This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

Online UAV-mounted Edge Server Dispatching for Mobile-to-Mobile Edge Computing

Jingrong Wang, *Student Member, IEEE,* Kaiyang Liu, *Member, IEEE,* and Jianping Pan, *Senior Member, IEEE*

***Abstract*—Mobile edge computing has been considered as a promising technology to handle computation-intensive and delay-sensitive tasks in the Internet of Things (IoT) ecosystem such as smart city and smart tourism. However, due to user mobility, edge servers with fixed deployment are not flexible enough to handle time-varying user tasks in hot-spot areas. In this paper, a novel online unmanned aerial vehicle (UAV)-mounted edge server dispatching scheme is proposed to provide flexible mobile-to-mobile edge computing services. UAVs are dispatched to the appropriate hover locations by geographically merging tasks into several hot-spot areas. Theoretical analysis guarantees the worst-case performance bound. Extensive evaluation driven by real-world mobile requests shows that while maintaining a good latency fairness, the mobile server dispatching scheme can serve more UEs as well as achieving a high resource utilization. Moreover, the hybrid scheme can satisfy even more user demands while dispatching fewer UAVs with a higher server utilization.**

***Index Terms*—Mobile edge computing, mobile edge server, online dispatching scheme**

I. Introduction

RECENT years have witnessed the flourish of the Internet of Things (IoT) techniques to provide real-time analysis for smart cities, intelligent transportation, entertainment management, etc. To support a lot of devices and process massive data in time, mobile edge computing (MEC) enables these latency-sensitive applications by equipping ubiquitous computation resources close to mobile user equipments (UEs). Various research topics, e.g., task offloading, caching, and resource allocation, are all based on the assumption that the edge servers have been placed already [1]–[3]. How and where to deploy the edge servers need to be addressed.

Several solutions have been proposed to locate edge servers, among which, deploying more edge servers to hot-spot areas outperforms the uniform deployment [4]. Due to user mobility and dynamic demands, hot-spot areas at the current time may cool down soon afterward. To serve time-varying crowds, Yin et al. [5] mapped user clusters to the fixed edge servers periodically to reduce the infrastructure cost. However, due to unevenly distributed tasks, some fixed edge servers are unavoidably overloaded while other servers are idle. Therefore, techniques such as task migration need to be introduced to balance the workload among edge servers. This, in turn, results in extra communication and signaling overhead, and increases task latency as the tasks need to be transferred between servers [6]. On-demand network deployment has been seen as a promising proposal to serve dynamic hot-spot areas for big events or disaster. Meanwhile, it can improve the computing resource utilization with the on-demand provisioning when compared with the base station (BS) sleeping technologies.

Recently, UAVs have been extensively studied and acknowledged as a feasible way to assist the wireless communication networks [7]. With the development of UAVs, deploying edge servers on them draws significant attention due to their flexible mobility [8]. To overcome the limitations of flight time and battery power, existing solar power techniques can achieve 28+ continuous flight hours [9]. In addition, UAVs can also be powered over the tether, which provides unlimited flight time [10]. Moreover, general commercial UAVs such as DJI Matrice 600, DJI S900 and Tarot T-18 can take off with 6∼8 kg payloads while heavy lift drones can fly with up to 45 kg goods such as Hx8 Power XXL. This makes it sufficient for UAVs to carry a server and hover at specific places to collect and process the offloaded tasks. Recently, a prototype named SkyCore was built to support on-demand connectivity [11]. Network functions are softwarized and located in a single-board light-weight server, which can be directly deployed on DJI Matrice 600 Pro drones. The synchronization overhead of inter-UAV communication is reduced by segment-based routing with the label of the next tunnel segment tagged on the packets. Real-world experiment shows that mobile edge servers can not only provide timely services for certain hot-spot areas but also take advantage of their location flexibility to deal with the dynamic environment with negligible synchronization overhead.

Most existing work focused on UAV trajectory planning in which the location of UEs remains unchanged and UAVs maintain a continuous flight among several fixed UEs [12]–[14]. The limitation of the existing work lies in either 1) UE mobility which causes dynamic nonuniform tasks, or 2) network scalability, i.e., when a large number of UEs offload tasks, UAV trajectory will be affected by each individual UE, which is time-consuming and expensive to adjust. Observing the inefficiency of fixed edge server deployment and the limitation of the current UAV trajectory planning, we are motivated to investigate how to dispatch UAVs to appropriate hover locations among time-varying hot-spot areas and associate mobile UEs with mobile edge servers.

In this paper, mobile-to-mobile edge computing is considered in which both UEs and edge servers can move around. UAV-mounted edge servers are employed for flexible edge

This work was supported in part by NSERC, CFI and BCKDF, and in part by China Postdoctoral Science Foundation. (Corresponding author: Kaiyang Liu.)

J. Wang, K. Liu, and J. Pan are with the Department of Computer Science, University of Victoria, Victoria, BC, V8W 2Y2, Canada. (E-mail: {jingrongwang, liukaiyang, pan}@uvic.ca).

