Received November 25, 2019, accepted December 7, 2019, date of publication December 18, 2019, date of current version December 30, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2960547*

# Mobile Edge Computing Based Task Offloading and Resource Allocation in 5G Ultra-Dense Networks

**XIN CHEN[1,2], ZHIYONG LIU[1], YING CHEN[1], AND ZHUO LI[1]**
[1]School of Computer Science, Beijing Information Science and Technology University, Beijing 100101, China
[2]Advanced Equipment Intelligent Perception and Control, Beijing International Cooperation Base for Science and Technology, Beijing 100045, China

Corresponding author: Ying Chen (hialary2017@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61872044 and Grant 61902029, in part by the Key Research and Cultivation Projects, Beijing Information Science and Technology University under Grant 5211910958, in part by the Supplementary and Supportive Project for Teachers, Beijing Information Science and Technology University under Grant 5111911128, in part by the Beijing Municipal Program for Top Talent Cultivation under CIT and TCD201804055, and in part by the Open Program of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research under Grant ICDDXN001.

**ABSTRACT** Driven by the vision of 5G communication, the demand for mobile communication services has increased explosively. Ultra-dense networks (UDN) is a key technology in 5G. The combination of mobile edge computing (MEC) and UDN can not only cope with access from mass communication devices, but also provide powerful computing capacity for users at the edge of wireless networks. The UDN based on MEC can effectively process computation-intensive and data-intensive tasks. However, when a large number of users offload tasks to the edge server, both the network load and transmission interference would increase. In this paper, the problem of task offloading and channel resource allocation based on MEC in 5G UDN is studied. Specifically, we formulate task offloading as an integer nonlinear programming problem. Due to the coupling of decision variables, we propose an efficient task offloading and channel resource allocation scheme based on differential evolution algorithm. Simulation results show that the proposed scheme can obviously reduce energy consumption and has good convergence.

**INDEX TERMS** 5G, ultra-dense network (UDN), mobile edge computing (MEC), task offloading, resource allocation.

## I. INTRODUCTION

The rapid development of mobile Internet and Internet of Things has driven the explosive growth in the demand for mobile communication services. Global mobile data traffic in the 5G era is expected to be more than 1,000 times that in the 4G era [1]. The growing demand for mobile communications means that mobile devices connected to wireless networks will also grow rapidly. Massive mobile device communication leads to increasingly dense access points [2]. Ultra-dense network (UDN) came into being. Moreover, mobile devices have become smarter. Smart mobile devices have given rise to various emerging services such as autonomous driving, interactive games, virtual reality and augmented reality. But smarter businesses usually have higher computing requirements [3]. In order to achieve portability, mobile devices sacrifice some performance, so that they cannot fully meet the requirements of intelligent interactive applications for time delay, energy consumption and so on [4]. Mobile edge computing (MEC) is regarded as a promising technology to solve these problems [5]. MEC allows for the computation tasks to be executed at the nodes so that resource-constrained mobile devices can execute compute-intensive and data-intensive applications in real time.

UDN can meet the access of massive user equipment (UE) and improve the network capacity. MEC can process computation-intensive and data-intensive tasks in real time [6]. The combination of UDN and MEC can provide more UEs with immediate computation ability, which can not only meet the task requirements in computing power, but also reduce the energy consumption in UE processing tasks. In addition, the disadvantage of offloading tasks to the remote cloud with high latency is avoided [7]. However, in the UDN, sharing limited channel resources among a large number

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

of UEs would cause severe interference, which results in a decrease in the transmission rate [8]. The time delay and energy consumption of task offloading to edge server depends on transmission rate. Therefore, offloading decisions need to be made to select which tasks to perform on the local device to ensure the overall performance of the network system. In addition, how to allocate effective channel resources for devices is also worth discussing.

Offloading decision in MEC and channel resource allocation in UDN affect quality of service (QoS) of network and quality of experience (QoE) of users [9]–[11]. In MEC, task offloading has become a hot topic. Some work studies how mobile devices in MEC network make offloading decisions to improve task execution delay, energy consumption and offloading efficiency [12]–[14]. In addition, the resource allocation problem in UDN also received high attention. Some work has studied the resource allocation problem in UDN, and improved the system performance by allocating resources such as spectrum and power to reduce interference [15]–[18]. However, very few efforts combine UDN with MEC system features to consider both task offloading and resource allocation.

Modern intelligent optimization algorithm is widely used in solving optimization problems [19]–[22]. As a global optimization algorithm, differential evolution algorithm has strong global convergence ability and robustness, which is very suitable for solving optimization problems in some complex environments. The basic idea of this algorithm is to sum the vector difference between any two individuals and the third individual in the randomly generated initial population to generate a new individual. The new individual is compared with the corresponding individual in the contemporary population. Through continuous evolution, excellent individuals are retained and inferior individuals are eliminated, thus leading the search to the optimal solution [23]. Many researchers have proposed DE-based algorithms to solve resource allocation and task scheduling problems [24]–[26]. Not only has the complexity of the problem been reduced, but the optimized system performance has also been significantly improved.

In this paper, the problem of task offloading and channel resource allocation based on MEC in UDN is studied. The purpose is to minimize the system energy consumption under the constraint of the maximum computing delay. Considering the complexity of the problem and the coupling of decision variables, we propose a scheme based on differential evolution algorithm to solve this optimization problem. Finally, the simulation results verify the performance improvement and convergence of our scheme. The main contributions of this paper are as follows:

- We propose a MEC task offloading and resource allocation model based on heterogeneous UDN. The task offloading and channel resource allocation are optimized when MUE and SUE exist simultaneously.
- Then, we present an efficient task offloading and resource allocation scheme. By means of variable substitution, the problem is transformed without changing the essence of the problem, and the complexity of solving the problem is reduced. On this basis, the channel resource allocation algorithm (CRADE) based on differential evolution algorithm is used to give the most effective task offloading and resource allocation scheme.
- A large number of experiments have been conducted to evaluate the performance of CRADE scheme. The experimental results show that CRADE not only has an obvious effect in optimizing the system energy consumption in various scenarios, but also proves that this algorithm has a great convergence.

The remainder of this paper is as follows. In Section II, we review the related works. The system model and optimization problem are described in Section III. We propose the optimal offloading decision and channel resource allocation scheme in section IV. In Section V, we discuss the simulation results. Finally, Section VI concludes this paper and discusses the future work.

## II. RELATED WORK

Task offloading and resource allocation are of great significance in MEC and UDN networks, which have attracted more and more attention in recent years.

### A. TASK OFFLOADING IN MEC

Hao *et al.* [27] proposed an efficient and energy-saving scheme based on alternating iteration algorithm for tasks caching and offloading of mobile devices in MEC. You *et al.* [28] studied the resource allocation of multi-user MEC offloading system based on time division multiple access and orthogonal frequency division multiple access and proposed a sub-optimal resource allocation algorithm to optimize system energy consumption. Yang *et al.* [29] modeled the offloading energy consumption from the aspects of task computation and communication, and proposed a scheme based on artificial fish swarm algorithm to solve the energy consumption optimization problem. Sun *et al.* [30] designed a new index called computational efficiency to evaluate the performance of MEC systems, and used iterative and gradient descent methods to optimize system energy consumption. These works considered the problem of task offloading in MEC, but did not consider the problem of network density, resulting in that they were difficult to apply to the more complex environment in the UDN.

