A Constrained Multi-objective Computation Offloading Algorithm in the Mobile Cloud Computing Environment

Li Liu, Yuanyuan Du* and Qi Fan
School of Automation and Electrical Engineering University of Science and Technology Beijing, China
s20170578@xs.ustb.edu.cn
*Corresponding author: Yuanyuan Du

Received January 8, 2019; accepted February 22, 2019; published September 30, 2019

Abstract

Mobile cloud computing (MCC) can offload heavy computation from mobile devices onto nearby cloudlets or remote cloud to improve the performance as well as to save energy for these devices. Therefore, it is essential to consider how to achieve efficient computation offloading with constraints for multiple users. However, there are few works that aim at multi-objective problem for multiple users. Most existing works concentrate on only single objective optimization or aim to obtain a tradeoff solution for multiple objectives by simply setting weight values. In this paper, a multi-objective optimization model is built to minimize the average energy consumption, time and cost while satisfying the constraint of bandwidth. Furthermore, an improved multi-objective optimization algorithm called D-NSGA-II-ELS is presented to get Pareto solutions with better convergence and diversity. Compared to other existing works, the simulation results show that the proposed algorithm can achieve better performance in terms of energy consumption, time and cost while satisfying the constraint of the bandwidth.

Keywords: Mobile cloud computing, Computation offloading, Multi-objective optimization, NSGA-II

1. Introduction

Mobile cloud computing (MCC) is the integration of mobile computing and cloud computing, which aims to offload computing-intensive tasks from the mobile devices to the resource-rich cloud (such as Amazon Elastic Compute Cloud, Google App Engine [1]). It helps to save energy and improve the processing capability for mobile devices.

In the MCC environment, the complex tasks are able to be migrated from mobile devices to the remote cloud or cloudlet via wireless access (e.g. 3G/4G, Wi-Fi) [2]. However, the process of offloading through 3G/4G usually cause long delay and slow data transfers [3]. On the other hand, the cloudlet brings low communication latency connecting to the mobile devices with Wi-Fi networks, but the system scalability is poor when the number of mobile users increases fast [4]. It is quite significant to consider how to make offloading decisions while satisfying various requirement of multiple users.

To address the aforementioned challenges, we focus on how to make the computation offloading decisions for multi-user with the bandwidth limitation in the MCC environment. For a three-tier MCC architecture (the mobile users, nearby cloudlet and public cloud), the multi-objective optimization model is proposed to minimize the average energy consumption, time and cost of all users. Then, an improved multi-objective algorithm named is presented based on the NSGA-II algorithm. The main improvements of the proposed algorithm are as follows: Firstly, an adaptive Differential Evolution (DE) mutation operator and Normal Distribution Crossover (NDX) operator are applied to expand the searching space. Secondly, the Diversity Maintaining Strategy (DMS) is proposed to improve its diversity. Thirdly, the Elitist Learning Strategy (ELS) is designed to perform on the non-dominated solutions in each generation for better convergence.

To summarize, the major contributions of this paper are as follows:

1) The computation offloading decision making for multiple users in the MCC environment is modeled as a constrained multi-objective optimization problem, aiming to minimize the average energy consumption, time and cost;

2) During the process of mutation and crossover in the improved NSGA-II, an adaptive DE mutation operator and NDX operator are applied to expand the searching space;

3) In order to improve the diversity of Pareto solutions, the DMS is proposed according to the different state of the algorithm;

4) The ELS is designed to perform on the non-dominated solutions in each generation for better convergence;

5) Simulations show that the proposed algorithm behaves better than other compared algorithms in terms of energy, time and cost while meeting the constraint of bandwidth.

The remainder of the paper is organized as follows. Section 2 briefly reviews the related works of model and algorithm of computation offloading, followed by a description of the system and computation offloading model in Section 3. In section 4, an improved NSGA-II algorithm (D-NSGA-II-ELS) is presented in detail. Section 5 demonstrates the results and performance of the proposed algorithm. The conclusion is given in section 6.

2. Related Work

Usually, there are one or more optimization objectives for the offloading decisions under the MCC environment. However, most of the state-of-the-art researches focused on single-objective issues with different constraints. Rashidi et al. [5] proposed a queue model for their mobile cloud architecture and designed a hybrid heuristic algorithm to minimize mean completion time of all tasks. In [6], a dynamic algorithm based on Lyapunov optimization was proposed to save energy when meeting the time constraint. Taking the limitations of delay and energy into account, a new K-M-LARAC algorithm based on K-LARAC and M-LARAC algorithms was proposed by Haghighi et al. to minimize cost [7]. There are some researchers who have contributed to the study of multi-objective problems as well. In [8], Goudarzi et al. proposed a weighted cost model based on execution time and energy consumption to achieve a tradeoff between them for offloading. To solve this problem, a branch-and-bound algorithm with an optimal branching rule, a fast search strategy and an appropriate bounding function, was firstly designed for the small-scale applications, and then an improved Particle Swarm Optimization (PSO) algorithm was developed to achieve the best-possible offloading solution for the large-scale applications. In [9], aiming at minimizing energy consumption, delay and cost, a multi-objective optimization problem with a queuing model was formulated. This optimization problem was addressed by setting different weight values for various objectives, and the Interior Point Method was applied to achieve the optimal solution. In order to minimize both execution delay and energy consumption for computation offloading, two cases were taken into consideration [10]. For the special case of infinite energy capacity, polynomial-time optimal solution based on weighted bipartite matching problem was proposed. For the general case, an efficient heuristic algorithm was designed to obtain the tradeoff solution between execution delay and energy consumption.

In MCC environment, there are many previous works that have researched the single-user computation offloading problem. In [11], considering a heterogeneous environment with cloudlet and public clouds, an efficient multisite computation offloading algorithm is presented based on the Genetic Algorithm (GA) to minimize the cost of application execution. To minimize the weighted sum of energy, computation and delay, Chen et al. [12] formulated the offloading decision making as a non-convex constrained quadratic program and developed an efficient heuristic algorithm based on semi definite relaxation and a novel randomization mapping method. Meanwhile, a few works have focused on multiple mobile users. In [13], Cao et al. considered a multi-user multi-radio channel scenery and formulated this offloading decision making problem as a noncooperative game. Then, a distributed computation offloading algorithm, which introduced the machine learning technology, was proposed to solve this problem. In [14], a reverse auction-based offloading method was designed to make decisions for multiple users. This method aimed at minimizing average energy consumption while meeting the constraints of time and communication quality of the wireless channel. Kuang et al. [15] proposed a Dynamic Programming After Filtering algorithm for multi-user, which takes into account the bandwidth limitation of the Wi-Fi AP but only focuses on single-objective.

