IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 12, 2021, accepted May 1, 2021, date of publication May 10, 2021, date of current version May 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3078539

Computation Scheduling of Multi-Access Edge Networks Based on the Artificial Fish Swarm Algorithm

TIANTIAN LI, FENG YANG, DEPENG ZHANG, AND LINBO ZHAI
School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China

Corresponding authors: Feng Yang (yangfeng@sdnu.edu.cn) and Linbo Zhai (zhai@mail.sdu.edu.cn)

This work was supported in part by the Key Research and Development Program of Shandong Province, China, under Grant 2017GGX10142, and in part by the China Scholarship Fund.

ABSTRACT With the increasing requirements for computing in modern society, Multi-access Edge Computing (MEC) has received widespread attention for meeting low-latency. In MEC network, mobile devices can offload computing-intensive tasks to edge servers for computing. Wireless Power Transmission (WPT) provides initial energy for mobile devices, and the tasks of mobile devices consume energy when they are locally calculated or completely offloaded. The combination of the two technologies forms the Wireless Powered Mobile Edge Computing (WP-MEC) network. In this article, considering the impact of WPT transmission time τ_0, we study the offloading and scheduling of tasks for multiple mobile devices in the WP-MEC network, which is an NP-hard problem. We formulate this scheduling problem to minimize the time delay under the constraint of WPT transmission energy. We regard our problem studied in this paper as a multidimensional knapsack problem (MKP). The difference is that the knapsack capacity in MKP is limited, while in our problem, the knapsack that one item can choose is limited. Therefore, we improve the Artificial Fish Swarm Algorithm (AFSA) and propose Computation Scheduling Based on the Artificial Fish Swarm Algorithm (CS-AFSA) to find the optimal scheduling. We encode a scheduling scheme as an artificial fish and regard the delay corresponding to the scheduling as the optimization object. The optimal artificial fish can be gradually approached and determined through the swarm, follow and prey behavior of artificial fish. The optimal artificial fish is the optimal scheduling scheme. More importantly, based on the original behavior of AFSA, we also improve the scheme that does not meet the WPT energy constraint, including the modification of infeasible artificial fish and insufficient artificial fish. Besides, we also consider how to find the best WPT transmission time τ_0. Finally, we perform data simulation on the proposed algorithm.

INDEX TERMS Artificial fish swarm algorithm, WP-MEC network, WPT transmission time.

I. INTRODUCTION

With the explosive growth of the number of mobile devices and the emergence of many emerging applications, the traffic of mobile networks has grown exponentially. The traditional centralized network architecture cannot meet the needs of mobile devices due to the heavy backhaul link load and long delay. Therefore, a new architecture that opens network capabilities from the core network to the edge network is proposed, named Multi-access Edge Computing (MEC). However, mobile devices are resource-constrained in energy, which reduces the quality of service [1]. MEC can enhance the computing power of mobile devices [2] and [3]. It allows mobile devices to offload their intensive computing to edge devices within coverage [4].

Wireless Power Transfer (WPT) uses half-duplex communication to transmit and receive radio-frequency (RF) power from edge devices to mobile devices at the same time and the same frequency. The mobile devices can convert the received RF power into electrical energy. The energy collected by the mobile devices depends on the wavelength of the radio frequency signal, the transmission power, and the distance between the mobile device and the edge device [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Cunhua Pan.

Combined with Wireless Power Transfer (WPT), the Wireless Powered Multi-access Edge Computing (WP-MEC) network is proposed by many people. In the WP-MEC network, mobile devices harvest energy from edge devices through WPT and then offload intensive and delay-sensitive computing to edge devices through MEC. So, the energy consumed by the mobile device cannot exceed the energy it harvests through WPT. Due to half-duplex communication, WPT and offloading processes cannot be performed at the same time, whereas WPT and local calculation processes of mobile devices can be performed at the same time. At this time, the basic problem in the WP-MEC network becomes how to schedule the computations to minimize the total delay [6]–[8].

In this paper, we divide a continuous time into discrete time slots whose length is L. A time slot is called T. Each discrete slot is small enough so that each mobile device has only one task to be handled in a time slot T. We simplify the problem to study the computation scheduling in the time slot T. Therefore, the time slot T can be divided into two parts. The first part is WPT transmission time τ_0, and the other part is offloading time T-τ_0. Since the result processed by the edge device is far less than the amount of data before processing, the time and energy consumed by the edge device to send the result back to the mobile device can be ignored. We have to consider both WPT transmission time τ_0 and task scheduling in the WP-MEC network. At this time, the task scheduling problem of the WP-MEC network becomes more complex. We should consider the arrangement of WPT transmission time τ_0 and the factors such as half-duplex communication. How to find the optimal scheduling under the constraint of WPT energy is the problem that we should solve.

The contributions of this article are as follows:

- We summarize the scheduling problem of the WP-MEC network as minimizing the delay under the constraint of WPT transmission energy. We want to find the best WPT transmission time τ_0 and scheduling to minimize the total delay when the mobile devices perform task offloading and calculation according to the scheduling, while meeting the energy corresponding to the WPT transmission time τ_0.
- We study on the basis of a given WPT transmission time τ_0, how to perform task scheduling to minimize the total delay, that is, Scheduling Algorithm when τ_0 is known (STKA). Computation Scheduling Based on the Artificial Fish Swarm Algorithm (CS-AFSA) is proposed based on STKA. We find the candidate set of WPT transmission time τ_0, and perform the STKA algorithm on each element in the set to find the most suitable WPT transmission time τ_0 and scheduling.
- Using the above algorithm for data simulation, the simulation results show that the performance of our proposed algorithm has been greatly improved.

The rest organization of this article is as follows. In section II, we introduce the related work of this paper.

