Hindawi
Security and Communication Networks
Volume 2018, Article ID 1937670, 12 pages
https://doi.org/10.1155/2018/1937670

Research Article

Enhanced Adaptive Cloudlet Placement Approach for Mobile Application on Spark

Yiwen Zhang,[1] Kaibin Wang,[1] Yuanyuan Zhou,[1] and Qiang He[2]

[1]*Key Laboratory of Intelligent Computing & Signal Processing, Ministry of Education, Anhui University, Hefei, Anhui 230031, China*
[2]*School of Software and Electrical Engineering, Swinburne University of Technology, Melbourne, Australia*

Correspondence should be addressed to Yuanyuan Zhou; yyzhouahu@qq.com

Received 6 June 2018; Accepted 9 August 2018; Published 2 September 2018

Academic Editor: Yuan Yuan

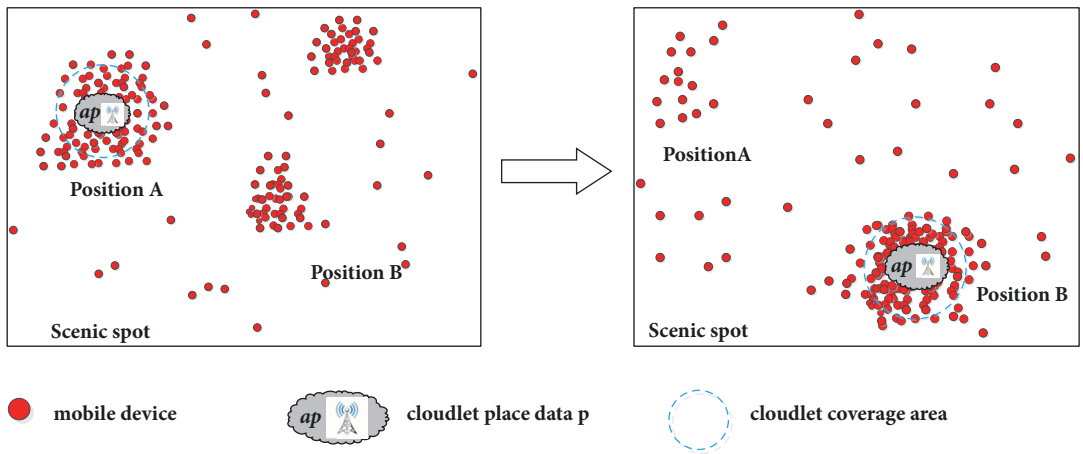
The applications of mobile devices are increasingly becoming computationally intensive while the computing capability of the user's mobile device is limited. Traditional approaches offload the tasks of mobile applications to the remote cloud. However, the rapid growth of mobile devices has made it a challenge for the remote cloud to provide computing and storage capacities with low communication delays due to the fact that the remote cloud is geographically far away from mobile devices. Reducing the completion time of applications in mobile devices through the technical expending mobile cloudlets which are moving collocated with Access Points (APs) is necessary. To address the above issues, this paper proposes EACP-CA (**E**nhanced **A**daptive **C**loudlets **P**lacement approach based on **C**overing **A**lgorithm), an enhanced adaptive cloudlet placement approach for mobile applications in a given network area. We apply the CA (Covering Algorithm) to adaptively cluster the mobile devices based on their geographical locations, the aggregation regions of the mobile devices are identified, and the cloudlet destination locations are also confirmed according to the clustering centers. In addition, we can also obtain the traces between the original and destination locations of these mobile cloudlets. To increase the efficiency, we parallelize CA on Spark. Extensive experiments show that the proposed approach outperforms the existing approach in both effectiveness and efficiency.

1. Introduction

The rapid development of the mobile Internet and the Internet of Things has promoted the emergence of various new types of services, which has led to explosive growth in mobile communication traffic over the past few years. Mobile devices have gradually replaced personal computers as one of the main tool, which people can use in daily work, socialization, and entertainment. Simultaneously, the mobile applications are also increasingly becoming computationally intensive [1–3]. These vast amounts of mobile applications accordingly bring huge computing capacity, storage capacity requirements, and low latency requirements. The previous methods allow these mobile devices to directly access the remote cloud (e.g., Amazon) [4–6], deploying all services to the remote cloud which will not only lead to the great increase of the network load and cause a long delay in the network, but also put higher demands on bandwidth and delay performance of network. The remote cloud is far away from its users and the network delay incurred by processing the requirements can be very costly. These above situations are especially intolerable in real-time demand of applications where the rapid response time is vital for users who take the mobile devices.

In order to cope with the long latency problem, recent works have proposed the cloudlets with strong computing and storage capacity, which are typically collocated at the AP in a network and can be accessed by users via wireless connection [3, 7, 8]. The critical advantage of the cloudlets is that the cloudlets can be deployed physical proximity close to users which can shorten the transmission latency and improve the user experience of using interactive applications.

Although there is an increasing number of researches in mobile cloudlet offloading technology [9–11], fewer researches about how to place cloudlets in a given network region are conducted [3, 7, 12–14]. It is necessary to pay



(a) The original mobile devices distribution and the cloudlet placement position

(b) The new generated cloudlet placement position after the movement of the mobile devices

Figure 1: Motivation example of adaptive cloudlet placement.

attention to how cloudlets should be placed in a given network since the suitable cloudlets placements contribute to enhancing the cloud service for dynamic context-aware mobile applications. The price of the cloudlets is so expensive that the number of the cloudlets is limited, so we should maximize the utilization of cloudlets by a suitable cloudlets placement. Some current researches pay attention to the cloudlets placement to alleviate the low utilities of cloudlets [3, 12–14], the cloudlets placement in their studies are fixed, but, in practice, users are constantly moving. Fixed-placed cloudlets cannot maintain efficiency in responding to the application requirements for the constant mobile devices. In order to explain this problem more vividly, we use Figure 1 to illustrate the issue. Figure 1 presents an example with many mobile devices and a cloudlet. In Figure 1(a), those mobile devices with GPS location are randomly distributed in the scenic spot and a cloudlet with AP being in position A. Those mobile devices are constantly moving over time. And at time t', the crowd moves close to position B. If the cloudlet is still deployed at position A, the coverage range of cloudlet is so small which leads to the low utility of the cloudlet. But if the cloudlet moves to position B described in Figure 1(b), the scope of cloud coverage has increased significantly. Therefore, to improve the performance of the cloudlet service in the dynamic scenario, it is significant to propose an enhanced adaptive mobile cloudlets placement according to practical distributions of mobile devices.