From the aforementioned related studies, most works concentrate on single-user multi-objective problem (e.g. [8]-[10]) or the multi-user single-objective optimization with constraints (e.g. [13]-[15]). There are few works that aim at multi-objective problem for multiple users. For example, in [16], an efficient three-step algorithm was proposed to minimize the cost of energy, computation, and delay for all users. Then, the multi-objective optimization problem was finally simplified as single-objective optimization problem with different weights for various objectives. However, it is difficult to assign the accurate weight values for all the users in terms of their preferences. Thus, in this paper, we study a multi-objective optimization problem with many constraints in a multi-user MCC environment. Moreover, an improved multi-objective optimization algorithm based on NSGA-II is proposed to solve this problem.

3. System Model and Problem Formulation

In this paper, we consider a three-tiered MCC environment consisting of mobile users, nearby cloudlet with limited resources, and remote rich-resource public cloud [4]. Under this environment, tasks can be executed either on the mobile devices or offloaded to the cloudlet through a Wi-Fi access point or the remote cloud through 3G/4G communication. The architecture is depicted as **Fig. 1**. Generally, not all tasks can be offloaded to the cloudlet for execution because there is limited bandwidth available at an access point. We assume that the bandwidth between mobile device and cloudlet is B_{CL}, and the bandwidth between mobile device and cloud is B_C. For most situations, $B_{CL} \geq B_C$ [6].

In this section, inspired by Chen et al. [17], the computation offloading model will be presented, and the computation offloading problem is formulated as a constrained multi-objective optimization problem.

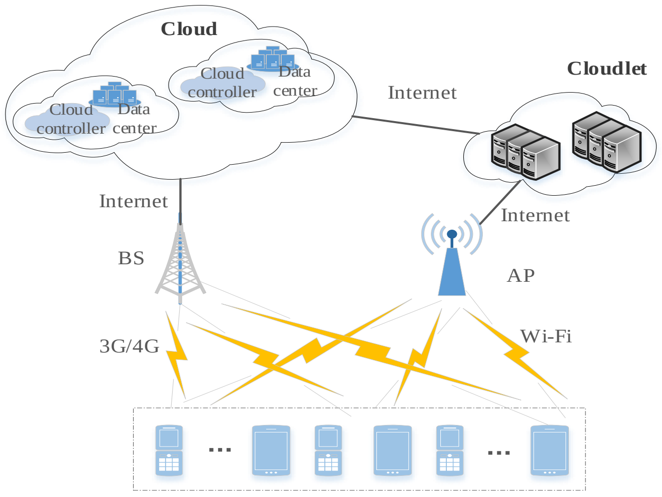


Fig. 1. The architecture of a mobile cloud offloading system

3.1 Offloading Decision

For each mobile user, we assume that there is only one task that needs to be executed locally or be offloaded. Meanwhile, an offloaded task may be processed on the cloudlet or be further forwarded to the remote cloud. Therefore, the offloading decisions could be expressed as follows:

$$x_i^L + x_i^{CL} + x_i^C = 1 \tag{1}$$

Where $x_i^L, x_i^{CL}, x_i^C \in \{0,1\}$ indicates whether the task of user *i* is executed locally, on the cloudlet, or on the cloud, respectively. And it is worth noting that only one location can be selected to perform the task for each user.

3.2 System Model

In this section, we mainly consider three performance metrics: execution energy consumption, execution time, execution cost. Details will be described as following. **Table 1** shows the definitions of terms and concepts in this article.

Table 1. Notations

Notation	**Description**
D_i	the number of required CPU cycles to execute task *i*
B_i	the size of computation input data of task *i*
f_i^L	the computation capability of the mobile device *i*
P_i^E	the local execution power of the mobile device *i*
$C_i^{L\_compute}$	the local computing cost of per unit time of the mobile device *i*
f^{CL} / f^C	the cloudlet/cloud computation capability
W_i^{CL} / W_i^C	the uplink bandwidth between the mobile device *i* and cloudlet/cloud
$P_i^{CL\_trans} / P_i^{C\_trans}$	the transmitting power between the mobile device *i* and cloudlet/cloud
$C^{CL\_trans} / C^{C\_trans}$	the transmitting cost of per unit time between the mobile device and cloudlet/cloud
C^{CL} / C^C	the computing cost of per unit time executed in the cloudlet/cloud

A. Local Processing

We assume the total number of mobile devices is *n*. D_i represents the number of required CPU cycles to execute task *i*, and f_i^L indicates the computation capability of the mobile device *i*. Here we allow that different mobile devices may have different computation capability. When the task *i* is processed locally, the execution time T_i^L, the execution energy consumption E_i^L and the execution cost C_i^L are defined as follows, respectively:

$$T_i^L = \frac{D_i}{f_i^L} \tag{2}$$

$$E_i^L = \frac{D_i}{f_i^L} P_i^E \tag{3}$$

$$C_i^L = \frac{D_i}{f_i^L} C_i^{L\_compute} \tag{4}$$

Where P_i^E, $C_i^{L\_compute}$ respectively mean the execution power and the computing charge of per unit time of device *i* executed locally.

B. Cloudlet Processing

Since there are multiple tasks offloaded to the cloudlet for processing, it's necessary to consider the computing and communication model at the same time [18]. Here, B_i denotes the size of computation input data and f^{CL} represents the cloudlet computation capability. Then, we can calculate the total time T_i^{CL}, total energy E_i^{CL} and total cost C_i^{CL} of mobile device user *i* can be calculated as:

$$T_i^{CL} = \frac{D_i}{f^{CL}} + \frac{B_i}{W_i^{CL}} \tag{5}$$

$$E_i^{CL} = \frac{D_i}{f^{CL}} P_i^I + \frac{B_i}{W_i^{CL}} P_i^{CL\_trans} \tag{6}$$

$$C_i^{CL} = \frac{D_i}{f^{CL}} C^{CL} + \frac{B_i}{W_i^{CL}} C^{CL\_trans} \tag{7}$$

Where W_i^{CL} denotes the uplink bandwidth for computation offloading between the mobile device and cloudlet; $P_i^{CL\_trans}$ denotes the transmitting power; the cost per unit time is represented as C^{CL_trans}; P_i^I means the idle power of device *i* and C^{CL} is the cost per unit time executed in the cloudlet.

