Online Service Provisioning and Updating in QoS-aware Mobile Edge Computing

1st Shuaibing Lu
Faculty of Information Technology
Beijing University of Technology
Beijing, China
lushuaibing@bjut.edu.cn

2nd Jie Wu
Center for Networked Computing
Temple University
Philadelphia, USA
jiewu@temple.edu

3rd Pengfan Lu
Faculty of Information Technology
Beijing University of Technology
Beijing, China
lu_peng_fan@emails.bjut.edu.cn

4th Jiamei Shi
Faculty of Information Technology
Beijing University of Technology
Beijing, China
shijiamei@emails.bjut.edu.cn

5th Ning Wang
Department of Computer Science
Rowan University
Glassboro, USA
wangn@rowan.edu

6th Juan Fang
Faculty of Information Technology
Beijing University of Technology
Beijing, China
fangjuan@bjut.edu.cn

***Abstract*—The vigorous development of IoT technology has spawned a series of applications that are delay-sensitive or resource-intensive. Mobile edge computing is an emerging paradigm which provides services between end devices and traditional cloud data centers to users. However, with the continuously increasing investment of demands, it is nontrivial to maintain a higher quality-of-service (QoS) under the erratic activities of mobile users. In this paper, we investigate the service provisioning and updating problem under the multiple-users scenario by improving the performance of services with long-term cost constraints. We first decouple the original long-term optimization problem into a per-slot deterministic one by using Lyapunov optimization. Then, we propose two service updating decision strategies by considering the trajectory prediction conditions of users. Based on that, we design an online strategy by utilizing the committed horizon control method looking forward to multiple slots predictions. We prove the performance bound of our online strategy theoretically in terms of the trade-off between delay and cost. Extensive experiments demonstrate the superior performance of the proposed algorithm.**

***Index Terms*—mobile edge computing, online service provisioning, mobility, quality-of-service (QoS).**

I. Introduction

The vigorous development of Internet of things (IoT) technology has led to the explosive growth of mobile terminal equipment and data volume. At the same time, a series of resource-intensive and delay-sensitive applications, such as augmented reality (AR)/virtual reality (VR), intelligent driving, and dynamic content delivery, have emerged and been widely used [1], [2], [4], [6]. It is difficult for the traditional cloud data center to meet the performance requirements due to the long distance from massive terminals. Mobile Edge Computing (MEC) is a promising framework to solve this problem by deploying edge servers at base stations to supply computation, storage, and networking resources for multiple users [3]. However, the finite capabilities of edge servers and the erratic activities of multiple end-users pose challenges in guaranteeing the quality of service (QoS). Therefore, there are two key problems: (i) How to guarantee the QoS to avoid service interruption with unknown trajectories when users are away from the original edge servers? (ii) How to realize service provisioning, and updating the services that can efficiently utilize the limited resources without overwhelming the cost constraint? In this paper, we investigate the service provisioning and updating problem under the multiple-users scenario by improving the performance of services with a long-term cost constraint.

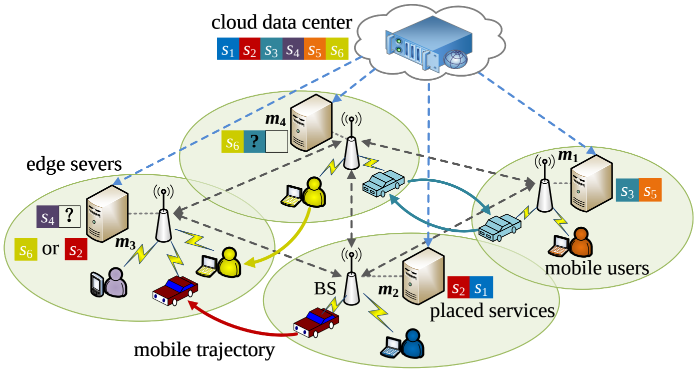


Fig. 1. An illustrating example.

A. Motivation and Challenges

We illustrate the motivation and challenges of the online service provisioning and updating problem by using an example in Figure 1. The squares with six different colors represent the services s_1 to s_6, which are initially provisioned in the cloud data center. We assume that the services required by the users have been deployed on the edge servers, and each service only serves one user. For mobile users, the QoS can be guaranteed through provisioning a replication or migration among edge servers. (i) The trajectories of multiple users are diverse and erratic, hence it is non-trivial to find an efficient strategy that can improve the QoS of mobile users by considering the cost constraint. Taking service s_3 as an example, we suppose that end user u_3 moves from an area in m_1 to m_4 at time slot t and goes back to m_1 after several slots. One extreme solution is to migrate or provision a replication of s_3 on edge server m_4 which may bring a lower delay for user u_3. However, the total cost will be the maximum one among all feasible assignments if the replication or migration costs of services are extremely high. Another extreme assignment is to retain service s_3 within m_1, which minimizes the extra (replication or migration) cost. When u_3 moves to m_4, the service can only be enjoyed through communication with m_1, which will make the quality of service decrease. Therefore, when and where to migrate or replicate services is crucial for balancing the trade-off between long-term cost and users' total delay. (ii). Since the capabilities of edge servers are limited, determining which services are chosen to be placed in order to obtain a better performance when multiple users make the same decision at the same time is non-trivial. Taking services s_2 and s_6 as an example, we suppose that users u_2 and u_6 move from m_1 towards m_4 during the same time slot. If both services want to migrate or replicate to m_3, there will be a conflict due to the fact that the remaining capacity can only receive one service. Therefore, the problem of how to make a better choice by joint considering the resource efficiency and users' performance is a challenge.

B. Contributions and Paper Organization

In this paper, we investigate the service provisioning and updating problem under the multiple users scenario by improving the performance of services with long-term cost constraints. Our contributions can be summarized as follows:

- We investigate the service provisioning and updating problem by formulating to minimize the average long-term delay of multiple users, and we decouple the original long-term optimization problem into a per-slot deterministic one by using Lyapunov optimization.
- We propose two service updating decision strategies by considering the trajectory prediction conditions of users. For one scenario, namely the service updating without available predictive information, we propose a novel strategy by introducing the conflict resolution factor. For the other scenario, which is the service updating with multi-step prediction, we optimize the total delay of users per-slot by converting a weighted graph under the constructed activity set.
- Based on that, we design an online strategy by utilizing the committed horizon control method looking forward to multiple slots predictions. We prove the performance bound of our online strategy theoretically in terms of the trade-off between delay and cost.
- We conduct extensive experiments to compare our strategy with several baselines based on the Microsoft GPS trajectory dataset which was reconstructed by 40 users. The results are shown from different perspectives to provide conclusions. Extensive experiments demonstrate the superior performance of the proposed algorithm.

The remainder of this paper is organized as follows. Section II surveys related works. Section III describes the model and then formulates the problem. Section IV investigates the service provisioning and updating problem based on Lyapunov optimization. Section V investigates the online optimization provisioning strategy. Section V includes the experiments. Finally, Section VI concludes the paper.

