A dynamic service migration strategy based on mobility prediction in edge computing

Lanlan Rui, Shuyun Wang, Zhili Wang, Ao Xiong and Huiyong Liu

Abstract
Mobile edge computing is a new computing paradigm, which pushes cloud computing capabilities away from the centralized cloud to the network edge to satisfy the increasing amounts of low-latency tasks. However, challenges such as service interruption caused by user mobility occur. In order to address this problem, in this article, we first propose a multiple service placement algorithm, which initializes the placement of each service according to the user’s initial location and their service requests. Furthermore, we build a network model and propose a based on Lyapunov optimization method with long-term cost constraints. Considering the importance of user mobility, we use the Kalman filter to correct the user's location to improve the success rate of communication between the device and the server. Compared with the traditional scheme, extensive simulation results show that the dynamic service migration strategy can effectively improve the service efficiency of mobile edge computing in the user’s mobile scene, reduce the delay of requesting terminal nodes, and reduce the service interruption caused by frequent user movement.

Keywords
Edge computing, service migration, Lyapunov optimization, mobility prediction, Kalman filter

Date received: 27 February 2020; accepted: 18 January 2021

Handling Editor: Zheng Chang

Introduction
With the rapid development of mobile Internet, mobile data traffic is increasing at an alarming rate. According to a report by Cisco VNI, the total amount of mobile data will increase sevenfold from 2016 to 2021. The huge amount of data has increased the burden on the central network and caused higher transmission delay. Traditional cloud centers are overwhelmed and cannot meet the demand for new technologies such as virtual reality (VR), WIT120, and connected cars. In order to cope with the increasing mobile traffic, great progress has been made in recent years in providing closer cloud services to users.

Compared with traditional cloud computing models, edge computing can better support mobile computing and Internet of things (IoT) applications. According to Cisco’s statistics and forecast data, most cloud data are temporary data that do not need long-term storage. Migrating the computing and storage capabilities of servers to the network edge can greatly reduce the amount of temporary data processed at the network edge. Short distance edge services can effectively reduce network jitter and delay, and enhance the responsiveness of services; at the same time, in view of the...
insufficiency of cloud computing that cannot provide fine-grained control for user data access and use, edge computing. Can improve data security by providing dedicated use and storage infrastructure equipment for critical private data, and use operation restrictions.

Researchers have proposed many conceptual models, including micro cloud, Cloudlet, Fog Computing, and Follow-Me Cloud (FMC). The core of these models is to run applications and related processing tasks near mobile users, reducing network congestion, and improving service experience.

With the standardization of various architectures in 5G network, the concept of mobile cloud computing has gradually developed into the latest mobile edge computing (MEC) standard architecture. In 2014, the European Telecommunications Standards Institute (ETSI) established the MEC Industry Standards Group (ISG) to focus on the standardization of MEC in wireless access networks, which is depicted in Figure 1.

MEC has become a key technology to realize the bright vision of future network development. The contradiction between the limited total geographic deviation server and the mobility of the user terminal will lead to significant network performance degradation, which will lead to a sharp decline in quality of service (QoS) and even interruption. Therefore, for the ongoing edge service, it is difficult to ensure the continuity of the service.

In order to ensure continuity, the following decision should be made without interrupting the ongoing edge service: when an arbitrary user moves it out of the service area of the connected edge server, whether and where to migrate the ongoing edge service. We can (1) continue to run services on the current edge server and exchange data with mobile users through the core network or other edge servers or (2) migrate the service to another edge server covering the new area. The migration decision involves such issues as when to migrate and whether the migration destination is optimal.

In this article, a Dynamic awareness Service placement and Migration scheme in edge networks (DSM) is proposed where the distribution and set coverage of users is used to obtain the initial placement of services that meet the requirements of multiple users; this multi-service placement algorithm (MSPA) reduces the repeated placement of services as much as possible while meeting the experimental constraints of most users. In the process of user dynamic movement, by correcting the user’s location, using Lyapunov optimization to obtain a suitable service migration strategy, to achieve a balance between user-perceived delay and long-term cost.

The rest of this article is organized as follows. The motivation and the contributions of this article are presented in section “Motivations and contribution.” In section “System architecture and workflow,” we introduce the system model and workflow in this article. In section “Place service initially,” a multi-user service initial placement algorithm is proposed. In section “Dynamic service migration,” we introduce the dynamic service migration strategy based on Lyapunov and we propose a mobility-based service migration strategy. Section “Simulation and evaluation” describes simulations and results analysis of this article. The work of this article and prospects for possible future research are summarized in section “Conclusion.”

Motivations and contribution

MEC is now one of the key technologies supporting 5G operations. Services need to be dynamically placed in the edge network to ensure user service experience.

However, a new challenge is that the resources (processing power, storage space, etc.) of the MEC server are much smaller than traditional data centers. Therefore, due to cost considerations, it is impossible to delegate the services of each data center to edge servers. In other words, it is to find the minimum set of replicas that can satisfy the constraints of latency between users requesting the same service.