K. Liu is with the School of Information Science and Engineering, Central South University, Changsha, 410075, China. (E-mail: liukaiyang@csu.edu.cn).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

services. Constrained by the limited computation capacity and communication range, the edge server dispatching problem is formulated as a variable-sized bin-packing problem with geographic constraints, which is NP-hard [15]. An online mobile edge server dispatching scheme is proposed to determine the hover locations of the mobile edge servers sequentially. With the gradually increased communication radius, hot-spot areas are identified based on the task intensity. The performance of the proposed scheme is theoretically analyzed with the worst-case performance guarantee. A hybrid scheme is also evaluated in which UAVs are dispatched to assist the fixed BSs with task offloading. Simulation results show that the mobile dispatching scheme excels at handling dynamic nonuniformly distributed tasks and maintaining a good task latency fairness. When compared with the fixed server deployment, the number of served UEs increases 59% on average. The server utilization achieves 98% during the daytime. In addition, the hybrid scheme can satisfy even more demands while dispatching fewer UAVs with a better server utilization.

The rest of the paper is organized as follows. The related work is introduced in Section II. The motivation for deploying mobile edge servers is illustrated in Section III. The communication model and the computation model are introduced in Section IV. In Section V, the mobile dispatching problem is formulated and an online mobile edge server dispatching scheme is proposed with the performance guarantee. The performance of the proposed algorithm and the impact of key parameters are evaluated and illustrated in Section VI. Section VII presents the conclusion and future work.

II. Related work

A. Fixed edge server placement

Most existing work in MEC assumed that edge servers are deployed following a certain distribution such as uniform distribution [1]–[3]. How to locate edge servers has been heavily studied, which plays an important role in improving the quality of service (QoS). Li et al. [4] compared the performance of two different edge server deployment schemes, i.e., uniform distribution and nonuniform distribution based on UE density. Evaluation results showed that UE distribution-aware server deployment can achieve a better performance than the uniform distribution. Facing dynamic crowds, Yin et al. [5] first used farthest point clustering to group UEs and calculated the ideal locations of edge servers in each cluster by minimizing the total communication distance. Wang et al. [16] located the edge servers by solving the mixed integer programming problem. Lai et al. [17] deployed edge servers and maximized the number of served UEs through lexicographic goal programming. However, existing optimization problems are formulated based on selected BS locations among which edge servers are deployed.

Moreover, UE mobility can not only affect mobility management in mobile networks but also influence the network workload dynamically. Ceselli et al. [18] located cloudlet facilities among the candidate locations and introduced VM migration to re-balance the system. Locations of APs and cloudlets are determined based on the fixed locations of aggregation nodes.

Due to dynamic nonuniformly distributed tasks, the limitations of the fixed deployment are: 1) multi-hop communications are needed if the available edge servers are not close enough to mobile UEs, and 2) computation resources cannot be fully utilized at off-peak hours.

B. Mobile edge server trajectory planning

To tackle the limitations of the fixed server deployment, extensive efforts have been dedicated to deploying edge servers on UAVs. Considering the flexible movement of UAVs, Cheng et al. [8] designed the architecture of UAV-BS integrated mobile edge network for road safety scenarios. UAVs are dispatched to the area of interest and help process computation-intensive tasks. Zhou et al. [13] and Cheng et al. [14] determined the UAV trajectory by solving the mixed-integer non-convex problem with the objective of computation rate maximization and communication rate maximization, respectively. Similarly, Jeong et al. [12] jointly optimized the UAV trajectory as well as the bit allocation for both communication and computation purposes. The formulated energy consumption minimization problem is then solved by successive convex approximation. Hence, mobile edge servers take advantage of handling dynamic nonuniform tasks and avoiding the waste of computation resources. However, most existing work does not consider the time and space-varying features of user tasks. Thus, this paper investigates how to dispatch UAV-mounted edge servers to dynamic hot-spot areas.

III. Motivation

A. Tencent trace description

Real-time mobile request distribution is of great importance in MEC. It can not only provide us with the geographic information of a single mobile request but also a macroscopic view of the dynamic changes. Benefited from the development of GPS-enabled devices, mobile UEs are offered geo-spatial and point of interest (POI)-related services.

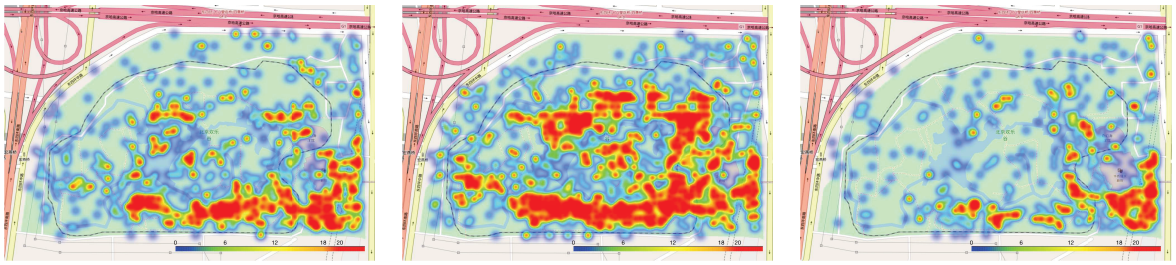
Tencent provides real-time Tencent user density data (RTUD) based on the collected geographic information when UEs are using its services[1]. According to the Tencent Big Data report, more than 1.3 billion monthly active devices are using Tencent location-based applications, e.g., WeChat and QQ, in 2018 [19]. In RTUD traces, UEs are dynamically distributed due to UE mobility. The device location (latitude, longitude, and region ID) and query time slot index are provided for each UE request. Then, the intensity of the geo-spatial requests can be derived from the traces. The time interval of the query time slot is 5 min.

