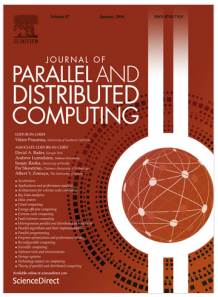
Accepted Manuscript



Edge server placement in mobile edge computing

Shangguang Wang, Yali Zhao, Jinlinag Xu, Jie Yuan, Ching-Hsien Hsu

PII: S0743-7315(18)30439-8
DOI: https://doi.org/10.1016/j.jpdc.2018.06.008
Reference: YJPDC 3900

To appear in: *J. Parallel Distrib. Comput.*

Received date : 12 May 2017
Revised date : 4 December 2017
Accepted date : 6 June 2018

Please cite this article as: S. Wang, Y. Zhao, J. Xu, J. Yuan, C.-H. Hsu, Edge server placement in mobile edge computing, *J. Parallel Distrib. Comput.* (2018), https://doi.org/10.1016/j.jpdc.2018.06.008

***Highlights (for review)**

The following are highlights of the contribution of this manuscript.

- Edge server placement is formulated as a multi-objective constraint optimization problem in this paper.
- The proposed techniques can balance the workloads of edge servers and minimize the access delay between the mobile user and edge server.
- Experimental results based on Shanghai Telecom's base station dataset show that our approach outperforms several representative approaches in terms of access delay and workload balancing.

Edge Server Placement in Mobile Edge Computing

Shangguang Wang[1], Yali Zhao[1], Jinlinag Xu[1], Jie Yuan[1], Ching-Hsien Hsu[2*]
[1]State Key Laboratory of Networking and Switching Technology; [2]Department of Computer Science and Information
[1]Engineering Beijing University of Posts and Telecommunications; [2]Chung Hua University
[1]Beijing 100876, China; [2]Hsinchu 707, Taiwan
{sgwang; zhaoyali2015; jlxu; yuanj}@bupt.edu.cn; chh@chu.edu.tw

Abstract—With the rapid increase in the development of the Internet of Things and 5G networks in the smart city context, a large amount of data (i.e., big data) is expected to be generated, resulting in increased latency for the traditional cloud computing paradigm. To reduce the latency, mobile edge computing has been considered for offloading a part of the workload from mobile devices to nearby edge servers that have sufficient computation resources. Although there has been significant research in the field of mobile edge computing, little attention has been given to understanding the placement of edge servers in smart cities to optimize the mobile edge computing network performance. In this paper, we study the edge server placement problem in mobile edge computing environments for smart cities. First, we formulate the problem as a multi-objective constraint optimization problem that places edge servers in some strategic locations with the objective to make balance the workloads of edge servers and minimize the access delay between the mobile user and edge server. Then, we adopt mixed integer programing to find the optimal solution. Experimental results based on Shanghai Telecom's base station dataset show that our approach outperforms several representative approaches in terms of access delay and workload balancing.

Keywords—*Mobile Edge Computing; Smart City; Edge Server Placement; Workload Balancing; Access Delay*

1. INTRODUCTION

In recent years, mobile smart devices have become increasingly important as a tool for entertainment, learning, news, businesses, and social networking for smarter living [1],[2]. The next-generation mobile networks aim to accelerate the development of smart cities, by not only increasing the data delivery rates but also increasing the amount of infrastructures used by smart city services and applications [2]. Although mobile applications are emerging and becoming computation-intensive, the computing capacity of mobile devices remains limited owing to the resource constraints of mobile devices (e.g., processing power, battery lifetime, and storage capacity), such that mobile users do not receive the same satisfaction compared to desktop device users [3]. An effective approach to enhancing the performance of mobile applications is to offload some of their tasks to remote resource-rich clouds [4],[5],[6],[7].

However, the cloud is often remotely located and far from mobile users, and the data transfer delays between users and the cloud can be long and unpredictable. This is especially undesirable for mobile applications in which an immediate response time is critical to users, such as reality-augmenting applications and mobile multiplayer gaming systems. To overcome the above-mentioned problem, cloudlet-based offloading has been proposed, where mobile devices offload computational process to a computing infrastructure (i.e., cloudlet) that is in relatively close proximity to the users using Wi-Fi access points (APs) [8]. Compared to cloud computing, cloudlets are less effective for the following reasons: cloudlets can be accessed only by

* Corresponding author

a Wi-Fi AP, which covers only small regions, and cloudlets are less resourceful compared to the cloud, so they are not scalable in terms of service and resource provisioning.

To overcome the above challenges, mobile edge computing is proposed [9],[10]. Mobile edge computing enables mobile users to access IT and cloud computing services in close proximity within the range of radio access networks [11],[12]. This approach enables the computation and storage capacity from the core network to be transferred to the edge network in order to reduce latency. Edge servers can be deployed in close proximity to enable devices to offload some of their mobile application workload to realize significant improvements in the quality of mobile user experiences.

Most existing studies have focused on offloading the workloads of mobile users to cloudlets to enable mobile devices to realize energy savings, and this approach assumes that the cloudlets have already been placed [13],[14],[15],[16]. Little attention has been paid to the effect of offloading the workloads of mobile users to edge servers and the placement of edge servers on the performance of mobile applications. In this paper, we focus on the edge server placement in a mobile edge computing environment that provides wireless internet coverage for mobile users in a large-scale metropolitan area. First, a large number of mobile users access edge servers in mobile edge computing environments because the metropolitan area that it covers has a high population density [17]. Secondly, because of the size of the network, service providers can take advantage of economies of scale when offering edge server services by making edge server services more affordable to the general public.

However, the placement of edge servers in mobile edge computing environments is challenging. The locations of edge servers are critical to the access delays of mobile users and the resource utilization of edge servers, especially in smart cities that include several hundreds or thousands of base stations through which mobile users access the edge servers. Owing to the large size of these networks, inefficient edge server placement will result in long access delays and heavily unbalanced workloads among edge servers, i.e., some of the edge servers will be overloaded while others are underutilized, or even idle. Therefore, the strategic placement of edge servers will significantly improve the performance of various mobile applications such as edge server access delay.