### B. RESOURCE ALLOCATION IN UDN

Zhou *et al.* [15] made localized prediction of traffic load of UDN base station by using deep LSTM learning technology, which can avoid or alleviate congestion in an intelligent way. Xu *et al.* [31] proposed an improved version of the intelligent method based on the non-dominant sorting genetic algorithm II, which combined the allocation of transmitted power and resource blocks to improve energy efficiency and spectral efficiency. Liu *et al.* [32] developed a two-step joint clustering and scheduling scheme to optimize

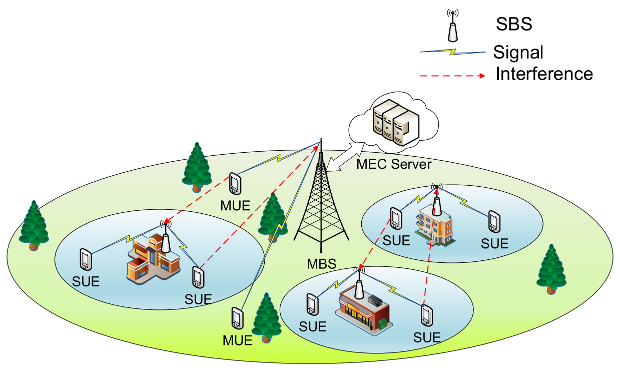


**FIGURE 1.** 5G Ultra-Dense Network Scenario.

inter-cell resource scheduling. Zhang *et al.* [33] designed an enhancement learn-based downlink power control algorithm to manage the interference of ultra-dense small cellular networks. These studies deeply considered the resource allocation in the UDN. However, if a large number of devices have computation-intensive or data-intensive tasks, these tasks are difficult to ensure that the requirements of the device can be processed efficiently in real time.

### C. RESOURCE ALLOCATION AND MEC TASK OFFLOADING IN UDN

Chen and Hao [12] proposed an effective software-defined task offloading scheme to study the task offloading problem in the UDN and minimized the task delay under the constraint of energy consumption. Mao *et al.* [34] designed a dynamic computational offloading algorithm based on Lyapunov optimization to optimize the task execution cost. Guo *et al.* [35] developed a heuristic greedy offloading scheme and studied the MEC offloading problem in UDN to minimize the overall energy consumption of mobile devices. Zhang *et al.* [36] proposed an iterative search algorithm that combined internal penalty functions with DC (difference between two convex functions/sets) programming to obtain optimal offloading decisions and resource allocation for single-cell and multi-cell networks. In these works, the task offloading within MEC and resource allocation problem in the UDN were well studied. However, they did not consider the cross-tier interference caused by the heterogeneity in the UDN.

## III. SYSTEM MODEL AND PROBLEM FORMULATION

In Fig. 1, we present the system model of 5G ultra-dense network architecture, in which a two-tier heterogeneous cellular network is shown, including the macro cell tier and the small cells tier. Table 1 lists major the notations and definitions used in this paper.

### A. COMMUNICATION ARCHITECTURE

In the network scenario, $N$ small cells are distributed in a macro cell. A macro base station (MBS) is placed in the center of the macro cell and a small base station (SBS) is deployed in

**TABLE 1.** Table of notations.

| Notation | Definition |
|---|---|
| $\mathcal{N}$ | The set of SBSs |
| $\mathcal{U}_n$ | The set of SUEs for each SBS service |
| $\mathcal{U}_M$ | The set of macro MUEs |
| $\mathcal{C}$ | The set of all BS |
| $\mathcal{K}$ | The set of channels |
| $B$ | The total bandwidth |
| $p_{u_M}$ | The transmission power between the MUE $u_M$ and MBS |
| $g_{u_M}^k$ | The channel gain from MUE $u_M$ to the MBS |
| $\sigma^2$ | The variance of additive white gaussian noise |
| $x_{u_n}^k$ | The channel allocation indicator |
| $Q_{u_c}$ | Computation task of UE |
| $\omega_{u_c}$ | Computation amount of task $Q_{u_c}$ |
| $s_{u_c}$ | Size of input computation task $Q_{u_c}$ |
| $T_{u_c}^{\max}$ | The maximum tolerance delay |
| $f_{u_c}^l$ | The CPU computing capability of the local UE |
| $f^C$ | The computation resources that MEC allocates to task |
| $\rho$ | The coefficient that depends on the chip architecture |
| $\alpha_{u_c}$ | Task decision indicator |

the center of each small cell. A MEC server on the MBS can perform multiple computing-intensive tasks. The set of SBS is expressed as $\mathcal{N} = \{1, 2, \ldots, n, \ldots, N\}$. The set of small user equipments (SUEs) for each SBS service is represented by $\mathcal{U}_n = \{1, 2, \ldots, u_n, \ldots, U_n\}(n \in \mathcal{N})$, and the set of macro UE (MUE) as $\mathcal{U}_M = \{1, 2, \ldots, u_M, \ldots, U_M\}$. It is assumed that the total bandwidth $B$ is divided into $K$ channels. Channels set can be represented by $\mathcal{K} = \{1, 2, \ldots k, \ldots, K\}$.

In our work, we assume that channels are allocated orthogonally within each cell but can be reused by multiple cells. UE is assigned to a channel during uplink transmission. Therefore, SUE who reuses the same channel resources will cause interference to the MBS when an MUE $u_M$ reuses uplink channel resources. In the macro cell, when the channel $k$ is allocated to the MUE $u_M$, its signal to interference-plus-noise ratio (SINR) is given by

$$\gamma_{u_M}^k = \frac{p_{u_M} g_{u_M}^k}{\sum_{n=1}^{N} \sum_{u_M=1}^{U_M} x_{u_n}^k p_{u_n} g_{u_n}^k + \sigma^2}, \tag{1}$$

where $p_{u_M}$ is the transmission power between the MUE $u_M$ and MBS, $g_{u_M}^k$ is the channel gain from MUE $u_M$ to the MBS, $\sigma^2$ is the variance of additive white gaussian noise (AWGN). $x_{u_M}^k$ is a binary variable that represents the channel allocation indicator. That is

$$x_{u_n}^k = \begin{cases} 1, & \text{the channel } k \text{ is allocated to the SUE } u_n, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Similarly, in the small cell $n$, when SUE $u_n$ reuses uplink channel resources, MUE who reuses the same channel resources will cause interference to SBS $n$, and SUE $u_i$ who reuses the same channel resources in other small cells will also cause interference to SBS $n$. Therefore, the SINR of the SUE $u_n$ in the uplink transmission process can be expressed as

$$\gamma_{u_n}^k = \frac{p_{u_n} g_{u_n}^k}{\sum_{u_M=1}^{U_M} x_{u_M}^k p_{u_M} g_{u_M}^k + \sum_{\substack{i=1,\\ i\neq n}}^{N} \sum_{u_i=1}^{U_i} x_{u_i}^k p_{u_i} g_{u_i}^k + \sigma^2}. \tag{3}$$