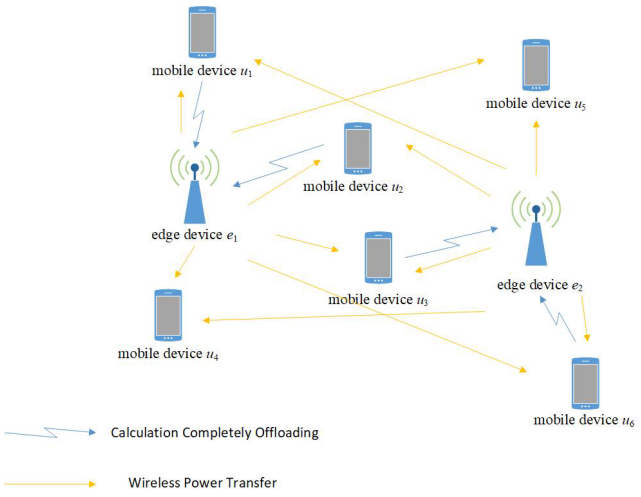

FIGURE 1. An example of the WP-MEC network.

In section III, the problem definition and basic model of how to minimize the total delay in the WP-MEC network are carried out. Section IV proposes CS-AFSA to solve the scheduling problem. The data simulation of the algorithm is in section V. Section VI concludes the paper.

II. RELATED WORKS

Scheduling is an important issue in the Internet of Things (IoT) [14]–[17]. The computing scheduling problem of multi-access edge network has been deeply studied in many articles, and the research was summarized in [2] and [18]. Authors in [2] focused on how to schedule to minimize the time delay in the MEC system. Authors in [18] studied how to schedule to minimize energy consumption in a deadline-aware MEC system.

Authors in [3] maximized the perceived satisfaction, by offloading portion of computing tasks to the different MEC servers. In [25], deep learning and edge computing were combined and used for efficient processing of huge amount of data with distinct accuracy. But these networks were not associated with WPT.

There are also some works about WPT. Authors in [26] considered how to maximize the energy efficiency via joint optimization of channel selection, time allocation, power control, and so on. A cooperative secrecy transmission mechanism, which can use the transmitter signal of primary user (PU) as a dedicated RF source, was proposed in [27].

The WP-MEC Network had been also considered in some papers [6]–[11]. These works, with multiple mobile devices and one edge device, can be divided into two types. The first type of work focused on maximizing the computing performance of the WP-MEC network. Authors in [6] studied the use of only one mobile device and one edge to maximize the computational probability of the WP-MEC network equipment. The algorithm proposed in [7] could maximize the weighted sum calculation rate of all mobile devices in the WP-MEC network.

The second type of work is used to minimize the implementation cost of the WP-MEC network. The authors in [9] considered that there was a WP-MEC network with only one mobile device and one edge device. They proposed a dynamic algorithm based on Lyapunov optimization to minimize the execution cost of the WP-MEC network. The authors in [10] considered the WP-MEC network with one edge device and two nearby mobile devices. They proposed an algorithm to minimize the energy consumption during transmission. However, all the work above on WP-MEC Network was based on one edge device.

There is more than one edge device in WP-MEC Network [12], [13], and [19]. The authors in [12] discussed energy consumption with a single cloud server first, then considered multiple cloud servers. [13] proposed a new network model aiming at improving users' experience. The objective in [19] was to minimize the total cost incurred.

Therefore, it is discussed in this paper that the WP-MEC Network with multiple edge devices and multiple mobile devices.

III. PROBLEM DEFINITION

A. SYSTEM MODEL

In this section, we assume a WP-MEC Network. There are n mobile devices and m edge devices. The set of mobile devices is defined as $U = \{u_1, u_2.....u_n\}$, and the collection of edge devices is defined as $E = \{e_1, e_2.....e_m\}$.

Assuming that in the time slot T, the data amount of the task that the mobile device $u_i \in U$ needs to process is c_i bits. The task is indivisible, which means that the mobile device can only choose to complete all the data locally or completely offload to an edge device for processing.

B. ENERGY HARVEST MODEL

All edge devices simultaneously transmit RF power in WPT transmission time τ_0. Energy waves transmitted by multiple edge devices simultaneously can be combined constructively or in a destructive manner [28]. Constructive combination refers to the fact that the received power from the combined energy wave of multiple edge devices is higher than the power received from a single edge device. Besides, the mobile devices remain stationary in the time slot T. It is assumed that the initial energy of all mobile devices is zero at the beginning of the time slot T.

When all edge devices start to transmit RF power at the same time in WPT transmission time τ_0, m RF power will form a combined wave. Then, the energy harvested by the mobile device u_i in time τ_0 can be expressed as (1).

$$E_i^H = \mu P_i g_i \tau_0, \quad \forall u_i \in U \tag{1}$$

where P_i represents the combined wave transmission power of m edge devices to the mobile devices $u_i \in U$. g_i represents the channel gain that is the gain of the downlink transmitting the combined energy wave. It is a white Gaussian noise with a constant mean, whose mean is a constant, and $\mu \in (0,1)$ represents the energy conversion efficiency for the mobile device.

C. COMPUTATION MODEL

Any task from the mobile device u_j in the time slot T can only be calculated locally or be offloaded completely to an edge device, and cannot be offloaded partially. If the energy harvested by the mobile device u_j is neither sufficient for computing locally nor for offloading completely, the mobile device will abandon its computing task in the time slot T. We divide mobile devices into two sets (M and $\overline{M}$) according to whether they have completed the computation task in the time slot T. The purpose of this operation is to facilitate classification and subsequent processing of data that has been offloaded to the same edge device. The set M means that the mobile devices in the set have completed the calculation task in the time slot T. The set $\overline{M}$ means that the mobile devices in the set have given up the task in the time slot T. The set M can be further divided into $m+1$ sets, $M_0, M_1, M_2, ... M_m$. Among them, the set M_0 indicates that the mobile devices in the set have completed the calculation locally. M_k $(1 \leq k \leq m)$ indicates that the mobile devices in the set have completely offloaded the data to the edge device e_k for calculation. The relationship between sets is:

$$\begin{cases} M = \bigcup_{k=0}^{m} M_k \\ \overline{M} = U/M \end{cases} \tag{2}$$

