On Cost Aware Cloudlet Placement for Mobile Edge Computing

Qiang Fan, *Student Member, IEEE* and Nirwan Ansari, *Fellow, IEEE*

***Abstract*— As accessing computing resources from the remote cloud inherently incurs high end-to-end (E2E) delay for mobile users, cloudlets, which are deployed at the edge of a network, can potentially mitigate this problem. Although some research works focus on allocating workloads among cloudlets, the cloudlet placement aiming to minimize the deployment cost (i.e., consisting of both the cloudlet cost and average E2E delay cost) has not been addressed effectively so far. The locations and number of cloudlets have a crucial impact on both the cloudlet cost in the network and average E2E delay of users. Therefore, in this paper, we propose the Cost Aware cloudlet PlAcement in moBiLe Edge computing (CAPABLE) strategy, where both the cloudlet cost and average E2E delay are considered in the cloudlet placement. To solve this problem, a Lagrangian heuristic algorithm is developed to achieve the suboptimal solution. After cloudlets are placed in the network, we also design a workload allocation scheme to minimize the E2E delay between users and their cloudlets by considering the user mobility. The performance of CAPABLE has been validated by extensive simulations.**

***Index Terms*—Cloudlet placement, mobile cloud computing, mobile edge computing.**

I. Introduction

RECENT mobile applications, such as augmented reality, on-line gaming, and image processing, are computation-intensive while the resource of battery powered mobile devices remains limited. Mobile Cloud Computing (MCC) [1] is introduced to offload user tasks to a centralized data center via Internet and thus reduces the task execution time and energy consumption of users. However, the cloud is usually remotely located and far away from its users, and thus inherently incurs a long end-to-end (E2E) delay between a user and the cloud. Although this E2E delay may meet the demands of some applications such as web browsing, it is unbearable for many delay sensitive applications such as augmented reality and on-line gaming [2]–[4]. Hence, the concept of cloudlets is employed to reduce the user E2E delay by moving the remote cloud resources to the network edge [5]–[7]. Since cloudlets, which are tiny versions of data centers, are generally placed at access points in the network that are close to users, users can access the computing resources with a lower E2E delay [8].

Although the cloudlet concept is a promising technique to reduce user E2E delay, how to place cloudlets to minimize the E2E delay as well as the cloudlet cost in the network has not been addressed. The budget of a cloudlet provider is always limited. The cloudlet cost of a cloudlet mainly comes from renting a site and installing a certain number of servers, which indicate the capacity of the cloudlet. As site rentals are geographically dynamic, and thus the location of a cloudlet poses a significant impact on the cloudlet cost. Meanwhile, once the location of a cloudlet is decided, the cloudlet provider still needs to determine how many servers (i.e., the amount of computing resources) to be installed in the cloudlet, according to the user density (i.e., workload density) near the cloudlet. Thus, the cloudlet cost of a cloudlet provider depends on the locations and quantity of cloudlets and their servers. In addition, when placing the cloudlets, the cloudlet provider should ensure a low E2E delay between users and their cloudlets to improve the quality of experience (QoE) of MCC applications. It can be observed that users can achieve lower E2E delay from cloudlets in their physical proximity than from cloudlets far away. The E2E delay of users is determined by the location and quantity of cloudlets and their servers. On one hand, if cloudlets with high capacities are deployed at some strategic positions (i.e., regions with high user density), more users are able to access the computing resources in their proximity, thus reducing the total E2E delay in the network. On the other hand, the number of cloudlets also affects the total E2E delay of users. If more cloudlets are placed in the network, each user is more likely to access to a closer cloudlet, thus incurring the lower E2E delay between the user and its cloudlet. In the extreme case, when cloudlets are placed at every base station (BS), most of users can be served by their local cloudlets, which are attached to their BSs, and the cloudlet network can thus provision the lowest E2E delay between users and their cloudlets.

A cloudlet provider aims to minimize the cloudlet cost while improving the quality of experience (QoE) for its users, in terms of the E2E delay between users and their cloudlets. In this case, only optimizing the cloudlet cost or E2E delay cannot meet the cloudlet provider's objective. Therefore, we propose the Cost Aware cloudlet PlAcement in moBiLe Edge computing (CAPABLE) strategy to optimize the tradeoff between the cloudlet cost and average E2E delay between users and their cloudlets. Below are major contributions of the paper.

- The proposed CAPABLE strategy adopts a tradeoff coefficient η to balance the weight of the cloudlet cost and

Manuscript received December 7, 2018; revised February 2, 2019; accepted March 27, 2019. This work was supported in part by the National Science Foundation (CNS-1647170). Recommended by Associate Editor MengChu Zhou. *(Corresponding author: Qiang Fan.)*

Citation: Q. Fan and N. Ansari, "On cost aware cloudlet placement for mobile edge computing," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 4, pp. 926–937, Jul. 2019.

Q. Fan and N. Ansari are with Advanced Networking Lab, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, USA (e-mail: qf4@njit.edu; nirwan.ansari@njit.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2019.1911564

average E2E delay in the cloudlet placement. If η is large, the strategy tends to place cloudlets at the sites with low rentals and deploy as few servers as possible. In contrast, if η is small, the proposed strategy focuses on the average E2E delay, and thus tends to place a large number of cloudlets and servers in the network.

- We have proved that the cloudlet placement is an NP-hard problem. Hence, we design a Lagrangian heuristic algorithm to obtain the sub-optimal solution.
- After cloudlets are placed in the network, with consideration of user mobility in the network, we design a dynamic workload allocation scheme to distribute users' virtual machines among cloudlets in each time slot to minimize the E2E delay between the users and their cloudlets.

The remainder of this paper is organized as follows. In Section II, we briefly review related works. In Section III, we illustrate the cloudlet network architecture and describe the system model. In Section IV, we formulate and analyze the cloudlet placement problem. In Section V, the CAPABLE algorithm is proposed to obtain the suboptimal solution of the cloudlet placement problem. In Section VI, the dynamic Avatar (i.e., virtual machine) allocation scheme is proposed to minimize the E2E delay in different time slots by considering the user mobility. Section VII shows the simulation results, and concluding remarks are presented in Section VIII.

II. Related Works

As the cloud is usually physically far away from their users, offloading tasks from users to the remote cloud suffers from prohibitively long latency and increases the burden of the network [1]. This can be especially problematic for mobile applications where the response time is critical to their user experience, such as augmented reality applications and on-line games. Recently, many studies have proposed to utilize cloudlets deployed at the network edge to reduce the E2E delay between users and their computing resources, and thus improve the performance of MCC applications. Tawalbeh *et al.* [9] proposed a mobile cloud computing model to run the big data applications in cloudlets rather than remote clouds. Quwaider and Jararweh [10] proposed a cloudlet-based wireless body area network. The huge amount of data generated by the users are transmitted to a nearby cloudlet through WiFi. The nearby cloudlet stores and processes the data streams locally to reduce the latency as well as communications power consumption as compared to the traditional cloud-based wireless body area network. Satyanarayanan *et al.* [11] proposed the GigaSight architecture to perform the video processing in local cloudlets to reduce the latency while saving the bandwidth of core networks. Sun and Ansari [12] proposed a profit maximization virtual machines (VMs) placement strategy for mobile edge computing, referred to as PRIMAL, which makes a tradeoff between the E2E delay reduction and migration overheads by selectively migrating VMs to their optimal cloudlets. In addition, Sun *et al.* [13], [14] proposed a green cloudlet network architecture, where all cloudlets are powered by both green energy and on-grid energy. To minimize the on-grid energy consumption, users' designated VMs are migrated to cloudlets with excessive green energy while ensuring low E2E delay for users.

Although the research on cloudlets has recently received much attention, few has addressed the cloudlet placement problem, which poses a crucial impact on the E2E delay. Xu *et al.* [15], [16] formulated a capacitated cloudlet placement problem and placed K capacitated cloudlets to some strategic locations to minimize the average E2E delay between mobile users and their cloudlets. Jia *et al.* [17] proposed a model to place K cloudlets in the network and realize the load balancing among the cloudlets to minimize the response time of users. However, the above works only minimize the E2E delay by placing a certain number of cloudlets, without considering the cloudlet cost and E2E delay at the same time. Moreover, these works assume that the cloudlet capacity of each cloudlet is given before the cloudlet placement. In contrast, another work [18] determines the locations of cloudlets and their servers based on each BS's workload by jointly considering the cloudlet cost and average E2E delay. As compared to existing studies, this paper presents several enhancements. First, we have proposed the cloudlet network architecture, where each user is assigned a dedicated VM (i.e., an Avatar) to process its own data and applications. Second, we consider the average E2E delay of users as well as the cloudlet cost in the cloudlet placement and adopt a coefficient η to adjust their tradeoff relationship. Third, when placing the cloudlets, we not only choose strategic locations for cloudlets, but also determine the optimal number of cloudlets. Fourth, when the number and locations of cloudlets are decided, we also determine the optimal number of servers (i.e., the cloudlet capacity) for different cloudlets based on the geographical user density around them. Fifth, considering the user mobility, we have proposed the dynamic Avatar allocation scheme to optimize the locations of all the Avatars in each time slot in order to minimize the E2E delay for all users during the time slot.