According to the SINR expression of MUE $u_M$ and SUE $u_n$, the uplink rate of MUE $u_M$ in the channel $k$ can be calculated by Shannon formula as

$$R_{u_M}^k = \frac{B}{K}\log_2(1+\gamma_{u_M}^k), \tag{4}$$

and the uplink rate of SUE $u_n$ in the channel $k$ can be expressed as

$$R_{u_n}^k = \frac{B}{K}\log_2(1+\gamma_{u_n}^k). \tag{5}$$

The rate of MUE $u_M$ is given by

$$R_{u_M} = \sum_{k=1}^{K} x_{u_M}^k R_{u_M}^k. \tag{6}$$

Similarly, the rate of SUE $u_n$ can be expressed as

$$R_{u_n} = \sum_{k=1}^{K} x_{u_n}^k R_{u_n}^k. \tag{7}$$

### B. COMPUTATION MODEL

We denote the set of all BS as $\mathcal{C} = \{1, 2, \ldots, N, M\}$. MUE and SUE are collectively referred to as local UE $u_c \in \mathcal{U}_c(c \in \mathcal{C})$. Each UE carries a computation task to be executed locally or offloading to the edge server. We adopt the commonly used task model to represent task $Q_{u_c}$, i.e., $Q_{u_c} = \left\{\omega_{u_c}, s_{u_c}, T_{u_c}^{\max}\right\}$, where $\omega_{u_c}$ is computation amount of this task $Q_{u_c}$, i.e., the total number of CPU cycles needed in accomplish the computation task, $s_{u_c}$ denotes size of input computation task $Q_{u_c}$, and $T_{u_c}^{\max}$ stands for the maximum tolerance delay. When a task is computed, it can choose whether to compute the task locally or offload it to the edge server.

#### 1) LOCAL COMPUTING

If the task is computed locally, we define $f_{u_c}^l$ as the CPU computing capability of the local UE. Therefore, the local execution time of the task $Q_{u_c}$ can be calculated as

$$T_{u_c}^L = \frac{\omega_{u_c}}{f_{u_c}^l}. \tag{8}$$

The local execution energy consumption of task $Q_{u_c}$ can be expressed as

$$\varepsilon_{u_c}^L = \rho (f_{u_c}^l)^2 \omega_{u_c}, \tag{9}$$

where $\rho = 10^{-26}$, and it is a coefficient whose value depends on the chip architecture [36].

#### 2) MOBILE EDGE COMPUTING

For mobile edge computing, UE in uplink transmission task will generate transmission delay and energy consumption. The communication between the MBS and the SBS is conducted by wired optical fiber. When the task $Q_{u_c}$ is transmitted to the MEC server through the SBS, the transmission delay and energy consumption between the MEC server and the SBS are ignored. According to the uplink rate of MUE, the uplink transmission delay for UE offloading the task $Q_{u_c}$ to edge server can be expressed as

$$T_{u_c}^t = \frac{s_{u_c}}{R_{u_c}} = \frac{s_{u_c}}{\sum_{k=1}^{K} x_{u_c} R_{u_c}^k}, \tag{10}$$

where $R_{u_c}^k \in R_{u_M}^k \bigcup R_{u_n}^k$. Then we can obtain that transmission energy consumption for UE offloading the task $Q_{u_c}$ to edge server is

$$\varepsilon_{u_c}^t = p_{u_c} T_{u_c}^t. \tag{11}$$

After the computation task is offloaded to MEC server, MEC server allocates some computation resources to the task. We set up MEC to assign a fixed number of computing resources to each task [36]. $f^C$ stands for the computation resources that MEC allocates to each computation task. Therefore, the edge server execution time for UE offloading the task $Q_{u_c}$ to edge server can be expressed as

$$T_{u_c}^E = \frac{\omega_{u_c}}{f^C}. \tag{12}$$

In this work, we mainly consider the energy consumption and time delay of the UE, so the calculation energy consumption of the MEC server is ignored [12]. Similar to [37], we ignore the transmission energy consumption and delay when the result of the task was returned. This is because the processed data size is very small.

Therefore, the total execution time of the task $Q_{u_c}$ can be calculated as

$$T_{u_c} = \left[(1-\alpha_{u_c})T_{u_c}^L + \alpha_{u_c}(T_{u_c}^t + T_{u_c}^E)\right], \tag{13}$$

where $\alpha_{u_c}$ is a binary variable that indicates whether the task $Q_{u_c}$ is processed locally or in the edge server. That is

$$\alpha_{u_c} = \begin{cases} 1, & \text{the task } Q_{u_c} \text{ is processed on the edge server,} \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

The total execution energy consumption of task $Q_{u_c}$ can be expressed as

$$\varepsilon_{u_c} = \left[(1-\alpha_{u_c})\varepsilon_{u_c}^L + \alpha_{u_c}\varepsilon_{u_c}^t\right]. \tag{15}$$

### C. PROBLEM FORMULATION

Our goal is to minimize the total energy consumption of the system with a limited tolerance delay. The total energy consumption of the system can be expressed as

$$\varepsilon = \sum_{u_M=1}^{U_M} \varepsilon_{u_M} + \sum_{n=1}^{N}\sum_{u_n=1}^{U_n} \varepsilon_{u_n}, \tag{16}$$

where $\varepsilon_{u_M}$ and $\varepsilon_{u_n}$ are the energy consumption generated by task of MUE and SUE when they are executed, respectively. There are two problems to consider:

1) Task decision problem: decide whether the task will be processed locally or on the edge server.
2) Channel resource allocation problem: the task chooses which channel to reuse when offloading to the edge server.

Finally, the task decision problem and channel resource allocation problem can be expressed as

$$P1 : \min_{\alpha,\mathbf{x}} \varepsilon \tag{17}$$

$$\text{s.t. } T_{u_c} \le T_{u_c}^{\max}, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \tag{17a}$$

$$\varepsilon_{u_c} \le \kappa_{u_c} E_{u_c}^{\max}, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \tag{17b}$$

$$\alpha_{u_c} \in \{0,1\}, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \tag{17c}$$

$$x_{u_c}^k \in \{0,1\}, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C},\ \forall k \in \mathcal{K}, \tag{17d}$$

$$\sum_{u_c=1}^{U_c} x_{u_c}^k \le 1, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C},\ \forall k \in \mathcal{K}, \tag{17e}$$

$$\sum_{k=1}^{K} x_{u_c}^k \le 1, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C},\ \forall k \in \mathcal{K}. \tag{17f}$$

The objective function (17) is to minimize the total energy consumption of the system. The first constraint (17a) guarantees that the execution time of the task is less than the maximum tolerable delay of the task. The second constraint (17b) means that the total execution energy consumption of task is less than the remainder battery capacity of UE, where $\kappa_{u_c}$ is the ratio of the remaining energy of the UE to the total battery capacity. The third constraint (17c) indicates that tasks can only be executed locally or offloaded to MEC server. The fourth constraint (17d) means that the allocation state of channel resources. The fifth constraint (17e) indicates that each channel is only allocated to one UE in each small cell or macro cell. The last constraint (17f) states that each UE can be assigned to one channel at most.