1) LOCAL COMPUTING MODEL

Every mobile device u_i in the M_0 set completes the calculation of c_i bits data locally. ϕ_i *is* the CPU cycles required by the mobile device to process 1 bit of data. f_i is the CPU frequency (cycles per second), whose maximum value is f_{max}. τ_i is the time used by mobile device u_i to process data in the time slot T, $\tau_i < T$. If mobile device u_i can complete local calculation in the time slot T, τ_i is calculated as:

$$\tau_i = \frac{c_i \varphi_i}{f_i}, \quad \forall u_i \in M_0 \tag{3}$$

At this point, the total time delay that mobile device u_i completes the task locally is

$$t_i^L = \tau_i, \quad \forall u_i \in M_0 \tag{4}$$

The energy consumption *EL i* of the local calculation is

$$E_i^L = P_i \tau_i, \quad \forall u_i \in M_0 \tag{5}$$

where P_i represents the local computing power of mobile device u_i.

At the same time, energy consumption *EL i* must satisfy (6).

$$E_i^L \leq E_i^H, \quad \forall u_i \in M_0 \tag{6}$$

2) COMPLETELY OFFLOADING MODEL

In this part, the communication model is based on Non-Orthogonal Multiple Access (NOMA) technique. As mentioned in the previous part, the time when the result is returned

from the edge device is ignored. Therefore, we only focus on the rate of the uplink from the mobile device to the edge device, and how long the uplink takes.

First, some parameters are assumed. The data that mobile device u_j completely offloads to the edge device e_k for calculation is c_j bits in the time slot T. The transmission power from mobile device u_j to edge device e_k is P_{jk}, and the wireless channel gain is h_{jk}. The additive white Gaussian noise is w_{jk}. If $|h_{1k}|^2 > |h_{2k}|^2 > \ldots\ldots > |h_{jk}|^2 > \ldots\ldots > |h_{bk}|^2$(the number of mobile devices $u_j \in M_k$ *is* b), the upstream rate of mobile device $u_j \in M_k$ can be expressed as R_{jk}. The transmission delay uplink takes is τ_{jk}. We can obtain

$$\begin{cases} R_{jk} = \log_2(1 + \dfrac{P_{jk}\left|h_{jk}\right|^2}{\sum_{q=j+1}^{b} P_{qk}\left|h_{qk}\right|^2 + w_j}) \\ \tau_{jk} = \dfrac{c_j}{R_{jk}}, \quad \forall u_j \in M_k, \ 1 \le k \le m \end{cases} \tag{7}$$

The CPU cycle required by the edge device to process 1 bit of data is ϕ_k. f_k is the CPU frequency (cycles per second), and the maximum is f_{kmax}. τ_{kj} is the time taken by the edge device e_k to process the data of the mobile device u_j.

$$\tau_{kj} = \frac{c_j \varphi_k}{f_k}, \quad \forall u_j \in M_k, \ 1 \le k \le m \tag{8}$$

The edge device e_k can process n_k tasks at the same time. The n_{k+1} task that is offloaded to the edge device will enter the waiting queue. If the task of mobile device u_j is the n-th task offloaded to the edge device e_k, the waiting delay at this time is

$$\tau_{jk}^{W} = \begin{cases} 0, & n \le n_k \\ \tau_j, & n > n_k \end{cases}, \quad \forall u_j \in M_k \tag{9}$$

where τ_j represents the processing delay of the fastest completed task among the current processing tasks of the edge device.

At this point, the total delay for the mobile device u_j to completely offload and complete the calculation is

$$t_j^{O} = \tau_{jk} + \tau_{kj} + \tau_{jk}^{W}, \quad \forall u_j \in M_k \tag{10}$$

Energy consumption $EO\,jk$ is defined as (we only consider the energy consumption of mobile device offloading)

$$E_{jk}^{O} = P_{jk}\tau_{jk}, \quad \forall u_j \in M_k, \ 1 \le k \le m \tag{11}$$

At the same time, (12) is satisfied.

$$E_{jk}^{O} \le E_j^{H}, \quad \forall u_j \in M_k, \ 1 \le k \le m \tag{12}$$

D. MODEL SUMMARY

We formulate the scheduling problem of the WP-MEC network as minimizing the time delay under the constraint of WPT transmission energy. When the WPT transmission time τ_0 is known, the total model is as follows:

$$\begin{aligned} &\min \mathrm{t} = \sum\nolimits_{\mathrm{u_i} \in M_0} t_i^{L} + \sum\nolimits_{k=1}^{m} \sum\nolimits_{u_j \in M_k} t_j^{O} \\ &s.t.\ E_j^{L} \le E_j^{H}, \quad \forall u_j \in M_k, \ 1 \le k \le m \\ &E_i^{L} \le E_i^{H}, \quad \forall u_i \in M_0 \\ &0 \le f_i \le f_{\max}, \quad \forall u_i \in M_0 \\ &0 \le f_k \le f_{k\max}, \quad k \in [1, m] \end{aligned} \tag{13}$$

TABLE 1. Notation summary.

Notation	Notation Description
$u_i\ (1 \le i \le n)$	mobile device
$e_k\ (1 \le k \le m)$	edge device
c_i	the data mobile device u_i needs to process in the time slot T
τ_0	WPT transmission time τ_0
E_i^{H}	the energy that mobile device u_i harvests in τ_0
τ_i	the delay that mobile device u_i processes data locally
t_i^{L}	the time delay that mobile device u_i completes calculation locally
E_i^{L}	the energy consumption that mobile device u_i completes calculation locally
τ_{jk}	the transmission delay that mobile device u_j offloads its data to edge device e_k
τ_{kj}	the delay that edge device e_k processes the data of mobile device u_j
τ_{jk}^{W}	the waiting delay before edge device e_k processes the data of mobile device u_j
t_j^{O}	the time delay that mobile device u_j completes calculation by offloading data to edge device e_k
E_{jk}^{O}	the energy consumption that mobile device u_j completes calculation by offloading data to edge device e_k

The problem in (13) is an NP-hard problem because it can be understood as Generalized Assignment Problem (GAP) [20]–[22], which is a special NP-hard problem [23]. To solve the problem in (13), we propose CS-AFSA in the next section.