B. Dynamic task requests

With the idea of IoT, smart theme park integrates distributed sensors and mobile devices to enhance the user experience such as UAV-assisted interactive attractions, traffic congestion prediction, tour design and finding missing people. Happy Valley, a theme park in Beijing, is selected as the focused

[1]The geographic information of the mobile requests can be found at https://heat.qq.com/heatmap.php.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal



(a) 9:30 AM. (b) 3:30 PM. (c) 9:30 PM.

Fig. 1: Tencent requests distribution on Oct. 1st, 2018 (a national holiday in China) at Happy Valley, Beijing.

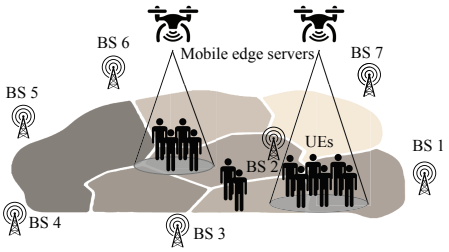


Fig. 2: System model of MEC with UAV-mounted edge servers.

scenario in this paper due to the following features: 1) the size of the park is reasonable ($1,000$ m $\times$ 500 m)—neither too small that all requests can be covered by one BS and thus multiple UAVs are not needed, nor too large so that UAVs cannot reach specific locations on time, and 2) dynamic requests—UEs have obvious group effect and form different dense crowds with bursty requests over time. As shown in Fig. 1, dynamic requests are nonuniformly distributed in the park on Oct. 1st, 2018. UEs keep forming hot-spot areas at different places. In the morning, people gather at the entrance and then enter the park to take amusement rides. At night, people gradually exit the park and it is nearly empty.

As shown in Fig. 2, the locations of the fixed LTE BSs are mostly around the theme park[2]. In MEC, if the edge server is deployed at each BS, it will result in an unbalanced workload among BSs and fewer served UEs. When UEs offload tasks to edge servers, some BSs are fully utilized such as the ones located near the entrance (BS 1, BS 2 and BS 3). However, other BSs located along the road outside the park are underloaded and thus their computation resources are wasted. Even with a better server placement scheme, the fixed deployment scheme still faces the problem of computation resource inefficiency in a long run. In contrast, mobile edge servers can be dispatched to handle burst requests in hot-spot areas. With fewer requests, some UAVs can stay on while others can fly back to the warehouse for maintenance.

How to dispatch mobile edge servers effectively and efficiently interests us most. Mobile edge servers can be dispatched closer to the crowds to serve UEs as much as possible, and meanwhile, take advantage of their flexibility to increase server utilization.

[2]The location of the fixed LTE BSs in real world can be obtained from https://opencellid.org.

TABLE I: Definition and notation.

Symbol	Definition and notation
$\mathcal{N}$	Set of mobile edge servers, and $N = \lvert\mathcal{N}\rvert$
$\mathcal{M}$	Set of geo-spatial tasks, and $M = \lvert\mathcal{M}\rvert$
$(u_n, v_n, 0)$	Coordinates of task m, $m \in \mathcal{M}$
(u_m, v_m, h)	Coordinates of mobile edge server n, $n \in \mathcal{N}$, at height h
d_{mn}	Distance between server n and task m
γ_0	Reference SNR at a distance of 1 m
N_0	Noise power
A	Data size of each task
$B(d_{mn})$	Data transmission rate which depends on the distance between mobile UEs and edge servers
x_{mn}	Binary variable: task m is served by mobile edge server n ($x_{mn} = 1$) or not ($x_{mn} = 0$), $m \in \mathcal{M}$, $n \in \mathcal{N}$
S_n	Set of tasks assigned to mobile edge server n, $n \in \mathcal{N}$
r_n	Radius of the coverage of mobile edge server n, $r_{\min} \le r_n \le r_{\max}$, $n \in \mathcal{N}$
Δr	Increment of the radius r
P	Computation intensity of each task
Q^{E}	Computation capacity of a mobile edge server in cycles
φ	Latency deadline of each task
ϕ	Total budget for mobile edge servers

IV. System Model

In this section, the communication and computation models of MEC are introduced. The major notations used in this paper are summarized in Table I. As shown in Fig. 2, BSs and UAVs are all equipped with edge servers. Generally, UE $m \in \mathcal{M}$ (with $\lvert\mathcal{M}\rvert = M$) can offload its computing task to edge server $n \in \mathcal{N}$ (with $\lvert\mathcal{N}\rvert = N$) through either the fixed BSs or access points on UAVs. Tasks at the same location will be processed one by one. If no confusion arises, we will use m to denote tasks in the following statement instead of UEs.