II. Related work

As an emerging paradigm, edge computing extends services closer to end-users. However, the finite capabilities of edge servers and the erratic activities of users pose new challenges [5]. One of the main open branches is the service provisioning problem, which is well-investigated in edge computing under mobility scenarios [6]. Various works have been studied from different aspects of this problem. Yu et al. [7] investigated the service provisioning problem in mobile edge computing, which aimed to minimize the traffic load caused by service request forwarding, and proposed an efficient decentralized algorithm based on the matching theory. Nezami et al. [8] formulated a decentralized load-balancing problem for IoT service provisioning, and they introduced a decentralized multi-agent system that utilized edge servers to balance the workloads and minimized the costs involved in service execution. Zhang et al. [9] solved the computation and delay costs minimization problem by proposing an efficiently approximate algorithm based on semi-definite relaxation. The above works optimized the service cost and delay from the offline scenario.

In the online scenario, Chen et al. [10] studied the service collaboration with master-slave dependency among service chains of mobile users and jointly optimized the cost and delay by introducing a distributed algorithm based on Markov approximation. Xu et al. [11] proposed an efficient online algorithm based on Gibbs sampling which can achieve provable close-to-optimal performance. Han et al. [12] transformed the online multi-component service placement into an ant colony optimization problem, and they proposed a level traversal component ranking method to achieve faster convergence. These works focus on optimizing the cost and delay of the service provisioning problem, however, they ignore the erratic movements of users.

In order to tackle the challenge of users' mobility, some existing works were proposed based on service migration. Ning et al. [13] studied the service provisioning problem by constructing a stochastic mobility system, and they introduced a distributed Markov approximation algorithm which is linear to the number of users in order to determine the services provisioning configurations. Zeng et al. [14] formulated an optimization problem to jointly decide the service provisioning policy and the routing decision, and they developed an online distributed algorithm with provable performance guarantees in terms of convergence and competitive ratio. Li et al. [15] focused on the service migration problem for mobile users through modeling a Markov Decision Process (MDP) model, and they solved it by using deep reinforcement learning. In addition, some works consider using the information of the prediction. Liu et al. [16]. introduced a prediction-based dynamic task assignment algorithm that assigned the workloads to edge servers based on the prediction of capacities and costs in each time slot. Jin et al. [17] designed a set of novel polynomial-time algorithms to make adaptive decisions by solving continuous solutions. These continuous solutions are based on the predicted inputs about the dynamic and uncertain cloud-edge environments via online learning. Ma et al. [18] propose a multiple slots predictive service placement algorithm to incorporate the prediction of user mobility based on a frame-based design. However, these works do not take into account the impact of additional prediction error on the service provisioning. In this paper, we study the online service provisioning and updating problem in mobile edge computing. Our objective is to improve the QoS by minimizing the total delay while considering maintaining the long-term cost under the constraint.

III. Model and Problem Formulation

1) System Model: As shown in Figure 1, we consider a three layer network architecture that includes the cloud data center, edge servers, and the mobile end-users. We suppose that the services required by users are initially provisioning in the cloud data center, which is denoted as set $\mathbf{S} = \{s_h\}$. Let $\mathbf{M} = \{m_j\}$ denote a substrate set of edge servers that supported by the operators. Let $\mathbf{U} = \{u_i\}$ denote the set of mobile users, and these users subscribe to the services one-to-one. In order to capture the mobility of users, we assume that the system in a slotted structure and its timeline is discretized into time frame $t \in \{0, 1, 2, ...T-1\}$ [18]–[20]. In this paper, we suppose that users move erratically and frequently among several edge servers. At each time slot, the operators determine whether provisioning replications or migration follow with users according to navigating the trade-off between delay and cost.

2) QoS model: In our study, the QoS of users is determined by computing delay, communication delay, and updating delay. We use $\mathbb{D}(t) = \sum_{i=1}^{|\mathbf{U}|} \mathbb{D}_{u_i}(t)$ to denote the total delay at time slot t, where $\mathbb{D}_{u_i}(t)$ is the delay of u_i. The computing delay is defined as $D^c_{u_i}(t) = \sum_{m_j \in \mathbf{M}} \frac{r_{u_i}(t)}{z^c_{m_j}}$, where $r_{u_i}(t)$ is the service request of user u_i at time slot t, and $z^c_{m_j}$ is the computing capacity of m_j measured by the number of CPU cycles. We use $D^l_{u_i}(t)$ to represent the communication delay produced when users are far away from the location of the service. Let t_{u_i,m_j} denote the maximum transmission rate, where $t_{u_i,m_j}(t) = b_{u_i,m_j}(t) \cdot \log_2(1 + \frac{\beta \cdot g(u_i,m_j)}{N})$ [20], [21]. The communication delay is defined as $D^l_{u_i}(t) = \sum_{m_j \in \mathbf{M}} \frac{d_{u_i}(t)}{t_{u_i,m_j}(t)}$, where $d_{u_i}(t)$ denotes the data size of the request [20], [21]. We use $D^u_{u_i}(t)$ to represent the updating delay, which occurs when the location of service s_i that is serving u_i changes. Here, we consider two scenarios. One is that the operator can place a replication on the edge server to which u_i is currently connected. The other is that operator can migrate service s_i to the edge server to which user u_i goes forward. The costs of both scenarios are discussed in the next subsection. The updating delay is defined as $D^u_{u_i}(t) = \Upsilon(v_i) + \Psi(s_i)$, where $\Upsilon(s_i)$ is the delay of rebooting software resources, and $\Psi(s_i)$ is the delay of transmission service profiles [22].

3) Cost Model: We use $\mathbb{C}(t)$ to denote the total cost of users in set $\mathbf{U}$ at time slot t, where $\mathbb{C}(t) = \sum_{h=1}^{|\mathbf{S}|} \mathbb{C}_{s_h}(t)$. Let $\mathbb{C}_{s_h}(t)$ denote the cost of service s_h, where $\mathbb{C}_{s_h}(t) = C^m_{s_h}(t) + C^r_{s_h}(t)$. We use $C^m_{s_h}(t)$ and $C^r_{s_h}(t)$ to represent the migration cost and replication cost, respectively. Let $x_{s_h}(t)$ denote the decision of s_h, when s_h decides to stay at the edge server same with the location in the previous step, $x_{s_h}(t) = 0$, otherwise, $x_{s_h}(t) = 1$.