However, user mobility will bring challenges to service deployment. Noting that the delay perceived by the user is jointly determined by the communication delay. Therefore, if the service configuration file of each user is actively placed on the nearest MEC node, some MEC nodes may be overloaded, resulting in increased calculation delay. When the services are not used and do not migrate (i.e. the services are saved in the first server they are allocated), it will cause the distance between the service and the user to increase,
resulting in performance degradation. In summary, making appropriate migration decisions and shortening service migration time are essential to improve service continuity and service quality.

In this article, we focused on the service migration strategy in MEC, hoping to reduce service interruption caused by user movement and to ensure user QoS as much as possible to reduce transmission delay and improve service efficiency. The main contributions of this article are as follows:

An algorithm for the initial placement of services is proposed to reduce the repeated placement of services while meeting the delay constraints of most users. Build QoS and migration cost model and design a service migration strategy based on Lyapunov. Lyapunov is used to get the service migration decision set with minimum system delay in the case of guaranteeing long-term cost stability, that is better adapted to the mobile environment, reduce network latency, and improve service efficiency. Propose a method based on user mobility considerations to decide which edge server to migrate the service to. We consider the mobility of mobile nodes using the Kalman filter that is more mobile-oriented to ensure service continuity.

Related work

It is demonstrated that user movement in cellular network under FMC scenario can be modeled as Markov chain model. Through the analysis of data such as the average delay and migration cost for users to obtain services after virtual machine migration, it is proved that the cost caused by all migrations is greater than the cost caused by partial migrations, and as the distance between users and the optimal server increases, the migration cost gap will decrease.

Service migration problem under the mobile cloud computing scenario is modeled as a one-dimensional Markov decision process (MDP). In Sharma et al., the distance between the mobile user and micro cloud was defined as a state function, and the network costs were defined as a reward function. At the same time, the existence of the so-called migration optimal threshold was proved. When users reach the threshold, the benefits of service migration would be greater than the benefits of no service migration, and the decision would be made for service migration. Some researchers further optimized the above model, simplified the solving process of MDP, and achieved good results. In Zhang et al., the authors proposed a cloud service migration decision system and established an MDP model to cope with server performance degradation in mobile edge networks by actively migrating services. Through the state collection module, system senses and monitors the changes of response time of each server in the collection network, and predicts QoS according to the collected data. Specifically, if the predicted QoS exceeds a given QoS threshold, the QoS prediction module will trigger the search of the target server to migrate the service to another edge cloud that provides the best QoS. In order to reduce the large number of complex computing and statistical processes in the above MDP migration scheme, researchers such as IBM’s Rahul et al. designed a new MDP service migration solution, using the decoupling property of MDP to transform the constrained MDP problem into a simple deterministic optimization problem, which simplifies the calculation.

In addition to traditional heuristic algorithms, deep reinforcement learning has also been used in the field of edge computing in recent years. In Feng et al., a cooperative computation offloading and resource allocation framework for blockchain-enabled MEC systems is designed, and used A3C to solve cooperation computation offloading and resource allocation problem. In Zhang et al. proposed a deep Q-network for task migration in MEC system, which can learn the optimal task migration policy from previous experience without necessarily acquiring the information about users’ mobility pattern in advance. Florain et al. propose a general framework for optimizing edge service migration based on reinforcement learning techniques. Using their solution, the migration policies can be learned with respect to a large variety of optimization goals. Gao et al. designed a reinforcement learning–based framework for a single-user edge computing service migration system. And they take many requirements into account, but not user movement and link change. In Chen et al., Chen et al. proposed a service migration mechanism in Edge Cognitive Computing, in which the service is migrated based on the behavioral cognition of a mobile car. Besides the movement of cars, this article focused on the service awareness, including emotion detection and video streaming.

Some researchers have also considered some personalization factors. Huang investigates service migration in MEC by taking the risk of location privacy leakage into account, and formulates the service migration problem as an MDP and proposes an efficient algorithm to find the optimal solution that minimizes the long-term total cost. In Varasteh et al., a power and latency aware optimum cloudlet selection strategy was proposed for multi-cloudlet environment with a proxy server. Power consumption and latency are reduced by approximately 29%–32% and 33%–36%, respectively.

Through the study of service migration strategies in mobile edge network in recent years, we found that
most mobile migration strategies currently used MDP to model and solve mobile service migration strategies, but some of the scenarios were difficult to extend to 5G applications. In addition, most researchers often consider only part of the factors to decide whether to migrate while ignoring the user mobility on the performance of migration strategies. Therefore, we proposed MSPA and DSM to make up for these shortcomings.

System architecture and workflow

As shown in Figure 2, we consider a backend cloud contains a set \( M = \{1, 2, \ldots, M\} \) of MEC nodes, and the network controller is placed in the backend cloud.

The network control center collects all user request information, delivers the service to the edge node, and provides the service to the user through the edge server. After the requested service is deployed to the edge node, it detects the user status as the user moves. The move behavior determines whether to migrate the service to the specified server.

The environment for the applications to run on each mobile device is assigned to a dedicated virtual machine or container. To better capture the user mobility, the system is assumed to operate in a slotted structure and its timeline is discretized into time frames \( t \in T = \{0, 1, 2, \ldots, T\} \). At all discrete time slots, each mobile user sends a service request to the local MEC node, and then the network controller responds the request and determines the optimal node to migrate to serve the user.

