2019 IEEE International Conference on Web Services (ICWS)

Mobility-aware and Migration-enabled Online Edge User Allocation in Mobile Edge Computing

Qinglan Peng[1], Yunni Xia[1*], Zeng Feng[2], Jia Lee[1], Chunrong Wu[1], Xin Luo[3*], Wanbo Zheng[4], Shanchen Pang[5], Hui Liu[6], Yidan Qin[1] and Peng Chen [7]

[1]Software Theory and Technology Chongqing Key Lab, Chongqing University, Chongqing, China
[2]DISCOVERY TECHNOLOGY (shenzhen) limited, Shenzhen, China
[3]Chinese Academy of Sciences, Chongqing Institute of Green and Intelligent Technology, Chongqing, China
[4]Data Science Research Center, Kunming University of Science and Technology, Kunming, China
[5]School of Computer and Communication Engineering, China University of Petroleum, Qingdao, China
[6]School of Software, Xinjiang University, Urumqi, China
[7]School of Computer and Software Engineering, Xihua University, Chengdu, China
**xiayunni@hotmail.com, luoxin21@gmail.com*

***Abstract*—The rapid development of mobile communication technologies prompts the emergence of mobile edge computing (MEC). As the key technology toward 5th generation (5G) wireless networks, it allows mobile users to offload their computational tasks to nearby servers deployed in base stations to alleviate the shortage of mobile resource. Nevertheless, various challenges, especially the edge-user-allocation problem, are yet to be properly addressed. Traditional studies consider this problem as a static global optimization problem where user positions are considered to be time-invariant and user-mobility-related information is not fully exploited. In reality, however, edge users are usually with high mobility and time-varying positions, which usually result in users reallocations among different base stations and impact on user-perceived quality-of-service (QoS). To overcome the above limitations, we consider the edge user allocation problem as an online decision-making and evolvable process and develop a mobility-aware and migration-enabled approach, named MobMig, for allocating users at real-time. Experiments based on real-world MEC dataset clearly demonstrate that our approach achieves higher user coverage rate and lower reallocations than traditional ones.**

***Keywords*-Edge User Allocation; Mobile Service Computing; Mobile Edge Computing; Mobility; Quality-of-Service;**

I. Introduction

Nowadays, mobile computing is becoming popular in supporting versatile computational-intensive applications, e.g., Augmented Reality (AR), Virtual Reality (VR), and Artificial Intelligence (AI) [1]. Nevertheless, the inherent scarcity of resources of mobile devices, such as memory and battery, poses an obstacle preventing mobile applications becoming more effective and successful. Various existing solutions [2]–[4] addressed this issue by offloading tasks to clouds to alleviate the shortage of mobile resources. For example, the stream project[1] powered by Google aims at building a cloud-based game platform where mobile users are allowed to enjoy high definition video games on their mobile devices without a full-game client. This platform lets all computational tasks performed in the remote clouds due to the shortage of resource and computing power of mobile devices. However, offloading tasks to remote clouds can be ineffective due to the fact that additional communication overhead and energy consumption [5], [6].

Fortunately, recent advances of mobile communication, especially 5th generation (5G) technologies [7], prompts the emergence of mobile edge computing (MEC) [8] and sheds lights on the above-mentioned problem. In the MEC paradigm, base stations are not only responsible for wireless communication but also equipped with a proper amount of computing infrastructures, i.e., MEC servers. Mobile application providers, such as Google, Facebook, or Netflix, are thus allowed to deploy their services to these MEC servers [8], [9]. In turn, mobile users are allowed to access these service directly without performing computational-intensive tasks by themselves or resorting to remote clouds. In this way, the majority of tasks are performed at MEC servers, which are featured by much less communication overhead and energy consumption than traditional clouds.

However, various challenges, especially the edge-user-allocation problem, are yet to be properly addressed. In the typical edge computing paradigm, MEC servers are usually equipped with lightweight computing components and limited storage. Besides, edge users can only reach services within the signal range of MEC servers and the user-server connections can be unstable due to user mobility. In case of a connection loss, a user has to re-connect another MEC server and probably experiences service interruptions which potentially affect to user-perceived quality-of-service (QoS) [10]. Therefore, QoS-guaranteed allocation of users

*Yunni Xia (xiayunni@hotmail.com) and Xin Luo (luoxin21@gmail.com) are the corresponding authors of this work.

[1]https://projectstream.google.com

to suitable MEC servers with maximized user coverage rate and minimized reallocation overhead becomes a hot issue.