A. Problem Formulation

On the basis of the models above, our problem is formulated to minimize the long-term average delay under the resource and cost constraints, which is shown as follows:

$$\mathbf{P}_1 : \text{minimize} \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^{|U|} \mathbb{D}_{u_i}(t) \tag{1}$$

$$\text{s.t.} \quad \mathbb{D}_{u_i}(t) = D^c_{u_i}(t) + D^l_{u_i}(t) + x_{s_i}(t) \cdot D^u_{u_i}(t), \tag{2}$$

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T} \sum_{h=1}^{|\mathbf{S}|} \mathbb{C}_{s_h}(t) \leq \overline{\Gamma}, \mathbb{D}_{u_i}(t) \leq \overline{D}, \forall u_i \in \mathbf{U}, \tag{3}$$

$$\sum_{\mathbf{S}_{m_i} \in \mathbf{S}} W(\mathbf{S}_{m_i}(t)) \leq R^s_{m_i}, \forall m_i \in \mathbf{M}, \tag{4}$$

$$x_{s_h}(t) \in \{0, 1\}, \forall s_h \in \mathbf{S}. \tag{5}$$

$\mathbf{P}_1$ is the objective function, and equations (2) to (5) are the constraints. Equation (2) is the total delay of each user, which needs to be lower than $\overline{D}$ to ensure the QoS. Equation (3) states that the long-term average cost cannot overwhelm the threshold $\overline{\Gamma}$. Equation (4) states the constraint on the resource, which means the services placed on m_i should be under the limitation $R^s_{m_i}$. Equation (5) states the decision of s_h which provides service for u_h at time slot t.

IV. Service Updating Decision Strategy based on Lyapunov Optimization

A. Decoupling based on Lyapunov Optimization

In this subsection, we first decouple the original problem into per-frame deterministic problems by applying the Lyapunov optimization. In order to deal with the constraint on average cost $\overline{\Gamma}$ in Equation (3), we introduce a virtual queue $Q(t)$ which denotes the historical measurement of the extra cost of services at time slot t. The queue updates according to

$$Q(t+1) = \max\{Q(t) + \mathbb{C}(t) - \overline{\Gamma}, 0\} \tag{6}$$

Intuitively, the condition of the total extra cost $\mathbb{C}(t)$ that is produced by the replication or migration of services can be evaluated by $Q(t)$. When the value of $Q(t)$ is large, it represents that the cost has exceeded the long-term cost $\overline{\Gamma}$. Specifically, Equation (6) implies $Q(t+1) \geq Q(t) + \mathbb{C}(t) - \overline{\Gamma}$, and then we have $\mathbb{C}(t) - \overline{\Gamma} \leq Q(t+1) - Q(t)$. By summing this inequality during all time slots, we have $\sum_{t=0}^{T-1}(\mathbb{C}(t) - \overline{\Gamma}) \leq$

$Q(T) - Q(0)$. Initialize $Q(0) = 0$ and divide by t time slots. One can take expectations and derive that the expected backlog over time slot in $[0, T-1]$ is less than the threshold.

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\mathbb{C}(t)] \leq \lim_{T\to\infty}\frac{1}{T}\mathbb{E}[Q(T)] + \overline{\Gamma} \tag{7}$$

As shown in Equation 7, we have that the constraint on the cost can be guaranteed by stabilizing the virtual queue $Q(t)$. Therefore, a quadratic Lyapunov function for each slot t is defined as $L(Q(t)) \triangleq \frac{1}{2}Q(t)^2$ [13], [18], [23], where $Q(t)$ is a vector that evolves over slots in $[0, T-1]$. Here, the quadratic Lyapunov function can be considered as a scalar measure of queue deviation which is similar to $Q(t)$. In order to keep the queue stable, which means enforcing the extra cost constraint by promoting the Lyapunov function to lower values continuously, we introduce the one-step conditional Lyapunov drift as follows.

$$\Delta(Q(t)) \triangleq \mathbb{E}[L(Q(t+1)) - L(Q(t))|Q(t)] \tag{8}$$

***Lemma** 1:* Given the updating decisions of services in set $\mathbf{S}$ according to multiple mobile users $\mathbf{U}$ in each time slot t, the following statement holds:

$$\Delta(Q(t)) \leq \text{ß} + Q(t)\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})|Q(t)] \tag{9}$$

, where $\text{ß} \triangleq \frac{1}{2}(\tilde{\mathbb{C}}(t)^2 + \overline{\Gamma}^2)$.

Proof: We rearrange Equation (8) for a concise form, where $\Delta(Q(t)) \triangleq \mathbb{E}[L(Q(t+1)) - L(Q(t))|Q(t)] = \frac{1}{2}\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})^2|Q(t)] + Q(t)\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})|Q(t)]$. For each service, we use $\tilde{\mathbb{C}}_{s_h}(t)$ to denote the cost of updating the decision of s_h in set $\mathbf{S}$ by choosing the minimum delay of user $u_h \in \mathbf{U}$ at time slot t. Based on that, the total cost of all services will be $\tilde{\mathbb{C}}(t) = \sum_{s_h \in \mathbf{S}}\tilde{\mathbb{C}}_{s_h}(t)$. Since the division of the time space taking into account the user's mobility on the boundary, the service provider will not change in one time slot. Thus, we have $\tilde{\mathbb{C}}(t) \geq \mathbb{C}(t)$. Then, we have $\Delta(Q(t)) \leq \frac{1}{2}(\tilde{\mathbb{C}}(t) - \overline{\Gamma})^2 + Q(t)\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})|Q(t)] \leq \frac{1}{2}(\tilde{\mathbb{C}}(t)^2 + \overline{\Gamma}^2) + Q(t)\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})|Q(t)]$. Therefore, we can obtain that the one-step conditional Lyapunov drift holds $\Delta(Q(t)) \leq \text{ß} + Q(t)\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})|Q(t)]$ at each time slot t, where $\text{ß} \triangleq \frac{1}{2}(\tilde{\mathbb{C}}(t)^2 + \overline{\Gamma}^2)$. ■

According to the Lyapunov optimization framework, we obtain the upper bound of the Lyapunov drift function by introducing a Lyapunov drift-plus-penalty function in each time slot t.

$$P(t) \triangleq \Delta(Q(t)) + V\mathbb{E}[\mathbb{D}(t)|Q(t)] \tag{10}$$

Here, we define V as a non-negative parameter for adjusting the trade-off between the extra cost queue and the delay. In each time slot, the performance of the service provisioning strategy is guaranteed by minimizing an upper bound of the following function.

$$P(t) \leq \text{ß} + Q(t)\mathbb{E}[(\mathbb{C}(t) - \overline{\Gamma})|Q(t)] + V\mathbb{E}[\mathbb{D}(t)|Q(t)] \tag{11}$$

Based on that, the service provisioning and updating problem is formulated by minimizing the right side of Equation (11) at each time slot, which is formulated as follows.

$$\mathbf{P}_2 : \text{minimize} \quad \text{ß} + Q(t)(\mathbb{C}(t) - \overline{\Gamma}) + V\mathbb{D}(t) \tag{12}$$

$$\text{s.t.}(2) - (5). \tag{13}$$

Algorithm 1 Updating Strategy with No Prediction (USNP)

```
Input:   Sets of edge servers $\mathbf{M}$, users $\mathbf{U}$, and services $\mathbf{S}$;
Output:  Service updating decision $\mathbf{X}(t)$ of $\mathbf{U}$ at time slot $t$;
1: for users $k = 1$ to $k = |\mathbf{U}|$ in $\mathbf{U}$ do
2:     Choose the updating decision by optimizing $\mathbf{P}_2$;
3:     for edge servers $i = 0$ to $i = |\mathbf{M}|$ in $\mathbf{M}$ do
4:         if $\sum_{\mathbf{S}_{m_i} \in \mathbf{S}} W(\mathbf{S}_{m_i}(t)) \geq R^s_{m_i}$ then
5:             Choose service by $i = \arg\min\{\eta_h\}$;
6:         end if
7:     end for
8: end for
9: return  Service updating decision $\mathbf{X}(t)$ of $\mathbf{S}$;
```

B. Optimal Services Updating Decision Strategy

In this subsection, we propose a service updating decision strategy by optimizing $\mathbf{P}_2$ under the constraints in each time slot. We start with a definition as follows.