A. Communication model

Let (u_n, v_n, h) and $(u_m, v_m, 0)$ denote the coordinates of mobile edge server n and task m, respectively. Mobile servers are assumed to hover at the same height h and can adjust the communication coverage through their antenna angle [20]. The distance between server n and task m can be calculated as

$$\begin{aligned} d_{mn} &= \|(u_n, v_n, h) - (u_m, v_m, 0)\|, \\ &= \sqrt{(u_n - u_m)^2 + (v_n - v_m)^2 + h^2}, \end{aligned} \tag{1}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

where $\|\cdot\|$ is the Euclidean norm.

The path loss of UAV channel can be modeled as

$$PL_{mn} = PL(d_0) + 10\varsigma \log\left(\frac{d_{mn}}{d_0}\right), m \in \mathcal{M}, n \in \mathcal{N}, \quad (2)$$

where ς is the path-loss exponent by jointly considering UAV line-of-sight (LoS) and non-LoS (NLoS) channel, and $PL(d_0)$ is the path loss at the reference distance d_0. The data transmission rate can be obtained by

$$B(d_{mn}) = B_0 \log(1 + \frac{P^{\mathrm{TX}}|h_{mn}|^2}{N_0}), \quad (3)$$

where B_0 is the channel bandwidth, P^{TX} is the transmission power of UEs, N_0 is the noise power, and h_{mn} is the total channel gain which consists of pathloss, the effect of shadowing and the effect of multipath fast fading.

B. Computation model

Let x_{mn} denote whether task m is offloaded to edge server n ($x_{mn} = 1$) or not ($x_{mn} = 0$). As the computation capacity of each edge server is limited, if task m is offloaded to edge server n, the task latency D_m^{Offload} consists of the time for communication $D_m^{\mathrm{Comm.}}$, i.e., sending the task and receiving the results, and the computation latency $D_m^{\mathrm{Comp.}}$, which can be calculated as

$$\begin{aligned} D_m^{\mathrm{Offload}} &= D_m^{\mathrm{Comm.}} + D_m^{\mathrm{Comp.}} \\ &= \frac{A}{B(d_{mn})} + \frac{\omega}{Q^{\mathrm{E}}}, \end{aligned} \quad (4)$$

where $A/B(d_{mn})$ is the communication latency, and $B(d_{mn})$ is the data transmission rate with the distance d_{mn}. Assume that each task has the same data size A. This work can be extended to variable-sized tasks as the tasks can be divided into small tasks equally. ω/Q^{E} is the queuing latency of task offloading, $\omega = P + P'$ is the computation workload P (in CPU cycles) of task m plus the current workload P' of the assigned edge server n, and Q^{E} is the CPU processing capacity (in CPU cycles/s) of the edge server. Several VMs are deployed at each edge server. If the number of offloaded tasks exceeds the number of VMs, tasks will be queued up and executed in a first-come-first-serve (FCFS) manner.

V. Online mobile server dispatching scheme

In this section, an efficient online UAV-mounted edge server dispatching scheme is introduced to provide mobile-to-mobile services. The optimization problem is first formulated as the variable-sized bin-packing problem with the geographic constraint. Then, inspired by the heap-map generation, a greedy dispatching algorithm is proposed to determine the hover locations of the edge servers in an online fashion. Theoretical analysis is also provided with the worst-case performance guarantee.

A. Problem formulation

Considering the dynamic crowds and nonuniform distribution of task requests, mobile edge servers can be deployed flexibly to provide offloading services timely and spatially. The hover locations of the mobile edge servers need to be coordinated with time-varying task requests. Intuitively, more UAV-mounted edge servers should be dispatched to serve the dense crowds with more requests. Our purpose is to serve tasks as many as possible with a given task deadline. If we assign all tasks to one server, although the server facility cost is the least, the number of served tasks is limited by the computation capacity and thus the server cannot finish all tasks on time. If we assign tasks to several servers inefficiently, the deployed computation resources of some servers are wasted. This features items (tasks) and bins (UAV-mounted servers), making it feasible to be formulated as a bin-packing problem.

Bin-packing problems have been widely studied to associate a set of items to a set of bins. In the original bin-packing problem, the goal is to minimize the number of used bins or alternatively maximize the pre-defined profit with a fixed number of bins. In the server dispatching problem, mobile servers are dispatched based on task distribution and their profits can be defined as the number of served tasks given a task deadline. We aim to determine the hover locations and the corresponding communication range of mobile edge servers such that the number of served tasks is maximized. With the objective of maximizing the number of served tasks at each time slot, a variable-sized bin-packing problem with the geographic constraint (Geo-VBP) can be formulated as follows

$$\max_{((u_n,v_n,h),r_n)} \quad \sum_{n\in\mathcal{N}}\sum_{m\in\mathcal{M}} x_{mn}, \quad (5)$$

$$\text{s.t.} \quad x_{mn} \in \{0,1\}, \forall m \in \mathcal{M}, \forall n \in \mathcal{N}, \quad (6)$$

$$r_{\min} \le r_n \le r_{\max}, \forall n \in \mathcal{N}, \quad (7)$$

$$S_n = \{m \in \mathcal{M} : x_{mn} = 1\}, \forall n \in \mathcal{N}, \quad (8)$$