Traditional methods in this direction usually consider that users are with time-invariant positions [11]–[14], and thus formulate it as a static optimization problem. However, such methods can be ineffective due to the fact that real-world edge users are usually with high mobility. Instead of considering time-invariant user positions and static global optimization, in this paper, we consider the edge user allocation problem as an online-decision-making and refinable one, and propose a mobility-aware and migration-enabled approach, short for MobMig, to solve it. Our proposed method involves multiple steps: 1) migrating edge users from every overloaded MEC server to other nearby servers to increase the user coverage rate; 2) allocating unhandled edge users to suitable MEC servers according to their motion status. Experimental results show that our approach clearly outperforms traditional ones in terms of user coverage rate and the number of reallocations.

II. Related Work

Recent years have witnessed the trend of mobile edge computing, through which data is allowed to be produced by internet of things (IoT) devices closer to where it is created instead of sending it across long routes to data centers or clouds. According to the latest Cisco global cloud index [15], by 2021, 75% of data produced by human, machine, or things will be stored or processed with the help of edge or fog computing. The core idea of mobile edge computing is that the computation should be performed near to data source [16]. As a novel paradigm, it also well complies pre-existing computing infrastructures such as cloud computing [17] or fog computing [18] in a flexible and on-demanding way to fill the resource gap between mobile devices and remote cloud.

As a promising computing paradigm, mobile edge computing has attracted a lot of attentions in recent years. Extensive studies are conducted toward the emergent problems such as optimal user allocation and task offloading in mobile edge environment. For example, Lai *et al.* [11] for the first time proposed the problem of optimal edge user allocation from the perspective of application providers. In their study, the optimal edge user allocation problem is formulated into a vector bin packing problem whose optimization targets are maximizing the user coverage and minimizing the number of MEC servers hired. Then they utilized an IBM CPLEX tool to solve it. However, they do not consider the mobility of edge users and assume that edge users are stationary with fixed positions. Chen *et al.* [19] proposed a game-theoretic approach for the multi-user computation offloading problem in the MEC environment. They formulated the distributed offloading-decision-making problem as a multi-user offloading game and designed a distributed offloading algorithm to achieve the Nash equilibrium between energy efficiency and offloading performance. However, Similar to [11], they considered that users are immobile. Yao *et al.* [20] develop a heuristic method for the cost-effective cloudlet server placement problem in the fog computing environment. In their study, they assumed that each access points in wireless networks can support a cloudlet server while hosting multiple mobile tasks and the deployment of cloudlet servers should be cost-effective while meeting user-constrained QoS constraints. Then they formulated the problem into a combinatorial optimization problem and developed a heuristic method to find the near-optimal solution. Tran *et al.* [21] proposed a solution for the problem of joint task offloading and resource allocation in the MEC environment. In their study, they first formulated the problem into a mixed-integer-non-linear-programing problem and then decomposed it into two separated problems, namely, resource allocation and task offloading. They formulated the former problem as a convex and quasi-convex optimization problem to find the optimal solution and they proposed a heuristic method to yield near-optimal solution for the later problem. However, their algorithms are with high time-complexity and thus could be inefficient when dealing with large-scale systems.

A careful investigation into aforementioned studies shows that they are yet limited in many ways: 1) many studies, e.g., [11], [19], [20], treated the user allocation problem as a static global optimization problem and aimed at finding the optimal or near optimal solutions. Such methods can be ineffective in dealing with real-world systems with real-time connectivity and performance changes, where timely decision-making is required, due to the fact that they are with high time-complexity. 2) various existing studies, e.g., [11], [22], [23], considered the distance between edge users and MEC servers as the major constraint and tended to ignore the motion information of users. Their resulting allocation schedules can thus have bad user-perceived performance when users are highly mobile. However, such mobility-related information, e.g., moving direction and speed, is supposed to be well exploited in guiding the decision making of allocation at real-time.

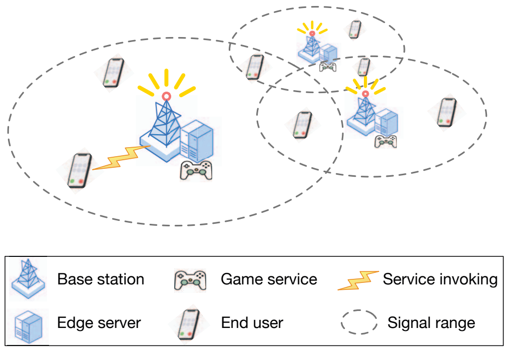


Figure 1: An example of application scenario.

