Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 6441791, 13 pages
https://doi.org/10.1155/2023/6441791

Research Article

Virtual Machine Migration Strategy Based on Markov Decision and Greedy Algorithm in Edge Computing Environment

Xiaoxue Ma,[1] Wangkai He,[2] and Yan Gao[3]

[1]*Department of Computer Teaching, Hebei University, Baoding, China*
[2]*Hebei Key Laboratory of Highly Trusted Information System, Hebei University, Baoding, China*
[3]*School of Cyber Security and Computer, Hebei University, Baoding, China*

Correspondence should be addressed to Yan Gao; gymorsiback@gmail.com

Received 26 March 2022; Revised 30 August 2022; Accepted 6 October 2022; Published 29 April 2023

Academic Editor: Ghufran Ahmed

In view of the problem of long migration time and low availability of VM (virtual machine) migration in MD (mobile devices) in the MEC (mobile edge computing) environment, this paper designs a MD2N (Mobile Device to Network) model. This model no longer uploads requests to the cloud center, but centers on the edge server, scheduling and selecting migration services through the communication between the servers. And, a virtual machine migration strategy based on Markov decision and greedy algorithm is proposed. Model the edge server based on its geographic location, imitating the traditional cellular network method, and then the improved greedy algorithm is used to dynamically select the best edge device to migrate VM. Simulation and comparison experiments show that the proposed scheme has small migration fluctuations, stable migration status, short migration time, low average time delay, and high virtual machine migration availability after migration.

1. Introduction

With the explosive growth of the number of mobile devices (MD) and the resulting large amount of data, cloud computing has not been able to meet user requirements in terms of time delay and energy efficiency. Therefore, edge computing [1–4] has achieved tremendous development as a supplementary computing paradigm to overcome the above challenges. Among them, mobile edge computing (MEC), as a product of the integration of the Internet and Communication Technology (ICT), has become a key technology for the next generation of mobile networks [5, 6] enabling mobile devices to distribute their computing tasks to servers located on the "edge" of the radio access network [7], and it creates more practical value while reducing core network congestion, such as lower latency, location awareness, and network context information. [8, 9].

Due to the limited coverage of edge servers and the random mobility of users, sinking the computing power of cloud servers to the edge will bring new challenges. For example, when users are receiving continuous services, they may roam in wireless areas served by different edge servers. In order to ensure user satisfaction with service performance, dynamic service migration needs to be considered.

If the MEC allows the MD to migrate its computing tasks to the MEC server through the wireless access network, the computing overhead and the time delay caused by mutual communication must be considered at the same time. In order to solve the above problems, this paper proposes a dynamic programming algorithm based on Markov decision and greedy algorithm. It selects computing nodes (local equipment, edge server, and remote cloud) to run its services by predicting the behavior trajectory and migration needs of mobile users. Use Markov model to deduce the location of MD, prepare for its migration in advance, and then use greedy algorithm to select the optimal migration strategy.

2. Related Work

Mobile edge computing has received more and more attention in recent years [10], especially some previous studies that focused on service placement issues [11–13]. The issue of user mobility management is a major challenge for service placement in MEC. Existing mobility management work can

be roughly divided into three categories. (1) Always migrate, as long as the mobile device enters the new MEC server service range, immediately migrate the computing task from the source MEC server to the current MEC server. (2) Never migrate, no matter how the mobile device moves, the calculation task is always executed on the original MEC server, and communicates with the source server through the transmission of information. (3) Partial migration, by weighing the communication delay and migration time delay, and making the optimal migration decision according to the actual application situation. Obviously, the first method reduces communication delays, but it will bring additional migration costs, such as insufficient bandwidth usage, potential service interruption, or even handover failure, and the second method will cause longer communication delays as the network distance increases. Although the third method has the highest benefit, it often consumes a lot of computing resources when making a decision. Therefore, how to pay the minimum cost when making a decision becomes the main research goal.

The challenge mentioned above can be described as dynamically keeping all services following the user when the user moves, which will lead to the migration of VM. Raad et al. [14] considered large-scale real-time VM migration, and evaluated the locator/identifier separation protocol (LISP) based on flat message definition to improve migration efficiency. Machen et al. [15] proposed a three-layer service migration protocol using Linux container (LXC) and kernel-based virtual machine (KVM) technologies. Cerroni and Callegati [16] designed sequential and parallel multiple VMs and compared the performance of the two methods to achieve parallel migration with lower latency but higher resource consumption. Although the above methods can solve the problems of time delay and dynamic migration, the designed solution occupies more resources and cannot be realized in the edge environment with limited resources. In this paper, to adapt to the shortage of limited resources in the edge environment, a sequential migration method is used to migrate services.

The service migration method is a key component of the right management in MEC [17–19]. When there is user migration, it is difficult to decide what is the best choice to migrate the running service. Although VM migration is closer to the user's location, it will cause an increase in cost. Nadembega et al. [20] use a mobile-based prediction scheme to balance execution overhead and transmission delay. Wang et al. [21] place services by predicting the cost of future data transmission, processing and service migration. W. Zhang et al. [22] observe the correlation between bandwidth and geographic location based on historical data, and use this to predict user location to optimize wireless transmission time.

But in actual operation, it is more difficult to accurately predict the future information of MD based on historical data, and it will cause a large computational overhead, and may also cause a larger time delay in an edge-computing environment. Taleb and Ksentini [23] used Markov chains to analyze the possibility of user mobility. Ksentini et al. [24] and Wang et al. [25] both tried to model the entire service migration process and design the optimal migration decision through the Markov decision process. The research of T. Ouyang et al. [26] does not require any prior knowledge about the future mobility of users, and uses Lyapunov optimization to solve the service placement problem.

But, these works are mainly concentrated on the management and optimization of service placement within the system, and the scheduler understands complete system information. In actual use, the scheduler may not necessarily have the future location information of the mobile device, and it also faces the uncertainty of edge server selection. In order to meet these challenges, Sun et al. [27] proposed a learning-based service placement framework, using the theory of MAB (multiarmed bandit) to enable vehicles to provide their services by learning the best neighboring vehicle selection.

Although the above method solves the problem of service placement, it does not consider the time dependence of sequential decisions when tasks are migrated from one server to another. Sun et al. [28], user-centric EMM (energy-aware mobility management) solution, continuously optimize the time delay caused by wireless access and computing, and select BS and MEC servers for users and when to handover. Alfakih et al. [29] proposed a RL-SARSA (state-action-reward-state-action algorithm based on reinforcement learning) to solve the problem of resource management in edge servers, and through the best foaming decision to minimize the system cost, including energy consumption and calculation time delay. This method is called SARSA based on offloading decision (OD-SARSA).

Pavlos et al. [30] regarded the MEC server option as a public resource pool with uncertain user returns, and local computing as a security option for each user, following the attributes of prospect theory, and using local computing in the case of probabilistic uncertainty and uninstall overhead options to develop the user's prospect theory utility. The solution MIGRATE of Santa et al. [31] proposed the concept of virtual mobile devices (vMDs). When the devices are moved to different management domains to access the network, virtual mobile nodes in the form of digital twins must be able to "migrate" to the new edge virtualization field. Liang et al. [32] proposed a multiuser task offloading scheme based on edge-cloud joint computing, designed a greedy algorithm based on submodule theory and fully utilized the computing and communication resources of the cloud and edge. This solution still has the participation of the cloud. Although the transmission speed is slightly better than the traditional migration method, the communication delay from the edge to the cloud still exists. The solution proposed in this article does not depend on the cloud center, and the main work is carried out in the edge network. There is no remote communication delay, which can change the traditional migration method and improve the transmission efficiency.