$$\sum_{n\in\mathcal{N}} \mathbf{1}\{|S_n| > 0\} \le \phi, \quad (9)$$

$$\frac{A}{B(d_{mn})} + \frac{|S_n|P}{Q^{\mathrm{E}}} \le \varphi, \forall m \in S_n, \forall n \in \mathcal{N}, \quad (10)$$

$$d_{mn} \le r_n, \forall m \in S_n, \forall n \in \mathcal{N}, \quad (11)$$

$$\sum_{n\in\mathcal{N}} x_{mn} \le 1, \forall m \in \mathcal{M}, \quad (12)$$

where r_n is the communication range of mobile edge server n, and $r_{\min}$ and $r_{\max}$ are the minimum and maximum communication range, respectively. S_n denotes the set of tasks assigned to edge server n, ϕ is the maximum number of mobile edge servers to be dispatched, and φ is the pre-defined task deadline. User requests change with time and space due to user mobility.

Constraint (6) shows that task m can either be assigned to server n ($x_{mn} = 1$) or not ($x_{mn} = 0$). Constraint (7) shows the limitation of the communication range. Constraint (8) denotes the set of tasks assigned to server $n, \forall n \in \mathcal{N}$. Constraint (9) shows that the total facility cost cannot exceed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

Algorithm 1 Online Mobile Edge Server Dispatching Scheme

for $t = 1, \dots T$ **do**
  for $n \in \mathcal{N}$ **do**
    Proactively synchronize UE state and location among UAVs.
    Invoke Algorithm 2 for server dispatching.
    Reach the destination and start off on its missions.
  end for
end for

the expected budget ϕ. The indicator function is equal to 1 if there exist tasks assigned to server $n, \forall n \in \mathcal{N}$ and 0 otherwise. Constraint (10) shows that the latency of the assigned tasks should be less than a given deadline φ. Otherwise, the work cannot be finished on time. Constraint (11) shows that the distance between the server and the assigned tasks should not exceed radius r_n. Constraint (12) shows that each task can be served by at most one edge server at the same time. Further, the UAV movement time is ignored in this work as commercial UAVs can reach 100 m/s at maximum and 30 m/s on average, which is much faster than pedestrians [21]. Compared with the static model in [17], UAVs can leverage their adjustable antenna angle to provide flexible coverage and move dynamically.

B. Online dispatching scheme

Bin-packing problems have been proved to be NP-hard. There exist well-known one-dimensional bin-packing algorithms that do not take geographic constraints of the items packed in the same bin into consideration. Take Next-Fit Decreasing as an example, items are reindexed in a non-increasing order according to their sizes. An item is packed to the current bin if the total size of the assigned items does not exceed the capacity of the bin. Otherwise, a new bin will be opened. However, the geographic constraint of the tasks assigned to the same edge server needs to be considered in the mobile edge server dispatching problem. Tasks covered by the same edge server need to be constrained by a circular area, i.e., the distance between every two tasks should not exceed the diameter of the communication coverage. On the other hand, the mobile edge server can adjust its coverage to meet the demands in either hot-spot or sparse areas. A longer distance between the task and the server results in a larger communication latency due to a lower data transmission rate.

Existing location-based clustering algorithms are not applicable as both the server capacity and communication range need to be jointly considered. To solve the Geo-VBP problem, an online dispatching scheme is proposed to determine the hover locations of the deployed UAVs. As shown in Algorithm 1, mobile edge servers proactively synchronize the UE information with other mobile servers and determine their potential hover locations. Considering the geographic constraint in Geo-VBP, a greedy algorithm is designed to merge hot-spot areas and assign tasks to the edge servers. After that, mobile edge servers reach their hover locations and start handling the offloaded tasks. Details of system implementation

Algorithm 2 UAV hover location decision making (HOLD)

Input: A set $\{(u_m, v_m, 0), \forall m \in \mathcal{M}\}$: location of each task, communication range $[r_{\min}, r_{\max}]$, radius increment Δr, and server budget ϕ.
Output: a set of tuples $\mathcal{R} = \{((u_n, v_n, h), r_n), n \in \mathcal{N}\}$: server locations (u_n, v_n, h) and the corresponding coverage radius r_n, and the number of dispatched servers H.
Initialization: $H \leftarrow 0$, $r \leftarrow r_{\min}$, $I \leftarrow \lfloor \frac{r_{\max} - r_{\min}}{\Delta r} \rfloor$, discretized grid J with radius $r_{\min}$ and center $\{(u_j, v_j, h), \forall j \in J\}$, task intensity $\{\varepsilon_j \leftarrow 0, \forall j \in J\}$, $\mathcal{R} \leftarrow \emptyset$, $\mathcal{M}' \leftarrow \mathcal{M}$, and $x_{mn} \leftarrow 0, \forall n \in \mathcal{N}, \forall m \in \mathcal{M}'$;