III. System Model and Problem Formulation

Table I: List of notations

Notation	Description
$\vec{\alpha}$	The direction vector of users' movement
$\vec{\beta}$	The direction vector formed by (x, y) and (a, b)
θ	The angle between $\vec{\alpha}$ and $\vec{\beta}$
(a, b)	The position of base station
$d_{i,j}^t$	The distance between u_i and s_j at time t
m	The number of MEC servers in an area
r_i^t	A boolean indicator of whether u_i is reallocated at time t
s_j	The j-th base station in S
u_i	The i-th user in U_t
v_i^t	The speed of u_i at time t
$x_{i,j}^t$	A boolean indicator of whether u_i is allocated to server s_j at time t
(x, y)	The position of a user
A	An area
C_j^t	The capacity of base station s_j at time t
$E_{i,j}^t$	the expected moving distance for user u_i in the signal range of server s_j at time t
$M_{i,j}^t$	The expected duration time for edge user u_i to stay within the range of a MEC server s_j at time t
R_j	The signal range of base station s_j
S	The set of base stations
T	The working period
U_t	The set of users at time t

In the MEC environment, base stations are equipped with computing resource, i.e., MEC servers, and mobile application providers are allowed to deploy their services on these MEC servers to serve edge users. As a case illustrated in Figure 1, a game company deploys its game service on the MEC servers to improve the users' gaming experience. In this way, edge users offload a great portion of computation to the nearby MEC servers by invoking the deployed services instead of resorting to remote clouds with high latency or local infrastructures, e.g., mobile phones or laptops with restricted capacity and energy supply. In this way, users can enjoy high definition and low latency game. However, such offloading could fail when mobile users move out from the signal range of MEC servers which they are previously allocated to and the corresponding service is interrupted. Reallocation is thus required and a loss of QoS can be perceived by users.

The problem we are interested in can thus be formulated in the following way: there are total of m base stations $S = \{s_1, s_2, ..., s_m\}$ in an area A, each of them is equipped with an amount of computing and storage facilities and can work as an MEC server; the working period of MEC servers can be divided into T time slots; R_j is the signal range of station s_j, C_j^t the maximum capacity of users that can be served for MEC server s_j at time t; $U_t = \{u_1, u_2, ...\}$ denotes the users who need to connect the MEC services in area A at time t. Due to the mobility of edge users, $|U_t|$ can varies with the time and C_j^t is thus decided by the remaining resource of MEC servers at real-time as well. It is assumed that the computing resources of MEC servers are provided in an on-demand and pay-as-you-go way [9]. Service providers are thus allowed to scale their services elastically according to the requests from edge users. However, unlike cloud-based scaling usually with unlimited resource, scaling of MEC servers is subject to the resource limit and thus the maximum resource capacity that can be scaled to is usually bounded. The resulting problem is thus:

$$Max: \ mean(\sum_{t\in T}\sum_{i=1}^{|U_t|}\sum_{j=1}^{|S|} x_{i,j}^t) \tag{1}$$

$$Min: \ \sum_{t\in T}\sum_{i=1}^{|U_t|} r_i^t \tag{2}$$

$$s.t: \ \sum_{i=1}^{|U_t|} x_{i,j} \leq C_j^t \tag{3}$$

$$d_{i,j}^t \leq R_j, \ \exists x_{i,j}^t = 1 \tag{4}$$

$$j \in [1, m]$$

$$i \in [1, |U_t|]$$

where $x_{i,j}^t$ is a boolean indicator of whether u_i is allocated to server s_j at time t, r_i^t a boolean indicator of whether u_i is reallocated at time t, and $d_{i,j}^t$ the distance between user u_i and server s_j at time t.

As shown in (1) and (2), the optimization problem aims at both maximizing the average user coverage rate and minimizing the count of reallocations. (3) is the capacity constraint, i.e., the number of users that an MEC can serve is bounded. (4) is the geographical location constraint, i.e., users can only get served when they are within the signal range of MEC servers.

IV. Proposed Mobility-aware Approach

For the problem described in the previous section, we propose a mobility-aware and migration-enabled approach, short for MobMig, to yield refinable and dynamic allocation schedules at the real-time. It includes a mobility-aware allocation and a mobility-aware migration parts.

As shown in Algorithm 1, MobMig involves following steps: 1) finding the overloaded MEC servers. The status of an MEC server turns into overloaded when it has no remaining capacity for more users and there are users to be allocated in its signal range at the same time. An overloaded server turns back into an underloaded one if enough users on it are moved to other nearby MEC servers; 2) for each overloaded servers, finding the users who can be reallocated to nearby servers and the unallocated users within the signal range of the currently overloaded server (as shown in lines

Algorithm 1 MobMig

```
Input: mobile users $U$; MEC servers $S$;
1: for each server $s \in S$ do
2:   if $s$ is overloaded then
3:     $c \leftarrow$ get users allocated to $s$
4:     $u \leftarrow$ get users can be reallocated to nearby servers from $c$
5:     $p \leftarrow$ get unallocated users within the signal range of $s$
6:     MobilityAwareMigration($s$, $p$, $u$)
7:   end if
8: end for
9: $X \leftarrow$ get all unallocated users in $U$
10: MobilityAwareAllocation($X$)
```

3-5); 3) deciding which users are most suitable to be moved out, destination servers that they are going to be moved to, and which users are most suitable to be accepted (as shown in line 6); and 4) trying to allocate newly coming and pre-existing unallocated edge users (as shown in lines 9-10).

