A Multi-objective Computation Offloading Method in Multi-cloudlet Environment

Kai Peng[1,2(✉)], Shuaiqi Zhu[1], Lixin Zheng[2], Xiaolong Xu[3], and Victor C. M. Leung[4]

[1] College of Engineering, Huaqiao University, Quanzhou, China
pkbupt@gmail.com
[2] Fujian Provincial Academic Engineering Research Centre in Industrial Intellectual Techniques and Systems, Quanzhou, China
[3] Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, Nanjing University of Science and Technology, Nanjing 210094, People's Republic of China
[4] Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada

Abstract. Computation offloading is becoming a promising technology that can improve quality of service for mobile users in mobile edge computing. However, it becomes much difficult when there are multi-cloudlet near to the mobile users. The resources of the cloudlet are heterogeneous and finite, and thus it is challenge to choose the best cloudlet for the multi-user. The issue for multi-user in multi-cloudlet environment is well-investigated in this study. Firstly, we establish a multi-objective optimization model with respect to time consumption and energy consumption of mobile devices. Moreover, we devise a multi-objective computation offloading method based on improved fast and elitist genetic algorithm for selecting the optimal offloading strategies. Finally, compared with other methods, numerous experiments proved that our proposed method have advantages in effectiveness and efficiency.

Keywords: Mobile edge computing · Multi-cloudlet · Energy consumption · Time consumption

1 Introduction

As the computer network are gaining colossal popularity, Mobile devices (MDs) are becoming a key role affecting people's lives [1–3]. Nevertheless, differ from the traditional devices, e.g. desktop computers, MDs are limited by the aspects with respect to the storage capacity, operational performance, especially for battery life. Mobile cloud computing (MCC) can provide mobile users with a variety of extended services. However, an envisioned disadvantage of MCC is

X. Zhang et al. (Eds.): CloudComp 2019/SmartGift 2019, LNICST 322, pp. 20–32, 2020.
https://doi.org/10.1007/978-3-030-48513-9_2

that the centralized cloud is distantly for mobile users to exchange data, which is likely to incur high round-trip delay. A resounding paradigm that emerges recently, namely mobile edge computing (MEC), has been considered the optimal paradigm to solve this issue [4–11]. A cloudlet which is in the vicinity to the mobile users provisions low-latency services for the interactive response. Hence, if the users choose to offload the applications to cloudlet, the time and energy consumption will be decreased.

Suppose the computing resources of cloud in MCC are infinite. In contrast, the computing resources in MEC are finite. A queue latency will arise if a crowd of mobile users are concurrently requesting services from a single cloudlet. In addition, resources in different cloudlets are heterogeneous. Moreover, the difficulty of computing offloading is further increased under the scenarios that there are multi-cloudlet nearby. More specifically, how to choose the best cloudlet for the applications for the multi-user is becoming very important. Additionally, the limited of resources in a single cloudlet needs to be considered. With the increasing of the number of application, the cloudlets may become overload. For executing the applications successfully, the applications need to be taken into consideration to be run locally or offloaded to cloud. In another word, it is essential to balance these three locations, namely, local, multi-cloudlet, as well as cloud.

The issues that computation in single cloudlet have been well investigated [12–14]. Moreover, the computation offloading which is focused on the selection of cloudlet and the allocation of resource in multi-cloudlet MEC has also been well studied in [15–21]. Differ from them, in this study, the energy consumption and time consumption optimization for applications in multi-cloudlet in MEC are taken in to consideration jointly. The main contributions of this study can be concluded as three aspects.

1) We investigated the computation offloading over multi-cloudlet environment in MEC in this study. Both the energy consumption and time consumption are seen as the optimization goals.
2) A multi-objective computation offloading method for multiple applications based on improved Fast and Elitist Multiobjective Genetic Algorithm [22] is proposed.
3) We conduct comprehensive evaluations and analysis to demonstrate the proposed method can provide effective offloading strategies.

The remaining of the paper is arranged as follows. Section 2 firstly introduce the overview of system model and present problem formulation. And then Sect. 3 describe a multi-objective computation offloading algorithm over multi-cloudlet. In Sect. 4, we indicate the experimental evaluation. And Sect. 5 conclude the related work. Section 6, we summarize the paper and describe the future work in the final section.