IV. ALGORITHM PROCESS

In this section, CS-AFSA is proposed to minimize time delay under the constraint of WPT transmission energy. We mainly use the modified Artificial Fish Swarm Algorithm (AFSA) to determine the optimal scheduling. Since the change of the WPT transmission time τ_0 will affect the determination of the scheduling, we first study how to find a scheduling scheme with the smallest delay when the WPT transmission time is known, and propose STKA. Then we determine the candidate set of WPT transmission time τ_0, and perform STKA for each element in the set, compare and determine the most suitable WPT transmission time τ_0 and scheduling.

A. THE MODIFIED AFSA

First, we briefly introduce the knapsack problem. The backpack problem is defined as: given a set of items, a backpack.

Each item has its own value and weight, and the backpack has a certain capacity. How to get the highest value under the premise that the weight of the items loaded into the backpack does not exceed the capacity of the backpack. When the backpack is expanded from one to multiple, and each backpack may have different capacity limits, how to place the items can maximize the total value obtained? This becomes the problem of multiple backpacks. AFSA can be well used for the multidimensional knapsack problem (MKP).

The problem studied in this paper can be roughly regarded as MKP. The mobile device is regarded as an item and the edge device is regarded as a backpack. At this point, our problem becomes how to arrange scheduling can minimize the delay under the constraint of WPT transmission energy. The difference between our problem and MKP is that the capacity of the backpack in the MKP is limited, whereas the capacity of our backpack is unlimited. In this paper, due to the limitation of the obtained energy, the tasks of mobile devices are intelligently offloaded to some edge devices. There is no limit to the number of tasks received by the edge device, but when the number of tasks received exceeds the number of tasks that the edge device can handle at the same time, the task must enter the waiting queue.

According to the difference between the problem in this article and the MKP, we modify AFSA, which is mainly reflected in the repair of infeasible artificial fish and insufficient artificial fish, and the calculation of waiting time delay.

1) THE REPAIR OF INFEASIBLE ARTIFICIAL FISH AND INSUFFICIENT ARTIFICIAL FISH

Infeasible artificial fish means that the total weight of a backpack placed in the corresponding scheduling method of the artificial fish exceeds the capacity limit of the backpack. It can be repaired by the "random repair" strategy. In this paper, the energy consumed by local computing is an order of magnitude less than the energy consumed by complete offloading, so the algorithm steps are as follows:

(1) If a task is calculated locally, we compare the energy consumed by the local calculation and the amount of energy obtained, and consider whether (6) is satisfied. If the obtained energy cannot meet the local calculation, the task will be abandoned in the time slot T and set it to -1, put the task into the $\overline{M}$ set;

(2) If a task is completely offloaded and calculated, we compare the energy consumed by the complete offloading and the amount of energy obtained, and consider whether (12) is satisfied. If the acquired energy cannot satisfy the complete offloading, other mobile devices are randomly selected for mobile devices to offload, and the comparison of the energy relationship is repeated. For mobile devices that cannot be completely offloaded, the task will be given up in the time slot T and set it to -1. Then put the task into the $\overline{M}$ set;

Insufficient artificial fish means that the volume in some backpacks is not fully utilized. After repairing the infeasible artificial fish, there may be some tasks that cannot be offloaded to any mobile device and are abandoned. These tasks may be executed locally. The algorithm steps are as follows:

For each task in the $\overline{M}$ set, judging the local calculation can be performed according to (6). If (6) is satisfied, the task is set to 0. Then we put the task into the M_0 set.

We put the repair of infeasible artificial fish after AFSA performs any action, and put the repair of inadequate artificial fish behind the infeasible artificial fish.

2) THE CALCULATION OF WAITING TIME DELAY

Since each edge device can only process n_k tasks at the same time, the tasks that come later must enter the waiting queue to wait. The calculation steps of waiting time delay are:

(1) For each scheduling, an artificial fish, all tasks are sorted and stored according to which edge device it is offloaded. For example, task 5, 6, and 8 are offloaded to the edge device e_1 and stored in array $a1$.

(2) The tasks that are offloaded to the same edge device e_k are sorted according to the offload delay and stored in array a_k.

(3) The first n_k tasks of the edge device e_k begin to execute at the same time. When one of the tasks is completed, the first task in array a_k which is not calculated is marked and calculated. The waiting delay of the task is the calculation delay of the completed task plus the task calculation delay, assuming that the edge device starts to calculate after all tasks arrive.

The basic algorithm of the AFSA will not be introduced in detail. For the detailed process and algorithm introduction of the Artificial Fish Swarm Algorithm, please read [24].

Then, the scheduling process of Algorithm 1 is elaborated. For example, during initialization, an artificial fish $\{-1, 2, 3, 0, 3\}$ is generated. This artificial fish represents a scheduling scheme that the mobile device u_1 gives up its task, the mobile device u_2 offloads its task to edge device e_2, the mobile device u_3 and u_5 offload their tasks to edge device e_3, and the mobile device u_4 completes its task locally. Then the swarming, following, and preying behaviors are performed in turn to approach the optimal artificial fish. After a finite number of iterations, the optimal artificial fish is the optimal scheduling scheme.

B. SCHEDULING ALGORITHM WHEN τ_0 IS KNOWN (STKA)

In the previous part, we proposed modified AFSA. On this basis, we study how to perform optimal scheduling with the minimum delay when τ_0 is known. The specific steps of the algorithm are shown in Algorithm 2, which is also called Scheduling Algorithm when τ_0 is known (STKA). The algorithm steps are as follows:

Step 1 (line 1 to 6). the initialization of network parameters and calculation of local delay, local energy consumption, completely offloading delay, and complete offload energy consumption.

Step 2 (line 7). the Modified AFSA is called.

Algorithm 1 Modified AFSA

Input: the number of mobile devices n; the number of edge devices m; maxcircle and maxiteration.