III. System Model

A cloudlet network architecture is illustrated in Fig. 1, where cloudlets are collocated with some selected BSs. Meanwhile, the software defined network (SDN) based cellular network is employed as the cellular core network to provide efficient and flexible communications paths between BSs [19], [20]. Meanwhile, mobile providers offer seamless wireless communications between a user and its BS, and thus each user can access its BS and then connect to a nearby cloudlet. Based on the geographical distribution of users, the numbers of servers in cloudlets are different (i.e., the capacities of cloudlets are different). In the cloudlet network, each user is mapped to a specific Avatar in a cloudlet (i.e., a designated VM for the user), which only runs task requests offloaded from the corresponding user. An Avatar is a software clone of a user and always offers service to the user wherever it moves [14], [21]. We assume that every user's Avatar is homogeneous (i.e., the configurations of Avatars are the same) although the workloads of Avatars are different. Also, the computational resources of every server is the same, and thus each server is assumed to accommodate the same

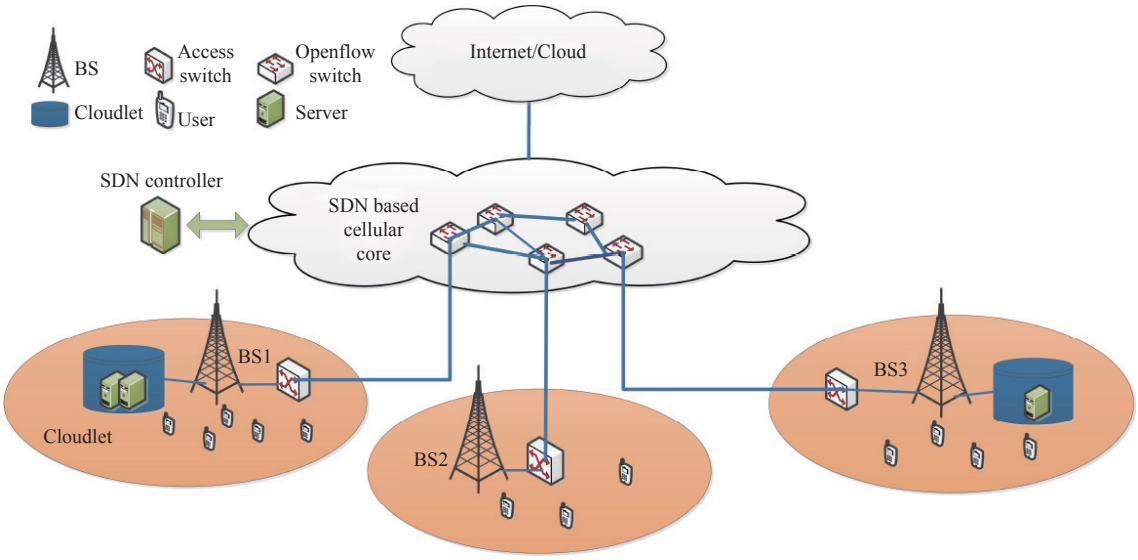


Fig. 1. Cloudlet network architecture.

number of Avatars.

Denote $\mathcal{I}$ as the set of potential sites of cloudlets; denote $\mathcal{K}$ as the set of BSs; denote $\mathcal{J}$ as the set of users. To indicate the locations of cloudlets in the network, a binary variable y_i is introduced to represent whether a cloudlet is placed at site i (i.e., $y_i = 1$) or not (i.e., $y_i = 0$). For brevity, we define cloudlet i as the cloudlet located at site i. Meanwhile, we denote z_i (i.e., a non-negative integer variable) as the number of servers installed in cloudlet i, and thus z_i should not exceed the maximum number of servers in cloudlet i, which is denoted as e. Note that if no cloudlet is deployed at site i (i.e., $y_i = 0$), z_i will also be zero. The key notations of this paper are summarized in Table I.

TABLE I
LIST OF SYMBOLS

Symbol	Definition
$\mathcal{I}$	Set of potential sites of cloudlets.
$\mathcal{K}$	Set of BSs.
$\mathcal{J}$	Set of mobile users.
x_{ij}	Binary variable of user j's Avatar being assigned to cloudlet i.
y_i	Binary variable of a cloudlet being placed at site i.
z_i	Number of servers installed in cloudlet i.
f_i	Fixed cost of a cloudlet at site i.
ξ_i	Price of a server at cloudlet i.
d_{ij}	Average E2E delay between user j and cloudlet i.
γ	Weight of the average E2E delay in the objective function.
η	Tradeoff coefficient.
p_{jk}	Occurrence probability of user j in the coverage of BS k.
μ_j	Lagrangian multipliers.
LB	Lower bound of the original problem.
UB	Upper bound of the original problem.
$\mathcal{J}_1$	Set of users whose Avatars are waiting to be allocated.
$\mathcal{I}_1$	Set of cloudlets that can still host at least one more Avatar.

A. Cloudlet Cost Model

When cloudlet providers decide to deploy a cloudlet at a BS, they have to rent a facility (site), whose cost depends on the geographical location, and then install the basic equipment. Thus, this part of the cloudlet cost is considered as the fixed cost, which is decided by the location of the cloudlet. In addition, cloudlet providers also need to install servers into a cloudlet to provide computing resources. Given the price of a server, the cost of servers in a cloudlet, which is considered as the dynamic cost, just depends on the number of servers in the cloudlet. Therefore, the cloudlet cost in the cloudlet network consists of the fixed cost and the dynamic cost, which can be expressed as

$$P_i = \sum_{i \in \mathcal{I}} f_i y_i + \sum_{i \in \mathcal{I}} \xi_i z_i \tag{1}$$

where f_i is the fixed cost of a cloudlet at site i and ξ_i is the price of a server at cloudlet i.

B. E2E Delay

When a request of a mobile user is sent to a cloudlet, the request goes through its BS and the SDN-based cellular core network. Therefore, the E2E delay between a user and its Avatar consists of two parts: first, the E2E delay between the user and its associated BS, i.e., the wireless delay; second, the E2E delay between the BS and the cloudlet that hosts the user's Avatar. However, changing the locations of cloudlets does not affect the wireless delay, which only depends on the user's service plan and the mobile provider's bandwidth allocation strategy [22]. Thus, we just consider the E2E delay between the BS and cloudlet in this paper (i.e., the E2E delay between a user and its Avatar is defined as the E2E delay between its BS and the cloudlet that hosts its Avatar). Denote τ_{ki} as the E2E delay between BS k and cloudlet i. The value of τ_{ki} can be measured and recorded by the SDN controller [23], [24].

As we know, mobile user movement often follows a repetitive pattern, i.e., a user usually commutes among several places such as home, workplace and gym for most of the time

of one day [17], [25]. In this case, we can estimate the occurrence probability of users in different BSs, based on the historical data of user movements [21]. Thus, the average E2E delay between user j and cloudlet i can be expressed as

$$d_{ij} = \sum_{k} p_{jk}\tau_{ki} \tag{2}$$

where p_{jk} is the occurrence probability of user j in the coverage of BS k.

Denote x_{ij} as a binary variable indicating whether user j's Avatar is located at cloudlet i (i.e., $x_{ij}=1$) or not (i.e., $x_{ij}=0$); denote d_j as the average E2E delay between user j and its Avatar; then, we have:

$$d_j = \sum_{i\in\mathcal{I}} x_{ij}d_{ij}. \tag{3}$$

IV. Problem Formulation and Analysis

A cloudlet provider, in placing cloudlets, has to consider two issues: cloudlet cost and average E2E delay [26]. On one hand, the average E2E delay between users and their Avatars should be minimized to improve the QoE of mobile applications. Hence, a cloudlet provider should place as many cloudlets as possible in the cloudlet network, thus enabling each user's Avatar to be allocated to a nearby cloudlet. If the number of cloudlets and their capacities are increasing, the average E2E delay of users will be reduced accordingly. We assume the maximum number of cloudlets in the network allowed by cloudlet providers is p. On the other hand, the cloudlet provider aims to minimize the capital expenditures (CAPEX) in deploying cloudlets, i.e., placing fewer cloudlets on suitable positions. Thus, only optimizing the cloudlet cost or the average E2E delay cannot meet the provider's requirement. Therefore, we design the Cost Aware cloudlet PlAcement in moBiLe Edge computing (CAPABLE) strategy to optimize the tradeoff between the cloudlet cost and the average E2E delay. In this paper, the deployment cost of a cloudlet network is defined as the sum of the cloudlet cost and the average E2E delay cost. Denote ρ as the deployment cost; we have

$$\rho = \sum_{i\in\mathcal{I}} f_i y_i + \sum_{i\in\mathcal{I}} \xi_i z_i + \gamma \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} x_{i,j}d_{i,j}. \tag{4}$$

Note that γ is the weight of the average E2E delay, and is modeled as follows:

$$\gamma = \frac{\sum_{i\in\mathcal{I}_{max}} f_i + \xi_i e}{\sum_j d_{Jj}} \times \frac{1-\eta}{\eta}. \tag{5}$$

Here, d_{Jj} indicates the largest average E2E delay for user j, where cloudlet J is the farthest cloudlet for user j. Furthermore, the maximum number of cloudlets in the network allowed by cloudlet providers is p; let $\mathcal{I}_{max}$ be the set of p sites with the highest rental costs in the network (i.e., $|\mathcal{I}_{max}| = p$). Thus, $\sum_{i\in\mathcal{I}_{max}} f_i + \xi_i e$ represents the maximum cloudlet cost of the network by deploying p cloudlets at sites in $\mathcal{I}_{max}$ and each cloudlet has the maximum number of servers (i.e., e). Meanwhile, $\sum_j d_{Jj}$ represents the largest average E2E delay of all users (i.e., when each user is served by its farthest cloudlet). In order to set different tradeoff relations between the cloudlet cost and average E2E delay, we introduce a tradeoff coefficient η, where $\eta \in (0,1)$. Increasing the value of η would reduce the weight ratio of the average E2E delay to the cloudlet cost, and encourage the cloudlet provider to place fewer cloudlets. Thus, η is used to adjust the tradeoff between the cloudlet cost and average E2E delay, and can be chosen between 0 and 1 via experiments based on the cloudlet provider's practical requirements.