C. Cloud Processing

If a task *i* is offloaded to the cloud for execution, its calculation process of time T_i^C, energy E_i^C and cost C_i^C is similar to that in the cloudlet. Therefore, those can be calculated as:

$$T_i^C = \frac{D_i}{f^C} + \frac{B_i}{W_i^C} \tag{8}$$

$$E_i^C = \frac{D_i}{f^C} P_i^I + \frac{B_i}{W_i^C} P_i^{C\_trans} \tag{9}$$

$$C_i^C = \frac{D_i}{f^C} C^C + \frac{B_i}{W_i^C} C^{C\_trans} \tag{10}$$

Where f^C, C^C represents the computation capability and the cost of per unit time in the cloud separately. For the communication process of device i to cloud, W_i^C indicates the bandwidth, $P_i^{C\_trans}$ is the transmitting power and C^{C_trans} denotes the transmitting cost of per unit time.

Notice that, in this paper, the process of returning results from the cloudlet or cloud to the device is neglected, which is similar to existing work [15], [18] - [20]. The reason is that, in general, the computation outcome is much smaller for many applications when compared with the size of input data.

3.3 Problem Formulation

The computation offloading problem among multiple users aims at minimizing the average energy consumption E_{avg}, average time T_{avg} and average cost C_{avg} of all users. They could be calculated as follows:

$$E_{avg} = \sum_{i=1}^{n} \left(E_i^L x_i^L + E_i^{CL} x_i^{CL} + E_i^C x_i^C \right) / n \tag{11}$$

$$T_{avg} = \sum_{i=1}^{n} \left(T_i^L x_i^L + T_i^{CL} x_i^{CL} + T_i^C x_i^C \right) / n \tag{12}$$

$$C_{avg} = \sum_{i=1}^{n} \left(C_i^L x_i^L + C_i^{CL} x_i^{CL} + C_i^C x_i^C \right) / n \tag{13}$$

Moreover, the maximum tolerated time $T_{\max}$, maximum tolerated energy $E_{\max}$ and maximum tolerated cost $C_{\max}$ need to be considered according to the expectations of mobile users and providers. Simultaneously, the available bandwidth at an access point is limited, denoted by $W_{\max}$. Thus, in this paper, the offloading problem under MCC environment is formulated as a constrained multi-objective optimization problem.

$$\text{Minimize: } \left\{ E_{avg}, T_{avg}, C_{avg} \right\} \tag{14}$$

Subject to:

$$E_{avg} \le E_{\max} \tag{15}$$

$$T_{avg} \le T_{\max} \tag{16}$$

$$C_{avg} \le C_{\max} \tag{17}$$

$$\sum_{i=1}^{n} W_i^{CL} x_i^{CL} \le W_{\max} \tag{18}$$

$$x_i^L + x_i^{CL} + x_i^C = 1 \tag{19}$$

4. Computation Offloading Algorithm

The multi-objective optimization model for computation offloading under MCC environment is modeled as a nonlinear multi-objective optimization problem with many constraints and decision variables. The multi-objective evolutionary algorithm (MOEA) is an efficient method to deal with it. Among MOEAs, NSGA-II [21] is a widely adopted algorithm, which has low computational complexity and good convergence. There are three main features in NSGA-II, including the non-dominated sorting, crowding distance calculation and elitist strategy. In an evolutionary cycle of the algorithm, for parent population and offspring population, a rank is assigned to each solution based on the non-domination sorting between the solutions. Then, the whole population will be classified into various non-domination levels. After that, the crowding distance is employed to estimate the density of individuals for every solution in the same level. In the elite strategy, these better individuals, which have higher nondominated sets level and larger crowding distance, will be retained into the next generation.

However, in order to further improve the diversity and convergence of Pareto optimal solutions when solving the complicated and constrained optimization problems, an improved NSGA-II algorithm (D-NSGA-II-ELS) based on traditional NSGA-II is proposed in this paper. The key improved operations are presented as follows.

4.1 Generate the Offspring

Since the offspring are generated by the crossover and mutation from father chromosomes, the two operators play an important role in the algorithm, whose results have an impact on the direction and scope of the search. Thus in this paper, the DE mutation operator and the NDX operator is applied to replace the original ones.

A. Mutation operator

Due to multiple mutation strategies of DE algorithm, the adaptive mutation operator, which comprehensively considers the strong global search capability of DE/rand/1/bin in Eq. 20 and the fast convergence speed of DE/best/1/bin in Eq. 21, is adopted finally [22]. It can be expressed in Eq. 22.

$$\text{DE/rand/1/bin: } v_{i,G} = x_{r1,G} + F \cdot \left(x_{r2,G} - x_{r3,G}\right) \tag{20}$$

$$\text{DE/best/1/bin: } v_{i,G} = x_{best,G} + F \cdot \left(x_{r1,G} - x_{r2,G}\right) \tag{21}$$

$$v_{i,G} = \left(1 - w_G\right) \cdot x_{r1,G} + w_G \cdot x_{best,G} + F \cdot \left(x_{r2,G} - x_{r3,G}\right) \tag{22}$$

Where G denotes the current generation; $w_G = G / G_{\max}$ is the weight factor in generation G; $G_{\max}$ represents the maximum number of generation; w_G is linearly increased from 0 to 1; $x_{r1,G}$ accounts for a significant proportion in the early stage of evolution; $x_{best,G}$ is more considered in later period, which improves the convergence speed of the algorithm.

