EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

An energy- and cost-aware computation offloading method for workflow applications in mobile edge computing

Kai Peng[1,5], Maosheng Zhu[1], Yiwen Zhang[2*], Lingxia Liu[1], Jie Zhang[3], Victor C.M. Leung[4] and Lixin Zheng[5]

Abstract

Mobile edge computing is becoming a promising computing architecture to overcome the resource limitation of mobile devices and bandwidth bottleneck of the core networks in mobile cloud computing. Although offloading applications to the cloud can extend the performance for the mobile devices, it may also lead to greater processing latency. Usually, the mobile users have to pay for the cloudlet resource or cloud resource they used. In this paper, we bring a thorough study on the energy consumption, time consumption, and cost of using the resource of cloudlet and cloud for workflow applications in mobile edge computing. Based on theoretical analysis, a multi-objective optimization model is established. In addition, a corresponding multi-objective computation offloading method based on non-dominated sorting genetic algorithm II is proposed to find the optimal offloading strategy for all the workflow applications. Finally, extensive experimental evaluations are performed to show that our proposed method is effective and energy- and cost-aware for workflow applications in MEC.

Keywords: Mobile edge computing, Computation offloading, Time consumption, Multi-objective, Energy consumption, Cost

1 Introduction

With the development of computer network, cloud computing, as well as wireless sensor network, mobile devices (MDs) have become an indispensable part of people's daily lives [1–7]. In addition, the development of cyber-physical-social systems and big data has further affected people's life [8–14]. However, compared to a traditional device such as personal computer, a MD has certain limitations in computing power, storage capacity, especially for the battery capacity. Mobile cloud computing (MCC) brings new services and facilities to mobile users (MUs) to take full advantage of cloud computing [15–20]. However, the remote cloud is usually located far away from the MUs, which may result in high network latency in the process of data transmission. This inevitably reduces the quality of user service (QoS), in particular, for some applications, such as workflow applications (WAs), which generally have strict execution deadlines. The task may fail to be finished if the transmission latency is too large.

To solve the issue of network latency, a new paradigm called mobile edge computing (MEC) has been proposed [21–23]. MEC has become a key technology for realizing the Internet of Things and 5G [24]. The MEC can be regarded as a special example of MCC. A cloudlet is a type of edge server that provides various services to users in close proximity to MDs [25–27]. That means it can reduce the latency and energy consumption by offloading the WAs to cloudlet.

It is assumed that the resources of cloud are unlimited in MCC. Meanwhile, the resources of the MEC are limited. If there are multiple MUs requesting services from the cloudlet at the same time, a queue latency will occur. When a MU requests a service that exceeds the ability of a cloudlet, the cloudlet service cannot be obtained. In order to ensure the successful execution of the WA, further consideration needs to be taken to execute the task locally or offload them to the cloud. Moreover, the execution of WA by cloudlet or cloud needs to meet the task

*Correspondence: zhangyiwen@ahu.edu.cn
[2]School of Computer Science and Technology, Anhui University, Hefei, China
Full list of author information is available at the end of the article

deadline constraint, which further increases the difficulty of computing offloading.

Although the computation offloading issue in MCC has been well investigated (to list a few here [28–33], they cannot be used directly in computation offloading in MEC. The main reason is that MEC and MCC have completely different architectures. Existing studies on computation offloading in MEC have mainly focused on the optimization of a single objective (time consumption or energy consumption) [34, 35], but seldom considers multi-objective optimization. It is still a challenge to balance the multi-objective. Moreover, they focus on general applications and pay little attention to computation offloading of WAs in MEC [36]. For WA, it is very important to consider time constraints [37, 38].

Inspired by the idea that the MU has to pay for the resources they used in cloudlet or the cloud [36], in this study, we consider a joint energy consumption, time consumption, and price-cost optimization for WAs in MEC while the completion time of WA is considered as a constraint condition. The main contributions can be summarized as follows:

(1) The computation offloading for WAs in MEC has been well investigated in this paper. Both the energy consumption and time consumption, as well as the cost of MU by using cloudlet or cloud, are considered as the optimization objectives. Besides, based on theoretical analysis, a multi-objective optimization model is established.

(2) We propose a method named multi-objective computation offloading method for WAs (MCOWA) based on non-dominated sorting genetic algorithm for the solution. Some parameters in the algorithm step are improved to suit the needs of this problem.

(3) Compared to the other methods, such as the no offloading method, full offloading to cloud method, and full offloading to cloudlet method, extensive experiments and simulations have shown that our proposed method is effective and can provide optimization offloading strategy for MUs.

The abbreviations in this paper are shown in Table 1. The remainder of this paper is recognized as follows. In Section 2, system model and problem formulation are introduced. Section 3 elaborates multi-objective computation offloading algorithm for WAs (MCOWA). Section 4 presents the comparison analysis and performance evaluation. Section 5 summarizes the related work, and Section 6 concludes the paper and describes the future work.

2 System model and problem formulation

In this section, system model and problem formulation are presented. The basic architecture of MEC is described firstly. And then the basic mode is introduced. In addition, the time consumption mode, the energy consumption mode, and the cost mode are described.

Table 1 The abbreviations in this paper

Local area network	LAN
Multi-objective computation offloading method for workflow application	MCOWA
Mobile device(s)	MD(s)
Mobile edge computing	MEC
Mobile user(s)	MU(s)
Non-dominated sorting genetic algorithm II	NSGA-II
Quality of service	QoS
Virtual machine(s)	VMs
Workflow application(s)	WA(s)
Wide area network	WAN

The complete organization of MEC is shown in Fig. 1. Cloud is a combination of data centers. MD could be mobile phone or tablet. Each MD have one or more WAs which need to be processed. In general, these applications are time constrained. These applications can be executed directly locally, and users can migrate a part of the application or full application to cloudlet via local area network (LAN) or cloud via wide area network (WAN) according to their needs, to reduce user execution time or energy consumption or both of them.

2.1 Basic mode

As shown in Fig. 2, the WA is modeled by a direct acyclic graph $G_f(V,E)$, where f represents the fth WA($1 \leq f \leq F$) and F represents the total number of the WAs. Each application consists of multiple tasks and any node in Fig. 2 can be seen as a task. $V = \{v_{1,f}, v_{2,f}, \dots, v_{N,f}\}$ represents the set of tasks, and edge in E represents the set of dependency between any two tasks. Each edge is associated with a weight $d_{i,j}$ which represents the size of data transmission from task $v_{i,f}$ to task $v_{j,f}$. Cloudlet is configured as multiple virtual machines(VMs) for concurrent processing the WAs, which is modeled by a 3-tuple and denoted as $CIT = (M, f_{cl}, L_{\mathrm{LAN}})$. It is assumed that the capacity of cloudlet equals to the number of VMs in the cloudlet, thus M is the number of VMs in the cloudlet, f_{cl} is the processing capacity of the cloudlet and $L_{\mathrm{LAN}}$is the transmission latency in LAN. Each task $v_{i,f}$ in$V$ is modeled as a 2-tuple $v_{i,f} = (w_{i,f}, s_{i,f})$, where $w_{i,f}$ is the average number of instructions of task $v_{i,f}$, and $s_{i,f}$ is the offloading strategy for task $v_{i,f}$ which can be expressed as a one-dimensional vector $S = \{s_{i,f} | i = 1, 2, \dots, N_f, f = 1, 2, \dots, F\}$, where $N_f$represents the number of the tasks in the fth WA and $s_{i,f} = 0$ represents the task $v_{i,f}$ is processed locally, $s_{i,f} = 1$ represents $v_{i,f}$is offloaded to the cloudlet. Similarly, $v_{i,f} = 2$ represents $v_{i,f}$ is offloaded to the cloud.

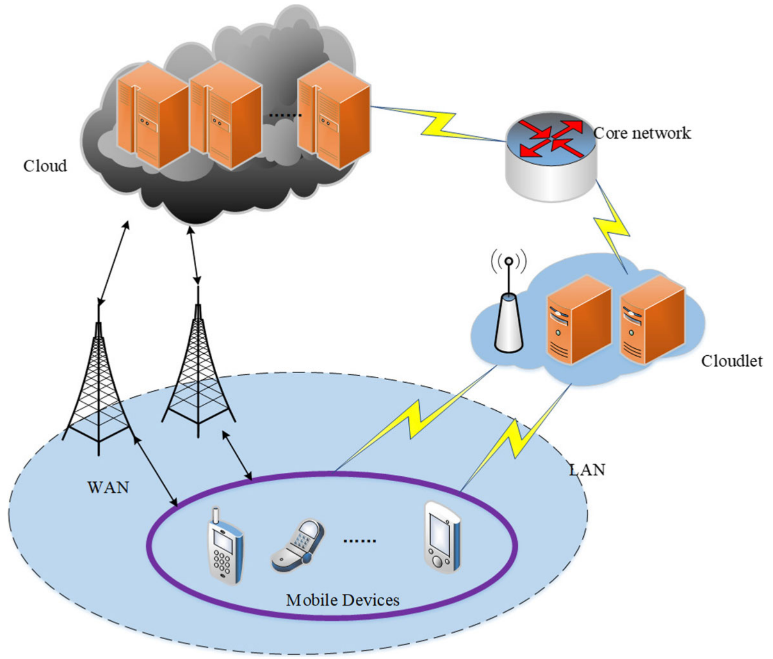


Fig. 1 MEC architecture

2.2 Time consumption mode

The total time consumption mainly contains three aspects, namely, waiting time, processing time, and transmission time.