## IV. OPTIMAL OFFLOADING DECISION AND CHANNEL RESOURCE ALLOCATION SCHEME

In this section, we propose an efficient task offloading decision and channel resource allocation scheme based on the optimization problem in (17). According to the communication model and calculation model in section III, we can see that the offloading decision of UE is coupled with channel allocation. If there are too many UE that simultaneously reuse the same wireless channel to offload computation tasks to the MEC server, each UE in this channel will receive very serious interference, resulting in a lower uplink communication rate. When the uplink rate of UE is low, the delay of task transfer to the MEC server will increase, and the energy consumption will also increase. In this case, it is more advantageous for the task to be performed locally.

However, performing tasks locally takes up a lot of system resources for UE, and performing other tasks can be especially difficult. The computing resources of the MEC server are also wasted. Therefore, an efficient offloading decision and channel resource allocation scheme are needed to ensure the execution of tasks with low delay and energy consumption.

In the UDN, the number of small cells and UEs is huge, so it is very difficult to find the best task offloading decision and channel resource allocation scheme.

*Theorem 1:* The problem $P1$ that computes the minimize the total energy consumption of the system is NP-hard.

*Proof:* We first introduce the bin packing problem [38]. There are $N$ bins of the same size with a capacity of $C$ and $M$ items of the same size of $v_i (i \in 1, 2, \ldots, M)$. Our goal is to allocate the maximum number of items to a fixed number of boxes without violating the capacity constraint. This problem can be expressed as

$$\max \sum_{i=1}^{M}\sum_{j=1}^{N} v_i \alpha_i^j \tag{18}$$

$$\text{s.t. } \sum_{i=1}^{M} v_i \alpha_i^j \le C, \quad \forall i = 1, 2, \ldots, M, \quad \forall j = 1, 2, \ldots, N, \tag{18a}$$

$$\alpha_i^j \in \{0,1\},\ \forall i = 1, 2, \ldots, M,\ \forall j = 1, 2, \ldots, N, \tag{18b}$$

$$\sum_{j=1}^{N} \alpha_i^j \le 1, \quad \forall i = 1, 2, \ldots, M. \tag{18c}$$

It is known from [38] that the bin packing problem above is NP-hard.

Our problem of channel resource allocation can be seen as a bin packing problem. The channel resources and UE in this article can be regarded as bins and items in bin packing problem, respectively. The product of transmitted power and channel gain can be thought of as the size of the item in bin packing problem.

Therefore, if we have an algorithm that can find the optimal solution of the channel allocation problem, then we can also get the optimal solution of the packing problem. Since the packing problem is NP-hard, the channel assignment problem is also NP-hard. The offloading decision problem is similar. This proves that the problem of minimizing the total energy consumption of the system is also NP-hard. □

### *A. PROBLEM TRANSFORMATION*

In this paper, the offloading decision of UEs is coupled with channel allocation. If $P1$ problem is decoupled, the problem is decomposed into two sub-problems to find the optimal solution of offloading decision and channel resource allocation respectively. Finally, the optimal solution obtained by decoupling may only be the sub-optimal solution of the target problem, and the energy consumption of the system cannot be minimized. Therefore, we decide to convert the optimization problem of two decision variables into a univariate optimization problem by variable substitution.

*Theorem 2:* The offloading decision of UE can be determined by its allocation status among all channel resources. That is

$$\alpha_{u_c} = \sum_{k=1}^{K} x_{u_c}^k. \tag{19}$$

*Proof:* If UE $u_c$ offloads the computation task to the edge server, $\alpha_{u_c} = 1$. In this case, the UE $u_c$ must reuse a channel in the cell during uplink communication with the BS, i.e., $\sum_{k=1}^{K} x_{u_c} = 1$. If UE $u_c$ chooses to execute the computation task locally, $\alpha_{u_c} = 0$. At this point, UE $u_c$ does not communicate with the base station and therefore it does not reuse any channel resources. i.e. $\sum_{k=1}^{K} x_{u_c} = 0$. Therefore, $\alpha_{u_c} = \sum_{k=1}^{K} x_{u_c}^k$. □

By making $\alpha_{u_c} = \sum_{k=1}^{K} x_{u_c}^k$, the total execution energy consumption of all MUEs can be converted into

$$\varepsilon_{u_M} = \sum_{u_M=1}^{U_M} \left[ (1 - \sum_{k=1}^{K} x_{u_M}^k)\varepsilon_{u_M}^L + (\sum_{k=1}^{K} x_{u_M}^k)\varepsilon_{u_M}^t \right], \tag{20}$$

and the total execution energy consumption of SUE in small cell can be converted into

$$\varepsilon_{u_n} = \sum_{u_n=1}^{U_n} \left[ (1 - \sum_{k=1}^{K} x_{u_n}^k)\varepsilon_{u_n}^L + (\sum_{k=1}^{K} x_{u_n}^k)\varepsilon_{u_n}^t \right]. \tag{21}$$

The original optimization problem $P1$ can be transformed into the following optimization problem:

$$P2 : \min_{\mathbf{x}} \varepsilon_{u_M} + \sum_{n=1}^{N} \varepsilon_{u_n} \tag{22}$$

$$\text{s.t. } T_{u_c} \le T_{u_c}^{\max}, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \tag{22a}$$

$$\varepsilon_{u_c} \le \kappa_{u_c} E_{u_c}^{\max}, \quad \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \tag{22b}$$

$$x_{u_c}^k \in \{0, 1\},\ \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C},\ \forall k \in \mathcal{K}, \tag{22c}$$

$$\sum_{u_c=1}^{U_c} x_{u_c}^k \le 1,\ \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \forall k \in \mathcal{K}, \tag{22d}$$

$$\sum_{k=1}^{K} x_{u_c}^k \le 1,\ \forall u_c \in \mathcal{U}_c,\ \forall c \in \mathcal{C}, \forall k \in \mathcal{K}. \tag{22e}$$

Now, the target problem is transformed into a problem of allocating only channel resources. If the UE chooses to reuse channel resources, this means that the UE offloads the computation task to the MEC server. If UE does not choose to reuse channel resources, the UE executes the computation task locally.

### *B. CHANNEL RESOURCE ALLOCATION*

For the transformed optimization problem, the decision variable $x_{u_c}^k$ has not changed, so problem $P2$ is still NP-hard. Exhaustive method can get exact solution when solving NP-hard problem, but it is not applicable when the result of large-scale problem is too complicated. Therefore, we propose a offline channel resource allocation algorithm based on differential evolution algorithm (CRADE).