***Definition** 1 (Optimal Service Updating (OSU) Problem):* Given the distribution of users $\mathbf{U}$, the topology of edge network $\mathbf{G}$, and the function $\Theta(t)$, an OSU problem is how to find a decision for services in $\mathbf{S}$ to minimize $\mathbf{P}_2$ under the constraints at time slot t.

On the basis of Definition 1, we discuss two scenarios. One is the services updating without prediction, and the other is the service updating with prediction.

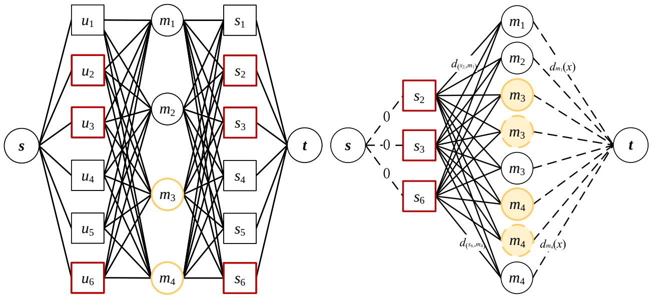
1) OSU with no prediction: The first scenario we considered is the OSU problem without available information caused by the inaccurate prediction results or in the initial or training stages of mobile users in per-slot. The specific steps are shown in Algorithm 1. We use the sets of edge servers $\mathbf{M}$, users $\mathbf{U}$, and services $\mathbf{S}$ as the input. The output is the service updating decision $\mathbf{X}(t)$ at time slot t. For each user in set $\mathbf{U}$, we choose the updating decision by optimizing $\mathbf{P}_2$ in lines 1 to 2. Then, we check the feasibility of services on edge servers by checking whether $\sum_{\mathbf{S}_{m_i} \in \mathbf{S}} W(\mathbf{S}_{m_i}(t)) \geq R^s_{m_i}$. Here, we use $\sum_{\mathbf{S}_{m_i} \in \mathbf{S}} W(\mathbf{S}_{m_i}(t))$ to denote the total number of services provisioning on m_i. In order to avoid conflicts caused by aggregation requests of multiple users, we introduce a definition of the conflict resolution factor for the service, and the specific definition is as follows.

***Definition** 2 (conflict resolution factor):* Let η_h indicate the conflict resolution factor of service s_h and $\eta_h = \mathbb{C}_{s_h}(t)/\overline{\mathbb{D}^l_{u_h}(t)}$, where $\overline{\mathbb{D}^l_{u_h}(t)} = \mathbb{D}^l_{u_h}(t)|_{s_h \notin \mathbf{S}_{m_i}(t)}$.
Here, we use $\mathbb{C}_{s_h}(t)$ to denote the total extra cost of service s_h when it migrates or replicates on edge server m_i at time slot t, where $s_h \in \mathbf{S}_{m_i}(t)$. In line 4, we choose a service by an increasing order $i = \arg\min\{\eta_h\}$. Finally, the service updating decision $\mathbf{X}(t)$ is returned in line 6.

2) OSU with prediction:

***Lemma** 2:* The decision of the OSU problem can be solved by minimizing $\Theta(t)$, where $\Theta(t) = Q(t)\mathbb{C}(t) + V\mathbb{D}(t)$.

Proof: We first rearrange $\mathbf{P}_2$ by introducing an intermediate variable $\mathbb{P}$, where $\mathbb{P}(t) = \text{ß} + Q(t)(\mathbb{C}(t) - \overline{\Gamma}) + V\mathbb{D}(t) = \text{ß} + Q(t)\mathbb{C}(t) - Q(t)\overline{\Gamma} + V\mathbb{D}(t)$. The value of ß is related to



(a) original connectivity graph. (b) extracted connectivity graph.

Fig. 2. The connectivity graphs of Fig 1.

the distribution of users in set $\mathbf{U}$ which is a constant value. Meanwhile, the value of $Q(t)$ depends on the decision of services in the previous time interval $[0, t-1]$, which means that the decision at time slot t has no effect on the value of $Q(t)$. We reconstruct $\mathbb{P}(t)$ as $\mathbb{P}(t) = W + \Theta(t)$, where $W = ß + Q(t) - \overline{\Gamma}$ and $\Theta(t) = Q(t)\mathbb{C}(t) + V\mathbb{D}(t)$. Therefore, we can obtain that the network determines the service updating strategies by solving the optimization of $\Theta(t)$ in each time slot. ■

Based on the conversion above, we rearrange $\Theta(t)$ by considering the combinational decision-making where $\Theta(t) = Q(t)\sum_{h=1}^{|\mathbf{S}|}\mathbb{C}_{s_h}(t) + V\sum_{i=1}^{|U|}\mathbb{D}_{u_i}(t)$. The value of the total extra cost of service s_h depends on the decision choosing to migrate or place replications, i.e., $\mathbb{C}_{s_h}(t) = C_{s_h}^m(t) + C_{s_h}^r(t)$, which will affect the result of the delay. Taking the decision of s_h as an example, if service s_h decides to migrate or place replications on other edge servers, it will produce a migration cost $C_{s_h}^m(t)$ or replication cost $C_{s_h}^r(t)$. Meanwhile, the communication part of $D_{u_i}^l(t)$ will decrease while the updating part of $D_{u_i}^u(t)$ will increase for $\mathbb{D}_{u_i}(t)$. We reconstruct $\Theta(t)$ on the basis of the interaction based on the relationship between services and users, where $\Theta(t) = \sum_{h=1}^{|\mathbf{S}|}\Theta_h(t)$. For each service, we have $\Theta_h(t) = Q(t)\mathbb{C}_{s_h}(t) + (D_{u_h}^l + D_{u_h}^u) + D_{u_h}^c$.