Place service initially

In the process of users requesting services, service resources will be deployed in a large number of locations on the edge network, which can locate resources closer to users, reduce bandwidth costs, and increase availability. Faced with the massive amount of requested data, edge networks often need to place service copies in multiple locations on the premise of ensuring user service quality. Therefore, reducing the total cost of service deployment to ensure load balancing in the data center is one of the important issues that need to be resolved. However, there are a few studies in this area.

MSPA algorithm

In this problem, we assume that the network graph is \( G = (U, ES, W) \). Let the set of nodes \( U \) represents the users, \( ES \) be the set of edge servers, and \( W \) record the location of users and edge servers.

Assuming that each edge server has capacity \( C \), and the total number of replicas is \( R, p_{i,j}, r_{i,j} \in \{0, 1\} \) is used to indicate the initial placement, if the service \( s_i \) of user \( u_i \) is placed in the edge server \( es_j \), \( p_{i,j} = 1 \), and if the copy
of $s_i$ is allocated to $es_j$, $r_{i,j} = 1$. We use cost to represent the total storage cost, which is defined as

$$cost_s = \sum_{i=1}^{\vert U \vert} \varphi_i \sum_{j=1}^{\vert S \vert} r_{i,j}$$  \hspace{1cm} (1)$$

where $\varphi_i$ represents the storage cost of service $i$.

To ensure the experience of users, we also need to guarantee user-perceived latency, whose threshold of tolerate delay is $\text{threshold}_L$, while optimizing the cost, the latency can be defined as

$$l_{i,j} = \alpha \ast \text{dis}_{i,j}$$  \hspace{1cm} (2)$$

Thus, the network service placement in edge cloud problem is as follows

$$\min : cost_s$$  \hspace{1cm} (3)$$

C1 $p_{i,j} + r_{i,j} \leq 1$  \hspace{1cm} (4)$$

C2 $\sum_{i=1}^{\vert U \vert} p_{i,j} \leq C$  \hspace{1cm} (5)$$

C3 $\sum_{i=1}^{\vert U \vert} \sum_{j=1}^{\vert S \vert} r_{i,j} \leq R$  \hspace{1cm} (6)$$

C4 $l_{i,j} \leq \text{threshold}_L$  \hspace{1cm} (7)$$

Equation (4) ensures that each service can only be placed on one server once; equation (5) ensures that each data center has a capacity limit when saving users’ data; and equation (6) ensures that the total number of replicas cannot exceed the predefined limit. Equation (7) requires that the delay between the user and the service does not exceed a given threshold.

To solve this problem, we use latency constrained matrix to describe the coverage relationship of edge servers to users, which is defined as follows

$$LCM_{i,j} = \begin{cases} es_j \in ES, & \text{latency}(i,j) \leq \text{threshold}_L \\ \emptyset, & \text{otherwise} \end{cases}$$  \hspace{1cm} (8)$$

Then, we adopt a heuristic algorithm to obtain the minimum set of servers, so as to optimize the storage cost of edge network. Algorithm 1 presents the calculation process of service placement scheme. The input includes a set of users, tolerate latency, and $\vert ES \vert$ edge servers.

**Algorithm complexity**

The algorithm is divided into three steps. Step one establishes a discernibility matrix, step two performs a greedy search, and step three reduces the data center. The complexity of these three steps is $O(n^2)$, so the time complexity of algorithm is $O(n^2)$.

**Algorithm 1.** MSPA algorithm.

**Input** The set of $U$, the set of $ES$, tolerate latency, and services set $N$

**Output** Placement scheme $S$

**Initialization** $LCM_{\vert U \vert \times \vert U \vert} = \{\emptyset\}_{\vert U \vert \times \vert U \vert}$

1. for $i = 1$ to $\vert N \vert$ do
2. for $j = 1$ to $\vert U \vert$ do
3. for $k = 1$ to $\vert ES \vert$ do
4. $LCM_{i,j} = \{es_k \in ES | \text{latency}(i,j) \leq \text{threshold}_L\}$
5. end for
6. end for
7. end for
8. for $i = 1$ to $\vert U \vert$ do
9. $s(i) = \{\emptyset\}$
10. end for
11. COST = 0
12. for $i = 1$ to $\vert U \vert$ do
13. for $j = 1$ to $\vert ES \vert$ do
14. Statistics on the frequency of $es_k$ in line $i$ of a matrix
15. $Num(es_k)_{i,j}$
16. $cost(es_k)_i = (es_k)_i \ast \varphi_i$
17. end for
18. $\sum_{i=1}^{\vert U \vert} Num(es_k)_i = Num(es_k)$
19. $\sum_{i=1}^{\vert U \vert} \sum_{j=1}^{\vert S \vert} cost(es_k)_i = cost(es_k)$
20. end for
21. while $LCM_{\vert U \vert \times \vert U \vert} \neq \{\emptyset\}_{\vert U \vert \times \vert U \vert}$ do
22. $es_q = \arg \max \{Num(es_k)\}$
23. for $i = 1$ to $\vert U \vert$ do
24. for $j = 1$ to $\vert U \vert$ do
25. if $es_k \in LCM_{i,j}$ then
26. $s(i) = s(i)_{j}^{(es_k)}$
27. $COST = COST + cost(es_k)$
28. $Num(es_k)_{i,j} = Num(es_k)_{i,j} - Num(es_k)_i$
29. end if
30. $LCM_{i,j} = \emptyset$
31. end for
32. end for
33. return $S$
34. end while
35. end for
36. end for
37. return $S$

