Wireless Networks
https://doi.org/10.1007/s11276-020-02409-w

Multi-objective resource allocation in mobile edge computing using PAES for Internet of Things

Qi Liu[1,2,3] · Ruichao Mo[1] · Xiaolong Xu[4] · Xu Ma[5]

Abstract
In recent years, mobile edge computing (MEC), as a powerful computing paradigm, provides sufficient computing resources for Internet of Things (IoT). Generally, the deployment of MEC servers closer to mobile users has effectively reduced access delays and the cost of using cloud services. However, the multi-objective resource allocation for IoT applications to meet service requirements (i.e., the shortest completion time of IoT applications, the load balance and lower energy consumption of MEC servers, etc.) still faces severe challenges. To address this challenge, a multi-objective resource allocation method, named MRAM, is proposed in this paper for IoT. Technically, the pareto archived evolution strategy is leveraged to optimize the time cost of IoT applications, load balance and energy consumption of MEC servers. Furthermore, the multiple criteria decision making and the technique for order preference by similarity to ideal solution are utilized to obtain the optimal multi-objective resource allocation strategy. Ultimately, the comprehensive analysis of MRAM is introduced in detail.

Keywords MEC · IoT · Resource allocation · PAES

1 Introduction

With the rapid development of mobile communications and the Internet of Things (IoT), a large number of IoT devices (e.g., smartphones, laptops, sensors, etc.) are connected to the Internet through wireless networks, which effectively guarantees the network connection reliability of IoT devices [1–3]. Besides, the platform provided by the mobile communication enables emerging IoT applications (e.g., intelligent manufacturing, intelligent transportation, etc.) to collect data generated by IoT devices in a timely manner through wireless networks, and analyze the data to provide the users with high-quality services [4, 5].

Currently, most IoT devices are limited in size, resulting in limited computing power and battery life [6]. Therefore, the computing tasks generated on the IoT application need to be sent from the distributed IoT device to the distant cloud [7, 8]. By leveraging the powerful computing and storage capabilities deployed on the cloud, the data generated by IoT applications is effectively processed to ensure that IoT applications provide effective services to end users [9]. However, as the number of IoT devices connected to the network continues to increase, the data

✉ Xu Ma
xma@xidian.edu.cn

Qi Liu
qi.liu@nuist.edu.cn

Ruichao Mo
ruichaomo@gmail.com

Xiaolong Xu
cnxlxu@foxmail.com

1 School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, People's Republic of China

2 Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing 210044, People's Republic of China

3 Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing, People's Republic of China

4 Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, Nanjing University of Science and Technology, Nanjing 210094, People's Republic of China

5 School of Software, Qufu Normal University, Qufu 273165, People's Republic of China

Published online: 02 July 2020

generated by the devices will also increase dramatically, which puts great pressure on the cloud [10]. Moreover, due to the cloud is deployed in the center of the core network away from the user that the physical distance between the cloud and the IoT device is too long, resulting in additional transmission delay and energy consumption for the execution of IoT applications [11].

Mobile edge computing (MEC) is an emerging computing paradigm that provides better services by spreading infrastructure-based cloud resources (such as computing, storage, bandwidth, etc.) to the edge of the network. Compared with traditional cloud, the introduction of MEC shortens the physical distance between IoT applications and MEC, and effectively reduces the data transmission delay [12]. In addition, the participation of 5G also enables MEC to provide users with faster and more reliable services.

However, due to the computing and storage resources of each server are limited, and the resources in the MEC are heterogeneous and coupled, it is difficult for traditional resource allocation methods to adapt to network structure of MEC [13]. Therefore, when allocating limited heterogeneous MEC resources (i.e., computing resources and network resources etc.) for IoT applications, resource allocation to achieve other goals while minimizing the completion time requirements of IoT applications still faces severe challenges [14].

On the one hand, as an important evaluation standard for MEC, the load balance of the MEC server needs to be considered. When the load of the MEC server is not balanced, it will affect the performance of the MEC server and even cause server failure. On the other hand, the MEC server consumes a lot of energy while running, and it also consumes a certain amount of energy in order to reduce the temperature of the server. Therefore, when implementing resource allocation for IoT applications, it is necessary to optimize the load balance and energy consumption of a large number of deployed MEC servers to reduce the cost of edge device deployment and operation [15, 16]. In general, realizing multi-objective resource allocation while ensuring the shortest completion time of IoT applications, minimum load balance and minimum energy consumption still face great challenges. Therefore, a multi-objective resource allocation method, named MRAM, is proposed in this paper to achieve joint optimization of completion time for IoT applications, load balance, and energy consumption for MEC servers.

Specifically, the main contributions of this paper are the following:

- The time cost model for IoT applications, the load balance model and the energy consumption model for MEC servers are designed.
- The pareto archived evolution strategy (PAES) [17] is leveraged to find the solution set of the multi-objective resource allocation strategies with the shortest completion time, minimum load balance for MEC server and minimum energy consumption in the MEC.
- The technique for order preference by similarity to ideal solution (TOPSIS) and the multiple criteria decision making (MCDM) are utilized to obtain the optimal resource allocation strategy for IoT applications.
- A large number of experiments are conducted to verify the effectiveness of MRAM.

The rest of this paper is organized as follows. In Sect. 2, the related work is listed. Then, system model of multi-objective resource allocation is defined in Sect. 3. Futhermore, a multi-objective resource allocation method is proposed in Sect. 4. Experimental evaluation is presented in Sect. 5. Finally, conclusion and future works are drawn in Sect. 6.

2 Related work

Nowadays, as numerous IoT devices are connected to the Internet, people's life becomes more intelligent. For example, the maturity of technologies of Internet of Vehicles (IoV) makes people's journey more convenient and faster [14, 18]. However, as the number of IoT devices connected to the Internet continues to increase, it also puts a lot of bandwidth pressure on cloud that handle devices from the Internet of things [19]. As a computing paradigm where computing resources are placed near the end-user, the introduction of MEC enables a large amount of data generated by the device of IoT to be sent to the edge node for processing, thus reducing the load of the cloud while effectively processing the data [20–22].

However, due to the deployed the MEC servers have a wide variety of computing structures, it poses a significant challenge for edge devices to implement dynamic resource allocation [23, 24]. Therefore, according to the computing demands of IoT devices, dynamic resource allocation at the MEC servers will make full use of computing resources at the MEC servers [25, 26].