To address the above issue, this paper proposes EACP-CA (**E**nhanced **A**daptive **C**loudlets **P**lacement approach based on **C**overing **A**lgorithm), a novel approach for cloudlets placement. Given a known mobile devices activity area, according to the characteristics of the data and being independent of the initial centers [8, 15], EACP-CA first employs CA with "blind" features to adaptively partition mobile devices into clusters based on the physical distance between their positions. Then we adjust the central positions obtained by CA through the obtained graph path. Next, we execute *confirmation of the center positions operation* and get the desired cloudlets central positions according to the distribution of the mobile devices. We also need to obtain the pair of the original mobile cloudlet positions and the destination position of them. Finally, we conduct the enhanced adaptive cloudlets placement approach to finish the mobile cloudlets placement. To improve the efficiency, we parallelize CA on Spark.

The major contributions of this paper are as follows:

(1) We employ the adaptive clustering algorithm CA to cluster the mobile devices in given regions. CA based on the quotient space theory has "blind" features without determining the number of clusters in advance; the algorithm can automatically identify the number of clusters based on the characteristics of the data and is independent of the initial centers. And we parallelize the CA on Spark to increase the efficiency of EACP-CA in processing big data.

(2) We implement the enhanced adaptive cloudlets placement and obtain the traces of mobile cloudlets any distribution of the mobile devices with changing over time.

(3) We conduct extensive experiments to comprehensively compare EACP-CA with existing k-means based approach.

This paper is organized as follows. Section 2 reviews the related work. Section 3 overviews the preliminary knowledge. Section 4 introduces enhanced adaptive cloudlet placement approach for mobile applications. Section 5 presents the experimental results and analysis, and Section 6 concludes this paper.

2. Related Work

Mobile cloud computing provides information technology service environment and cloud computing capabilities within the radio access network closest to the user's mobile devices,

aiming to further reduce latency, enhance network operating efficiency, boost service distribution, and improve user experience, which can also increase the computing capacity of mobile devices by offloading the workload to clouds [1, 2, 9, 16–19]. Many kinds of researches have been conducted on mobile cloud computing [2, 16, 20–22], but most of them are about remote clouds which are physically far away from the uses and cloudlet offloading technology. The remote clouds lead to a long latency, which is negative for improving the user experience and the performance of cloud services [4, 5]. In view of the above issues, researchers present the cloudlets with computing and storage resource-rich which can deploy at APs in a network [3, 7]. The cloudlets are deployed physically close to the users and act as offloading destinations of the mobile user which can significantly short the response time for the users' requirements and also can reduce the energy consumption of mobile devices.

There are some studies investigated the cloudlets. To name a few, Jia et al. [3] study the cloudlet placement and mobile user allocation to the cloudlets at user dense region of the wireless metropolitan area network (WMAN) and assign mobile users to the placed cloudlets while balancing their workload. Xu et al. [12] focused the cloudlet placement problem in a large-scale WMAN consisting of many wireless APs, in which capacitated cloudlets need to find the best deployment locations within a given set of candidate locations in WMAN. The objective is to minimize the average access delay between the mobile users and the activated cloudlets serving the users. Ma et al. [13] propose a New Heuristic Algorithm (NHA) and a Particle Swarm Optimization (PSO) algorithm for the delaying problem. In the above approaches, cloudlets are fixed in the placement, which cannot maintain efficiency in response to the requirements of applications for frequently moving devices. There are some researchers proposed the movable cloudlets to enhance the performance of the cloud service with the mobile users. Xiang et al. [7] propose self-adaptive edge cloud placement based on the locations of mobile applications. The core idea of the method is to maximize the number of mobile devices that are covered by the known active area of the edge cloud. They use the k-means algorithm to conduct the clustering process. However, k-means clustering algorithm has some drawbacks that the number of clusters k cannot be easily determined, and the clustering results heavily rely on the initial centers of random selection. And the selection of the initial center has a significant impact on the final clustering result and is easily disturbed by outliers [15, 23, 24].

In this paper, we propose EACP-CA, a novel approach for enhanced adaptive cloudlets placement. EACP-CA efficiently addresses the limitations and issues of most existing clustering approaches with a novel covering-based clustering technique. The CA algorithm we employed has "blind" features, without determining the number of clusters in advance, which can automatically identify the number of clusters based on the characteristics of the data and is independent of the initial centers. And EACP-CA can efficiently mitigate the issue of data scalability on Spark.

3. Preliminaries

In this section, we formally define the problem accurately to facilitate further introduction on enhanced adaptive cloudlet placement for mobile applications. Before conducting the introduction, we summarize the notions used throughout this paper in Table 1 to simplify the discussion.

Most existing studies in mobile edge computing focus on the limited of computing capacity, storage, and energy savings of mobile devices by offloading high-complexity and computing-intensive tasks from mobile devices to remote clouds [2, 9, 10]. However, the approach of offloading computing tasks to the cloud computing center not only brings about a large amount of data transmission and increases the network load, but also increases the transmission delay, which has a certain impact on the delay-sensitive service and the user experience [15]. Therefore, to effectively solve the requirements of high bandwidth and low latency brought about by the rapid development of the mobile Internet and the Internet of Things, the mobile edge computing has received extensive attention from the academic community and the industry. The cloudlets play an important role in the mobile edge computing which can significantly enhance the performance of mobile devices and meet the mobile users' response time requirements simultaneously, as shown in Figure 1. The mobile cloud framework consists of three main parts presented in Figure 2. Part 1 is mobile device clients, Part 2 is cloudlets, and Part 3 is the remote cloud. The mobile device clients can directly access the cloud service through the APs or can connect to the network through the wireless network. What is more, there is also a remote cloud which can be accessed from APs through the Internet. When the cloudlets cannot meet the request of the mobile devices, some computationally intensive tasks and data can be offloaded to the remote cloud for processing. Mobile devices can quickly access nearby cloudlets through APs to obtain cloud storage and cloud computing resources, so cloudlets can help reduce the access latency.

Effectively dealing with the cloudlet placement problem in WMAN consisting of many APs can contribute to enhancing the cloud services for dynamic mobile computationally intensive applications and improve the user experience. In this paper, we apply a rectangle activity area to place the cloudlets. In addition, other shaped areas can also be split by multiple different size rectangles.

3.1. Cloudlet Placement Strategy. If cloudlet l_m with the central position $p(x, y)$ is providing service to the users with the mobile devices at time t, the number of the mobile devices covered by l_m at time t should meet the condition that $dc\_l_m^{p_{(t)}} \geq \sigma$. Our goal is to find optimal cloudlet placement in A and maximize the total number of covered mobile devices which is calculated by the following objective function:

$$AN(t) = \left| \bigcup_{m=1}^{M} dc\_l_m^{p_{(t)}} \right| \tag{1}$$

Assume cloudlet l_m is placed at $p(x, y)$ at time t, and, in the time range $(t, t']$, the mobile devices in A move randomly.