3. Scheme Design

As shown in Figure 1, when a user is roaming or has a migration request, in order to obtain a lower perceived delay, users tend to place their services on the most appropriate MEC server through 4G/5G/WiFi access networks.

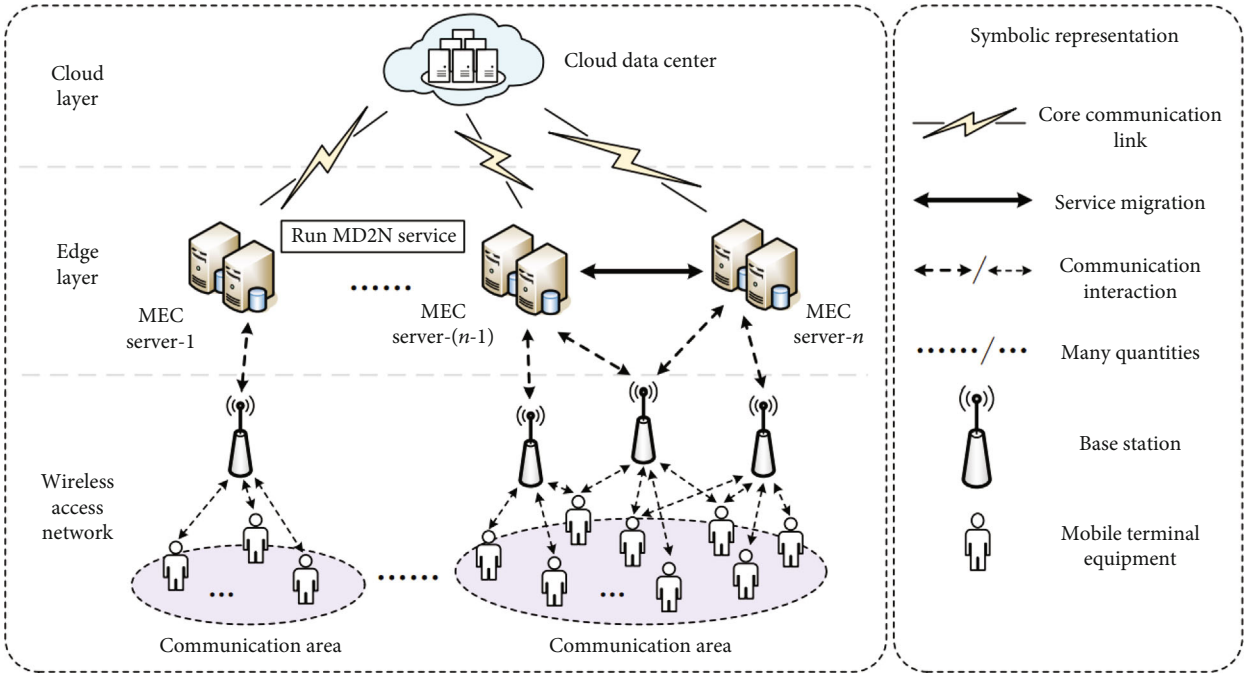

Figure 1: System model of MD2N.

In order to ensure the continuity of services, it is necessary to take the migration time into consideration and use the predictive migration method [33, 34] to migrate virtual machines. In this paper, we design the Mobile Device to Network (MD2N) model based on the mobile device's movement and service conditions in real life. The specific model is shown in Figure 1. Suppose there are N servers arranged by index $\mathbb{N} = \{1, 2, \cdots, N\}$, each of which is directly associated with one or more base stations (BS), and all MEC servers can communicate with each other through wired channels. Any MEC server in the edge network environment can provide multiple MD2N services for mobile devices. The mobile device selects the best base station according to the signal strength and communicates with the MEC server associated with the base station. In addition, the MEC server can be used as a router, so that mobile devices can communicate with the MEC server remotely. Under this premise, multiple MEC servers can provide services for the same mobile device.

Divide the actual physical time into continuous time gaps, represented by t, $t = 1, 2, 3, \cdots$, a time gap is assumed to be τ seconds; the state of the mobile edge network is represented by an $N \times N$ matrix, the (i-, j-) th element $(G)_{i,j} \in [0, \eta_{\max})$ ($\eta_{\max}$ is the maximum channel capacity) in the matrix indicates whether there is a direct connection between the i -th and j-th MEC server, if $(G)i, j = 0$, means no connection. Assuming that G is a constant matrix that does not change with time, that is, within a specified time, the state of the mobile edge network will not change, so a time parameter $\alpha(t) \in [\alpha_{\min}, \alpha_{\max}]$ is defined to represent the channel resources available in the network. $\alpha_{\min}$ represents the minimum channel resources required to ensure communication. In fact, for the service in the network, in order to ensure its stable transmission in the case of channel congestion and flow fluctuations, excessive channel resources should be provided. In addition, the more congested the channel, the fewer available resources, so α can also indicate the number of mobile devices in the channel. $\eta_{i,j}$ (bits/s) represents the available channel capacity between MEC server i and j, which can be calculated using G according to the RIP (Routing Information Protocol).

We assume that according to the MEC server indexed by $\mathcal{M} = \{1, \cdots \cdots, M\}$, connected mobile devices can use up to M MD2N services; we can define a $Q \times M$ matrix represented by π to record the mobile device's future. The migration strategy in the next Q time slot, where Q is an integer use $(\pi)_{q,m} \in \{0,1,2, \cdots \cdots, N\} (q \in \{1, \cdots \cdots, Q\})$ to represent the target MEC server hosting service m in the time slot $t_q = t_0 + q - 1$ (t_0 represents the start time), correspondingly in the matrix, if the MEC server will not host the service in the future, it will be represented by $(\pi)_{q,m} = 0$. Based on the definition of the above matrix, we can use the vector $\pi(t)$ to describe the service position in the time slot $t \in \{t_0, \cdots \cdots, t_0 + Q - 1\}$, therefore, we can use the time slot $\{t_1, t_2, \cdots \cdots, t_q\}$ in the vector $\pi(t_1, t_2, \cdots \cdots, t_q)$ to rewrite the migration strategy. Similarly, for any time slot $t \in \{t_0 + 1, \cdots, t_0 + Q - 1\}$, you can use $\pi(t-1, t)$ to rewrite the migration strategy.