The objective of the CAPABLE strategy is to minimize the deployment cost of a cloudlet network. Consequently, we formulate the problem as follows:

$$P1: \min_{x_{ij}y_iz_i} \sum_{i\in\mathcal{I}} f_i y_i + \sum_{i\in\mathcal{I}} \xi_i z_i + \gamma \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} x_{ij}d_{ij} \tag{6}$$

$$\text{s.t.} \sum_{i\in\mathcal{I}} x_{ij} = 1 \quad \forall j \in \mathcal{J} \tag{7}$$

$$\sum_{j\in\mathcal{J}} x_{ij} \le s z_i \quad \forall i \in \mathcal{I} \tag{8}$$

$$z_i \le e y_i \quad \forall i \in \mathcal{I} \tag{9}$$

$$\sum_{i\in\mathcal{I}} y_i \le p \quad \forall i \in \mathcal{I} \tag{10}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in \mathcal{I} \quad \forall j \in \mathcal{J} \tag{11}$$

$$y_i \in \{0,1\} \quad \forall i \in \mathcal{I} \tag{12}$$

$$z_i \ge 0 \ \text{ integer } \quad \forall i \in \mathcal{I}. \tag{13}$$

Here, s is the capacity of a server in terms of the number of Avatars, and e is the maximum number of servers in cloudlet i. Constraint (7) ensures that the Avatar of each user is assigned to only one cloudlet. Constraint (8) imposes the number of Avatars hosted by a cloudlet not to exceed the maximum number of Avatars in the cloudlet. Constraint (9) imposes the number of servers in a cloudlet to be no more than the maximum number of servers in the cloudlet. Constraint (10) imposes the number of cloudlets not to exceed the maximum number of cloudlets in the network.

Lemma 1: The cloudlet placement problem (i.e., P1) is an NP-hard problem.

Proof: Suppose the price of a server (i.e., ξ_i) is zero; p equals the total number of BSs in the network. Therefore, P1 can be transformed into:

$$R1: \min_{x_{ij}y_iz_i} \sum_{i\in\mathcal{I}} f_i y_i + \gamma \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} x_{i,j}d_{ij} \tag{14}$$

$$\text{s.t.} \sum_{i\in\mathcal{I}} x_{ij} = 1 \quad \forall j \in \mathcal{J} \tag{15}$$

$$\sum_{j\in\mathcal{J}} x_{ij} \le s e y_i \quad \forall i \in \mathcal{I} \tag{16}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in \mathcal{I} \quad \forall j \in \mathcal{J} \tag{17}$$

$$y_i \in \{0,1\} \quad \forall i \in \mathcal{I}. \tag{18}$$

Obviously, the above problem R1 is a capacitated facility location problem, which is a well known NP-hard problem [27]. Thus, the capacitated facility location problem is reducible to the problem P1, i.e., P1 is NP-hard. ■

V. CAPABLE Algorithm

Some studies have shown that Lagrangian relaxation is an efficient algorithm to solve the capacitated facility location problem [28], [29]. Thus, we design the CAPABLE algorithm (i.e., a Lagrangian heuristic algorithm) to place cloudlets to suitable sites. The basic idea of the algorithm is to iteratively improve the lower bound (LB) and upper bound (UB) of the original problem by the subgradient method. In each iteration, given a sequence of Lagrangian multipliers, the LB can be obtained by solving a Lagrangian relaxation problem. Based on the solution of the Lagrangian relaxation problem, the UB can be obtained by a heuristic algorithm. Then, the subgradient method is used to adjust the Lagrangian multipliers based on the UB and LB. When the Lagrangian multipliers are updated in each iteration, the gap between the UB and LB becomes smaller, i.e., the solution is approaching the optimal one.

A. The Lower Bound

We can derive the following Lagrangian relaxation problem by relaxing Constraint (7) of P1:

$$LR:\{g(\mu)=\min_{x_{ij}y_iz_i}\sum_{i\in\mathcal{I}}f_iy_i+\sum_{i\in\mathcal{I}}\xi_iz_i+\gamma\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}x_{ij}d_{i,j}+\sum_{j\in\mathcal{J}}\mu_j\left(1-\sum_{i\in\mathcal{I}}x_{ij}\right)$$
$$\text{s.t. } Constraints(8),(9),(10),(11),(12),(13), \tag{19}$$

where μ_j ($j\in\mathcal{J}, \mu_j\geq 0$) are Lagrangian multipliers. When the Lagrangian multipliers are given, the optimal objective value of the Lagrangian relaxation problem becomes a LB on any feasible solution of the original problem (i.e., P1).

We first ignore Constraint (10). Then, for any given set of multipliers, it can be observed that LR can be decomposed into $|I|$ subproblems, i.e., one for each cloudlet site. Thus, for each site, we have

$$LR_i: g_i=\min_{x_{ij}y_iz_i} f_iy_i+\xi_iz_i+\sum_{j\in\mathcal{J}}(\gamma d_{ij}-\mu_j)x_{ij} \tag{20}$$

$$\text{s.t. } \sum_{j\in\mathcal{J}}x_{ij}\leq sz_i \tag{21}$$

$$z_i\leq ey_i \tag{22}$$

$$x_{ij}\in\{0,1\}\quad \forall j\in\mathcal{J} \tag{23}$$

$$y_i\in\{0,1\} \tag{24}$$

$$z_i\geq 0 \text{ integer.} \tag{25}$$

For each cloudlet i, define vector $\boldsymbol{\Lambda}_i=\{\Lambda_{ij}|j\in\mathcal{J}\}$, where $\Lambda_{ij}=\min\{\gamma d_{ij}-\mu_j,0\}$. Given the number of servers in cloulet i (i.e., z_i), the capacity of cloudlet i in terms of the number of Avatars is z_is; thus, we define a user set $\mathcal{J}_{iz_i}$ ($\forall i\in\mathcal{I}$, $0\leq z_i\leq e$) for cloudlet i, where $|\mathcal{J}_{iz_i}|=z_is$ and $\{\Lambda_{ij}|j\in\mathcal{J}_{iz_i}\}$ is composed of z_is smallest elements of $\boldsymbol{\Lambda}_i$. Furthermore, let $\Psi_{iz_i}=\xi_iz_i+\sum_{j\in\mathcal{J}_{iz_i}}\Lambda_{ij}$; we define vector $\boldsymbol{\Psi}_i=\{\Psi_{iz_i}|0\leq z_i\leq e\}$, where the minimum value in $\boldsymbol{\Psi}_i$ is denoted as $\Psi^*_{iz_i}$ and the corresponding number of servers is z_i'. It can be observed that the optimal objective value of of LR_i can be expressed as $\min\{f_i+\Psi^*_{iz_i},0\}$.

Define vector $\boldsymbol{\Phi}=\{\Phi_i|i\in\mathcal{I}\}$, where $\Phi_i=\min\{f_i+\Psi^*_{iz_i},0\}$. Since the maximum number of cloudlets allowed by cloudlet providers is p, we define a cloudlet set $\mathcal{I}'$, where $\{\Phi_i|i\in\mathcal{I}'\}$ is composed of p smallest elements of $\boldsymbol{\Phi}$.