In She et al. [27] establish a cross-layer architecture to optimize user affinity, packet offload rates, and bandwidth scheduling for mission-critical Internet of things (Mc-IoT) services with short packets. Moreover, an optimization algorithm is proposed to shorten the communication delay when the throughput of eMBB service is much higher than that of Mc-IoT service. In Tang et al. [28] describe the cross-layer resource scheduling problem as a mixed-integer nonlinear programming (MINLP) and propose a low Shaping-and-Pruning (SP) algorithm to obtain the sparse

solution active RRH set, so that the cross-layer resource scheduling strategy reduce energy consumption effectively. In Lyu et al. [29] propose a new integration architecture for the cloud, MEC, and Internet of things, as well as a lightweight request and allow framework to address network scalability issues. In Zhao et al. [30] proposed a collaborative computing unloading and resource allocation optimization (CCORAO) scheme based on MEC and cloud computing, designed a distributed computing unloading and resource allocation algorithm, and realized the joint optimization of computing unloading decision and computing resource allocation in the Internet of vehicles. Moreover, based on the analysis of the completion time and energy consumption of the computational task, Liu et al. [31] propose a power-constrained delay minimization problem and a one-dimensional search algorithm to find the optimal task scheduling strategy. In You et al. [32] proposed a suboptimal resource allocation algorithm for a cloud with limited capacity, and obtained a corresponding resource allocation strategy by defining an average offload priority function.

Meanwhile, in the process of computing tasks, as the main evaluation criteria for moving edge computing, tasks completion time, energy consumption of MEC servers and load balance of the MEC servers also need to be considered.

In Zhang et al. [33] propose a motion aware hierarchical MEC framework for green and low latency IoT. Moreover, a game theory-based computational offloading algorithm is proposed to optimize the utility of service providers while reducing the energy consumption and task execution time of smart devices. In Chunlin et al. [34] propose an energy-aware method for moving cloud resource allocation across layers. By taking advantage of the system context and mobile user preferences, the energy-aware mobile cloud resource allocation method is used across layers to optimize cloud resource consumption and system performance. In [35], to solve the problem of computation offloading and content cache strategy, Wang et al. considers computation offloading, resource allocation and content cache as an optimization problem. Furthermore, an alternate direction algorithm based on distributed convex optimization is proposed to solve the optimization problem. In Tran et al. [36] breaks down computational resource allocation on the MEC server into resource allocation (RA) problems with fixed task offload decisions. In addition, a new heuristic algorithm using convex and quasi-convex optimization techniques is proposed to solve the problem of suboptimal solutions in polynomial time. In Wang et al. [37] developed an innovative framework to improve MEC performance by co-optimizing the energy transfer beamforming of the AP and the number of bits unloaded by the user, as well as the time allocation among users. Based on this framework, the author proposes an optimal resource allocation scheme, which minimizes the total energy consumption of AP on the premise of satisfying the delay constraint of individual computing.

To the best of our knowledge, there are few studies on the resource allocation for IoT devices in mobile edge computing. In addition, due to the importance of task completion time, energy consumption of the MEC servers and load balance of the MEC servers, it is necessary to take them all into consideration. Therefore, a multi-objective resource allocation method is proposed in this paper.

3 System model and problem formulation

3.1 Multi-objective resource allocation architecture and resource model

In MEC, the computing demands of the end-user are addressed by means of a MEC server deployed close to the end-user. However, due to the weak computing power of the MEC servers, when a computing task with a large amount of computation is encountered, a reasonable resource allocation is required to ensure the execution of the computing task. The key notations of system model are given in Table 1.

As shown in Fig. 1, a variety of IoT applications through the wireless network is transmitted to the nearest the MEC servers to perform, when the computing resources of the MEC server are not able to meet the task, it is necessary through the adjacent MEC server for the resource allocation to ensure effective execution of the tasks on the the MEC servers. Technically, assumed that the set of various applications is $APT = \{apt_1, apt_2, \ldots, apt_M\}$, where $apt_i (1 \le i \le M)$ is represented as an IoT application. Besides, assumed that each application generates only one computing task in the same time period, the set of computing tasks is denoted as $ATN = \{atn_1, atn_2, \ldots, atn_M\}$, where atn_i is the computing task of is the computing task of apt_i. Furthermore, assumed that the set of the MEC servers is $MEN = \{men_1, men_2, \ldots, men_N\}$. In addition, the number of virtual machines (VM) contained in $men_n (1 \le n \le N)$ is defined as mnv_n.

3.2 Time cost model

The computing tasks generated in the IoT applications need to be transferred over the wireless network to the nearest MEC server for processing. Therefore, the completion time of $atn_m (1 \le m \le M)$ is divided into task transfer time TM_m,

Table 1 Notations

Notation	Description
M	The amount of IoT applications
N	The amount of MEC servers
APT	The IoT application set $APT = \{apt_1, apt_2, \ldots, apt_M\}$
ATN	The computing task set $ATN = \{atn_1, atn_2, \ldots, atn_M\}$
MEN	The MEC server set $MEN = \{men_1, men_2, \ldots, men_N\}$
TE_m	The time cost of the task atn_m
EC	The energy consumption of all MEC servers
MSU	The load balance variance of MEC server
K	The number of the resource allocation strategy
RAS	The cross-layer resource allocation strategy set $RAS = \{ras_1, ras_2, \ldots, ras_K\}$