A. Mobility-Aware Allocation

The mobility-aware allocation part aims at allocating users to suitable MEC servers according to their motion status. As shown in algorithm 2, the proposed mobility-aware reallocation involves two main steps: 1) calculating the mobility-related fitness value M of every possible mapping between the unallocated edge users and their available MEC servers, and finding the most suitable MEC server, i.e., the MEC server with the highest fitness for every unallocated user by the way (as shown in lines 1-11); 2) sorting all the user-server mappings according to their fitness in a descending order and performing allocations according to such order (as shown in lines 12-15). In this way, the allocations with higher fitness are performed first to avoid unnecessary reallocations. The time complexity of the algorithm is $O(|U_t| \times m)$.

The mobility-related fitness value M is calculated as the expected duration time for a user to stay within the range of a certain MEC server and it indicates the suitability of a user-server mapping, i.e., allocation of a user to a certain server. As shown in Figure 2, the M value can be calculated as follow:

$$M_{i,j}^t = \frac{E_{i,j}^t}{v_i^t} \tag{5}$$

where $E_{i,j}^t$ is the expected moving distance for user u_i in the signal range of server s_j at time t, v_i^t the speed of u_i at time t. We use $d_{i,j}^t$ to denote the distance between u_i and s_j at time t. Therefore, the expected moving distance $E_{i,j}^t$ can be calculated as follow:

$$E_{i,j}^t = \sqrt{R_j^2 - (d_{i,j}^t)^2 + 2R_j d_{i,j} \cos\theta} \tag{6}$$

Algorithm 2 MobilityAwareAllocation

```
Input: unallocated users $X$;
1: for each user $x \in X$ do
2:   $S \leftarrow$ get the available MEC servers for user $x$
3:   $s \leftarrow 0$;  $M \leftarrow -\infty$
4:   for each server $s' \in S$ do
5:     $M' \leftarrow$ calculate $M_{x,s}$ according to E.q (5)
6:     if $M' > M$ then
7:       $s \leftarrow s'$;  $M \leftarrow M'$
8:     end if
9:   end for
10:  $x.s \leftarrow s$;  $x.M \leftarrow M$
11: end for
12: sort $X$ according to its $M$ value in a descending order
13: for each user $x \in X$ do
14:   allocate user $x$ to server $x.s$
15: end for
```

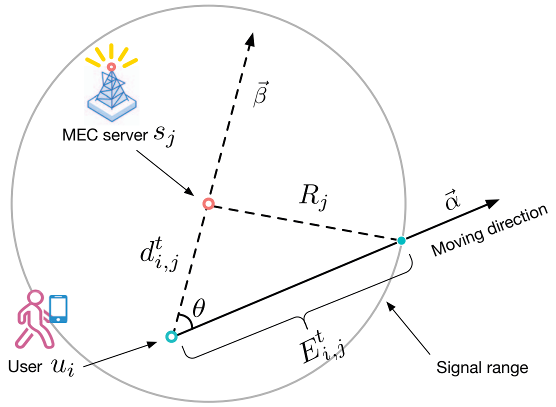


Figure 2: Deciding the fitness M.

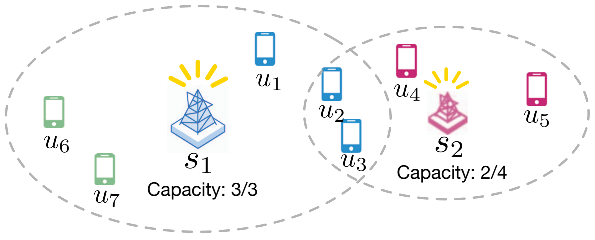
where θ is the angle between direction vector $\vec{\alpha}$ and $\vec{\beta}$. As shown in Figure 2, $\vec{\alpha}$ is the moving direction vector of an edge user, which usually obtained through its GPS device. $\vec{\beta}$ is the direction vector pointing from user to server. Suppose that the position of u_i is (x, y) and s_j is (a, b), then we have that $\vec{\beta} = (a - x, b - y)$. Angle θ can thus be calculated as:

$$\theta = \arccos \frac{\vec{\alpha} \cdot \vec{\beta}}{|\vec{\alpha}| \times |\vec{\beta}|} \tag{7}$$

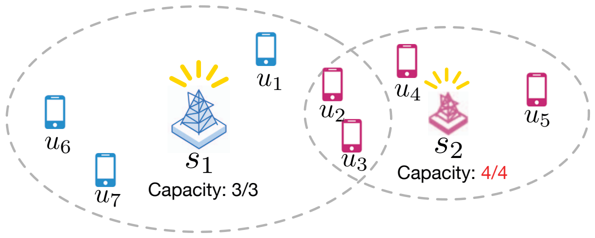
B. Mobility-Aware Migration

The mobility-aware migration aims at increasing the coverage rate of users through consolidating users which are already allocated. Figure 3 shows an example of migration.