We let $F_{\max} = 0.9$, $F_{\min} = 0.4$ [23]. An adaptive mutation operator is shown as Eq. 23, which improves the diversity to avoid premature in the early iteration and accelerates convergence to optimal solution in the later stages.

$$F = F_{\max} - (F_{\max} - F_{\min}) \cdot G / G_{\max} \tag{23}$$

B. Crossover operator

In order to enhance the search ability in the space, NDX is applied in the algorithm, which is described as follows [24]:

$$\begin{cases} v_{i,j} = (x_{r1,j} + x_{r2,j})/2 + 1.481 \cdot |N(0,1)| \cdot (x_{r1,j} - x_{r2,j})/2, u \le 0.5 \\ v_{i,j} = (x_{r1,j} + x_{r2,j})/2 - 1.481 \cdot |N(0,1)| \cdot (x_{r1,j} - x_{r2,j})/2, u > 0.5 \end{cases} \tag{24}$$

Where $v_{i,j}$ denotes the *j*-th variable of the offspring chromosome *i*; $N(0,1)$ is a normal distribution random variable and u is a random number uniformly distributed from 0 to 1.

With the expansion of search capabilities, the problem of local optimization and unstable evolutionary processes is avoided. And as a result, the quality of the pareto solutions can be improved.

Algorithm 1 the Improved Diversity Maintenance Strategy

```
Input: the non-dominated set PF with the size of M
Output: the best non-dominated setPFwith the size of N
For i=1:M
    For j=i+1:M
        Calculate the Euclidean distance ED(i, j) between individual i and individual j by Eq. 26;
        ED(i, j)= ED(j, i);
    End
End
While M>N
    SortED=sort(ED, 2, 'ascend'); //sort each row of ED in ascending order.
    i=1;
    While i<M
        EDC=[]; EDC= SortED( : , i+1); //Extract the ith small Euclidean distance of each individual.
        [value, sit]=min(EDC); // Obtainthe minimum and corresponding location.
        Allsit=find(value);
        If length(Allsit)==1
            i=M+1; delete= Allsit;
        End
        i=i+1;
    End
    If length(Allsit)~=1
        delete= Allsit(1); //Randomly select a deletion.
    End
    PF(delete, : )=[];
    ED(delete, : )=[]; ED( : , delete)=[]; //Remove the Euclidean distance corresponding to the deleted individual from ED.
    M=M-1;
End
```

4.2 Diversity Maintaining Strategy

Based on the crowding distance (CD), the population diversity maintaining strategy of the original NSGA-II is that: we assume that the size of the population is N and the number of non-dominated solutions is M. When M>N, the M individuals are arranged in descending order according to their respective CD values. Then, we remove the last M-N solutions with the lower CD value from the population at one time.

The CD of individual *i* is calculated by Eq. 25. Where $Q_j^{\max}$ and $Q_j^{\min}$ are the max and min normalized values of the objective *j* respectively.

$$CD_i = \sqrt{\sum_{j=1}^{M}\left(\frac{Q_j^{i+1} - Q_j^{i-1}}{Q_j^{\max} - Q_j^{\min}}\right)^2} \tag{25}$$

However, some issues may occur by using the above method. For example, in **Fig. 2**, solid black spots A~G represent non-dominated individuals, individual C, D and E are deleted from the non-dominated set for their small CD values, resulting that some parts of the set are too crowded and some parts are too sparse [25].

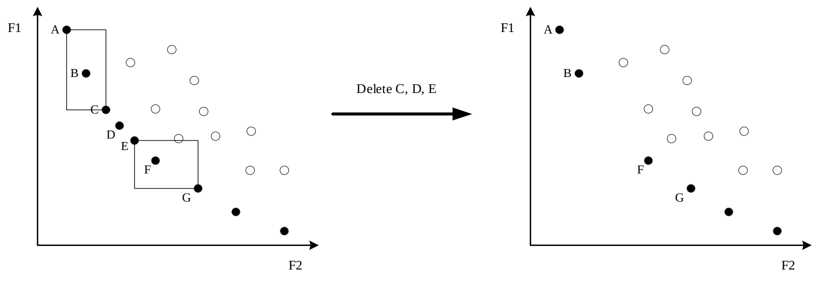


Fig. 2. The individual distribution of the population

Thus, in this paper, a dynamic method of deleting individuals one by one is introduced into the algorithm to improve the uniformity of non-dominated solutions. Furthermore, in this method, the Euclidean distance in Eq. 26 is utilized to evaluate the relationship between individual *i* and other individuals. **Algorithm 1** shows the process in detail.

$$d = \sqrt{\sum_{j=1}^{M}\left(x_j - y_j\right)^2} \tag{26}$$

4.3 Elitist Learning Strategy

In order to avoid falling into local optimum and achieve better converge for Pareto solutions, the Elitist Learning Strategy (ELS) is employed. Since the better individual means closer to the true pareto optimal boundary, it is more beneficial to select individuals in the non-dominated set to perform the ELS than other ones. And in order to reduce the computational burden, the learning strategy is only used in part of the non-dominated solutions in each iteration.

In the non-dominated set, the larger CD values of individuals indicates that other non-dominated solutions distributed around them are sparser, and learning them can effectively improve the uniformity of non-dominated solutions. Therefore, in ELS, the Binary Tournament Selection is adopted to choose solutions from the non-dominated set to perform neighborhood learning, which can increase the chances of individuals with larger CD values being selected. The elitist learning operation is as follows:

$$v_{i,j} = x_{i,j} + \left(x_{i,j}^{ub} - x_{i,j}^{lb}\right) \cdot Gaussian\left(\mu, \sigma\right) \quad (27)$$

$$\sigma = \sigma_{\max} - \left(\sigma_{\max} - \sigma_{\min}\right) \cdot G / G_{\max} \quad (28)$$

Where $x_{i,j}^{ub}$, $x_{i,j}^{lb}$ are the upper and lower bounds of the *j*-th variable for the individual *i* respectively. The elitist learning rate σ is linearly decreasing as the number of iterations increases. The upper bound and lower bound of it are set as $\sigma_{\max} = 1.0$, $\sigma_{\min} = 0.1$ [26]. $Gaussian(\mu, \sigma)$ is a random number of a Gaussian distribution with mean μ and standard deviation σ.

4.4 The Process of D-NSGA-II-ELS

With above approaches, the D-NSGA-II-ELS algorithm can be summarized as follows:

Step 1: Set the relevant parameters and let the number of iterations be $t = 1$;

Step 2: Initialize a parent population pop_0 and an elitist learning population $epop_0$;

Step 3: Repeat the following steps, until the termination condition is satisfied.