Using a model similar to Markovian to derive the location of MD, divide the network into regular hexagonal units (as shown in Figure 2), where one hexagonal unit represents the service range of an MEC server. Consider a random walking movement model, in which MD visits any of the six adjacent cells with probability $p = 1/6$. Figure 2 shows a ring with a service area of $k = 5$. When the MD's position

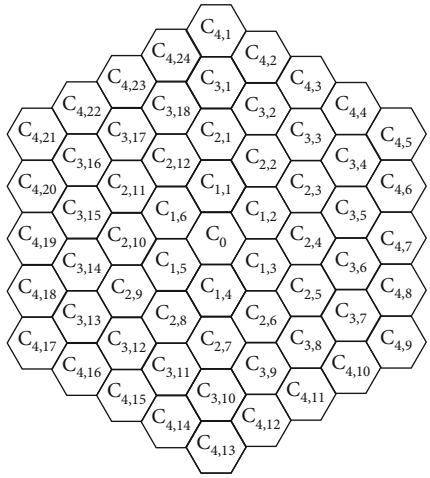

Figure 2: Cellular network diagram.

is k hops away from the initial position, the service migration and relocation for the MD starts. Let $X(t)$ denote the distance from the MD to the initial server at time t (calculated in hop count). The system $\{X(t), t \geq 0\}$ forms a CTMC (Continuous-Time Markov Chain), the state space is $\{C(m, n) \mid 0 \leq m \leq (k-1), 1 \leq n \leq 6\, m\}$.

In Figure 2, it can be seen that the MD in the 1st ring moves to any adjacent unit with probability p. The MD in the second ring returns to the 1st ring with probability p, remains in the second ring with probability $2p$, and moves to the third ring with probability $3p$, increasing the distance, so all the rings in the first ring can be aggregated into one state. For the third ring, there are two types of rings: (1) similar to ring 1, there are three 3-rings, two 2-rings, and one 1-ring adjacent to it; (2) on the three rings, each has two rings adjacent to it. Depending on where the MD is in the second ring, there may be a $2p$ or $3p$ probability to move to the third ring. Therefore, two aggregation states are obtained: $C_2\{C_{2,1}, C_{2,3}, C_{2,5}, C_{2,7}, C_{2,9}, C_{2,11}\}$ and $C(1)\,2\{C_{2,2}, C_{2,4}, C_{2,6}, C_{2,8}, C_{2,10}, C_{2,12}\}$. In the same way, any ring h has aggregation states C_h and $C(z)\,h$, where $1 \leq z \leq \lceil (h-1)/2 \rceil$.

Figure 3 shows a Markov chain in each aggregation state with k maximum of 5. According to the research of Langar et al. [35], the ring number and superscript are used to indicate the aggregation state of each ring h, and the equilibrium state of its migration strategy is shown in formulas (1)–(9):

$$\begin{cases} \pi_0 = \dfrac{1}{6}\pi_1 + \dfrac{1}{2}\pi_{\text{k-1}} + \dfrac{1}{3}\displaystyle\sum_{y=1}^{\lceil \frac{k-2}{2} \rceil} \pi_{k-1}^{y}, \\ \pi_1 = \pi_0 + \dfrac{1}{3}\pi_1 + \dfrac{1}{6}\pi_2 + \dfrac{1}{3}\pi_2^{(1)}, \\ \pi_2 = \dfrac{1}{6}\pi_1 + \dfrac{1}{6}\pi_3 + \dfrac{1}{3}\pi_2^{(1)} + \dfrac{1}{6}\pi_3^{(1)}, \\ \pi_{k-1} = \dfrac{1}{6}\pi_{k-2} + \dfrac{1}{6}\pi_{k-1}^{(1)}, \\ \pi_h = \dfrac{1}{6}\pi_{h-1} + \dfrac{1}{6}\pi_{h+1} + \dfrac{1}{6}\pi_{h-1}^{(1)} + \dfrac{1}{6}\pi_{h+1}^{(1)}, \forall 3 \leq h \leq k-2, \end{cases} \tag{1}$$

$$\begin{cases} \pi_2^{(1)} = \dfrac{1}{3}\pi_1 + \dfrac{1}{3}\pi_2 + \dfrac{1}{6}\pi_3^{(1)}, \\ \pi_3^{(1)} = \dfrac{1}{3}\pi_2 + \dfrac{1}{3}\pi_3 + \dfrac{1}{3}\pi_2^{(1)} + \dfrac{1}{6}\pi_3^{(1)} + \dfrac{1}{6}\pi_4^{(1)} + \dfrac{1}{3}\pi_4^{(2)}, \\ \pi_4^{(1)} = \dfrac{1}{3}\pi_2 + \dfrac{1}{3}\pi_4 + \dfrac{1}{6}\pi_3^{(1)} + \dfrac{1}{6}\pi_5^{(1)} + \dfrac{1}{3}\pi_4^{(2)} + \dfrac{1}{6}\pi_5^{(2)}, \\ \pi_{k-1}^{(1)} = \dfrac{1}{3}\pi_{k-2} + \dfrac{1}{3}\pi_{k-1} + \dfrac{1}{6}\pi_{k-2}^{(1)} + \dfrac{1}{6}\pi_{k-1}^{(2)}, \\ \pi_h^{(1)} = \dfrac{1}{3}\pi_{h-1} + \dfrac{1}{3}\pi_h + \dfrac{1}{6}\pi_{h-1}^{(1)} + \dfrac{1}{6}\pi_{h+1}^{(1)} + \dfrac{1}{6}\pi_h^{(2)} + \dfrac{1}{6}\pi_{h+1}^{(2)}, \forall 5 \leq h \leq k-2, \end{cases} \tag{2}$$

$$\pi_h^{(y)} = \frac{1}{6}\pi_h^{(y-1)} + \frac{b_1}{6}\pi_h^{(y+1)} + \frac{1}{6}\pi_{h-1}^{(y-1)} + \frac{1}{6}\pi_{h-1}^{(y)} + \frac{b_2}{6}\pi_{h+1}^{(y)} + \frac{b_2}{6}\pi_{h+1}^{(y+1)}, \forall 6 \leq h \leq k-2 \text{且} 2 \leq \text{y} \leq \left\lceil \frac{h-1}{2} \right\rceil - 1. \tag{3}$$

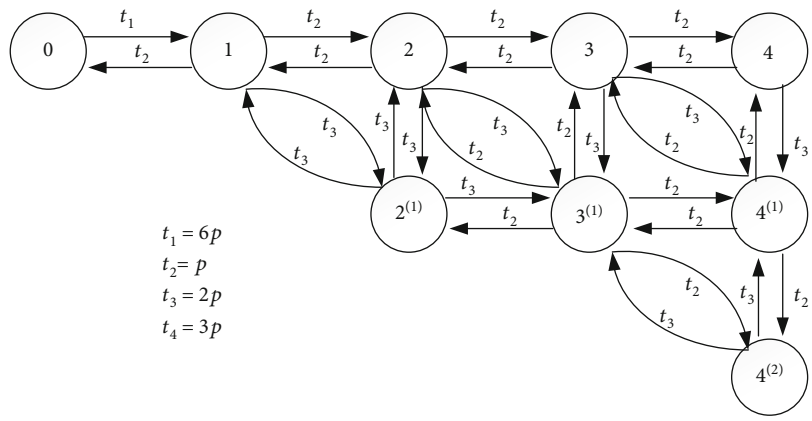

Figure 3: Markov chain with k max 5.