Table 1: Mathematical notation.

Symbol	Explanation
A	Denotes the set of points in mobile devices activity area, $A = \{(x, y) \mid 0 \le x \le W,\ 0 \le y \le H\}$
MD	Denotes the set of mobile devices, $D = \{d_1, d_2, \ldots, d_N\}$
MDP	Denotes the set of mobile devices positions
d_n	Denotes the n^{th} $(1 \le n \le N)$ mobile device in D
$d_n^{P(t)}$	Denotes the position of dn at time t, $d_n^{P(t)} = (d_n^{P(t)}\_x, d_n^{P(t)}\_y)$, where $d_n^{P(t)}\_x$ and $d_n^{P(t)}\_y$ represents the latitude and longitude of the location
L	Denotes the set of the cloudlets, $L = \{l_1, l_2, \ldots, l_N\}$
l_m	Denotes the m^{th} cloudlet
CL	Denotes the set of the central position of the cloudlets
$l_m^{P(t)}$	Denotes the central position of l_m at time t, $l_m^{P(t)} = (l_m^{P(t)}\_x, l_m^{P(t)}\_y)$
AP	Denotes the set of the Aps, AP = $\{ap_1, ap_2, \ldots ap_m\}$
rm	Denotes the converage radius for l_m
$dc\_l_m^{P(t)}$	Denotes the mobiles devices collection of l_m^p at time t, $dc\_l_m^{P(t)} = \left\{d_n^{P(t)} \mid dis\left(mdp_n^{p(t)}, l_m^{P(t)}\right) \le r,\ 1 \le\le N\right\}$
$dis\left(mdp_n, l_m^{P(t)}\right)$	Denotes the Euclidean distance between $mdp_n^{p(t)}$ and $l_m^{P(t)}$, $dis(mdp_n^{p(t)}, l_m^{P(t)}) = \sqrt{(mdp_n^{p(t)}\_x - l_m^{P(t)}\_x)^2 + (mdp_n^{p(t)}\_y - l_m^{P(t)}\_y)^2}$
dc_cp_i	Denotes the mobile devices collection of cp_i
σ	Denotes the density threshold for cloudlet placement judgment
$AN(t)$	Denotes the number of covered mobile devices by all cloudlets
$dc\_l_m^{P(t')}$	Denotes the mobile device collection of l_m^p at time t'
V	Denotes the available positions that cloudlets can only stay
v_u	Denotes the u^{th} available position that cloudlets can stay
E	Denotes the edges between adjacent position in V
e_{uv}	Denotes the edge between two APs v_u and v_v
W	Denotes the weight of all cloudlet available positions
G	Denotes the paths graph in A, $G = \{E, V, W\}$
$p(x, y)$	Denotes the coordinates of position

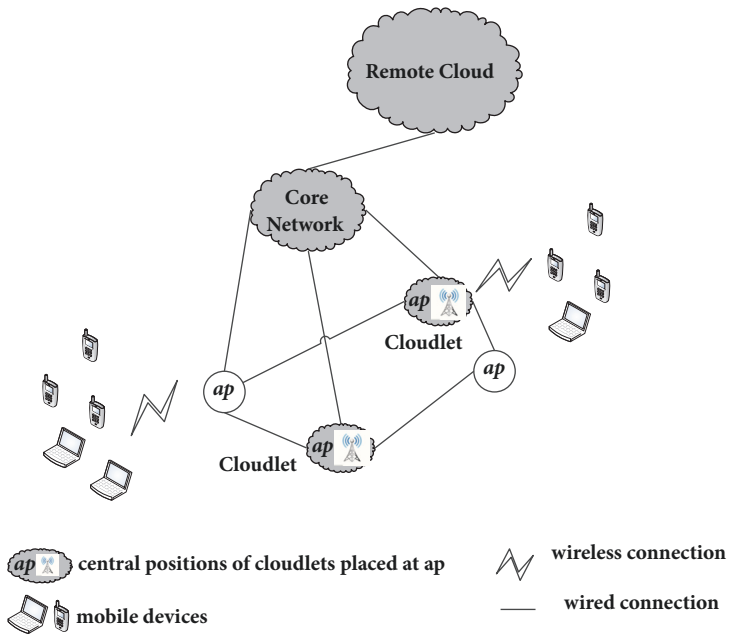


Figure 2: The mobile cloud framework.

The place position of cloudlet l_m at time t may not suitable for distribution of mobile devices after moving at time t'. In order to maintain the cloudlets working efficiently, we should propose a cloudlet movement strategy as the move reference for the cloudlets.

3.2. Cloudlet Movement Strategy. After mobile devices have moved at time t', if the cloudlet l_m is still placed at the position $p(x, y)$ at the time t, the mobile devices' collection of position $p(x, y)$ at time t' is denoted as $dc\_l_m^{p_{(t')}}$, and another place position $p'(x', y')$ for cloudlet l_m at time t' is obtained by adaptive cloudlet placement approach, and satisfying the cloudlet placement strategy is denoted as $dc\_l_m^{p'_{(t')}}$. If meet the condition that $dc\_l_m^{p'_{(t')}} \geq dc\_l_m^{p_{(t')}}$ is met, and there are no other cloudlets placing around p' at time t' within radius r, cloudlet l_m will move from $p(x, y)$ to $p'(x', y')$.

4. Enhanced Adaptive Cloudlet Placement Approach for Mobile Applications

The placement of the cloudlets has a significant impact on the resource utilization of the cloudlets. Inappropriate placement of cloudlets can cause the severe imbalance in the edge cloud load. Some cloudlets are overloaded, while others are underloaded or even idled and reducing the mobile users' response time requirements.

To obtain the proper cloudlet placement, now we introduce our main approach to the cloudlet placement. The enhanced adaptive cloudlet placement consists of three stages. Stage 1 performs the parallel CA clustering algorithm to obtain the central positions of mobile devices gathering place. Stage 2 confirms the cloudlet locations. Stage 3 realizes the adaptive cloudlet placement. The three stages are discussed in detail as follows.

Stage 1. We observe that mobile devices-dense regions of A areas are suitable to place the cloudlets, which means that cloudlets are situated close to a large number of devices and can reduce the average network latency between mobile devices and cloudlets. Therefore, we propose a clustering algorithm named CA to adaptive identify the central positions of device gathering place. Algorithm 1 presents the pseudocode for the CA algorithm and we use Table 2 to simply introduce these functions in Algorithm 1. The detailed description of these functions is in Section 3.3 in [23]. In order to better illustrate the effectiveness, we also discuss the time complexity of the CA.