2.2.1 *Average waiting time*

It is assumed that the interval of the task arrival time obeys the negative exponential distribution of the parameter λ, and the service time of the cloudlet is subjected to the negative exponential distribution of the parameter μ . Based on the queuing theory [39], the waiting time mode can be established. The probability that the cloudlet in idle is expressed as

$$p_0 = \left[\sum_{n=0}^{M-1} + \frac{\rho^M}{M!\,(1-\rho_M)} \right]^{-1} \tag{1}$$

where $\rho = \frac{\lambda}{\mu}$ indicates the utilization of cloudlets and $\rho_M = \frac{\rho}{M}$. Let p_n be the probability that the queue size reaches when the cloudlet is running in a steady state. Then, p_n is given as

$$p_n = \begin{cases} \frac{p^n}{M!\,M^{n-M}} \cdot p_0 & n \geq M \\ \frac{p^n}{n!} \cdot p_0 & n < M \end{cases} \tag{2}$$

When $n \geq M$, the probability of the tasks waiting in the cloudlet is given as

$$C_w(M,\rho) = \sum_{n=M}^{\infty} p_n = \frac{\rho^M}{M!\,(1-\rho_M)} \cdot p_0 \tag{3}$$

Based on the little theorem, the average waiting queue length L_q and the average queue length L_M are given as

$$L_q = \sum_{n=M+1}^{\infty} (n-M)p_n = \frac{p_n \cdot \rho^M}{M!} \cdot \sum_{n=M}^{\infty} (n-M)\rho_M^{n-M} \tag{4}$$

$$L_M = L_q + \rho \tag{5}$$

Overall, the average waiting time for tasks in the cloudlet is given as

$$W_q = \frac{L_M}{\lambda} - \frac{1}{\mu} = \frac{1}{M \cdot \mu - \lambda} C_w(M,\rho) \tag{6}$$

2.2.2 *Processing time and transmission time*

It is assumed that MD has a single CPU with processing capacity f_l, and the processing capacity of cloud is denoted as f_c. In addition, the processing capacity of each VMs in cloudlet is denoted as f_{cl}. If $s_{i,f} = 0$, the local processing time for the task $v_{i,f}$ is given as

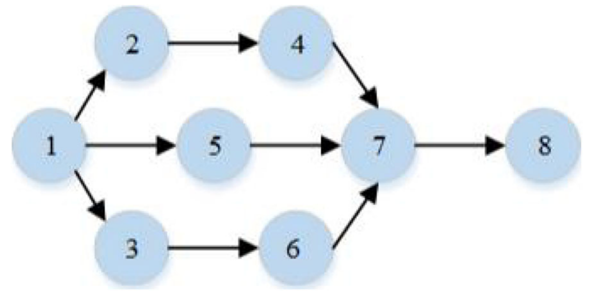


Fig. 2 Direct acyclic graph of workflow application

$$T_{\text{pro}}(v_{i,f}) = \frac{w_{i,f}}{f_l} \tag{7}$$

If $s_{i,f} = 1$, the processing time of the task $v_{i,f}$ is given as

$$T_{\text{pro}}(v_{i,f}) = W_q + \frac{w_{i,f}}{f_{cl}} + L_{\text{LAN}} \tag{8}$$

where L_{LAN} represents the transmission delay between the MD and the cloudlet. If $s_{i,f} = 2$, the processing time of the task $v_{i,f}$ is descried as

$$T_{\text{pro}}(v_{i,f}) = \frac{w_{i,f}}{f_c} + L_{\text{WAN}} \tag{9}$$

where L_{WAN} represents the transmission delay between the MD and the cloud. Let R denote the data rate of the wireless communication between the MD and the cloudlet or cloud. The computation time of data transmitted from the task $v_{i,f}$ to $v_{j,f}$ is given as

$$T_{\text{trans}}(v_{i,f}, v_{j,f}) = \frac{d_{i,j}}{R} \tag{10}$$

It is assumed that if task $v_{i,f}$ and task $v_{j,f}$ are processed at the same side, $s_{i,f} = s_{j,f}, T_{\text{trans}} = 0$. If $(s_{i,f}, s_{j,f}) \in \{(0,1),(1,0)\}$, it means the data is communicated between MD and cloudlet through LAN with the rate R_{cl}. If $(s_{i,f}, s_{j,f}) \in \{(0,2),(2,0)\}$, it means the data is communicated between MD and cloud through WAN with the rate R_c. If $(s_{i,f}, s_{j,f}) \in \{(1,2),(2,1)\}$, it means the data is transmitted between cloudlet and cloud and the transmission time can be ignored, namely $T_{\text{trans}} = 0$. Therefore, the total time consumption of the $f - th$ WA which includes the average waiting time, the processing time and transmission time is given as follows.

$$T_{wa,f}(S) = \sum_{i=1}^{N} T_{\text{pro}}(v_{i,f}) + \sum_{i=1}^{j-1} \sum_{j}^{N} T_{\text{trans}}(v_{i,f}, v_{j,f}) \tag{11}$$

2.3 Energy consumption model

The total energy consumption of WA consists of the energy consumption of processing and transmission. $E_{\text{pro}}(v_{i,f})$ indicates the energy consumption generated during the processing of the task $v_{i,f}$, while $E_{\text{trans}}(v_{i,f}, v_{j,f})$ represents the energy consumption generated by the data transmission from the task $v_{i,f}$ to the task $v_{j,f}$ on MDs. The formulation is given as

$$E_{\text{pro}}(v_{i,f}) = \begin{cases} \dfrac{w_{i,f}}{f_l} \cdot p_A & s_{i,f} = 0 \\ \left(W_q + \dfrac{w_{i,f}}{f_{cl}} + L_{\text{LAN}}\right) \cdot p_I & s_{i,f} = 1 \\ \left(\dfrac{w_{i,f}}{f_c} + L_{\text{LAN}}\right) \cdot p_I & s_{i,f} = 2 \end{cases} \tag{12}$$

where p_A and p_I, respectively, represent the power when the MD is in the active state and idle state. The transmission energy consumption from task $v_{i,f}$ to task $v_{j,f}$ is described as

$$E_{\text{trans}}(v_{i,f}, v_{j,f}) = \frac{d_{i,j}}{B} \cdot p_T \tag{13}$$

where, p_T represents the transmitted power of the MD. Therefore, the total energy consumption of the fth WA is given as

$$E_{wa,f}(S) = \sum_{v_{i,f} \in V} E_{\text{pro}}(v_{i,f}) + \sum_{i=1}^{j-1} \sum_{j}^{N} E_{\text{trans}}(v_{i,f}, v_{j,f}) \tag{14}$$

2.4 Cost mode

Additionally, the MU has to pay for the resources it used in the cloudlet or the cloud. It is assumed that the per price for the cloudlet is a and the remote cloud is $2a$. The expression means that if the user's task is processed locally and the cost is 0. If the task is offloaded to the cloudlet for processing, the cost will be a. Similarly, if the task is offloaded to the cloud, the cost will be $2a$. The average cost of WA is given by Eq. (16), where N is the total nodes (tasks) of a WA.

$$C = \begin{cases} 0 & s_{i,f} = 0 \\ a & s_{i,f} = 1 \\ 2a & s_{i,f} = 2 \end{cases} \tag{15}$$

$$E_{wa,f}(S) = \frac{1}{N} \cdot \sum_{v_{i,f} \in V} C \tag{16}$$

2.5 Problem formulation

The object in this study is to optimize the energy consumption and time consumption, as well as cost of all WAs while meeting the deadline constraint given by WAs. It can be formulated as follows

$$Min\ T_{wa,f}(S), \forall f \in \{1,2\ldots F\} \tag{17}$$

$$Min\ E_{wa,f}(S), \forall f \in \{1,2\ldots F\} \tag{18}$$

$$Min\ C_{wa,f}(S), \forall f \in \{1,2\ldots F\} \tag{19}$$

$$s.t.\ T_{wa,f}(S) \le D(f), \forall f \in \{1,2\ldots F\} \tag{20}$$

$$s_i \in \{0,1,2\} \tag{21}$$

where $D(f)$ represents the deadline of the fth WA which is given in advance.