We assume that each edge server has the same limited computing resource to process mobile user requests, i.e., in this study, each edge server is identical, and edge servers are placed at some base station locations for mobile user access. The objective is to balance the workload among edge servers and minimize the edge server access delay. The challenge associated with such placements are determining 1) the locations in which edge servers should be placed, and 2) which base stations should be assigned to which edge servers, which we show to be an NP-hard problem. The main contributions of this paper are as follows:

1) We formulate the edge server placement problem as a multi-objective constraint optimization problem.

2) We adopt mixed integer programming (MIP) to find the optimal edge server placement with workload balancing among edge servers and minimizing the edge server access delay.

3) Experimental results that are based on datasets from about 3000 of the base stations operated by Shanghai Telecom show that our approach outperforms other approaches.

The rest of the paper is organized as follows: In Section 2, we review related work. In Section 3, we introduce the system model and problem definitions, and we propose our edge server placement approach. In Section 4, we illustrate the comparative experimental evaluation results, while in Section 5, we conclude the paper, including an outlook on our future work.

2. RELATED WORK

Few studies have focused on edge server placement in mobile edge computing environments. To the best of our knowledge, this is the first study to consider the placement of edge servers in a mobile edge computing environment. However, in recent years, there have been many works on cloudlet placement [18],[19],[20]. Cloudlets are typically

described as computers that are deployed at Wi-Fi APs in a network and act to offload the destinations of mobile users [21],[8],[22]. In the mobile edge computing environment, mobile users can access edge servers that are in close proximity within the range of base stations [11]. Edge servers can now also be considered as offloading destinations of mobile users, with the aim of reducing the access latency between mobile users and remote clouds. This is realized by importing the computation and storage capacity from the core network to the edge server [12].

It is believed that there are many similarities between the placement of cloudlets and edge servers [18], [19],[20]. Mike Jia et al. [18] proposed an offloading system model for multi-user mobile task offloading, and they studied cloudlet placement and mobile user allocation to the cloudlet. Then, they devised an algorithm to solve the problem with the aim of enabling the placement of cloudlets at regions with high user density, and assigned mobile users to the placed cloudlets while balancing their workloads. Zichuan Xu et al. [19] also studied the cloudlet placement problem using many wireless APs. First, they formulated the problem as a novel capacitated cloudlet placement problem that placed K cloudlets in different strategic locations with the objective of minimizing the access delay between mobile users and the cloudlets serving the users. Secondly, they proposed an exact solution to the problem by formulating it as an integer linear programming, and they then devised an efficient solution.

There are also a few studies [23],[24] on the computation offloading for mobile edge cloud computations. For example, Xu Chen et al. [24] studied the multi-user computation offloading problem for mobile edge cloud computing in a multi-channel wireless interference environment. They formulated the distributed computation offloading decision-making problem among mobile device users as a multi-user computation offloading game, and they designed a distributed computation offloading algorithm that achieves a Nash equilibrium.

Although these above-mentioned studies on the cloudlet placement problem are effective, Mike Jia et al. [18] focused only on balancing the workloads of cloudlets, and other studies [19],[20],[23],[24],[25],[26],[27]only considered the access delay. Inspired by this, in our approach, we propose to place edge servers for base stations in smart cities, and we then consider both the access delay and workload balancing as optimization objectives in mobile edge computing environments.

3. OUR APPROACH

In Section 3.1, we first introduce the system model and define some notations that are used throughout the rest of the paper. Then, we present the problem definition in Section 3.2. In Section 3.3, we formalize the edge server placement problem as a multi-objective optimization problem, and we describe the edge server placement model. Finally, in Section 3.4, we adopt MIP to find the optimal solution in terms of workload balancing and access delay. Related notations are explained in detail in Table 1.

Table 1. Notations

Symbol	Meaning
G	Mobile edge computing network
K	Number of edge servers
B	Set of all base stations in the network
S	Set of all edge servers that will be placed in the network
E	Set of links between a base station and an edge server at a location in S.
t_b	Workload of base station b and $b \in B$
T_s	Workload of edge server s and $s \in S$
l_b	Location of base station b and $b \in B$
l_s	Location of edge server s and $s \in S$
E_s	Set of base stations that are in control of edge server $s \in S$
d	Access delay between base station and edge server

3.1 System Model

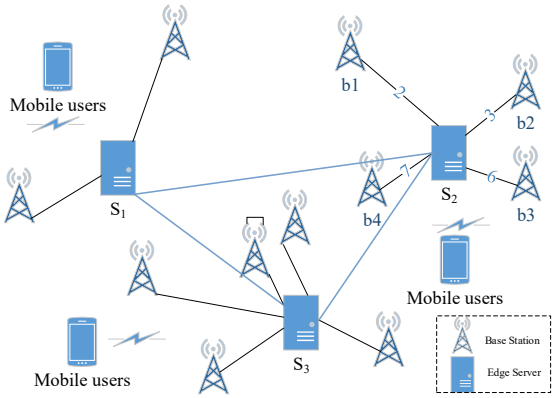


Fig. 1. An example of mobile edge computing with edge server placement.

In mobile edge computing environments, the edge server placement problem can be considered as a network, which is an undirected graph $G=(V,E)$ consisting of many mobile users, many base stations, and a set of potential locations for edge servers. $V=B\cup S$, where B is the set of base stations and S is the set of potential locations of edge servers. E represents the set of links between a base station and an edge server at a location in S. Note that the links between two base stations and the links between a mobile user and a base station are neglected in Fig. 1, because these links are not our focus. Mobile users receive services by sending requests to a base station in the mobile edge computing network, where each base station has a fixed radio coverage. Therefore, each base station processes mobile user requests, i.e., the number of mobile user requests to a specific edge server is its workload. We use t_b and l_b to represent the workload and location, respectively, for base station $b(b\in B)$. Similarly, l_s denotes the potential location for edge server $s(s\in S)$.