for $i = 1, \dots, I$ **do**
  Calculate $C(r) = \left\lfloor \frac{Q^{\mathrm{E}}\left(\varphi - \frac{A}{B(r)}\right)}{P} \right\rfloor$;
    ▷ The current service capacity
  Update task intensity $\{\varepsilon_j, \forall j \in J\}$ with r ;
    ▷ Hot-spot determination
  while $\max(\varepsilon_j) \geq C(r)$ **or** $i = I$ **do**
    $\mathcal{R} \leftarrow \mathcal{R} \bigcup \{(u_j, v_j, h), r)\}$;
      ▷ Server location and coverage radius
    Update $x_{mn} = 1$ for such $m \in \mathcal{M}'$ that m is one of the nearest $C(r)$ tasks of server $n, \forall n \in \mathcal{N}$;
      ▷ Task assignment
    $\mathcal{M}' \leftarrow \mathcal{M}' \backslash \{m \in \mathcal{M}' : x_{mn} = 1\}$;
      ▷ Delete assigned tasks
    $H \leftarrow H + 1$;
    if $H = \phi$ or $|\mathcal{M}'| < \theta \cdot C(r)$ **then**
      Return $\mathcal{R}, H$; ▷ Algorithm terminates
    end if
    Update task intensity $\{\varepsilon_j, \forall j \in J\}$ with r ;
  end while
  $r \leftarrow r + \Delta r$; ▷ Increase influence radius
end for

such as mobility management (handover for active mode or tracking for idle mode) and inter-UAV communication can be found in [11].

To identify and track the dynamic hot-spot areas, several approaches, e.g., machine learning and hot-spot monitoring with wireless sensors, have been investigated [22], [23]. However, they either require many computation resources or introduce high overhead. Moreover, the communication and computation constraints in existing algorithms are not taken into consideration. In the heat-map generation, tasks will be merged or clustered under different influence radius to graphically represent the task intensity. When the influence radius of the tasks increases, the task intensity of the overlapped areas adds up. That means that if a server is deployed in the overlapped area with the corresponding influence radius as the communication range, the tasks that influence the server location can be covered by the server. This is similar to the geographic constraint in the Geo-VBP problem.

As illustrated in Algorithm 2, inspired by the merging and generation of the heat map, the intuition of the proposed UAV hover location decision-making (HOLD) scheme is to merge the hot-spot areas, pack UEs sequentially and dispatch the mobile edge servers accordingly. The service area is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

discretized in J grids with a discretization radius $r_{\min}$. At each time slot, based on the Euclidean distance, the task intensity of grid $j \in J$ can be calculated as

$$\varepsilon_j = \sum_{m=1}^{M} \varepsilon_{mj}, \forall j \in J, \tag{13}$$

where $\varepsilon_{mj} \in \{0, 1\}$ denotes whether the distance between task m and the center of grid j is within radius r ($\varepsilon_{mj} = 1$) or not ($\varepsilon_{mj} = 0$). Once a grid is above the server capacity $C(r)$, which is defined as the maximum number of served tasks, a server can be deployed to the center of that grid, and meanwhile, the served tasks will be removed. Otherwise, r will be increased. Each time a server is dispatched or r changes, the heat map will be updated. As the last processed task of a server determines whether all assigned tasks can be finished on time, $C(r)$ is estimated as

$$C(r) = \left\lfloor \frac{Q^{\mathrm{E}}\left(\varphi - \frac{A}{B(r)}\right)}{P} \right\rfloor. \tag{14}$$

If r reaches the maximum communication range, no matter whether the intensity of the grids exceeds server capacity or not, servers will also be dispatched to the grids in the non-increasing order of task intensity. Repeat these steps until the number of servers reaches the server budget or all tasks have been served. To guarantee the server utilization, the algorithm will also terminate when the number of remaining tasks falls below a threshold $\theta \cdot C(r)$, as shown in Line 14. Let Δr denote the increment of radius r. After iterating

$$I = \left\lfloor \frac{r_{\max} - r_{\min}}{\Delta r} \right\rfloor \tag{15}$$

times, the hover locations of mobile servers are then determined one by one.

C. Theoretical analysis

In this section, the worst-case performance of the proposed algorithm is theoretically analyzed when compared with the optimal solution of (5) under the same spatial granularity.

Lemma 1. *In HOLD, H mobile edge servers are dispatched sequentially with the assigned tasks. Let $b_n := \sum_{m \in \mathcal{M}} x_{mn}$ denote the number of tasks assigned to server n. The number of served tasks is in the non-increasing order as n increases.*

Proof. Assume the earlier dispatched server is n_1 and the later one is n_2. Let b_1 and b_2 denote the number of assigned tasks in n_1 and n_2, respectively. Let C_1 and C_2 denote the capacity of n_1 and n_2, respectively. As the coverage radius of the dispatched servers gradually increases, Lemma 1 will be proved in the following two cases: 1) n_1 and n_2 have the same communication coverage, and 2) n_2 has a larger communication coverage than n_1.

If n_1 and n_2 cover the same-sized area, edge servers are dispatched in the decreasing order of task intensity. As n_1 is first dispatched to the hot-spot area with a higher intensity of the tasks, $b_2 \leq b_1$.