3 Multi-objective computation offloading algorithm for workflow applications (MCOWA)

In this section, the details of MCOWA are described. We firstly introduce the main steps of the MCOWA in Section 3.1, and then the description and the corresponding algorithm pseudocode of MCOWA are shown in Section 3.2.

3.1 The main steps of mCOWA

In this paper, our objective is to optimize the energy consumption, time consumption, and cost for multi-WA. The computation offloading problem is defined as a multi-objective problem. NSGA-II [40] is used to obtain the optimal computation offloading strategies for all WAs.

Compared to the traditional genetic algorithm, NSGA-II can find the global optimal solutions among the feasible solutions quickly and accurately. The implementation of NSGA-II is shown in Fig. 3, which consists of five steps. Notice that the detail implementation of step 4 is shown in Algorithm 1 and we also introduce how it will be called by Algorithm 2 in Section 3.2. The details of each step are shown as follows.

3.1.1 Step 1: Encoding

The WA is numbered by using the results of a topological with an integer index and starts from $\{0, 1, \ldots, \}$. The gene denotes the value of each decision variable and also represents the offloading strategy of each task of WA. Multiple genes form a complete chromosome which can also be seen as an individual, representing one solution to the optimization problem. Numbers of individuals form a population, which indicates the diversity of solution. Each chromosome represents a computational offloading strategy for F WAs. The integer coding method is used, namely, each offloading strategy is encoded as $\{0, 1, 2\}$. The number 0 indicates that each task of WA is processed by MD itself and the number 1 represents the task of WA is offloaded to the cloudlet. Similarly, the number 2 represents the task of WA is offloaded to the cloud on the basis of the offloading strategies.

As shown in the Fig. 4, an example of encoding is given. Each box on the lower chromosome represents a gene and also indicates a task of the WA. The possible value of each gene is $\{0, 1, 2\}$ and denoted as $\nu_i = 0$ $\nu_i = 1$ o$\nu_i = 2$, which is represented above the gene. In addition, the gene with the same color means that they have the same offloading strategy.

3.1.2 Step 2: Fitness functions and constraints

Fitness function is the criterion for evaluating individual quality, which is given by Eqs. (17), (18), and (19). The three fitness functions of the computing offloading model (17), (18), (19) represent the time consumption, the energy consumption and the cost, respectively. It is necessary to make tradeoffs among the multiple objective functions. Namely, we need to obtain the best offloading strategy to make these three fitness functions well.

Additionally, for each WA, the completion time is obtained according to the computation offloading strategy. If the time constraint of formula (20) is not satisfied, the chromosome represented by the offloading strategy will not be considered in the selection process. The chromosome that satisfies the time constraint is called the valid chromosome.

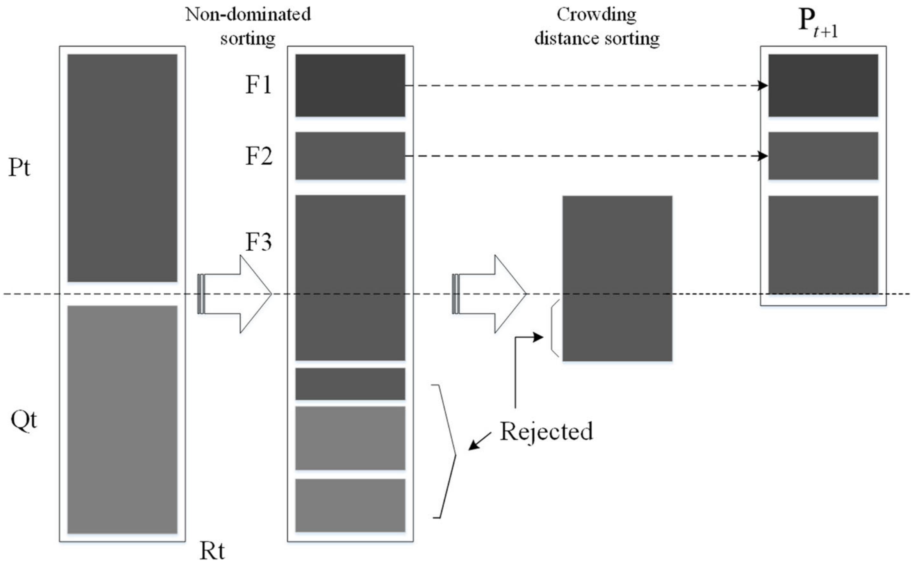


Fig. 3 The implementation of NSGA-II

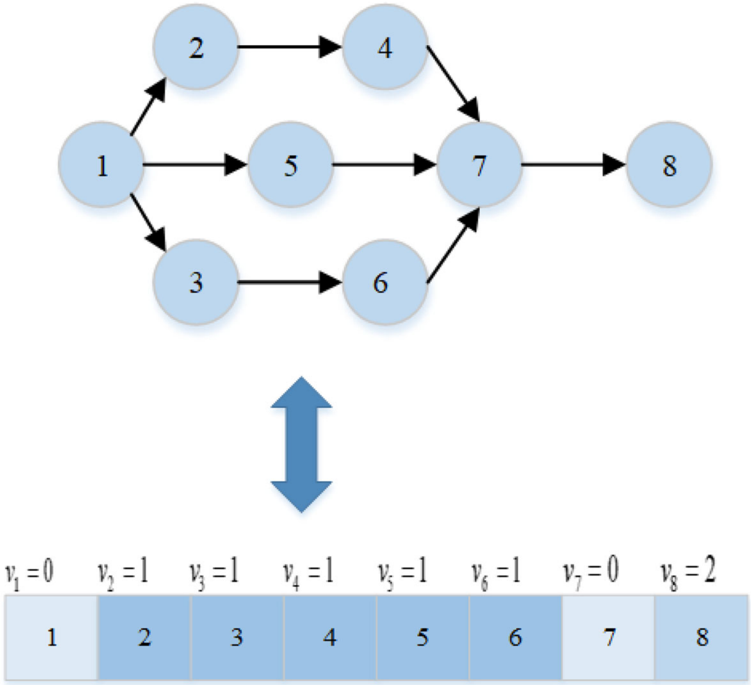


Fig. 4 An example of encoding

3.1.3 Step 3: Initialization, selection, crossover, mutation

To generate initialized population P_0 randomly and then the binary tournament selection, crossover and mutation are performed on P_0 to obtain new population Q_0.

The crossover operation is to cross the corresponding genes of two individuals and select two points on chromosome based on the gene fragments which enhance the adaptability, i.e., crossover point, to exchange the middle part of gene value v_i. An example of crossover operation is shown in Fig. 5.

A mutation operation based on enhancing population adaptability is proposed in MCOWA. Each gene value of each individual is mutated with the mutation probability, which is given in advance. And the gene fragment with an added value of 1 is mutated. An example of mutation is given in Fig. 6.

3.1.4 Step 4: Non-dominated sort

Form a new group population $R_t = P_t \bigcup Q_t$, where $t = 0$. Additionally, the non-dominated front-end $F_1, F_2, \ldots$ is obtained by non-dominated sort of R_t.

The main steps are shown as follows [40, 41]

(1) The parameters are initialed, and the population size SCALE is determined at the same time. In addition, the attribute of the individual chromosomes dominated the number of those in the tagged population is set, if dominated $=$ 0, then the individual chromosome set the individual dominated empty.