Based on that, we use $d_{(s_h,m_i)}$ to represent the weight between service s_h and edge server m_i at time slot t, where $d_{(s_h,m_i)}(t) = Q(t)\mathbb{C}_{s_h}(t)$. We suppose that $d_{m_i}(x)$ is the delay function. We replace in $\mathbf{G}^\circ$ each edge with $|\hat{\mathbf{U}}(t)|$ parallel edges between the same server m_i and the destination t, and each with weight $d_{m_i}(x)|_{u_x \in \hat{\mathbf{U}}(t)}$. Then, the weight between edge server m_i and the destination t that is connected to it is $d_{m_i}(x) = (D_{u_x}^l + D_{u_x}^u) + D_{u_x}^c$. Therefore, we have $\Theta_h(t) = d_{(s_h,m_i)}(t) + d_{m_i}(x)(t)$.

On the basis of the interaction, we propose a novel Updating Strategy with no Prediction (USNP) to optimize the provisioning strategy at each time slot, which is shown in Algorithm 2. We first construct a weighted graph by considering the information and connection between services and edge servers. We add two virtual nodes which are source **s** and destination **t**, and the middle two layers are services and storage resources of edge servers. The original connectivity graph is shown in Figure 2(a). We use thick red lines to mark services where their users are far away from the original locations, and thick yellow lines to mark edge servers with remaining resources. In each time slot, the activities of users are independent. This means that the locations of some users may be remaining in their original locations, while some may be far away from the connected edge servers. For the users whose locations are not changing, the corresponding service will not be migrated or placed by a replica, so there is no extra cost or delay produced. Therefore, we consider optimizing the provisioning of services by constructing an activity set $\hat{\mathbf{U}}(t)$ to reduce the dimensional space. The formal definition is given as follows.

***Definition** 3 (Activity Set):* Let $\hat{\mathbf{U}}(t)$ indicate the activity set of users at time slot t, where $u_i \in \hat{\mathbf{U}}(t)$ is the user whose current location $L_{u_i}(t)$ is going far away from the edge server for initial connection $L_{u_i}(t-1)$.

Here, we use $L_{u_i}(t)$ to denote the edge server that user u_i becomes connected to at time slot t. Since one user can only be served by one service, the numbers between users and services are equal. Based on that, we do an extraction by considering the current status of users and the topology of the edge network. The extracted connectivity graph is shown in Figure 2(b). We use the white circle to indicate that a container on the edge server has been occupied while a yellow one indicates a free storage resource on the edge server.

Algorithm 2 Updating Strategy with Prediction (USP)

```
Input:  Sets of edge servers $\mathbf{M}$, users $\mathbf{U}$, and services $\mathbf{S}$;
Output: Service updating decision $\mathbf{X}(t)$ of $\mathbf{S}$ at time slot $t$;
 1: Construct the original connectivity graph $\mathbf{g}$ based on the
    provisioning of $\mathbf{S}$, the connections of $\mathbf{G}$, and $\mathbf{U}$;
 2: for users $i = 1$ to $i = |\mathbf{U}|$ in $\mathbf{U}$ do
 3:     Calculate $\varsigma_{u_i}(t) = (L_{u_i}(t-1), L_{u_i}(t))$;
 4:     if $\varsigma_{u_i}(t)$==1 then
 5:         Construct the activity set with $\hat{\mathbf{U}}(t) \leftarrow u_i$;
 6:         Update user set at time slot $t$ with $\mathbf{U}(t) = \mathbf{U}(t)/u_i$;
 7:     else
 8:         Update $\mathbf{U}(t) \leftarrow u_i$;
 9:     end if
10: end for
11: Construct the extracted connectivity graph $\mathbf{G}^\circ$ based on
    the activity set $\hat{\mathbf{U}}(t)$;
12: Replace the link with $|\hat{\mathbf{U}}(t)|$ parallel ones with weight
    $d_{m_i}(x)|_{u_x \in \hat{\mathbf{U}}(t)}$;
13: Find a feasible service updating decision with min-cost
    flow of $\hat{\mathbf{U}}(t)$;
14: return Service updating decision $\mathbf{X}(t)$ of services $\mathbf{S}$;
```

The specific steps are shown in Algorithm 2. We use the sets of $\mathbf{M}$, $\mathbf{U}$, and $\mathbf{S}$ as the inputs. The service updating decision $\mathbf{X}(t)$ of $\mathbf{S}$ at time slot t is used as the output. We construct the original connectivity graph $\mathbf{g}$ based on the provisioning of $\mathbf{S}$, the connections of $\mathbf{G}$, and $\mathbf{U}$ in line 1. In line 2, we start to construct the activity set $\hat{\mathbf{U}}(t)$. We first check the locations of users in set $\mathbf{U}$, where $\varsigma_{u_i}(t) = (L_{u_i}(t-1), L_{u_i}(t))$ in line 3. If $\varsigma_{u_i}(t) = 1$, this denotes that u_i has gone away from the edge server at time slot $t-1$. Then, we construct the activity set by adding u_i into set $\hat{\mathbf{U}}(t)$, where $\hat{\mathbf{U}}(t) \leftarrow u_i$; Otherwise, it denotes that u_i always stays near the edge server from $t-1$

to t, and we update $\mathbf{U}(t) \leftarrow u_i$. Based on this, we start to construct the extracted connectivity graph $\mathbf{G}^\circ$ based on the activity set $\hat{\mathbf{U}}(t)$ in line 9. In line 10, we replace the link with $|\hat{\mathbf{U}}(t)|$ parallel ones with weight $d_{m_i}(x)|_{u_x \in \hat{\mathbf{U}}(t)}$ between edge servers and destination t. Then, we find a feasible service updating decision with min-cost flow of $\hat{\mathbf{U}}(t)$ and return the updating decision $\mathbf{X}(t)$ of services $\mathbf{S}$ in line 12.

Algorithm 3 Online Optimization of Service Provisioning strategy (O-OSP$_\omega$)

```
Input:  Sets of edge servers $\mathbf{M}$, users $\mathbf{U}$, and services $\mathbf{S}$;
Output: Service updating decision $\mathbf{X}$ of $\mathbf{S}$ in each time slot;
1:  for $t = 0$ to $t = \tau$ do
2:      Get service updating decision $\mathbf{X}(t)$ using Algorithm 1;
3:  end for
4:  for $t = \tau$ to $t = T - 1$ do
5:      Get service updating decision $\mathbf{X}(t)$ using Algorithm 2
        based on $\widehat{\mathbf{L}}_{\mathbf{U}|[t,t+\omega]}$;
6:      Set $\tilde{t} = (t - \tau) \mod \omega$;
7:      if $\tilde{t}$ = $t - \tau$ then
8:          Set $\mathbf{X}(t) = \mathbf{X}(\tilde{t})$;
9:      else
10:         for service $h = 1$ to $h = |\mathbf{S}|$ do
11:             Updating the decision value of $s_h$ into $A_{s_h}^{(\omega)}$;
12:             Calculate the decision policy frequencies;
13:             Set $X_{s_h}(\tilde{t}) = \arg\max_{a^\circ \in A_{s_h}^{(\omega)}} \{\varrho_{s_h|\omega}^{a^\circ}\}$;
14:         end for
15:         Set $\mathbf{X}(t) = \{X_{s_h}(\tilde{t})|_{s_h \in \mathbf{S}}\}$;
16:     end if
17: end for
18: return Service updating decision $\mathbf{X}(t)$ of $\mathbf{S}(t)$;
```

