicates the system delay with different number of tasks N. The system delay of seven algorithms increase with N. FOSA increases with the number of tasks, while the delay of other six partial offloading algorithms increasing less, which shows the advantage of partial offloading in system delay. JPOA-J-S-$p_{\max}$ and JPOA-C-S-$p_{\max}$ reach the lowest delay because the max transmission powers are allocated. JPOA-J-S-$p_{\max}$ and JPOA-C-S-$p_{\max}$ uses a little more energy than JPOA-J and JPOA-C. This demonstrates that a lot of energy saving can be achieved at the expense of latency performance degradation. As the number of tasks N increases, the difference in delay performance between JPOA-J and JPOA-J-S-$p_{\max}$, JPOA-C and JPOA-C-S-$p_{\max}$ becomes obvious gradually. From the four curves of JPOA-J and JPOA-J-S, and JPOA-C and JPOA-C-S, the lower system delay are obtained by JPOA-J and JPOA-C with the increase of the number of tasks, and the effect of offloading decision scheduling becomes more and more obvious.

Fig. 9 presents the energy consumption of the seven algorithms under different η conditions. When $\eta = 0$, the energy consumption of each algorithm reaches maximum. Since $\eta = 0$, the transmit power is not optimized for the seven algorithms, i.e., $p_i = p_{\max}$. When $\eta \neq 0$, energy consumption is optimized, and the system energy consumption is rapidly reduced and converges. The system energy consumption of FOSA is higher than other six algorithms. This is because the δ_L is much smaller than the δ_S, and some tasks are being executed locally. As we can be seen from the figure, JPOA-J and JPOA-C obtain the best performance.

Fig. 10 compares the energy consumption of the seven algorithms under different number of tasks. As we can see from the figure, when $N \leq 17$, JPOA-J gets higher energy consumption

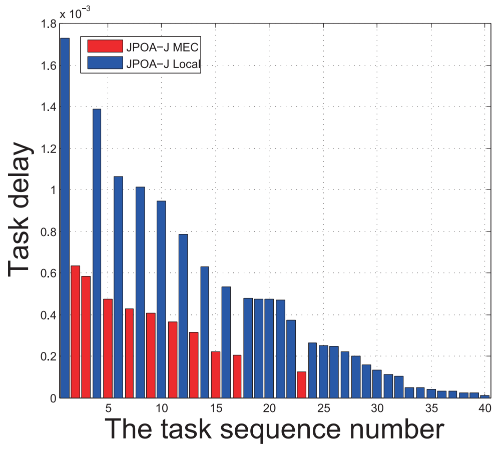


Fig. 11. Task execution delay of each task for JPOA-J, $N = 40$.

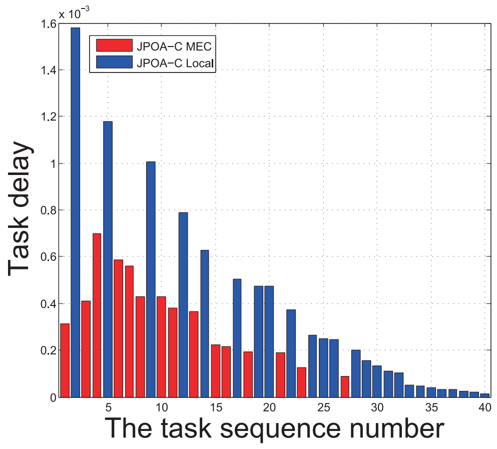


Fig. 12. Task execution delay of each task for JPOA-C, $N = 40$.

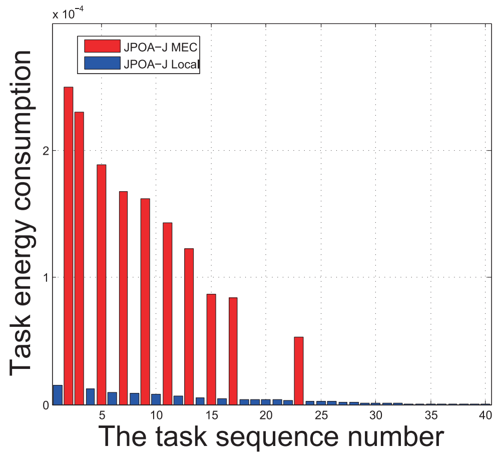


Fig. 13. Task energy consumption of each task for JPOA-J, $N = 40$.