(2) An individual chromosome is selected sequentially in the population, which is compared to other individuals in the population based on the dominance relation. If the compared individual dominate the selected individual chromosome, let dominated $=$ dominated $+$ 1; if the

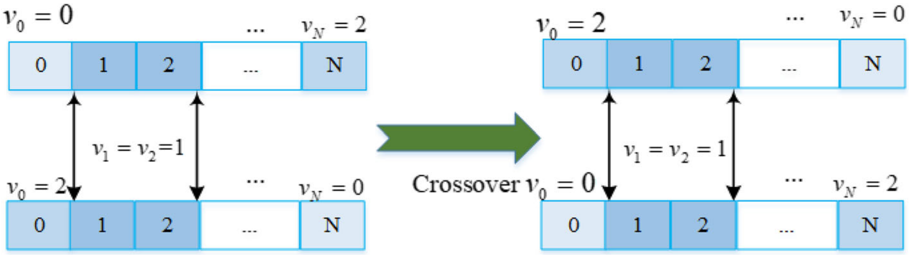


Fig. 5 An example of crossover

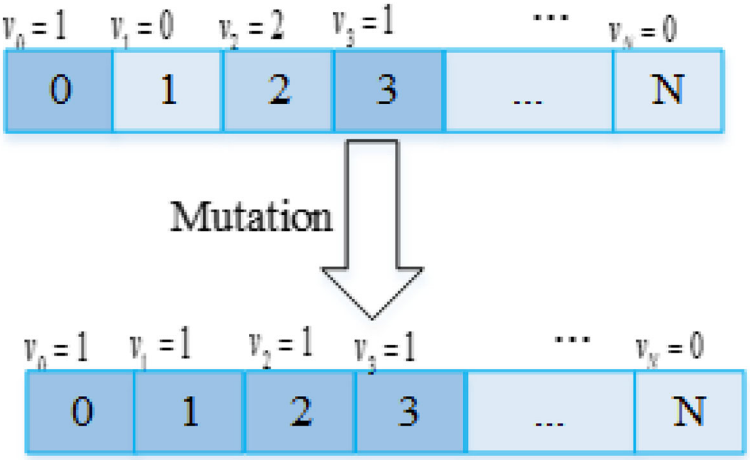


Fig. 6 An example of mutation

compared individual is dominated by the selected individual, add the compared individual to the individual set which is dominated by the chromosome.

(3) Repeat (2) until processing the dominated attribute of the N chromosomes and their dominated individual set.

(4) The population is traversed and the chromosome whose dominated attribute is 0 is added to the rank 1.

(5) The individual chromosome in the rank established is selected sequentially in (2), meanwhile, all of the individuals' attribute in the set of individuals is operated by auto-decrement, and thus, dominated $=$ dominated $-$ 1. If dominated $=$ 0, add the individual to the rank of next level.

(6) Repeat (5) until it is empty that the dominated individual set of the individual chromosome is.

The pseudo-code of the fast non-dominated sorting approach is shown in Algorithm 1.

Algorithm 1 Fast Non-dominated Sort(R)

```
Input: The population of $2N_{pop}$ size;
Output: Non-dominated sets;
for each $r \in R$ do
    $S_r = \emptyset$
    $n_r = 0$
    for each $q \in R$ do
        if $(r \prec q)$ then
            $S_r = S_r \cup q$
        else if $(q \prec r)$ then
            $n_r = n_r + 1$
        end if
    end for
    if $n_r = 0$ then
        $r_rank = 1$
        $H_1 = H_1 \cup \{r\}$
    end if
end for
$i = 1$
while $H_i \neq \emptyset$ do
    $Q = \emptyset$
    for each $r \in H_i$ do
        for each $q \in S_r$ do $n_q = n_q - 1$
            if $n_q = 0$ then
                $q_rank = i + 1$
                $Q = Q \cup q$
            end if
        end for
$i = i + 1$
$H_i = Q$
    end for
end while
return $E_n$;
```

3.1.5 *Step 5: Crowding distance calculation*

All F_i are sorted according to the crowding distance comparison operation $\prec_n$, and the best N individuals are selected to form a population P_{t+1}. The congestion distance formula is shown in (22)

$$
\begin{aligned}
i_d = i_d^T + i_d^E + i_d^C = \sum_{f=1}^{F}\left(|T_{wa,f}^{i+1}(S) - T_{wa,f}^{i-1}(S)|\right) \\
+ \sum_{f=1}^{F}\left(|E_{wa,f}^{i+1}(S) - E_{wa,f}^{i-1}(S)|\right) \quad (22) \\
\sum_{f=1}^{F}\left(|C_{wa,f}^{i+1}(S) - C_{wa,f}^{i-1}(S)|\right)
\end{aligned}
$$

where i_d represents the crowding distance of the its offloading strategy, $s_{i,f}$ represents the fth WA. i_d^T, i_d^E, and

i_d^C represent the objective functions, respectively. $T_{wa,f}^{i+1}(S)$ represents the value of the offloading strategy $s_{i+1,f}$ to the objective function $T_{wa,f}(S)$. In addition, $E_{wa,f}^{i+1}(S)$ represents the value of the offloading strategy $s_{i+1,f}$ to the objective function $E_{wa,f}(S)$. Similarly, $C_{wa,f}^{i+1}(S)$ represents the value of the offloading strategy $s_{i+1,f}$ to the objective function $C_{wa,f}(S)$.

The population P_{t+1} is subjected to replication, crossover, and mutation operations to form a population Q_{t+1}. If the termination condition is true (the maximum number of iterations is met), then it ends. Otherwise, $t = t + 1$ and goes to Step 2.

3.2 MCOWA pseudocode

The pseudocode of MCOWA method is shown in Algorithm 2. The input of the Algorithm 2 are multiple WA. The algorithm starts from the first iteration (Line 1). Firstly, the population is initialized, the chromosomes whose complete time does not satisfy the deadline constraint are removed from the population, and the new generation is denoted as $P_t^{'}$ (Line 3 to 8). Two populations $P_t^{'}$ and O_t of size N are randomly generated and form a population R_t with a population size of $2N$ (Line 9 to 10). The population R_t is divided into multiple non-dominated layers by calling Algorithm 1(Line 11). F is prepared for the selection operation, and population P_{t+1} is set to empty and store the new generation of the population (Line 12). Additionally, the excellent individuals are selected to fill in a new population of size N according to crowding distance (Line 13 to 18). Then the offspring population is generated after the crossover and mutation and put into Q_{t+1} (Line 19). The offspring population is merged with the parent population and iterated again until the algorithm ends (Line 2 to 22). Finally, the optimal offloading strategies, the time consumption, energy consumption, and cost of all WAs are output (Line 23).

Algorithm 2 MCOWA

```
Input: WAs, population size N,the maximum number of iterations $Generationmax$
Output: Optimal computation offloading strategy $S_t$,energy consumption $E$,time consumption $T$,cost $C$
1:  $Gen = 1, t = 1$
2:  while $Gen \leq Generationmax$ do
3:      Initialize the population$P_t$
4:      for every chromosome in$P_t$ do
5:          if each WA completion time meets the time constraint$D(f)$ then
6:              $P'_t = P_t - M \parallel P'_t$valid chromosome
7:          end if
8:      end for
9:      $Q_t$ =selection, crossover and mutation$P'_t$
10:     $R_t = P'_t + Q_t$
11:     $F$ =Algorithm 1$(R_t)$
12:     $P_{t+1} = \emptyset$
13:     $i = 0$
14:     while $len(P_{t+1}) + len(F[\,i\,]\,) < N$ do
15:         $crowdingdistanceassignment(F[\,i\,]\,)$ by formula 22
16:         $P_{t+1} += F[\,i]$
17:         $i = i + 1$
18:         $P_{t+1} += F[\,i]\,[\,0 : N - len(P_{t+1})]$
19:         $Q_{t+1}$=make new generation$(P_{t+1})$
20:         $t = t + 1$
21:         $gen = gen + 1$
22:     end while
23: end while
24: return $S_t$,energy consumption E,time consumption T,cost C
```

4 Experimental evaluation

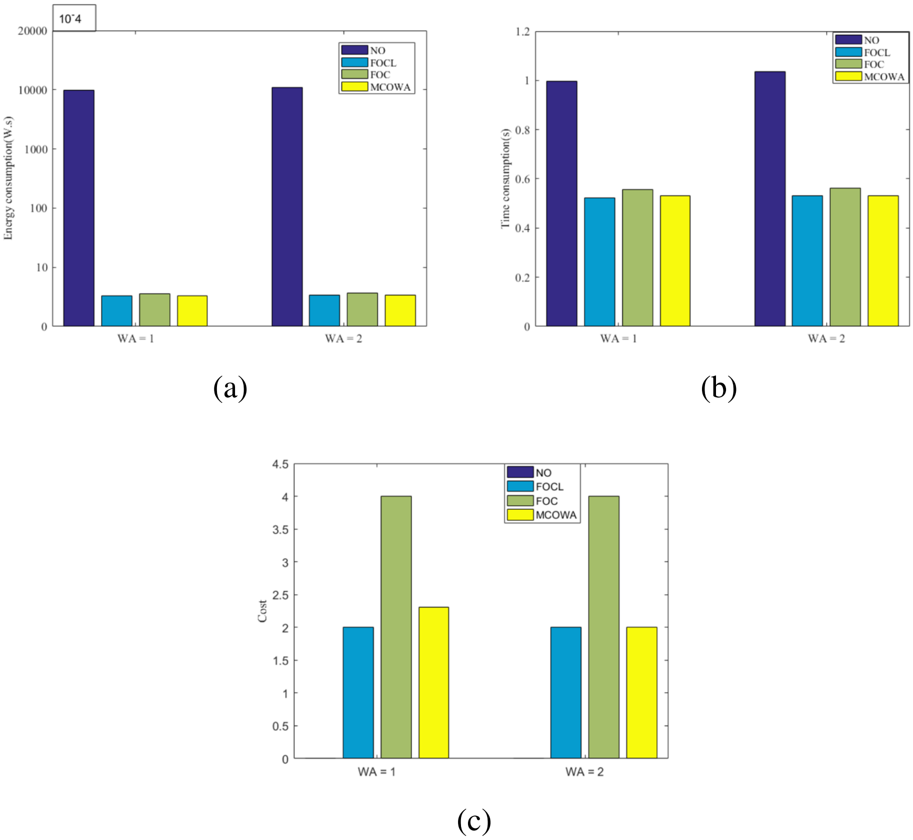
In this section, a comprehensive simulation and experiment are carried out to evaluate the performance of the proposed MCOWA method. Specifically, the simulation setup is introduced firstly, including the experimental parameter settings and other comparative methods. Then, the influence of different WA scales on the energy consumption performance, time consumption performance, and cost performance of the compared methods and MCOWA is evaluated.