Among them, the values of b_1 and b_2 are

$$b_1 = \begin{cases} 1, & \text{if } h \text{ is odd,} \\ 1, & \text{if } h \text{ is even, and } 2 \le y \le \left\lceil \frac{h-1}{2} \right\rceil - 2, \\ 2, & \text{if } h \text{ is even, and } y = \left\lceil \frac{h-1}{2} \right\rceil - 1, \end{cases} \tag{4}$$

$$b_2 = \begin{cases} 0, & \text{if } 6 \le h \le k-2, \\ 1, & \text{if } h = k-1, \end{cases} \tag{5}$$

$$\pi_{2l}^{(l)} = \frac{1}{6}\pi_{2l}^{(l-1)} + \frac{1}{6}\pi_{2l-1}^{(l-1)} + \frac{c_1}{6}\pi_{2l+1}^{(2l)}, \forall 2 \le l \le \left\lceil \frac{k-1}{2} \right\rceil. \tag{6}$$

Among them, the value of c_1

$$c_1 = \begin{cases} 0, & \text{if } l = \frac{k-1}{2}, \\ 1, & \text{other situations,} \end{cases}$$

$$\pi_{2l+1}^{(l)} = \frac{1}{6}\pi_{2l+1}^{(l-1)} + \frac{1}{6}\pi_{2l+1}^{(l)} + \frac{1}{6}\pi_{2l}^{(l-1)} + \frac{1}{6}\pi_{2l}^{(l)} + \frac{c_2}{6}\pi_{2l+2}^{(l)} + \frac{c_2}{6}\pi_{2l+2}^{(l+1)}, \forall 2 \le l \le \frac{k-2}{2}. \tag{7}$$

Among them, the value of c_2

$$c_2 = \begin{cases} 0, & \text{if } l = \frac{k-2}{2}, \\ 1, & \text{other situations,} \end{cases} \tag{8}$$

$$\sum_{h=0}^{k-1} \pi_h = \sum_{h=2}^{k-1} \sum_{x=1}^{\left\lceil \frac{h-1}{2} \right\rceil} \pi_h^{(x)} = 1. \tag{9}$$

Among them, π_x represents the current position, formula (1) represents the possible position and probability of the next step when it is located in the corresponding position in Figure 3, and formula (2) represents the corresponding position in Figure 3. The possible position and probability of the second step, (3) is the prediction of the possible position and probability of the next h step according to its law. equations (4)–(9) are the reasoning and simplification of (3).

Let $E[\text{Dist}]$ denote the average distance from MD to the MEC server. From (1)–(9), we can see that the size of $E[\text{Dist}]$ depends on the value of k, so the average distance from MD to the initial position is

$$E[\text{Dist}] = \sum_{h=1}^{k-1} h\pi_h + \sum_{h=2}^{k-1} \sum_{z=1}^{\left\lceil \frac{h-1}{2} \right\rceil} h\pi_h^{(z)}. \tag{10}$$

4. Migration Decision

4.1. Transmission and Calculation Consumption. The widely used three-parameter model is used to describe each MD2N service m; input the data that needs to be processed with the size of λ_m (bits), the calculation intensity γ_m (CPU cycles/bit) represents the requirement of each service requiring CPU resources to calculate one bit of input data, and the maximum allowable delay D_m (seconds); in addition, define $f_m, m \in \mathcal{M}$ to represent the frequency of message sending.

Data is transmitted from one MEC server to another MEC server through a wired channel. Assuming that the wireless transmission consumption between the mobile device and the serving base station BS is a constant, the transmission consumption of MD2N service m is

$$\mathrm{d}_{\text{trans}}(t) = \frac{\lambda_m}{\alpha(t)\eta_{L(t),j}} + D_{\text{add}}. \tag{11}$$

Among them, D_{add} is a constant, representing the consumption of wireless transmission, $\eta_{L(t),j}$ represents the channel capacity from $L(t)$ to $j = (\pi)_{t,m}$.

Each MEC server can provide computing services for multiple tasks of connected mobile devices at the same time. Use $\sigma_{m,n}$ to describe the CPU frequency that MEC server n can allocate to MD2N service m. Because MEC server has powerful computing capabilities, $\sigma_{m,n}$ will not change. If the MEC server n is selected to provide service m, the

calculated delay is

$$d_{\text{com}}(t) = \frac{\gamma_m \lambda_m}{\sigma_{m,n}}. \tag{12}$$

When the migration strategy vector is $\pi(t)$, we consider the communication delay $U_m(t)$ of the sum of data transmission and processing in the specified time slot t

$$U_m(t) = d_{\text{trans}}(t) + d_{\text{com}}(t). \tag{13}$$

4.2. Migration Cost. Add a new parameter $\theta_m \in [0, \theta_{\max}]$ (in bits) to indicate the size of the service sample data, where $\theta_{\max}$ is the maximum possible size of the sample data, for each MD2N service $m \in \mathscr{M}$, execute $\pi(t-1, t)$ in the time slot t. The time delay of the migration strategy is

$$W_m(t, \pi(t-1, t)) = \frac{\theta_m}{\alpha(t)\eta_{s,e}}. \tag{14}$$

Among them, $\eta_{s,e}$ is the channel capacity from $s = (\pi)_{t-1,m}$ to $e = (\pi)_{t,m}$.

The goal of this paper is to minimize the average time delay of service migration. Therefore, once the MD's future movement trajectory is obtained through probability calculation, a migration strategy is proposed to reduce the time delay as much as possible based on the consideration of communication and migration costs. For service m, the total service delay in time slot t is

$$D_m(t) = U_m(t) + \frac{W_m(t, \pi(t-1, t))}{\tau f_m}. \tag{15}$$

The delay defined in equation (15) represents the response time delay of a single message between the MEC server and the MD. When the migration of the VM also occurs in the time slot t, it is obvious that this will lead to an increase in the response time. The response time is the sum of communication costs and migration costs. If in the time slot t, service m chooses not to migrate, the migration cost at this time is zero, the response time represents the communication time delay, and the formula (15) is expressed as

$$D_m(t) = U_m(t). \tag{16}$$

Generally speaking, MD2N services can always be completed within 1 s [36], which is much smaller than the time gap τ. The use of VM migration technology makes the migration time of each service not lasting too long. In view of this, the following assumptions can be made; even under the minimum available channel resource $\alpha_{\min}$, the migration of each MD2N service can be completed within a time gap, namely

$$\begin{aligned} W_m(t, \pi(t-1, t)) \le \tau, \forall t, \forall m, \\ D_m \le \tau, \forall m. \end{aligned} \tag{17}$$

For each service $m \in M$, the importance of MD equipment is different, so use $\omega_m \in [0, 1]$ to represent the weight of each service. Although different MD2N services have great differences in actual delay and end time, according to the importance of the service, the time delay of equation (15) is standardized using $\omega_m(D_m(t)/D_m)$. Therefore, the standardized average time delay of multiple MD2N services can be expressed as

$$D(t) = \frac{1}{M} \sum_{m=1}^{M} \omega_m \frac{D_m(t)}{D_m}. \tag{18}$$

MD2N associates a series of possible future state transition behaviors to A_s according to the user's current state s. For a given behavior a, when switching from one state s to another state s', an instantaneous reward $r(s, s', a)$, the reward is based on the size of migration overhead, $r(s, s', a) = \alpha(t) - D(t)$. Therefore, the Markov Decision Process (MDP) can be expressed as

$$\left(S, (A_s, s \in S), q\left(s' \middle| s, a\right), r\left(s, s', a\right)\right). \tag{19}$$