Step 3.1: Generate a new offspring population $opop_{G+1}$ through the selection, improved crossover and improved mutation operations in section 4.1 on the parent population of the last iteration pop_G;

Step 3.2: Combine the population pop_G, $opop_{G+1}$ and $epop_G$, calculate the fitness and then calculate the non-dominated rank and the crowding distance of each individual according to the fitness, next sort this population;

Step 3.3: If the number of non-dominated solutions (M) is larger than the capacity of the parent population (N), the improved diversity maintenance strategy in section 4.2 is adopted to choose N individuals to enter into the new parent population pop_{G+1}, otherwise the elite strategy in the traditional NSGA-II is adopted to select the top N individuals with low non-dominated levels and larger CD values;

Step 3.4: Perform the ELS in section 4.3 for non-dominated solutions in pop_{G+1} to obtain the population.

5. Example Analysis

In order to verify the performance of the proposed algorithm D-NSGA-II-ELS, we compare it with original NSGA-II algorithm [21] and another improved algorithm (LS-NSGA-II-DE) [27]. The partial related parameters of the algorithms are set as follows: the size of the population is 50; the max generation is 200; and others are shown in **Table 2**. Each instance is independently run 30 times. All algorithms are written with Matlab and implemented in a PC with the Inter (R) Core (TM) CPU i5-8400 (2.81 GHz and 8G RAM).

Table 2. Algorithm parameters

parameter	P_c	P_m	$\sigma_{\max}$	$\sigma_{\min}$	$F_{\max}$	$F_{\min}$
value	0.8	0.2	1.0	0.1	0.9	0.4

5.1 Experiments on Multi-objective Benchmark Functions

1) Multi-Objective Benchmark Functions

Seven different widely used multi-objective Benchmark functions are adopted to evaluate the proposed algorithm with two other MOEAs, which include four unconstrained test instances (ZDT1-ZDT3, UF2) and three constrained problems (Binh2, Srinivas, CTP1) [28] [29] [30]. Among all of these 7 functions, UF2 and CTP1 have much more complicated search space. The details of these optimization problems are depicted in **Table 3**.

Table 3. Description of Benchmark Functions

Function	**Nature**	**Variable**	**Objectives**	**Pareto shape**
ZDT1	Unconstrained	30	2	Convex
ZDT2	Unconstrained	30	2	Concave
ZDT3	Unconstrained	30	2	Discontinuous
UF2	Unconstrained	10	2	Convex
Binh2	Constrained	2	2	Convex
Srinivas	Constrained	2	2	Convex
CTP1	Constrained	2	2	Linear

2) Performance Metrics

In order to quantify the convergence and diversity of Pareto optimal front obtained by the proposed algorithm (D-NSGA-II-ELS) and other comparison algorithms, the inverted generational distance (IGD) [31] and Spread [32] are used to evaluate the performance of the algorithm.

IGD: This value reflects the convergence of the solutions, and it can be obtained by Eq. 29 for a uniformly distributed along the true Pareto fronts and the obtained solution points.

$$IGD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{29}$$

Where n is the number of the obtained solutions and d_i is the Euclidean distance between the obtained solution i and the closest solution of the true Pareto front. The smaller value of IGD is, the better quality of the obtained solution is.

Spread: This metric measures the diversity of the non-dominated solutions along the obtained approximation Pareto front. The value can be calculated by Eq. 30.

$$Spread = \frac{d_f + d_l + \sum_{i=1}^{n} \left| d_i - \bar{d} \right|}{d_f + d_l + (n-1) \cdot \bar{d}} \tag{30}$$

Where d_i and n are the same as in IGD. d_f, d_l represent the Euclidean distance of the two boundary points in the obtained Pareto front set respectively, and $\bar{d}$ is the average of d_i. When the value is close to zero, the solutions we obtained distributes uniformly.

3) Experimental Results

Table 4. The IGD and Spread obtained by three different algorithms

Algorithm		Unconstrained functions				Constrained functions		
		ZDT1	ZDT2	ZDT3	UF2	Binh2	Srinivas	CTP1
NSGA-II	IGD	1.68E-02	1.45E-02	2.37E-02	1.78E-02	1.21E-01	1.69E-01	7.29E-03
	Spread	3.81E-01	3.79E-01	5.21E-01	4.31E-01	5.27E-01	4.31E-01	4.15E-01
LS-NSGA-II-DE	IGD	1.05E-02	9.07E-03	2.56E-02	2.16E-02	1.03E-01	1.41E-01	6.99E-03
	Spread	2.76E-01	2.97E-01	2.94E-01	2.99E-01	4.58E-01	2.79E-01	3.35E-01
D-NSGA-II-ELS	IGD	9.98E-03	8.71E-03	2.15E-02	1.48E-02	1.12E-01	1.54E-01	6.59E-03
	Spread	1.77E-01	1.59E-01	3.38E-01	2.65E-01	1.63E-01	1.62E-01	1.72E-01

Table 4 lists the mean IGD and Spread values of the 7 Benchmark functions generated by above three algorithms through 30 independent experiments. From **Table 4**, it can be seen that the D-NSGA-II-ELS obtains 5 best values of IGD and 6 best values of Spread, which means the Pareto fronts of the proposed algorithm is closer to the true Pareto fronts than other algorithms for these Benchmark functions. Especially, with the complexity of the search space on Benchmark functions, like UF2 and CTP1, the proposed algorithm still performs better in both the value of IGD and the value of Spread.

5.2 Experiments on multi-objective computation offloading

In order to evaluate the effectiveness of the proposed optimization algorithm for the constrained computation offloading problem in MCC environment, it is compared with the existing algorithms. The goal of this proposed algorithm is to achieve better offloading decision for multiple users with minimal mean execution energy, time and cost.

1) Simulation Settings

Various simulation parameters are set as follows. The number of mobile devices varies from 100 to 500. For each device, the CPU cycles are set from 1000 to 5000 Mega cycles and the size of computation input data is generated from 50 kB to 200 kB randomly. The CPU clock frequency is between 0.9 GHz and 1.1 GHz. The computation capability of the cloudlet and cloud server are set 3.0 GHz and 8.0 GHz respectively. When the device is idle, the power is set between 100 mW and 150 mW. Conversely, the processing power of the device is between 300 mW and 355 mW.