4.1 Experimental settings

In order to make a comparative analysis, we propose some other computation offloading methods in addition to our MCOWA method. The comparative methods are introduced as follows.

Table 2 Parameter settings

Parameter	Value
The power of MDs when CPU is idle state	0.001 W
The power of MDs when CPU is active state	0.5 W
The transmission power of MDs	0.1 W
The processing capacity of MDs	500 MHZ
The processing capacity of the cloudlet	2000 MHZ
The processing capacity of the cloud	3000 MHZ
The latency of LAN	1 ms
The latency of WAN	30 ms
The bandwidth of LAN	100 kb/s
The bandwidth of WAN	50 kb/s
The average waiting time of tasks in the cloudlet	20 ms
The cost of cloudlet for each task	2
The cost of cloud for each task	4

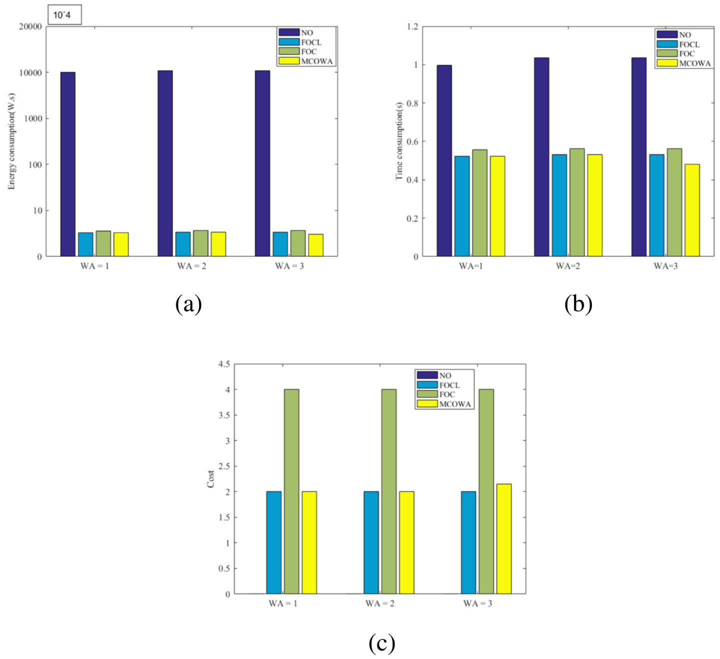


(a)

(b)

(c)

Fig. 7 The energy consumption, time consumption, and cost of each method where WAs = 2. **a** Energy consumption. **b** Time consumption. **c** Cost

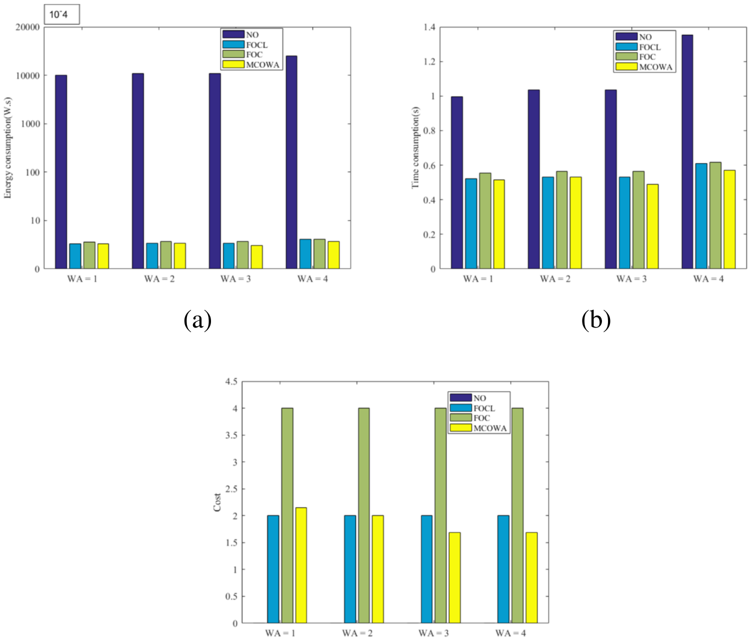


(a)

(b)

(c)

Fig. 8 The energy consumption, time consumption, and cost of each method where WAs = 3. **a** Energy consumption. **b** Time consumption. **c** Cost



(a)

(b)

(c)

Fig. 9 The energy consumption, time consumption, and cost of each method where WAs = 4. **a** Energy consumption. **b** Time consumption. **c** Cost

No offloading (NO): All tasks of a WA are processed on the MD. There is no transmit overhead between any two tasks. In addition, there is no cost of using resources of cloudlet of cloud, named as NO.

Full offloading to cloud (FOC): All computation tasks of WAs are moved from the local MD to the remote cloud for processing, named as FOC.

Full offloading to cloudlet (FOCL): All computation tasks of WAs are moved from the local MD to the cloudlet for processing, named as FOCL.

MCOWA: With the help of the MCOWA, all tasks are partitioned into three sets, one for local processing on the MD and another for remote processing on cloud, and the other for cloudlet processing.

We use the same parameter settings with reference [42] and the value of some new parameters are presented. The details are shown in Table 2. The methods are implemented base on JAVA language by using the tool of Eclipse on a PC machine with 2 Intel Core i5-5200U 2.20GHz pr ocessors and 4GB RAM. The operating system is Win7 64.

4.2 Performance evaluation

We have received different results under the different parameters of WAs number. Fifty experiments are performed in the case of convergence for each WA scale. Firstly, we discuss how MCOWA balance the three objectives.

As shown in Fig. 7, we can see that MCOWA is effective for two mobile users. Similary, it also can be used for the scenario of three users and more users based on the experimental results which are shown in Figs. 8, 9, 10, 11, and 12. More specifically, we can conclude that each user can obtain the best results from the perspective of energy consumption and time consumption.

We can conclude that MCOWA is effective with the increasing of number of WAs. More specially, a WA consists of 13 tasks. So two WAs are 26 tasks and so on. Additionally, as shown in Figs. 7, 8, 9, 10, 11, and 12, compared to FOC and FOCL, MCOWA has a smaller cost. If we only consider the cost factor, there is no cost for local processing and it seems that MCOWA is not better than

Table 3 The number of WAs = 2

Location	NO	FOCL	FOC	MCOWA
Local	26	0	0	0
Cloudlet	0	26	0	24
Cloud	0	0	26	2

Table 4 The number of WAs = 3

Location	NO	FOCL	FOC	MCOWA
Local	39	0	0	2
Cloudlet	0	39	0	35
Cloud	0	0	39	2

Table 5 The number of WAs = 4

Location	NO	FOCL	FOC	MCOWA
Local	52	0	0	4
Cloudlet	0	52	0	47
Cloud	0	0	52	1

NO. However, MCOWA minimizes the time consumption and energy consumption of WA while ensures the cost is within a certain acceptable range. Overall, MCOWA is effective as the offloading strategy make these three objects better, not just to make one of them work best.

Secondly, we discuss how MCOWA provide effective strategy to balance three offloading destinations, namely local, cloudlet, and cloud. As shown in Tables 3, 4, 5, 6, 7, and 8, as the cost is in an acceptable range, the task is mainly offloaded to the cloudlet. That means the cloudlet is the optimal offloading destination. If the number of tasks exceeds the processing capacity of the cloudlet, some tasks will be offloaded to the cloud in order to reduce the queue latency while meeting the deadline constraint of task. In addition, as the number of WAs increases, the number of corresponding tasks increases. The competition for computing resources of cloudlet among WAs will be more intense as cloudlet is still the first choice to offload. Considering the limitations of cloudlet resources and the cost processing tasks in cloud, the number of tasks which are executed locally will increase and the number of tasks which are offloaded to the cloud will be decreased, which also proves that our method strategy is effective and can balance each offloading destinations well.