Assume that there are K edge servers that will be assigned to K different locations, where K is a constant. We assume that each edge server has the same limited computing resource to process mobile user requests, and each base station accessed the edge server directly, i.e., each base station is co-located with an edge server. Given the $K(K=1,2,3,...)$ edge servers, base stations can offload their tasks to the edge servers. We assume that each edge server is responsible for a subset of base stations in B to process the mobile user requests, and the same base station is not shared between any two edge servers, while the union sets of base stations in charge of edge servers is B. The workload to be processed by each edge server is distributed as evenly as possible, so we can balance workloads among limited edge servers and extend the service life of edge servers. Because each base station has its own edge server, the edge server access delay is proportional to the distance between the base station and edge server. We need to consider the account access delay in order to optimize the edge server placement in mobile edge computing networks.

From Fig. 1, we use edge server s_2 as an example to introduce the K edge server placement problem directly. We assume that four base stations access s_2 to obtain services, and the workload of base station b_1 is 2, b_2 is 3, b_3 is 6, and b_4 is 7. Therefore, we need to determine how to place s_2 among base stations to minimize the access delay. In addition to solving the above problem, it is also important to determine the dominant area of each edge server to realize a balanced workload.

3.2 Problem Definitions

The edge server placement problem in mobile edge computing networks $G=(V,E)$ is defined as follows. Given the network G, a set $\{s_1,s_2,...,s_K\}$ of K edge servers is to be placed in K potential locations, and the edge server processes all of the workloads of the base stations, i.e., the number of mobile user requests at each base station $b_j\in B$, which are in control of it.

In the mobile edge computing network, for a given set $\{s_1,s_2,...,s_K\}$ of K edge servers to be placed in K locations, i.e., $l=(l_{s_1},l_{s_2},...,l_{s_K})$, find an optimization placement solution for the edge server such that the workloads of the edge server can be balanced, and the access delay should be minimized subject to the following two constraints:

- No two edge servers share the same base station, and the combined set of base stations in charge of the edge servers is *B*.
- Each base station that is co-located with an edge server and an edge server will process all mobile user requests from the base station.

3.3 Edge Server Placement Model

In this paper, the objective of edge server placement is to balance the workload among edge servers and minimize the access delay between base stations and edge servers. Thus, edge server placement in a mobile edge computing network can be formulated as a multi-objective constraint optimization problem, i.e., a minimization problem for the workload difference of any two edge servers, and a minimization problem for access delay given by

$$T(l) = \underset{i,j \in S}{Min\ Max}(T_i - T_j), \tag{1}$$

$$D(l) = \underset{s \in S}{Min}\ \underset{b \in E_s}{Max}\, d(l_b, l_s), \tag{2}$$

where $l \in L$ is an edge server placement scheme that contains K edge server locations, and L represents all possible edge server placement schemes. $T(l)$ represents the workload balancing among edge servers and $D(l)$ is the access delay of edge servers. We consider the following two constraints:

- Assuming that all edge servers are placed, there must be no common base station for each edge server, and all base stations must be in control of edge servers, i.e.,

$$E_i \cap E_j = \varnothing, \tag{3}$$

$$\bigcup_{s \in S} E_s = B. \tag{4}$$

- Each base station will be co-located with an edge server, and the edge server will process all mobile user requests from the base station, i.e.,

$$T_s = \sum_{b \in E_s} t_b \ . \tag{5}$$

We assume that the access delay between a base station at location l_b and an edge server at location l_s is defined as $d(l_b, l_s)$, i.e., the distance between them.

In essence, the problem is to identify K locations from multiple potential locations, and place the K edge servers. Further, we need to determine at which location each edge server s_i should be placed such that the workloads among edge servers can be balanced and a minimum access delay is realized. The MIP has been used to solve server or cloudlet placement problems [28-30]. Inspired by MIP, we propose an approach to solve the optimization problem for edge server placement, which is presented in more detail later.

Lemma 1. The edge server placement problem in a mobile edge computing network $G = (V, E)$ is NP-hard.

Proof. We reduce the metric *K*-median problem to the edge server placement problem as follows. Consider the metric *K*-median problem in a given metric complete graph $G' = (V', E')$. We construct a mobile edge computing network $G = (V, E)$ from G', where $V = V'$ and $E = E'$. Now, we consider placing K edge servers into G. We see that an optimal solution to the edge server placement problem in G is an optimal solution to the metric *K*-median problem in G'. Because the metric *K*-median problem is NP-hard [31], the edge server placement problem is also NP-hard.

In this section, we adopt the weighting method to transform the edge server placement problem into a signal-objective optimization problem with an (Pareto) optimal solution.

As shown in (1–5), our edge server placement problem can be described as follows:

$$\begin{cases} find \quad l = (l_{s_1}, l_{s_2}, ..., l_{s_K})^T \\ which \quad \min T(l), \min D(l) \\ subject \quad to \quad E_i \cap E_j = \varnothing, \ \bigcup_{s \in S} E_s = B, \\ \qquad\qquad T_s = \sum_{b \in E_s} t_b \end{cases} \tag{6}$$

where l is an m-dimensional vector of decision variables, $T(l)$, $D(l)$ represents functions that are defined for two constraints L. Then, we can transfer (1–5) into (6) as the same multi-objective optimization problem, i.e., Problem 1.

Definition 1. $\hat{l} \in L$ is said to be a Pareto optimal solution of Problem 1 if and only if there exists no other $l \in L$ such that $T(l) \le T(\hat{l})$, $D(l) \le D(\hat{l})$.

Definition 2. $l^* \in L$ is said to be a weakly Pareto optimal solution of Problem 1 if and only if there exists no other $l \in L$ such that $T(l) \le T(l^*)$, $D(l) \le D(l^*)$.

In order to obtain the Pareto optimal solution or weakly Pareto optimal solution of Problem 1, we modify the multi-objective optimization problem into a single-objective optimization $w_1+w_1=1$ problem using the weighting method.