Lemma 2: For a given set of multipliers $\boldsymbol{\mu}=\{\mu_j|j\in\mathcal{J}\}$, the optimal solution of the Lagrangian relaxation problem (i.e., LR) can be expressed as follows:

$$\forall i\in I, y_i^*=\begin{cases}1, f_i+\Psi^*_{iz_i}<0;\ i\in\mathcal{I}'\\ 0, \text{otherwise}\end{cases} \tag{26}$$

$$\forall i\in I, z_i^*=z_i'y_i^*, \tag{27}$$

$$\forall i\in I, x_{ij}^*=\begin{cases}1, \gamma d_{ij}-\mu_j<0\ ;\ j\in\mathcal{J}_{iz_i'};y_i^*=1,\\ 0, \text{otherwise}\end{cases} \tag{28}$$

Proof: For each cloudlet site i, y_i^* can equal 1 or 0 for a given set of $\boldsymbol{\mu}=\{\mu_j|j\in\mathcal{J}\}$. Hence, to validate (27) and (28), we only need to consider two cases for each LR_i (i.e., $y_i^*=1$ or $y_i^*=0$). In the first case, $y_i^*=1$, and thus the problem LR_i can be transformed into:

$$K_i:\ \kappa_i=\min_{x_{ij}z_i}\xi_iz_i+\sum_{j\in\mathcal{J}}(\gamma d_{ij}-\mu_j)x_{ij} \tag{29}$$

$$\text{s.t. } \sum_{j\in\mathcal{J}}x_{ij}\leq sz_i \tag{30}$$

$$z_i\leq e \tag{31}$$

$$x_{ij}\in\{0,1\}\quad \forall j\in\mathcal{J} \tag{32}$$

$$z_i\geq 0 \text{ integer.} \tag{33}$$

As we know, z_i can be selected from 0 to e. For any fixed z_i, the problem K_i can be transformed into the following.

$$M_i:\ \kappa_i=\min_{x_{ij}}\sum_{j\in\mathcal{J}}(\gamma d_{ij}-\mu_j)x_{ij} \tag{34}$$

$$\text{s.t. } \sum_{j\in\mathcal{J}}x_{ij}\leq sz_i \tag{35}$$

$$x_{ij}\in\{0,1\}\quad \forall j\in\mathcal{J}. \tag{36}$$

In order to optimize the problem M_i, if $\gamma d_{ij}-\mu_j<0$, it is beneficial to allocate user j's Avatar to cloudlet j (i.e., $x_{ij}=1$). Since the capacity of cloudlet site i in term of the number of Avatars is sz_i, we need to select sz_i users (i.e. the user set $\mathcal{J}_{iz_i}$) with sz_i smallest Λ_{ij} (i.e., $\min\{\gamma d_{ij}-\mu_j,0\}$) in $\boldsymbol{\Lambda}_i$. Thus, the optimal objective value of M_i can be expressed as $\sum_{j\in\mathcal{J}_{iz_i}}\Lambda_{ij}$. Consequently, in the first case, for a certain z_i of cloudlet i, we can derive $x_{ij}=1$ only if $\gamma d_{ij}-\mu_j<0 \& j\in\mathcal{J}_{iz_i}$.

As we know, when $y_i^*=1$, z_i of cloudlet i varies from 0 to e.

For each z_i, based on the optimal solution of Problem M_i, the objective value of Problem K_i can be expressed as $\Psi_{iz_i} = \xi_i z_i + \sum_{j\in\mathcal{J}_{iz_i}} \Lambda_{ij}$. Thus, the optimal z_i^* can be achieved by $z_i^* = \arg\min_{z_i} \Psi_{iz_i}$. As $\Psi_{iz_i}^*$ is denoted as the minimum value of $\boldsymbol{\Psi}_i$ and $z_{i'}$ is the corresponding number of servers, z_i^* can be expressed as $z_i^* = z_i^{'} y_i^*$.

In the second case, $y_i^* = 0$; thus, both z_i^* and x_{ij}^* should be set as 0. Specifically, z_i^* can be expressed as $z_i^* = z_i y_i^*$. Therefore, (27) and (28) have been proved.

Since the optimal objective value of K_i is $\Psi_{iz_i}^*$, if $y_i^* = 1$, the objective value of LR_i is $f_i + \Psi_{iz_i}^*$, and 0, otherwise. It is beneficial to set $y_i^* = 1$, only if $f_i + \Psi_{iz_i}^* < 0$. Hence, the minimum objective value of LR_i can be derived as $\Phi_i = \min\{f_i + \Psi_{iz_i}^*, 0\}$. At this time, to achieve the optimal solution of LR, we should consider Constraint (10) of LR, which is the limitation of the number of cloudlets (i.e., p). In other words, if Φ_i is within p of smallest values of $\boldsymbol{\Phi}$ (i.e., $i \in \mathcal{I}^{'}$) and $f_i + \Psi_{iz_i}^* < 0$, $y_i^* = 1$; otherwise, $y_i^* = 0$. Thus, (26) is validated.

After obtaining the optimal solution of LR by Lemma 2, we still need to derive its optimal objective value. As we know, Φ_i is the minimum objective value of the problem LR_i based on its definition. Denote $\hat{\mathcal{I}}$ as the set of the selected cloudlet sites, i.e., $\hat{\mathcal{I}} = \{i | y_i^* = 1, i \in \mathcal{I}\}$. Consequently, the minimum objective value of LR, which is a LB of the original problem, is expressed as:

$$LB = \sum_{i\in\hat{\mathcal{I}}} \Phi_i + \sum_{j\in\mathcal{J}} \mu_j. \tag{37}$$

■

B. The Upper Bound

The Lagrangian relaxation problem yields the LB of the original problem. However, the solution may not be a feasible solution with respect to the original problem, i.e., the relaxed Constraint (7) may not be satisfied. Specifically, by solving the Lagrangian relaxation problem, we obtain a cloudlet vector $\mathcal{Y}^* = \{y_i^* | i \in \mathcal{I}\}$ and a server vector $\mathcal{Z}^* = \{z_i^* | i \in \mathcal{I}\}$. However, the cloudlet capacity in the network, corresponding to $\mathcal{Y}^*$ and $\mathcal{Z}^*$, may be insufficient to host all users' Avatars. Furthermore, without constraint (7), some users' Avatars are allocated to multiple cloudlets (i.e., $\sum_{i\in\mathcal{I}} x_{ij} > 1$), while some are not allocated to any cloudlet (i.e., $\sum_{i\in\mathcal{I}} x_{ij} = 0$). Hence, based on the result of the Lagrangian relaxation problem, we design a heuristic algorithm to achieve a feasible solution of the original problem, which is also an UP of the problem.

1) Adjust the capacity of the cloudlet network

If the total capacity of all cloudlets is insufficient to serve all users, new servers should be added to some cloudlets until all users can be served. In each iteration, the algorithm checks each cloudlet i that can still accommodate more servers (i.e., $\{i | z_i^* < e, i \in \mathcal{I}\}$), and calculates the weight of cloudlet i (i.e., denoted as ω_i):

$$\omega_i = \begin{cases} \xi_i(z_i^* + 1) & \text{if } y_i^* = 1, \\ \xi_i(z_i^* + 1) + f_i & \text{if } y_i^* = 0, \end{cases} \tag{38}$$

Then, the cloudlet with the smallest ω_i is selected, and z_i^* of the selected coudlet is increased by one. Meanwhile, if y_i^* of the selected cloudlet is equal to 0, let $y_i^* = 1$. The above procedure is repeated (i.e., $\mathcal{Y}^*$ and $\mathcal{Z}^*$ are updated in each iteration) until the capacity of the cloudlet network is large enough.

2) Heuristic Avatar allocation algorithm

When the updated $\mathcal{Y}^*$ and $\mathcal{Z}^*$ provide sufficient capacity, the original problem can be transformed into:

$$P2: \min_{x_{ij}} \sum_{i\in\hat{\mathcal{I}}} \sum_{j\in\mathcal{J}} d_{ij} x_{ij} \tag{39}$$

$$\text{s.t.} \sum_{i\in\hat{\mathcal{I}}} x_{ij} = 1 \quad \forall j \in \mathcal{J} \tag{40}$$

$$\sum_{j\in\mathcal{J}} \lambda_j x_{ij} \le s z_i^* \quad \forall i \in \hat{\mathcal{I}} \tag{41}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in \hat{\mathcal{I}} \quad \forall j \in \mathcal{J}. \tag{42}$$

where $\hat{I}$ is the set of selected cloudlet sites.

Lemma 3: P2 can be equivalently transformed into:

$$\begin{aligned} P3: \min_{x_{ij}} & \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} \frac{d_{ij}}{y_i^* + \epsilon} x_{ij} \\ \text{s.t.} & \sum_{i\in\mathcal{I}} x_{ij} = 1 \quad \forall j \in \mathcal{J} \\ & \sum_{j\in\mathcal{J}} \lambda_j x_{ij} \le s z_i^* \quad \forall i \in \mathcal{I} \\ & x_{ij} \in \{0,1\} \quad \forall i \in \mathcal{I} \quad \forall j \in \mathcal{J}. \end{aligned}$$

where ϵ is a small positive value close to zero.