5 Related work

MCC brings new services and facilities to MUs to take full advantage of cloud computing. However, the remote cloud is usually located far away from the MUs, which may result in high network latency. This inevitably reduces QoS of MUs. In addition, MUs have to pay for the resources of cloud they use. MEC is a new paradigm can be seen as a example of MEC. Different from the MCC, MEC is a three-layer architecture. The edge server is a bridge which well connects the MU and cloud. MU can connect to edge server or cloud according to requirement of service. There are many kinds of edge servers [43]. Cloudlet is a type of edge server, which has been widely used in LAN and WMAN [25–27, 44–46]. Cloudlet is a low-cost infrastructure with rich computer resources, high bandwidth, and sufficient power. With the help of cloudlet, MU can improve QoS by computation offloading.

Table 6 The number of WAs = 5

Location	NO	FOCL	FOC	MCOWA
Local	65	0	0	7
Cloudlet	0	65	0	56
Cloud	0	0	65	2

Table 7 The number of WAs = 6

Location	NO	FOCL	FOC	MCOWA
Local	78	0	0	11
Cloudlet	0	78	0	66
Cloud	0	0	78	1

Computation offloading was originally studied in MCC [28–33]. The offloading mode in MEC is similar with the mode in MCC, but the main difference is the location of offloading. It is commonly assumed that the offloading destination of computation offloading for MCC is the remote cloud, while the offloading destination of MEC is a edge server such as cloudlet. Different from the cloudlet, it is assumed that the resources in remote cloud are unlimited. Besides, MCC can be seen as a two-layer architecture, while MEC can be seen as a three-layer architecture.

Jia et al. [28] made a thorough study on how to divide and migrate applications in MCC by using heuristic algorithm. The relationship between tasks in the WA is abstracted into serial and parallel, and the general WA is seen as a combination of the two. The core idea is to reduce the total processing latency of the task by increasing the parallelism between the local and the cloud. According to the unstable wireless channel and unstable service node in the MCC, Wu et al. [29] proposed a min-cost offload partitioning algorithm to find the best partitioning plan and minimize processing time and energy consumption. In view of the sequence of task processing in the WA, dynamic voltage and frequency scaling is used to set a number of flag bits to construct a joint optimization function of latency and energy consumption in [30]. The task scheduling strategy based on simulated annealing algorithm is proposed to optimize the processing time consumption and energy consumption of the WA.

There are many studies on WA schedule in MCC. Deng et al. [31] propose a novel offloading system to design robust offloading decisions for mobile services. The dependency relations among component services are taken into consideration. The objectives of them are to optimize execution time and energy consumption of mobile services. Xu et al. [32] propose an energy

Table 8 The number of WAs = 7

Location	NO	FOCL	FOC	MCOWA
Local	91	0	0	14
Cloudlet	0	91	0	74
Cloud	0	0	91	3

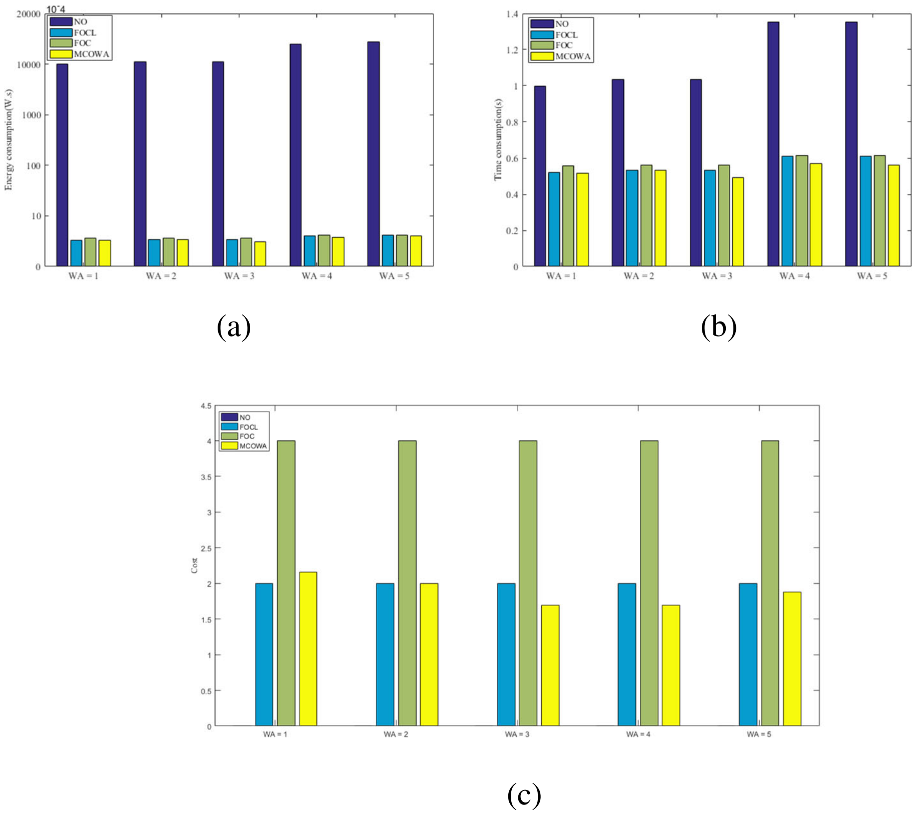


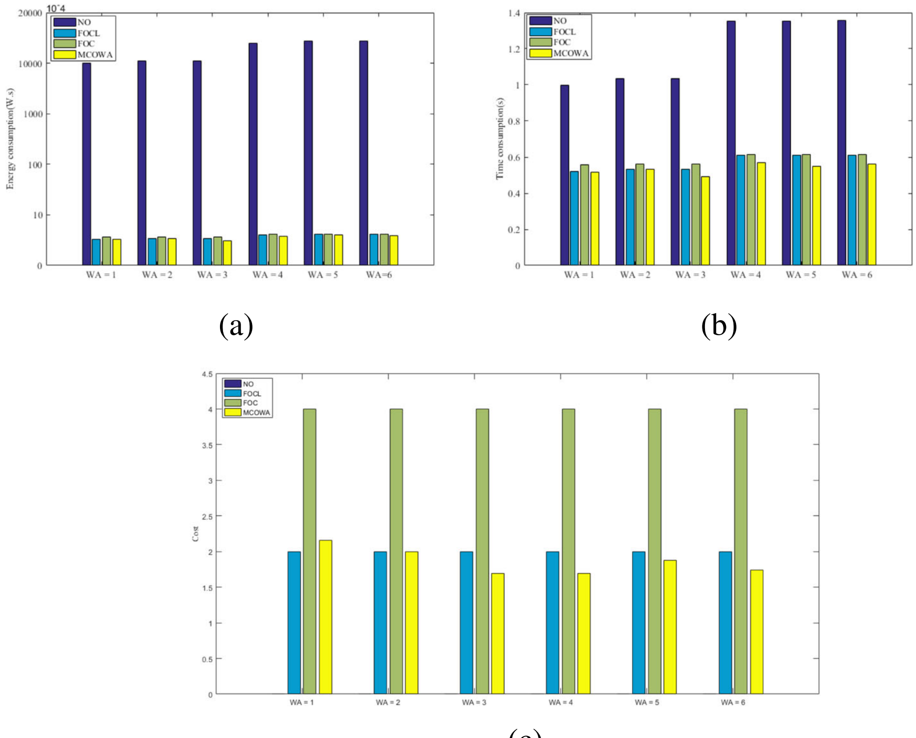
Fig. 10 The energy consumption, time consumption, and cost of each method where WAs = 5. **a** Energy consumption. **b** Time consumption. **c** Cost

consumption model for applications deployed across cloud computing platforms, and a corresponding energy-aware resource allocation algorithm for VMs scheduling to accomplish scientific workflow executions. Aiming at the problem of scientific WA scheduling with deadline constraints in multi-cloud environment, an adaptive discrete particle swarm optimization algorithm is proposed in [33], which can reduce the processing cost of WA while meeting the deadline of WA. As MEC and MCC have different architectures, the computation offloading methods in MCC cannot be used for the MEC scenario directly.