2 System Model and Problem Formulation

In this section, the network architecture with respect to MEC system is firstly established. After that we introduce the system model and problem formulation with two objectives in terms of decreasing the time consumption and the energy consumption in MEC.

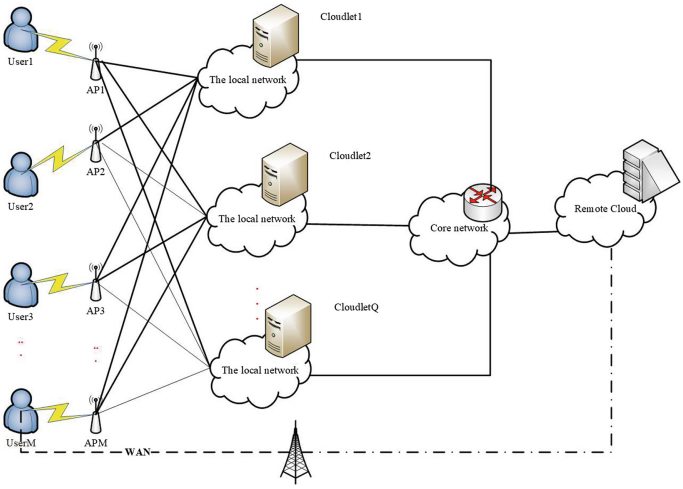

Fig. 1. Multi-cloudlet MEC system

The multi-cloudlet MEC system is shown in Fig. 1. Cloud has infinite resources. Mobile user could be mobile phone or other devices. Every user has a certain number of applications which need to be processed. These applications can be run locally, and also can be offloaded to the remote cloud over Wide Area Network (WAN) or to cloudlets over Local Area Network (LAN). In addition, cloudlet can communicate with cloud directly.

R is a collection of mobile users, which can be modeled as $R = \{r_{1,1}, ..., r_{1,n}, r_{2,n}, ..., r_{i,n}\}$, representing the first application of the first user, the second application of the first user, ..., the n-th application of the o-th user. There are multiple virtual machines in cloudlet which can be used to process multiple applications concurrently. The resource can be expressed as $CL = (L_{LAN}, f_{cl}, U)$, where L_{LAN} is the transmission latency between the cloudlet and MD, f_{cl} denotes the ability of the cloudlet to process the application, and U indicates the total amount of VMs.

$r_{i,n} = (l_{i,n}, o_{i,n})$ is defined as a 2-tuple, where the average instruction length of the application requested by the user is denoted as $l_{i,n}$, and $o_{i,n}$ is the offloading strategy assigned to the f-th request of the i-th user, $o_{i,n} = 0$ indicates that the application request is run locally, $\{o_{i,n} = 1, 2, ..., Q\}$ represent that the application is offloaded to cloudlet for processing, $o_{i,n} = Q + 1$ indicates that the application is offloaded to cloud.

2.1 Time Consumption Model

There are three kinds of time to consider when establishing the time consumption model, e.g., waiting time, processing time and transmission time.

Average Waiting Time. The requests of users' are uncertain, and the resources of cloudlet are limited. That means there is a high possibility that user requests will be queued in the cloudlet. Thus, we need to establish a mathematical model to effectively evaluate this queuing situation. According to the queuing theory [23], we can establish the waiting time mathematical model.

Assuming that the interval of arrival of the application obeys the parameter λ of the negative exponential distribution. Likewise, we assume that the cloudlet has fixed service rate μ. The possibility of the idle state in cloudlet is calculated as

$$P_0 = [\sum_{u=0}^{U-1} \frac{\varphi^U}{U!(1-\varphi_U)}]^{-1} \tag{1}$$

where $\varphi = \frac{\lambda}{\mu}$ shows the intensity of the service. Moreover, the average intensity of the service for U virtual machines in cloudlet is denoted as $\varphi_U = \frac{\varphi}{U}$.