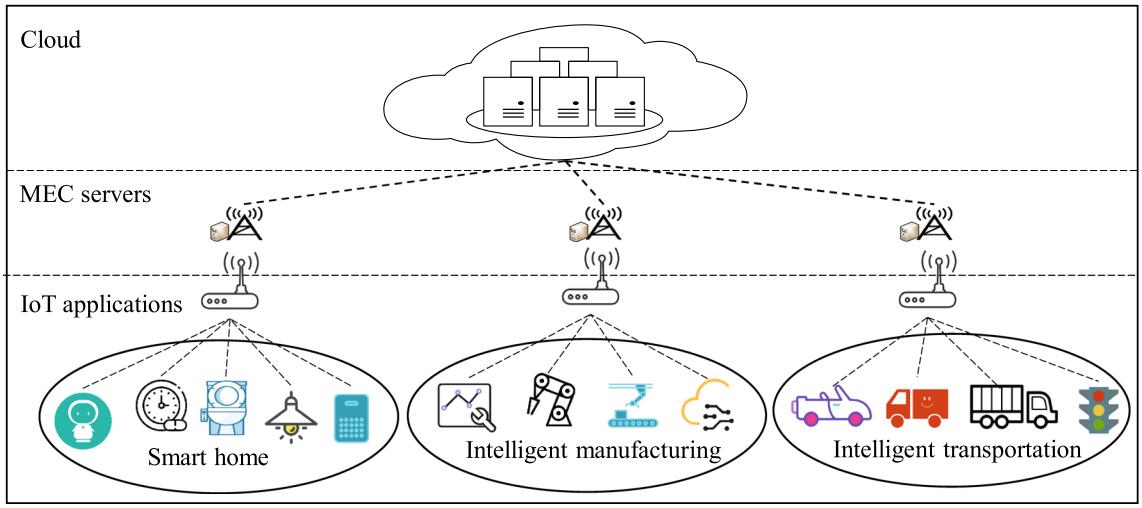


Fig. 1 A resource allocation architecture for IoT in MEC

task completion time TW_m and result return time TB_m, which is calculated by

$$TE_m = TM_m + TW_m + TB_m. \tag{1}$$

Assume that the wireless network bandwidth between the IoT device and the MEC server is WK_p, the network bandwidth among MEC servers is WK_q and the network bandwidth between cloud and the MEC servers is WK_r. Moreover, the variable j is introduced to represent three kinds of resource allocation of computing task at the MEC server: (1) $j = 0$ represents the computing resources that the MEC server satisfy the task; (2) $j = 1$ means that the computing resources of the MEC server cannot meet requirements of the task, and resources are allocated from other servers to meet demends of the task; (3) $j = 2$ means that the computing resources of the MEC server cannot meet requirements of the task, and resources are allocated from the cloud to meet demends of the task.

Thus, assume that the data size of task atn_m is G_m, the required transmission time of the task is measured by

$$TM_m = \begin{cases} \dfrac{G_m}{WK_p}, & j = 0, \\ \dfrac{G_m}{WK_p} + \dfrac{G_m}{WK_q}, & j = 1, \\ \dfrac{G_m}{WK_p} + \dfrac{G_m}{WK_q} + \dfrac{G_m}{WK_r}, & j = 2. \end{cases} \tag{2}$$

Based on the above analysis, assume that the data size of the returned result is RG_m, and the return time is calculated by

$$TB_m = \begin{cases} \frac{RG_m}{WK_p}, & j = 0, \\ \frac{RG_m}{WK_p} + \frac{RG_m}{WK_q}, & j = 1, \\ \frac{RG_m}{WK_p} + \frac{RG_m}{WK_q} + \frac{RG_m}{WK_r}, & j = 2. \end{cases} \tag{3}$$

Furthermore, due to the computing performance of VM in MEC servers is consistent, assuming that the data processing capacity of men_n is s_n, the execution time required by task atn_m is measured by

$$TW_m = \sum_{n=1}^{N} \frac{GR_m \cdot c(men_n)}{s_n}. \tag{4}$$

where $c(men_n)$ is denoted as a binary variable to judge the men_n is occupied, which is measured by

$$c(men_n) = \begin{cases} 1, & \text{if men}_n \text{ is occupied}, \\ 0, & \text{otherwise}. \end{cases} \tag{5}$$

3.3 Load balance model

As an important evaluation criterion, the load balance of each MEC server needs to be considered in resource allocation. In this paper, the virtual machine usage contained in each MEC server is used to evaluate the load balance of the MEC server. Technically, supposed that the VM size needed to execute $atn_m (1 \le m \le M)$ is mev_m. Thus, the average utilization msu_n of $men_n (1 \le n \le N)$ is calculated by

$$msu_n = \frac{1}{mnv_m} \sum_{i=0}^{mnv_m - 1} \varphi(atn_m) \cdot mev_m. \tag{6}$$

Denoted $\varphi(atn_m)$ as a binary variable aiming to estimate whether the atn_m is deployed on men_n which is measured by

$$\varphi(atn_m) = \begin{cases} 1, & \text{if } \ atn_m \text{ runs on } men_n, \\ 0, & \text{otherwise}. \end{cases} \tag{7}$$

Furthermore, ZYS is represented as the total amount of occupied VM, which is measured by

$$ZYS = \frac{1}{N} \sum_{n=0}^{N} msu_n. \tag{8}$$

Finally, the load balance variance MSU of the MEC servers is deduced by

$$MSU = \frac{1}{N} \sum_{n=1}^{N} (msu_n - ZYS)^2. \tag{9}$$

3.4 Energy consumption model

In this paper, the energy consumption of the MEC servers is mainly generated in three aspects: the basic operating energy consumption of the MEC servers, the energy consumption of occupied VM and the energy consumption of unoccupied VM. Among them, the energy consumption of occupied VM depends on the complexity of computing task, that is, the time required to complete the task. The maximum task completion time of the MEC server is recorded as the working time, which is measured by

$$wnt_n = \max_{m=1}^{M} (\varphi(atn_m) \cdot TW_m). \tag{10}$$

The basic operating energy consumption of the MEC server is calculated by

$$ECB = \sum_{n=1}^{N} wnt_n \cdot rx, \tag{11}$$

where rx is utilized to represent the power rate of MEC server.

Furthermore, the energy consumption of occupied VM is measured by

$$ECF = \sum_{n=1}^{N} \sum_{m=1}^{M} \varphi(atn_m) \cdot wnt_n \cdot rf, \tag{12}$$

where rf represents the power rate of occupied VM.

The energy consumption of unoccupied VM is acquired by

$$ECK = \sum_{n=1}^{N} (ZYS - \sum_{m=1}^{M} \varphi(atn_m)) \cdot wnt_n \cdot rn, \tag{13}$$

where rn is used to represent the power rate of unoccupied VM.

Finally, the energy consumption of all MEC servers is calculated by

$$EC = ECB + ECF + ECK. \tag{14}$$

3.5 Problem formulation

The multi-objective resource allocation for IoT applications in MEC is focused on optimizing the time cost, load balance and the energy consumption of all MEC servers. Thus, the problem to be solved in this paper is defined by

$$\min TE_m, EC, MSU. \tag{15}$$

$$\text{s.t.} \ \forall \varphi(atn_m) = 1, \ men_i \in MEN \tag{16}$$

$$\sum_{i=1}^{M} \varphi(atn_i) \cdot mev_i \le mnv_n, \tag{17}$$

where (16) and (17) indicate that when resources are allocated for IoT applications, the VMs obtained by the application cannot exceed the sum of the number of idle VMs in the current MEC servers.