4.1. Time Complexity Analysis of the Algorithm 1. In Algorithm 1, the computational complexity of line (5) is $O(n)$ because dataset *MDP* contains a maximum of n points. Similarly, the computational complexities of lines (5) and (6) are also $O(n)$, and those of lines (7), (8), and (9) are also $O(n)$ because the number of clusters is smaller than n. Lines (10)-(15) will be repeated until the data points in the cluster do not change. Lines (3)-(15) must also be repeated until all of the data points in *MAP* are covered, and the number of repetitions *num_C* is much smaller than n. In line (3), the radius of a cluster is the average distance between the center of the cluster and all of the data points that are not covered by any clusters. On average, each newly created cluster covers half of the uncovered data points, and the computational complexity is $O(\log n)$. In line (17), the computational complexity is $O(p)$ because there is a maximum of p clusters after the initial covering process. Similarly, the computational complexity of line (18) is $O(p)$

Input: *MDP*
Output: Results of parallel covering with granularity analysis–A
set of clusters $CP = \{CP_1, CP_2, \ldots\}$
Begin
(1) center c = null
(2) **Set** $C_u = MDP$
(3) **do**
(4) center $c \longleftarrow$get_center(C_u)
(5) radius $r \longleftarrow$ get_weight_radius(c,C_u)
(6) Covering C_{form} = get_covering(c,r,C_u)
(7) c $\longleftarrow$ get_centroid(C_{form})
(8) $r \longleftarrow$ get_weight_radius(c,C_u)
(9) Covering C_{last} = get_covering(c,r,C_u)
(10) **while** C_{last}.subtractByKey(C_{form}) > 0
(11) $C_{form} \longleftarrow C_{last}$
(12) c $\longleftarrow$ get_centroid(C_{form})
(13) $r \longleftarrow$ get_ radius_centroid(c, C_u)
(14) C_{last} = get_covering(c,r,C_u)
(15) **end while**
(16) **while**($C_u \neq \emptyset$)
(17) **Do** Split Operation
(18) **Do** Merge Operation
(19) return $CP = \{CP_1, CP_2, \ldots\}$
End

Algorithm 1: CA (*MDP*).

Input: *MDP*, *V*
Output: A set of mobile device central positions *CCP*
Begin
(1) centers $\longleftarrow$ CA(*MD*)
(2) $CP \longleftarrow \emptyset$
(3) **for** i = 1 to k
(4) **do**
(5) $Pos \longleftarrow {}^{\text{argmin}}_{v_j} dis(cp_i, v_j)$ (j=1 to U)
(6) add *Pos* to *CCP*
(7) **end for**
(8) return *CCP*
End

Algorithm 2: Center position adaptive identification (*MDP*, *V*).

because there is a maximum of p clusters. The number of clusters is much smaller than n. Thus, the computational complexity of Algorithm 1 is $O(n) \times O(\log n) + O(p) = O(n \log n)$.

Therefore, we obtain the central positions of mobile devices gathering place through the CA clustering algorithm.

Considering the security and efficiency of path researching in adaptive cloudlet placement, the whole cloudlets can only place to APs [25]. Then we employ the path graph G to adjust the obtained central positions and generate moving traces of the mobile cloudlets. Algorithm 2 presents the pseudocode for the adjusting of the central positions obtained by CA.

Stage 2. We obtained several mobile device central positions after performing Stage 1. Then we will conduct the confirmation operation to filter the undesired center positions. The confirmation of center positions operation is introduced as follows.

4.2. Confirmation of Center Positions Operation. Through Stage 1, we get the mobile device central positions set *CCP*. For each center position cp_i in *CCP*, the confirmation of center positions operation filters the mobile devices that the distances between cp_i and each mobile device position are shorter than radius r. And if cp_i center position contains more than σ mobile devices, then add the cp_i center position to the set of cloudlet central position denoted by *CP*. After the confirmation operation, we get the set of cloudlet central position *CP*. We use the following formula for a brief explanation.

$$CP = \left\{ cp_i \mid cp_i \in CCP \text{ and } \left| dc_{cp_i} \right| \geq \sigma \right\} \tag{2}$$

Stage 3. After the above confirmation operation, we get the desired mobile devices central positions. According to the distribution of the mobile devices in A, if each cloudlet moves to the center position which is surrounded by dense mobile devices, these cloudlets will achieve high utilities. Therefore we will propose a method to choose a suitable center position in *CP* as its destination position. Assuming we have already known the previous cloudlets, we describe the selection strategy for cloudlets location to introduce the selection of each cloudlet. For the subproblem of the cloudlet selection, a straightforward idea is to greedily select the

Table 2: Explanations of functions.

Functions	Explanation
get_center(C_u)	Denotes a function has the function of obtaining the center of the data set C_u. The specific acquisition process of the center of the circle is to calculate the data point closest to the center of gravity of the data set C_u.
get_weight_radius(c,C_u)	Denotes a function has the function of obtaining the weighted radius based on center c and data set C_u.
get_covering(c,r,C_u)	Denotes a function has the function of obtaining the centroids of the current spheres continually according to the obtained center and radius and obtain new clusters until the number of clusters in the data points does not increase.
get_centroid(C_{form})	Denotes a function has the function of obtaining the center of gravity of data set C_u.
Split Operation	Denote a function has the function of combing the most similar pair of clusters into a new cluster.
Merge Operation	Denote a function has the function of splitting the clusters with more data points.

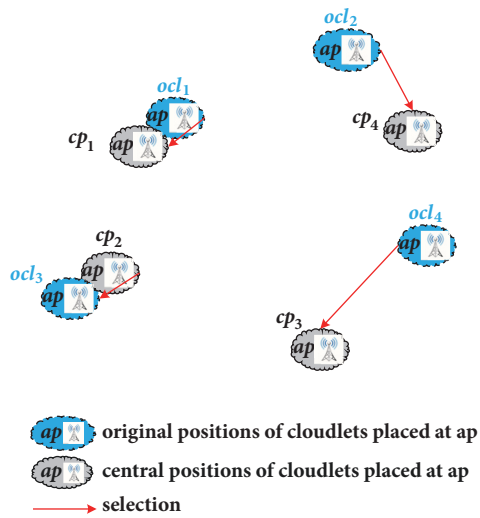


FIGURE 3: The example of locations selection mechanism.

cloudlet with the minimum distance. The center position selected by each cloudlet is according to the distance between cloudlet previous location and the mobile devices central positions.