Let p_n be the probability of the cloudlet will reach when it is stable. The probability of waiting for an application is calculated by

$$C_w(U, \varphi) = \sum_{u=U}^{\infty} p_n = \frac{\varphi^U}{U!(1-\varphi_U)} \cdot p_0 \tag{2}$$

The average waiting time of the application can be calculated in the follow

$$W_q = \frac{1}{U \cdot (\mu - \lambda)} \cdot \sum_{u=U}^{\infty} \frac{\varphi^U}{U!(1-\varphi_U)} \cdot p_0 \tag{3}$$

Processing Time and Transmission Time. The processing time of the n-th application from i-th user is given as

$$T_{exe}(r_{i,n}) = \begin{cases} \frac{l_{i,n}}{f_l} & o_{i,n} = 0 \\ W_q + \frac{l_{i,n}}{f_{cl}} + L_{LAN} & o_{i,n} = 1 \ or \ 2, or \ldots or \ Q \\ \frac{l_{i,n}}{f_c} + L_{WAN} & o_{i,f} = Q + 1 \end{cases} \tag{4}$$

The application is processed locally when the offloading strategy is 0. The total time consumption is the sum of the processing time, the average waiting time, as well as the transmission time in LAN when the application is offloaded to the cloudlet. The total time consumption includes the transmission time and the processing time in WAN when the application is processed on the cloud.

The transmission time of the $n-th$ application of the i-th user is given as follows

$$T_{trans}(r_{i,n}) = \frac{l_{i,n}}{B} \tag{5}$$

where

$$B = \begin{cases} \infty & o_{i,n} = 0 \\ B_{cl} & o_{i,n} = 1\, or\, 2, or \ldots or\, Q \\ B_c & o_{i,n} = Q + 1 \end{cases} \tag{6}$$

In summary, the total time required for the n-th application request execution of the i-th user is

$$T(r_{i,n}) = T_{exe}(r_{i,n}) + T_{trans}(r_{i,n}) \tag{7}$$

2.2 Energy Consumption Model

The transmission energy consumption and processing energy consumption make up the energy consumption of MDs. And the energy consumption for executing the $n-th$ request of $i-th$ user is expressed as $E_{exe}(r_{i,n})$, which is calculated as

$$E_{exe}(r_{i,n}) = \begin{cases} \frac{l_{i,n}}{f_l} \times \delta_A & o_{i,n} = 0 \\ (W_q + \frac{l_{i,n}}{f_{cl}} + L_{LAN}) \times \delta_I & o_{i,n} = 1,\ 2, \ldots\ or\ Q \\ (\frac{l_{i,n}}{f_c} + L_{WAN}) \times \delta_I & o_{i,n} = Q + 1 \end{cases} \tag{8}$$

where δ_I and δ_A represent energy consumption of a MD in idle time and active state. The transmission energy consumption is shown as

$$E_{trans}(r_{i,n}) = \frac{l_{i,n}}{B} \cdot \delta_T \tag{9}$$

where δ_T is the energy consumption of the MD during transmission. In summary, it can be concluded that the power consumption required for the n-th application request execution of the i-th user is

$$E(r_{i,n}) = E_{exe}(r_{i,n}) + E_{trans}(r_{i,n}) \tag{10}$$

2.3 Problem Formulation

The objectives of the computational offloading are to achieve the goal of decreasing the energy consumption and time consumption. The multi-objective problem formulation is shown as follows.

$$Min \sum_{i=1}^{I} \sum_{n=1}^{N} T(r_{i,n}); \forall o \in \{1, 2, 3, \ldots, N\} \tag{11}$$

$$Min \sum_{i=1}^{I} \sum_{n=1}^{N} E(r_{i,n}); \forall o \in \{1, 2, 3, \ldots, N\} \tag{12}$$

$$s.t. o_{i,n} \in \{0, 1, 2, ..., Q + 1\} \tag{13}$$

3 Multi-objective Computation Offloading Method in Multi-cloudlet

In this section, we present the details of our proposed method: computation offloading method in multi-cloudlet (COMC). In addition, pseudocode of this algorithm are described.

3.1 Method Review

In this study, our objectives are optimizing the consumption of time and energy for all applications. We model the computational migration problem as a multi-objective problem and use the improved NSGA-II to get the best strategies. The details are shown in the following subsection.

3.2 Encoding

Each application is numbered according to this form $\{1, 2, \ldots, n\}$. The gene represents the offloading strategy for each application. Multi-gene forms a full chromosome, and it can be regards as an individual, which represents a solution to our issue.