V. Online Optimization of Service Provisioning Strategy

In this section, we design an Online Optimization of Service Provisioning Strategy (O-OSP$_\omega$) by utilizing the committed horizon control method with ω steps prediction. The main idea of O-OSP$_\omega$ is to leverage the prediction model to look forward the trajectories of users in multiple steps and use the information to realize the service provisioning. The specific steps are shown in Algorithm 3. We suppose that the information of the chosen prediction model in the first τ steps is unavailable. Thus, we get service updating decision $\mathbf{X}(t)$ using Algorithm 1 in line 2. After that, we obtain the service updating decision $\mathbf{X}(t)$ using Algorithm 2 based on $\widehat{\mathbf{L}}_{\mathbf{U}|[t,t+\omega]}$ in line 4. Here, $\widehat{\mathbf{L}}_{u_i|[\tau,\tau+\omega]}$ is the trajectory of user u_i in a ω time steps prediction window starting at time τ, where $\widehat{\mathbf{L}}_{u_i|[\tau,\tau+\omega]} = \{\hat{L}_{u_i}(\tau), \hat{L}_{u_i}(\tau+1), ..., \hat{L}_{u_i}(\tau+\omega)\}$. In line 5, we set $\tilde{t} = (t - \tau) \mod \omega$, and we check whether the prediction steps are less than ω. In lines 9 to 13, we update the service provisioning for services by introducing a novel factor feasible decision frequency. We use $a_{s_h}(t) = x_{s_h}(t) \cdot y_{s_h}(t)$ to represent the decision value of s_h, where $y_{s_h}(t) \in \{-1, 1\}$ and $x_{s_h}(t) \in \{0, 1\}$ shown in equation (5). Since $x_{s_h}(t) = 0$ when service s_h decides to stay at the original location, the decision value will be $a_{s_h}(t) = 0$. On the contrary, when service s_h makes a decision on migration, $y_{s_h}(t) = -1$ and $x_{s_h}(t) = 1$, and then the value of $a_{s_h}(t) = -1$. Similarly, when service s_h makes a decision on replication, $y_{s_h}(t) = 1$ and $x_{s_h}(t) = 1$, and then the value of $a_{s_h}(t) = 1$. Based on that, we use a queue $A_{s_h}^{(x)}$ to record the decision values of service s_h in x time steps, i.e., $A_{s_h}^{(\omega)} = \{a_{s_h}(t+1), a_{s_h}(t+2), ..., a_{s_h}(t+\omega)\}$.

***Definition** 4 (feasible decision frequency):* Let $\varrho_{s_h|\omega}^{a^\circ}(t)$ indicate the feasible decision frequency of s_h under the value a°, where $\varrho_{s_h|\omega}^{a^\circ}(t) = \frac{1}{\omega}\sum_{x=0}^{x=\omega-1} f(A_{s_h}^{(x)}, a^\circ)$.

Here, $f(A_{s_h}^{(x)}, a^\circ)$ is a function to indicate whether the result in queue $A_{s_h}^{(x)}$ is equal to a°, i.e., $a_{s_h} = a^\circ$.

***Theorem** 1:* By applying OSP, the time-average system delay satisfies: $\frac{1}{T}\sum_{t=0}^{t=T-1} \mathbb{D}(t) \le \frac{1}{2}(OPT + ß + V|\mathbf{U}|\overline{D}) + \epsilon + \frac{1}{\omega} W \cdot \alpha \cdot T$.

Proof: We conduct the proof via introducing $\mathbb{P}_{OSP}(t)$, where $\mathbb{P}_{OSP}(t) = ß + Q(t)(\mathbb{C}(t) - \overline{\Gamma}) + V\mathbb{D}(t)$ under the OSP strategy. For each time slot, we use $\mathbb{P}(t)$ to represent the decision policy with random frequency. We use $\delta(t)$ to denote the prediction error at time slot t. Then, we have the average value $\frac{1}{\omega}\sum_{t+1}^{t+\omega} b(t) \le \frac{1}{\omega} \cdot \omega \cdot \epsilon = \epsilon$. Thus, we have

$$\mathbb{P}_{OSP}(t) \le \frac{1}{\omega}\sum_{t+1}^{t+\omega} \mathbb{P}(t) \le OPT + 2\epsilon + \frac{2}{\omega} W \cdot \alpha \cdot T, \quad (14)$$

which can be obtained in [24]. Here, $W = \max_{m_i \in \mathbf{M}}\{W(\mathbf{S}_{m_i}(t))\}$, which denotes the maximum available storage resource of edge servers. Since each server needs to make decisions for the services that are placed on it, there exists a stationary and randomized policy π for $\mathbf{P}_2$ that satisfy $\mathbb{E}[\mathbb{C} - \overline{\Gamma}] \le \delta$. Thus, we have $\mathbb{P}_{OSP}(t) \le ß + Q(t) \cdot \delta + V\mathbb{D}(t)$. By letting δ go to zero, we have $\mathbb{P}_{OSP}(t) \le ß + V\mathbb{D}(t)$.

$$\mathbb{P}_{OSP}(t) \le ß + V\mathbb{D}(t) \le ß + V|\mathbf{U}|\overline{D}. \quad (15)$$

We sum the inequality Equations (14) and (15), we have

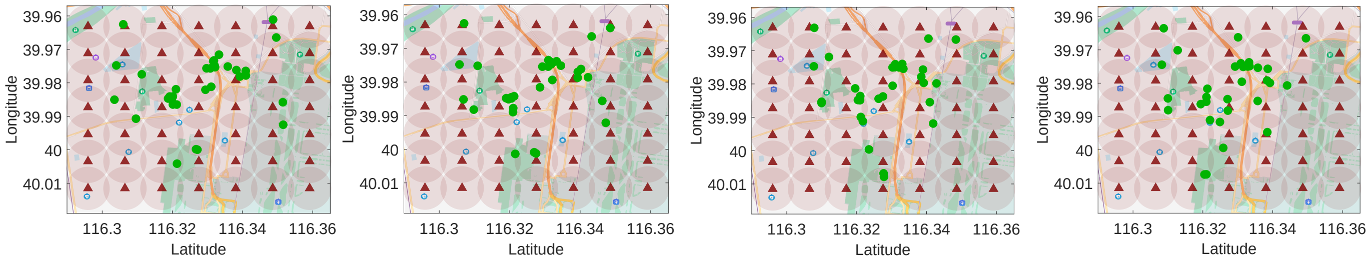
$$\mathbb{P}_{OSP}(t) \le \frac{1}{2}(OPT + ß + V|\mathbf{U}|\overline{D}) + \epsilon + \frac{1}{\omega} W \cdot \alpha \cdot T. \quad (16)$$