4 Multi-objective resources allocation method using PAES

In this paper, the resource allocation is defined as a multi-objective optimization problem, which ensures the shortest task completion time for IoT devices, the lowest energy consumption of the MEC server, and the minimum load balance of the MEC server. To solve the resource allocation problem, PAES, as a multi-objective optimization algorithm, is applied to execute resource allocation for IoT. Compared with the traditional multi-objective optimization algorithm, the Pareto frontier is obtained faster by PAES. Furthermore, the algorithm complexity of PAES is lower.

4.1 Chromosome representation

In multi-objective genetic algorithms, the target of a problem is usually defined as genes, which are made up of chromosomes to find solutions that satisfy multiple goals. In this paper, while solving the problem of resource allocation, the goals of shortest task completion time, minimum load balance and minimum energy consumption of MEC servers needed to be satisfied. Therefore, the three targets are defined as genes, constituting a chromosome. In addition, gene manipulation works on chromosomes to quickly find resource allocation strategies. On this basis, the resource allocation strategy set is represented as $RAS = \{ras_1, ras_2, \ldots, ras_K\}$.

4.2 Fitness function and constraints

In MRAM, fitness function is used to identify chromosomes of the population. Therefore, the chromosomes are initially screened by fitness function in the population. In addition, various constraints are used to limit the population of chromosome evolution. In this paper, the task completion time, load balance and energy consumption of the MEC server as fitness function and represented by (1), (9) and (14). In addition, in the process of resource allocation, the constraint that the number of resources required by the task should not exceed the resources owned on the MEC server is expressed by (17).

4.3 Selection

In the selection process, PAES generates the evolutionary population and the elite population through the initialization operator, while the appropriate elite strength parameters are also obtained. Then evolutionary population and elite population compete to produce the next generation. Moreover, in the process of evolution, the scale and degree of elite individuals' participation will be limited by elite intensity, the sampling process is controlled by sampling operators, and vary operators generate the next generation of individuals through mutation.

By using the elite retention strategy, the dominant solution with good quality produced by PAES is preserved, effectively avoiding the loss of excellent individuals, and maintaining the diversity of the population through crossover and mutation. In addition, in the search process of PAES, the external solution set approximation is taken as the current non-dominant frontier, providing auxiliary information for the selection process of parent and child individuals, thus guiding the process to produce more excellent individuals.

4.4 Iteration

As an important part of PAES, adaptive grid algorithm is used to choose between the current solution and the mutation solution during iteration. In addition, external files are updated according to the situation of the candidate solution. If the candidate solution of the resource allocation is dominated by any solution in the file, the solution is deleted to ensure that the solution in the file is not dominated by the candidate solution. In PAES, the number of solutions in the archive will be in a dynamic state during the iteration process, so the size of the grid will be adjusted automatically during the iteration, and the location will be repositioned according to the distribution of solutions, so as to ensure that each solution is in a certain grid.

4.5 Optimal multi-objective resource allocation strategy acquired

In fact, there will be multiple resource alloation strategies generated during the MRAM iteration. Therefore, the multiple criterion decision method (MCDM) and TOPSIS will be utilized to obtain the optimal resource allocation strategy after the iteration. In TOPSIS, the order is made by discriminating the distance between the optimal solution and the worst solution of the evaluation individual. If the evaluation individual is closest to the optimal solution and farthest from the worst solution, the evaluation individual

is regarded as the best one; otherwise, the evaluation individual is regarded as the worst one.

As the resource allocation strategy set obtained by the algorithm, there are K solutions in RAS, each solution contains the values of task completion time, load balance and energy consumption, and the corresponding values of the strategy are expressed as $SH = \{sh_1, sh_2, \ldots, sh_K\}$, $ZS = \{zs_1, zs_2, \ldots, zs_K\}$ and $NH = \{nh_1, nh_2, \ldots, nh_K\}$. The optimal resource allocation strategy is obtained through the following operations in TOPSIS.

Firstly, the task completion time for all solutions is processed as follows to complete the normalization

$$SN_i^S = \frac{sh_i}{\sqrt{\sum_{i=1}^{K} sh_i}}. \tag{18}$$

Besides, the load balance of the MEC servers and energy consumption of all solutions are regularized as above to obtain ZSN_i^Y and NSN_i^H.

Supposed that the weights of task completion time, load balance and energy consumption are expressed as ρ_t, ρ_u and ρ_e. Then the weighted normalization value of task completion time is calculated by

$$WNV_i^S = \rho_t \cdot SN_i^S. \tag{19}$$

Furthermore, the weighted regularization value of the load balance and energy consumption are acquired as WNV_i^Y and WNV_i^N respectively.

Then, the distance from each target to the ideal solution is calculated, and the range scale is calculated by the Euclidean Distance. The distance from the target to the ideal solution of ISD_k^T is calculated as

$$\begin{aligned} ISD_k = &(WNV_k^S - \max_{k=1}^{K}(WNV_k^T))^{\frac{1}{2}} \\ &+ (WNV_k^Y - \max_{k=1}^{K}(\mathrm{WNV}_k^Y))^{\frac{1}{2}} \\ &+ (WNV_k^N - \max_{k=1}^{K}(WNV_k^N))^{\frac{1}{2}}. \end{aligned} \tag{20}$$

Moreover, the distances of the anti-ideal solution is measured by

$$\begin{aligned} FSD_k = &(WNV_k^S - \min_{k=1}^{K}(WNV_k^T))^{\frac{1}{2}} \\ &+ (WNV_k^Y - \min_{k=1}^{K}(\mathrm{WNV}_k^Y))^{\frac{1}{2}} \\ &+ (WNV_k^N - \min_{k=1}^{K}(WNV_k^N))^{\frac{1}{2}}. \end{aligned} \tag{21}$$

Finally, the closeness degree s of the ideal solution is calculated as

$$DIS_k = \frac{FSD_k}{FSD_k + ISD_k}. \tag{22}$$

According to the degree of closeness of the ideal solution, the ranking result with the largest degree of closeness is regarded as the optimal resource allocation strategy.

4.6 Method overview of MRAM

In this paper, MRAM is divided into four steps, the generation of candidate solutions for multi-objective resource allocation, the selection of candidate solutions, the utilization of the non-dominated solutions and the acquisition of the optimal solution. Initially, MRAM produces a solution by simply randomly mutation and the goal value of the initial solution is calculated and placed into the file that has been set. Then, a new solution is generated by mutating the single parent solution, and the target value of the new solution is also calculated. There will be two scenarios: (1) If the new solution is dominated by the parent solution, the new solution will be generated again by mutating the parent solution; (2) If it is not dominated, the new solution is compared with other solutions in the file, the file is updated through iteration, and a solution is randomly selected in the file as the new parent solution. Furthermore, MRAM iteration process ends at the end of the set number of iterations. Finally, the optimal multi-objective resource allocation strategy is obtained by MCDM and TOPSIS. The specific flow of the algorithm is shown in Algorithm 1.