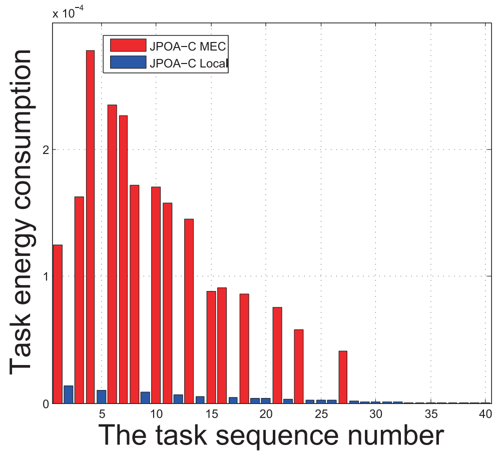


Fig. 14. Task energy consumption of each task for JPOA-C, $N = 40$.

than JPOA-J-S, because there are fewer available tasks for scheduling in JPOA-J. When $N > 17$, with the increase of N, the energy consumption of JPOA-J is lower than that of JPOA-J-S.

Figs. 11 and 12 show the execution delay of each task for JPOA-J and JPOA-C, respectively. The number of tasks is 40 ($N = 40$). We denote the "JPOA-J MEC" and "JPOA-J Local" as the execution delay of each task in MEC sever and local mobile device, respectively. As we can see from Fig. 11, the offloaded tasks execution delay in MEC is lower than the tasks execution delay in local. The reason is that the CPU processing frequency of MEC is higher than that of local device. The task execution delay decreases with the task execution sequence for MEC. The reason is that the task with large execution delay is offloaded to MEC with high priority based on POJ. After the offloading decision has been done, the large optimization space is provided for transmission power allocation and offloading scheduling. The local task execution delay decreases with the task execution sequence as well. Because the tasks with large execution delay are scheduled with high priority. As we can see from Fig. 12, the offloaded tasks execution delay in MEC is lower than the tasks execution delay in local as well. The task execution delay increases, and then decreases with the task execution sequence for MEC. The reason is that the task T_i with little D_i and large C_i are offloaded to MEC based on POC, and those tasks with little data and large the number of CPU processing cycles are processed first based on the scheduling principle of flow shop.

Figs. 13 and 14 show the energy consumption of each task for JPOA-J and JPOA-C, respectively. The number of tasks is 40 ($N = 40$). Denoting the JPOA-J MEC and JPOA-J Local as the energy consumption of each task in MEC and local, respectively. The energy consumption includes CPU processing task energy consumption and task transmission energy consumption, in which the CPU processing task energy consumption is accounted for the major part. The local task energy consumption is much less than the MEC server task energy consumption because the server consumes more energy per CPU cycle than local device, that is $\delta_S > \delta_L$. As we can be seen from Figs. 13 and 14, the energy consumption decreases with the task execution sequence for MEC because the CPU cycle frequency of the MEC server is δ_S, and the CPU processing task energy consumption equals the execution delay of MEC multiplied the δ_S. The local energy consumption decreases with the task execution sequence. The reason is that the execution delay of each local tasks decreases with the task execution sequence, and the CPU cycle frequency of the local device is δ_L.

V. Conclusion

In this paper, we investigated the task partial offloading scheduling and resource allocation mechanisms in MEC systems. We formulated an energy-efficient and low-delay POSP problem in a single-user MEC system, which is a nonconvex mixed-integer optimization problem. In order to minimize the execution delay and energy consumption, we proposed a two-level alternation method framework based on decomposition. The upper level was to solve the task offloading decision and offloading scheduling, and the lower level was to obtain the suboptimal power allocation. In the upper level, the POJ algorithm and the POC algorithm were proposed. In the lower level, the POSP algorithm was proposed. Two JPOA (JPOA-J and JPOA-C) iterative algorithms based on POJ and POC were proposed. In future, the mobile devices with the scalable clock frequency of the CPU will be investigated.