The WiFi radio power is generated from 155 mW to 205 mW, and 3G/4G radio power is from 200 mW to 255 mW. The corresponding cost per unit time is set 0.15 dollars and 0.25 dollars respectively. And we assume the cost per unit time is 0.25 dollars in cloudlet and 0.45 dollars in cloud respectively.

2) Experimental Results

In order to verify the practicality of the proposed algorithm, we compare it with other algorithms. The constraints are set as follows: $W_{\max}=7500$, $E_{\max}=500$, $T_{\max}=2500$ and $C_{\max}=32$. **Table 5** shows the average (Avg) values, Standard deviation (Sd) and 95% Confidence Interval (CI) of the energy consumption, time and cost under above constraints. **Fig. 4** gives the box plots of the three indicators' values obtained by all algorithms. The Pareto front obtained by three different algorithms is illustrated in **Fig. 3**.

Table 5. Comparison of energy, time and cost values obtained by the different algorithms

Algorithm	Energy(mj)			Time(ms)			Cost(￠)		
	Avg.	Sd	95% CI	Avg.	Sd	95% CI	Avg.	Sd	95% CI
NSGA-II	322.2	6.75	[319.1 , 325.4]	1686.8	18.76	[1678.2, 1695.7]	26.39	0.32	[26.23, 26.59]
LS-NSGA-II-DE	316.4	3.93	[314.6, 318.5]	1661.1	12.65	[1655.1, 1666.9]	27.37	0.37	[27.19, 27.55]
D-NSGA-II-ELS	301.2	2.69	[299.9, 302.4]	1623.3	6.38	[1620.3, 1026.2]	26.9	0.28	[26.85, 27.13]

From **Table 5**, **Fig. 3** and **Fig. 4**, it can be observed that the average energy and average time obtained by proposed D-NSGA-II-ELS is the lowest compared with other two algorithms, while NSGA-II is more prominent in the indicators of average cost. What' s more, in terms of the standard deviation and Confidence interval, the proposed algorithm performs better than those obtained by other algorithms, which means that the D-NSGA-II-ELS is the most stable among these three algorithms.

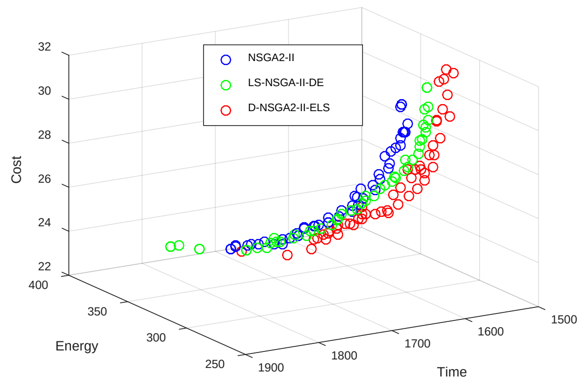


Fig. 3. The Pareto front obtained by the different algorithms for computation offloading

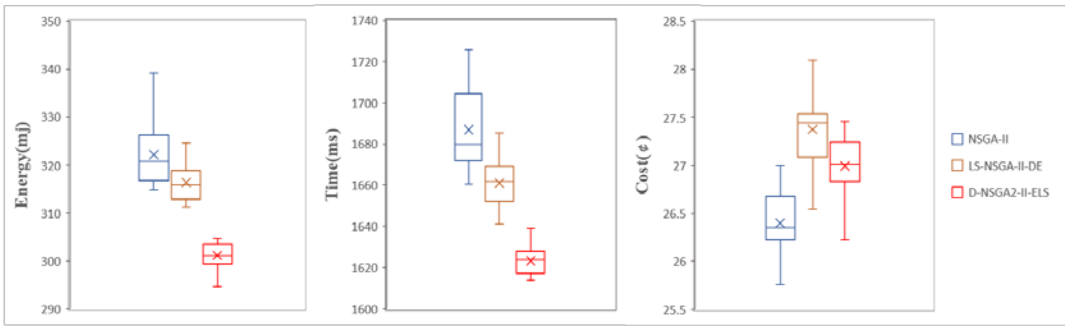


Fig. 4. The values obtained by three the different algorithms

Furthermore, we will pay attention to the impact of bandwidth and the changes of the number of users on the simulation performance. Firstly, the bandwidth limitation is set as 1000 kbps lower than the previous time, and the range of successive reductions is 15000 to 10000 kbps. And in this simulation, the number of users is 200 and other constraints is same as before. **Fig. 5** gives the figures of energy, time and cost. It can be seen that the proposed algorithm can maintain the optimal performance under three indexes (energy, time and cost) when the constraints become stricter. And although all three algorithms show a growing trend, the D-NSGA-II-ELS is relatively slow compared to the other two algorithms. This is because the more available bandwidth is not provided, the more tasks cannot be executed in cloudlet. In all the situations, the proposed algorithm is always able to give better solutions compared to other algorithms.

Then, in order to evaluate the performance when the number of users changes, some experimental parameters have been changed. In this simulation, each time 50 tasks are generated randomly based on last round offloading tasks. The number of offloading tasks varies from 100 to 500. **Fig. 6** shows the trend graph of the three algorithms. Obviously, with the number of the tasks increasing, the energy and time obtained by D-NSGA-II-ELS are still the lowest. And from the perspective of cost, compared to other algorithms, the proposed algorithm is the lowest when the number of users is above 200. This means that D-NSGA-II-ELS still has excellent search capabilities even when the search space becomes larger.