Algorithm 1 Multi-objective resource allocation method using PAES.

Ensure: The resource allocation strategies set RAS for the IoT applications.
1: Randomly generate the set of parent solutions.
2: Calculates the parent solution and puts the target value into the file.
3: **while** Iteration not finished. **do**
4: The parent solution performs the mutation to produce the offspring.
5: Calculate offspring solution.
6: **if** The descendant solution is not dominated by the parent solution. **then**
7: Compare the dominance of subsolutions with other solutions in the file.
8: Adaptive grid algorithm is used to update files.
9: Select a solution in the file as the parent solution.
10: **else** Continue.
11: **end if**
12: **end while**
13: Optimal resource allocation strategy obtain by MCDM and TOPSIS.
14: **return** ras_i

5 Comparison and analysis of experimental results

5.1 Simulation setup

In this paper, several simulation experiments are conducted to verify the performance of MRAM. Specifically, the simulation experiment is performed on a laptop with Intel Core i5-9500, 3.00 GHZ CPU, 8 GB RAM and Window 10. In addition, the specific setup in this simulation is given in Table 2.

5.2 Compared algorithms

To illustrate the performance of the MRAM, three resource allocation algorithms are selected for comparison, i.e., Benchmark, FFD-D, and BFD-D. And the basic principles of these three methods are listed.

- *Benchmark* The basic idea of Benchmark does not consider additional factors in the process of resource allocation and attempts to randomly select one of all MEC servers that meet the requirements of the Internet of Things as a resource allocation strategy, but when the capacity of the node is not enough to meet the capacity required for the IoT application, the method will be randomly generated again until the strategy found.
- *First fit decreasing-based resource allocation method (FFD-D)* When FFD-D is utilized to allocate resources for IoT applications, it will first sort in ascending order according to the current number of unoccupied VM of all MEC servers. Then, FFD-D will select MEC servers from the ordered MEC servers set according to the number of VM required by the current IoT application. In this process, the MEC server first found by FFD-D will be assigned to the IoT application to ensure the execution of the application.
- *Best fit decreasing-based resource allocation method (BFD-D)* In the process of resource allocation by BFD-D, BFD-D first sorts all running MEC servers in descending order according to the number of unoccupied virtual machines. Then, BFD-D will select the MEC server that has the smallest number of remaining VM but satisfies the current IoT application, thereby ensuring the execution of the IoT application.

Table 2 Parameter settings

Parameter	Value
The amount of IoT applications	50, 100, 150, 200, 250
The amount of VM on each MEC server	8
The amount of VM required for IoT applications	[1, 7]
The power rate of the MEC server	350 W
The power rate of employed VM instances	75 W
The power rate of unemployed VM instances	45 W

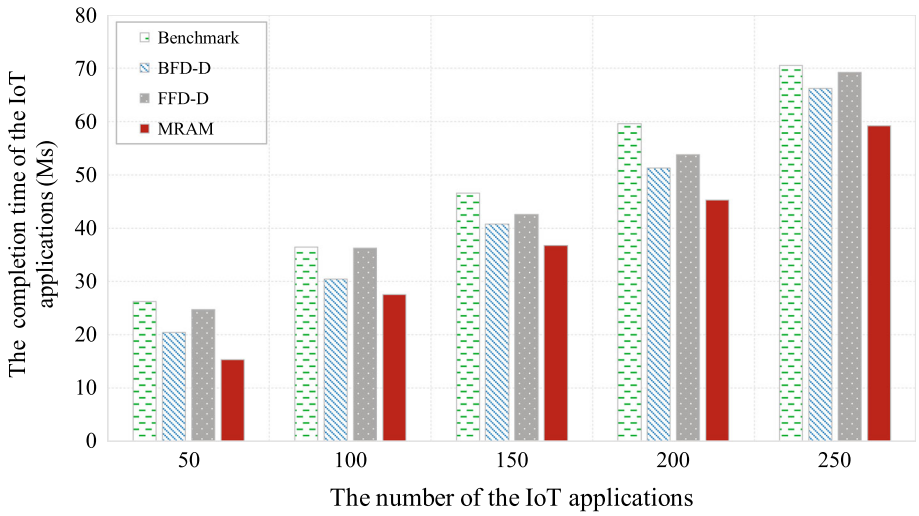


Fig. 2 Comparison of the IoT application completion time by Benchmark, BFD-D, FFD-D, and MRAM

5.3 Performance evaluation

5.3.1 Comparison of completion time for IoT application

In this paper, the completion time of the IoT application as a key evaluation standard for evaluating resource allocation methods requires detailed analysis. The results of the completion time of Benchmark, BFD-D, FFD-D and MRAM in different numbers of IoT applications are shown in Fig. 2. Compared with Benchmark, BFD-D, and FFD-D, MRAM has fully iterated in the process of finding the optimal resource allocation strategy, so that the resource allocation strategy obtained by MRAM makes the completion time of the application significantly lower than Benchmark, FFD-D and FFD-D effectively ensure the execution efficiency of IoT applications.

5.3.2 Comparison of load balance for VM

Load balance as an important indicator of the MEC server also needs to be evaluated by calculating the number of VM occupied by each server. The load balance variances of Benchmark, FFD-D, BFD-D and MRAM are shown in Fig. 3. Compared with Benchmark, FFD-D and BFD-D, MRAM allocates global resources to make the number of VM occupied by each MEC server more balanced, thus ensuring a lower variance of load balance during the execution of different numbers of IoT applications.