Jia et al. [34] proposed a computational offloading algorithm for augmented reality applications in MEC environment. They hold the opinion that such applications are multi-user participation and have high latency requirements. They have established a multi-user augmented reality game system model and proposed a corresponding multi-user computing offloading algorithm. Li et al. [35] proposed a migration algorithm that divides the application into multiple parts and migrates them to multiple cloudlets to minimize task computation latency. However, their methods mainly focus on latency optimization. Liu et al. [36] utilize a queuing theory to study on the energy consumption, processing latency, and price cost of offloading process in MEC system. The secularization scheme and interior point method are used to address the formulated problem. They are mainly for general applications in the MEC and do not consider the computation of WAs in the MEC.

Zhang et al. [47] propose an energy-efficient offloading strategy for home automation applications in MEC. An improved particle swarm optimization algorithm is implemented to schedule mobile services which minimizes the total energy consumption of the WAs within a given constant deadline. However, their approach focuses on the single-user and single-objective optimization scenario. Huang et al. [48] proposed a computation offloading method for multimedia workflows with deadline constraints in cloudlet-based mobile cloud. The objective of them is to minimize the energy consumption with the constraints of meeting the deadline of each multimedia workflow. In addition, a multi-objective computation offloading method for WA is proposed in terms of energy consumption and time consumption [42].

Different from the existing research, we investigate the multi-objective computation offloading for WAs in terms of time consumption, energy consumption, and cost for WAs in MEC.

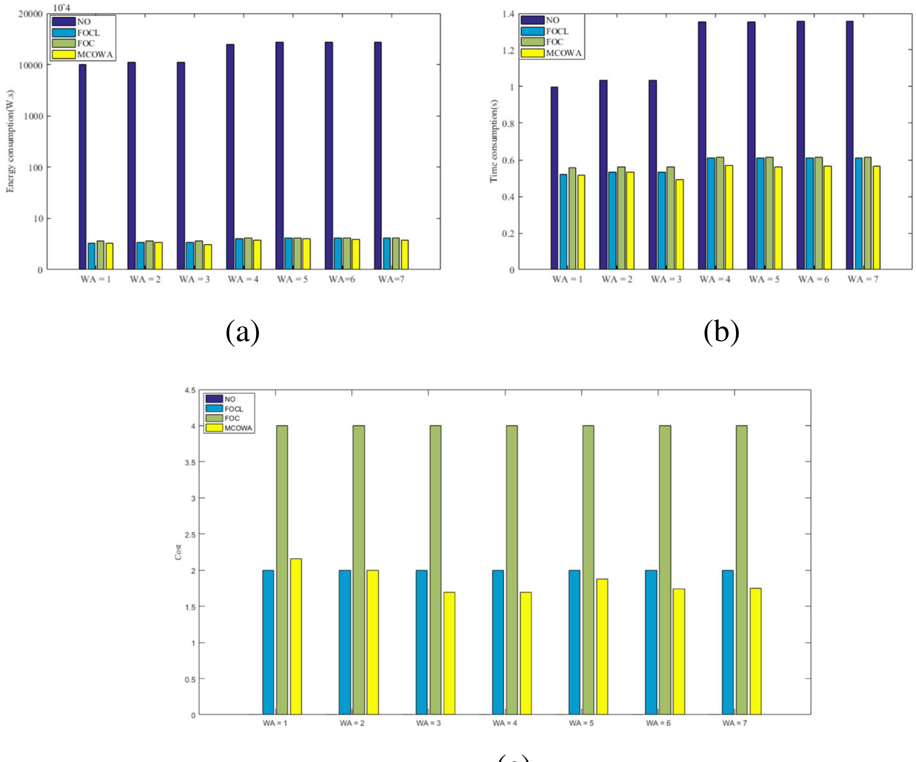


(a)

(b)

(c)

Fig. 11 The energy consumption, time consumption, and cost of each method where WAs = 6. **a** Energy consumption. **b** Time consumption. **c** Cost



(a)

(b)

(c)

Fig. 12 The energy consumption, time consumption, and cost of each method where WAs = 7. **a** Energy consumption. **b** Time consumption. **c** Cost

6 Conclusion

In this paper, we investigate the multi-objective computation offloading method for WAs in MEC. To tackle the problem, we have proposed a computation-offloading algorithm (MCOWA) that finds the optimal application strategy while meeting the deadline-constrained of WAs. Extensive experimental evaluations have conducted to show the efficiency and effectiveness of our proposed method.

In future work, we will focus on multi-objective optimization computation offloading from the perspective of edge servers in MEC. For one thing, the computation offloading for WAs in multi-cloudlet scenario will be studied [49, 50]. For another, the revenue of the edge server will be investigated [51].

Acknowledgements

This work is supported by The Natural Science Foundation of Fujian Province (Grant No.2018J05106), the Education and Scientific Research Projects of Young and Middle-aged Teachers in Fujian Province (JZ160084), and the Scientific Research Foundation of Huaqiao University under Grant No. 14BS316. China Scholarship Council (CSC) awarded Kai Peng's 1-year research abroad at the University of British Columbia.

Authors' contributions

KP, MZ, YZ, LL, JZ, VCML, and LZ conceived and designed the study. MZ and LL performed the simulations. KP wrote the paper. All authors reviewed and edited the manuscript. All authors read and approved the final manuscript.

Availability of data and materials

The details of experimental parameters are given in Section 4.2.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]College of Engineering, Huaqiao University, Quanzhou, Fujian, China. [2]School of Computer Science and Technology, Anhui University, Hefei, China. [3]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210008, China. [4]Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver V6T 1Z4, BC, Canada. [5]Fujian Provincial Academic Engineering Research Centre in Industrial Intellectual Techniques and Systems, Fujian, China.