In this way, we assume that the weighting coefficients w_1 and w_2 are real numbers such that $w_1, w_2 \ge 0$ $w_1, w_2 \ge 0$. Usually, we also assume that the weights are normalized. Problem 1 is transformed into the following single-objective optimization problem, i.e., Problem 2:

$$\begin{cases} find \quad l = (l_{s_1}, l_{s_2}, ..., l_{s_K}) \\ which \quad \min(w_1 \cdot T(l) + w_2 \cdot D(l)) \\ subject \quad to \quad E_i \cap E_j = \varnothing, \ \bigcup_{s \in S} E_s = B, \\ \qquad\qquad T_s = \sum_{b \in E_s} t_b \end{cases} \tag{7}$$

where $w_1, w_2 \ge 0$ and $w_1+w_1=1$.

Theorem 1. The solution of Problem 2 is a weakly Pareto optimal solution of Problem 1.

Proof: Let $\hat{l} \in L$ be a solution of Problem 2. Let us assume that it is not a weakly Pareto optimal solution of Problem 1. In this case, there exists a solution $l \in L$ such that $T(l) \le T(\hat{l})$, $D(l) \le D(\hat{l})$. According to the assumptions, we set the weighting coefficients, $w_1, w_2 \ge 0$ for at least one that is larger than zero. Thus, we have $w_1 \cdot T(l) + w_2 \cdot D(l) < w_1 \cdot T(\hat{l}) + w_2 \cdot D(\hat{l})$. This is a contradiction of the assumption that $\hat{l}$ is a solution of Problem 2. That is, $\hat{l}$ is a weakly Pareto optimal solution of Problem 1.

Theorem 2. The solution of Problem 2 is Pareto optimal if the weighting coefficients are positive, i.e., $w_1, w_2 > 0$.

Proof: Let $l^* \in L$ be a solution of Problem 2 with positive weighting coefficients. Let us assume that is not Pareto optimal. This means that there exists a solution $l \in L$ such that $T(l) \le T(l^*)$, $D(l) \le D(l^*)$ and $T(l) < T(\hat{l})$, $D(l) < D(\hat{l})$ for at least one. Because $w_1, w_2 < 0$, we have $w_1 \cdot T(l) + w_2 \cdot D(l) < w_1 \cdot T(l^*) + w_2 \cdot D(l^*)$. This contradicts the assumption that l^* is a solution of Problem 2. That is, l^* must be Pareto optimal.

3.4 Finding an Optimal Solution

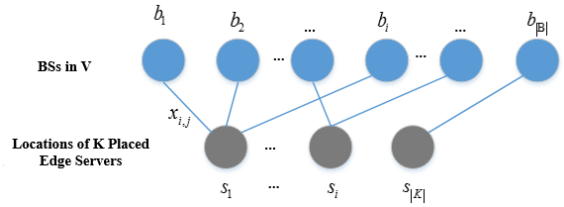


Fig. 2. Assignment of base stations to edge servers.

In this section, we present details of the process employed to utilize MIP to place edge servers. Fig. 2 shows the assignment of base stations to edge servers. We used a binary decision variable $x_{i,j} \in \{0,1\}$ to indicate whether a base station b_i will be assigned to an edge server s_j, where $x_{i,j}=1$ if base station b_i is assigned to s_j; otherwise, $x_{i,j}=0$ for all *i* and *j* with $1 \le i \le |B|$, $|B|$ is the total number of base stations and $1 \le j \le K$.

With this approach, we assume that each edge server is placed along with one of the base stations. Because the number of base stations is $|B|$, the number of available locations at which edge servers can be placed is $|B|$. For each assignment, we give a mark $c_{i,j}$ to measure whether it is suitable for an edge server. This mark considers both the distance and workload.

For the distance, we prefer a smaller distance from all base stations to their assigned edge servers.

For the workload, we prefer that the workload of each edge server be more balanced, and we evaluated this objective using the standard deviation. Below, we introduce the procedure for applying MIP to this problem.

Then, we should obtain a variable for each assignment, i.e., $x_{i,j}$. For each $x_{i,j}$, there exist two indices, i.e., the distance between base station b_i and edge server s_j (denoted by symbol $d_{i,j}$) and the workload of base station b_i (denoted by symbol t_i). To simplify the subsequent processing, we should normalize these two indices. The normalizing methods are as follows:

- For the distance, it is conventional to consider that the edge server should be placed in the denser area. Therefore, for each base station b_i, we find N ($N = \frac{|B|}{K}$) base stations nearest to b_i as a set $B_{nearb_i} = \{b_{i_{near1}}, b_{i_{near2}}, ..., b_{i_{nearN}}\}$, and we select the farthest one from B_{nearb}, the farthest distance value as d_i. For each d_i, we compute $\overline{d_i} = \frac{d_i\text{-}Mind}{Maxd - Mind}$ where $Mind$ and $Maxd$ respectively represent the minimal and maximal d_i for all base stations. The value of $\overline{d_{i,j}}$ is within the range $[0,1]$;
- For the workload, we used the sum of the squared difference to balance the probability of each placement for the edge server. First, we find the average workload of each edge server based on all base stations $\overline{W} = \frac{\sum_{i=1}^{|B|} t_i}{K}$. Then, we calculate the sum of the squared difference as $w_i = \left(\sum_{i=1}^{N} t_i - \overline{W}\right)^2$. Finally, we normalize each w_i using $\overline{w_i} = \frac{w_i - Minw}{Maxw - Minw}$ wher $Minw$ and $Maxw$ respectively represent the minimal and maximal w_i. We can see that the value of $\overline{w_i}$ is within the range $[0,1]$.

Then, we used the weighted sum of the normalized distance and workload to transform the two indices into a single index. We denote that single index as $c_{i,j}$, which is obtained by $c_{i,j} = \mu\overline{d}_{i,j} + (1-\mu)\overline{w_i}$. Here, the variable μ represents the weight, and we define $u = 0.5$, and it is in the range $(0,1)$.