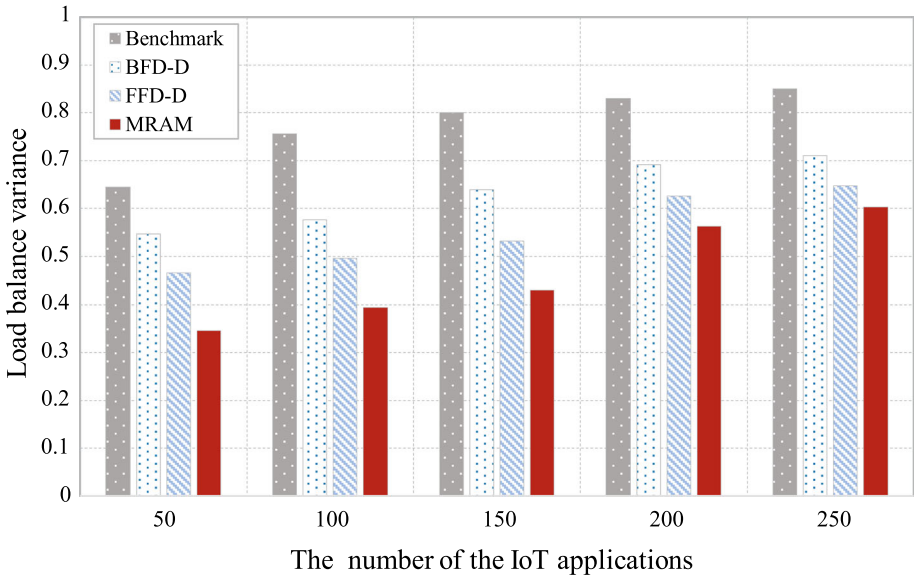


Fig. 3 Comparison of the load balance by Benchmark, BFD-D, FFD-D, and MRAM

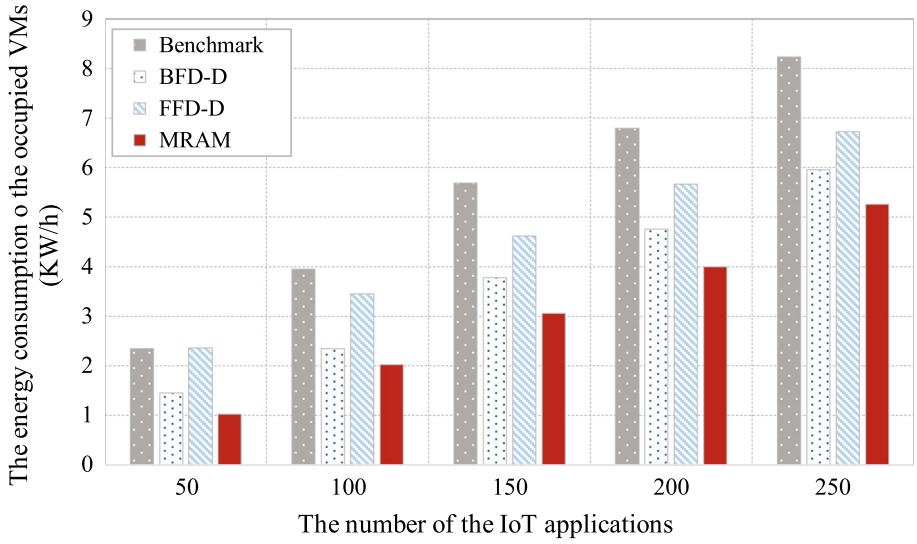


Fig. 4 Comparison of the energy consumption of the occupied VM by Benchmark, BFD-D, FFD-D, and MRAM

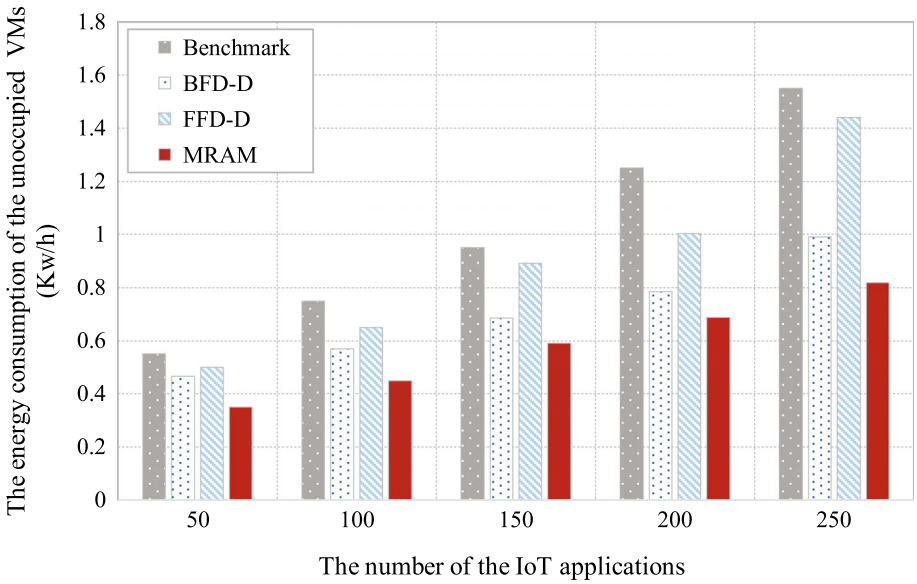


Fig. 5 Comparison of the energy consumption of the unoccupied VM by Benchmark, BFD-D, FFD-D, and MRAM

5.3.3 Comparison of the energy consumption of the occupied VM

In the experiment, the energy consumption of the MEC server as an important evaluation criterion is divided into the energy consumption of occupied VM, the energy consumption of unoccupied VM, and the total energy consumption. Initially, the energy consumption of occupied VM is analyzed and compared. In the case of different numbers of IoT applications, the consumption of VM occupied by Benchmark, BFD-D, FFD-D and MRAM is shown in Fig. 4. Overall, as the number of IoT applications continues to increase, The energy consumption of VM is also rising. Compared with Benchmark, BFD-D, FFD-D, and MARM, the resource allocation strategy found is lower in the case of different numbers of IoT applications.

5.3.4 Comparison of the energy consumption of the unoccupied VM

Furthermore, the energy consumption of unoccupied VM is analyzed and compared. As shown in Fig. 5, Benchmark, FFD-D, BFD-D and MRAM do not occupy the virtual machine energy consumption under different numbers of IoT applications. The unoccupied virtual machine energy consumption increases with the increasing number of IoT applications. Compared with Benchmark, FFD-D and BFD-D, MRAM has a lower energy consumption of unoccupied VM in the case of different numbers of IoT applications.