Proof: For each $j \in \mathcal{J}$, if $y_i^* = 0$ (i.e., $i \in \mathcal{I}\backslash\hat{\mathcal{I}}$), then $d_{ij}/(y_i^* + \epsilon) = +\infty$, and thus x_{ij} should equal 0 to optimize $P3$. If $y_i^* = 1$ (i.e., $i \in \hat{\mathcal{I}}$), since ϵ is a small positive value close to zero, the value of $d_{ij}/(y_i^* + \epsilon)$ is approximately equal to d_{ij}. Hence, P2 and P3 are equivalent. ■

Based on Lemma 3, problem P2 can be transformed into:

$$P4: \min_{x_{ij}} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} c_{ij} x_{ij} \tag{43}$$

$$\text{s.t. } Constraints\ (40) - (42), \tag{44}$$

where $c_{ij} = d_{ij}/(y_i^* + \epsilon)$ is the weighted average E2E delay between user j and its Avatar located in cloudlet i. It can be seen that P4 is an integer programming problem, which is computationally expensive to achieve the optimal solution. In this case, we design a HEuristic Avatar allocaTion (HEAT) algorithm to obtain the suboptimal solution.

Let $\mathcal{J}_1$ be the set of users, whose Avatars are waiting to be allocated among cloudlets, and $\mathcal{I}_1$ be the set of cloudlets having excess resources to host at least one more Avatar. Note that at the beginning of HEAT, all users are included in $\mathcal{J}_1$, while all available cloudlets are included in $\mathcal{I}_1$. For user j (i.e., $j \in \mathcal{J}_1$), the optimal cloudlet in $\mathcal{I}_1$ is the one incurring the lowest weighted average E2E delay, i.e. $i' = \arg\min_i \{c_{ij} | i \in \mathcal{I}_1\}$; the suboptimal cloudlet is the one that incurs the second lowest c_{ij} among the cloudlets in $\mathcal{I}_1$, i.e.,

$i'' = \arg\min_i\{c_{ij}|i \in \{\mathcal{I}_1 \backslash i'\}\}$. The basic idea of HEAT is to select a suitable user in each iteration, whose suboptimal cloudlet incurs a significant E2E delay degradation as compared to the optimal cloudlet, and then allocate the user's Avatar into the optimal cloudlet. In other words, if allocating user j's Avatar to its suboptimal cloudlet has a significantly negative impact on the weighed average E2E delay, we should place it into its optimal cloudlet to avoid the corresponding E2E delay degradation.

Denote Δc_j as the E2E delay degradation by allocating user j's Avatar from the optimal cloudlet i' to the suboptimal cloudlet i'', i.e.,

$$\Delta c_j = c_{i''j} - c_{i'j}, \quad \forall j \in \mathcal{J}_1. \tag{45}$$

Therefore, in an iteration, HEAT will choose to allocate a suitable user $\hat{j}$'s Avatar, where $\hat{j} = \arg\max_j\{\Delta c_j|j \in \mathcal{J}_1\}$, to its optimal cloudlet in order to mitigate the E2E delay degradation. Then, user $\hat{j}$ is removed from $\mathcal{J}_1$. Afterwards, if the optimal cloudlet has no extra space for hosting more Avatars, it is removed from $\mathcal{I}_\infty$. Note that once $\mathcal{I}_1$ is updated, the algorithm has to recalculate i', i'' and Δc_j for each user $j \in \mathcal{J}_1$. The above iteration is repeated until all users' Avatars are allocated, i.e., $\mathcal{J}_1 = \emptyset$. As a result, we obtain a feasible solution of the original problem, yielding an UP. The details of the algorithm is shown in Algorithm 1.

Algorithm 1: HEAT algorithm

Input: The cloudlet placement vector $\mathcal{Y}^* = \{y_i|i \in \mathcal{I}\}$. The server vertor $\mathcal{Z}^* = \{z_i|i \in \mathcal{I}\}$. The average E2E delay vector, i.e., $C = \{c_{ij}|i \in \mathcal{I}, j \in \mathcal{J}\}$.

Output: Avatar allocation vector, i.e., $\mathcal{X}^* = \{\S_{\rangle|}\rangle \in \mathcal{I}, | \in \mathcal{J}\}$.

1: Initialize $\mathcal{J}_1$ and $\mathcal{I}_1$ based on their definitions;
2: $\forall j \in \mathcal{J}_1$, calculate Δc_j based on Eq. (45);
3: **while** $\mathcal{J}_1 \neq \emptyset$ **do**
4: Find user $\hat{j}$, where $\hat{j} = \arg\max\{\Delta c_j|j \in \mathcal{J}_1\}$;
5: Allocate user $\hat{j}$ to its optimal cloudlet i' (i.e., $i' = \arg\min_i\{c_{i\hat{j}}|i \in \mathcal{I}_1\}$), and let $x_{i'\hat{j}} = 1$;
6: Update the user set $\mathcal{J}_1$, i.e.; $\mathcal{J}_1 = \mathcal{J}_1 \backslash \hat{j}$.
7: **if** cloudlet i' is full **then**
8: Update $\mathcal{I}_1$, i.e., $\mathcal{I}_1 = \mathcal{I}_1 \backslash i'$;
9: $\forall j \in \mathcal{J}_1$, recalcuate Δc_j based on Eq. (45);
10: **end if**
11: **end while**
12: **return** $\mathcal{X}$.

C. Subgradient Algorithm

When the LB and UP of the original problem are obtained, we need to update the Lagrangian multipliers u_j by solving the following Lagrangian dual problem to improve the LB and UP of the next iteration. In this paper, we apply the subgradient method to derive the Lagrangian multipliers in each iteration [30], [31]. The Lagrangian dual problem is expressed as

$$LD: \max_{\mu} g(\mu). \tag{46}$$

The subgradients are calculated as follows:

$$\Psi_j = 1 - \sum_{i \in I} x_{ij}^*, \quad \forall j \in \mathcal{J}, \tag{47}$$

where x_{ij}^* is derived from the optimal solution of the Lagrangian relaxation problem.

Moreover, the step size t in the subgradient algorithm is defined as [31]:

$$t = \frac{\sigma(UB^{opt} - LB)}{\sum\limits_{j \in \mathcal{J}} \Psi_j^2} \tag{48}$$

where UB^{opt} indicates the best UP so far, and σ is a scalar ranging from 0 and 2. If the improvement of LB cannot be achieved within a fixed number of subsequent iterations, σ is halved to facilitate the search for a better LB.

Therefore, the Lagrangian multipliers μ_j in iteration $n+1$ can be updated as follows:

$$\mu_j^{n+1} = \mu_j^n + \Psi_j t^n. \tag{49}$$

If the number of iterations exceeds the predefined maximum number M, the subgradient algorithm stops.

The details of the complete CAPABLE algorithm are shown in Algorithm 2.

Algorithm 2: The CAPABLE algorithm

1: Initialize the Lagrangian multipliers $\mu_j, \forall j \in \mathcal{J}$;
2: Solve the problem LR to obtain the optimal solution (i.e., $\mathcal{Y}^*$, $\mathcal{Z}^*$ and $\mathcal{X}^*$) according to Eq. (26) (27) (28); calculate LB.
3: Based on the optimal solution of LR, use the HEAT algorithm to get a UP of the original problem;
4: **if** $UP < UP^{opt}$ **then**
5: Let $UP^{opt} = UP$;
6: **else**
7: Let $\sigma = \sigma \backslash 2$;
8: **end if**
9: Update the step size t based on Eq. (48);
10: Update the Lagrangian multipliers $\mu_j, \forall j \in \mathcal{J}$, based on Eq.(49);
11: **if** $iter < M$ **then**
12: Go to Step 2;
13: **else**
14: STOP;
15: **end if**.

VI. Dynamic Avatar Allocation (DARA)

As we know, users move around in the network dynamically. If a user roams to a new BS in a time slot, it is desirable to reallocate the user's Avatar to a nearby cloudlet, thus reducing the E2E delay between the user and its Avatar. Therefore, after cloudlets have been deployed by the above CAPABLE algorithm, it is necessary to design a Dynamic AvataR Allocation (DARA) scheme to reallocate Avatars among cloudlets to minimize the E2E delay between users and their Avatars in each time slot. Note that if a user's Avatar is assigned to a different cloudlet as compared to the previous time slot, the Avatar will migrate from the previous cloudlet to the destination cloudlet by the live migration method.

Let $\mathcal{I}_2$ be the cloudlet vector generated by the above CAPABLE strategy, i.e., $\mathcal{I}_2 = \{i|y_i^* = 1, i \in \mathcal{I}\}$. Let x_{ij}' be the

binary variable indicating whether user j's Avatar is located at cloudlet i in the current time slot. Denote k_j as the BS where user j is located, and t_{ij} as the E2E delay between user j and its Avatar in the current time slot, which is equal to the E2E delay between user j's BS and its cloudlet (i.e., $t_{ij} = \tau_{k_j i}$). Therefore, we formulate the Avatar allocation problem in each time slot as follows:

$$P5: \min_{x'_{ij}} \sum_{i \in \mathcal{I}_\epsilon} \sum_{j \in \mathcal{J}} t_{ij} x'_{ij} \tag{50}$$

$$\text{s.t.} \sum_{i \in \mathcal{I}_\epsilon} x'_{ij} = 1 \quad \forall j \in \mathcal{J} \tag{51}$$

$$\sum_{j \in \mathcal{J}} \lambda_j x'_{ij} \le s z_i^* \quad \forall i \in \mathcal{I}_\epsilon \tag{52}$$

$$x'_{ij} \in \{0,1\} \quad \forall i \in \mathcal{I}_2 \quad \forall j \in \mathcal{J}. \tag{53}$$

Based on Lemma 3, problem P5 can be transformed into:

$$P6: \min_{x'_{ij}} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} c'_{ij} x'_{ij} \tag{54}$$

$$\text{s.t. } \textit{Constraints } (51), (52), (53), \tag{55}$$

where $c'_{ij} = t_{ij}/(y_i^* + \epsilon)$ is the weighted E2E delay between user j and its Avatar located in cloudlet i in the time slot.

