Received 23 December 2019; revised 15 January 2020; accepted 30 January 2020. Date of publication 4 February 2020; date of current version 20 February 2020.

Digital Object Identifier 10.1109/OJCOMS.2020.2971647

Joint Offloading and Charge Cost Minimization in Mobile Edge Computing

KEHAO WANG [1], ZHIXIN HU [1], QINGSONG AI [1], YI ZHONG [1], JIHONG YU [2], PAN ZHOU [3] (Senior Member, IEEE), LIN CHEN [4], AND HYUNDONG SHIN [5] (Senior Member, IEEE)

[1] School of Information Engineering, Wuhan University of Technology, Wuhan 4300700, China

[2] School of Information and Electronics, Beijing Institute of Technology, Beijing 100811, China

[3] School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China

[4] School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China

[5] Department of Electronic Engineering, Kyung Hee University, Seoul 02447, South Korea

CORRESPONDING AUTHOR: Y. ZHONG (e-mail: zhongyi@whut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672395, Grant 61972448, Grant 61603283, and Grant 61911540481, in part by the Fund of Hubei Key Laboratory of Inland Shipping Technology under Grant NHHY2019004, in part by the Fundamental Research Funds for the Central Universities under Grant 2018-IB-020, and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) under Grant 2019K2A9A2A06024389.

ABSTRACT Mobile edge computing (MEC) brings a breakthrough for Internet of Things (IoT) for its ability of offloading tasks from user equipments (UEs) to nearby servers which have rich computation resource. 5G network brings a huge breakthrough on transmission rate. Together with MEC and 5G, both execution delay of tasks and time delay from downloading would be shorter and the quality of experience (QoE) of UEs can be improved. Considering practical conditions, the computation resource of an MEC server is finite to some extent. Therefore, how to prevent the abuse of MEC resource and further allocate the resource reasonably becomes a key point for an MEC system. In this paper, an MEC system with multi-user is considered where a base station (BS) with an MEC server, which can not only provide computation offloading service but also data cache service. Especially, we take the charge for both data transmission and task computation as one part of total cost of UEs, and then explore a joint optimization for downlink resource allocation, offloading decision and computation resource allocation to minimize the total cost in terms of the time delay and the charge to UEs. The proposed problem is formulated as a mixed integer programming (MIP) one which is NP-hard. Therefore, we decouple the original problem into two subproblems which are downlink resource allocation problem and joint offloading decision and computation resource allocation problem. Then we address these two subproblems by using convex and nonconvex optimization techniques, respectively. An iterative algorithm is proposed to obtain a suboptimal solution in polynomial time. Simulation results show that our proposed algorithm performs better than benchmark algorithms.

INDEX TERMS Mobile edge computing, offloading decision, resource allocation, charge to UEs.

I. INTRODUCTION

THE EMERGENCE of IoT brings enormous challenges to existing technologies because IoT allows thousands of UEs including smartphones, Pads and intelligent wearable devices connected to Internet simultaneously [1]. Moreover, some novel applications requiring high computation capability and high energy consumption are on demands, such as massive multiplayer online game, virtual reality (VR), augmented reality (AR) and face recognition. Meanwhile, these applications are also sensitive to latency, which puts higher requirement for UEs, especially CPU and battery. However, it is not easy to meet these requirements

for UEs because of their stringent equipment-size constraint.

To address these challenges, mobile cloud computing (MCC) is considered as a possible solution for IoT. The main idea of MCC is to allow UEs offload their computation intensive tasks to a powerful centralized cloud, so that the delay and energy of tasks execution of UEs could be reduced sharply [2]. Generally, the cloud center is too far from UEs, so the UEs would suffer huge uploading delay and cause extra uploading energy consumption. To deal with this problem, MEC is put forward with advantage of deploying servers with abundant computation resource on the edge of network which are close to UEs [3]. As the emergence of 5G network brings a huge breakthrough on transmission rate, MEC-enable IoT was proved as a promising solution to reduce the delay of task and save the energy of UEs in some IoT scenarios [4]. In [5], MEC was used to reduce the delay of tasks execution and improve computation efficiency in an unmanned aerial vehicle (UAV). Reference [6] showed that MEC was suitable for vehicular networks because it significantly reduced delay and average system cost.

Moreover, the theoretical speed of 5G has exceeded the rate of a mechanical hard disk [7]. In this case, rate of UEs reading local files may be slower than downloading files from MEC, which implies that reading only memory (ROM) of UEs may not need to existed any more. Such that UEs must download what they need all the time, which puts higher requirements on downlink bandwidth. Meanwhile, some UEs need to offload their high computation required tasks to MEC, which put higher demand on computation resource of MEC servers. Therefore, offloading decision, resource allocation or other policy should be considered to improve the QoE of UEs since resource of an MEC system is finite.

Offloading decision is to decide how to offload tasks. Basically, an offloading decision has three choices: no offloading, full offloading and partial offloading to servers [8]. Specifically, no offloading means the tasks are executed locally and no data is uploading to MEC servers. Full offloading means a whole task is offloaded to an MEC server. However, it is not suitable for all UEs offloading their tasks to MEC because of the constraints of bandwidth resource and computation resource of MEC. Partial offloading allows a task to be cut into two parts, one part executed locally, and the other part executed at the MEC server. According to [9], partial offloading is very challenging because of the dependency of offloadable components from tasks. Thus, most researches consider the binary offloading policy which allows each UE to either execute task locally or offload task to a remote MEC server [10].

On the other hand, resource allocation should be considered since the resource on an MEC server is limited. Inappropriate resource allocation policies could cause additional overhead in terms of energy consumption and latency, i.e., the delay of uploading task data to the MEC server and execution time [11]. Resource allocation involves bandwidth, power and computation resource allocation. For bandwidth resource allocation, many papers only consider uplink bandwidth and minimize uploading time [12]. In fact, the downlink resource, i.e., bandwidth and power of MEC, is also a significant factor for reducing latency if some task-related data needs to be download from MEC the server.

For computation resource allocation, it should be noticed that the computation resource of MEC servers is not infinite although much richer than that of UEs. Moreover, extra cost would be paid for the maintenance of MEC servers. Existing tasks offloading policies only consider the delay and energy consumption of uploading data to MEC servers, and the aftermath is that those tasks offloaded to MEC servers would abuse the computation resource and further increase server maintenance cost of the MEC servers. Such as in [13], the MEC server would keep full load even though the number of UEs is very small, which will incur extra burden on MEC server but has little promotion on QoE of UEs.

In this paper, we consider that UEs should pay cost to use the MEC server, that is UEs will be charged for occupying computation and bandwidth resource in task offloading. Specifically, we jointly explore the offloading decision, resource allocation including downlink bandwidth, downlink power and computation resource allocation, and the charge for computation and bandwidth resource. We consider data transmission delay and task execution delay as one part of cost in offloading tasks to the MEC server, and the charge to UEs as another part of cost, and try to minimize the total cost of all UEs. Specially, we set the price of computation resource according to computation rate because the purpose of offloading is to reduce execution time of a task. In this context, the main contributions of our paper can be summarized as follows:

1) We consider the charge as a new cost of offloading and model the minimization of total cost in the weight-sum of tasks completion time and the charge. Then, we formulate this problem as a Joint Optimization for Downlink Resource Allocation, Offloading Decision and Computation Resource Allocation (JODOC) to minimize the total cost of all UEs.
2) We decompose the JODOC problem into two subproblems. One is Downlink Resource Allocation (DRA) problem, and another is Joint Offloading Decision and Computation Resource Allocation (JOCRA) problem.
3) We address DRA and JOCRA by using convex and nonconvex optimization techniques, respectively. Combining the solutions of these two problems, we propose a novel low-complexity algorithm to solve the original JODOC problem and obtain a suboptimal solution.
4) Simulation results show our proposed algorithm performs better than three benchmark algorithms.