4.3. Locations Selection Mechanism. Supposing that we already know the original positions and the destination positions of cloudlets, i.e., the central positions, the cloudlets denoted by $L = \{l_1, l_2, \ldots, l_m\}$ and the original positions denoted by $OCL = \{ocl_1, ocl_2, \ldots, ocl_m\}$, the destination positions of cloudlets are obtained by the above stages which are denoted by $CP = \{cp_1, cp_2, \ldots, cp_m\}$. We compute all the distances between cl_i and cp_j, and get distance set $OCP = \{ocp_1, ocp_2, \ldots, ocp_m \mid ocp_i = (cp_1, \ldots, cp_p \ldots, cp_q, \ldots, cp_m)$ where $1 \le p \le q \le m$ and $dis(ocl_i, cp_p) < dis(ocl_i, cp_q)\}$. While not all cloudlets have selected a central position, we add ocl_i to the intermediate set i_c_j that cp_j is the nearest central position of l_i in ocp_i, and then for each i_c_j that is not empty. If each cp_i corresponding to i_c_j have not selected a cloudlet or a nearest cloudlet selected cp_i this time, we obtain cp_i' nearest l_j in i_c_u and cp_i selects ocl_j. If there are some existing cloudlets in i_c_j are not selected, we delete the cp_i from ocp_a that ocl_a in i_c_j is not selected. We repeat those operations until all cloudlets have selected a central position. In order to explain the locations select mechanism process visually, we use a specific example to explain which are presented in Figure 3.

As we can see from Figure 3, there are 4 cloudlets, l_1, l_2, l_3, l_4, 4 original cloudlet positions, ocl_1, ocl_2, ocl_3, ocl_4, and 4 central positions, cp_1, cp_2, cp_3, cp_4. For ocp_1, ocp_2, ocp_3, *and* ocp_4, *we, respectively, get* $ocp_1 = (cp_1, cp_2, cp_4, cp_3)$, $ocp_2 = (cp_4, cp_1, cp_2, cp_3)$, $ocp_3 = (cp_2, cp_1, cp_3, cp_4)$, and $ocp_4 = (cp_4, cp_3, cp_1, cp_2)$. Now, each of cloudlets does not select a *cp*. For all cloudlets, we figure out that $i\_c = \{i\_c_1 = \{ocl_1\}, i\_c_2 = \{ocl_3\}, i\_c_4 = \{ocl_2, ocl_4\}\}$. Then we will deal with the unempty i_c_i. For each cloudlet in i_c_1, cp_1 has not selected a cloudlet, we get the cp_1' nearest cloudlet ocl_1 in i_c_1 and cp_1 selects ocl_1. For each cloudlet in i_c_2, cp_2 has not choose a cloudlet, we obtain the cp_2' nearest cloudlet ocl_3 in i_c_2, and cp_2 selects ocl_3. For each cloudlet in i_c_4, cp_4 has not selected a cloudlet, we acquire the cp_4' nearest cloudlet ocl_2 in i_c_4 and cp_4 chooses ocl_2, ocl_4 in i_c_4 is not selected. So we delete the cp_4 in ocp_4, now the $ocp_4 = (cp_3, cp_1, cp_2)$. We can find that ocl_4 is still not selected a *cp*, so we will repeat these operations. For each cloudlet in *OCL* which has not selected a *cp*, we come to the updated $i\_c = \{i\_c_1 = \{ocl_1\}, i\_c_2 = \{ocl_3\}, i\_c_3 = \{ocl_4\}, i\_c_4 = \{ocl_2\}\}$. For each cloudlet in i_c_1, cp_1' nearest cloudlet ocl_1 chooses cp_1, we obtain the cp_1' nearest cloudlet ocl_1 in i_c_1 and cp_1 selects ocl_1. For each cloudlet in i_c_2, cp_2' nearest cloudlet ocl_3 chooses cp_2, we get the cp_2' nearest cloudlet ocl_3 in i_c_2 and cp_2 selects ocl_3. For each cloudlet in i_c_3, cp_3 has not choose a *cp*, we acquire the cp_3' nearest cloudlet ocl_4 in i_c_3 and cp_3 selects ocl_4. For each cloudlet in i_c_4, cp_4' nearest cloudlet ocl_2 chooses cp_4, we get the cp_4' nearest cloudlet ocl_2 in i_c_2 and cp_4 chooses ocl_2. So far, all the cloudlets have selected a *cp*.

After the selection, we will employ the enhanced adaptive cloudlet placement approach to obtain the moving trace of each cloudlet. As introduced above, the previous cloudlet locations and the destination positions of all cloudlets all belonged to *V*. Consequently, the problem of the generation of moving trace can transform to the shortest path problem. We use the Dijkstra algorithm for generation of moving trace and then the moving traces will be transmitted to corresponding cloudlet moving. The ultimate overall process of enhanced cloudlets placement is presented in Algorithm 3.

5. Experience Evaluation

In this section, we present the performance evaluation of our proposed EACP-CA approach. And we also experimentally compare our proposed approach with k-means based approach.

The Spark cluster used to implement CA is built on a cluster with three connected nodes. Master/Slave is ThinkCentre M8500t-N000 Intel(R) Core(TM) i7-4790 CPU 3.60GHZ, 4cores, 8CPUs, DDR3 1600MHz SDRAM; 1 disk on the master and 2 disks on the slave: 1TB, 7.2K RPM SATA Hard Drive.

5.1. Experimental Setup. The experiments are conducted on the randomly generated location datasets deferring to the Gaussian distribution which can better simulate real-world applications. The parameters of Gaussian distribution used for generating location datasets are illustrated in Table 3, where μ and σ represent mean and standard deviation, respectively. 200, 300, and 400 represent the number of mobile devices. We will also introduce the parameters used in our experiments. The shape of mobile device activity area A is set to square and the ranges of x-axis and y-axis of this area are both in [0m, 180m]. The locations of mobile device are randomly generated and distributed in A. The numbers of the mobile devices are moderate values of $N \in \{200, 300, 400\}$. We set the density threshold σ for cloudlet placement judgment as 30, the coverage radius for cloudlet as 30m, and the time of period of experiments t as 30 minutes. We change the number of APs M from 1 to 7. This way,

Table 3: The parameters of Gaussian distribution.