Appendix A
Proof of the Lemma 1

With the optimal transmit power allocation vector $\mathbb{P}^*$, we construct a new timing sequence $t'_{\widetilde{S}_k}$ for $P2$ as follows:

$$t'_{\widetilde{S}_k} = \begin{cases} \dfrac{D_{\widetilde{S}_k}}{R_{\widetilde{S}_k}} + \dfrac{C_{\widetilde{S}_k}}{f_{\text{ser}}}, & k = 1 \quad (21) \\ \max\left\{ t_{\widetilde{S}_{k-1}}, \displaystyle\sum_{j=1}^{k} \frac{D_{\widetilde{S}_j}}{R_{\widetilde{S}_j}} \right\} + \dfrac{C_{\widetilde{S}_j}}{f_{\text{ser}}}, & k \geq 2 \quad (22) \end{cases}$$

where $t'_{\widetilde{S}_{Ns}}$ is no larger than any of the feasible $t_{\widetilde{S}_{Ns}}$ for $P2$ given $\mathbb{P}^*$, $t'_{\widetilde{S}_{Ns}} = (t_{\widetilde{S}_{Ns}})^*$ and $(\mathbb{P}^*, t'_{\widetilde{\mathbb{S}}})$ is also optimal for $P2$. Therefore, by substituting $\mathbb{P}^*$ into $P1$ given scheduling decision $\widetilde{\mathbb{S}}$, the value of the objective function $V_{P1}(\mathbb{P}^*, \widetilde{\mathbb{S}})$ is equal to $V_{P2}{}^*$, where $V_{P2}{}^*$ is the optimal value for $P2$. Since $P2$ is a relaxation of $P1$ given the task offloading scheduling decision, i.e., $V_{P1}{}^* \geq V_{P2}{}^*$, where $V_{P1}{}^*$ is the optimal value for $P1$ given $\widetilde{\mathbb{S}}$, we have $V_{P1}(\mathbb{P}^*, \widetilde{\mathbb{S}}) = V_{P1}{}^*$, where $\mathbb{P}^*$ is optimal value for $P1$ given $\widetilde{\mathbb{S}}$. Therefore, we can solve $P2$ to obtain the optimal transmit power allocation vector.

Appendix B
Proof of the Lemma 2

We denote $\chi(\gamma) \triangleq \gamma(2^{(1/\omega\gamma)} - 1)$. The second-order derivative of $\chi(\gamma)$ is obtained by

$$\frac{d^2\chi(\gamma)}{d\gamma^2} = 2^{\frac{1}{\omega\gamma}} \frac{\ln^2 2}{\omega^2\gamma^2} \geq 0 \quad \forall\gamma \geq 0 \tag{23}$$

so $\chi(\gamma)$ is a convex function of $\gamma(\gamma > 0)$. Since the objective function is a summation of convex and linear functions, and the constraints are linear, $P3$ is a convex optimization problem.

References

[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[2] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[3] Y. Bi *et al.*, "Mobility support for fog computing: An SDN approach," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 53–59, May 2018.

[4] C. Mouradian *et al.*, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 416–464, 1st Quart., 2018.

[5] Z. Kuang, G. Liu, G. Li, and X. Deng, "Energy efficient resource allocation algorithm in energy harvesting-based D2D heterogeneous networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 557–567, Feb. 2019.

[6] X. Cao, L. Liu, Y. Cheng, and X. Shen, "Towards energy-efficient wireless networking in the big data era: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 303–332, 1st Quart., 2018.

[7] P. He, S. Zhang, L. Zhao, and X. S. Shen, "Energy-efficient power allocation with individual and sum power constraints," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5353–5366, Aug. 2018.

[8] P. He, S. Zhang, L. Zhao, and X. Shen, "Multichannel power allocation for maximizing energy efficiency in wireless networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5895–5908, Jul. 2018.