This problem is also an integer programming problem, similar to P2, and thus it is computationally expensive to get the optimal solution. Since we have to solve this problem in each time slot to dynamically allocate Avatars among cloudlets, we apply the above HEAT algorithm to obtain the suboptimal solution.

Note that the DARA scheme will incur some specific overhead in the network. When users' Avatars are reassigned to a new cloudlet in a time slot, the Avatars have to be migrated to the new cloudlet to provision service to the users. Meanwhile, the HEAT algorithm has to be performed in each time slot to reassign the Avatars among cloudlets, and thus the complexity of the algorithm directly impacts the overhead. We now analyze the computational complexity of the algorithm. The complexity of Step 2 is $|\mathcal{J}|$. Meanwhile, the complexity of Steps 4 and 5 is $O(|\mathcal{J}|+|\mathcal{I}|)$ in the worst case. As the algorithm allocates one user's Avatar to its optimal cloudlet in each iteration, the number of iterations is $|\mathcal{J}|$, and thus the total complexity of Steps 4 and 5 can be expressed as $O((|\mathcal{J}|+|\mathcal{I}|)|\mathcal{J}|)$. Meanwhile, as Steps 8 and 9 repeat for $|\mathcal{I}|$ times in the worst case, the corresponding complexity is $O((|\mathcal{J}|+\infty)|\mathcal{I}|)$. Summarizing all these steps, the complexity of the algorithm can be expressed as $O((|\mathcal{J}|+|\mathcal{I}|)|\mathcal{J}|)$.

VII. Simulation Results

In this section, we set up the simulation of the CAPABLE strategy and the DARA scheme to evaluate their performance. We begin by describing the simulation environment setting. To demonstrate the effectiveness of the CAPABLE algorithm, we compare the solutions of CAPABLE and the CPLEX tool in a small-scale randomly generated network. Then, we apply a large-scale network in the real world to further evaluate the performance of the proposed strategies.

A. Simulation Environment

In the simulation, the capacity of each server in terms of the number of Avatars is set as 30; the maximum number of servers in a cloudlet is 10. The rental cost at each site f_i is determined by the Normal distribution with an average of 500, and a variance of 100; the price of a server is 50. To construct the E2E delay matrix, we assume that the E2E delay between BS k and cloudlet i (i.e., τ_{ki}) is a function of the geographic distance between them [32], [33], i.e., $\tau_{ki} = 3.3 * L_{ki}$, where L_{ki} is the distance between BS k and cloudlet i in km and the unit of τ_{ki} is ms. Note that if $k = i$, it means that cloudlet i is attached to BS k, and thus $\tau_{ki} = 0$. We can apply historical data to calculate the occurrence probability p_{kj}[1] of user j among BSs.

B. Performance in the Small-scale Randomly Generated Network

In the small-scale randomly generated network, 100 BSs are placed in an 10*10 km^2 area, where each BS has a coverage area of 1 km^2. Meanwhile, the number of users in the network is 500. To emulate the real scenario, each user randomly chooses five BSs and randomly moves among the five BSs in different time slots.

For the small-scale network, we can use the CPLEX tool to generate the optimal solution of the cloudlet placement problem. To evaluate the performance of the proposed algorithm, we can compare the results of CAPABLE with that of CPLEX. Hence, both deployment cost and simulation time are examined to verify the effectiveness of the CAPABLE algorithm.

In Fig. 2, we study the deployment cost delivered by CAPABLE and CPLEX, where η are set as 0.2 and 0.4, respectively. Note that the deployment cost of the proposed algorithm is only a little higher than that of CPLEX, which suggests that the performance of CAPABLE does not degrade significantly as compared to CPLEX because the proposed algorithm is able to improve the UB and LB of the original problem in each iteration, and so it yields a desirable feasible solution when the gap between the UB and LB becomes small. Similarly, Fig. 3 and Fig. 4 compare the cloudlet cost and average E2E delay of the two algorithms, respectively. Both the cloudlet cost and average E2E delay of the proposed algorithm is very close to the near optimal solution of CPLEX, which suggests that the number and locations of the cloudlets selected by CAPABLE is close to that of CPLEX.

As seen from Fig. 5, the simulation time of the proposed scheme is 90.6% and 77.8% fewer than that of CPLEX respectively, when η equals 0.2 and 0.4. In each iteration, we can apply a closed-form solution of Lemma 2 to calculate the LB, and then obtain the UB by a heuristic algorithm. Consequently, its simulation time is remarkably decreased as compared to that of CPLEX. As the proposed algorithm can achieve a suboptimal solution by consuming much less time as compared to CPLEX, it is effective to solve the cloudlet placement problem.

[1] $p_{kj} = \frac{\textit{the amount of time that user } j \textit{ is associated with BS } k}{\textit{the total time period}}$

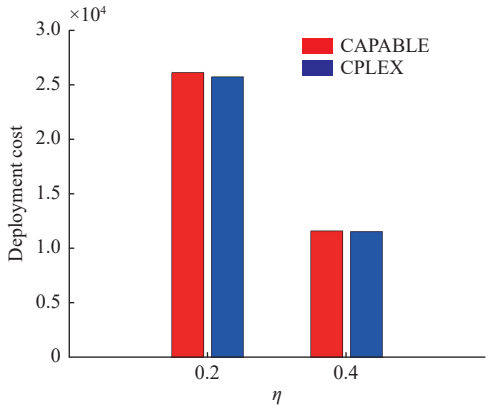


Fig. 2. Deployment cost comparison.

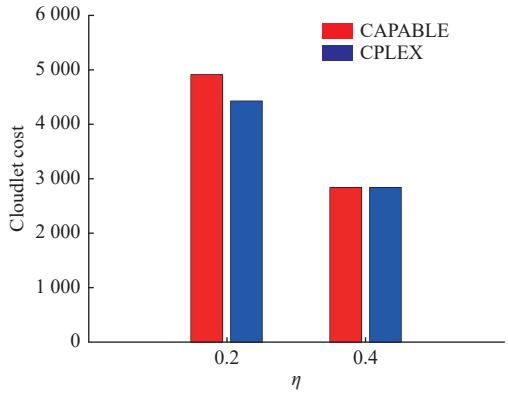


Fig. 3. Cloudlet cost comparison.

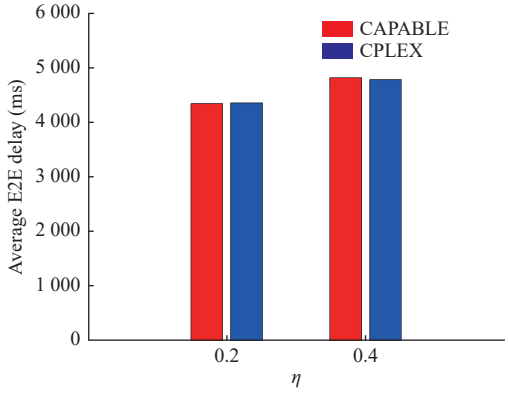


Fig. 4. Total average E2E delay comparison.

C. Performance in A Large-scale Network

To further evaluate the performance of the proposed algorithms, we study the impact of system parameters on the proposed algorithms in a real mobile network in Harbin, China, with 8826 users and 500 BSs and extract user mobility activities in one day. To continuously track user locations, each packet of a user is monitored, and thus we can extract the BS information from each packet and consider the BS's location as the current location of the user (e.g., if a packet of the user contains the information of BS A, we can say that the user is currently associated with BS A). As shown in below,

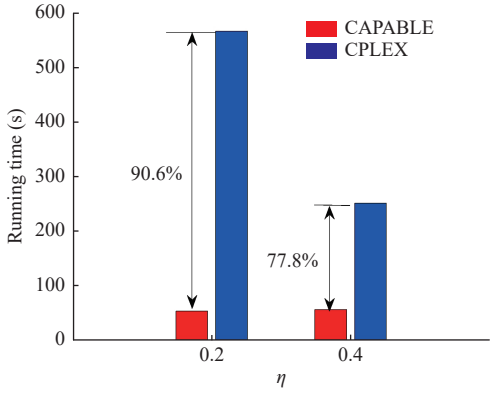


Fig. 5. Running time comparison.

the proposed CAPABLE algorithm can also effectively solve the cloudlet placement problem in the large-scale network.