Differential evolution (DE) algorithm is an important branch of evolutionary algorithm (EA). EA is a kind of group-oriented random search method by simulating the process and mechanism of biological gene inheritance and population evolution in nature. EA can automatically acquire the knowledge of search space in the process of search, and accumulate the effective knowledge of search space to reduce the search scope. Therefore, the EA can control the search process adaptively, reduce the complexity of the problem dynamically and obtain the optimal solution of the original problem.

Like other EAs such as genetic algorithm, DE is an optimization algorithm based on modern intelligence theory. The DE guides the direction of search optimization through the cooperation and competition among individuals in the group. In particular, in differential evolutionary algorithms, changes in each gene position depend only on differences between other individuals, taking full advantage of information about other individuals in the population. This can not only expand the diversity of the population, but also avoid the randomness and blindness caused by the operation of variation in the individual. In the process of searching optimization, the group of differential evolution calculation has the characteristic of collaborative search and strong searching ability. Therefore, differential evolution algorithm has strong global convergence ability and robustness, which is very suitable for solving optimization problems in some complex environments. Through continuous evolution, the excellent individuals are retained and the inferior ones are eliminated, and the search is guided to the optimal solution.

In the following, the realization process of CRADE algorithm is introduced in detail.

#### 1) POPULATION INITIALIZATION

Randomly generating $N_p$ individuals in the solution space, which can be expressed as

$$\left\{ X_i(0) | x_{i,j}^L \le x_{i,j}(0) \le x_{i,j}^U \right\}, \tag{23}$$

**Algorithm 1** Population Initialization Algorithm

```
Input: Population size: $N_p$; Individual dimensions: $D$;
       Upper and lower bounds of decision variables:
       $x_{i,j}^U$, $x_{i,j}^L$
Output: Initial population
1 some description;
2 for $i = 1$ to $N_p$ do
3     for $j = 1$ to $D$ do
4         $x_{i,j}(0) = x_{i,j}^L + rand\{0, 1\}(x_{i,j}^U - x_{i,j}^L)$
5     end
6 end
```

where $x_{i,j}^L = 0$ and $x_{i,j}^U = 1$ denotes lower and upper bounds of decision variables, respectively. $i = 1, 2, \ldots, N_p$, which is size of population, $j = 1, 2, \ldots, D(D = (N \times U_n + U_M) \times K)$, which is the dimension of the objective decision variable. Here, we carry out dimensionality reduction expansion for decision variables $x_{u_c}^k$ to facilitate calculation. The population initialization method is shown in Algorithm 1.

In this paper, the objective decision variable is binary discrete variable. Therefore, $rand\{0, 1\}$ denotes that the random value is "0" or "1".

### 2) MUTATION OPERATOR

The CRADE algorithm realizes individual variation through differential strategy. The difference strategy used by CRADE is to randomly select two different individuals in the population, and then scale their vector difference to make vector synthesis with the individual to be mutated. The difference strategy can be expressed as

$$V_i(g+1) = x_{r1}(g) + F \times (x_{r2}(g) - x_{r3}(g)), \tag{24}$$

where $r1, r2, r3$ are three unequal random numbers, with the value interval of $[1, N_p]$, $F$ is the zoom factor, and $g$ denotes the g-generation population.

The mutation operator can control the convergence speed of the population by increasing the diversity of the population and reduce the risk of falling into the local optimal solution in the optimization process. In the process of evolution, the validity of the solution must be guaranteed. When performing the mutation operation, it is necessary to determine whether the genes of each individual in the population meet the boundary conditions. If the boundary condition is not met, the gene is regenerated in the same way as the initial population.

### 3) CROSSOVER OPERATOR

In order to maintain the diversity of the population, the parent individual $X_i(g)$ and the offspring individual $V_i(g+1)$ after the difference variation carried out the crossover operation.The cross mode gram is expressed as

$$u_{i,j}(g+1) = \begin{cases} x_{i,j}(g), & rand_{i,j} \le P_c \text{ or } j = J_{rand}, \\ v_{i,j}(g+1), & rand_{i,j} \ge P_c \text{ or } j \ne J_{rand}, \end{cases} \tag{25}$$

**Algorithm 2** Channel Resource Allocation Algorithm Based on Differential Evolution Algorithm (CRADE)

```
Input: Model parameters; Algorithm parameters: $N_p$,
       $D$, $G$
Output: Optimal channel allocation matrix: $\mathbf{x}$;
        System minimum energy consumption: $f(\mathbf{x})$
1  Initialize population based on Algorithm 1,
   generation $g = 0$;
2  while $g \le G$ do
3      for $i = 1$ to $N_p$ do
4          Mutation Process;
5          Based on (24) produce mutation individuals;
6          for $j = 1$ to $D$ do
7              if $x_{i,j}^U \le v_{i,j}(g) \le x_{i,j}^U$ then
8                  The mutation individuals gene is
                   $v_{i,j}(g)$;
9              else
10                 Individual genes are regenerated
                   based on the Algorithm 1;
11             end
12         end
13         Crossover Process;
14         Two individual genes were randomly selected
           for crossover based on (25);
15         Selection Process;
16         if individual satisfies the constraint in $P2$ then
17             According to (27), select the parent of the
               next generation;
18         end
19     end
20     $g \leftarrow g + 1$
21 end
```

where $P_c$ is crossover probability. The purpose of introducing $J$ is to force at least one gene of progeny individuals to be provided by the new mutated individual, so as to improve the diversity of the population. The diversity of population ensures the evolution of individuals, thus avoiding premature convergence in the optimization process of resource allocation.

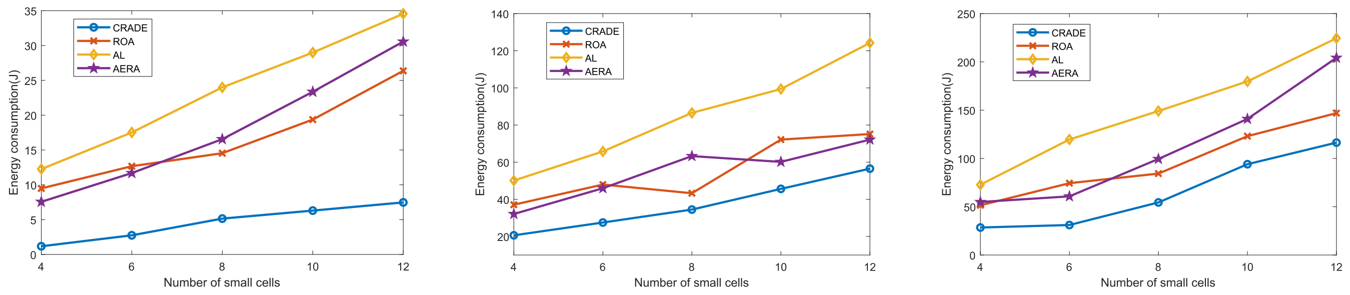
### 4) SELECTION OPERATOR

According to question $P2$, the fitness function is as follows

$$f(\mathbf{x}) = \varepsilon_{u_M} + \sum_{n=1}^{N} \varepsilon_{u_n}. \tag{26}$$