Output: the scheduling scheme with the minimum delay, that is, the optimal artificial fish.

```
AF_init(afx);
afx1 ←afx; afx2 ←afx;
for 1≤circle≤maxcircle
 for 1≤iteration≤maxiteration
   AF_swarm(afx1);
   AF_infeasible(afx1);
   AF_inadequate(afx1);
   best1 ← AF_value(afx1);
   AF_follow(afx2);
   AF_infeasible(afx2);
   AF_inadequate(afx2);
   Best2 ← AF_value(afx2);
   if(best1<best2)
      afx=afx1;
   else
      afx=afx2;
  end if
  the bulletin board updates the best artificial fish and the minimum delay
 end for
 output the optimal artificial fish and minimum delay recorded on the bulletin board
end for
```

Algorithm 2 STKA

Input: the number of mobile devices n; the number of edge devices m; τ_0; network parameters, including μ, B, T, maxcircle and maxiteration.

Output: the scheduling scheme with the minimum delay, that is, the optimal artificial fish.

```
data initialization;
EH i ← Eq. (1);
delaylocal_i ← Eq. (3);
volumelocal_i ← Eq. (5);
delay (i, k) ← Eq. (7);
volume (i, k) ← Eq. (8);
call the Modified AFSA;
```

C. COMPUTATION SCHEDULING ALGORITHM BASED ON AFSA (CS-AFSA)

In STKA, we assume the value of WPT transmission time τ_0, and find the optimal scheduling under this condition. In CS-AFSA, we will get a set of WPT transmission time τ_0 possible values, and output the scheduling with the least delay under the energy constraint and the value of WPT transmission time τ_0. Applying STKA to each time in the WPT transmission time set (WPTS), we can find the scheduling and WPT transmission time that can minimize the delay in the time slot T. Specific steps are as follows:

Algorithm 3 CS-AFSA

Input: the number of mobile devices n; the number of edge devices m;
network parameters, including μ, B, T, β, maxcircle and maxiteration.

Output: WPT time τ_0 and the scheduling scheme with the minimum delay, that is, the optimal artificial fish.

```
t_i0 ← the time when (6) takes the equal sign, the energy harvested just meets the energy transmission time consumed by local calculations;
t_i ← the time when (12) takes the equal sign;
add τmin 0=min {t_i0, t_i|1≤ i ≤ n} to WPTS;
add τmax 0=max {T-t_i|1≤ i ≤ n} to WPTS;
add elements {t_i0, t_i|1≤ i ≤ n} that not in WPTS to WPTS;
sorts of elements in WPTS;
for 1≤ l ≤ δ
if (WPTS[l]-WPTS[l-1] ≤ β (T -WPTS[l]))
   add WPTS[l]- β (T -WPTS[l]) to WPTS;
 end if
end for
for 1≤ l ≤ δ
  τ_0 =WPTS[l];
  call the algorithm 2;
end for
output the scheduling scheme with the minimum delay and the value of τ_0;
```

(1) Determine the boundary of the candidate set WPTS. The acquired energy should allow at least one mobile device to complete the calculation. The shortest energy transmission time at this time is τ_0, and add it to the WPTS. The maximum energy transmission time should allow at least one task of mobile devices to be offloaded to the edge device, which is recorded as τ_0, and add it to the WPTS.

(2) The energy consumption of all mobile devices locally or completely offload is calculated. Let the energy consumed equal to the energy harvested. That is the equal sign in (6) or (12) is satisfied, then τ_0 is obtained. If there is no such element in the WPTS, τ_0 is added to WPTS.

(3) Then we sort all elements in WPTS. Supposing there are δ elements in total, and WPTS[l]-WPTS[l-1] $\leq \beta$ (T -WPTS[l]) is satisfied, $1< l < \delta$, $\beta >0$. If the formula is not satisfied, add WPTS[l]- β (T -WPTS[l]) to the WPTS.

D. ANALYSIS OF COMPUTATIONAL COMPLEXITY

The complexity analysis of CS-AFSA is calculated below. CS-AFSA consists of two parts. The first part is STKA. In the STKA, what affects its complexity is the Modified AFSA. The complexity is $O(mn^2)$, where m represents the number of edge devices and n represents the number of mobile devices. In the Modified AFSA, the complexity of prey behavior plays

TABLE 2. Execution time of CS-AFSA.

the number of mobile devices	execution time /s
n=30	0.7520
n=40	0.8637
n=50	1.0202

an important role. In prey behavior, first of all, all mobile devices should be traversed, and each mobile device should be traversed again to find the optimal solution within the visual range. So, in $O(mn^2)$, n needs to be squared.

The second part is CS-AFSA. We call STKA in a loop statement. Therefore, it affects the complexity of CS-ASFA. The complexity is $O(\delta mn^2)$, where δ represents the number of candidate elements in WPTS.

Therefore, the complexity of our entire algorithm is $O(\delta mn^2)$.

Finally, the execution time of the proposed CS-AFSA is provided in Table 2. Please refer to Part V for the setting of system parameters. From Table 2, we can see that CS-AFSA can be implemented in real-time or close to a real-time manner.

V. SIMULATION

In this section, we use data simulations to evaluate the performance of the proposed CS-AFSA. This article studies the scheduling problem of the WP-MEC network with multiple edge devices to minimize the delay. We use mobile phones as mobile devices. The uplink frequency we use is 890-909MHz, and the downlink frequency is 935-954MHz. The bandwidth is 19MHz. First, we compare the proposed algorithm with two special cases.

Only Completely Offloading Scheduling (OCOS): In OCOS, we do not allow local calculations. Every task must be completely offloaded to one edge device and calculated. The rest of the algorithm is the same as CS-AFSA.

Only Local Computation Scheduling (OLCS): In OLCS, we set the WPT transmission time to T. All tasks that satisfy (6) can be computed locally, and those that are not satisfied will be given up in the time slot T.

Refer to [8] and [23] for the setting of simulation parameters. The length of the time slot T is $L = 0.5$s. The transmission power of each edge device is $P = 3$W. The local computing power of mobile device u_i is $P_i \in [0.02, 0.08]$ W.