An example of encoding is shown in Fig. 2. We use integer coding for our issue. More specifically, we have encoded each offloading strategy as $\{0\ 1,\ 2, \ldots,\ n\}$. Each value in the strategy means the corresponding destination the application offload. For example, number 0 means this application is processed locally, similarly, number 1 means this application is offloaded to the cloudlet1 and number 2 means that this application is executed on the cloudlet2, and so on. Notice that number $\{Q+1\}$ means the application is run on the cloud. Fitness function is used to evaluate individual quality, which is obtained by Eq. (11) and (12). These two fitness functions (11) and (12) indicate the consumption of time and energy. Our goal is finding the best offloading solution which makes these two functions well.

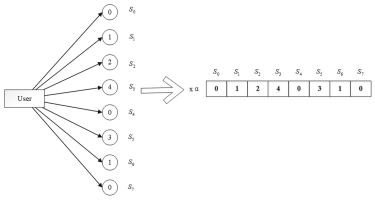

Fig. 2. Encoding.

3.3 Initialization

Randomly generate initialized population P_0, then perform non-dominant sorting on P_0. Moreover, binary tournament selection, crossover, as well as mutation is executed on P_0 to form new population Q_0.

3.4 Selection

To form new group population $R_t = P_t \cup Q_t$, where $t = 0$. In addition, based on non-dominated sort of R_t, we have gotten the non-dominated front-end $\{F_1, F_2, \ldots, F_i\}$ [22] in this step.

3.5 Crossover

In this step, the crossover combines two parental chromosomes and parts of gene fragment between them are exchanged to get better spring chromosomes Q. Some excellent genes are reserved. The fitness is improved by using the similar gene fragments of these chromosomes.

3.6 Mutation

In this step, the mutation slightly modifies certain genes on chromosomes to produce the individuals who are well-fitness while avoiding early convergence. Therefore, an improved mutation is proposed. For example, if cloudlet1 is the best offloading location, the probability of mutating to $o_{i,n} = 1$ should be increased.

3.7 Calculation of Crowding Distance

Based on the crowding distance comparison operation $\prec_n$, we sort all F_i to choose the best N individuals forming a population P_{t+1}. The crowding distance is formulated as:

$$\begin{aligned} i_d = i_d^T + i_d^E = & \sum_{n=1}^{N} (|T_{a,n}^{i+1}(S) - T_{a,n}^{i-1}(S)|) \\ & + \sum_{n=1}^{N} (|E_{a,n}^{i+1}(S) - E_{a,n}^{i-1}(S)|) \end{aligned} \tag{14}$$

where $1 <= i <= n$ and i_d indicates the crowding distance for the $i-th$ offloading strategy $o_{i,n}$. i_d^T, i_d^E indicate the objective functions (11) and (12), respectively. $T_{a,n}^{i+1}(S)$ is the value for offloading strategy $o_{i+1,n}$ to the objective function $T_{a,n}(S)$. Likewise, $E_{a,n}^{i+1}(S)$ indicates offloading strategy value $o_{i+1,n}$ to the objective function $E_{a,n}(S)$.

3.8 The Algorithm Pseudocode

Based on the above steps, the pseudocode of our method is shown as follows.

Algorithm 1. COMC

Input: Applications, the size of population N, the maximum iterations number G_{max}
Output: Offloading solution S, time consumption as well as energy consumption

$G_{cur} = 1$,$\ell = 1$
while $G_{cur} \leq G_{max}$ **do**
  Initialize P_{cur}
  Q_ℓ =selection, crossover and mutation $P_\ell^{'}$
  $R_\ell = P_\ell^{'} + Q_\ell$
  $F = fastnondominatesort(R_\ell)$
  $P_{\ell+1} = \emptyset$
  $s = 0$
  while $len(P_{\ell+1}) + len(F[s]) < N$ **do**
    Using formula (14) to acquire crowding distance ($F[s]$)
    $P_{\ell+1} += F[s]$
    $s = s + 1$
    $P_{\ell+1} += F[s][0 : N - len(P_{\ell+1})]$
    $Q_{\ell+1}$=form new generation($P_{\ell+1}$)
    $\ell = \ell + 1$
    $G_{cur} = G_{cur} + 1$
  end while
end while
return S,energy consumption,time consumption

4 Experimental Evaluation

In this section, we present the experimental evaluation of our method. Firstly, the experimental setting is introduced, which includes comparative methods and experimental parameters. Then, the experimental result and discussion is described.