Among them, $q(s'|s, a)$ represents the conversion rate when switching from state s to another state s' under behavior a. Define an attenuation factor $\gamma \in [0, 1]$, then the migration strategy $\pi(t_1, t_2, \cdots \cdots, t_q)$ can be expressed as

$$v_\gamma^\pi = \lim_{q \longrightarrow \infty} \left\{ E_\gamma^\pi \sum_{t=1}^{q} \gamma^{t-1} r_t \right\} = E_\gamma^\pi \left\{ \sum_{t=1}^{\infty} \gamma^{t-1} r_t \right\}. \tag{20}$$

According to the research of Taleb et al. [37], the solution of the optimal equation of (20) is equivalent to the optimal decision $\pi * (t)$, which determines which MEC server MD chooses in the future and whether to perform the migration decision. The algorithm is described as follows

Algorithm 1 uses the dynamic programming algorithm to obtain the optimal migration strategy matrix of MD migration. The user performs migration according to the matrix. Use $\mathscr{M} * (t)$ to represent the MD queue that is about to face the migration decision, and define ρ to limit the migration overhead to a time slot t. The algorithm of the migration service is described as follows:

Algorithm 2 is a migration service algorithm based on Algorithm 1. Considering the communication delay and migration cost, we will serve it according to the migration queue obtained by Algorithm 1 under the premise of minimizing the delay and cost as much as possible.

5. Simulation Results and Performance Analysis

5.1. Experimental Configuration. This section evaluates the system utility of the proposed dynamic programming algorithm based on Markov decision and greedy algorithm under the edge-computing environment through simulation experiments. In order to be consistent with the environment

1: Initialize: t_0=1, π^* for random.
2: **do until**$\pi = \pi^*$
3: Input: $\pi(t_0\text{-}1)$.
4: $\pi \longleftarrow \pi^*$;
5: **for**$m \in \mathscr{M}$ **do**
6: $v_\gamma^\pi = \lim_{q \longrightarrow \infty} E_\gamma^\pi \{\sum_{t=1}^{q} \gamma^{t-1} r_t\} = E_\gamma^\pi \{\sum_{t=1}^{\infty} \gamma^{t-1} r_t\}$
7: **end for**
8: **loop**

Algorithm 1: Dynamic programming to solve the optimal migration decision.

1: Input: $\pi(t)$
2: Initialize: $\mathscr{M}^*(t) \longleftarrow \varnothing$, ρ=0.
3: **for**$m \in \mathscr{M}$ **do**
4: Update $U_m(t)$, $W_m(t)$ by (13), (14)
5: **end for**
6: **for**$m \in \mathscr{M}$ **do**
7: **if**$\rho + W_m(t) \leq \tau$**then**
8: $\rho \leftarrow \rho + W_m(t)$,
9: Put m into $\mathscr{M}^*(t)$.
10: **end if**
11: **end for**
12: **return**$\mathscr{M}^*(t)$

Algorithm 2: Best Migration Service (BMS).

of the comparison experiment, this article uses KubeEdge on the Dell PowerEdge R720 server to simulate an area with 26 edge servers and 1 cloud server. Each edge server is configured with dual-core CPUs. Since all edge servers are cloned, here, we assume that the computing power of all edge servers is the same, and consistent with the comparison experiment, it is 10 GHz, and the computing tasks of edge devices are randomly and evenly distributed between (500 and 1000) cycles/bit. The user's mobility data is generated by the currently widely used, the one simulator [38].

In the experiment, the migration time delay is counted, and the time of the migration failure case cannot be counted. Therefore, only the time consumption and time delay of the successfully migrated VM are recorded in this experiment. In terms of standardized average time delay $D(t)$ and actual average time delay, this experiment compares the PDOA [39] and MDP schemes and compares them with the optimal solution; in terms of actual running time and VM availability after migration, this experiment compares the PDOA, MDP, and AUSP [38] schemes and compares them with the completely offline scheme and the optimal solution.

5.2. Evaluation Index. This experiment mainly evaluates from the aspects of standardized time delay, channel available resources, number of servers, and the availability of virtual machines after migration.

5.2.1. Standardized Time Delay. Comparing the actual standardized average time delay and the average standardized time delay of each program in different time slots, through the fluctuation of each program, it can be judged whether the program is stable.

5.2.2. Channel Available Resources. Set different channel available resource rates, run each plan under different channel available resource rates, count the running time, and compare the applicability of the plans.

5.2.3. Number of Servers. Change the number of servers from small to large, count and compare the actual running time of each program.

5.2.4. The Availability Rate of the Virtual Machine after Migration. Count the ratio of the number of available virtual machines to the total number of virtual machines after the migration of each program is completed, so as to evaluate the pros and cons of each program in practical applications.

Finally, comprehensively compare all the indicators and draw a conclusion.

5.3. Experimental Results. Figure 4 compares the standardized average time delay of each time slot t, namely formula (18), the dotted line in the figure represents the calculated value, which does not change with time, for comparison, and the solid line represents the actual measured value. Here we assume that $m = 5$ MD2N services are used, and the available channel resource $\alpha = 0.5$. When $m = 5$, the migration time of each scheme is close (Figure 5), which is convenient for comparison. At the same time, starting from $\alpha = 0.5$, the average time extension tends to be horizontal (Figure 6), and less resources are more representative. It can be seen from the experimental results that the BMS scheme and PDOA scheme proposed in this paper are close to the optimal solution, and the mean value of the standardized delay is much smaller than the MDP scheme. Although the average value of the BMS scheme proposed in this paper is slightly higher than that of the PDOA scheme, the difference is very small and can be ignored, and the fluctuation of the BMS scheme is smaller and the most stable among the four schemes.

Since the PDOA solution is mainly for devices in the Internet of Vehicles, the application range is limited. In order to highlight the practicability of this scheme, an experiment was first compared with the PDOA scheme on a certain street, as shown in Figure 5. According to the change of the average time delay under the conditions of different channel available resource rates, it can be seen that as the channel available resource rate α increases, the time delay

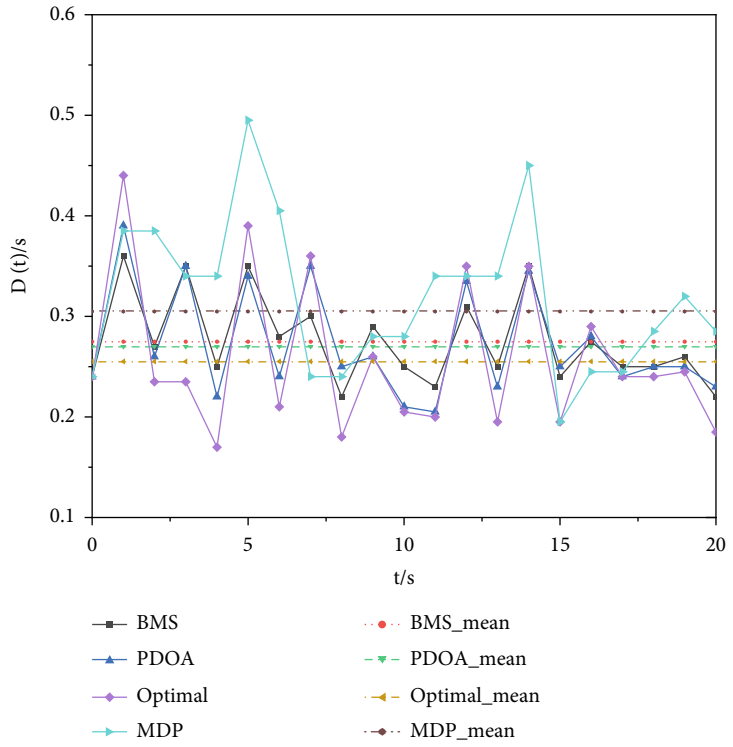

Figure 4: Normalized average time delay of different time slots t.