[9] Z. Kuang, Z. Chen, J. Pan, and D. Sajjadi, "Joint optimization of spectrum access and power allocation in uplink OFDMA CR-VANETs," *Wireless Netw.*, vol. 25, no. 1, pp. 1–11, 2019.

[10] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 1st Quart., 2017.

[11] T. Ling and S. He, "Multi-user computation offloading in mobile edge computing: A behavioral perspective," *IEEE Netw.*, vol. 32, no. 1, pp. 48–53, Jan./Feb. 2018.

[12] F. Zhang, G. Liu, X. Fu, and R. Yahyapour, "A survey on virtual machine migration: Challenges, techniques and open issues," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1206–1243, 2nd Quart., 2018.

[13] T. G. Rodrigues, K. Suto, H. Nishiyama, N. Kato, and K. Temma, "Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration," *IEEE Trans. Comput.*, vol. 67, no. 9, pp. 1287–1300, Sep. 2018.

[14] L. Zhao and J. Liu, "Optimal placement of virtual machines for supporting multiple applications in mobile edge networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6533–6545, Jul. 2018.

[15] S. Zhang *et al.*, "Cooperative edge caching in user-centric clustered mobile networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 8, pp. 1791–1805, Aug. 2018.

[16] Z. Tan, F. R. Yu, X. Li, H. Ji, and V. C. M. Leung, "Virtual resource allocation for heterogeneous services in full duplex-enabled SCNS with mobile edge computing and caching," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1794–1808, Feb. 2018.

[17] J. Gao, L. Zhao, and L. Sun, "Probabilistic caching as mixed strategies in spatially-coupled edge caching," in *Proc. IEEE 29th Biennial Symp. Commun.*, 2018, pp. 1–5.

[18] S. Zhang *et al.*, "Towards fresh and low-latency content delivery in vehicular networks: An edge caching aspect," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2018, pp. 1–6.

[19] Y. Mao, J. Zhang, and K. B. Letaief, "Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2017, pp. 1–6.

[20] N. Janatian, I. Stupia, and L. Vandendorpe, "Optimal resource allocation in ultra-low power fog-computing SWIPT-based networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2018, pp. 1–6.

[21] M. Qin *et al.*, "Power-constrained edge computing with maximum processing capacity for IoT networks," *IEEE Internet Things J.*, to be published.

[22] Z. Ning, P. Dong, X. Kong, and F. Xia, "A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things," *IEEE Internet Things J.*, to be published.

[23] S. Mu, Z. Zhong, D. Zhao, and M. Ni, "Joint job partitioning and collaborative computation offloading for Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1046–1059, Feb. 2019.

[24] H. Guo, J. Liu, and J. Zhang, "Efficient computation offloading for multi-access edge computing in 5G HetNets," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

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

[26] E. Meskar, T. D. Todd, D. Zhao, and G. Karakostas, "Energy aware offloading for competing users on a shared communication channel," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 87–96, Jan. 2017.

[27] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.

[28] H. Guo and J. Liu, "Collaborative computation offloading for multi-access edge computing over fiber-wireless networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4514–4526, May 2018.

[29] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[30] Y. Yang *et al.*, "MEETS: Maximal energy efficient task scheduling in homogeneous fog networks," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 4076–4087, Oct. 2018.

[31] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, Sep. 2018.

[32] H. Guo, J. Liu, J. Zhang, W. Sun, and N. Kato, "Mobile-edge computation offloading for ultradense IoT networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4977–4988, Dec. 2018.

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

[34] Z. Wang, Z. Zhong, D. Zhao, and M. Ni, "Vehicle-based cloudlet relaying for mobile computation offloading," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11181–11191, Nov. 2018.

[35] Y. Gu, Z. Chang, M. Pan, L. Song, and Z. Han, "Joint radio and computational resource allocation in IoT fog computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7475–7484, Aug. 2018.

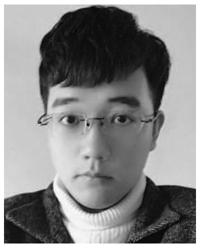
[36] H. Emmons and G. Vairaktarakis, *Flow Shop Scheduling: Theoretical Results, Algorithms, and Applications*. New York, NY, USA: Springer, 2013.