After the mutation and crossover operation, the offspring and the parent were substituted into fitness function for comparison, and the individuals with high fitness were selected as the parents of the next generation through greedy strategy. Greedy strategy selection method can be expressed as

$$X_i(g+1) = \begin{cases} U_i(g+1), & f(U_i(g+1)) \le f(X_i(g)), \\ X_i(g), & \text{Otherwise.} \end{cases} \tag{27}$$



(a) Number of SUE in a small cell: 2 (b) Number of SUE in a small cell: 6 (c) Number of SUE in a small cell: 10

**FIGURE 2. Impact of environment parameters.**

To summarize all the above discussions, we propose CRADE algorithm to solve our optimization problem, as shown in Algorithm 2.

The complexity of CRADE algorithm is related to dimension. In this model, the complexity of the four operations (i.e., initialization, mutation, crossover, selection) is proportional to $N_p$ at each evolution, and $N_p$ is related to the size of dimension $D$. Therefore, the complexity of CRADE algorithm can be calculated as $\Theta(G * 4 * N_p * D) = \Theta(D^2)$.

## V. PERFORMANCE EVALUATION

In this section, we evaluate the CRADE algorithm for solving the task offloading and channel resource allocation problem. The CRADE algorithm, proposed in this paper, is compared with three different offloading strategies: (i) random offloading and allocation scheme (ROA); (ii) all local scheme (AL); (iii) all MEC and random allocation scheme (AERA). The experimental results are mainly divided into three parts. Firstly, we compare and evaluate the influence of network environment indicators (i.e., number of SBS, number of MUE and number of SUE) on task offloading and channel allocation performance. Secondly, we compare and evaluate the impact of task indicators (i.e., task size and task computation) on performance. Finally, we evaluate the convergence of the algorithm.

### A. SIMULATION ENVIRONMENT

We consider a UDN with a macro cell and $N$ small cells. The diameter of the macro cell is 500 m. The small cell diameter is 100 m, randomly distributed in the macro cell. Similar to [39], the wireless channel considered is the Rayleigh fading channel, where the channel gain $g_{u_c}^k$ follows an exponential distribution with a mean value of 1. The MEC server is deployed in the MBS, and it has 4 GHz of computation resources allocated for each task. Local UE have a CPU frequency from 0.2 to 1 GHZ. The number of channels in the macro cell is 20. The settings of the main simulation parameters in this paper are shown in Table 2 [12], [31].

### B. IMPACT OF ENVIRONMENT PARAMETERS ON ENERGY CONSUMPTION

We first consider the impact of environment parameters on energy consumption. We compared the energy consumption

**TABLE 2. Parameter settings.**

| Simulation Parameter | Value |
|---|---|
| System total bandwidth | 10 MHz |
| Total number of channels $K$ | 20 |
| Number of small cell $N$ | 4, 6, 8, 10, 12 |
| Number of MUE $U_M$ | 15 |
| Number of SUE in a small cell $U_n$ | 2, 6, 10 |
| Transmission power of MUE $p_M$ | 27 dBm |
| Transmission power of SUE $p_n$ | 23 dBm |
| CPU frequency of UE $f_{u_c}^l$ | [0.1, 1] GHz |
| CPU frequency of MEC $f^C$ | 4 GHz |
| Size of the computation task $s_{u_c}^l$ | [2, 10] MB |
| CPU cycles required by the task $\omega_{u_c}^l$ | [0.5, 2.5] GHZ |
| User tolerates maximum delay $T_{u_c}^{max}$ | [1, 4] s |

of three kinds of task offloading and channel resource allocation schemes with different number of small cells and different amount of UE respectively. The data size and the computation amount of task follows a normal distribution with mean value of 5 MB and 1 GHZ respectively.

As shown in Fig. 2, we can conclude that if each small cell has the same number of people, the energy consumed is positively correlated with the number of small cell. And if the number of small cells remains the same, the energy consumed is positively correlated with the number of people in the small cell. In addition, compared with ROA, AL and AERA, the proposed CRADE has lower energy consumption. This is because that in task offloading and channel allocation, the computational amount of task content, data size and the interference of the same channel are considered comprehensively, so as to minimize the task energy cost and thus obtain an efficient offloading scheme.

In Fig. 2(a), when the number of people in the small cell is 2, the proposed CRADE can reduce energy consumption by 72% to 81% compared with the other three schemes. As shown in Fig. 2(b), when the number of people in the small cell is 6, the proposed CRADE can reduce energy consumption by 33% to 57%. As shown in Fig. 2(b), when

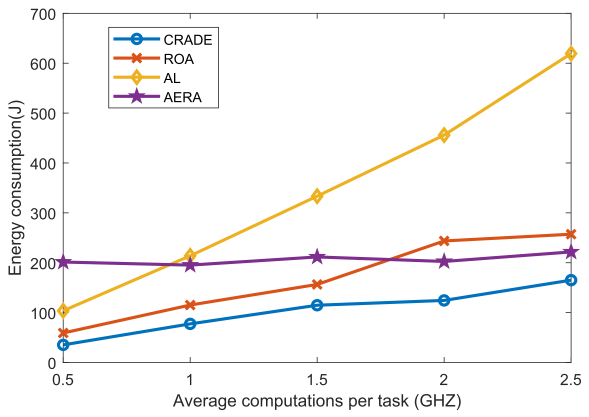


**FIGURE 3.** **Impact of computation amount on energy consumption.**

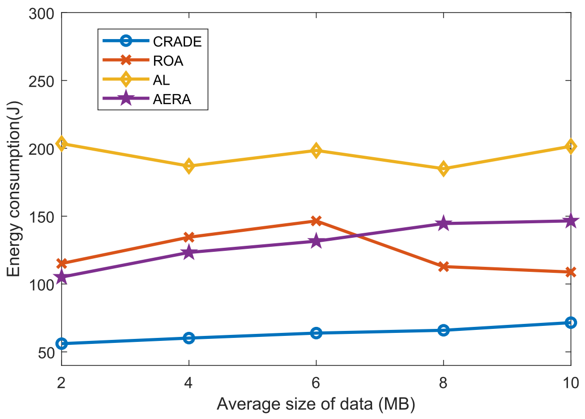


**FIGURE 4.** **Impact of data size energy consumption.**

the number of people in the small cell is 6, the proposed CRADE can reduce energy consumption by 32% to 56%. When the number of SUEs in a small cell is 2, CRADE is most effective in reducing energy consumption, which is because there are enough idle channels in the case of a small number of users. All UEs can offload tasks to the MEC server without interference from other UEs on the same channel. Thus, UE has a very high uplink rate, which makes the transmission delay and energy consumption lower. When the number of users in a small area is 10, the effect of CRADE to reduce energy consumption is also relatively obvious. Because when the number of users increases, CRADE algorithm can take into account the interference caused by channel reuse. UE can choose whether to offload the task to the MEC server or execute it locally. The uplink transmission channel with the highest uplink rate is selected to reduce the energy consumption.