Devices	200 (μ, σ)	300 (μ, σ)	400 (μ, σ)
Fixed devices	(30,25),(50,16), (110,18),(150,19)	(30,25),(50,16), (110,18),(150,19)	(30,25),(50,16), (110,18),(150,19)
Moving devices of t_0	(60,19),(80,13), (150,25),(140,30)	(60,22),(65,20), (130,30),(140,22)	(55,35),(40,40), (110,15),(130,30)
Moving devices of t_1	(55,28),(60,15), (150,25),(120,30)	(60,22),(65,20), (130,30),(140,22)	(55,35),(40,40), (110,40),(130,30)
Moving devices of t_2	(60,19),(80,15), (150,25),(150,30)	(60,22),(65,20), (130,30),(140,22)	(55,35),(40,40), (110,40),(130,30)

Input: G
Output: The moving trace of all cloudlets
Begin
(1) **for** $t = 1$ to $t_{\max}$
(2) **do**
(3) $CCP \longleftarrow$ **Algorithm** 2 (***MDP***, ***V***)
(4) $CP \longleftarrow$ filteringCCP through ***Confirmation of center positions operation***
(5) ($ocl_i \in OCL$, ocl_i selected cps) $\longleftarrow$ ***Locations selection mechanism***
(6) **for** $i = 1$ to $|OCL|$
(7) **do**
(8) **if** l_i have selected a cp **then**
(9) shortest trace tra_i from ocl_i to its selected $cp \longleftarrow$ Dijkstra algorithm (G, l_i)
(10) transmit tra_i as the moving description to l_i
(11) **end if**
(12) **end for**
(13) **end for**
End

Algorithm 3: Enhanced adaptive cloudlets placement (G).

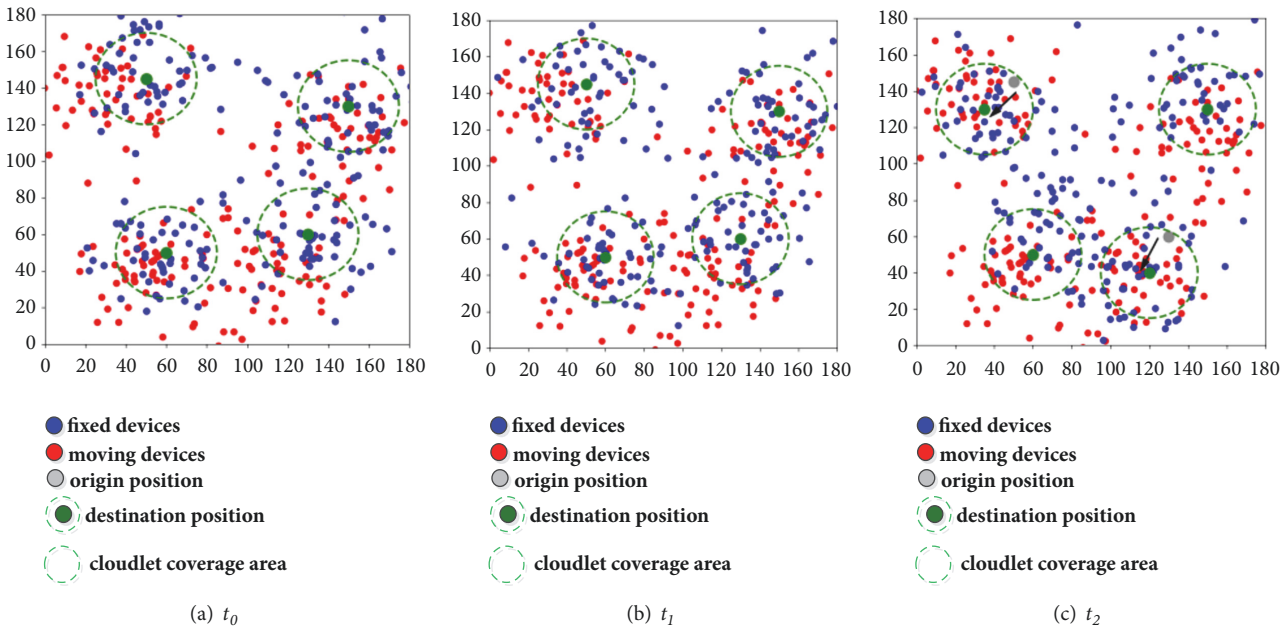
we comprehensively evaluate EACP-CA's ability to handle datasets with different characteristics in various application scenarios.

5.2. Performance Evaluation and Comparison. In this section, we will evaluate the performance of our proposed approach and compare it with k-means based approach through the number of movable cloudlets covered mobile devices.

In order to intuitively display the distributions of mobile devices and cloudlets in A, we show three sequential record instances, i.e., t_0, t_1, and t_2 in our experiments. The different times distributions of mobile devices and cloudlets are illustrated in Figure 4. Figure 4(a) describes the distributions at time t_0. And at time t_1, since the positions of the mobile devices are not changed significantly, there is no corresponding change in the locations of the mobile cloudlets, as we can see in Figure 4(b). But at time t_2, the distribution of mobile devices has changed greatly and we can also find that the two cloudlets move to new locations illustrated in Figure 4(c).

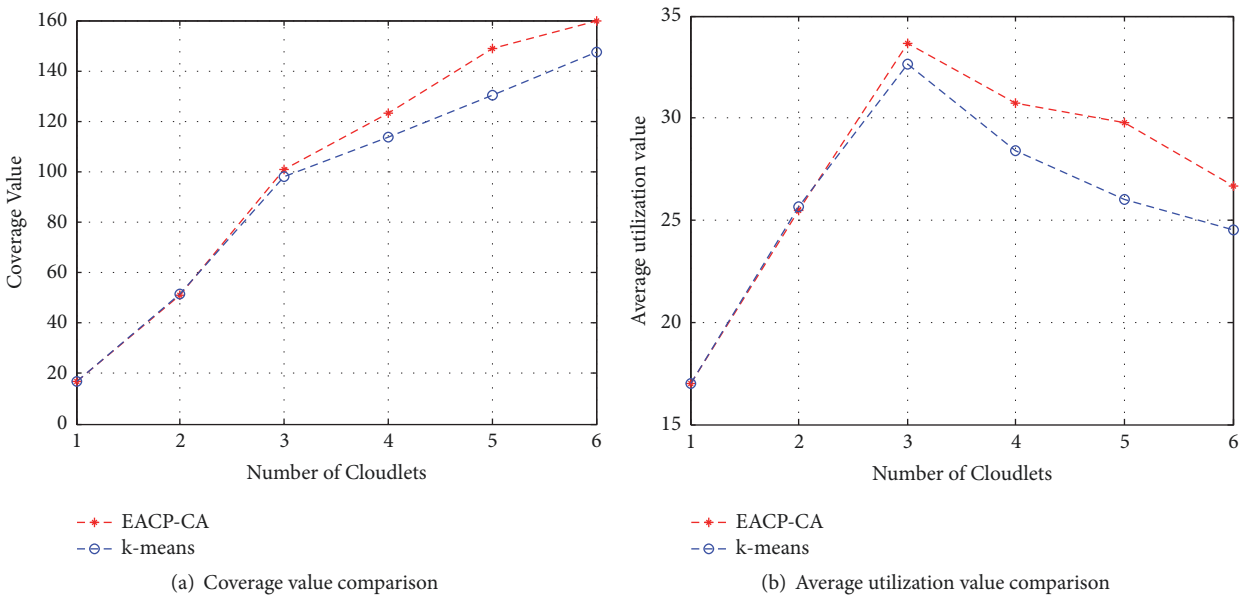
We investigate the performance of our approach and k-means based approach with different cloudlet numbers. We simulate to generate three datasets, which include 200, 300, and 400 mobile devices. The experimental results are shown in Figures 5, 6, and 7 demonstrating that our approach outperforms k-means based approach. Figure 5 presents the results for the dataset with 200 mobile devices. Figure 5(a) shows the coverage value by all cloudlets through executing our EACP-CA approach and k-means based approach, while Figure 5(b) shows the average utilization value of each cloudlet in L. And the experimental results for datasets with 300 mobile devices and 400 mobile devices which are shown in Figures 6 and 7, respectively, are the same as the experimental results of dataset with 200 mobile devices. Figures 6(a) and 7(a) show the coverage value by all cloudlets through executing our EACP-CA approach and k-means based approach for the datasets with 300 mobile devices and 400 mobile devices, respectively, while Figures 6(b) and 7(b) show the average utilization value of each cloudlet in L.