■

VI. Experiments

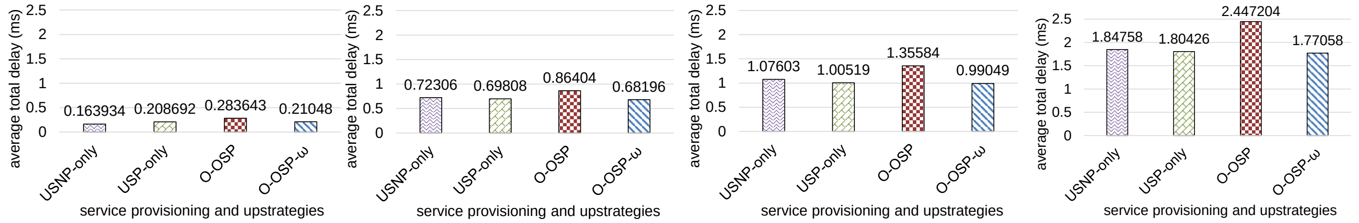
A. Basic Setting

We build our prototype on a workstation that runs a Linux operating system with E5-2620 CPU, NVIDIA RTX5000 GPU, 128Gb memory, and a 2Tb hard disk. We choose the Social-LSTM model to predict the future trajectories of users which can achieve an average accuracy of over 70%. We used the published Microsoft GPS trajectory dataset which has been collected in the Geolife project [25], [26]. Since this dataset recorded 182 users' outdoor trajectories in a broad range, we process it according to the features of users' activities. We first observed the activity tracks of 182 users and marked the longitude and latitude of the origin center coordinates $[116.327544, 39.987317]$. Then, we take this location as the central point and divide the area by a radius of 2.5 kilometers.



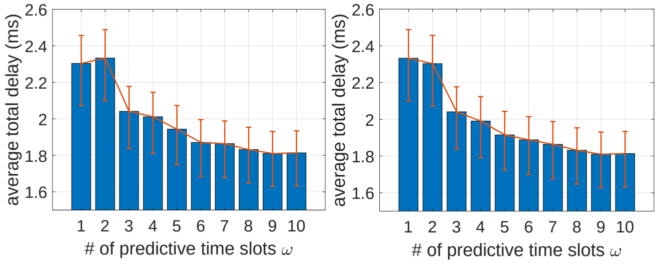
(a) time slot 0. (b) time slot 20. (c) time slot 40. (d) time slot 59.

Fig. 3. Users distribution at different time slots.



(a) # of users (10). (b) # of users (20). (c) # of users (30). (d) # of users (40).

Fig. 4. Average total delay under different strategies.



(a) group with 71.7% accuracy. (b) group with 56.6% accuracy.

Fig. 5. Average total delay with different ω.

We traverse user trajectories to find the ones within this range of area during 60 consecutive time slots. Based on that, 40 users were selected to construct our dataset $\mathbf{U}$. The distribution of users in different time slots is shown in Figure 3, which includes the initial locations in time slots 0, 20, 40, and 59 in Figures 3(a), (b), (c), and (c). We found that the location of users varies in different time slots, however, the number of connected users will remain at a high level for edge servers with a high frequency of utility. Based on that, we simulate the edge computing network based on $\mathbf{U}$, and we set up 49 edge servers with the service range of 450 meters. We set the computing capacity of each server to range from 2GHz to 5GHz, and the data size of each service is 1GB. The storage of each edge server ranges from 5GB to 10GB, which also denotes the number of services that can be placed on edge servers. Compared with the proposed online service provisioning strategy, three baselines are used.

- USNP-only: Services provisioning and updating without using the prediction information, and the decisions are only made by USNP.
- USP-only: Services provisioning and updating by using the prediction information, and the decisions are only made by USP.
- O-OSP: Online services provisioning and updating based on O-OSP$_\omega$ without considering ω steps prediction.

B. Experiment Results

1) Average total delay under different strategies: We investigate the average total delay under these four strategies with three groups of users in a 60 timescale. The results are shown in Figure 4. In addition, we have the following observations. (i) The numbers and trajectories of users in set $\mathbf{U}$ affect the results of strategies. As shown in Figures 4(b), (c), and (d), O-OSP$_\omega$ has the lowest average total delays under the groups of 20, 30, and 40. Meanwhile, the average total delay decreases notably with the increasing number of users. However, as shown in Figure 4(a), USNP-only obtains the lowest delay in the group with 10 users. On the one hand, there are abundant resources when there are fewer users which leads to deviation in the decision-making under the prediction trajectory. On the other hand, we found that the trajectories for the selected 10 users in group one hardly changed which results in errors in the algorithms using the prediction information. (ii) Prediction with ω slots in O-OSP$_\omega$ can effectively reduce the problem of service quality degradation caused by erratic activities of mobile users. As shown in Figure 4(d), the average total delay of O-OSP becomes significantly higher than that of the other algorithms. In this case, besides the lowest average total delay of O-OSP$_\omega$, USNP-only and UPS-only can also achieve better performances. The reason is that the increase in delay is due to the scaling of users under limited resources. Especially in the case of the trajectories of users changing frequently, it may be inappropriate to determine the location of the service only by one step, which will affect the delay of other users.

2) Average total delay with different ω time slots: Based on the compared results above, we study the average total delay of O-OSP$_\omega$ with different predictive ω slots. We predict the trajectories of users using the Social-LSTM model in multiple groups, and we choose two groups with 71.7% and 56.6% percent accuracy for the comparative experiments. The results are shown in Figure 5. Additionally, we have the following observations. (i) The value of ω can influence the efficiency of O-OSP$_\omega$. As shown in Figures 5(a) and 5(b), when the ω steps range from 1 to 9, the average total delay of users keeps decreasing. For each group, we can see that there is an obvious change between $\omega = 2$ and $\omega = 3$. However, when the slots scale into $\omega = 9$ and $\omega = 10$, the average total delay does not change obviously. The reason for this is that the prediction of users' trajectories too far ahead of their movements may cause inaccurate results which may lead to invalid decisions. Therefore, the total average delay under the O-OSP$_\omega$ strategy decreases in a range with the increasing value of ω, and the setting of ω is related to the characteristics of users and the prediction model. (ii) The accuracy of the chosen prediction model has little effect on the results of O-OSP$_\omega$. As shown in Figure 5(a), the average total delay under $\omega = 1$ is different between these two groups. The group with higher accuracy obtains a lower delay compared to the other group. However, the gap between these two groups became narrower with the increase of ω. As shown in Figure 5(a), the average total delay under $\omega = 6$ in a group with 56.6% is basically the same. Therefore, we have that even if the accuracy of the prediction model is imprecise, O-OSP$_\omega$ still can obtain a better result.