**Dynamic service migration**

**Dynamic migration decision**

Relying on edge computing technology, we can reduce the latency of network tasks and improve user QoS. However, frequent service interruptions caused by the movement of mobile nodes are still a bottleneck affecting network performance. Therefore, in order to improve the service efficiency, reduce the request delay of mobile nodes, and minimize the service interruption in the dynamic environment, cost-efficient trade-off should be considered.

In this section, network architecture, latency model, and costs model of MEC are introduced. We use $a_k(t)$
to indicate the migration decision taken by $ES_k$ node for the service $k$ during the state detection time slot $t$. When $a_k(t) = 1$, it means that $ES_k$ decides to migrate service $k$ at this time. When $a_k(t) = 0$, it means that $ES_k$ has decided not to migrate the service at this time and still runs the service locally.

$$a_k(t) = \begin{cases} 1 & \text{migration} \\ 0 & \text{no migration} \end{cases}$$ (9)

QoS model

Communication latency. In this article, we use response delay to present QoS. The response time between a mobile device and the MEC node generally contains the network propagation delay and the data transmission delay.

Since the network condition and the data transmission information in current time slot are available from the system-level perspective, we can extend the above cases into a general model, which we do not impose structural assumption on. Given the service request information as well as the current location of user, the communication delay to MEC node $i$ can be characterized by a general model $D_i(t)$.

Computing latency. Each MEC node may provide computing resource to multiple users and the computing latency of MEC nodes may not be ignored. Thus, computing latency should be concerned in the system model. We use $C_i(t)$ to the computing delay of MEC node $i$ at time slot $t$ and $C_i(t - 1)$ at time slot $t - 1$.

When considering the service migration decision $a_k(t)$, the user latency experienced by user can be further expressed as

$$\text{Delay}(t) = (1 - a_k(t))D_i(t) + (1 - a_k(t))C_i(t - 1) + a_k(t)C_i(t)$$ (10)

Cost model. In order to achieve satisfactory QoS, it is often necessary to migrate services among edges to follow the user mobility. However, it is worth noting that service migration across edge servers often brings additional operational costs. To model the operational costs incurred by cross-edge service migration, we use $E_k(t)$ to denote the costs of migrating the service $k$ from source MEC to the optimum MEC. Considering the service migration decision $a_k(t)$, the migration costs by user $k$ can be further expressed as

$$E(t) = a_k(t)E_k(t)$$ (11)

When to migrate

Since our goal is to minimize user latency with long-term average cost constraints to obtain performance-cost trade-off, which is difficult to directly solve in traditional methods, in this section we adopt the Lyapunov theory to obtain service migration decision-making in edge computing. We first convert the original problem to a queue stability control problem.

Input queue $Q(t)$ is defined as

$$Q(t + 1) = \max[Q(t) + a_k(t)E_k(t) - E_a, 0]$$

where $E_a$ is a long-term time-averaged costs and $Q(t)$ is the queue length at time $t$. The virtual queue can be regarded as a constraint on the migration costs. When the queue is stable, the transmission cost will meet the preset migration costs.

Lemma 1. If the virtual queues $Q(t)$ is rate stable, that is, $\lim_{T \to \infty} Q(T)/T = 0$, then we have $\lim_{T \to \infty} \sum_{i = 1}^{T} E_i(t)/T \leq E_a$.

Proof. We first rewrite equation (4) as follows

$$Q(t + 1) = \begin{cases} Q(t) + E(t) - E_a, & Q(t) \geq E_a - E(t) \\ 0, & \text{otherwise} \end{cases}$$

Then, we have

$$Q(t + 1) - Q(t) = \begin{cases} E(t) - E_a, & Q(t) \geq E_a - E(t) \\ -Q(t), & \text{otherwise} \end{cases}$$

By taking the average from time index 0 to time index $T - 1$ on both sides and letting $T$ goes to infinite, we have

$$\frac{Q(T) - Q(0)}{T} = \frac{1}{T} \sum_{i = 1}^{T} E_i(t) - E_a$$

If the virtual queue $Q(T)$ is rate stable, that is, $\lim_{T \to \infty} Q(T)/T = 0$, we have

$$\lim_{T \to \infty} \frac{\sum_{i = 1}^{T} E_i(T)}{T} - E_a \leq 0$$

$$\lim_{T \to \infty} \frac{\sum_{i = 1}^{T} E_i(T)}{T} \leq E_a$$

This completes the proof.