The offloading transmission power from u_i to e_k is $P_{ik} \in [0.05, 0.15]$ W. The amount of data needed to be computed by the mobile device u_i in the time slot is $c_i \in [50,200]$ kb. At the same time, some system parameters are $\mu = 0.6$, $B = 1$MHz, $f_{max} = 0.5$GHz, $g_i = 1.4375$, $\kappa_i = 10^{-28}$, $\phi_i = 1000$cycles/bit, $f_{kmax} \in [f^*_{max}2, f^*_{max}3]$. There are also some AFSA related parameters: $afstep = 2$, $afvisual = 4$, $\delta = 0.0618$. After this, we evaluate the system performance of CS-AFSA through two parts: the network size and the system parameters.

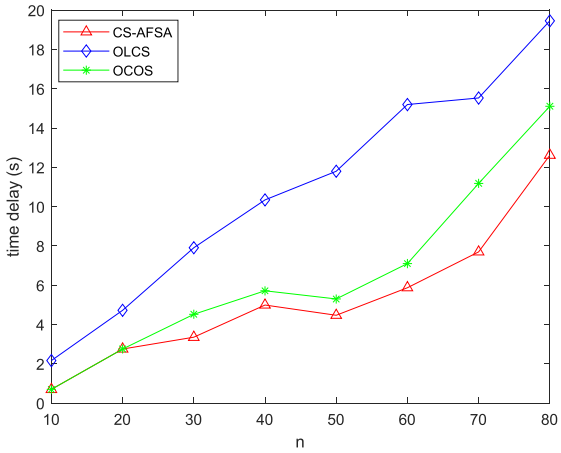

FIGURE 2. The impact of n on time delay.

A. THE IMPACT OF NETWORK SIZE

In this part, we evaluate the impact of network size on CS-AFSA, the OCOS, and the OLCS. In the following simulation, the system parameters of CS-AFSA are set to $T = 0.5$, $\beta = 0.6$, $\mu = 0.6$.

First, we evaluate the minimum delay of CS-AFSA. In this simulation, $m = 5$, n changes from 10 to 80. Fig. 2 compares the performance of the CS-AFSA, the OLCS, and the OCOS. In Fig. 2, we can see that with the increase of n, the minimum delay of the three algorithms tends to increase overall. This is because when the number of n increases, the number of tasks offloaded to each edge device increases, which causes the edge device to be overloaded and delays the completion of tasks. It can also be seen that when n is less ($n = 10$, and 20), there is basically no difference in performance between CS-AFSA and OCOS. The reason is that the processing speed of edge devices is much faster than that of mobile devices. When n is small, the delay caused by queuing is very small. Therefore, almost all the mobile devices in CS-AFSA choose to offload their tasks to edge devices, so the performance is close to OCOS. As n increases, the delay caused by queuing increases. Some mobile devices have to abandon offloading calculations and switch to local calculations. Therefore, the gap between OCOS and CS-AFSA has gradually increased.

Then in this simulation, $n = 30$ and m changes from 1 to 8. Fig. 3 compares the performance of the CS-AFSA, the OLCS, and the OCOS. In Fig. 3, we can see that as m increases, the minimum delay of OLCS is almost straight. It is because the increase in the number of edge devices has no impact on the system that can only compute locally. Since the amount of data c_i of our mobile device in the time slot T is randomly generated within 50kb-200kb, our simulation results are not a completely straight line. With the increase of m, the performance of OCOS and CS-AFSA are gradually getting better. Overall, the performance of our proposed CS-AFSA is better.

B. THE IMPACT OF SYSTEM PARAMETERS

In this part, we evaluate the impact of system parameters on the CS-AFSA. In the following simulation, when each

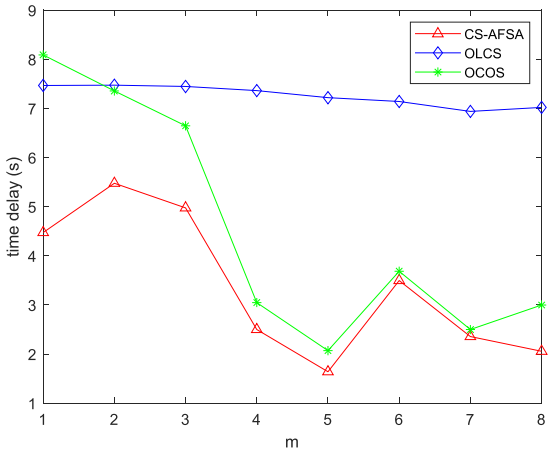

FIGURE 3. The impact of m on time delay.

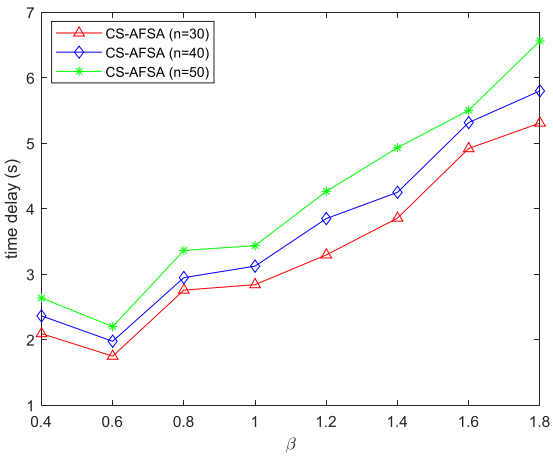

FIGURE 4. Time delay varies with β.

parameter changes, we draw three curves corresponding to $n = 30$, $n = 40$, and $n = 50$.

First, from Fig. 4, we can see that as β increases, the time delays of the three curves gradually become larger; when β remains unchanged and the three curves are compared, the CS-AFSA's time delay has gradually become larger. In CS-AFSA, we know that the generation of candidate values in WPTS is related to β. When β is too large or too small, it may affect the generation of WPTS, which in turn affects the value of τ_0, and affects the final time delay. We can also see that the system delay when $\beta = 0.6$ is the smallest.