If n_2 covers a larger area than n_1, this means the radius of n_1 does not reach the largest communication range and b_1 must have reached the maximum number of served tasks C_1 in HOLD. In (14), as the capacity of a server decreases with the increase of communication range, $b_2 \leq C_2 \leq C_1 = b_1$. □

Let $OPT(\mathcal{M})$ and $ALG(\mathcal{M})$ represent the number of served tasks in the optimal algorithm (OPT) and HOLD under the same spatial granularity, respectively. Thus $OPT(\mathcal{M}) = \chi \leq |\mathcal{M}|$. For $\forall \mathcal{M}$, the asymptotic worst-case approximation ratio R_{HOLD} of HOLD is [24]

$$R_{HOLD} = \lim_{\chi \to \infty} \inf \left(\min \left\{ \frac{ALG(\mathcal{M})}{\chi} : OPT(\mathcal{M}) = \chi \right\} \right). \tag{16}$$

Thus, $R_{HOLD} \geq \max(\rho)$ if there exist two constants ρ and σ such that [25]

$$ALG(\mathcal{M}) \geq \rho \cdot OPT(\mathcal{M}) + \sigma. \tag{17}$$

Since HOLD terminates in two cases as shown in Line 14: $|\mathcal{M}'| < \theta \cdot C(r)$ (case 1) and $H = \phi$ (case 2). Therefore, we prepare Lemma 2 to analyze the case that HOLD terminates because of case 1 and Lemma 3 because of case 2. For ease of presentation, we say a server n is saturated if $b_n = C_n$. Otherwise, we say the server is unsaturated.

Lemma 2. *If HOLD terminates due to the condition of $|\mathcal{M}'| < \theta \cdot C(r)$, $R_{HOLD} = \frac{k}{k+\theta}$, where $k \geq 1$ is the number of saturated servers.*

Proof. Recall that H is the number of dispatched servers output by HOLD. From $|\mathcal{M}'| < \theta \cdot C(r)$, we know either $\mathcal{M}' = \emptyset$ or $0 < |\mathcal{M}'| < \theta \cdot C(r)$.

If all tasks have been served by HOLD, i.e., $\mathcal{M}' = \emptyset$, it is trivial to prove as we have $ALG = OPT = |\mathcal{M}|$. Clearly, $OPT \leq (1 + \frac{\theta}{k}) \cdot ALG$.

If not all tasks can be served by the dispatched servers, $|\mathcal{M}'| < \theta \cdot C_H$. From Lemma 1, we know that the number of tasks served by the dispatched saturated edge servers is in a non-increasing order, which means $ALG \geq b_1 + b_2 + ... + b_k \geq k \cdot b_k$. Therefore, we have $|\mathcal{M}'| < \theta \cdot C_H \leq \theta \cdot C_k = \theta \cdot b_k \leq \frac{\theta}{k} \cdot ALG$.

As the optimal solution can serve at most $|\mathcal{M}|$ tasks, thus we have

$$\begin{aligned} OPT &\leq |\mathcal{M}| \\ &= \sum_{i=1}^{H} b_i + |\mathcal{M}'| \\ &= ALG + |\mathcal{M}'| \\ &\leq ALG + \frac{\theta}{k} \cdot ALG \\ &= (1 + \frac{\theta}{k}) \cdot ALG, \end{aligned} \tag{18}$$

which concludes the proof.

□

Lemma 3. *If HOLD terminates due to the condition of $H = \phi$, $R_{HOLD} = 1 - e^{-1}$.*

Proof. Recall that b_n denotes the number of served tasks of server n. Let d_n denote the difference between ALG and OPT at the n-th dispatched server, i.e., $d_n = OPT - \sum_{j=1}^{n} b_j$,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

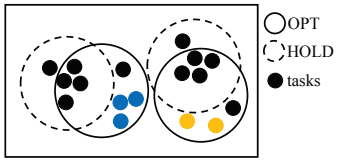


Fig. 3: Illustration of Lemma 3.

where $\sum_{j=1}^{n} b_j$ is the number of served tasks up to the n-th server. Thus, $d_0 = OPT$.

We assume that OPT tasks are served by $1 \leq K \leq \phi$ servers in the optimal solution. In the generalized pigeonhole principle, if n pigeons are placed into $m \leq n$ pigeonholes, there exists at least one pigeonhole contains at least $\frac{n}{m}$ pigeons. As d_n tasks are covered by the K subsets in OPT, by the pigeonhole principle, one of the K subsets in OPT must serve at least $\lceil d_n/K \rceil$ tasks [26]. As HOLD greedily chooses the set covering the maximum number of unserved tasks,

$$b_{n+1} \geq d_n/K. \tag{19}$$

As illustrated in Fig. 3, let us assume OPT has covered $OPT = 15$ tasks with $K = 2$ servers and HOLD has covered $\sum_{j=1}^{2} b_j = 10$ tasks up to the 2-nd iteration. There exist at least $d_2 = 5$ tasks (in blue and yellow) which have been covered by OPT but not HOLD. Within these 5 tasks, one of the servers in OPT needs to cover at least $\lceil d_2/K \rceil = 3$ tasks. As the next server dispatched by HOLD covers the most intensive uncovered area which consists of at least 3 tasks, $b_3 \geq 3 \geq d_2/K$.