The quadratic Lyapunov function is defined as

$$L(t) = \frac{1}{2} Q^2(t)$$

$$L(t + 1) - L(t) = \frac{1}{2} [Q^2(t + 1) - Q^2(t)]$$

Then, we can get the Lyapunov drift function as
\[
\begin{align*}
Q^2(t + 1) &= \left( \max \{Q(t) - E_a + a_k(t)E_k(t)\} \right)^2 \leq Q^2(t) \\
&+ \left( E_a^2 + (a_k(t)E_k(t))^2 - 2Q(t)E_a - 2E_aE_k(t) \right) \leq Q^2(t) \\
&+ (E_a^2 + 2Q(t)*(a_k(t)E_k(t) - E_a) + E_{\text{max}})
\end{align*}
\]

Therefore, the drift function can be collated as
\[
L(t + 1) - L(t) \leq Q(t)(a_k(t)E_k(t) - E_a) + \frac{1}{2}(E_a)^2
\]

After constructing the input queue, our goal is to find a migration policy to coordinate the perceived latency and migration costs. So, a drift plus penalty function is defined as
\[
\begin{align*}
L(t + 1) - L(t) + VD &\leq Q(t)(a_k(t)E_k(t) - E_a) \\
&+ \frac{1}{2}(E_a)^2 + (1 - a_k(t))C(t) + (1 - a_k(t))(1 - C(t)) \leq C(t-1) \\
&+ a_k(t)C(t) = a_k(t)(Q(t)E_k(t) - V[D_i(t) \\
&+ C_i(t-1) - C_i(t)]) + \frac{1}{2}(E_a)^2 - Q(t)E_a
\end{align*}
\]

where the parameter \( V = 0 \) is the penalty weight, which represents the importance of the objective function compared to the constraints at every time slot.

Our goal is to minimize the value of the right expression. In order to simplify the problem, we omit the fixed value part such as \( (E_a)^2 \), \( Q(t)E_a \) at a certain moment. The problem can be simplified as

\[
\begin{align*}
\text{min} &\quad a_k(t)(Q(t)E_k(t) - V[D_i(t) + C_i(t-1) - C_i(t)]) \\
\text{s.t.} &\quad a_k(t) \in \{0, 1\}
\end{align*}
\]

Define \( X(t) \) as
\[
X(t) = (Q(t)E_k(t) - V[D_i(t) + C_i(t-1) - C_i(t)])
\]

We can calculate the value of \( X(t) \) and determine the best offloading strategy \( a_k(t) \) with
\[
a_k(t) = \begin{cases} 
1 & X(t) > 0 \\
0 & X(t) < 0
\end{cases}
\]

Note the optimal action \( a_k(t) \) is not static, since the reward function involves the real-time QoS that captures the dynamic link states. The optimal action \( a_k(t) \) will be updated with the input of the real-time QoS. Our proposed model reflects the dynamic link states. Therefore, the dynamic optimal actions generated maintain the long-term balance between QoS and costs.

**Select the server based on user mobility prediction**

With the presence of user mobility, it is intuitive to ensure a desirable level of QoS, the service should be actively migrated to follow the user mobility. However, frequent migration would incur excessive operational costs in return. Then, a natural question is how to navigate such a performance-cost trade-off in a cost-efficient manner, so a dynamic service migration strategy is introduced in section “Dynamic service migration.”

In this article, we use the Kalman filter to predict user mobility and decide where to migrate the service to ensure the connectivity between the terminal and the ES. A Kalman filter\(^{26,27}\) is a set of mathematical equations that efficiently estimate the state of a linear system that minimizes the estimated error covariance to reach optimization. The user state vector \( \hat{x} \) is defined by the set of data \( [p_x, v_x, p_y, v_y] \) which describes the user’s movement at time \( t \), where \( p_x \) and \( v_x \) correspond to the latitude point and the velocity, respectively, and \( p_y \) and \( v_y \) correspond to the longitude parameters of a user. Because user’s movement can be regarded as a linear model within short time, we utilize uniform linear motion model to measure the change in the state of the user within \( \Delta t \). Then, the state vector \( x_k \) and the observed value \( z_k \) vector are given by
\[
\begin{align*}
x_{k+1} &= Ax_k + \nu_k \\
z_k &= x_k + u_k
\end{align*}
\]

where \( \nu_k \) and \( u_k \) represent a normal probability distribution of the process white noise. To produce the estimated state \( x_{k+1} \) at time \( k \), the matrix \( x \) is multiplied by the state model \( A \)
\[
A = \begin{bmatrix} 1 & t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & t \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]

The Kalman filter works in two steps recursively at each time step \( \Delta t \): time update that predicts the next estimation of a current state, and measurement update that adjusts the current state estimation with actual measurements at time \( t \), they are defined as follows:

**Predict step**
\[
\begin{align*}
x_{k+1}^* &= Ax_k + w_k \\
P_{k+1}^* &= AP_kA^T + Q
\end{align*}
\]

where \( w_k \) represent a normal probability distribution of the process white noise. \( P_{k+1}^* \) is the priori estimate error covariance matrix, and \( Q \) is the process noise covariance.
Algorithm 2. Dynamic service migration strategy based on user mobility.