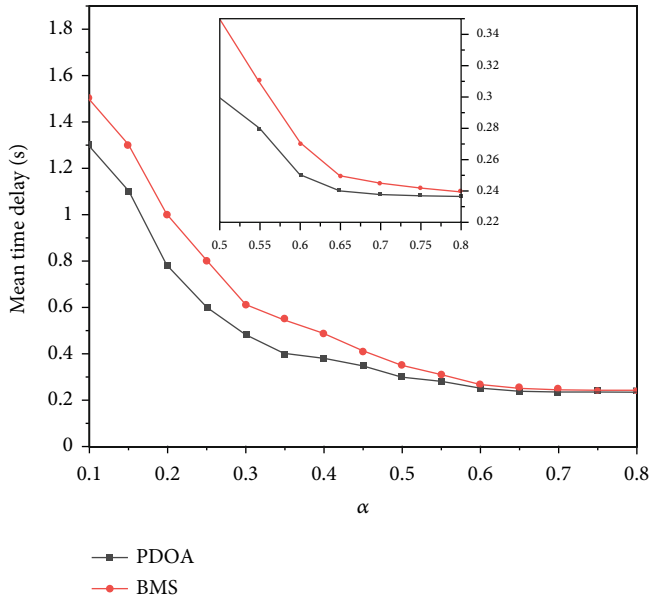

Figure 5: Average time delay under different channel available resource conditions in the Internet of Vehicles.

gradually decreases. When α increases to a certain degree (0.65), the decreasing trend of the average time delay is no longer obvious. The small graph in Figure 5 shows how the average time delay changes when α is between 0.5 and 0.8. It can be seen that the overall performance of the PDOA solution is slightly better than this solution. This is because the PDOA scheme adopts a predictive algorithm, and the behavior of cars driving on the highway is regular, so the time consumption in calculation is less than this scheme. But when the user's behavior is not regular, the prediction algorithm of the PDOA scheme no longer has the advantage. The specific results are reflected in the following experiment.

Figure 6 shows the effect of channel available resource α on the long-term normalized average time delay. In the experiment, α was increased from 0.1 to 0.8. It can be seen from the experimental result graph that as α increases, the

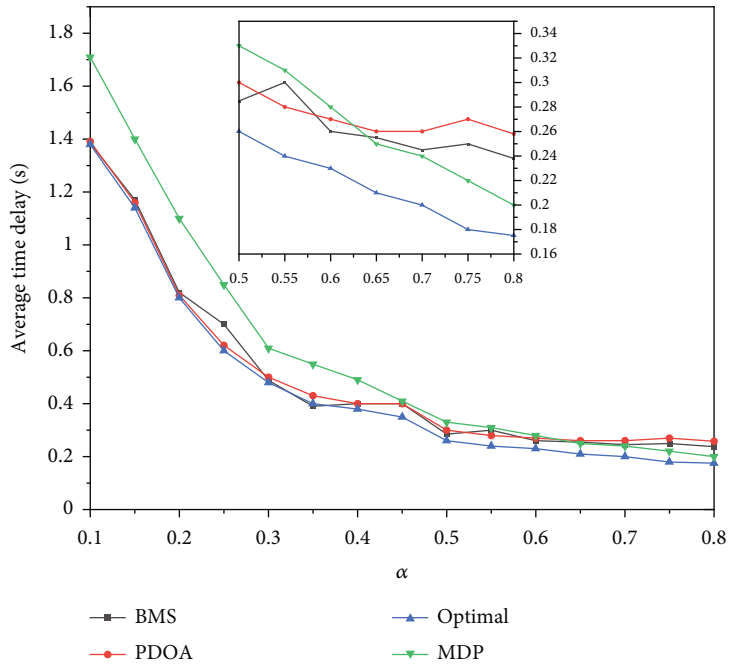

Figure 6: Average time delay under different channel available resource conditions.

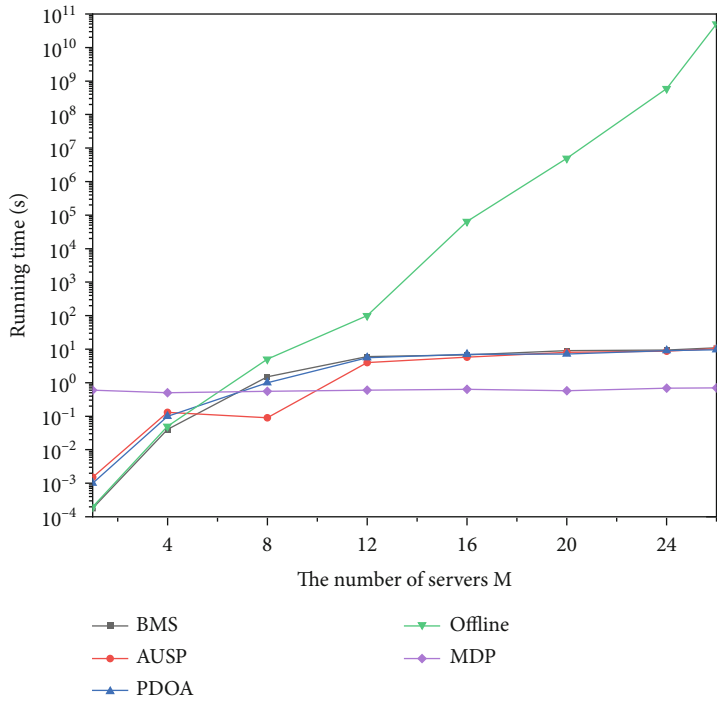

Figure 7: Running time under different number of servers.

average time delay gradually decreases. This is because as the channel's available resources increase, the available resources increase, the queues waiting for migration will decrease, and the time required will decrease accordingly. It can be seen that when the channel available resources are less (from 0.1 to 0.4), the time delay of MDP is the longest, and the BMS and PDOA schemes are very close to the optimal solution. This is because the MDP scheme treats all MD2N services equally, while the BMS, PDOA, and optimal solutions are dynamically migrating MD2N services. When channel resources are sufficient (after 0.65), it can be seen from the small graph in Figure 6 that the average time delay of the BMS and PDOA solutions is slightly higher than that of the MDP solution. This is because with the increase of

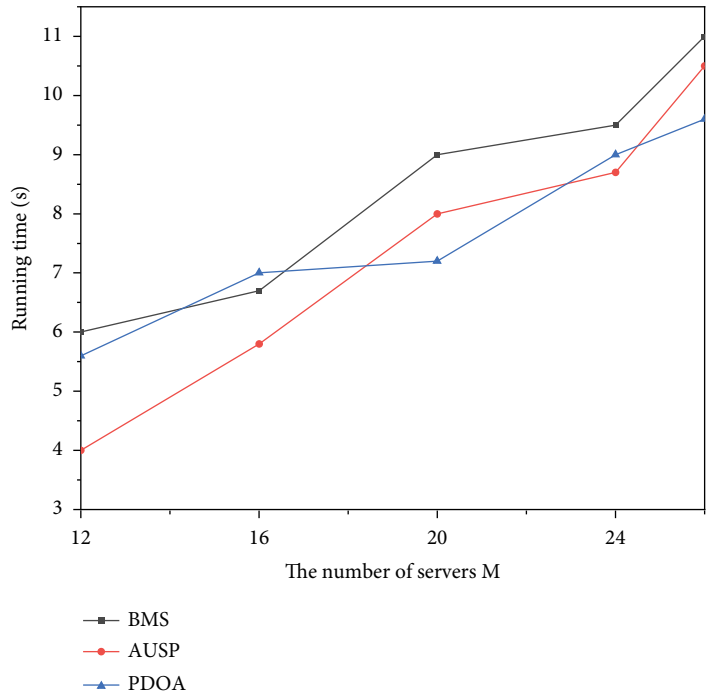

Figure 8: Running time under different number of servers (12~ 26).