VII. Conclusion

In this paper, we investigate the service provisioning and updating problem under the multiple-users scenario by improving the performance of services with the long-term cost constraint. We first decouple the original long-term optimization problem into a per-slot deterministic one by using Lyapunov optimization. Based on that, we propose two service updating decision strategies by considering the trajectory prediction conditions of users. Based on this, we design an online strategy by utilizing the committed horizon control method while looking ahead to ω slots predictions. We prove the performance bound of our online strategy theoretically in terms of the trade-off between delay and cost. Finally, we conduct extensive experiments based on the Microsoft GPS trajectory dataset, and we demonstrate the superior performance of the proposed algorithm.

References

[1] Tu, S., Waqas, M., Rehman, S. U., Mir, T., Halim, Z., & Ahmad, I. (2021). “Social phenomena and fog computing networks: A novel perspective for future networks,” IEEE Transactions on Computational Social Systems, 9(1), 32-44.

[2] Waqas, M., Tu, S., Halim, Z., Rehman, S. U., Abbas, G., & Abbas, Z. H. (2022). “The role of artificial intelligence and machine learning in wireless networks security: principle, practice and challenges. Artificial Intelligence Review,” 1-47.

[3] Dang, T. K., Mohan, N., Corneo, L., Zavodovski, A., Ott, J., & Kangasharju, J. (2021). ”Cloudy with a chance of short RTTs: analyzing cloud connectivity in the internet.” In Proceedings of the 21st ACM Internet Measurement Conference, pp. 62-79.

[4] Chen, Y., Wu, J., & Ji, B. (2018, September). “Virtual network function deployment in tree-structured networks,” In 2018 IEEE 26th International Conference on Network Protocols (ICNP) (pp. 132-142). IEEE.

[5] Siriwardhana, Y., Porambage, P., Liyanage, M., & Ylianttila, M. (2021). “A survey on mobile augmented reality with 5G mobile edge computing: architectures, applications, and technical aspects,” IEEE Communications Surveys & Tutorials, 23(2), 1160-1192.

[6] Salaht, F. A., Desprez, F., & Lebre, A. (2020). “An overview of service placement problem in fog and edge computing. ACM Computing Surveys (CSUR),” 53(3), 1-35.

[7] Yu, N., Xie, Q., Wang, Q., Du, H., Huang, H., & Jia, X. (2018, December). “Collaborative service placement for mobile edge computing applications,” In 2018 IEEE Global Communications Conference (GLOBECOM) (pp. 1-6). IEEE.

[8] Nezami, Z., Zamanifar, K., Djemame, K., & Pournaras, E. (2021). “Decentralized edge-to-cloud load balancing: Service placement for the Internet of Things,” IEEE Access, 9, 64983-65000.

[9] Zhang, G., Zhang, S., Zhang, W., Shen, Z., & Wang, L. (2021). “Joint service caching, computation offloading and resource allocation in mobile edge computing systems,” IEEE Transactions on Wireless Communications, 20(8), 5288-5300.

[10] Chen, H., Deng, S., Zhu, H., Zhao, H., Jiang, R., Dustdar, S., & Zomaya, A. Y. (2022). “Mobility-Aware Offloading and Resource Allocation for Distributed Services Collaboration,” IEEE Transactions on Parallel and Distributed Systems, 33(10), 2428-2443.

[11] Xu, J., Chen, L., & Zhou, P. (2018, April). “Joint service caching and task offloading for mobile edge computing in dense networks,” In IEEE INFOCOM 2018-IEEE Conference on Computer Communications (pp. 207-215). IEEE.

[12] Han, P., Liu, Y., & Guo, L. (2021). “Interference-aware online multicomponent service placement in edge cloud networks and its ai application,” IEEE Internet of Things Journal, 8(13), 10557-10572.

[13] Ning, Z., Dong, P., Wang, X., Wang, S., Hu, X., Guo, S., ... & Kwok, R. Y. (2020). “Distributed and dynamic service placement in pervasive edge computing networks,” IEEE Transactions on Parallel and Distributed Systems, 32(6), 1277-1292.

[14] Zeng, Y., Huang, Y., Liu, Z., & Yang, Y. (2020, June). “Online Distributed Edge Caching for Mobile Data Offloading in 5G Networks,” In 2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS) (pp. 1-10). IEEE.

[15] Li, Z., Jiang, C., & Lu, J. (2021, December). “Distributed Service Migration in Satellite Mobile Edge Computing,” In 2021 IEEE Global Communications Conference (GLOBECOM) (pp. 1-6). IEEE.

[16] Liu, E., Deng, X., Cao, Z., & Zhang, H. (2018, December). “Design and evaluation of a prediction-based dynamic edge computing system,” In 2018 IEEE Global Communications Conference (GLOBECOM) (pp. 1-6). IEEE.

[17] Jin, Y., Jiao, L., Qian, Z., Zhang, S., & Lu, S. (2021, May). “Learning for learning: predictive online control of federated learning with edge provisioning,” In IEEE INFOCOM 2021-IEEE Conference on Computer Communications (pp. 1-10). IEEE.

[18] Ma, H., Zhou, Z., & Chen, X. (2020). “Leveraging the power of prediction: Predictive service placement for latency-sensitive mobile edge computing,” IEEE Transactions on Wireless Communications, 19(10), 6454-6468.

[19] Ouyang, T., Zhou, Z., & Chen, X. (2018). “Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing,” IEEE Journal on Selected Areas in Communications, 36(10), 2333-2345.

[20] Lu, S., Wu, J., Shi, J., Lu, P., Fang, J., & Liu, H. (2022). “A Dynamic Service Placement Based on Deep Reinforcement Learning in Mobile Edge Computing,” Network, 2(1), 106-122.

[21] Taleb, T., Ksentini, A., & Frangoudis, P. A. (2016). “Follow-me cloud: When cloud services follow mobile users,” IEEE Transactions on Cloud Computing, 7(2), 369-382.

[22] Gao, B., Zhou, Z., Liu, F., & Xu, F. (2019, April). “Winning at the starting line: Joint network selection and service placement for mobile edge computing,” In IEEE INFOCOM 2019-IEEE conference on computer communications (pp. 1459-1467). IEEE.

[23] Neely, M. J. (2010). “Stochastic network optimization with application to communication and queueing systems,” Synthesis Lectures on Communication Networks, 3(1), 1-211.

[24] Comden, J., Yao, S., Chen, N., Xing, H., & Liu, Z. (2019). “Online optimization in cloud resource provisioning: Predictions, regrets, and algorithms,” Proceedings of the ACM on Measurement and Analysis of Computing Systems, 3(1), 1-30.

[25] Zheng, Y., Li, Q., Chen, Y., Xie, X., & Ma, W. Y. (2008, September). “Understanding mobility based on GPS data,” In Proceedings of the 10th international conference on Ubiquitous computing (pp. 312-321).

[26] Zheng, Y., Zhang, L., Xie, X., & Ma, W. Y. (2009, April). “Mining interesting locations and travel sequences from GPS trajectories,” In Proceedings of the 18th international conference on World wide web (pp. 791-800).