Secondly, from Fig. 5, we can see that the system delay corresponding to $T = 0.5$ is the smallest. When $T > 0.5$ or $T < 0.5$, the delay increases to a certain extent. When T is too small, tasks of some mobile devices may not be able to complete offloading computation due to the time constraint, and can only choose local computation. When T is too large, tasks of some mobile devices that originally calculated locally have time to offload. The computational burden of edge devices is increased, and therefore the delay becomes longer.

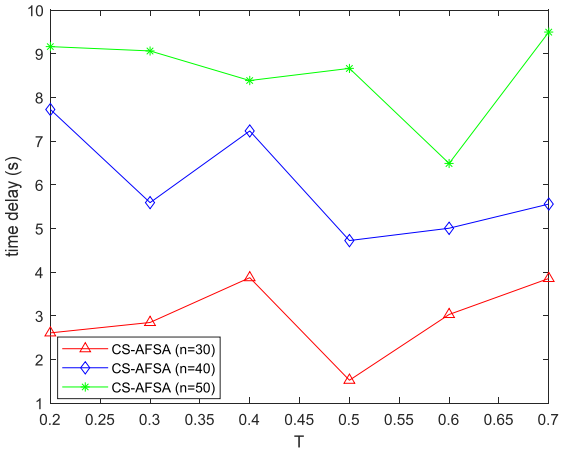

FIGURE 5. Time delay varies with T.

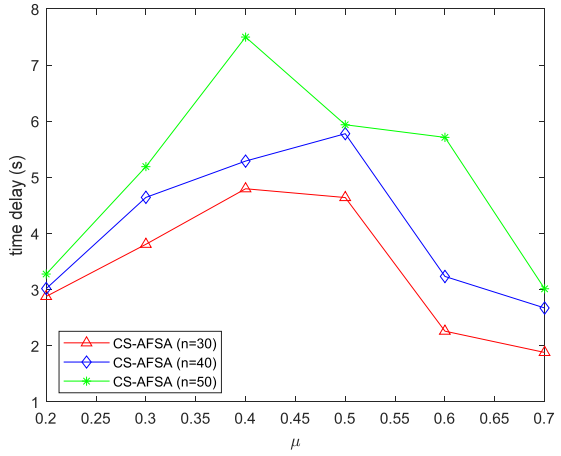

FIGURE 6. Time delay varies with μ.

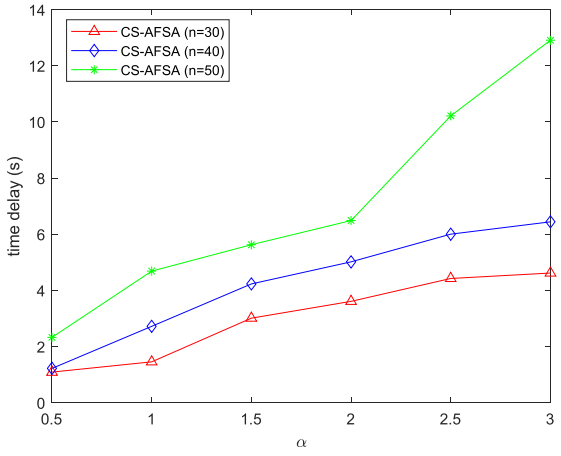

FIGURE 7. Time delay varies with different computation intensities.

Then we consider the impact of energy conversion efficiency μ on the proposed algorithm. From Fig. 6, we can see that when μ is large ($\mu > 0.5$), the system delay is small, and decreases as μ increases. In theory, we hope μ as large as possible, because $\mu = 1$ is equivalent to the mobile device

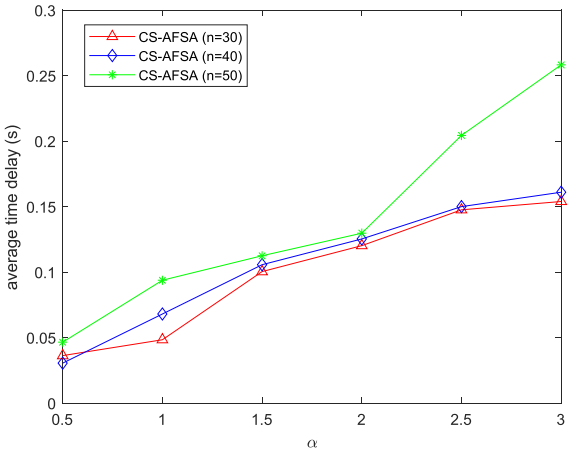

FIGURE 8. Average time delay varies with different computation intensities.

receiving all the energy transmitted by the edge device. It means that there is no loss of energy during transmission. However, in actual situations, due to interference from factors such as noise, mobile devices cannot receive the same amount of energy as the transmitting end. In this article, to obtain a better time delay, we set $\mu = 0.6$.

Finally, the impacts of different computational intensities are considered in Fig.7. We express the different computational intensities by multiplying the amount of data by a coefficient α. So that we can get some indicative scalability results. From Fig.7, we can see that with the increase of α, that is, the increase of calculation intensity, the time delay to complete all the calculations is also increasing. It must be pointed that time delay is the sum of the time it takes to complete all the tasks from the mobile devices. From Fig.8, we derive the average time delay required for a mobile device to complete its task, whether the task is locally calculated or offloaded to one edge device. For example, when $\alpha = 1$ and $n = 40$, the average time delay is 0.068s.

VI. CONCLUSION

In this paper, we study the scheduling problem of the WP-MEC network with minimum delay under the constraint of WPT transmission energy. We need to jointly consider WPT transmission time allocation and mobile device computing scheduling in the WP-MEC network, which is an NP-hard problem. Therefore, we propose CS-AFSA. The first step is to modify the Artificial Fish Swarm Algorithm and apply it to our WP-MEC network. Then we consider the offloading problem of the WP-MEC network when WPT transmission time τ_0 is given. STKA is proposed to solve this problem. Finally, the candidate set of WPT transmission time τ_0 is determined, and STKA is called for each element in the set, and finally CS-AFSA is formed. CS-AFSA changes the WPT transmission time while considering the calculation and scheduling of mobile devices to find the minimum delay that satisfies the constraints. Finally, extensive simulations are performed to verify the performance of the proposed algorithm.