The rest of this paper is organized as follows. In Section II, we show some related studies on tasks offloading. In Section III, we propose a network model and formulate the joint optimization for downlink bandwidth allocation,

offloading decision and computation resource allocation problem. And we decouple this problem and propose a suboptimal solution in Section IV. The simulation results are given in Section V. Finally, we conclude this paper in Section VI.

II. RELATED WORK

Ever since MEC is put forward, the cost in terms of delay and energy consumption is regarded as an important parameter of evaluating the quality of an MEC system. Most researches aim to minimization of execution delay or minimization of energy consumption while satisfying execution delay constraint or trade-off between energy consumption and execution delay.

For the minimization of execution delay, in [14], the authors proposed a low-complexity online Lyapunov optimization-based dynamic computation offloading (LODCO) algorithm to minimize execution delay and assumed that the UEs exploited energy harvesting techniques in [15]. But the harvested energy is not enough to provide all the energy consumption when the UEs are too many. In [16], the authors aimed to minimize the transmission delay by a multilayer data flow processing system including edge devices, access point, MEC servers and cloud center. In [17], the authors considered partial offloading and proposed an iterative heuristic MEC resource allocation (IHRA) algorithm to reduce the time delay of MEC system. In [18], the authors proposed a two-step radio and computing resources allocation scheme for minimizing total processing completion time.

For the minimization of energy consumption while satisfying execution delay constraint, in [19], the authors proposed an offline strategy to minimize the average energy consumed by all the user terminals to process their mobile applications under average delay constraints. In [20], the authors aimed to minimize the network-level energy consumption in heterogenous network by jointly considering computation resources, latency requirements and power consumption.

In previous works, most papers aimed to minimize trade-off between energy consumption and execution delay. In [21], the authors proposed a framework considering physical resource block (PRB) along with computation resource allocation. In [22], the authors considered to share the computation results so that the computation resource would be saved. In [23], the authors modeled the MEC system as a Stackelberg game, and then let the MEC servers and UEs compete for their own best profits. In [24], the authors proposed a novel online SBS peer offloading framework to maximize the long-term system performance and meanwhile satisfy the energy constraints. In [25], the authors allowed multi-user to choose a better node from multi-node and avoid too many UEs choosing the same edge node by using a non-cooperative exact potential game (EPG). In [26], the author jointly solved the transmission power control problem and tasks offloading problem to get minimization of the trade-off between energy consumption and execution delay.

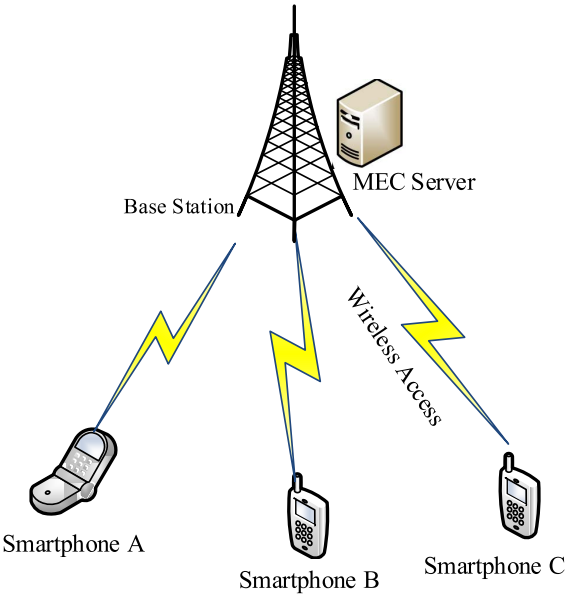

FIGURE 1. Network model.

There are also same researches that considered the charge problem. In [27], the authors assumed UEs will be charged for transmitting computation input data and computation resource that different from local resource. In [23], the authors used Stackelberg game to find the minimization of cost which is defined as time delay and payment for MEC servers. In [28], the authors aimed to maximize the profit of the network operators considering the cost of maintain resource of MEC and the profit from UEs.

These papers all above, however, didn't jointly consider offloading decision and the radio and computation resource allocation with charge. Meanwhile, the joint downlink resource allocation, i.e., bandwidth and power resource allocation, is not considered. So, in this paper we aim to minimize the total cost of UEs in terms of delay and charge, by jointly considering downlink resource, offloading decision and computation resource allocation.

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. NETWORK MODEL

In this paper, we consider a multiple-user, one-server MEC system which provides both computation offloading service and content cache service, as shown in Fig. 1. Denote the set of UEs as $\mathcal{N} = \{1, 2, 3, \ldots, N\}$. Each UE n has a computation task to be offloaded and a content cache task to be downloaded. For UE n, let C_n denote CPU cycles required to accomplish a computation task, D_n^u denote the data size of a computation task, and D_n^d denote the cache date size to be downloaded, i.e., pictures and videos that UE n needs. Formally, the task of UE n can be expressed as

$$I_n = \left\{C_n, D_n^u, D_n^d\right\}, \quad \forall n \in \mathcal{N} \tag{1}$$

We define the offloading decision profile as $\boldsymbol{x} = \{x_n, \ \forall n \in \mathcal{N}\}$, herein $x_n = 1$ indicates the computation task of UE n will be offloaded to MEC server while $x_n = 0$ shows it will be executed locally.

B. COMMUNICATION MODEL

We denote $\mathcal{N}_{\mathrm{f}} = \{n \mid x_n = 1,\ \forall n \in \mathcal{N}\}$ as the set of UEs which tend to offload their computation tasks to the MEC server. Then $N_{\mathrm{f}} = |\mathcal{N}_{\mathrm{f}}| = \sum_{n=1}^{N} x_n$. We assume that the total uplink spectral bandwidth is w^u which is divided into N_{f} sub-channels, and UE n is allocated with $w_n^u = w^u/N_{\mathrm{f}}$.

We denote p_n^u and H_n as the fixed transmit power of UE n and the channel gain between UE n and the MEC, respectively. N_0 is the variance of complex white Gaussian channel noise power. Accordingly, the transmission rate between UE n and the MEC is given by

$$r_n^u = w_n^u \log_2\left(1 + \frac{p_n^u H_n}{N_0}\right) \tag{2}$$

Then we can get the uploading time of task D_n^u as follows

$$t_n^u = \frac{D_n^u}{r_n^u},\ \forall n \in \mathcal{N}_{\mathrm{f}} \tag{3}$$

When UE n asks for downloading task D_n^d from the MEC server, the total time cost of this task mainly consists of three parts. The first part is the delay in sending the request from UE n to the MEC server which is neglected because of the short data size of requesting packet. The second part is the time required to search for specific data D_n^d on MEC server, which we denote by t_n^w. The third part is the downloading delay denoted by t_n^d.

We assume that the downlink bandwidth, w^d, different from w^u, is allocated dynamically according to the number of downloaded tasks. The downlink resource allocation policy is defined as $\boldsymbol{w} = \{w_n^d \mid n \in \mathcal{N}\}$. Then the downloading delay for the content cache task D_n^d can be obtained as follows

$$t_n^d = \frac{D_n^d}{r_n^d},\ \forall n \in \mathcal{N} \tag{4}$$

where r_n^d is the downloading data rate given by

$$r_n^d = w_n^d \log_2\left(1 + \frac{p_n^d H_n}{N_0}\right) \tag{5}$$

where p_n^d is the transmission power of MEC to UE n.

Meanwhile, the downlink transmission power allocation policy is defined as $\boldsymbol{p} = \{p_n^d \mid n \in \mathcal{N}\}$ and the MEC server has limited transmission power, which is denoted by p_m. Therefore, $\boldsymbol{p}$ should meet the total power limit

$$\sum_{n \in \mathcal{N}} p_n^d \le p_m. \tag{6}$$

C. OFFLOADING MODEL

For local execution, we denote f_n^l as local computation resource of UE n. Let t_n^l be the local execution time of computation task which can be expressed as

$$t_n^l = \frac{C_n}{f_n^l} \tag{7}$$

As for offloading, the MEC server allocates computation resource f_n to offloading task from UE n. We define the computation resource allocation policy as $\boldsymbol{f} = \{f_n | n \in \mathcal{N}_{\mathrm{f}}\}$. f_m is defined as the maximum computation capacity of the MEC server. Therefore, all the f_n allocated to UEs should meet the computation resource constraint as follows

$$\sum_{n \in \mathcal{N}_{\mathrm{f}}} f_n \le f_m \tag{8}$$

Let t_n^{exe} be the execution time of the computation task at the MEC server, which can be expressed as

$$t_n^{exe} = \frac{C_n}{f_n},\ \forall n \in \mathcal{N}_{\mathrm{f}} \tag{9}$$

We ignore the delay of transmitting the result back because the output data has smaller size and MEC sever has a higher transmission power.