Received: 20 May 2019 Accepted: 22 July 2019
Published online: 14 August 2019

References

1. T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, Q. Jin, A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing. IEEE Internet Things J. (2018). https://doi.org/10.1109/JIOT.2018.2870288
2. X. Xu, S. Fu, Q. Cai, W. Tian, W. Liu, W. Dou, A. X. Liu, Dynamic resource allocation for load balancing in fog environment. Wirel. Commun. Mob. Comput. **2018**, 15 (2018). https://doi.org/10.1155/2018/6421607
3. S. Raza, S. Wang, M. Ahmed, M. R. Anwar, A survey on vehicular edge computing: architecture, applications, technical issues, and future directions. Wirel. Commun. Mob. Comput. **2019**, 19 (2019). https://doi.org/10.1155/2019/3159762
4. L. Qi, R. Wang, C. Hu, S. Li, Q. He, X. Xu, Time-aware distributed service recommendation with privacy-preservation. Inf. Sci. **480**, 354–364 (2019)
5. K. Peng, V. C. Leung, Q. Huang, Clustering approach based on mini batch Kmeans for intrusion detection system over big data. IEEE Access. **6**, 11897–11906 (2018)
6. Y. Zhang, K. Wang, Q. He, F. Chen, S. Deng, Z. Zheng, Y. Yang, Covering-based web service quality prediction via neighborhood-aware matrix factorization. IEEE Trans. Serv. Comput. (2019). https://doi.org/10.1109/TSC.2019.2891517
7. X. Xu, Q. Liu, Y. Luo, K. Peng, X. Zhang, S. Meng, L. Qi, A computation offloading method over big data for IoT-enabled cloud-edge computing. Futur. Gener. Comput. Syst. **95**, 522–533 (2019)
8. L. T. Yang, X. Wang, X. Chen, J. Han, J. Feng, A tensor computation and optimization model for cyber-physical-social big data. IEEE Trans. Sustain. Comput. (2017). https://doi.org/10.1109/TSUSC.2017.2777503
9. L. Qi, Y. Chen, Y. Yuan, S. Fu, X. Zhang, X. Xu, A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems. World Wide Web (2019). https://doi.org/10.1007/s11280-019-00684-y
10. X. Xu, S. Fu, L. Qi, X. Zhang, Q. Liu, Q. He, S. Li, An IoT-Oriented data placement method with privacy preservation in cloud environment. J. Netw. Comput. Appl. **124**, 148–157 (2018)
11. L. Qi, X. Zhang, W. Dou, Q. Ni, A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data. IEEE J. Sel. Areas Commun. **35**(11), 2616-2624 (2017)
12. X. Wang, L. T. Yang, X. Chen, M. J. Deen, J. Jin, Improved multi-order distributed HOSVD with its incremental computing for smart city services. IEEE Trans. Sustain. Comput. (2018). https://doi.org/10.1109/TSUSC.2018.2881439
13. X. Xu, Y. Xue, L. Qi, Y. Yuan, X. Zhang, T. Umer, S. Wan, An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. Futur. Gener. Comput. Syst. **96**, 89–100 (2019)
14. L. T. Yang, X. Wang, X. Chen, L. Wang, R. Ranjan, X. Chen, M. J. Deen, A multi-order distributed HOSVD with its incremental computing for big services in cyber-physical-social systems. IEEE Trans. Big Data. https://doi.org/10.1109/TBDATA.2018.2824303
15. T. Wang, G. Zhang, M. Z. A. Bhuiyan, A. Liu, W. Jia, M. Xie, A novel trust mechanism based on fog computing in sensor-cloud system. Futur. Gener. Comput. Syst. (2018). https://doi.org/10.1016/j.future.2018.05.049
16. K. Wang, H. Yin, W. Quan, G. Min, Enabling collaborative edge computing for software defined vehicular networks. IEEE Netw. **32**(5), 112–117 (2018)
17. X. Xu, Y. Li, T. Huang, Y. Xue, K. Peng, L. Qi, W. Dou, An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks. J. Netw. Comput. Appl. **133**, 75–85 (2019)
18. T. Wang, J. Zhou, A. Liu, M. Z. A. Bhuiyan, G. Wang, W. Jia, Fog-based computing and storage offloading for data synchronization in IoT. IEEE Internet Things J., 2018. https://doi.org/10.1109/JIOT.2018.2875915
19. T. Wang, J. Zeng, Y. Lai, Y. Cai, H. Tian, Y. Chen, B. Wang, Data collection from WSNs to the cloud based on mobile fog elements. Futur. Gener. Comput. Syst. (2017). https://doi.org/10.1016/j.future.2017.07.031
20. W. Li, K. Liao, Q. He, Y. Xia, Performance-aware cost-effective resource provisioning for future grid IoT-cloud system. J. Energy Eng. (2019). https://doi.org/10.1061/(ASCE)EY.1943-7897.0000611
21. W. Shi, H. Sun, J. Cao, Q. Zhang, W. Liu, Edge computing: An emerging computing model for the internet of everything era. J. Comput. Res. Dev. **54**(5), 907–924 (2017)
22. P. Mach, Z. Becvar, Mobile edge computing: A survey on architecture and computation offloading. IEEE Commun. Surv. Tutor. **19**(3), 1628–1656 (2017)
23. K. Peng, V. Leung, X. Xu, L. Zheng, J. Wang, Q. Huang, A survey on mobile edge computing: Focusing on service adoption and provision. Wirel. Commun. Mob. Comput. **2018**(8267838), 16 (2018). https://doi.org/doi.org/10.1155/2018/8267838
24. X. Xu, D. Li, Z. Dai, S. Li, X. Chen, A Heuristic Offloading Method for Deep Learning Edge Services in 5G Networks. IEEE Access. **7**, 67734–67744 (2019). https://doi.org/10.1109/ACCESS.2019.2918585
25. M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for vm-based cloudlets in mobile computing. IEEE Pervasive Comput. **4**, 14–23 (2009)
26. M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, K. Ha, The role of cloudlets in hostile environments. IEEE Pervasive Comput. **12**(4), 40–49 (2013)
27. M. Satyanarayanan, Z. Chen, K. Ha, W. Hu, W. Richter, P. Pillai, in *6th International Conference on Mobile Computing, Applications and Services*. Cloudlets: at the leading edge of mobile-cloud convergence (IEEE, Austin, 2014), pp. 1-9
28. M. Jia, J. Cao, L. Yang, *Heuristic Offloading of Concurrent Tasks for Computation-Intensive Application in Mobile Cloud Computing. 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. (IEEE, Toronto, 2014), pp. 352–357

29. H. Wu, W. Knottenbelt, K. Wolter, Y. Sun, *An Optimal Offloading Partitioning Algorithm in Mobile Cloud Computing. Evaluation of Systems*. (Springer, Cham, 2016), pp. 311–328
30. H. Hu, R. Liu, Hu H., Multi-objective optimization for task scheduling in mobile cloud computing. J. Comput. Res. Dev. **54**(9), 1909–1919 (2017)
31. S. Deng, L. Huang, J. Taheri, A. Y. Zomaya, Computation offloading for service workflow in mobile cloud computing. IEEE Trans. Parallel Distrib. Syst. **26**(12), 3317–3329 (2014)
32. X. Xu, W. Dou, X. Zhang, J. Chen, EnReal: An energy-aware resource allocation method for scientific workflow executions in cloud environment. IEEE Trans. Cloud Comput. **4**(2), 166–179 (2015)
33. B. Lin, W. Guo, G. Chen, Scheduling strategy for science workflow with deadline constraint on multi-cloud. J. Commun. **39**(1), 56–69 (2018)
34. M. Jia, W. Liang, *Delay-Sensitive Multiplayer Augmented Reality Game Planning in Mobile Edge Computing. Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. (ACM, Montreal, 2018), pp. 147–154
35. B. Li, M. He, W. Wu, A. K. Sangaiah, G. Jeon, Computation offloading algorithm for arbitrarily divisible applications in mobile edge computing environments: An OCR case. Sustainability. **10**(17), 196–210 (2018)
36. L. Liu, Z. Chang, X. Guo, T. Ristaniemi, *Multi-objective optimization for computation offloading in mobile-edge computing. 2017 IEEE Symposium on Computers and Communications (ISCC)*. (IEEE, Heraklion, 2017), pp. 832–837
37. X. Li, L. Qian, R. Ruiz, Cloud Workflow scheduling with deadlines and time slot availability. IEEE Trans. Serv. Comput. **11**(2), 329–340 (2018)
38. Z. Li, J. Ge, H. Hu, W. Song, H. Hu, B. Luo, Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds. IEEE Trans. Serv. Comput. **11**(4), 713–726 (2018)
39. J. Vilaplana, F. Solsona, I. Teixido, J. Mateo, F. Abella, J. Rius, A queuing theory model for cloud computing. J. Supercomput. **69**(1), 492–507 (2014)
40. K. Deb, A. Pratap, S. Agarwal, T. A. M. T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
41. Y. Sun, F. Lin, H. Xu, Multi-objective optimization of resource scheduling in Fog computing using an improved NSGA-II. Wirel. Pers. Commun. **102**(2), 1369–1385 (2018)
42. X. Xu, S. Fu, Y. Yuan, Y. Luo, L. Qi, W. Lin, W. Dou, Multi-objective computation offloading for workflow management in cloudlet-based mobile cloud using NSGA-II.2018 (2018). https://doi.org/10.1111/coin.12197
43. S. Wang, Y. Zhao, J. Xu, J. Yuan, C. Hsu, Edge server placement in mobile edge computing. J. Parallel Distrib. Comput. **127**, 160–168 (2019)
44. X. Xu, R. Huang, R. Dou, Y. Li, J. Zhang, T. Huang, W. Yu, Energy-Efficient Cloudlet Management for Privacy Preservation in Wireless Metropolitan Area Networks. Secur. Commun. Netw. **2018**, 13 (2018). https://doi.org/10.1155/2018/8180451
45. Z. Pang, L. Sun, Z. Wang, E. Tian, S. Yang, *A survey of cloudlet based mobile computing. 2015 International Conference on Cloud Computing and Big Data (CCBD)*. (IEEE, Shanghai, 2015), pp. 268–275
46. X. Xu, Y. Li, Y. Yuan, K. Peng, W. Yu, W. Dou, A. X. Liu, *An Energy-Aware Virtual Machine Scheduling Method for Cloudlets in Wireless Metropolitan Area Networks. 2018 IEEE Cyber, Physical and Social Computing (CPSCom)*. (IEEE, Halifax, 2018), pp. 517–523
47. J. Zhang, Z. Zhou, S. Li, L. Gan, X. Zhang, L. Qi, W. Dou, Hybrid computation offloading for smart home automation in mobile cloud computing. Pers. Ubiquit. Comput. **22**(1), 121–134 (2018)
48. T. Huang, F. Ruan, S. Xue, L. Qi, Y. Duan, Computation offloading for multimedia workflows with deadline constraints in cloudlet-based mobile cloud. Wirel. Netw. **2019**, 1–15 (2019)
49. D. G. Roy, D. De, A. Mukherjee, R. Buyya, Application-aware cloudlet selection for computation offloading in multi-cloudlet environment. J. Supercomput. **73**(4), 1672–1690 (2017)
50. A. Mukherjee, D. De, D. G. Roy, A power and latency aware cloudlet selection strategy for multi-cloudlet environment. IEEE Trans. Cloud Comput. **7**(1), 141–154 (2016)
51. X. Xu, Q. Cai, G. Zhang, J. Zhang, W. Tian, X. Zhang, A. X. Liu, An incentive mechanism for crowdsourcing markets with social welfare maximization in cloud-edge computing, Concurrency and Computation: Practice and Experience, e4961 (2018). https://doi.org/10.1002/cpe.4961

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.