REFERENCES

[1] X. Zheng, Z. Cai, J. Li, and H. Gao, "Scheduling flows with multiple service frequency constraints," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 496–504, Jun. 2016.
[2] 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, Aug. 2017.
[3] P. A. Apostolopoulos, E. E. Tsiropoulou, and S. Papavassiliou, "Cognitive data offloading in mobile edge computing for Internet of Things," *IEEE Access*, vol. 8, pp. 55736–55749, 2020.
[4] X. Fang, Z. Cai, W. Tang, G. Luo, J. Luo, R. Bi, and H. Gao, "Job scheduling to minimize total completion time on multiple edge servers," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2245–2255, Oct. 2020.
[5] X. Lu, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "Wireless networks with RF energy harvesting: A contemporary survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 757–789, 2nd Quart., 2015.
[6] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1757–1771, May 2016.
[7] 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.
[8] 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.
[9] 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, Sep. 2016.
[10] X. Hu, K.-K. Wong, and K. Yang, "Wireless powered cooperation-assisted mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2375–2388, Apr. 2018.
[11] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Apr. 2017.
[12] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Aug. 2016.
[13] X. Zheng, Z. Cai, J. Li, and H. Gao, "An application-aware scheduling policy for real-time traffic," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst.*, Jun. 2015, pp. 421–430.
[14] J. Yu, B. Huang, X. Cheng, and M. Atiquzzaman, "Shortest link scheduling algorithms in wireless networks under the SINR model," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2643–2657, Mar. 2017.
[15] X. Zheng, Z. Cai, J. Li, and H. Gao, "A study on application-aware scheduling in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1787–1801, Jul. 2017.
[16] D. Yu, Y. Zou, J. Yu, X. Cheng, Q.-S. Hua, H. Jin, and F. C. Lau, "Stable local broadcast in multihop wireless networks under SINR," *IEEE/ACM Trans. Netw.*, vol. 26, no. 3, pp. 1278–1291, May 2018.
[17] 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, Mar. 2017.
[18] T. Zhu, T. Shi, J. Li, Z. Cai, and X. Zhou, "Task scheduling in deadline-aware mobile edge computing systems," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4854–4866, Jun. 2019.
[19] D. B. Shmoys and É. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, nos. 1–3, pp. 461–474, Feb. 1993.
[20] G. T. Ross and R. M. Soland, "A branch and bound algorithm for the generalized assignment problem," *Math. Program.*, vol. 8, no. 1, pp. 91–103, Dec. 1975.
[21] R. Cohen, L. Katzir, and D. Raz, "An efficient approximation for the generalized assignment problem," *Inf. Process. Lett.*, vol. 100, no. 4, pp. 162–166, 2006.
[22] C. Chekuri and S. Khanna, "A polynomial time approximation scheme for the multiple knapsack problem," *SIAM J. Comput.*, vol. 35, no. 3, pp. 713–728, Jan. 2005.
[23] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Dec. 2018.

[24] M. Neshat, G. Sepidnam, M. Sargolzaei, and A. N. Toosi, "Artificial fish swarm algorithm: A survey of the state-of-the-art, hybridization, combinatorial and indicative applications," *Artif. Intell. Rev.*, vol. 42, no. 4, pp. 965–997, Dec. 2014.
[25] M. Z. Khan, S. Harous, S. U. Hassan, M. U. G. Khan, R. Iqbal, and S. Mumtaz, "Deep unified model for face recognition based on convolution neural network and edge computing," *IEEE Access*, vol. 7, pp. 72622–72633, 2019.
[26] Z. Zhou, C. Zhang, J. Wang, B. Gu, S. Mumtaz, J. Rodriguez, and X. Zhao, "Energy-efficient resource allocation for energy harvesting-based cognitive Machine-to-Machine communications," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 3, pp. 595–607, Sep. 2019.
[27] B. Ji, Y. Li, D. Cao, C. Li, S. Mumtaz, and D. Wang, "Secrecy performance analysis of UAV assisted relay transmission for cognitive network with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7404–7415, Jul. 2020.
[28] M. Y. Naderi, P. Nintanavongsa, and K. R. Chowdhury, "RF-MAC: A medium access control protocol for re-chargeable sensor networks powered by wireless energy harvesting," *IEEE Trans. Wireless Commun.*, vol. 13, no. 7, pp. 3926–3937, Jul. 2014.

TIANTIAN LI is currently pursuing the master's degree with the School of Information Science and Engineering, Shandong Normal University. Her current research interests include offloading algorithm, antenna array, and distributed network optimization.

FENG YANG received the B.S. and M.S. degrees in radio electronics from Shandong University, in 1985 and 1988, respectively. He is currently a Professor with the School of Information Science and Engineering, Shandong Normal University. He is also the Director of the Department of Communication Engineering. He has published more than 30 articles, edited textbooks, edited five books, obtained seven national invention patents, four utility model patents, and participated in one national fund project. He has presided over five provincial and university-level education reform projects. He was a recipient of the One Provincial-Level Teaching Achievement Award, the School-Level Teaching Achievement Award, and the three provincial awards for scientific and technological progress.

DEPENG ZHANG is currently pursuing the master's degree with the School of Information Science and Engineering, Shandong Normal University. His current research interests include deep learning and recommended systems.

LINBO ZHAI received the B.S. and M.S. degrees from the School of Information Science and Engineering, Shandong University, in 2004 and 2007, respectively, and the Ph.D. degree from the School of Electronic Engineering, Beijing University of Posts and Telecommunications, in 2010. Since then, he has been working as a Teacher with Shandong Normal University. His current research interests include cognitive radio, crowdsourcing, and distributed network optimization.