## C. IMPACT OF TASK INDICATORS ON ENERGY CONSUMPTION

In terms of task indicators, we analyze the impact of task computation and data size on energy consumption.

### 1) IMPACT OF COMPUTATION AMOUNT ON ENERGY CONSUMPTION

For task indicators, we first consider the impact of computation amount on energy consumption of system. The number of small cells follows a normal distribution with average value of 8. The number of SUE follows a poisson distribution with average value of 6. The data size of computation task is a normal distribution with average value of 5 MB. The computation amount of computation task is a normal distribution with average value of 0.5 to 2.5 GHZ.

As shown in Fig. 3, we can conclude that the larger the computation of a task, the more energy it will consume, except for the AERA scheme. Because the amount of computation of task has no impact of the transmission rate. Furthermore, our proposed CRADE exhibits lower energy consumption than ROA, AL and AERA scheme.

In Fig. 3, when the mean of task computation is from 0.5 to 2.5 GHZ and the mean of data size is 5 M, compared with the other three schemes, it is shown that our proposed CRADE can reduce 37% to 70% of the energy consumption. Especially when the mean of task computation is 2.5 GHZ, the energy consumption reduction of CRADE scheme is the most obvious compared with AL.

### 2) IMPACT OF DATA SIZE ON ENERGY CONSUMPTION

We second consider the impact of data size on energy consumption of system. The number of small cells follows a normal distribution with average value of 8. The number of SUE follows a poisson distribution with average value of 6. The computation amount of computation task is a normal distribution with average value of 1 GHZ. The data size of computation task is a normal distribution with average value of 2 to 10 MB.

As shown in Fig. 4, we can conclude that the larger the data size of the task, the more energy consumption, except for the ROA and AL scheme. Because the energy consumption of local execution is not related to the size of the data, but to the amount of computation. The reason why the system energy consumption calculated by ROA scheme does not increase with the increase of data size is that the task offloading in ROA scheme is random. Random offloading causes uncertainty in the number of tasks to be executed at the edge. Even if the data size of each task increases, the total data size of the task during the uplink does not necessarily increase. Therefore, the uplink transmission delay time and energy consumption do not increase with the increase of the data size of the task. What's more, our proposed CRADE exhibits lower energy consumption than ROA, AL and AERA scheme.

In Fig. 4, when the mean of data size is from 2 to 10 MB and the mean of task computation is 1 GHZ, compared with the other three schemes, it is shown that our proposed CRADE can reduce 49% to 61% of the energy consumption. According to Fig. 4, we can infer that when the data size of the task is large enough, the transmission delay is too high and the transmission power is too high, CRADE will make the task

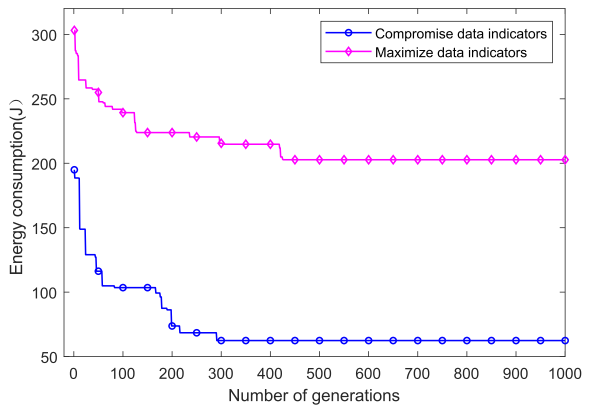


**FIGURE 5. Convergence of different data indicators.**

execute locally. The energy consumption of AL will be the upper limit of the power generated by the CRADE scheme.

### *D. CONVERGENCE ANALYSIS OF CRADE*

Finally, we evaluate and analyze the convergence of CRADE algorithm. We set the data indicators as two cases, one is the compromise data indicators and the other is the maximize data indicators. The corresponding are the general case and the worst case respectively.

- Compromise data indicators: The number of small cells follows a normal distribution with average value of 8. The number of SUE follows a poisson distribution with average value of 6. The computation amount of computation task is a normal distribution with average value of 1 GHZ. The data size of computation task is a normal distribution with average value of 5 MB.
- Maximize data indicators: The number of small cells is 12. The number of SUE is 10. The computation amount of computation task is a normal distribution with average value of 2.5 GHZ. The data size of computation task is a normal distribution with average value of 10 MB.

As shown in Fig. 5, we simulate the convergence of CRADE algorithm in the compromise data indicators and the maximized data indicators. We can get two observations from Fig. 5. First, when the data indicators is a compromise value, CRADE algorithm converges to the optimum when the number of iterations is about 300. Secondly, when the data indicators is the maximum, the number of convergence iterations is 450. Therefore, we can conclude that CRADE algorithm has a great effect in minimizing system energy consumption, and can still converge to the optimal value even in the worst case considered.

## VI. CONCLUSION

In this paper, we study the problem of task offloading and channel resource allocation based on MEC in UDN. Specifically, we express task offloading as a NP-hard integer nonlinear programming problem. To solve this problem, firstly, we eliminate one of the decision variables through variable substitution without changing the nature of the problem, and then propose an efficient task offloading and channel resource allocation scheme based on differential evolution algorithm. Simulation results show that compared with ROA, AL and AERA schemes, this scheme can significantly reduce the system energy consumption and has good convergence. In the future work, we will consider the more complex resource allocation problem under the multi-edge server in the 5G UDN.