In Figure 3(a), MEC servers s_1 and s_2 are capabilities of 3 and 4 respectively. Users u_1, u_2, and u_3 are allocated to s_1. u_4 and u_5 are allocated to s_2. As newly coming users, u_6 and u_7 are refused by s_1 due to the load constraint. In this case, the user coverage is $5/7 \approx 71\%$. It's clear



(a) Before migration.



(b) After migration.

Figure 3: An example of migration.

to see that an edge user migration from s_1 to s_2 certainly brings extra capacity. As shown in Figure 3(b), u_2 and u_3 are migrated to s_2 and the saved capacity for s_1 accommodates newly coming users u_6 and u_7. The resulting user coverage is increased to $7/7 = 100\%$.

However, deciding the most appropriate destination user and server from multiple candidates itself can be complicated and the resulting problem is a kind of knapsack problems which are well-acknowledged to be NP-hard [24]. We propose a mobility-aware and Best-Fit-Decreasing (BFD)-based algorithm, i.e., Algorithm 3, to solve it. As shown in the pseudocodes, B denotes the users that can be moved out for an overloaded server o, U the unallocated users within the range of o. The algorithm involves three steps: 1) evaluating the M value of users in B and U (as shown in lines 1-3); 2) sorting the users in U and B according their M value in a descending order and merging them into a migration plan p with length of $|B|$ (as shown in lines 4-5). Figure 4 shows an example of such a process; 3) comparing each element in p and B, if $p[i] \neq B[i]$, $B[i]$ will be moved out to the available MEC server with highest M value and $p[i]$ will be allocated to server o (as shown in lines 6-19). The time complexity of such a mobility-aware migration is $O(|U_t| \times m)$.

V. Experiments and Analysis

In this section, we conduct a series of case studies based on a real-world MEC environment to evaluate our approach.

We choose a 6.2 km^2 central business district (CBD) area in Melbourne, Australia, as the test case. The geographic information including the map and coordinates are from Google Map[2], and the geographical location of the base stations within this area, e.g., longitude and latitude, are from Australian Communications and Media Authority (ACMA) as [11] did. Figure 5 shows a part of base stations in this area and their signal range. There are totally 126 base stations and every corner in this area is covered by the signal range.

Algorithm 3 MobilityAwareMigration

```
Input: overloaded server $o$; unallocated users $U$; users can
       be moved out $B$;
 1: for each user $u \in U \cup B$ do
 2:     $u.M \leftarrow$ calculate $M_{u,o}$ according to E.q (5)
 3: end for
 4: sort users in $U$ and $B$ according to its $M$ value in
    descending order
 5: $p \leftarrow$ merge $U$ and $B$ into a migration plan according to
    their $M$ value, and keep its length same to $|B|$
 6: for $i = 1$ to length($p$) do
 7:     if $p[i]$ != $B[i]$ then
 8:         $S \leftarrow$ get the available MEC servers for user $B[i]$
 9:         $s \leftarrow 0$;  $M \leftarrow -\infty$
10:         for each server $s' \in \{S - o\}$ do
11:             $M' \leftarrow$ calculate $M_{B[i],s'}$ according to E.q (5)
12:             if $M' > M$ then
13:                 $s \leftarrow s'$;  $M \leftarrow M'$
14:             end if
15:         end for
16:         reallocate user $B[i]$ to the server $s$
17:         allocate user $p[i]$ to server $o$
18:     end if
19: end for
```

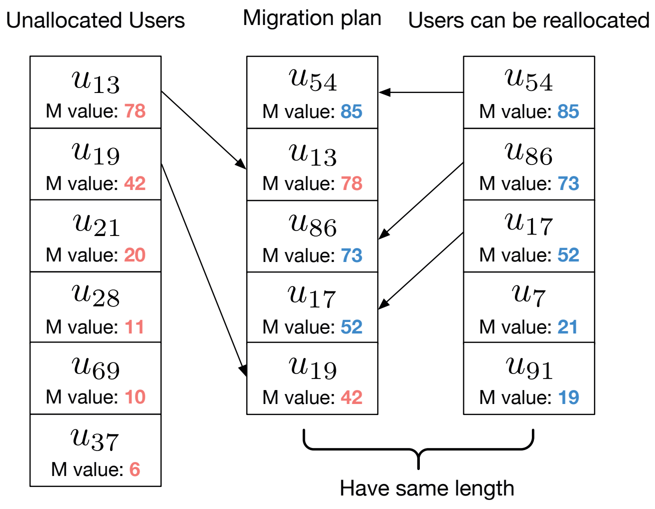


Figure 4: An example of migration plan generation.