D. CHARGE POLICY

In this part, we assume that UEs will be charged for not only task computation but also data transmission including both uploading and downloading data.

As for data charge, the operators, i.e., China Telecom and T-mobile, charge UEs for transmission flow rather than data rate, so we assume that the price is a function of data size. And operators will offer some fixed data plans (e.g., 20GB per month for xx$). We consider that the fixed data plan is more like a discount policy, so we set that UEs who have fixed data plan are charged by a lower price. Therefore, the price is set as e_d^n per bit and the price will be different depends on different UE n. As for task computation charge, the price is related to computation rate because the purpose of offloading is to reduce execution time. Therefore, we set that the price of task computation is a function of computation resource allocated to UE n, i.e., f_n, and the unit price being charged is defined as e_c per $cycle/s$.

Then the charge to UE n can be expressed as

$$G_n = x_n\left(e_c f_n + e_d^n D_n^u\right) + e_d^n D_n^d,\ \forall n \in \mathcal{N}. \tag{10}$$

E. PROBLEM FORMULATION

In this paper, we aim to minimize the total cost in terms of time delay and the charge.

The total time cost for UEs n can be given by

$$t_n = x_n\left(t_n^u + t_n^{exe}\right) + (1 - x_n)t_n^l + t_n^w + t_n^d \tag{11}$$

We introduce a function J_n to evaluate the weight cost of time delay and charge to the UE, which is defined as follows

$$J_n = \lambda_n^t t_n + \lambda_n^c G_n \tag{12}$$

where $\lambda_n^t \in [0, 1]$ and $\lambda_n^c \in [0, 1]$ $(\lambda_n^t + \lambda_n^c = 1)$ are the weight parameters for the cost of time delay and the charge to UE n, respectively.

Thus the problem can be formulated as follows

$$\min_{\boldsymbol{x},\boldsymbol{w},\boldsymbol{p},\boldsymbol{f}} \sum_{n \in \mathcal{N}} J_n\left(x_n, w_n^d, p_n^d, f_n\right)$$
$$\text{s.t.} \quad C1 : x_n = \{0, 1\},\ \forall n \in \mathcal{N}$$

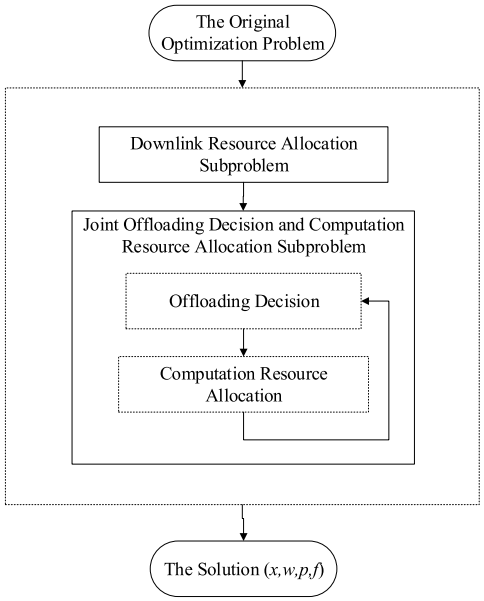

FIGURE 2. Proposed framework for solving the problem (13).

$$
\begin{aligned}
&C2: w_n^d > 0,\ \forall n \in \mathcal{N}, \sum_{n\in\mathcal{N}} w_n^d \le w^d \\
&C3: p_n^d > 0,\ \forall n \in \mathcal{N}, \sum_{n\in\mathcal{N}} p_n^d \le p_m \\
&C4: f_n > 0,\ \forall n \in \mathcal{N}_{\mathrm{f}}, \sum_{n\in\mathcal{N}_{\mathrm{f}}} f_n \le f_m
\end{aligned} \tag{13}
$$

The constraint $C1$ shows that each task can be either executed locally or offloaded to MEC server. $C2$ guarantees the dowlink bandwidth allocated to tasks is valid and the sum of them does not exceed the total bandwidth resource. $C3$ guarantees the downlink bandwidth resource allocated to the UEs is valid and the sum of them does not exceed the total transmission power. $C4$ plays the similar role in computation resource allocation.

Considering that $\boldsymbol{x}$ is binary, $\boldsymbol{w}$, $\boldsymbol{p}$ and $\boldsymbol{f}$ are continuous, this problem is an MIP one, which is NP-hard [29].

IV. DECOUPLED OPTIMIZATION

Considering the complexity of MIP, we decompose the original problem into two subproblems. One is DRA problem with all UEs, and another is JOCRA problem. First, we solve the DRA problem by bisection method to obtain the downlink bandwidth and power strategy profile. Second, we propose an iterative algorithm to solve the JOCRA problem and obtain the offloading decision and computation resource allocation profile. Last, we combine the results of DRA and JOCRA and proposed a polynomial time algorithm to obtain a global solution. The framework of our proposed algorithm is shown in Fig. 2.

A. DOWNLINK RESOURCE ALLOCATION

In this part, we aim to analyze and solve the DRA problem. Fixing $\boldsymbol{x}$ and $\boldsymbol{f}$ and removing all the parts not related to $\boldsymbol{w}$ and $\boldsymbol{p}$, we could obtain the following

$$
\begin{aligned}
\min_{\boldsymbol{w},\boldsymbol{p}}\ & \sum_{n\in\mathcal{N}} \frac{\lambda_n^t D_n^d}{r_n^d} \\
\text{s.t.}\ \ & C2: w_n^d > 0,\ \forall n \in \mathcal{N}, \sum_{n\in\mathcal{N}} w_n^d \le w^d \\
& C3: p_n^d > 0,\ \forall n \in \mathcal{N}, \sum_{n\in\mathcal{N}} p_n^d \le p^m
\end{aligned} \tag{14}
$$

Notice that the constraints C2 and C3 are convex and the domain of w_n^d and p_n^d are $0 < w_n^d < p_m$ and $0 < w_n^d < w^d$, respectively. By simple analysis, we know that the first order derivatives of the objective function with respect to w_n^d and p_n^d are less than zero, so the problem can be transformed to a convex one when one of these two variables is fixed.

Therefore, to solve problem (14), we assume that $\boldsymbol{p}$ is fixed and give optimal solution of $\boldsymbol{w}$, as shown in Lemma 1.

Lemma 1: For (14), when $\boldsymbol{p}$ is fixed, the optimal solution of $\boldsymbol{w}$ is

$$
w_n^{d'} = \frac{w^d \sqrt{q_n}}{\sum_{n\in\mathcal{N}} \sqrt{q_n}} \tag{15}
$$

where $q_n = \frac{\lambda_n^t D_n^d}{\log_2(1+\frac{p_n^d H_n}{N_0})}$.

Proof: First we can write the Lagrangian function of (14) as

$$
L\left(w_n^d, \alpha\right) = \sum_{n\in\mathcal{N}} \frac{q_n}{w_n^d} + \alpha\left(\sum_{n\in\mathcal{N}} w_n^d - w^d\right) \tag{16}
$$

where α is the nonnegative Lagrange multiplier corresponding to constraint C2.