The MIP model is as follows:

$$Min \sum_{i=1}^{|B|} \sum_{j}^{K} c_{i,j} x_{i,j} \tag{8}$$

subject to the following constraints:

$$\sum_{j=1}^{K} x_{i,j} = 1, \tag{9}$$

$$x_{i,j} \in \{0,1\}, \tag{10}$$

where $\sum_{j=1}^{K} x_{i,j} = 1$ ensures that each base station should be assigned to one and only one edge server.

4 PERFORMANCE EVALUATION

In this section, we implement our proposed approach to verify the performance, and we evaluate our proposed approach compared with several representative placement approaches in terms of workload balancing, access delay under various edge server workloads, and the placement of different numbers of edge servers. Results of extensive experimental evaluations show that our proposed approach is both effective and efficient.

4.1 Experiment Setup

To validate our proposed approach, we adopted a real base station dataset, and we implemented the experiments

using Python 3.5 and IBM cplex tool[2] with source code[3]. Our experiments primarily consisted of three parts: 1) we compared our approach with other known placement approaches; 2) we determined the number of edge servers K that are placed using our approach; and 3) we investigated the parameters of our approach.

4.2 Dataset Description

In our experiments, we utilized the dataset for Shanghai Telecom's base stations, which contains internet information for mobile users who access 3233 base stations. According to our analysis, there are 3000 effective base stations because some of them are idle, while for others, the data are invalid. The data contain the detailed start time and end time of base station access for each mobile user. Shanghai is a typical densely populated city, so it can meet the requirement of mobile edge computing network perfectly. Fig. 3 shows clearly the distribution of base stations included in Shanghai Telecom's base station dataset.

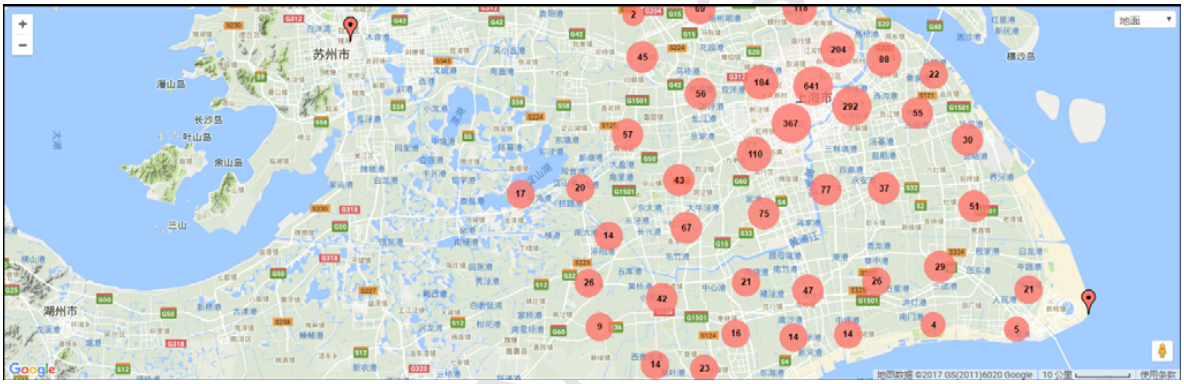


Fig. 3. Distribution of base stations included in Shanghai Telecom's base station dataset.

Fig. 3 shows the distribution of 3233 base stations. The number represents the number of base stations within the range of the red circles. Fig. 3 illustrates that in Shanghai, the base station distribution is very dense.

Table 2. Workload of some base stations included in Shanghai Telecom's base station dataset.

Base Station ID	40	76	328	531	664	1039	1265	1927	2160	2513	2748
User Number	1824	6	108	81	151	254	499	821	3	162	74
Workload (min)	2571744	2830	140960	109145	190703	321462	624866	1042672	4226	205734	98623

For each base station b in B, let t_b represent the number of mobile user requests that use base stations to access edge servers in the mobile edge computing network, which is a positive integer, i.e., the workload of base station b. While base stations are often deployed at locations such as schools, shopping malls, and train stations, the number of mobile user requests at base station b can be estimated by the population density in that area, or the historical base station access information using a linear regression technique. From Shanghai Telecom's base station dataset, we let the number of mobile user requests that access each base station represent that base station's workload. In Table 2, we describe a subset of the base station information obtained from Shanghai Telecom's base station dataset.

From the dataset, we randomly selected some base stations, and Table 2 illustrates the workload of these base stations as an example. According to the start time and end time of requests by mobile users that access base stations, we can calculate the total request time as the base station workload.

From Table 2, we find that there is a large unbalanced load between base stations, i.e., some of the base stations are

[2] https://www.ibm.com/us-en/marketplace/ibm-ilog-cplex

[3] https://github.com/cbozi/edge-server-placement

overloaded, while others are underutilized, and even idle. Therefore, the strategic placement of edge servers to offload the workload of some overloaded base stations should be urgency performed to optimize the performance of mobile applications.

4.3 Comparison of Approaches and Evaluation Metric

We compared the performance of our proposed approach with other placement approaches in terms of both the workload balance and access delay as follows:

1) **K-means**. This approach is commonly used to automatically cluster a data set into *K* groups [32]. With this approach, *K* initial cluster centers are selected and then iteratively refined. We use K-means clustering algorithms to find *K* clusters of base stations, then place *K* edge servers into their centers to minimize the within-cluster sum of squares [33].
2) **Top-K**. This approach places the *K* edge servers at the Top-K base stations, where a base station is a Top-K base station if the number of mobile user requests that it receives is among the Top-K values. The rationale behind this approach is to place edge servers at the *K* busiest base stations that have more mobile user requests than others.
3) **Random**. This approach places the *K* edge servers at base stations in a random manner.

Our approach is different from the compared approaches that formulates the edge server placement model and then adopts the mixed integer programming to find the optimal solution.

Definition 3 (Workload Balancing). We used the standard deviation to evaluate the workload balancing of edge servers. We assume that the *K* edge servers are placed among the base stations, and we then calculate the workload of each edge server *i* as T_i, and the standard deviation of the workload can be computed as follows:

$$WB = \sqrt{\frac{\sum_{i=1}^{K}(T_i - \overline{T})^2}{K}}, \qquad (11)$$

where $\overline{T}$ represents the average value of workloads for all edge servers. The smaller the value of the standard deviation, the more balanced is the workload of each edge server.