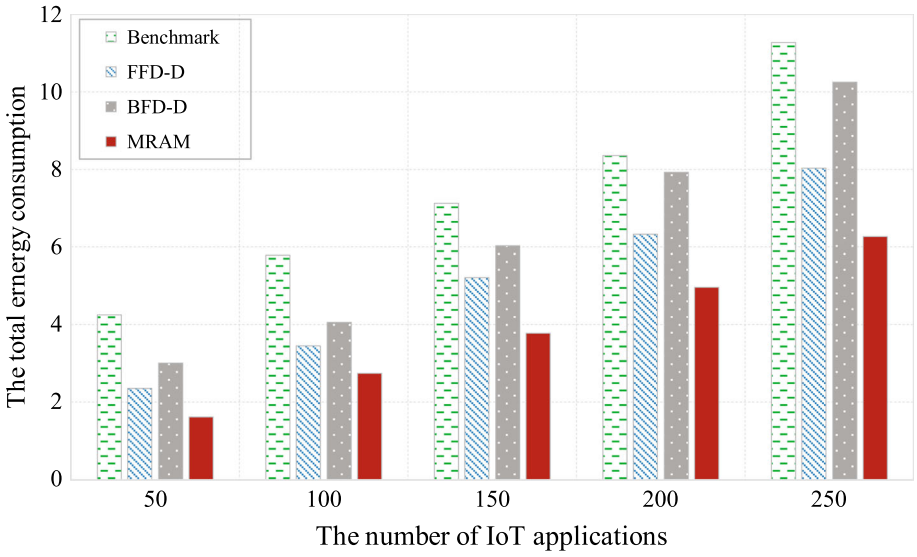


Fig. 6 Comparison of the total energy consumption of the VM by Benchmark, BFD-D, FFD-D, and MRAM

5.3.5 Comparison of the total energy consumption

Finally, the total energy consumption of the MEC server is analyzed and compared. As shown in Fig. 6, with the increase in the number of IoT applications, the resource allocation by Benchmark, BFD-D, FFD-D and MRAM makes the total energy consumption of the MEC server show an upward trend. However, compared with Benchmark, BFD-D, FFD-D, the resource allocation strategy obtained by MRAM makes the energy consumption generated by the execution of different numbers of IoT applications the lowest.

6 Conclusion and future work

In this paper, to achieve effective resource allocation for IoT applications in MEC, a multi-objective resource allocation method (MRAM) based on PAES was proposed in this paper. Technically, PAES was leveraged to identify resource allocation strategies that meet the minimum completion time, the minium load balance, and the minimum energy consumption. Then, the TOPSIS and MCDM were utilized to acquire the optimal resource allocation strategy. Finally, a large number of experiments were conducted to verify the effectiveness of MRAM.

In the future work, we will try to apply MRAM to real scenarios to conduct the resource allocation for IoT application in MEC. Moreover, the privacy issues of the IoT application need to be considered.

Acknowledgements This research is supported by the Basic Researc Programs (Natural Science Foundation) of Jiangsu Province (BK20191398), the National Natural Science Foundation of China under Grant No. 61702277. Besides, this work is also supported by the Fundamental Research Funds for the Central Universities, No. 30918014108 and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund.

Compliance with ethical standards

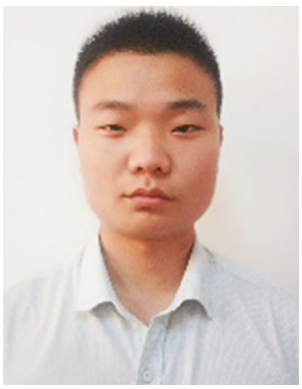
Conflict of interest The authors declare that they have no conflict of interest.

References

1. Mach, P., & Becvar, Z. (2017). Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys and Tutorials*, *19*(3), 1628–1656.
2. Xiaolong, X., Zhang, X., Khan, M., Dou, W., Xue, S., & Shui, Y. (2020). A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems. *Future Generation Computer Systems*, *105*, 789–799.
3. Gong, W., Qi, L., & Yanwei, X. (2018). Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment. *Wireless Communications and Mobile Computing*, *2018*, 1–8.
4. Hsieh, H.-C., Chen, J.-L., & Benslimane, A. (2018). 5g virtualized multi-access edge computing platform for iot applications. *Journal of Network and Computer Applications*, *115*, 94–102.
5. Qi, L., Zhang, X., Li, S., Wan, S., Wen, Y., & Gong, W. (2020). Spatial-temporal data-driven service recommendation with privacy-preservation. *Information Sciences*, *515*, 91–102.
6. Pan, J., & McElhannon, J. (2017). Future edge cloud and edge computing for internet of things applications. *IEEE Internet of Things Journal*, *5*(1), 439–449.
7. Chen, X., Shi, Q., Yang, L., & Jie, X. (2018). Thriftyedge: Resource-efficient edge computing for intelligent iot applications. *IEEE Network*, *32*(1), 61–65.
8. Xiaolong, X., Xihua, L., Zhanyang, X., Chuanjian, W., Shaohua, W., & Xiaoxian, Y. (2019). Joint optimization of resource utilization and load balance with privacy preservation for edge services in 5g networks. *Mobile Networks and Applications*, *25*, 713–724.
9. Yanwei, X., Qi, L., Dou, W., & Jiguo, Y. (2017). Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment. *Complexity*, *2017*, 1–9.
10. Lianyong, Q, Yi, C., Yuan, Y., Shucun, F., Xuyun, Z., & Xiaolong X. (2019). A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems. *World Wide Web*, pp. 1–23.
11. Xu, X., He, C., Xu, Z., Qi, L., Wan, S., & Bhuiyan, M. Z. A. (2019). Joint optimization of offloading utility and privacy for edge computing enabled IoT. *IEEE Internet of Things Journal*, *7*, 2622–2629.
12. Xu, X., Zhang, X., Liu, X., Jiang, J., Qi, L., & Bhuiyan, M. Z. A. (2020). Adaptive computation offloading with edge for 5G-envisioned internet of connected vehicles. *IEEE Transactions on Intelligent Transportation Systems*. https://doi.org/10.1109/TITS.2020.2982186.
13. Ning, Z., Huang, J., Wang, X., Rodrigues, J. J., & Guo, L. (2019). Mobile edge computing-enabled internet of vehicles: Toward energy-efficient scheduling. *IEEE Network*, *33*(5), 198–205.
14. Xu, X., Cao, H., Geng, Q., Liu, X., Dai, F., & Wang, C. (2020). *Dynamic resource provisioning for workflow scheduling under uncertainty in edge computing environment*. Concurrency and Computation: Practice and Experience. https://doi.org/10.1002/cpe.5674.
15. Zhang, T., Xu, Y., Loo, J., Yang, D., & Xiao, L. (2019). Joint computation and communication design for UAV-assisted mobile edge computing in IoT. *IEEE Transactions on Industrial Informatics*, *16*, 5505–5516.
16. Xu, X., Zhang, X., Gao, H., Xue, Y., Qi, L., & Dou, W. (2019). Become: Blockchain-enabled computation offloading for IoT in mobile edge computing. *IEEE Transactions on Industrial Informatics*, *16*, 4187–4195.
17. Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, *8*(2), 149–172.
18. Wang, F., Jiang, D., Qi, S., Qiao, C., & Shi, L. (2020). A dynamic resource scheduling scheme in edge computing satellite networks. *Mobile Networks and Applications*, pp. 1–12.
19. Xu, X., Fu, S., Yuan, Y., Luo, Y., Qi, L., Lin, W., et al. (2019). Multiobjective computation offloading for workflow management in cloudlet-based mobile cloud using NSGA-II. *Computational Intelligence*, *35*(3), 476–495.
20. Zhang, J., Zhou, Z., Li, S., Gan, L., Zhang, X., Qi, L., et al. (2018). Hybrid computation offloading for smart home automation in mobile cloud computing. *Personal and Ubiquitous Computing*, *22*(1), 121–134.
21. Xu, X., Shen, B., Yin, X., Khosravi, M. R., Wu, H., Qi, L., et al. (2020). Edge server quantification and placement for offloading