By setting the first-order derivative of $L(w_n^d, \alpha)$ with respect to w_n^d and α to zero, respectively, we have

$$
\begin{cases}
\alpha - \frac{q_n}{(w_n^d)^2} = 0 \\
\sum_{n\in\mathcal{N}} w_n^d - w^d = 0
\end{cases}
$$

Solving these two equations, we obtain the optimal downlink bandwidth allocation in (15). ■

Combining (15) and $\lambda_n^t D_n^d / \log_2(1 + \frac{p_n^d H_n}{N_0}) = q_n$ into (14), our problem is turned into following formulation.

$$
\begin{aligned}
\min_{\boldsymbol{p}}\ & \sum_{n\in\mathcal{N}} \left(w^d \sqrt{q_n} \sum_{n\in\mathcal{N}} \sqrt{q_n} \right) \\
\text{s.t.}\ \ & p_n^d > 0,\ \forall n \in \mathcal{N}, \quad \sum_{n\in\mathcal{N}} p_n^d = p_m
\end{aligned} \tag{17}
$$

Lemma 2: The optimal solution of (17) is the same with

$$
\begin{aligned}
\min_{\boldsymbol{p}}\ & \sum_{n\in\mathcal{N}} \sqrt{q_n} \\
\text{s.t.}\ \ & p_n^d > 0,\ \forall n \in \mathcal{N}, \quad \sum_{n\in\mathcal{N}} p_n^d = p_m
\end{aligned} \tag{18}
$$

Proof: First, we rewrite (17) as follows

$$
\min_{\boldsymbol{p}} \quad w^d \left(\sum_{n\in\mathcal{N}} \sqrt{q_n} \right)^2
$$

It is obvious that when $\sum_{n\in\mathcal{N}}\sqrt{q_n}$ achieves the minimum, $w^d(\sum_{n\in\mathcal{N}}\sqrt{q_n})^2$ will also achieve the minimum. Therefore, our objective function becomes (18). ■

Problem (18) is convex because the first order derivative of the objective function is greater than zero, and its constraint is also convex. Therefore, this problem can be transformed into an unconstrained problem by Lagrange multiplier method.

Then we can give the Lagrange function of (18) as follows

$$L\left(p_n^d,\nu\right)=\sum_{n\in\mathcal{N}}\sqrt{\frac{\lambda_n^t D_n^d}{\log_2\left(1+\frac{p_n^d H_n}{N_0}\right)}}+\nu\left(\sum_{n\in\mathcal{N}}p_n^d-p_m\right) \tag{19}$$

where ν is the nonnegative Lagrange multiplier.

In general, this Lagrange function can obtain its optimal solution by setting the derivatives of the function with respect to p_n^d and ν equal to zero. That is

$$\begin{cases}-\frac{\partial\sqrt{q_n}}{\partial p_n^d}=\nu\\ \sum_{n\in\mathcal{N}}p_n^d=p_m\end{cases} \tag{20}$$

Specially, we set

$$\begin{aligned}\varphi\left(p_n^d\right)&=-\frac{\partial\sqrt{q_n}}{\partial p_n^d}\\&=\frac{H_n\sqrt{\lambda_n^t D_n^d}}{2(\ln 2)\left(\log_2\left(1+\frac{p_n^d H_n}{N_0}\right)\right)^{\frac{3}{2}}\left(N_0+p_n^d H_n\right)}\end{aligned} \tag{21}$$

Due to the logarithmic barrier, (20) has no analytical solution in general [30]. Therefore, we design an iterative algorithm to solve (20) in the following.

Notice that in the domain $0<p_n^d<f_m$, $\varphi(p_n^d)$ is a monotonically decreasing function. For fixed UEs number $\mathcal{N}$ and total transmission power p_m, the optimal ν is also fixed. Therefore, the closer $\varphi(p_n^d)$ is to ν, the better solution we will get. Moreover, from (20), we can derive that for two different UEs i and j, $\varphi(p_i^{d*})-\varphi(p_j^{d*})=0$ on the optimal solution. This implies that the closer $\varphi(p_i^{d*})-\varphi(p_j^{d*})$ is to zero, the better resource allocation for UEs i and j will be obtained.

Therefore, we propose to decompose this downlink power allocation problem for N UEs to $\lfloor\frac{N}{2}\rfloor$ downlink power allocation problems for two UEs. Each of these downlink power allocation problems for two UEs can be solved by a low complexity bisection method.

First, the total power resource is equally divided into N part, and each part is $p_0=p_m/N$. Then any two UEs are chosen to form a group and re-allocated power resource $2p_0$. If N is an odd number, the remaining UE will be allocated with p_0. The bisection method is used to allocate the power resource $2p_0$ between the two UEs in the same group. By evaluating $\varphi(p_i^{d*})-\varphi(p_j^{d*})$ in each iteration, we will finally obtain a suboptimal solution p_n^{d*}. The detail is shown in Algorithm **1**.

Algorithm 1 Bisection Method for Downlink Power Allocation

1: Given control threshold ξ
2: $p_0=p_m/N$, $i=1$
3: **repeat**
4: **if** $i+1>N$ **then**
5: $p_i^{d*}=p_0$
6: **else**
7: Initialize $p'=0$, $p''=2p_0$
8: **repeat**
9: $p_i^{d*}=(p'+p'')/2$, $p_{i+1}^{d*}=2p_0-p_i^{d*}$
10: Get $\varphi(p_i^{d*})$ and $\varphi(p_{i+1}^{d*})$ using (21)
11: **if** $\varphi(p_i^{d*})-\varphi(p_{i+1}^{d*})>0$ **then**
12: Set $p'=p_i^{d*}$
13: **else**
14: Set $p''=p_i^{d*}$
15: end **if**
16: **until** $p''-p'<\xi$
17: $p_i^{d*}=(p'+p'')/2$, $p_{i+1}^{d*}=2p_0-p_i$
18: end **if**
19: $i=i+2$
20: **until** $i>N$
21: **Output** $\boldsymbol{p}^*=\{p_n^{d*}\mid n\in\mathcal{N}\}$

In Algorithm **1**, the lines 8–16 (bisection) will be executed $\lceil\log_2\frac{2p_0}{\xi}\rceil$ times and the lines 3–20 will be executed $\lceil\frac{N}{2}\rceil$ times, where ξ is the control threshold. Thus the complexity of this algorithm is $\mathcal{O}(\frac{N}{2}\log_2\frac{2p_m}{\xi N})$.

Finally, we get the downlink power resource allocation profile $\boldsymbol{p}^*$. For the fixed $\boldsymbol{p}^*$, we can obtain the optimal downlink resource allocation policy $\boldsymbol{w}$ using (15).

B. JOINT OFFLOADING DECISION AND COMPUTATION RESOURCE ALLOCATION

When the downlink resource allocation policy $\boldsymbol{w}$ and $\boldsymbol{p}$ are fixed, the subproblem JOCRA could be rewritten as

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{f}}\ &\sum_{n\in\mathcal{N}}x_n\left(Z_n^f-Z_n^l\right)+\sum_{n\in\mathcal{N}}Z_n^l\\ \text{s.t.}\ &x_n=\{0,1\},\ \forall n\in\mathcal{N}\\ &f_n>0,\ \forall n\in\mathcal{N}_{\mathrm{f}},\quad\sum_{n\in\mathcal{N}_{\mathrm{f}}}f_n\le f_m\end{aligned} \tag{22}$$

where $Z_n^f=\lambda_n^t t_n^u+\lambda_n^t t_n^{exe}+\lambda_n^c e_c f_n+\lambda_n^c e_d^n D_n^u$, which describes the cost when computation task is offloaded to remote MEC server while $Z_n^l=\lambda_n^t t_n^l$ describes the cost when computation task is processed locally.

The second term of (22) could be ignored because it can be computed in advance. Thus our goal is to minimize

$$\sum_{n\in\mathcal{N}}Z_n^{pro}=\sum_{n\in\mathcal{N}}x_n\left(Z_n^f-Z_n^l\right)$$

For any UE n, when the cost of the computation task executed locally is larger than executed at the MEC server, i.e., $Z_n^{pro}<0$, the task will benefit from offloading. In another

word, the value of Z_n^{pro} shows how much a task is suitable for offloading. Moreover, the tasks of different UEs have different D_n^u and C_n. This implies that wether a task is suitable for offloading is only related to D_n^u and C_n, which is shown in the following proposition.