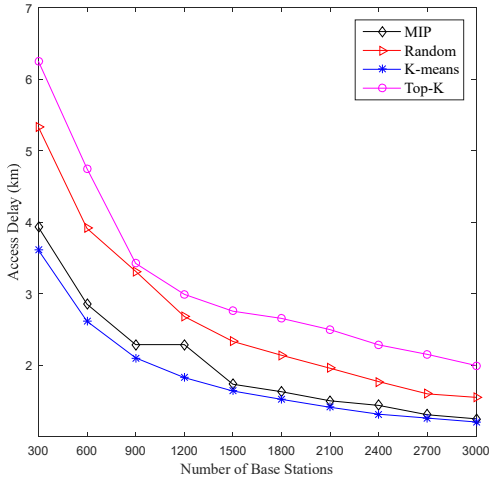
Definition 4 (Access Delay). We used the access delay between base stations and edge server to evaluate the performance for all approaches. In our experiments, we let the average distance between the base station and edge server represent the access delay.

4.4 Comparison of Results using Number of Base Stations

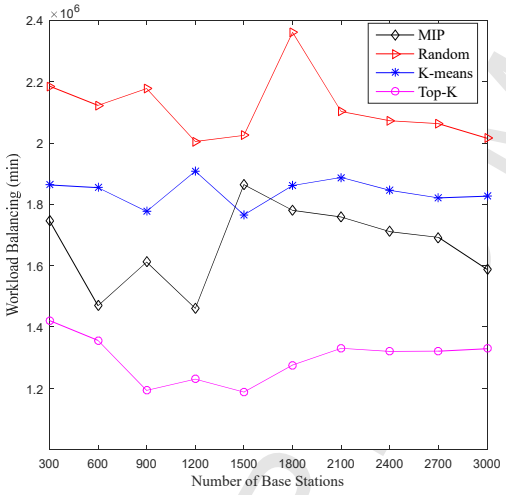
We evaluated the performance of different approaches using Shanghai Telecom's base station dataset. To do this, we increased the number of base stations *n* from 300 to 3000, while setting the ratio of the number of edge servers *K* to the number of base stations to 0.1, i.e., $R = K/n = 0.1$. Fig. 4 shows the performance evaluation curves for different approaches as the number of base stations increases.

From Fig. 4, we can see that our proposed approach generally outperforms the K-means, Top-K, and Random approaches. Fig. 4(a) and Fig. 4(b) show plots of the edge server access delay and workload balancing, respectively. With respect to the access delay, the performance ranking is K-means > MIP > Random > Top-K; in terms of workload balancing, which is a parameter whose value is used to evaluate the workload balancing, the performance ranking is Top-K > MIP > K-means > Random. Overall, our approach is more effective than other approaches in terms of both access delay and workload balancing.

Note that the Random approach outperforms the Top-K approach in terms of the access delay. This is an interesting result in our experiment based on the Shanghai Telecom's base station dataset. Because Shanghai is a large and densely populated city, and the distribution of base station is dense, as shown in Fig. 3, if we use the Top-K approach to place the edge servers, it may result in a poorer performance than the Random approach.



(a) Edge server access delay.



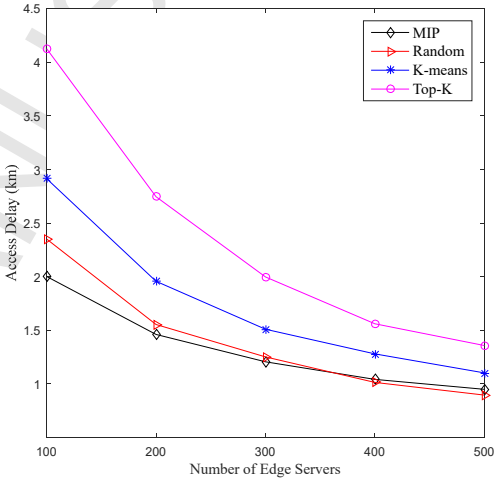
(b) Edge server workload balancing.

Fig. 4. Performance evaluation of different approaches as the number of base stations increases.

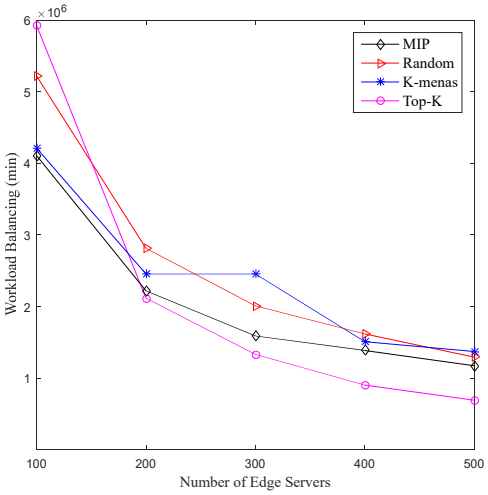
4.5 Comparison of Results with Number of Edge Servers

We used different approaches to study the impact of the number of edge servers K on the performance of the edge server in terms of the access delay and workload balancing. We performed experiments using data obtained for 3000 base stations by varying K from 100 to 500.

Fig. 5(a) and 5(b) illustrate the curves of the edge server access delay and workload balancing as the number of edge server K increases. From Fig. 5(a), we can see that the edge server access delay obtained by each mentioned algorithm decreases as K increases because the workload of each base station will have more chances to be assigned to its nearest edge server. We can also see that the MIP approach outperforms the other approaches as K increases. From Fig. 5, we can find an appropriate K subject to some constraints.



(a) Edge server access delay



(b) Edge server workload balancing

Fig. 5. Impact of the number of edge servers K on the performance of difference approaches.