[2]https://www.google.com/maps/@-37.8141369,144.9624365,15.67z

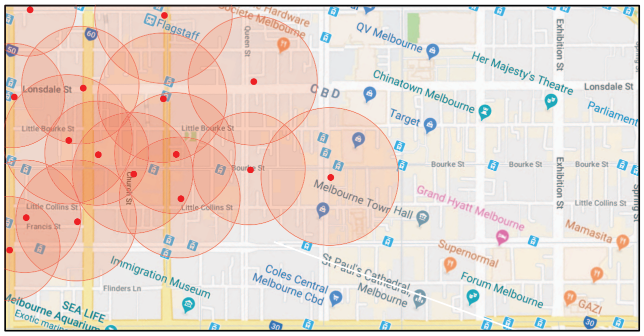


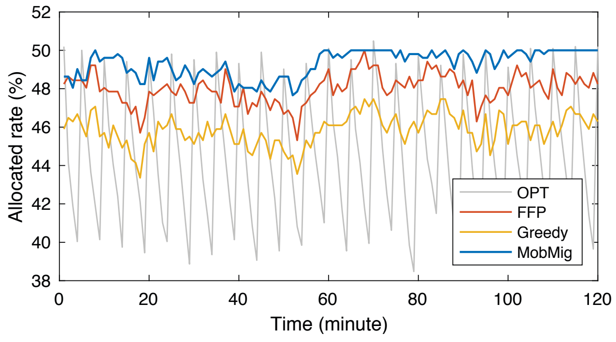
Figure 5: The experimental environment.

In our experiments, each base station is an MEC server. The capacity of MEC servers in this area is set to be able to support 50%∼150% as many edge users. We use a Random-Waypoint (RWP) mobility model [25] to generate the edge users' moving trajectories within 2 hours. RWP model is widely used for the simulation of mobile ad hoc networks [26]. In our experiments, we consider that there are totally 512 edge users. 80% of them are pedestrians with the speed of 1∼3 MPH and 20% of them are by vehicles with the speed of 6∼24 MPH.

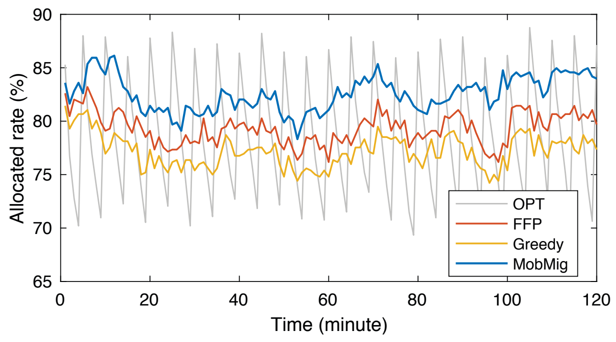
We consider OPT, FFP, and Greedy as baseline algorithms:

- **OPT** [11]: it regards the allocation problem as a static global optimization problem and utilizes an IBM CPLEX tool to yield optimal solutions.
- **FFP** [27]: it is an online heuristic for cloud server allocation, where allocation is formulated as a dynamic bin packing problem. It can also be used for allocating edge users.
- **Greedy**: it allocates unserved users to the nearest available MEC servers.

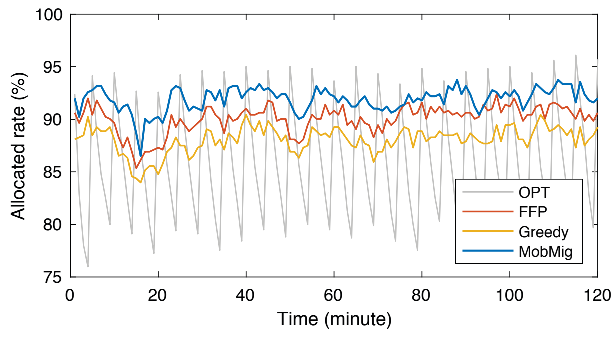
Figure 6 and Table II shows the comparisons of the user coverage rate between our approach and its peers at three different capacity levels. Clearly, our method, i.e., MobMig, achieves higher user coverage (5.07% higher than OPT on average; 4.67% higher than FFP on average; and 2.13% higher than Greedy on average) in all three cases. Besides, MobMig can also achieve more stable user coverage rates in most of cases, which reflects in a lower standard deviation of user coverage rates as shown in Table I.