Proposition 1:

- When those UEs have the same uploading data, i.e., $D_n^u = D^u,\ \forall n \in \mathcal{N}$, those UEs with larger required CPU cycles C_n, tend to offload their task to MEC server.
- When UEs have the same required CPU cycles, i.e., $C_n = C,\ \forall n \in \mathcal{N}$, those UEs with smaller uploading data D_n^u, tend to offload their task to MEC server.

Proof: Let's rewrite Z_n^{pro} as

$$Z_n^{pro}\left(D_n^u, C_n\right) = aD_n^u + bC_n + c \tag{23}$$

where $a = \lambda_n^t/r_n + \lambda_n^c e_d^n > 0$, $b = \lambda_n^t(1/f_n - 1/f_n^l)$ and $c = \lambda_n^c e_c f_n > 0$.

Then we have $\frac{\partial Z_n^{pro}}{\partial D_n^u} = a > 0$ and $Z_n^{pro}(D_n^u, C_n)$ increases with D_n^u.

According to principle of computation offloading that offloading a task to a richer computation resource server, we know that $f_n > f_n^l$ no matter what computation allocation strategy f_n is applied. Then we can get

$$\frac{\partial Z_n^{pro}}{\partial C_n} = b = \lambda_n^t\left(1/f_n - 1/f_n^l\right) < 0$$

which implies that $Z_n^{pro}(D_n^u, C_n)$ decrease with C_n.

Therefore we conclude the proposition. ∎

According to Proposition 1, we know that an UE n with smaller D_n^u and bigger C_n is more suitable for offloading task to the MEC server.

We assume that all the tasks are offloaded to the MEC server at first, i.e., $x_n = 1, n \in \mathcal{N}$. These tasks will be allocated with computation resource f_n which can be obtain by (25). Then we could easily get Z_n^{pro} for each UE n with fixed f_n. For these tasks with $Z_n^{pro} > 0$ which mean these tasks are not suitable for offloading in this scenario, we should choose the task which has the greatest Z_n^{pro} to executed locally, i.e., changing $x_n = 1$ to $x_n = 0$. So now we have a new offloading decision profile $\boldsymbol{x}'$. Then use the new $\boldsymbol{x}'$ to update computation resource allocation profile $\boldsymbol{f}'$ using (25) and the new $\boldsymbol{f}'$ will update Z_n^{pro} which lead to updating $\boldsymbol{x}'$. We will do this iteration until all the offloaded tasks get profits, i.e., $Z_n^{pro} < 0, n \in \mathcal{N}_\mathrm{f}$, or all the tasks are executed locally.

In the above iterating process, we should give the optimal computation resource allocation $\boldsymbol{f}$ for a fixed decision profile $\boldsymbol{x}$ in every iteration. Eliminating these parts which are not with respect to f_n, we rewrite the objective function when we get fixed $\boldsymbol{w}$ and $\boldsymbol{x}$ as follows

$$\begin{aligned} \min_{f} \ & \sum_{n\in\mathcal{N}_\mathrm{f}} Z_n^f = \frac{\lambda_n^t C_n}{f_n} + \lambda_n^c e_c f_n + d \\ \text{s.t.} \ & f_n > 0,\ \forall n \in \mathcal{N}_\mathrm{f}, \quad \sum_{n\in\mathcal{N}_\mathrm{f}} f_n \le f_m \end{aligned} \tag{24}$$

where $d = \lambda_n^t t_n^u + \lambda_n^c e_d^n D_n^u$.

We know (24) is a convex function because its second-order derivative with respect to f_n is greater than zero on constraint $f_n > 0,\ \forall n \in \mathcal{N}_\mathrm{f}$. Therefore, we could obtain the optimal solution of (24) by using Lagrangian multiplier method as showed in Lemma 3.

Lemma 3: The optimal solution of (24) is

$$f_n^{opt} = \begin{cases} \sqrt{\frac{\lambda_n^t C_n}{\lambda_n^c e_c}}, & \sum_{n\in\mathcal{N}_\mathrm{f}} f_n^{nor} \le f_m \\ \frac{f_m k_n}{\sum_{n\in\mathcal{N}_\mathrm{f}} k_n}, & \sum_{n\in\mathcal{N}_\mathrm{f}} f_n^{nor} > f_m \end{cases} \tag{25}$$

where $k_n = \sqrt{\lambda_n^t C_n}$, $f_n^{nor} = \sqrt{\frac{\lambda_n^t C_n}{\lambda_n^c e_c}}$.

Proof: There are two cases for the second constraint of (24) in which the optimal solution satisfies

$$\sum_{n\in\mathcal{N}_\mathrm{f}} f_n < f_m$$

or

$$\sum_{n\in\mathcal{N}_\mathrm{f}} f_n = f_m$$

For the first case $\sum_{n\in\mathcal{N}_\mathrm{f}} f_n < f_m$, this problem will turn to a simple convex problem with no constraint since the first constraint can be ignore. And problem will get its minimum as long as each Z_n^f gets minimum. So the computation resource allocation will be

$$f_n^{opt} = \sqrt{\frac{\lambda_n^t C_n}{\lambda_n^c e_c}} \tag{26}$$

As for the second case, i.e., $\sum_{n\in\mathcal{N}_\mathrm{f}} f_n = f_m$, we could rewrite (24) as

$$\begin{aligned} \min_{f} \ & \sum_{n\in\mathcal{N}_\mathrm{f}} \frac{\lambda_n^t C_n}{f_n} + \lambda_n^c e_c f_n \\ \text{s.t.} \ & f_n > 0,\ \forall n \in \mathcal{N}_\mathrm{f} \\ & \sum_{n\in\mathcal{N}_\mathrm{f}} f_n = f_m \end{aligned} \tag{27}$$

The problem (27) is a typical constrained optimization problem. Then we can write (27) to Lagrange function as follows

$$L(f_n, \mu) = \sum_{n\in\mathcal{N}_\mathrm{f}} \left(\frac{\lambda_n^t C_n}{f_n} + \lambda_n^c e_c f_n\right) + \mu\left(\sum_{n\in\mathcal{N}_\mathrm{f}} f_n - f_m\right) \tag{28}$$

where μ is the nonnegative Lagrange multiplier.

By solving this optimization problem, we can get the optimal solution of (27), which is expressed as

$$f_n^{opt} = \frac{f_m\sqrt{\lambda_n^t C_n}}{\sum_{n\in\mathcal{N}_\mathrm{f}} \sqrt{\lambda_n^t C_n}} \tag{29}$$

∎

C. OVERALL ALGORITHM

In this part, an overall algorithm is proposed which combines DRA and JOCRA as follows.

Algorithm 2 Joint Optimization for Downlink Bandwidth Allocation, Offloading Decision and Computation Resource Allocation (JODOC)

Initialize $t = 0$, $x_n(0) = 1, n \in \mathcal{N}$, $\mathcal{N}_\mathrm{f} = \mathcal{N}$
Get Z_n^l
Get $\boldsymbol{f}(t)$ using (25)
Obtain downlink power allocation $\boldsymbol{p}$ by Algorithm **1**
Obtain downlink bandwidth allocation $\boldsymbol{w}$ by (15)
repeat
  $t = t + 1$
  $n^* = \arg\max_n \{Z_n^f - Z_n^l > 0, n \in \mathcal{N}_\mathrm{f}\}$
  $\mathcal{N}_\mathrm{f} = \mathcal{N}_\mathrm{f} - \{n^*\}$
  Update $\boldsymbol{x}(t)$ by setting $x_{n^*}(t) = 0$
  Update $\boldsymbol{f}(t)$ using (25)
  Update $Z_n^f = \lambda_n^t t_n^u + \lambda_n^t t_n^{exe} + \lambda_n^c e_c f_n$
    $+\lambda_n^c e_d^n D_n^u, \quad \forall n \in \mathcal{N}_\mathrm{f}$
until $Z_n^f - Z_n^l \leq 0, \forall n \in \mathcal{N}_\mathrm{f}$ or $\mathcal{N}_\mathrm{f} = \phi$
Output $(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{p}, \boldsymbol{f})$

In Algorithm **2**, we choose one UE each time to change the offloading decision from $x_n = 1$ to $x_n = 0$ for updating $\boldsymbol{x}(t)$. In t, the chosen UE may be not the best choice for the next iteration, but it could make the next iteration to achieve a better performance, which implies that with the iteration goes on, the result will become better. Finally, the policy cannot be improved no matter which UE is chosen to change the offloading decision from $x_n = 1$ to $x_n = 0$ or $\mathcal{N}_\mathrm{f}$ becomes an empty set, which shows the termination of the algorithm.