Input $D_i(t), E_i(t)$, network situation 
Output Service migration decision. 
Initialization Set the costs queue $Q(0) = 0$ at beginning 

End Initialization
1. for each time slot $t = 1, 2, \ldots, \infty$
2. predict user location using Kalman filter, select closest server as optimum server 
3. Calculate
   $X(t) = Q(t)E_i(t) - V[D_i(t) + C_i(t - 1) - C_i(t)]$
4. if $X(t) < 0$
5. do not to migrate service 
6. else
7. migrate service to optimum server selected by step 2 
9. end if 
10. Update the virtual queue 
11. $Q(t + 1) = \max \{Q(t) + a_i(t)E_i(t) - L_i, 0\}$ 
12. end for

Correct step

$K_{k+1} = P_{k+1}^{-1}(P_{k+1} + R)^{-1}$

$x_{k+1} = x_{k+1} + K(z_{k+1} - x_{k+1})$

$P_{k+1} = (I - K_{k+1})P_{k+1}$

In this step, $x_{k+1}$ is the posteriori state estimation at time $t$, $P_{k+1}$ is the posteriori estimate error covariance matrix, and $R$ represents the measurement noise.

In this article, the connectivity between the terminal and the ES is determined using outage probability

$$SNR(d) = P_{TX} - L(d) - N_0$$

$$f_{SNR}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{(x - (P_{TX} - L(d) - N_0))^2}{2\sigma^2} \right\}$$

$$\text{succ} = \frac{1}{2} \left( 1 - \text{erfc} \left( \frac{(I - P_{TX} - L(d) - N_0))^2}{2\sigma^2} \right) \right)$$

where $P_{TX}$ represents the transmission power, $L(d)$ is the path loss, $N_0$ is the noise power, and $\Gamma$ is the signal-to-noise ratio (SNR) threshold.

The process of the proposed ES selection can be described: when user is asking a task that is held on ES and migration has to be performed, predict the location of the user and select the closest ES from all connectable ESs.

Thus, our mobility-aware service migration can be given by Algorithm 2.

Simulation and evaluation

Simulation setting

We set our simulation in MATLAB and Python. In order to realize the real simulation of urban scene with parameters, in this article, specific node movement trajectories and scenes were constructed based on the movement trajectory data set obtained from global positioning system (GPS) tracking of taxis in San Francisco bay area by CRAWDAD community. When we set up the simulation environment, the base station is approximately evenly distributed and referring to the mobile operator’s base station laying and area coverage setting, we set the coverage of each ES to 200 m and the overlapping area of adjacent ES to 20 m. The hop distance between two MEC nodes is calculated by the Manhattan distance. To simplify the problem, we assume that the transmission delay between ES and user follows uniform distribution within $[0.9, 1.1]$ of the distance, and 1 unit per hop. We simulate 100 time slots for our system. The settings of network parameters such as service request, packet size, bandwidth, delay, and other parameters used in DSM algorithm were shown in Table 1.

In this section, we compare our algorithm with the following four other methods: (1) Service Offloading in SDN (SO) in which algorithm services migrate to the nearest MEC node at every opportunity over a long period of time, (2) the method which migration policy based on deep Q-learning (DQN) is applied, (3) never migrate service (non-migration), and (4) genetic algorithm is used to solve the problem of data placement in cloud for online social networks.

Results

Average response time. Average task response time is defined as $ART_i = \sum_{j=1}^{i} Time_j/i$, where $ART_i$ is the average response time of the first $i$ seconds, and $Time_j$ represents the response time of j seconds. The performance of the algorithm can be seen by comparing the average task response time when sending packets of different sizes.

In the simulation experiment, the data packet sizes are tested at 2M and 16M (Figures 3 and 4), respectively, and the task response time under different strategies is compared for comprehensive analysis and discussion.

As can be seen from Figures 5 and 6, the latency gradually decreases and becomes stable. The response time of the service of non-migration strategy is significantly larger than the other two strategies. Through analysis, we believe that the reason is as follows: since the mobile node keeps moving, the ES handover (base station handover) process will always occur. Sub-switching means a service migration, which will significantly increase the task transmission delay. The process of SO and DSM according to their respective designs does not have a serious impact on network traffic.

The average response time of DSM is better than that based on the SO scheme. Compared with the SO
scheme, the DSM predicts the user’s movement trajectory, taking into account the mobility of the user when requesting the service, so that the user gets the service response in a closer place, and the reduced user perception delay.

Table 1. Parameter setting.

| Parameters                      | Values |
|---------------------------------|--------|
| ES service coverage radius      | 200 m  |
| ES service overlap area         | 20 m   |
| Number of ESs                   | 5, 10, 20 |
| Speed                           | 10 m/s |
| Run time per simulation         | 1 s    |
| Request data size               | 1~64M  |
| Bandwidth                       | 100M   |

ES: edge server.

Figure 3. The average delay when the packet is 2M.

Figure 4. The average delay when the packet is 16M.

Figure 5. Connection success probability with different speeds.

Figure 6. Costs with different latency constraints.