(a) Total capacity: 50%



(b) Total capacity: 100%

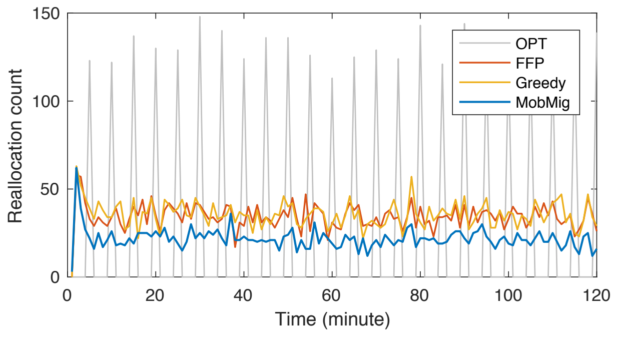


(c) Total capacity: 150%

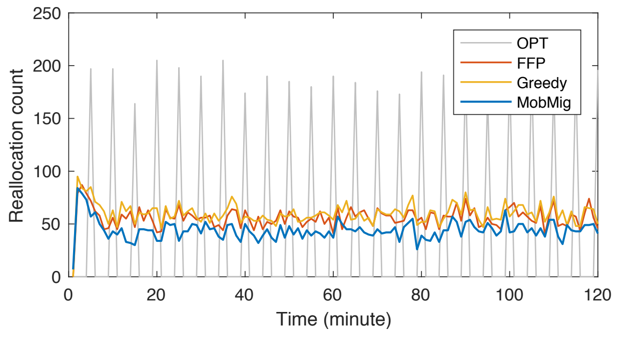
Figure 6: User coverage rate comparison.

Table II: User coverage

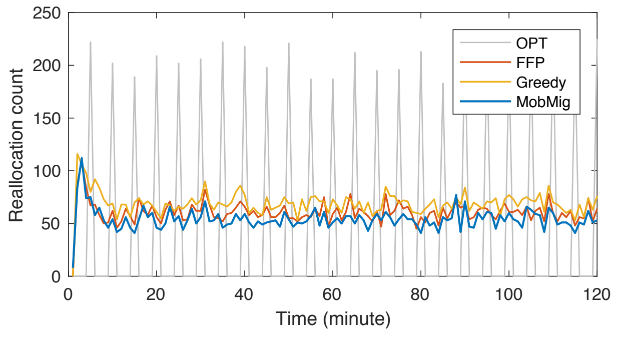
Capacity	Algorithm	Average coverage	Std
50%	OPT	45.85%	3.54%
	FFP	48.01%	0.79%
	Greedy	44.44%	1.53%
	MobMig	**49.31%**	**0.69%**
100%	OPT	78.63%	5.74%
	FFP	79.32%	1.49%
	Greedy	77.19%	1.53%
	MobMig	**82.54%**	1.64%
150%	OPT	84.07%	5.68%
	FFP	90.04%	1.31%
	Greedy	88.06%	1.25%
	MobMig	**91.90%**	**1.11%**



(a) Total capacity: 50%



(b) Total capacity: 100%



(c) Total capacity: 150%

Figure 7: The number of reallocations comparison.

Table III: The number of reallocation

Capacity	Algorithm	No. of reallocation	Std
50%	OPT	3116	52.29
	FFP	4116	7.21
	Greedy	4249	7.72
	MobMig	**2610**	**6.01**
100%	OPT	5193	75.95
	FFP	6709	9.99
	Greedy	7188	10.38
	MobMig	**4929**	**9.38**
150%	OPT	6501	83.29
	FFP	7260	10.66
	Greedy	8283	11.57
	MobMig	**6365**	**10.36**

Figure 7 and Table III shows the comparisons of the number of reallocations at there different capacity levels. We can clear see that our approach can achieve lower reallocation count (6.12% lower than OPT on average; 29.49% lower than FFP on average; and 23.12% lower than Greedy on average) in all cases. We can also find form the Table II that the standard deviation of the reallocation count of our method is the lowest one among its peers, which guarantees more stable user-perceived QoS when faced with complex situation where edge users are with high mobility.

Intuitively, the disadvantage of traditional approaches lies in that they ignore the mobility of edge users and tread edge user allocation as a static optimization problem. On the contrary, in our method, the motion status of edge user is well exploited to yield a smarter, mobility-aware allocation to reduce the occurrence of reallocation. Besides, the migration feature of our method also has the ability to further improve the user coverage rate.