1) Performance of the CAPABLE strategy

The tradeoff coefficient η $(0 < \eta < 1)$ determines the wight ratio of the cloudlet cost to the average E2E delay in the cloudlet placement. If η increases, the cloudlet providers focus more on the cloudlet cost than the average E2E delay. Thus, we should test the impact of η on the performance of the proposed algorithm to help cloudlet providers choose a proper η based on their practical requirement. For comparison, we consider the scenario that all users' Avatars are allocated to a remote data center that is placed at the southwest point of the network. Moreover, we also consider the Heaviest-AP First placement (HAF) strategy [17] for comparison, which places a certain number of cloudlets (i.e., 30 cloudlets in the simulation) at the BSs having the heaviest workloads to reduce the E2E delay between users and their cloudlets.

Fig. 6 shows that the average E2E delay achieved by CAPABLE is significantly reduced as compared to the traditional big data network and the HAF strategy as η varies. With the increase of η, the average E2E delay achieved by CAPABLE grows. When η is small, the cloudlet providers will deploy a large number of cloudlets at the BSs with high user density, and thus most users can access their Avatars in the nearby cloudlets, incurring a low average E2E delay between users and their Avatars. Then, with the increase of η, fewer cloudlets are available because cloudlet providers are more conscious of placing cloudlets in a cost-effective way, i.e., a user's Avatar has a higher probability to be allocated to a cloudlet with higher average E2E delay. The average E2E delay of the traditional big data network and HAF, on the other hand, is not impacted by η because each user's Avatar is located in the remote data center in the traditional big data network. With regards to HAF, the cloudlet placement is determined by the workloads of different BSs.

As shown in Fig. 7, we examine how the cloudlet cost of CAPABLE varies with the increase of η. It can be seen that the cloudlet cost of CAPABLE drops and finally plateaus. When η is high, the cloudlet cost of CAPABLE is less than that of HAF because the cloudlet providers tend to decrease the number of cloudlets and their servers to reduce the cloudlet cost, with the increase of η. However, the numbers of cloudlets and their servers are bounded from below by the

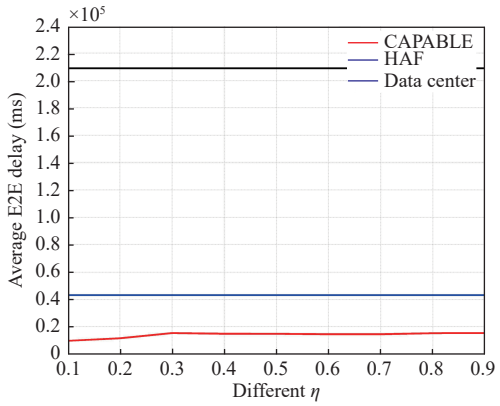


Fig. 6. Total average E2E delay with respect to different η.

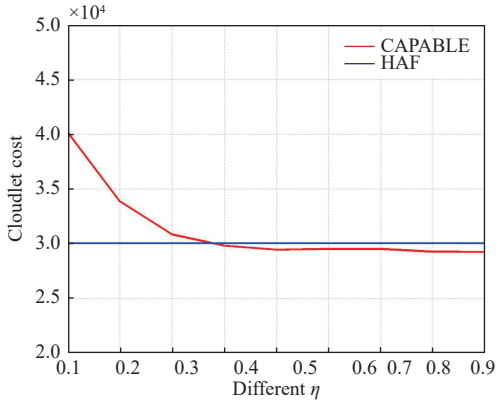


Fig. 7. Cloudlet cost with respect to different η.

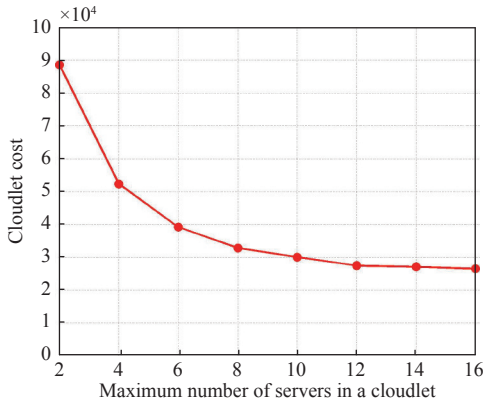


Fig. 8. Cloudlet cost with respect to maximum number of servers in a cloudlet ($\eta = 0.4$).

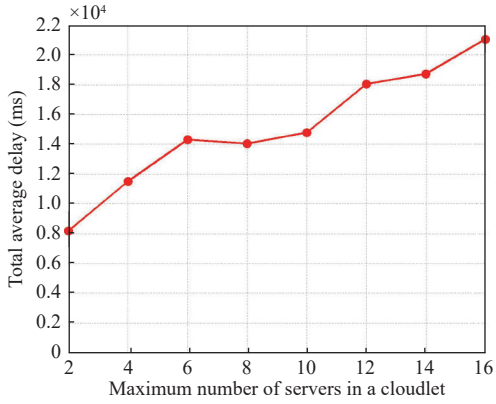


Fig. 9. Total average E2E delay with respect to maximum number of servers in a cloudlet ($\eta = 0.4$).

demands of all users' Avatars. When the capacity of cloudlets is close to the minimum value that can serve all users' Avatars, increasing η has very little effect on the numbers of cloudlets and servers.

CAPABLE aims to reduce the number of cloudlets and servers while decreasing the average E2E delay between users and their Avatars. With the increase of the cloudlet size (i.e., the maximum number of servers in a cloudlet), the cloudlet providers are more likely to install more servers to cloudlets in the areas with high user density so as to maintain low E2E delay, while reducing the total number of cloudlets in the network. To study how the maximum number of servers in a cloudlet impacts the performance of CAPABLE, we examine the cloudlet cost and average E2E delay of CAPABLE by changing the maximum number of servers in a cloudlet. As shown in Figs. 8 and 9, when the maximum number of servers in a cloudlet increases from 2 to 16 ($\eta = 0.4$), the cloudlet cost is reduced significantly while the average E2E delay increases. If the maximum number of servers of a cloudlet is small, the cloudlet providers need to deploy a large number of cloudlets to provision enough computing resources for users' Avatars, thus users' Avatars are more likely to be placed at a nearby cloudlet (i.e., incurring low average E2E delay). If the maximum number of servers of each cloudlet is large, in order to reduce the cloudlet cost, the cloudlet provider will put more servers to cloudlets in areas with high user density and reduce the total number of cloudlets in the network. As the number of cloudlets degrades, the average E2E delay between users and their Avatars also significantly increases. Moreover, the cloudlet cost of CAPABLE does not diminish significantly when the maximum number of servers of a cloudlet is close to 16. This is because when the number of cloudlets is very small, the further reduction of the number of cloudlets will tremendously increase the total average E2E delay, which has a negative impact on the deployment cost of the network. Hence, to minimize the deployment cost, the cloudlet cost of CAPABLE changes slowly when the maximum number of servers of a cloudlet is close to 16.

2) Performance of DARA

After cloudlets are placed at the selected BSs by CAPABLE, we further evaluate the performance of the DARA scheme. As we know, user mobilities follow the data trace that we sampled from the real world, while the cloudlet placement is static. As a result, it is important to reallocate users' Avatars among cloudlets to maintain the low E2E delay in different time slots.

In Fig. 10, we compare the E2E delay in different time slots delivered by the two strategies, given $\eta = 0.4$. It can be seen that the E2E delay of DARA is significantly reduced as compared to CAPABLE, which does not reallocate Avatars in different time slots. This is because when users roam to new BSs in a time slot, DARA can allocate users to new suitable

cloudlets to reduce their E2E delay. For instance, if user j roams to a new BS i, the new BS may be far away from user j's previous cloudlet A (i.e., incurring the E2E delay τ_{iA}). Meanwhile, there is an alternative cloudlet B (i.e., incurring the E2E delay τ_{iB}). In this case, if $\tau_{iB} < \tau_{iA}$, reallocating user j's Avatar from cloudlet A to cloudlet B can reduce the E2E delay between user j and its Avatar.

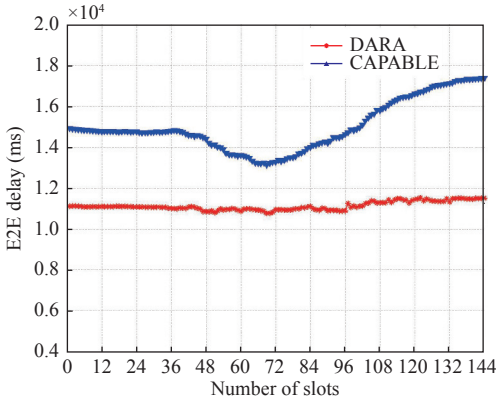


Fig. 10. Total E2E delay in different time slots.