Connection success probability. Figure 5(a)–(d) shows the success probability of task execution connection result when user is moved at speed 10, 20, 40, and 100 km/h. It shows that there is almost no difference in the success rate using the proposed method and the compared method when move speed is 10 km/h. This is because when user moves slowly, the moving distance of user during service processing at the selected ES is short and the connectivity of the ES at the requested timing can keep connectivity. When user speed is 20 km/h, it can
be observed that the connection success probability of DSM is better than SO and the average connection success probability of DSM is 15% higher than that of SO. Moreover, when user speed is higher and reaches 100 km/h, the success rate of the service connection of the SO solution has dropped sharply, while the success rate of the connection of the DSM solution can still remain stable.

In general, connection success probability decreases with the increase in speed, and the connection success probability of SO is more sensible to movement speed while DSM is more stable. Also, the connectivity becomes higher as the number of ESs increases because the density of ESs increases and the distance between the terminal and the ES used in the proposed method is shortened. This is because the longer the processing time is, the longer the moving distance of the terminal becomes, so that the connection between the user and the ES used in compared method becomes easy to break.

**Service initial placement costs with different latency constraints.** The cost of each algorithm under four constraints of latency is shown in Figure 6.

We compared the effectiveness of the initialization placement algorithm under different delay constraints. The full placement strategy refers to placing the service on the server closest to the user for each service request of the user. The random placement strategy refers to randomly selecting some servers to serve placement. GA algorithm uses genetic algorithm to select the optimal server to initialize the placement service. When the delay threshold is 200 ms, the costs of the DSM strategy is similar to those of the GA algorithm, slightly larger than the GA algorithm and much smaller than the full placement strategy; when the delay threshold is 175 ms, the costs of the DSM algorithm is 1% less than the GA algorithm. As the delay constraint shrinks, the advantages of the DSM algorithm are more obvious.

**Package loss rate with different package sizes.** In the process of calculating the packet loss rate, we conducted a total of 250 simulation experiments, each lasting 100 time slots. Then, we integrated all the simulation data to calculate the number of packets lost, and calculated the packet loss rate when the packet size is 1M, 2M, 8M, 16M, 32M, and 64M.

In general, we can find that with the increase in data packets, the packet loss rate gradually increases. This is because the increase in data packets will inevitably increase network traffic, and network transmission will be affected, causing packet loss in severe cases.

It can be seen from Figure 7 that the packet loss rate of SO is the highest. When the data packet is large, the packet loss rate of the proposed algorithm is 1% lower than that of the DQN algorithm. The SO migration strategy takes the user leaving the service coverage area as the triggering condition for the migration. When the user leaves the current server coverage area, the services that the user needs will be migrated to the new ES. This frequent migration method will transfer a large amount of data, resulting in the massive increase in network traffic may even cause network congestion, cause packet loss, and increase the probability of data transmission failure. Both DQN and DSM take server load, transmission delay, and other factors into account. However, the algorithm proposed in this article predicts and corrects the user’s request location, which can better cope with the dynamic changes of the network.

**Delay and migration costs with different Vs.** It is shown in Figure 8(a) that a large value of V makes system care more about latency. User latency decreases with V increasing and gradually approaches minimum latency in DSM, while SO remains stable. When the value of V is small, the proportion of delay in decision-making is small and the delay is high. As the value of V increases, the delay is continuously reduced and approaches the minimum value, and as V increases, so does the migration costs.

It is shown in Figure 8(b) that user latency decreases with long-term costs increasing and gradually approaches minimum latency in DSM. When the value of long-term costs is small, the delay is high.

**Conclusion**

In this article, we focus on service migration strategy and propose a service migration algorithm based on the Lyapunov optimization under the constraints of migration costs. The DSM algorithm achieves fast response of the edge of the service request, reduces the
request delay of the user node, and improves the overall service efficiency of the network. At the same time, the mobility-aware service migration process is effective. The probability of service interruptions caused by node motion is reduced, and the continuity of mobile services is guaranteed to better extent. In future research, we will continue to study balancing user-perceived latency and migration costs under multi-user conditions, while satisfying load balancing as much as possible.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The work is supported by National Key R&D Program of China (2020YFB1807802).