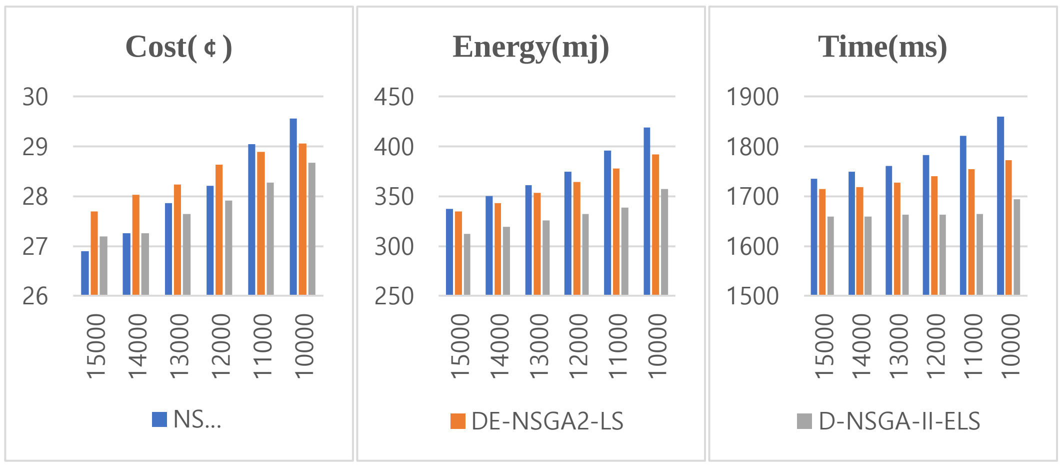
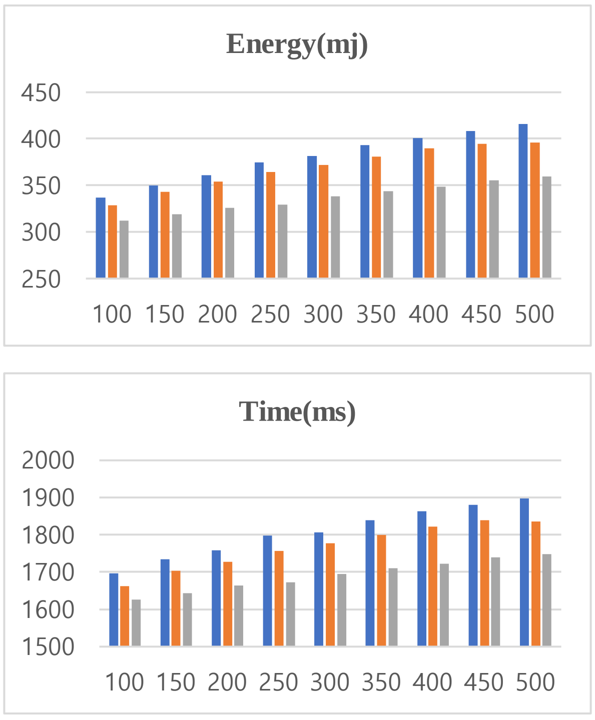


Fig. 5. The comparisons of different bandwidths in multiple users



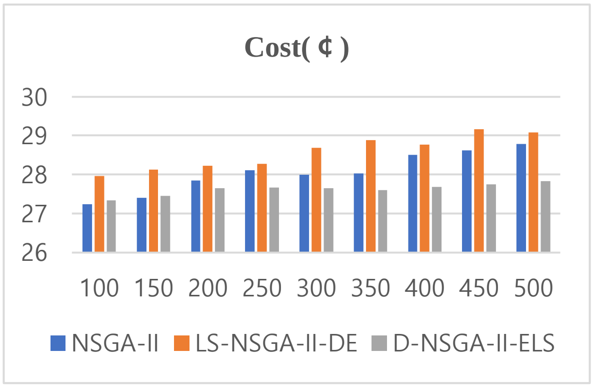


Fig. 6. The energy, time and cost under the conditions of different users

6. Conclusion

In this paper, a multi-objective optimization model for computation offloading is established to minimize the average energy consumption, average time and average cost of multiple users in the MCC environment. Furthermore, an improved algorithm D-NSGA-II-ELS is proposed to enhance the convergence and diversity of Pareto optimal solutions. The algorithm is based on NSGA-II algorithm, an adaptive DE mutation operator and NDX crossover operator are applied into this algorithm to expand the search space simultaneously. Then, an improved diversity maintaining strategy DMS is adopted to improve the uniformity of obtained Pareto optimal solution set. Finally, the elitist learning strategy ELS is performed on the non-dominated solutions in each generation, which avoids falling into local optimum and achieves better converge for Pareto solutions.

To demonstrate the advantages of the proposed algorithm, it is tested by several multi-objective benchmark functions and the computation offloading model. The results show that the D-NSGA-II-ELS has better performance in terms of optimization ability, convergence and diversity of the obtained Pareto optimal solutions compared to other algorithms. In addition, the proposed optimization method achieves better performance in the aspects of energy consumption, time and cost in computation offloading under MCC environment. In the future work, we will pay attention to the impact of users' mobility on this problem.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grants 61772068, 61370132 and 61472033.

References

[1] Z. Sanaei, S. Abolfazli, A. Gani, et al, "Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 369-392, May 2013. Article (CrossRef Link)

[2] X. Chen, L. Jiao, W. Li, et al, "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing," *IEEE.ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795-2808,

October 2015. Article (CrossRef Link)

[3] M. V. Barbera, S. Kosta, A. Mei, et al, "To Offload or Not to Offload? The Bandwidth and Energy Costs of Mobile Cloud Computing," in *Proc. of 2013 Proceedings IEEE INFOCOM*, July 2013. Article (CrossRef Link)

[4] Q. Xia, W. Liang, Z. Xu, et al, "Online Algorithms for Location-Aware Task Offloading in Two-Tiered Mobile Cloud Environments," in *Proc. of 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, February 2015. Article (CrossRef Link)

[5] S. Rashidi, S. Sharifian, "A hybrid heuristic queue based algorithm for task assignment in mobile cloud," *Future Generation Computer Systems*, vol. 68, pp. 331-345, March 2017. Article (CrossRef Link)

[6] H. Wu, y. Sun, K. Wolter, "Energy-Efficient Decision Making for Mobile Cloud Offloading," *IEEE Transactions on Cloud Computing*, January 2018. Article (CrossRef Link)

[7] V. Haghighi, N. S. Moayedian, "An Offloading Strategy in Mobile Cloud Computing Considering Energy and Delay Constraints," *IEEE Access*, vol. 6, pp. 11849 - 11861, March 2018. Article (CrossRef Link)

[8] M. Goudarzi, M. Zamani, A. T. Haghighat, "A fast hybrid multi-site computation offloading for mobile cloud computing," *Journal of Network and Computer Applications*, vol. 80, pp. 219-231, February 2017. Article (CrossRef Link)