The experimental results illustrated in Figures 5, 6, and 7 show that EACP-CA is better than k-means based approach because both the coverage value and average utilization of each cloudlet within different cloudlet numbers obtained by EACP-CA are higher than k-means based approach. The experimental results show that, with the rising number of cloudlets, the coverage value obtained by EACP-CA and k-means based approach also increases with the fixed number of mobile devices but becomes less significant. This is caused by the fixed number of mobile devices and the increased number of cloudlets, because, through performing the clustering algorithm, the dense mobile devices are covered by



(a) t_0 (b) t_1 (c) t_2

Figure 4: Distributions of mobile devices and cloudlets at t_0, t_1, and t_2.



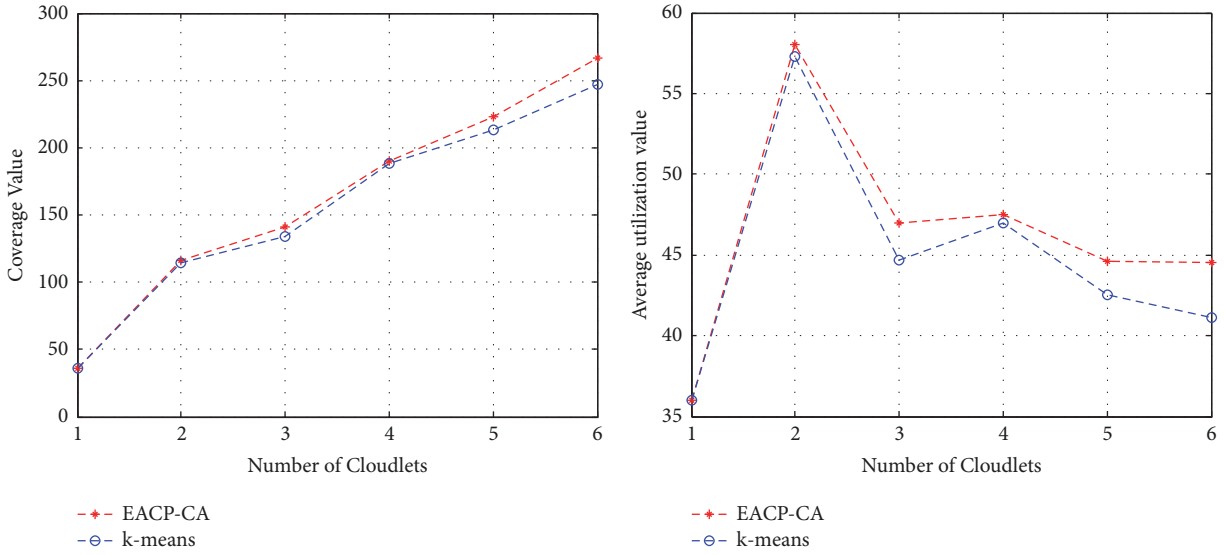
(a) Coverage value comparison (b) Average utilization value comparison

Figure 5: Comparison of performance with EACP-CA and k-means (number of mobile devices $N = 200$).

many cloudlets and the remained discrete mobile devices are also covered by another cloudlet which leads decreasing of the average utilization of each cloudlet. Therefore, we can find that the coverage value is impacted by the number of cloudlets while the number of mobile devices is fixed. With the growing number of cloudlets, the average utilization of each cloudlet is increasing at the beginning and then begins to decrease again. Simultaneously, we can also apply the average utilization value of cloudlets to select a suitable number of cloudlets. We can obtain the appropriate number of cloudlets when the average utilization value of cloudlets and the number of users covered become lager.