[37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

Zhufang Kuang (M'12) received the M.Sc. degree in computer science from the National University of Defense Technology, Changsha, China, in 2006, and the Ph.D. degree in computer science from Central South University, Changsha, in 2012.

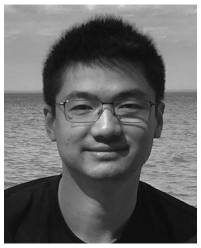
He was a Post-Doctoral Researcher with the School of Software, Central South University. From 2015 to 2016, he was a Visiting Scholar/Professor with the University of Victoria, Victoria, BC, Canada. He is currently a Full Professor with the Department of Computer Science, Central South University of Forestry and Technology, Changsha, and is also a Researcher Fellow with the Key Laboratory of Intelligent Information Perception and Processing Technology (Hunan Province), Zhuzhou, China. His current research interests include wireless communications and networking, mobile edge computing, and optimization algorithm and its application.

Dr. Kuang was the Vice Chair of CCF YOCSEF CHANGSHA from 2018 to 2019. He is a Senior Member of the CCF and a member of the CCF Network and Data Communications Council and the ACM.



Linfeng Li received the B.Eng. degree in communication engineering from the Lanzhou University of Technology, Lanzhou, China, in 2016. He is currently pursuing the M.Sc. degree in information and communication engineering at the Central South University of Forestry and Technology, Changsha, China.

His current research interests include mobile edge computing, optimization algorithms, and their application.



Jie Gao (S'13–M'17) received the B.Eng. degree in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2007, and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 2009 and 2014, respectively.

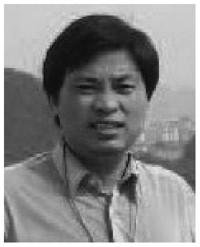
He is currently a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, and the Department of Electrical, Computer, and Biomedical Engineering, Ryerson University, Toronto, ON, Canada. His current research interests include the application of game theory and mechanism design for distributed decision making in communication networks and performance optimization of multiuser systems.

Dr. Gao was a recipient of the Natural Science and Engineering Research Council of Canada Post-Doctoral Fellowship Award.

Lian Zhao (S'99–M'03–SM'06) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2002.

In 2003, she joined the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, ON, Canada, and become a Professor in 2014. She earned early tenure and became an Associate Professor in 2006. Her current research interests include wireless communications, radio resource management, edge computing and caching, cognitive radio and cooperative communications, and optimization for complicated systems.

Dr. Zhao was a recipient of the Best Land Transportation Paper Award from the IEEE Vehicular Technology Society in 2016, the Top 15 Editor in 2015 for the IEEE TRANSACTION ON VEHICULAR TECHNOLOGY, the Best Paper Award from the 2013 International Conference on Wireless Communications and Signal Processing and Best Student Paper Award (with her student) from Chinacom in 2011, and the Canada Foundation for Innovation New Opportunity Research Award in 2005. She has been the Co-Chair for the IEEE ICC 2018 Wireless Communication Symposium, the Workshop Co-Chair for IEEE/CIC ICCC 2015, the Local Arrangement Co-Chair for IEEE VTC Fall 2017 and IEEE Infocom 2014, and the Co-Chair for the IEEE Global Communications Conference 2013 Communication Theory Symposium. She served as a committee member of the Natural Science and Engineering Research Council of Canada Discovery Grants Evaluation Group for Electrical and Computer Engineering from 2015 to 2018. She is a Licensed Professional Engineer in the Province of Ontario and a Senior Member of the Communication and Vehicular Society.



Anfeng Liu received the M.Sc. and Ph.D. degrees in computer science from Central South University, Changsha, China, in 2002 and 2005, respectively.

He was a Post-Doctoral Researcher with the School of Electronic Science and Technology, National University of Defense Technology, Changsha. From 2009 to 2012, he was a Visiting Scholar/Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. He is a Full Professor with the School of Computer Science and Engineering, Central South University. His current research interests include wireless sensor networks and mobile edge computing.

Dr. Liu is also a member (E200012141M) of the China Computer Federation.