[9] L. Liu, Z. Chang, X. Guo, et al, "Multi-objective optimization for computation offloading in mobile-edge computing," in *Proc. of 2017 IEEE Symposium on Computers and Communications (ISCC)*, September 2017. Article (CrossRef Link)

[10] X. Wang, J. Wang, X. Wang, et al, "Energy and Delay Tradeoff for Application Offloading in Mobile Cloud Computing," *IEEE Systems Journal*, vol. 11, no. 2, pp. 858 - 867, August 2015, Article (CrossRef Link)

[11] M. Goudarzi, Z. Movahedi, M. Nazari, "Efficient Multisite Computation Offloading for Mobile Cloud Computing," in *Proc. of 2016 Intl IEEE Conferences on Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*, January 2017. Article (CrossRef Link)

[12] M. H. Chen, B. Liang, M. Dong, "A semidefinite relaxation approach to mobile cloud offloading with computing access point," in *Proc. of 2015 IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2015. Article (CrossRef Link)

[13] H. Cao, J. Cai, "Distributed Multi-User Computation Offloading for Cloudlet based Mobile Cloud Computing: A Game-Theoretic Machine Learning Approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 752- 764, August 2017. Article (CrossRef Link)

[14] L. Li, X. Zhang, K. Liu, et al, "An Energy-Aware Task Offloading Mechanism in Multiuser Mobile-Edge Cloud Computing," *Mobile Information Systems*, vol. 2018, pp. 1 - 12, April 2018. Article (CrossRef Link)

[15] Z. Kuang, S. Guo, J. Liu, et al, "A quick-response framework for multi-user computation offloading in mobile cloud computing," *Future Generation Computer Systems*, vol. 81, pp. 166-176, April 2018. Article (CrossRef Link)

[16] M. H. Chen, B. Liang, M. Dong, "Multi-user Multi-task Offloading and Resource Allocation in Mobile Cloud Systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 10, pp. 6790 - 6805, August 2018. Article (CrossRef Link)

[17] M. H. Chen, M. Dong, B. Liang, "Multi-user Mobile Cloud Offloading Game with Computing Access Point," in *Proc. of 2016 5th IEEE International Conference on Cloud Networking*, vol. 4, pp. 64 - 69, December 2016. Article (CrossRef Link)

[18] X. Chen, "Decentralized Computation Offloading Game For Mobile Cloud Computing," *Parallel and Distributed Systems IEEE Transactions on*, vol. 26, no. 4, pp. 974-983, April 2014. Article (CrossRef Link)

[19] K. Kumar, Y. H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?," *Computer*, vol. 43, no. 4, pp. 51 - 56, April 2010. Article (CrossRef Link)

[20] D. Huang, P. Wang, D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Transactions on Wireless Communications*, vol. 11, no. 6, pp. 1991-1995, 2012. Article (CrossRef Link)

[21] K. Deb, A. Pratap, S. Agarwal, et al, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, August 2002. Article (CrossRef Link)

[22] S. Das, A. Abraham, et al, "Differential Evolution Using a Neighborhood-Based Mutation Operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526-553, June 2009. Article (CrossRef Link)

[23] N. Noman, H. Iba, "Enhancing differential evolution performance with local search for high dimensional function optimization," in *Proc. of Conference on Genetic and Evolutionary Computation. ACM*, pp. 967-974, January 2005. Article (CrossRef Link)

[24] X. Wang, Z Lv, Z. Tang, "Multi-objective dynamic optimal dispatching of grid-connected microgrid based on TOU power price mechanism," *Power System Protection and Control*, vol. 45, no. 4, pp. 9-18, February 2017. Article (CrossRef Link)

[25] R. Huang, X. Luo, B. Ji, et al, "Multi-objective optimization of a mixed-flow pump impeller using modified NSGA-II algorithm," *Science China Technological Sciences*, vol. 58, no. 12, pp. 2122-2130, December 2015. Article (CrossRef Link)

[26] S. Ding, C. Chen, B. Xin, et al, "A bi-objective load balancing model in a distributed simulation system using NSGA-II and MOPSO approaches," *Applied Soft Computing*, vol. 63, pp. 249-267, February 2018. Article (CrossRef Link)

[27] L. Liu, S. Gu, D. Fu, M. Zhang and R. Buyya, "A New Multi-objective Evolutionary Algorithm for Inter-Cloud Service Composition," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 1, pp. 1-20, January 2018. Article (CrossRef Link)

[28] K. Deb, A. Sinha, S. Kukkonen, "Multi-objective test problems, linkages, and evolutionary methodologies," in *Proc. of GECCO'06 Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp.1141-1148, July 2006. Article (CrossRef Link)

[29] E. Zitzler, K. Deb, L. Thiele, "Comparison of multi-objective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173-195, February 2000. Article (CrossRef Link)

[30] Q. Zhang, A. Zhou, S. Zhao, et al, "Multi-objective optimization Test Instances for the CEC 2009 Special Session and Competition," *Mechanical engineering*, January 2008. Article (CrossRef Link)

[31] E. Zitzler, L. Thiele, "Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, November 1999. Article (CrossRef Link)

[32] K. Deb, "Multi-objective Optimization Using Evolutionary Algorithms: An Introduction," *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, vol. 2, no. 3, pp. 3-34, September 2011. Article (CrossRef Link)

Li Liu was awarded PhD. from the University of Science and Technology Beijing in 2006. She is currently a professor in School of Automation and Electrical Engineering at University of Science and Technology Beijing, China. Her research interests are in the area of service composition and resource allocation in the Cloud Computing and Mobile Cloud Computing.

Yuanyuan Du received the B.S. degree in automation from University of Science and Technology Beijing, China, in 2017. She is currently pursuing the Master degree in Control Science and engineering at University of Science and Technology Beijing, China. Her research interest includes the computation offloading in the mobile cloud computing environment.

Qi Fan received the B.S. degree in automation from University of Science and Technology Beijing, China, in 2016. She is currently pursuing the Master degree in Control Science and engineering at University of Science and Technology Beijing, China. Her research interest includes the resource allocation and task scheduling in the mobile cloud computing environment.