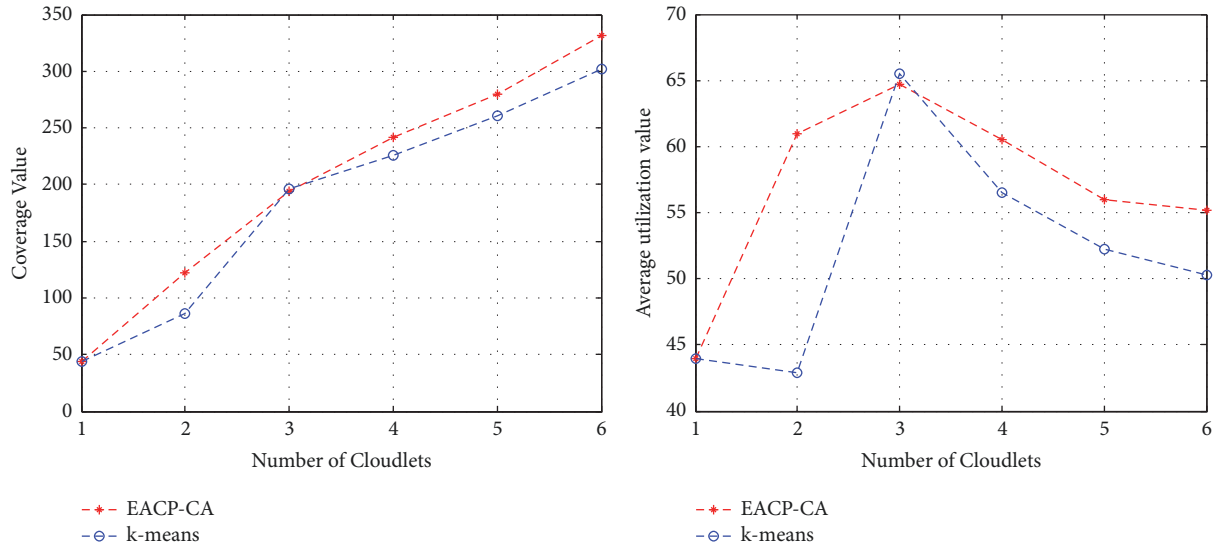
6. Conclusion and Future Work

In this paper, we propose EACP-CA, a novel covering-based approach for enhanced adaptive cloudlets placement. Given a known mobile devices activity area, according to the characteristics of the data and being independent of the initial centers, EACP-CA first employs CA with "blind" features to adaptively partition mobile devices into clusters based on the physical distances between their positions at time t. Then we adjust the central positions obtained by clustering algorithm through the graph path. Next, we execute confirmation of the center positions operation and get the desired



(a) Coverage value comparison (b) Average utilization value comparison

Figure 6: Comparison of performance with EACP-CA and k-means (number of mobile devices $N = 300$).



(a) Coverage value comparison (b) Average utilization value comparison

Figure 7: Comparison of performance with EACP-CA and k-means (number of mobile devices $N = 400$).

cloudlets central positions according to the distribution of the mobile devices. We also need to obtain the pair of the original mobile cloudlet positions and the positions. Finally, we conduct the enhanced adaptive cloudlets placement to finish the mobile cloudlets placement. We parallelize the portioning operation of EACP-CA on Spark to increase the efficiency of EACP-CA in processing big data. The results of the experiments on the simulated datasets demonstrate that EACP-CA significantly outperforms the k-means based approach, which also demonstrate that EACP-CA is more efficient in clustering and improving the utilities of mobile cloudlets.

In our future work, we will enhance EACP-CA to solve big data and consider the assignment of mobile users to the placed cloudlets.

Data Availability

The dataset used to support the findings of this study have been deposited in the repository and can be accessed via the next link: http://bigdata.ahu.edu.cn/paper.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Science Foundation of China (no. 61872002) and the Natural Science Foundation of Anhui Province of China (no. 1808085MF197).

References

[1] P. Asrani, "Mobile cloud computing," *International Journal of Engineering and Advanced Technology*, vol. 2, no. 4, pp. 606–609, 2013.

[2] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: can offloading computation save energy?" *The Computer Journal*, vol. 43, no. 4, Article ID 5445167, pp. 51–56, 2010.

[3] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 725–737, 2017.

[4] M. Chen, Y. Zhang, Y. Li, S. Mao, and V. C. M. Leung, "EMC: emotion-aware mobile cloud computing in 5G," *IEEE Network*, vol. 29, no. 2, pp. 32–38, 2015.

[5] M. Chen, Y. Zhang, L. Hu, T. Taleb, and Z. Sheng, "Cloud-based wireless network: virtualized, reconfigurable, smart wireless network to enable 5G technologies," *Mobile Networks and Applications*, vol. 20, no. 6, pp. 704–712, 2015.

[6] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[7] H. Xiang, X. Xu, H. Zheng et al., "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM 2016*, USA, December 2016.

[8] D. Niyato, P. Wang, P. C. H. Joo, Z. Han, and D. I. Kim, "Optimal energy management policy of a mobile cloudlet with wireless energy charging," in *Proceedings of the 2014 IEEE International Conference on Smart Grid Communications, SmartGridComm 2014*, pp. 728–733, Italy, November 2014.

[9] C. You and K. Huang, "Multiuser resource allocation for mobile-edge computation offloading," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM 2016*, USA, December 2016.

[10] Y. Zhang, D. Niyato, and P. Wang, "Offloading in Mobile Cloudlet Systems with Intermittent Connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, 2015.

[11] E. Ahmed and M. H. Rehmani, "Mobile Edge Computing: Opportunities, solutions, and challenges," *Future Generation Computer Systems*, vol. 70, pp. 59–63, 2017.

[12] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient Algorithms for Capacitated Cloudlet Placements," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, 2016.

[13] L. Ma, J. Wu, L. Chen, and Z. Liu, "Fast algorithms for capacitated cloudlet placements," in *Proceedings of the 21st IEEE International Conference on Computer Supported Cooperative Work in Design, CSCWD 2017*, pp. 439–444, New Zealand, April 2017.

[14] H. Yao, C. Bai, M. Xiong, D. Zeng, and Z. Fu, "Heterogeneous cloudlet deployment and user-cloudlet association toward cost effective fog computing," *Concurrency Computation*, vol. 29, no. 16, 2017.

[15] L. Bottou and Y. Bengio, "Convergence properties of the k-means algorithms," in *Proceedings of the Advances in neural information processing systems*, pp. 585–592, 1995.

[16] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[17] J. Li, Y. K. Li, X. Chen, P. P. C. Lee, and W. Lou, "A Hybrid Cloud Approach for Secure Authorized Deduplication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1206–1216, 2015.

[18] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, pp. 90–99, 2018.

[19] J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *IEEE Transactions on Computers*, 2013.

[20] J. Li, L. Huang, Y. Zhou, S. He, and Z. Ming, "Computation partitioning for mobile cloud computing in a big data environment," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2009–2018, 2017.

[21] K. Gai, M. Qiu, H. Zhao, L. Tao, and Z. Zong, "Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing," *Journal of Network and Computer Applications*, vol. 59, pp. 46–54, 2016.

[22] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[23] Y. Zhang, Y. Zhou, X. Guo et al., "Self-Adaptive K-means based on Covering Algorithm," *Complexity*, Article ID 7698274, 2018.

[24] Y.-W. Zhang, Y.-Y. Zhou, F.-T. Wang, Z. Sun, and Q. He, "Service recommendation based on quotient space granularity analysis and covering algorithm on Spark," *Knowledge-Based Systems*, vol. 147, pp. 25–35, 2018.

[25] R. H. Jhaveri, N. M. Patel, Y. Zhong, and A. K. Sangaiah, "Sensitivity Analysis of an Attack-Pattern Discovery Based Trusted Routing Scheme for Mobile Ad-Hoc Networks in Industrial IoT," *IEEE Access*, vol. 6, pp. 20085–20103, 2018.