VIII. Conclusion

In this paper, we have presented the cloudlet network architecture which facilitates mobile cloud computing at the network edge. Specifically, each user can access its Avatar, which is considered as the designated VM for the user, with low E2E delay. In order to minimize the average E2E delay between users and their cloudlets, some existing studies deploy a certain number of cloudlets to some suitable sites. However, aside from the E2E delay, a cloudlet provider also has to pay attention to the cloudlet cost. Hence, we have proposed the CAPABLE strategy to determine the location and capacity of each cloudlet, so as to optimize the tradeoff relation between the E2E delay and cloudlet cost. As the cloudlet placement problem is NP-hard, we have proposed a Lagrangian heuristic algorithm to achieve the suboptimal solution. After cloudlets are placed in the network, because users roams among the BSs, their Avatars should be allocated to cloudlets in each time slot. Hence, we have designed the DARA scheme to optimally allocate Avatars to suitable cloudlets, so that the E2E delay between users and their Avatars are minimized in different time slots. We have demonstrated that CAPABLE achieves a low deployment cost of the cloudlet network consisting of both the cloudlet cost and E2E delay cost. Moreover, DARA can improve the performance of E2E delay of users in different time slots.

References

[1] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, pp. 4, 2009.

[2] Q. Fan and N. Ansari, "Application aware workload allocation for edge," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2146–2153, Jun. 2018.

[3] Q. Fan and N. Ansari, "Towards workload balancing in fog computing empowered IoT," *IEEE Transactions on Network Science and Engineering*, DOI: 10.1109/TNSE.2018.2852762, 2018.

[4] P. Zhang, M. Zhou, and G. Fortino, "Security and trust issues in fog computing: A survey," *Future Generation Computer Systems*, vol. 88, pp. 16–27, Nov. 2018.

[5] 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.

[6] L. Gu, D. Zeng, S. Guo, A. Barnawi, and Y. Xiang, "Cost efficient resource management in fog computing supported medical cyber-physical system," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 1, pp. 108–119, 2017.

[7] A. Kiani and N. Ansari, "Edge computing aware NOMA for 5G networks," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1299–1306, Apr. 2018.

[8] Q. Fan and N. Ansari, "Towards traffic load balancing in drone-assisted communications for IoT," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3633–3640, Apr. 2019.

[9] L. A. Tawalbeh, W. Bakheder, and H. Song, "A mobile cloud computing model using the cloudlet scheme for big data applications," in *Proc. PWC. IEEE 1st Int. Conf. Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, Washington DC, USA, 2016, pp. 73–77.

[10] M. Quwaider and Y. Jararweh, "Cloudlet-based efficient data collection in wireless body area networks," *Simulation Modelling Practice and Theory*, vol. 50, pp. 57–71, 2015.

[11] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos, "Edge analytics in the internet of things," *IEEE Pervasive Computing*, vol. 14, no. 2, pp. 24–31, 2015.

[12] X. Sun and N. Ansari, "PRIMAL: PRofIt Maximization Avatar pLacement for Mobile Edge Computing," in *Proc. of IEEE Int. Conf. on Communications (ICC)*, Kuala Lumpur, Malaysia, May 2016.

[13] X. Sun, N. Ansari, and Q. Fan, "Green energy aware avatar migration strategy in green cloudlet networks," in *Proc. IEEE 7th Int. Conf. on Cloud Computing Technology and Science, (CloudCom)*, Vancouver, Canada, Nov. 2015.

[14] Q. Fan, N. Ansari, and X. Sun, "Energy driven avatar migration in green cloudlet networks," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1601–1604, 2017.

[15] 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, Oct. 2016.

[16] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Capacitated cloudlet placements in wireless metropolitan area networks," in *Proc. 40th IEEE Conf. on Local Computer Networks (LCN)*, Clearwater Beach, FL, Oct. 2015, pp. 570–578.

[17] 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.

[18] Q. Fan and N. Ansari, "Cost aware cloudlet placement for big data processing at the edge," in *IEEE Int. Conf. on Communications (ICC)*, Paris, France, May 21–25, 2017, pp. 1–6.

[19] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, "Softcell: scalable and flexible cellular core network architecture," in *Proc. of the 9th ACM conf. on Emerging Networking Experiments and Technologies,* Santa Barbara, CA, Dec.09–12 2013, pp. 163–174.

[20] Q. Fan and N. Ansari, "Workload allocation in hierarchical cloudlet networks," *IEEE Communications Letters*, vol. 22, no. 4, pp. 820–823, Apr. 2018.

[21] X. Sun and N. Ansari, "Avaptive avatar handoff in the cloudlet network," *IEEE Transactions on Cloud Computing*, DOI: 10.1109/TCC.2017.2701794, 2017.

[22] Q. Fan and N. Ansari, "Green energy aware user association in heterogeneous networks," in *Proc. of IEEE Conf. Wireless Communications and Networking*, Doha, Qatar, Apr. 2016.

[23] N. L. Van Adrichem, C. Doerr, and F. A. Kuipers, "Opennetmon: network monitoring in openflow software-defined networks," in *Proc. IEEE Network Operations and Management Symposium (NOMS)*, Krakow, Poland, May 2014, pp. 1–8.

[24] C. Yu, C. Lumezanu, A. Sharma, Q. Xu, G. Jiang, and H. V. Madhyastha, "Software-defined latency monitoring in data center networks," in *Proc. Int. Conf. Passive and Active Network Measurement*, vol. 8995, Mar. 2015, pp. 360–372.

[25] J. Ghosh, S. J. Philip, and C. Qiao, "Sociological orbit aware location approximation and routing in manet," in *Proc. 2nd Int. Conf. Broadband Networks*, Boston, MA, Oct. 2005, pp. 641–650.

[26] L. Yang, J. Cao, G. Liang, and X. Han, "Cost aware service placement

and load dispatching in mobile cloud systems," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1440–1452, 2016.

[27] P. B. Mirchandani and R. L. Francis, *Discrete Location Theory*, 1990.

[28] G. Cornujelos, R. Sridharan, and J.-M. Thizy, "A comparison of heuristics and relaxations for the capacitated plant location problem," *European Journal of Operational Research*, vol. 50, no. 3, pp. 280–297, 1991.

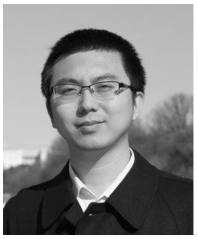
[29] G. Ghiani, L. Grandinetti, F. Guerriero, and R. Musmanno, "A lagrangean heuristic for the plant location problem with multiple facilities in the same site," *Optimization Methods and Software*, vol. 17, no. 6, pp. 1059–1076, 2002.

[30] M. L. Fisher, "The lagrangian relaxation method for solving integer programming problems," *Management Science*, vol. 27, no. 1, pp. 1–18, 1981.

[31] L. Y. Wu, X. S. Zhang, and J. L. Zhang, "Capacitated facility location problem with general setup cost," *Computers & Operations Research*, vol. 33, no. 5, pp. 1226–1241, 2006.

[32] R. Landa *et al*., "The large-scale geography of internet round trip times," in *Proc. Conf. IFIP Networking*, Brooklyn, NY, May 2013, pp. 1–9.

[33] R. Goonatilake and R. A. Bachnak, "Modeling latency in a network distribution," *Network and Communication Technologies*, vol. 1, no. 2, pp. 1–11, 2012.



Qiang Fan (S'15) received his M.S. degree in Electrical Engineering from Yunnan University of Nationalities, China, in 2013. He is currently a research Assistant and a Ph.D. Candidate in the Department of Electrical and Computer Engineering, New Jersey Institute of Technology (NJIT), Newark, New Jersey, USA. His research interests include mobile edge computing, internet of things, green communications, drone-assisted networking, and free space optical communications. He is also on Technical Program Committee for Cloud Computing 2018 and IEEE Wireless Africa 2019

Nirwan Ansari (S'78-M'83-SM'94-F'09) received a Ph.D. from Purdue University, an MSEE from the University of Michigan, and a BSEE (summa cum laude with a perfect GPA) from NJIT. He is Distinguished Professor of Electrical and Computer Engineering at the New Jersey Institute of Technology (NJIT). He has also been a visiting (chair) professor at several universities. He authored Green Mobile Networks: A Networking Perspective (Wiley-IEEE, 2017) with T. Han, and co-authored two other books. He has also (co-)authored more than 600 technical publications, over 250 published in widely cited journals/magazines. He has guest-edited a number of special issues covering various emerging topics in communications and networking. He has served on the editorial/advisory board of over ten journals. His current research focuses on green communications and networking, cloud computing, drone-assisted networking, and various aspects of broadband networks.

He was elected to serve in the IEEE Communications Society (ComSoc) Board of Governors as a member-at-large, has chaired some ComSoc technical and steering committees, has been serving in many committees such as the IEEE Fellow Committee, and has been actively organizing numerous IEEE International Conferences/Symposia/Workshops. He is frequently invited to deliver keynote addresses, distinguished lectures, tutorials, and invited talks. Some of his recognitions include several Excellence in Teaching Awards, a few best paper awards, the NCE Excellence in Research Award, the ComSoc TC-CSR Distinguished Technical Achievement Award, the ComSoc AHSN TC Technical Recognition Award, the IEEE TCGCC Distinguished Technical Achievement Recognition Award, the NJ Inventors Hall of Fame Inventor of the Year Award, the Thomas Alva Edison Patent Award, Purdue University Outstanding Electrical and Computer Engineering Award, NCE 100 Medal, and designation as a COMSOC Distinguished Lecturer. He has also been granted 40 U.S. patents.