Now we give a simple analysis of the complexity of our proposed algorithm. The original problem is decomposed into two subproblems. The first one in our algorithm is DRA which is achieved by Lagrange multiplier method and bisection mothed, hence the complexity is $\mathcal{O}(\frac{N}{2}\log_2\frac{2p_m}{\xi N})$. The second one in our algorithm is JOCRA. In each iteration, the complexity of finding the offloading decision and its computation resource allocation is $\mathcal{O}(N)$. Let β be the number of iterations required to update the offloading decision. Furthermore, β is less than the number of UEs N, which makes the complexity $\mathcal{O}(N^2)$ in the worst case, i.e., all UEs need be executed locally. As a result, the computational complexity of the proposed algorithm is $\mathcal{O}(\max\{\frac{N}{2}\log_2\frac{2p_m}{\xi N}, N^2\})$.

V. SIMULATION RESULTS

In this section, the proposed method is evaluated to investigate its performance and effectiveness in comparison with four benchmark algorithms.

- *Local Only:* All UEs perform their tasks locally, i.e., $x_n = 0, n \in \mathcal{N}$. The computation resource $\boldsymbol{f}$ is allocated by Lagrange multiplier method. DRA is employed to obtain downlink bandwidth resource $\boldsymbol{w}$ and downlink power resource $\boldsymbol{p}$.
- *All Offload:* All UEs offload their tasks to the MEC server, i.e., $x_n = 1, n \in \mathcal{N}$. DRA is employed to obtain downlink bandwidth resource $\boldsymbol{w}$ and downlink power resource $\boldsymbol{p}$.
- *Joint Offloading Decision, Bandwidth, and Computation Resource Allocation (JOBCA)* from [12]. This scheme aims to minimize the total cost in terms of time delay and energy consumption. We use the JOBCA scheme to obtain offloading decision profile $\boldsymbol{x}$ and computation resource profile $\boldsymbol{f}$. DRA is employed to obtain downlink bandwidth resource $\boldsymbol{w}$ and downlink power resource $\boldsymbol{p}$.
- *Joint Offloading and Computation Resource for Minimizing Delay (JOCD):* In this scheme, time delay is considered as the only cost of task offloading. Task offloading scheduling and computation resource allocation method from [13] is adopted to make offloading decision $\boldsymbol{x}$ and allocate the computation resource $\boldsymbol{f}$. And DRA is employed to obtain downlink bandwidth resource $\boldsymbol{w}$ and downlink power resource $\boldsymbol{p}$.

TABLE 1. The simulation parameters.

Symbol	Parameter Name	Parameter Value
N	Number of UEs	100
w^u	Uplink Bandwidth	200 MHz
w^d	Downlink Bandwidth	500 MHz
p_n^u	Uplink Transmission Power	20 dBm
p_m	Total Downlink Transmission Power	38 dBm
N_0	Noise Power	-80 dBm
f_n^l	CPU Frequency of UEs	0.7 GHz
f_m	Total CPU Frequency of MEC server	100 GHz
C_n	Task CPU Cycles	[0.1, 1.0] $Gigacycle$
D_n^u	Upload Date Size	[0.1, 1.0]$\times 10^3$ kB
D_n^d	Download Date Size	[0.2, 2.0]$\times 10^3$ kB
e_c	Charge for Computation Resource	0.05 $\$/GHz$
e_d^n	Charge for Data Transmission	$\{0.2, 0.3\}$ $\$/Mbit$
λ_n^t	Time Weighting Factor	0. 5
λ_n^c	Charge Weighting Factor	0. 5

A. PARAMETER SETTING

We consider a simulation scenario with the following setting and the detail is showed in Table 1.

B. SIMULATION

Fig. 3 shows the comparison of the cost versus the number of UEs under different algorithms. Overall, the cost of the five different algorithms increases with the number of UEs. The curve of our algorithm JODOC is always at the bottom, which shows that JODOC has the best performance among the five algorithms. The cost of 'local only' algorithm is relatively large and grows at a basically linear rate as the number of UEs increases. The cost of 'all offload' algorithm grows sharply as the number of UEs increases, this is because the resource of the MEC system, i.e., uplink bandwidth and computation resource, is limited such that it cannot afford too many UEs. The gap between 'local only' algorithm and JODOC shows how much benefit can be obtained from JODOC. And the benefit almost remains unchanged when the number of UEs exceeds a certain threshold, which means that the benefit of UEs from task offloading reaches the upper limit in the given simulation scenario. The curves

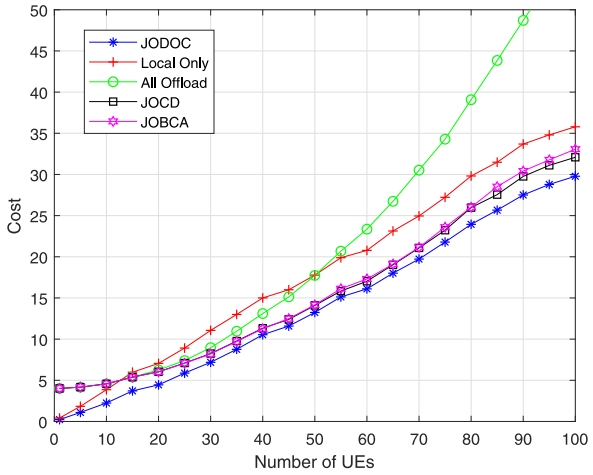

FIGURE 3. Cost versus the number of UEs under different algorithms.

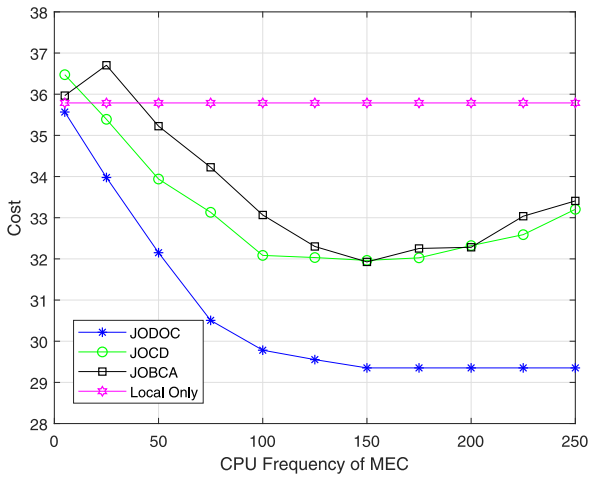

FIGURE 4. Cost versus computation resource under different algorithms.

'JOBCA' and 'JOCD' are overlapping with 'all offloading' at the first stage. That is because, different from our scheme, these two schemes will allocate all the computation resource to offloaded tasks. Thus the cost will be relatively large even though the number of UEs is small.