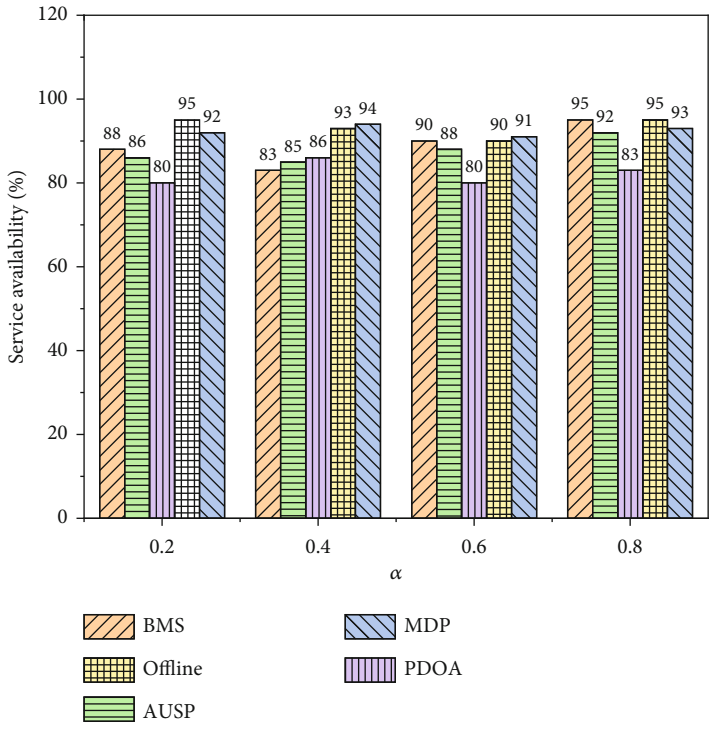

Figure 9: Service availability under different channel available resource conditions.

channel resources, the BMS solution proposed in this paper still maintains the previous migration strategy. Since the calculation scale has not changed, it is difficult to improve the time. Therefore, the average delay is slightly higher than the MDP solution, but on the whole, this time difference is very small, which can be tolerated in practical applications. In addition, although the average time delay of the BMS scheme is slightly higher than that of the PDOA scheme, it

is on the same level as a whole. Moreover, as can be seen from Figure 4, the stability of the BMS scheme is better than that of the PDOA scheme.

Figure 7 records the running time of each scheme under different numbers of servers in a certain time slot t. Figure 8 shows the running time chart of the number of servers in Figure 5 between 12 and 26. As can be seen from the figure, the time used by the BMS, AUSP, and PDOA solutions is significantly shorter than the offline algorithm time, and the three solutions consume basically the same time, especially with the increase in the number of servers, the time consumed by offline algorithms has increased dramatically, while the time consumed by other algorithms tends to stabilize. Although the time-consuming of the MDP scheme has been stabilizing, its average delay is longer and its stability is poor. Although the running time of the BMS solution proposed in this article is slightly higher than the other two, the difference is not large and can be ignored. It can be seen from Figure 4 that the BMS solution has the best stability.

Figure 9, respectively, counts the availability of VMs on the target device under the conditions of channel available resources of 0.2, 0.4, 0.6, and 0.8 after different migration schemes are executed. The specific situation is shown in Figure 9. It can be seen that the offline and MDP solutions always maintain a high service availability rate, but the running time of the offline solution varies greatly. When the number of servers is large, it takes a long time, which is not conducive to practical applications. The average time delay of the MDP scheme has large fluctuations and poor stability. With the increase of channel available resources, the service availability rates of BMS, AUSP, and PDOA tend to increase, but from an overall point of view, the service availability rate of the BMS solution proposed in this paper is higher than the other two solutions.

6. Conclusion and Future Work

Aiming at the problem of virtual machine migration in mobile devices in the edge environment, this paper proposes a dynamic programming scheme BMS based on Markov decision and greedy algorithm. Firstly, formulate the time delay and other costs incurred in the VM migration process, then use a method similar to cellular networks to model the geographic location, and use Markov decision and greedy algorithms to select the server to receive the VM to be migrated. Unlike traditional prediction algorithms, this is a probability-based choice, which improves accuracy to a certain extent and ensures the availability of the migrated VM on the target device. The innovation of the scheme proposed in this paper is mainly reflected in

(i) Abandon the traditional way of relying on cloud servers to migrate VMs in mobile devices. Instead, it relies on edge servers to migrate. The cloud data center only functions to manage edge servers and does not directly participate in the migration of VMs. Edge servers are closer to users. With the popularization of 5G technology, transmission speeds increase, response times are shorter, and time delays are shortened accordingly

(ii) Compared with the traditional relying only on MDP algorithm or prediction algorithm, this scheme adopts a combination of MDP algorithm and greedy algorithm. The impact of prediction accuracy on the migration result is removed, and at the same time it is ensured that the selected service is optimal or close to the optimal solution

The experimental results show that the stability of the BMS scheme is the best among the several schemes compared, and the running time and time delay are also of the best. Although using MEC servers close to users to serve users can reduce the transmission time delay, the computing power of MEC servers is not as good as that of cloud servers. Therefore, it will take some time to make the optimal migration decision, so it is slightly higher than AUSP and PDOA, but has high stability. And in terms of the availability of VMs after migration, the BMS solution is significantly better than the other two. To sum up, considering various factors, the BMS solution is the best choice.

Using the edge server closer to the user as the migrating dispatch center can reduce transmission delay, but due to the limited computing power of the edge server itself, the processing task speed is far lower than that of the cloud server, which will cause a large time overhead. Although the total execution time and time delay have been improved, they have not improved much. These problems will be solved through continuous improvement of the algorithm in subsequent research. But overall, the solution proposed in this paper has short running time, small average delay, best stability, and the highest availability on the target device, which is the best solution.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61572170) and the Natural Science Foundation of Hebei Province, China (F2019201290).

References

[1] S. Ashwini, R. I. Minu, and G. Nagarajan, "Edge computing: opportunities and challenges," in *Applied Edge AI: Concepts, Platforms, and Industry Use Cases*, pp. 1–22, Auerbach Publications, 2022.

[2] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, "A survey on mobile augmented reality with 5G mobile edge computing: architectures, applications, and

technical aspects," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1160–1192, 2021.