social media services in industrial cognitive IoV. *IEEE Transactions on Industrial Informatics*. https://doi.org/10.1109/TII.2020.2987994.
22. Kuang, Z., Li, L., Gao, J., Zhao, L., & Liu, A. (2019). Partial offloading scheduling and power allocation for mobile edge computing systems. *IEEE Internet of Things Journal*, *6*(4), 6774–6785.
23. Li, X., Wan, J., Dai, H.-N., Imran, M., Xia, M., & Celesti, A. (2019). A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing. *IEEE Transactions on Industrial Informatics*, *15*(7), 4225–4234.
24. Wang, K., Yu, X., Lin, W., Deng, Z., & Liu, X. (2019). Computing aware scheduling in mobile edge computing system. *Wireless Networks*. https://doi.org/10.1007/s11276-018-1892-z.
25. Wang, Q., Guo, S., Liu, J., & Yang, Y. (2019). Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing. *Sustainable Computing: Informatics and Systems*, *21*, 154–164.
26. Abbas, N., Zhang, Y., Taherkordi, A., & Skeie, T. (2017). Mobile edge computing: A survey. *IEEE Internet of Things Journal*, *5*(1), 450–465.
27. She, C., Duan, Y., Zhao, G., Quek, T. Q. S., Li, Y., & Vucetic, B. (2019). Cross-layer design for mission-critical IoT in mobile edge computing systems. *IEEE Internet of Things Journal*, *6*(6), 9360–9374.
28. Tang, J., Tay, W. P., & Quek, T. Q. S. (2015). Cross-layer resource allocation with elastic service scaling in cloud radio access network. *IEEE Transactions on Wireless Communications*, *14*(9), 5068–5081.
29. Lyu, X., Tian, H., Jiang, L., Vinel, A., Maharjan, S., Gjessing, S., et al. (2018). Selective offloading in mobile edge computing for the green internet of things. *IEEE Network*, *32*(1), 54–60.
30. Zhao, J., Li, Q., Gong, Y., & Zhang, K. (2019). Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks. *IEEE Transactions on Vehicular Technology*, *68*(8), 7944–7956.
31. Liu, J., Mao, Y., Zhang, J., & Letaief, K. B. (2016). Delay-optimal computation task scheduling for mobile-edge computing systems. In *2016 IEEE international symposium on information theory (ISIT)* (pp. 1451–1455). IEEE.
32. You, C., Huang, K., Chae, H., & Kim, B. (2017). Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Transactions on Wireless Communications*, *16*(3), 1397–1411.
33. Zhang, K., Leng, S., He, Y., Maharjan, S., & Zhang, Y. (2018). Mobile edge computing and networking for green and low-latency internet of things. *IEEE Communications Magazine*, *56*(5), 39–45.
34. Chunlin, L., Yanpei, L., & Youlong, L. (2017). Energy-aware cross-layer resource allocation in mobile cloud. *International Journal of Communication Systems*, *30*(12), e3258.
35. Wang, C., Liang, C., Yu, F. R., Chen, Q., & Tang, L. (2017). Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. *IEEE Transactions on Wireless Communications*, *16*(8), 4924–4938.
36. Tran, T. X., & Pompili, D. (2018). Joint task offloading and resource allocation for multi-server mobile-edge computing networks. *IEEE Transactions on Vehicular Technology*, *68*(1), 856–868.
37. Wang, F., Jie, X., Wang, X., & Cui, S. (2017). Joint offloading and computing optimization in wireless powered mobile-edge computing systems. *IEEE Transactions on Wireless Communications*, *17*(3), 1784–1797.

Qi Liu received the B.S. degree in Computer Science and Technology from Zhuzhou Institute of Technology, China in 2003, and M.S. and Ph.D. in Data Telecommunications and Networks from the University of Salford, UK in 2006 and 2010. His research interests include context awareness, data communication in MANET and WSN, and smart grid. His recent research work focuses on intelligent agriculture and meteorological observation systems based on WSN.



Ruichao Mo received the B.S. degree in software engineering in 2019 from Binjiang College, Nanjing University of Information Science and Technology, Nanjing, China, in 2019, from where he is currently working toward the master's degree in computer science and technology. His areas of interest are edge computing, big data, mobile edge computing, and machine learning.

Xiaolong Xu received the Ph.D. degree in computer science and technology from Nanjing University, Nanjing, China, in 2016. He is currently an Assistant Professor with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing. He worked as a Research Scholar with Michigan State University, USA, from April 2017 to May 2018. His research interests include mobile computing, edge computing, Internet of Things, cloud computing, and big data. Dr. Xu has authored or coauthored more than 80 peer review papers in international journals and conferences including IEEE Transactions on Industrial Informatics, IEEE Transactions on Cloud Computing, IEEE Transactions on Big Data, IEEE Internet of Tings Journal, IEEE Transactions on Computational Social Systems, IEEE Transactions on Emerging Topics in Computational Intelligence, World Wide Web Journal, Software: Practice and Experience, IEEE International Conference on Web Services, International Conference on Service-Oriented Computing, etc. He is the recipient of the Best Paper Award of IEEE International Conference on Advanced Cloud and Big Data 2016, and Best Student Paper Award of European Alliance for Innovation Cloudcomp 2019.

Xu Ma received the bachelor's degree of computer science from Ludong University in 2008 and the master and PhD degrees of information security and cryptography from Sun Yat-sen University in 2010 and 2013, respectively. He is currently an associate professor in the school of software, Qufu Normal University, China. He is also a post-doctor of the department of cyberspace security, Xidian University. His research focuses on applied cryptography, outsourcing computation and privacy-preserving machine learning.