4.1 Experimental Setting

Some other comparative computation offloading methods are proposed and shown as follows.

No offloading (NO): All applications are executed in MDs where no transmission overhead is considered. And it is named as NO.

Full offloading to cloud (FOC): All applications in MDs are offloaded to cloud for processing, named as FOC.

Random full offloading to cloudlet (RFOCL): All applications are offloaded to the multi-cloudlet for processing randomly, named as RFOCL.

The value of parameters are presented in Table 1.

We implement these methods by JAVA and Eclipse over Win10 64 OS with 2 Intel Core i7-7700U 2.80 GHz processors and 16 GB RAM.

Table 1. Experimental parameter.

Parameter description	Value
Power of MDs when CPU is in idle state	0.83 W
Power of MDs when CPU is in active state	0.1 W
Power of MDs when transmitting tasks	0.83 W
Processing capacity of MDs	500 MHZ
Processing capacities of the {cloudlet1-cloudlet3, cloud}	2300 MHZ, 2400 MHZ, 2100 MHZ, 2600 MHZ
The latency of LAN and WAN	0.025 ms, 0.1 ms
The bandwidth of LAN	220/240/230 kb/s
The bandwidth of WAN	110 kb/s
Average waiting time of applications in cloudlet	0.07 s
Crossover probability	0.9
Number of VMs per cloudlet	3

4.2 Experimental Result and Discussion

Different results are received with the different number of applications. We have done 50 experiments under convergence for different user scale and application scale.

Firstly, we describe how COMC makes equilibrium between these two targets, i.e. time consumption and energy consumption of MDs. Figure 3 shows the result of comparison in energy consumption for different number of applications using NO, FOC, RFOCL and COMC. Additionally, the comparison result of the average time consumption with different methods is shown in Fig. 4. We can see that COMC is effective while the number of applications is increasing. Overall, COMC is effective as it makes these two objectives better, not just one best.

Then, we analysis how COMC provides effective offloading strategies to balance multi offloading destinations, i.e. local, cloudlet1, cloudlet2, cloudlet3 and cloud. As shown in Tables 2, 3, 4 and 5, the applications are mainly offloaded to the cloudlet1, cloudlet2, and cloudlet3. That means the cloudlets are the most optimal offloading destinations. When the number of applications increases beyond the processing capacity of cloudlets, parts of applications is migrated into the cloud to avoid the queue latency.

Considering the finite of resource in cloudlet and transmission latency for applications between MDs and cloud. As the applications are increased, the number of them that are executed locally increases at the same time, while the number of applications that are executed on cloud is decreased, that also proves our method can balance every offloading dentation. Above all, our propose method can obtain the best strategy to minimize energy consumption and time consumption for all applications.

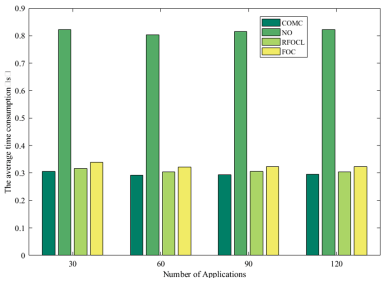

Fig. 3. Comparison of the average time consumption with NO, RFOCL, FOC and COMC

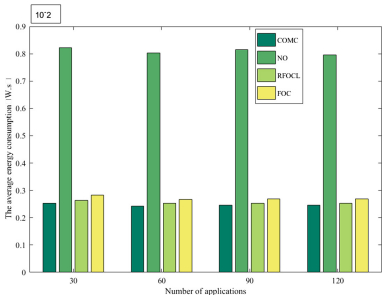

Fig. 4. Comparison of the average energy consumption with NO, RFOCL, FOC and COMC

Table 2. Applications = 30

Location	NO	RFOCL	FOC	COMC
Local	30	0	0	0
Cloudlet1	0	8	0	9
Cloudlet2	0	12	0	10
Cloudlet3	0	10	0	5
Cloud	0	0	30	3