ORCID iD
Huiyong Liu https://orcid.org/0000-0002-1061-4300

References
1. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update. 2016–2021 White Paper[R/OL] [2017–1213], https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visualnetworking-index-vni/mobile-white-paper-c11-520862.html (accessed 28 February 2017).
2. Luan TH, Gao L, Li Z, et al. Fog computing: focusing on mobile users at the edge. arXiv:1502.01815; 2015.
3. Wang S, Tu G-H, Ganti R, et al. Mobile micro-cloud: application classification, mapping, and deployment. In: Proceedings of annual fall meeting of ITA (AMITA), 2013, pp. 1–8. http://www.commsp.ee.ic.ac.uk/~wiser/publications/Shiqiang/P2.1_Ganti_Mobile_Micro-cloud_Long%20Paper.pdf
4. Bahl V. Cloud 2020: the emergence of micro datacenter for mobile computing, Redmond, WA: Microsoft, 2015.
5. Ha K, Abe Y, Chen Z, et al. Adaptive VM handoff across cloudlets. Technical report, Report. CMU-CS-15-113, June 2015. Pittsburgh, PA: School of Computer Science, Carnegie Mellon University.
6. Dastjerdi AV and Buyya R. Fog computing: helping the Internet of things realize its potential. Computer 2016; 49(8): 112–116.
7. Yang P, Zhang N, Bi Y, et al. Catalyzing cloudfog inter-operation in 5G wireless networks: an SDN approach. IEEE Netw 2017; 31(5): 14–20.
8. Taleb T, Ksentini A and Frangoudis P. Follow-me cloud: when cloud services follow mobile users. IEEE Trans Cloud Comput 2016; 7: 369–382.
9. Wang S, Xu J, Zhang N, et al. A survey on service migration in mobile edge computing. IEEE Access 2018; 6: 23511–23528.
10. Fu S, Su Z, Jia Y, et al. Interference cooperation via distributed game in 5G networks. IEEE Internet Things J 2017; 6: 311–320.
11. Taleb T and Ksentini A. An analytical model for follow me cloud. In: Proceedings of the 2013 IEEE global communications conference (GLOBECOM), Atlanta, GA, 9–13 December 2013, pp. 1291–1296. New York: IEEE.
12. Sharma R, Kumar S and Trivedi MC. Mobile cloud computing: a needed shift from cloud to mobile cloud. In: Proceedings of the 2013 5th international conference and computational intelligence and communication networks, Mathura, India, 27–29 September 2013, pp. 536–539. New York: IEEE.
13. Wang S, Uraonkar M, Zafer M, et al. Dynamic service migration in mobile edge-clouds. In: Proceedings of the 2015 IFIP networking conference (IFIP Networking), Toulouse, 20–22 May 2015, pp. 1–9. New York: IEEE.
14. Zhang W, Hu Y, Zhang Y, et al. SEGUE: quality of service aware edge cloud service migration. In: Proceedings of the 2016 IEEE international conference on cloud computing technology and science (CloudCom), Luxembourg City, 12–15 December 2016, pp. 344–35. New York: IEEE.
15. Rahul U, Shiqiang W, Ting H, et al. Dynamic service migration and workload scheduling in edge-clouds. In: Proceedings of the 33rd international symposium on computer performance, modeling, measurements, and evaluation, Volume 91. Sydney, NSW, Australia, September 2015, pp. 205–228. http://dx.doi.org/10.1016/j.peva.2015.06.013
16. Feng J, Yu FR, Pei Q, et al. Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: a deep reinforcement learning approach. IEEE Internet Things J 2020; 7(7): 6214–6228.
17. Zhang C and Zheng Z. Task migration for mobile edge computing using deep reinforcement learning. Fut Gen Comput Syst 2019; 96: 111–118.
18. Brandherm F, Wang L and Mühluhauer M. A learning based framework for optimizing service migration in mobile edge clouds. In: Proceedings of the 2nd international workshop on edge systems, Dresden, 25 March 2019.
19. Gao Z, Jiao Q, Xiao K, et al. Deep reinforcement learning based service migration strategy for edge computing. In: Proceedings of the 2019 IEEE international conference on service oriented system engineering (SOSE), San Francisco East Bay, CA, 4–9 April 2019, pp. 116–1165. New York: IEEE.
20. Chen M, Wei L, Giancarlo F, et al. A dynamic service migration mechanism in edge computing. ACM T Internet Techn 2019; 19(2): 1–15.
21. Huang H, Zhang H, Guo T, et al. Reliable redundant services placement in federated micro-clouds. In: Proceedings of the 2019 IEEE 25th international conference on parallel and distributed systems (ICPADS), Tianjin, China, 4–6 December 2020. New York: IEEE.
22. Varasteh A, De Andrade M, Machuca CM, et al. Power-aware virtual network function placement and routing using an abstraction technique. In: Proceedings of the 2018 IEEE global communications conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018, pp. 1–7. New York: IEEE.
23. Wang K, Shen M, Cho J, et al. MobiScud: a fast moving personal cloud in the mobile network. In: Proceedings of the ACM workshop on all things cellular: operations, applications and challenges, London, 17–21 August 2015, pp. 19–24. New York: ACM.
24. Khalajzadeh H, Yuan D, Grundy J, et al. Improving cloud-based online social network data placement and Replication. In: Proceedings of the IEEE international conference on cloud computing, San Francisco, CA, 27 June–2 July 2016, pp. 678–685. New York: IEEE.
25. Ouyang T, Zhou TZ and Chen X. Follow me at the edge: mobility-aware dynamic service placement for mobile edge computing. IEEE J Select Areas Commun 2018; 36(10): 2333–2345.
26. Aljeri N and Boukerche A. Performance evaluation of movement prediction techniques for vehicular networks. In: Proceedings of the IEEE international conference on communications IEEE, Paris, 21–25 May 2017. New York: IEEE.
27. Ojima T and Fuji T. Resource management for mobile edge computing using user mobility prediction. In: Proceedings of the 2018 international conference on information networking IEEE computer society, Chiang Mai, Thailand, 10–12 January 2018. New York: IEEE.
28. Michal P, Natasa S-D and Matthias G. CRAWDAD dataset epfl/mobility (v. 2009-02-24), https://crawdad.org/epfl/mobility/20090224, https://doi.org/10.15783/C7J010 (accessed 24 February 2009)