VI. Conclusion and Further Study

This paper target at the edge user allocation problem in mobile edge computing environment. Instead of considering user allocation as a static optimization problem, we regard it as a online allocation decision making and evolvable process. We also take the mobility of edge users into consideration and propose a mobility-aware and migration-enabled approach to make an online allocation. Experimental results show that our approach can achieve higher user coverage rate and lower reallocations compared with traditional ones.

In our future study, we will consider the following features: 1) some trajectory prediction method, e.g., social LSTM and social GAN neural networks, can be employed to predict the future position of users to further improve the quality of allocations; 2) we only consider one application provider in this paper, multiple application providers compete for limited MEC resources provided by multiple communication providers in an on-spot way will be considered in our future work; 3) we only consider one hop migration in this paper, the cascading user migration will be considered in our future work to achieve a higher user coverage rate.

Acknowledgment

This work is supported in part by the International Joint Project funded jointly by the Royal Society of the UK and the National Natural Science Foundation of China under grant 61611130209, the National Science Foundations of China under grants Nos.61472051/61702060, the Fundamental Research Funds for the Central Universities under Grant 2019CDXYJSJ0022, the Science Foundation of Chongqing under No. cstc2017jcyjA1276, China Postdoctoral Science Foundation No. 2015m570770, Chongqing Postdoctoral Science special Foundation No. Xm2015078, Universities' Sci-tech Achievements Transformation Project of Chongqing No. KJZH17104, Chongqing grand R&D

projects Nos. cstc2017zdcy-zdyf0120 and cstc2017rgzn-zdyf0118, Natural Science Foundation of Chongqing under Grant cstc2016jcyjA1315, and National Key R&D Program of China2018YFD1100304.

REFERENCES

[1] L. Yang, B. Liu, J. Cao, Y. Sahni, and Z. Wang, "Joint computation partitioning and resource allocation for latency sensitive applications in mobile edge clouds," *IEEE Transactions on Services Computing*, pp. 1–1, 2019.

[2] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation offloading for service workflow in mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 12, pp. 3317–3329, 2015.

[3] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, 2015.

[4] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE*. IEEE, 2016, pp. 1–9.

[5] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[6] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future generation computer systems*, vol. 29, no. 1, pp. 84–106, 2013.

[7] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5g be?" *IEEE Journal on selected areas in communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[8] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[9] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 workshop on mobile big data*. ACM, 2015, pp. 37–42.

[10] S. Scholz and H. Grob-Lipski, "Reallocation strategies for user processing tasks in future cloud-ran architectures," in *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*. IEEE, 2016, pp. 1–6.

[11] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, and Y. Yang, "Optimal edge user allocation in edge computing with variable sized vector bin packing," in *International Conference on Service-Oriented Computing*. Springer, 2018, pp. 230–245.

[12] L. Yang, B. Liu, J. Cao, Y. Sahni, and Z. Wang, "Joint computation partitioning and resource allocation for latency sensitive applications in mobile edge clouds," *IEEE Transactions on Services Computing*, 2019.

[13] M. Liu, F. R. Yu, Y. Teng, V. C. Leung, and M. Song, "Distributed resource allocation in blockchain-based video streaming systems with mobile edge computing," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 695–708, 2019.

[14] Z. Song, Y. Liu, and X. Sun, "Joint radio and computational resource allocation for noma-based mobile edge computing in heterogeneous networks," *IEEE Communications Letters*, vol. 22, no. 12, pp. 2559–2562, 2018.

[15] C. G. C. I. Cisco, "Forecast and methodology, 2016–2021 (2018)," *White Paper*.

[16] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[17] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[18] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[19] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, no. 5, pp. 2795–2808, 2016.

[20] H. Yao, C. Bai, M. Xiong, D. Zeng, and Z. Fu, "Heterogeneous cloudlet deployment and user-cloudlet association toward cost effective fog computing," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 16, p. e3975, 2017.

[21] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, 2018.

[22] S. Deng, L. Huang, D. Hu, J. L. Zhao, and Z. Wu, "Mobility-enabled service selection for composite services," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 394–407, 2016.

[23] S. Deng, L. Huang, J. Taheri, J. Yin, M. Zhou, and A. Y. Zomaya, "Mobility-aware service composition in mobile communities," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 3, pp. 555–568, 2017.

[24] M. R. Garey and D. S. Johnson, *Computers and intractability*. wh freeman New York, 2002, vol. 29.

[25] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile computing*. Springer, 1996, pp. 153–181.

[26] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless communications and mobile computing*, vol. 2, no. 5, pp. 483–502, 2002.

[27] R. Ren, X. Tang, Y. Li, and W. Cai, "Competitiveness of dynamic bin packing for online cloud server allocation," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1324–1331, 2017.