Fig. 4 shows the cost under different computation resource of MEC (CPU frequency). As can be seen, JODOC performs better than other algorithms. For JODOC, the cost will be less with more computation resource f_m at first. Then the cost will remain unchange. This is because when the computation resource of MEC is small, only a few tasks can be offloaded to MEC server and benefit from offloading. As the computation resource increasing, more tasks will be offloaded to MEC and the total benefit will increase. Finally, the computation resource increases to a relatively large value and all the offloaded tasks can obtain their optimal resource allocations, i.e., f_n^{opt}, thus the cost will remain unchange. For JOCD and JOBCA, their total cost is larger than 'local only' when the computation resource of MEC is small. This is because JOCD and JOBCA aims to minimize time delay

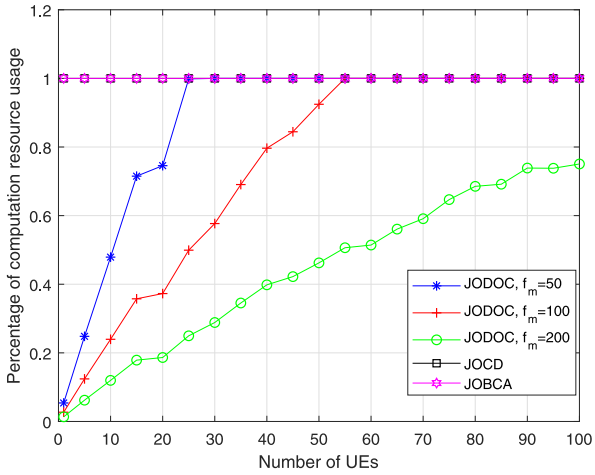

FIGURE 5. Percentage of computation resource of MEC usage.

and tradeoff of time delay and energy consumption, respectively. And these two schemes tend to offload more tasks to MEC server so to save the cost, but the benefit of time delay and energy consumption is too little to cover the cost of charge for computation resource. Then the cost of JOCD and JOBCA will also be less with more computation resource f_m, but it will increase with the computation resource. This is because the offloaded tasks already get their sufficient resource when the computation resource of MEC is relatively large and the execution time will not decrease significantly even if computation resource allocated to tasks increase, such that under this condition, the cost from the charge for computation resource dominates, so the total cost will increase.

Fig. 5 shows percentage of MEC server resource usage under different cases. JOBCA and JOCD will always use all the computation resource. For JODOC, when the number of UEs is small, the resource of MEC server will not be fully used. The percentage of resource usage will grow with the increasing of UEs number until it reaches to 1. If the number of offloaded tasks and their tasks are fixed, we can obtain the optimal computation resource allocation of UE n, i.e., f_n^{opt}, using (26). Thus, the optimal computation resource of MEC server can be estimated. We assume there are N_{ave} UEs requiring to offload their tasks to MEC server in average. C_{ave} and e_c are average required CPU cycles of tasks and the price of computation resource, respectively. We set that $\lambda_n^t = \lambda_n^e = 0.5$ and e_c depends on charge policy. Then we can we can estimate the optimal resource quantity of the MEC server

$$f_m^{opt} = N_{ave}\sqrt{\frac{C_{ave}}{e_c}} \tag{30}$$

Fig. 6 shows the offloading number of UEs under different algorithms. We can observe that the offloading number of UEs first increases rapidly as the number of UEs increases, then increases slowly, and finally remains stable when the number of UEs exceeds a certain threshold. The reason is that

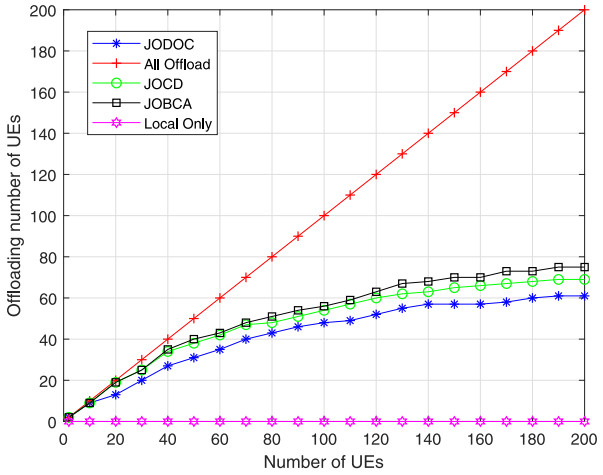

FIGURE 6. Offloading number of UEs under different algorithms.

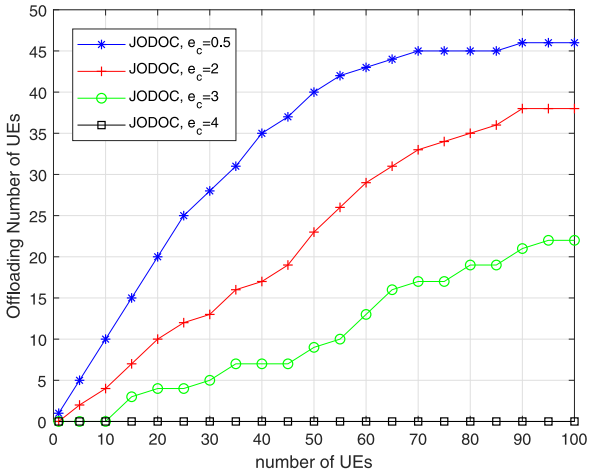

FIGURE 8. Offloading number under different charge policy e_C.

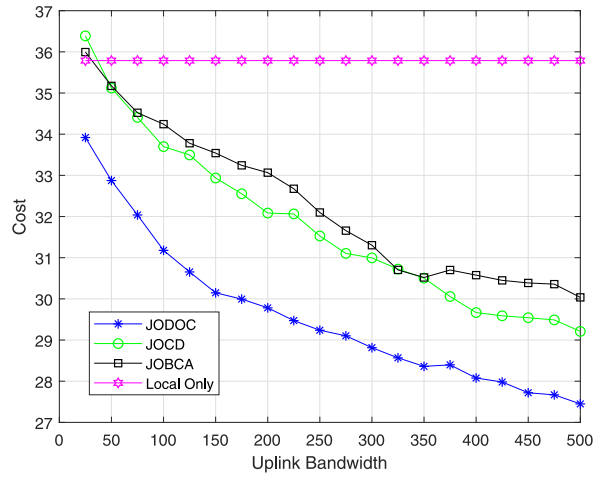

FIGURE 7. Cost versus uplink bandwidth under different algorithms.

each task can obtain enough computation resource in first stage, so all UEs choose to offload their tasks. In the second stage, each task gets less computation resource as the number UEs increases such that some UEs cannot benefit from task offloading and choose local execution. Finally, only a fixed number of UEs can benefit from task offloading and others abandon offloading task. The JOCD and JOBCA tend to offload more tasks of UEs to MEC server than JODOC. The reason is that these two schemes are designed to minimize time delay and energy consumption, which can be significantly reduced by task offloading, while JODOC will suffer from the charge for computation resource and therefore fewer tasks will be offloaded to MEC.

Fig. 7 shows how uplink bandwidth the total cost. It can be observed that our proposed algorithm JODOC obtain lower cost than JOBCA and JOCD. The cost of JODOC will decrease with the uplink bandwidth resource and then almost keeps unchange when the uplink bandwidth is relatively large. That is because, uploading time t_n^u of tasks is large when the uplink bandwidth is small, and only a few tasks can satisfy offloading conditions and are offloaded to the MEC server. When the uplink resource is abundant, the uploading time t_n^u of tasks becomes small, and more tasks can satisfy offloading conditions and are offloaded to MEC server. Finally, the uploading time almost reaches to 0 when uplink bandwidth is very large.

From Fig. 8 we can clearly observe that the number of offloading UEs will decrease with the increasing of e_c. This is because the cost of offloading tasks to the MEC server becomes larger than that of local execution with the increasing of e_c, which means UEs could not benefit from task offloading any more and the offloading number of UEs become small. Specifically, when $e_c = 4$, all UEs choose not to offload their tasks to the MEC server because of the huge charge.

VI. CONCLUSION

In this paper, we considered UEs will be charged for MEC servers and jointly optimized the downlink resource allocation, offloading decision and computation resource allocation. Then we formulated it as an MIP problem to get its optimal solution and decoupled the problem into two subproblem. Finally, we proposed an iterative algorithm to get its suboptimal solution. The simulation results show our solution performs better than benchmark algorithms and the MEC servers will not always be full load with the charge constraint.

REFERENCES

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.

[2] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture applications and approaches," *Wireless Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, 2011.