## REFERENCES

[1] Y. Teng, M. Liu, F. R. Yu, V. C. M. Leung, M. Song, and Y. Zhang, "Resource allocation for ultra-dense networks: A survey, some research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2134–2168, 3rd Quart., 2018.
[2] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-dense networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2522–2545, 4th Quart., 2016.
[3] T. Qiu, H. Wang, K. Li, H. Ning, A. K. Sangaiah, and B. Chen, "SIGMM: A novel machine learning algorithm for spammer identification in industrial mobile cloud computing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2349–2359, Apr. 2019.
[4] X. Chen, Y. Zhang, and Y. Chen, "Cost efficient request scheduling and resource provisioning in multi-clouds for Internet of Things," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2019.2948432.
[5] Z. Zhao, G. Min, W. Gao, Y. Wu, H. Duan, and Q. Ni, "Deploying edge computing nodes for large-scale IoT: A diversity aware approach," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3606–3614, Oct. 2018.
[6] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. Shen, "Energy efficient dynamic offloading in mobile edge computing for Internet of Things," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2019.2898657.
[7] Y. Xia, M. Zhou, X. Luo, Q. Zhu, J. Li, and Y. Huang, "Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 162–170, Jan. 2015.
[8] C. Pan, M. Elkashlan, J. Wang, J. Yuan, and L. Hanzo, "User-centric C-RAN architecture for ultra-dense 5G networks: Challenges and methodologies," *IEEE Commun. Mag.*, vol. 56, no. 6, pp. 14–20, Jun. 2018.
[9] Y. Chen, J. Huang, C. Lin, and J. Hu, "A partial selection methodology for efficient QoS-aware service composition," *IEEE Trans. Services Comput.*, vol. 8, no. 3, pp. 384–397, May 2015.
[10] J. Huang, Y. Lan, and M. Xu, "A simulation-based approach of QoS-aware service selection in mobile edge computing," *Wireless Commun. Mobile Comput.*, vol. 2018, 2018, Art. no. 5485461.
[11] Y. Zhang, K. Wang, Q. He, F. Chen, S. Deng, Z. Zheng, and Y. Yang, "Covering-based Web service quality prediction via neighborhood-aware matrix factorization," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2019.2891517.
[12] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.
[13] Y. Chen, N. Zhang, Y. Zhang, and X. Chen, "Dynamic computation offloading in edge computing for Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4242–4251, Jun. 2019.
[14] J. Huang, C. Zhang, and J. Zhang, "A multi-queue approach of energy efficient task scheduling for sensor hubs," *Chin. J. Electron.*, to be published.
[15] Y. Zhou, Z. M. Fadlullah, B. Mao, and N. Kato, "A deep-learning-based radio resource assignment technique for 5G ultra dense networks," *IEEE Netw.*, vol. 32, no. 6, pp. 28–34, Nov./Dec. 2018.
[16] Z. Liu, X. Chen, Y. Chen, and Z. Li, "Deep reinforcement learning based dynamic resource allocation in 5G ultra-dense networks," in *Proc. IEEE Int. Conf. Smart Internet Things (SmartIoT)*, Tianjin, China, Aug. 2019, pp. 168–174.
[17] L. Liu, Y. Zhou, L. Tian, B. Sun, and J. Shi, "Interference aware CoMP for macrocell-based heterogeneous ultra dense cellular networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.
[18] N. Sapountzis, T. Spyropoulos, N. Nikaein, and U. Salim, "Joint optimization of user association and dynamic TDD for ultra-dense networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Honolulu, HI, USA, Apr. 2018, pp. 2681–2689.

[19] T. Qiu, B. Li, W. Qu, E. Ahmed, and X. Wang, "TOSG: A topology optimization scheme with global small world for industrial heterogeneous Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3174–3184, Jun. 2019.

[20] Y. Zhang, G. Cui, S. Deng, F. Chen, Y. Wang, and Q. He, "Efficient query of quality correlation for service composition," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2018.2830773.

[21] Y. Wang, H. Liu, W. Zheng, Y. Xia, Y. Li, P. Chen, K. Guo, and H. Xie, "Multi-objective workflow scheduling with deep-q-network-based multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 39974–39982, 2019.

[22] T. Qiu, J. Liu, W. Si, and D. O. Wu, "Robustness optimization scheme with multi-population co-evolution for scale-free wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1028–1042, Jun. 2019.

[23] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[24] J. Chen, J. Jia, Y. Liu, X. Wang, and A. H. Aghvami, "Optimal resource block assignment and power allocation for D2D-enabled NOMA communication," *IEEE Access*, vol. 7, pp. 90023–90035, Jul. 2019.

[25] B. Cao, X. Kang, J. Zhao, P. Yang, Z. Lv, and X. Liu, "Differential evolution-based 3-D directional wireless sensor network deployment optimization," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3594–3605, Oct. 2018.

[26] P. Zhu, J. Zhang, and Y. Ji, "Resource allocation in energy efficient hybrid FSO/mmW fronthaul: A differential evolution approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.

[27] Y. Hao, M. Chen, L. Hu, M. S. Hossain, and A. Ghoneim, "Energy efficient task caching and offloading for mobile edge computing," *IEEE Access*, vol. 6, pp. 11365–11373, 2018.

[28] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[29] L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji, "Mobile edge computing empowered energy efficient task offloading in 5G," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6398–6409, Jul. 2018.

[30] H. Sun, F. Zhou, and R. Q. Hu, "Joint offloading and computation energy efficiency maximization in a mobile edge computing system," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3052–3056, Mar. 2019.

[31] S. Xu, R. Li, and Q. Yang, "Improved genetic algorithm based intelligent resource allocation in 5G Ultra Dense networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barcelona, Spain, Apr. 2018, pp. 1–6.

[32] L. Liu, Y. Zhou, V. Garcia, L. Tian, and J. L. Shi, "Load aware joint CoMP clustering and inter-cell resource scheduling in heterogeneous ultra dense cellular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2741–2755, Mar. 2018.

[33] H. Zhang, M. Min, L. Xiao, S. Liu, P. Cheng, and M. Peng, "Reinforcement learning-based interference control for ultra-dense small cells," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018, pp. 1–6.

[34] Y. Mao, J. Zhang, Z. Chen, 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.

[35] H. Guo, J. Liu, and J. Zhang, "Computation offloading for multi-access mobile edge computing in ultra-dense networks," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 14–19, Aug. 2018.

[36] J. Zhang, X. Hu, Z. Ning, E. C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

[37] L. Liu, Z. Chang, and X. Guo, "Socially aware dynamic computation offloading scheme for fog computing system with energy harvesting devices," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1869–1879, Jun. 2018.

[38] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[39] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. Shen, "TOFFEE: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2019.2923692.

**XIN CHEN** received the Ph.D. degree in computer science from the Beijing Institute of Technology, Beijing, China. He is currently a Professor with the School of Computer Science, Beijing Information Science and Technology University. His current research interest includes performance evaluation of wireless networks. He is also a Senior Member of the China Computer Federation (CCF), and a member of the CCF Technical Committee of Theoretical Computer Science and the CCF Technical Committee of Petri Nets. He received the Postdoctoral Fellowship in Computer Architecture from Tsinghua University, in 2006.

**ZHIYONG LIU** received the M.S. degree from the School of Computer Science, Beijing Information Science and Technology University, Beijing, China. His research interest includes ultra-dense networks and edge computing.

**YING CHEN** received the B.Eng. degree from the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China, in 2012, and the Ph.D. degree with the Computer Science and Technology Department, Tsinghua University, Beijing, in 2017. She was the Google Ph.D. Fellowship recipient, in 2016. She is currently an Associate Professor with the Computer School, Beijing Information Science and Technology University, Beijing. Her current research interests include edge computing, cloud computing, services computing, performance evaluation, and optimization.

**ZHUO LI** received the Ph.D. degree from Nanjing University, in 2012. He is currently an Associate Professor and the Associate Dean of the Graduate School, Beijing Information Science and Technology University. His research interests include wireless networks, distributed computing, and network security.