4.6 Study of Parameter *R*

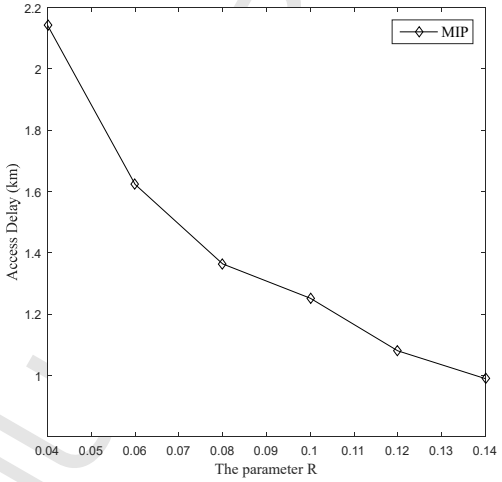
In our approach, we studied the parameter R by increasing it from 0.04 to 0.14. We let the parameter R to represent the ratio of the number of edge servers to the number of base stations. If $R = 0.1$ and $n = 3000$, then $K = 300$, i.e., an edge server processes all requests for 10 base stations.

Fig. 6 plots the curves of the performance of our MIP algorithm as the parameter R increased. We see that the edge server access delay decreases with increasing R because the workload of each base station will have more chances to be assigned to its nearest edge server. Similarly, the workload between edge servers is more balanced because the workloads of base stations will be assigned to more edge servers and the difference becomes smaller.

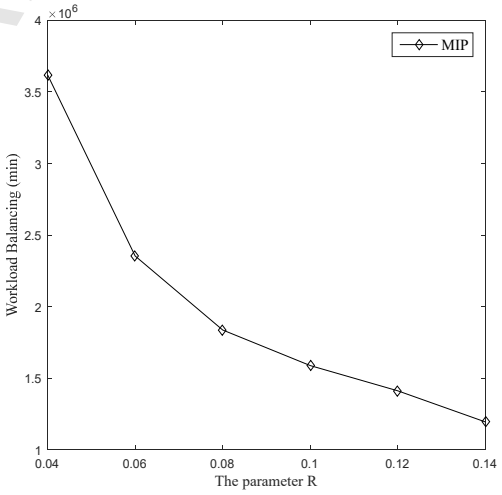
For smart cities, e.g., Shanghai, we need to determine how to select the optimal edge server placement scheme. To do this, we need to consider the following factors:

1) Availability of funds for infrastructural development. This determines the number of edge servers that can be placed. As the number of edge servers increases, so will the performance. As expected, a higher number of edge servers also corresponds to a higher cost.
2) Determining the best access delay that is within the acceptable range. We used the distance between each base station and edge server as well as the data rates to find the access delay.

Considering the above factors and according to the performance of our proposed approach, the manager of smart cities can determine the value of the parameter R by considering the access delay and workload balance.



(a) Edge server access delay



(b) Edge server workload balancing

Fig. 6. Variation in the MIP performance as the parameter R increases.

5 CONCLUSION

Edge computing is an important emerging technology that can be used to extend the computation and storage capabilities by offloading processing workload from the cloud in order to reduce mobile edge computing network latency on mobile devices. In this study, we first investigate the edge server placement problem in a large-scale mobile

edge computing environment, with the objective of balancing the workload between edge servers and minimizing the edge server access delay. Then, we formulate the problem as a multi-objective constraint optimization problem and propose an MIP edge server placement algorithm to find the optimal solution. Finally, we performed experiments to evaluate the performance of the proposed approach in a real mobile edge computing network dataset obtained from Shanghai Telecom, and we compare the results with those obtained using other approaches. The experimental results show that our proposed approach outperforms several representative approaches in terms of the access delay and workload balancing in mobile edge computing.

This paper assumes the edge servers are homogeneous, but they are heterogeneous in real system deployment. Moreover, the computation power of the edge servers are different. Hence, our future work will focus on the heterogeneous and green edge server placement problem.

Acknowledgments

This work was supported in part by the National Science Foundation of China (Grant No. 61472047).

REFERENCES

[1] E. Ahmed, A. Gani, M. Sookhak, S.H. Ab Hamid, F. Xia, Application optimization in mobile cloud computing: Motivation, taxonomies, and open challenges, Journal of Network and Computer Applications, 52 (2015) 52-68.

[2] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, H. Flinck, Mobile edge computing potential in making cities smarter, IEEE Communications Magazine, 55 (2017) 38-43.

[3] H.T. Dinh, C. Lee, D. Niyato, P. Wang, A survey of mobile cloud computing: architecture, applications, and approaches, Wireless communications and mobile computing, 13 (2013) 1587-1611.

[4] E. Ahmed, A. Akhunzada, M. Whaiduzzaman, A. Gani, S.H. Ab Hamid, R. Buyya, Network-centric performance analysis of runtime application migration in mobile cloud computing, Simulation Modelling Practice and Theory, 50 (2015) 42-56.

[5] J. Liu, E. Ahmed, M. Shiraz, A. Gani, R. Buyya, A. Qureshi, Application partitioning algorithms in mobile cloud computing: Taxonomy, review and future directions, Journal of Network and Computer Applications, 48 (2015) 99-117.

[6] A.E.C. Cloud, Amazon web services, Retrieved November, 9 (2011) 2011.

[7] M. Cusumano, Cloud computing and SaaS as new computing platforms, Communications of the ACM, 53 (2010) 27-29.

[8] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The Case for VM-Based Cloudlets in Mobile Computing, IEEE Pervasive Computing, 8 (2009) 14-23.

[9] P. Mach, Z. Becvar, Mobile Edge Computing: A Survey on Architecture and Computation Offloading, IEEE Communications Surveys & Tutorials, (2017).

[10] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: Vision and challenges, IEEE Internet of Things Journal, 3 (2016) 637-646.

[11] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, Mobile-edge computing introductory technical white paper, White Paper, Mobile-edge Computing industry initiative, (2014).

[12] A. Ahmed, E. Ahmed, A survey on mobile edge computing, in: 2016 Proceedings of the International Conference on Intelligent Systems and Control, pp. 1-8.

[13] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti, Clonecloud: elastic execution between mobile device and cloud, in: 2011 Proceedings of the 6th conference on Computer systems, ACM, pp. 301-314.