[3] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," Sophia Antipolis, France, ETSI, White Paper, 2015.

[4] P. Corcoran and S. K. Datta, "Mobile-edge computing and the Internet of Things for consumers: Extending cloud computing and services to the edge of the network," *IEEE Consum. Electron. Mag.*, vol. 5, no. 4, pp. 73–74, Oct. 2016.

[5] X. Zhang, Y. Zhong, P. Liu, F. Zhou, and Y. Wang, "Resource allocation for a UAV-enabled mobile-edge computing system: Computation efficiency maximization," *IEEE Access*, vol. 7, pp. 113345–113354, 2019.
[6] Y. Cui, Y. Liang, and R. Wang, "Resource allocation algorithm with multi-platform intelligent offloading in D2D-enabled vehicular networks," *IEEE Access*, vol. 7, pp. 21246–21253, 2019.
[7] R. Ford, M. Zhang, M. Mezzavilla, S. Dutta, S. Rangan, and M. Zorzi, "Achieving ultra-low latency in 5G millimeter wave cellular networks," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 196–203, Mar. 2017.
[8] 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.
[9] H. Flores, P. Hui, S. Tarkoma, Y. Li, S. Srirama, and R. Buyya, "Mobile code offloading: From concept to practice and beyond," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 80–88, Mar. 2015.
[10] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.
[11] 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.
[12] Q. Pham, L. B. Le, S. Chung, and W. Hwang, "Mobile edge computing with wireless backhaul: Joint task offloading and resource allocation," *IEEE Access*, vol. 7, pp. 16444–16459, 2019.
[13] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.
[14] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.
[15] S. Ulukus *et al.*, "Energy harvesting wireless communications: A review of recent advances," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 3, pp. 360–381, Mar. 2015.
[16] P. Wang, C. Yao, Z. Zheng, G. Sun, and L. Song, "Joint task assignment, transmission, and computing resource allocation in multilayer mobile edge computing systems," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2872–2884, Apr. 2019.
[17] 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.*, vol. 6, no. 3, pp. 4804–4814, Jun. 2019.
[18] R. Kobayashi and K. Adachi, "Radio and computing resource allocation for minimizing total processing completion time in mobile edge computing," *IEEE Access*, vol. 7, pp. 141119–141132, 2019.
[19] W. Labidi, M. Sarkiss, and M. Kamoun, "Joint multi-user resource scheduling and computation offloading in small cell networks," in *Proc. IEEE 11th Int. Conf. Wireless Mobile Comput. Netw. Commun. (WiMob)*, Oct. 2015, pp. 794–801.
[20] S. Li, Y. Tao, X. Qin, L. Liu, Z. Zhang, and P. Zhang, "Energy-aware mobile edge computation offloading for IoT over heterogenous networks," *IEEE Access*, vol. 7, pp. 13092–13105, 2019.
[21] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, Aug. 2017.
[22] S. Yu, R. Langar, X. Fu, L. Wang, and Z. Han, "Computation offloading with data caching enhancement for mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11098–11112, Nov. 2018.
[23] M. Liu and Y. Liu, "Price-based distributed offloading for mobile-edge computing with computation capacity constraints," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 420–423, Jun. 2018.
[24] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1619–1632, Aug. 2018.
[25] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.
[26] J. Liu, P. Li, J. Liu, and J. Lai, "Joint offloading and transmission power control for mobile edge computing," *IEEE Access*, vol. 7, pp. 81640–81651, 2019.
[27] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Joint computation offloading, resource allocation and content caching in cellular networks with mobile edge computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[28] Z. Jian, W. Muqing, and Z. Min, "Joint computation offloading and resource allocation in C-RAN with MEC based on spectrum efficiency," *IEEE Access*, vol. 7, pp. 79056–79068, 2019.
[29] Y. Pochet and L. A. Wolsey, *Production Planning by Mixed Integer Programming*. New York, NY, USA: Springer, 2006.
[30] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

KEHAO WANG received the B.S. degree in electrical engineering and the M.S. degree in communication and information system from the Wuhan University of Technology, Wuhan, China, in 2003 and 2006, respectively, and the Ph.D. degree from the Department of Computer Science, University of Paris-Sud XI, Orsay, France, in 2012. He is currently an Associate Professor with the Department of Information Engineering, Wuhan University of Technology. His research interests are cognitive radio networks, wireless network resource management, and data hiding.

ZHIXIN HU received the B.S. degree in electrical engineering from the Wuhan University of Technology, Wuhan, China, in 2018, where he is currently pursuing the Ph.D. degree with the Department of Information Engineering. His research interests include MEC, machine learning, transfer reinforcement learning, and deep reinforcement learning.

QINGSONG AI received the B.S. degree in electrical engineering and the M.S. and Ph.D degrees in communication and information system from the Wuhan University of Technology, Wuhan, China, in 2003, 2006, and 2010, respectively, where he is currently a Professor with the Department of Information. His research interests are digital watermark, data hiding, and information security.

YI ZHONG received the B.E.E.E. and M.S.E.E. degrees from the School of Information Engineering, Wuhan University of Technology in 1999 and 2002, respectively, and the Ph.D. degree in information and communication engineering from the Wuhan University of Technology in 2007. He has published numerous articles and papers in domestic and international journals, and has attended in many conferences on topics ranging from embedded control systems to system fault diagnosis.

JIHONG YU received the B.E. degree in communication engineering and the M.E. degree in communication and information systems from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2010 and 2013, respectively, and the Ph.D. degree in computer science from the University of Paris-Sud, Orsay, France, in 2016. In 2017, he was a Research Fellow with the School of Computing Science, Simon Fraser University, British Columbia, Canada. He is currently an Associate Professor with the Beijing Institute of Technology. His research interests include RFID technologies, wireless communications, and Internet of Things.

PAN ZHOU (Senior Member, IEEE) received the B.S. degree in advanced class of the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2006, and the Ph.D. degree from the School of Electrical and Computer Engineering, Georgia Institute of Technology (Georgia Tech), Atlanta, USA, in 2011. He is currently an Associate Professor with the School of Cyber Science and Engineering, HUST. From 2011 to 2013, he was a Senior Technical Member with Oracle, Inc., Boston, MA, USA. His current research interests include network security, machine learning and big data analytics, and information networks.

LIN CHEN received the B.E. degree in radio engineering from Southeast University, China, in 2002, the M.S. degree in networking from the University of Paris 6, and the Diploma degree in engineering and the Ph.D. degree from Telecom ParisTech, Paris, in 2005 and 2008, respectively. He currently works as an Associate Professor with the School of Data and Computer Science, Sun Yat-sen University. His main research interests include modeling and control for wireless networks, security and cooperation enforcement in wireless networks, and game theory.

HYUNDONG SHIN (Senior Member, IEEE) received the B.S. degree in electronics engineering from Kyung Hee University (KHU), Yongin, South Korea, in 1999, and the M.S. and Ph.D. degrees in electrical engineering from Seoul National University, Seoul, South Korea, in 2001 and 2004, respectively. From 2004 to 2006, he was a Postdoctoral Researcher with the Wireless Communication and Network Sciences Laboratory, Laboratory for Information Decision Systems, Massachusetts Institute of Technology. In 2006, he joined KHU, where he is currently a Professor with the Department of Electronic Engineering. His research interests include quantum information science, wireless communication, and nanonetworks. He was a recipient of the Knowledge Creation Award in the field of computer science from the Korean Ministry of Education, Science and Technology in 2010, the IEEE Communications Society Guglielmo Marconi Prize Paper Award in 2008, and the William R. Bennett Prize Paper Award in 2012. He served as the Publicity Co-Chair for IEEE PIMRC in 2018, and the Technical Program Co-Chair for IEEE WCNC (PHY Track, 2009), and IEEE GLOBECOM (Communication Theory Symposium 2012 and Cognitive Radio and Networks Symposium 2016). He was an Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2007 to 2012 and IEEE COMMUNICATIONS LETTERS from 2013 to 2015.