[3] R. Dautov, S. Distefano, D. Bruneo, F. Longo, G. Merlino, and A. Puliafito, "Data processing in cyber-physical-social systems through edge computing," *IEEE Access*, vol. 6, pp. 29822–29835, 2018.

[4] N. Hassan, S. Gillani, E. Ahmed, I. Yaqoob, and M. Imran, "The role of edge computing in Internet of Things," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 110–115, 2018.

[5] A. Singh, S. C. Satapathy, A. Roy, and A. Gutub, "AI-based mobile edge computing for IoT: applications, challenges, and future scope," *Arabian Journal for Science and Engineering*, vol. 47, pp. 9801–9831, 2022.

[6] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: the communication perspective," *IEEE Communication Surveys and Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[7] S. Zanafi, M. Giacobbe, M. Scarpa, and A. Puliafito, "Enabling sustainable smart environments using fog computing," in *The 1st International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS'18)*, Kenitra, Morocco, 2018.

[8] G. Cassales, H. Gomes, A. Bifet, B. Pfahringer, and H. Senger, "Balancing performance and energy consumption of bagging ensembles for the classification of data streams in edge computing," 2022, https://arxiv.org/abs/2201.06205.

[9] O. Kaiwartya, A. H. Abdullah, Y. Cao et al., "Internet of vehicles: motivation, layered architecture, network model, challenges, and future aspects," *Access*, vol. 4, pp. 5356–5373, 2016.

[10] X. Chen, Q. Shi, L. Yang, and J. Xu, "Thriftyedge: resource-efficient edge computing for intelligent iot applications," *IEEE Network*, vol. 32, no. 1, pp. 61–65, 2018.

[11] T. Liu, S. Ni, X. Li, Y. Zhu, L. Kong, and Y. Yang, "Deep reinforcement learning based approach for online service placement and computation resource allocation in edge computing," *IEEE Transactions on Mobile Computing*, p. 1, 2022.

[12] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[13] H.-S. Lee and J.-W. Lee, "Task offloading in heterogeneous mobile cloud computing: modeling, analysis, and cloudlet deployment," *IEEE Access*, vol. 6, pp. 14908–14925, 2018.

[14] P. Raad, S. Secci, D. C. Phung, A. Cianfrani, P. Gallard, and G. Pujolle, "Achieving sub-second downtimes in large-scale virtual machine migrations with LISP," *IEEE Transactions on Network and Service Management*, vol. 11, no. 2, pp. 133–143, 2014.

[15] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Migrating running applications across mobile edge clouds: poster," in *MobiCom '16: Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pp. 435-436, 2016.

[16] W. Cerroni and F. Callegati, "Live migration of virtual network functions in cloud-based edge networks," in *2014 IEEE International Conference on Communications (ICC)*, pp. 2963–2968, Sydney, NSW, Australia, 2014.

[17] Z. Liang, Y. Liu, T. M. Lok, and K. Huang, "Multi-cell mobile edge computing: joint service migration and resource allocation," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 5898–5912, 2021.

[18] W. Wu and Y. Jian, "Analysis and research on resource allocation and service migration in mobile edge computing," in *Innovative Computing*, J. C. Hung, J. W. Chang, Y. Pei, and W. C. Wu, Eds., vol. 791 of Lecture Notes in Electrical Engineering, pp. 1087–1094, Springer, Singapore, 2022.

[19] Y. Xu, T. Zhang, Y. Liu, D. Yang, L. Xiao, and M. Tao, "UAV-assisted MEC networks with aerial and ground cooperation," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 7712–7727, 2021.

[20] A. Nadembega, A. S. Hafid, and R. Brisebois, "Mobility prediction model-based service migration procedure for follow me cloud to support qos and qoe," in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kuala Lumpur, Malaysia, 2016.

[21] S. Wang, R. Urgaonkar, K. Chan, T. He, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," in *2015 IEEE International Conference on Communications (ICC)*, pp. 5504–5510, London, UK, 2015.

[22] W. Zhang, R. Fan, Y. Wen, and F. Liu, "Energy-efficient mobile video streaming: a location-aware approach," *ACM Transactions on Intelligent Systems and Technology*, vol. 9, no. 6, pp. 1–16, 2017.

[23] T. Taleb and A. Ksentini, "An analytical model for follow me cloud," in *2013 IEEE Global Communications Conference (GLOBECOM)*, pp. 1291–1296, Atlanta, GA, USA, 2013.

[24] A. Ksentini, T. Taleb, and M. Chen, "A Markov decision process-based service migration procedure for follow me cloud," in *2014 IEEE International Conference on Communications (ICC)*, pp. 1350–1354, Sydney, NSW, Australia, 2014.

[25] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. Chan, and K. K. Leung, "Mobility-induced service migration in mobile micro-clouds," in *2014 IEEE Military Communications Conference*, pp. 835–840, Baltimore, MD, USA, 2014.

[26] T. Ouyang, Z. Zhou, and X. Chen, "Follow me at the edge: mobility-aware dynamic service placement for mobile edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2333–2345, 2018.

[27] Y. Sun, X. Guo, S. Zhou, Z. Jiang, X. Liu, and Z. Niu, "Learning-based task offloading for vehicular cloud computing systems," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–7, Kansas City, MO, USA, 2018.

[28] Y. Sun, S. Zhou, and J. Xu, "Emm: energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2637–2646, 2017.

[29] T. Alfakih, M. M. Hassan, A. Gumaei, C. Savaglio, and G. Fortino, "Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA," *Access*, vol. 8, pp. 54074–54084, 2020.

[30] P. A. Apostolopoulos, E.-E. Tsiropoulou, and S. Papavassiliou, "Cognitive data offloading in mobile edge computing for Internet of Things," *IEEE Access*, vol. 8, pp. 55736–55749, 2020.

[31] J. Santa, A. F. Skarmeta, J. Ortiz et al., "MIGRATE: mobile device virtualisation through state transfer," *IEEE Access*, vol. 8, pp. 25848–25862, 2020.

[32] B. Liang and W. Ji, "Multiuser computation offloading for edge-cloud collaboration using submodular optimization," *Journal on Communications*, vol. 41, no. 10, pp. 25–36, 2020.

[33] H. Mezni, F. S. Hamoud, and F. B. Charrada, "Predictive service placement in cloud using deep learning and frequent

subgraph mining," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–20, 2022.

[34] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic resource allocation exploiting mobility prediction in mobile edge computing," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6, Valencia, Spain, 2016.

[35] R. Langar, N. Bouabdallah, and R. Boutaba, "A comprehensive analysis of mobility management in MPLS-based wireless access networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 918–931, 2008.

[36] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2020.

[37] T. Taleb, A. Ksentini, and P. A. Frangoudis, "Follow-me cloud: when cloud services follow mobile users," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 369–382, 2019.

[38] T. Ouyang, R. Li, X. Chen, Z. Zhou, and X. Tang, "Adaptive user-managed service placement for mobile edge computing: an online learning approach," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pp. 1468–1476, Paris, France, 2019.

[39] X. Yu, M. Guan, M. Liao, and X. Fan, "Pre-migration of vehicle to network services based on priority in mobile edge computing," *IEEE Access*, vol. 7, pp. 3722–3730, 2019.