Table 3. Applications = 60

Location	NO	RFOCL	FOC	COMC
Local	60	0	0	7
Cloudlet1	0	18	0	17
Cloudlet2	0	22	0	24
Cloudlet3	0	20	0	7
Cloud	0	0	0	60

Table 4. Applications = 90

Location	NO	RFOCL	FOC	COMC
Local	90	0	0	0
Cloudlet1	0	31	0	24
Cloudlet2	0	30	0	27
Cloudlet3	0	29	0	22
Cloud	0	0	90	4

Table 5. Applications = 120

Location	NO	RFOCL	FOC	COMC
Local	120	0	0	0
Cloudlet1	0	39	0	30
Cloudlet2	0	49	0	33
Cloudlet3	0	32	0	24
Cloud	0	0	120	10

5 Related Work

Computation offloading in single cloudlet scenario has been well investigated in [12–14]. Xu et al. [12] proposed a new method namely MCO to address the challenge that offloading resolvent for workflow applications. Zhang et al. [13] proposed a computation offloading algorithm based on particle swarm optimization to optimize the energy consumption of MDs while meeting time constraints of application. Liu et al. [14] leveraging of Interior Point Method and secularization method to solve the multi-objective problem with respect to average price cost, average execution time and average energy consumption in MEC system.

These methods have provided effective solutions for the computation offloading issue in single cloudlet environment, however they may not used in multi-cloudlet environment directly.

Computation offloading in multi-cloudlet has been studied in [15–21]. Li et al. [15] proposed theoretical model to partition the arbitrarily divisible application in MEC and derive the closed-form expressions to minimize the completion time for application. [16] firstly proposed a two-stage strategy to allocate the resource, and then proposed a mixed integer linear programming model is constructed to optimize computing resources allocation and the cloudlet selection.

An application-aware cloudlet selection method is proposed by Roy et al. [17] with respect to the scenario that multi-cloudlet, which can reduce the execution latency and energy consumption of MDs as well as the load balance of the system. Suppose different types of applications could be processed in different cloudlets, and the coming application type was first checked. Finally, the most suitable cloudlet is chosen according to the verified result. A power and latency aware cloudlet selection method is proposed by Mukherjee et al. [18]. In [19], Dilay Parmar et al. proposed a mechanism included two phases for identifying a cloudlet for computation offloading. Firstly, if the cloudlets that are covered in the WiFi range of the MD did not connect to any cloudlets, they would be identified. And then the second phase, it will accomplish the selection of the ideal offloading cloudlet.

In [20], the objective is to minimize the total energy consumption and they proposed a method to decide which tasks should be offloaded and determine where to offloaded. Mazouzi et al. [21] proposed the optimization problem of joint cloudlet selection and minimizing waiting time in fog environment. The problem is modeled as many-to-one matching game based on game theory and a corresponding algorithm is used to solve this game.

Different from the existing studies, the time consumption and energy consumption of applications in multi-cloudlet environment have been well studied in this study. More specifically, these cloudlets are heterogeneous in the problem we studied. Meanwhile, we will make full use of the resources in local, multi-cloudlet, as well as cloud to provide better services for users.

6 Conclusion

In this paper, we have studied the multi-objective computation offloading problem in multi-cloudlet environment. Aiming at resolving this problem, we proposed a multi-objective method to find the optimal processed destination for all the applications by minimizing time consumption and energy consumption at the same time. Extensive experimental evaluations have shown that our proposed method is efficiency and effectiveness.

In future work, we will explore cloudlet placement for hybrid applications in multi-cloudlet MEC.

Acknowledgment. The authors thank for the Natural Science Foundation of Fujian Province (Grant No. 2018J05106), the National Science Foundation of China (Grant No. 61902133), 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. The Fundamental Research Funds for the Central Universities (No. 30918014108).