[14] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, P. Bahl, MAUI: making smartphones last longer with code offload, in: 2010 Proceedings of the 8th international conference on Mobile systems, applications, and services, ACM, pp. 49-62.

[15] R. Kemp, N. Palmer, T. Kielmann, H. Bal, Cuckoo: a computation offloading framework for smartphones, in: 2010 Proceedings of the International Conference on Mobile Computing, Applications, and Services, Springer, pp. 59-79.

[16] S. Kosta, A. Aucinas, P. Hui, R. Mortier, X. Zhang, Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading, in: 2012 Proceedings of the IEEE Infocom, pp. 945-953.

[17] W. Stallings, Local and metropolitan area networks, Macmillan Publishing Co., Inc., 1993.

[18] M. Jia, J. Cao, W. Liang, Optimal Cloudlet Placement and User to Cloudlet Allocation in Wireless Metropolitan Area Networks, IEEE Transactions on Cloud Computing, PP (2015).

[19] Z. Xu, W. Liang, W. Xu, M. Jia, S. Guo, Efficient Algorithms for Capacitated Cloudlet Placements, IEEE Transactions on Parallel and Distributed Systems, 27 (2016) 2866-2880.

[20] H. Xiang, X. Xu, H. Zheng, S. Li, T. Wu, W. Dou, S. Yu, An Adaptive Cloudlet Placement Method for Mobile Applications over GPS Big Data, in: 2016 Proceedings of the IEEE Global Communications Conference, pp. 1-6.

[21] A. Wolbach, J. Harkes, S. Chellappa, M. Satyanarayanan, Transient customization of mobile computing infrastructure, in: 2008 Proceedings of the First Workshop on Virtualization in Mobile Computing, ACM, pp. 37-41.

[22] S. Clinch, J. Harkes, A. Friday, N. Davies, M. Satyanarayanan, How close is close enough? Understanding the role of cloudlets in supporting display appropriation by mobile users, in: 2012 Proceedings of the Pervasive Computing and Communications, pp. 122-127.

[23] G. Lewis, S. Echeverría, S. Simanta, B. Bradshaw, J. Root, Tactical cloudlets: Moving cloud computing to the edge, in: 2014 Proceedings of the Military Communications Conference, pp. 1440-1446.

[24] X. Chen, L. Jiao, W. Li, X. Fu, Efficient multi-user computation offloading for mobile-edge cloud computing, IEEE/ACM Transactions on Networking, 24 (2016) 2795-2808.

[25] M. Tao, K. Ota, M. Dong, Foud: Integrating Fog and Cloud for 5G-Enabled V2G Networks. IEEE Network, 31(2017): 8-13.

[26] H. Li, M. Dong, K. Ota, M. Guo, Pricing and Repurchasing for Big Data Processing in Multi-Clouds. IEEE Transactions on Emerging Topics in Computing, 4 (2016): 266-277.

[27] H. Li, M. Dong, X. Liao, H. Jin, Deduplication-Based Energy Efficient Storage System in Cloud Environment. The Computer Journal, 58(2015): 1373-1383.

[28] C.C.T. Mark, D. Niyato, T. Chen-Khong, Evolutionary optimal virtual machine placement and demand forecaster for cloud computing, in: 2011 Proceedings of the Advanced Information Networking and Applications, pp. 348-355.

[29] A. Ceselli, M. Premoli, S. Secci, Cloudlet network design optimization, in: 2015 Proceedings of the Interbational Federation for Information Processing Networking Conference, pp. 1-9.

[30] S.E. Dashti, A.M. Rahmani, Dynamic VMs placement for energy efficiency by PSO in cloud computing, Journal of Experimental & Theoretical Artificial Intelligence, 28 (2016) 97-112.

[31] M. Charikar, S. Guha, É. Tardos, D.B. Shmoys, A constant-factor approximation algorithm for the k-median problem, in: 1999 Proceedings of the 31th ACM symposium on Theory of computing, ACM, pp. 1-10.

[32] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, Constrained k-means clustering with background knowledge, in: 2001 Proceedings of the International Conference on Machine Learning, pp. 577-584.

[33] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: 1967 Proceedings of the 5th Berkeley symposium on mathematical statistics and probability, pp. 281-297.

Shangguang Wang is an associate professor at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT). He received his Ph.D. degree at BUPT in 2011. He is Vice Chair of IEEE Computer Society Technical Com- mittee on Services Computing, President of the Service Society Young Scientist Forum in China and served as General Chair of Collaborate- Com 2016, General Chair of ICCSA 2016, TPC Chair of IOV 2014, and TPC Chair of SC2 2014.

Yali Zhao received bachelor's degree in computer science and technology from Shandong University, in 2013. Currently, she is a Master Degree Candidate at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. Her research interests include service computing and edge computing.

Jinliang Xu received the bachelor's degree in electronic information science and technology from Beijing University of Posts and Telecommunications in 2014. Currently, he is a Ph.D. candidate in computer science at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His research interests include Service Computing, Information Retrieval, and Crowdsourcing.

Ching-Hsien Hsu is a professor and the chairman in the CSIE department at Chung Hua University, Taiwan; He was distinguished chair professor at Tianjin University of Technology, China, during 2012-2016. His research includes high performance

computing, cloud computing, parallel and distributed systems, big data analytics, ubiquitous/pervasive computing and intelligence. He has published 100 papers in top journals such as IEEE TPDS, IEEE TSC, IEEE TCC, IEEE TETC, IEEE System, IEEE Network, ACM TOMM and book chapters in these areas. Dr. Hsu is serving as editorial board for a number of prestigious journals, including IEEE TSC, IEEE TCC. He has been acting as an author/co-author or an editor/co-editor of 10 books from Elsevier, Springer, IGI Global, World Scientific and McGraw-Hill. Dr. Hsu was awarded nine times distinguished award for excellence in research from Chung Hua University. He is vice chair of IEEE TCCLD, executive committee of IEEE TCSC, Taiwan Association of Cloud Computing and an IEEE senior member.