References

1. Qi, L., et al.: Finding all you need: web APIs recommendation in web of things through keywords search. IEEE Trans. Comput. Soc. Syst. **6**, 1063–1072 (2019)
2. Zhang, Y., et al.: Covering-based web service quality prediction via neighborhood-aware matrix factorization. IEEE Trans. Serv. Comput. (2019). https://doi.org/10.1109/TSC.2019.2891517
3. Xu, X., Liu, Q., Zhang, X., Zhang, J., Qi, L., Dou, W.: A blockchain-powered crowdsourcing method with privacy preservation in mobile environment. IEEE Trans. Comput. Soc. Syst. (2019). https://doi.org/10.1109/TCSS.2019.2909137
4. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
5. Qi, L., Chen, Y., Yuan, Y., Fu, S., Zhang, X., Xu, X.: A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems. World Wide Web **23**(2), 1275–1297 (2019). https://doi.org/10.1007/s11280-019-00684-y
6. Zhang, Y.W., Zhou, Y.Y., Wang, F.T., Sun, Z., He, Q.: Service recommendation based on quotient space granularity analysis and covering algorithm on Spark. Knowl.-Based Syst. **147**, 25–35 (2018)
7. Raza, S., Wang, S., Ahmed, M., Anwar, M.R.: A survey on vehicular edge computing: architecture, applications, technical issues, and future directions. Wirel. Commun. Mob. Comput. **2019**, 1–19 (2019)
8. Wang, K., Yin, H., Quan, W., Min, G.: Enabling collaborative edge computing for software defined vehicular networks. IEEE Netw. **32**(5), 112–117 (2018)
9. Xu, X., et al.: A computation offloading method over big data for IoT-enabled cloud-edge computing. Future Gener. Comput. Syst. **95**, 522–533 (2019)
10. Xu, X., et al.: An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks. J. Netw. Comput. Appl. **133**, 75–85 (2019)
11. Peng, K., Leung, V., Xu, X., Zheng, L., Wang, J., Huang, Q.: A survey on mobile edge computing: focusing on service adoption and provision. Wirel. Commun. Mob. Comput. **2018** (2018). Article no. 8267838, 16 pages. https://doi.org/10.1155/2018/8267838
12. Xu, X., et al.: Multi-objective computation offloading for workflow management in cloudlet-based mobile cloud using NSGA-II (2018). https://doi.org/10.1111/coin.12197
13. Zhang, J., et al.: Hybrid computation offloading for smart home automation in mobile cloud computing. Pers. Ubiquit. Comput. **22**(1), 121–134 (2018)
14. Liu, L., Chang, Z., Guo, X., Ristaniemi, T.: Multi-objective optimization for computation offloading in mobile-edge computing. In: 2017 IEEE Symposium on Computers and Communications (ISCC), pp. 832–837. IEEE (2017)

15. Li, B., He, M., Wu, W., Sangaiah, A.K., Jeon, G.: Computation offloading algorithm for arbitrarily divisible applications in mobile edge computing environments: an OCR case. Sustainability **10**(17), 196–210 (2018)
16. Liu, L., Fan, Q.: Resource allocation optimization based on mixed integer linear programming in the multi-cloudlet environment. IEEE Access **6**, 24533–24542 (2018)
17. Roy, D.G., De, D., Mukherjee, A., Buyya, R.: Application-aware cloudlet selection for computation offloading in multi-cloudlet environment. J. Supercomput. **73**(4), 1672–1690 (2016). https://doi.org/10.1007/s11227-016-1872-y
18. Mukherjee, A., De, D., Roy, D.G.: A power and latency aware cloudlet selection strategy for multi-cloudlet environment. IEEE Trans. Cloud Comput. **7**, 141–154 (2016)
19. Parmar, D., Kumar, A.S., Nivangune, A., Joshi, P., Rao, U.P.: Discovery and selection mechanism of cloudlets in a decentralized MCC environment. In: Proceedings of the International Conference on Mobile Software Engineering and Systems, pp. 15–16. ACM (2016)
20. Ali, M., Riaz, N., Ashraf, M.I., Qaisar, S., Naeem, M.: Joint cloudlet selection and latency minimization in fog networks. IEEE Trans. Industr. Inf. **14**(9), 4055–4063 (2018)
21. Mazouzi, H., Boussetta, K., Achir, N.: Maximizing mobiles energy saving through tasks optimal offloading placement in two-tier cloud: a theoretical and an experimental study. Comput. Commun. **144**, 132–148 (2019)
22. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.A.M.T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
23. Vilaplana, J., Solsona, F., Teixidó, I., Mateo, J., Abella, F., Rius, J.: A queuing theory model for cloud computing. J. Supercomput. **69**(1), 492–507 (2014). https://doi.org/10.1007/s11227-014-1177-y