Thus, $b_1 > b_{n+1} \geq d_0/K = OPT \cdot \frac{1}{K}$. We further have

$$\begin{aligned} d_1 &= OPT - b_1 \\ &\leq OPT - OPT \cdot \frac{1}{K} \\ &= OPT \cdot (1 - \frac{1}{K}). \end{aligned} \tag{20}$$

By the definition of d_n and (19), we have

$$\begin{aligned} d_n &= d_{n-1} - b_n \\ &\leq d_{n-1} - d_{n-1} \cdot \frac{1}{K} \\ &= d_{n-1} \cdot (1 - \frac{1}{K}). \end{aligned} \tag{21}$$

By induction hypothesis, we have

$$d_n \leq OPT \cdot (1 - \frac{1}{K})^n. \tag{22}$$

Since $(1 - \frac{1}{K})^K \leq \frac{1}{e}$,

$$\begin{aligned} ALG &= OPT - d_\phi \\ &\geq OPT - d_K \\ &\geq OPT - OPT \cdot (1 - \frac{1}{K})^K \\ &\geq OPT - OPT \cdot \frac{1}{e} \\ &\geq OPT \cdot (1 - \frac{1}{e}), \end{aligned} \tag{23}$$

which concludes the proof. □

VI. Performance Evaluation

In this section, extensive simulations are conducted to evaluate the proposed mobile dispatching scheme. As shown in Fig. 2, according to the deployment in the real world, 7 fixed BSs are distributed in and around the theme park. The number of available mobile edge servers is set as $\phi = 7$ to be comparable to the number of fixed servers at BSs. For fair performance comparison purposes, we assume all servers have the same computation capacity by default. B_0, N_0 and the reference Signal-to-Noise ratio at the reference distance are 1 MHz, -60 dBm and -30 dB, respectively [27]. The data size and the computation intensity of the offloaded task are set according to the face recognition application, i.e., $A = 60$ kB and $P = 31,680$ cycles/bit [28]. The total computation capacity of each edge server is set as $Q^E = 10^{10}$ cycles/s with 10 VMs deployed [29]. The task deadline is set to 10 s. θ, Δr, $r_{\min}$ and $r_{\max}$ are set 1, 90 m, 100 m and 1,000 m [30].

Performance metrics are: 1) service capacity, i.e., the number of served tasks with a given task deadline; 2) service fairness in terms of latency defined in (24); 3) the number of UAVs needed; and 4) server utilization, which measures whether VMs in each server are fully utilized or not. Jain's fairness index is defined as [31]

$$\text{Fairness}(\mathbf{D}) = \frac{(\sum_{i=1}^{M} D_i)^2}{M \sum_{i=1}^{M} D_i^2} = \frac{\overline{\mathbf{D}}^2}{\mathbf{D}^2}, \tag{24}$$

where $\mathbf{D} = \{D_1, D_2, ..., D_M\}$ is the latency of all tasks. The fairness index reaches the maximum when all UEs experience the same latency, i.e., $\text{Fairness}(\mathbf{D}) = 1$.

For a fair comparison, the following server placement schemes are introduced:

- **Fixed deployment (Fixed only):** The fixed BSs are equipped with edge servers. Tasks are assigned to the BSs based on the strongest RSS.
- **Mobile dispatching (Mobile only):** Only UAV-mounted mobile edge servers are deployed to complete the offloaded tasks. Based on the proposed mobile dispatching scheme, mobile edge servers are dispatched to the hotspot areas and then serve the crowds.
- **Hybrid dispatching (Hybrid):** Both the fixed and mobile edge servers are participating in task offloading. A simple hybrid scheme is considered here where mobile edge servers assist to offload the tasks for the UEs that cannot be served by the fixed servers. **Hybrid-Mobile** refers to the performance of mobile edge servers in the hybrid dispatching scheme.
- **Spiral [32]:** UAVs are placed sequentially like a spiral. The algorithm starts from a random-selected UE at the boundary and places UAVs counterclockwise with the coverage radius 200 m. For each iteration, the UAV first aims to cover the boundary UEs as much as possible and then refines its location to cover inner UEs as much as possible by solving the 1-center facility location problem [33]. Due to the randomness of Spiral, we take the average value over 100 runs of the simulation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

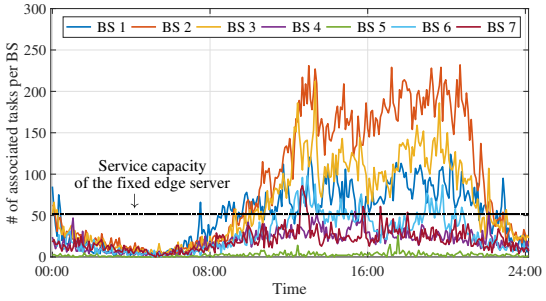


Fig. 4: The number of associated tasks for each fixed BS.

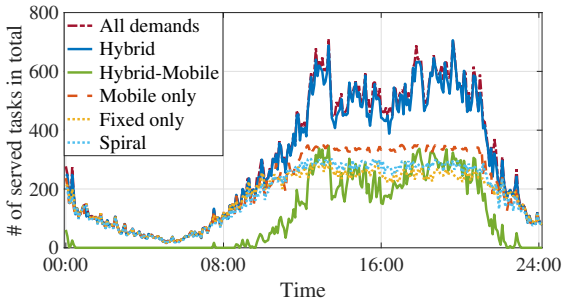


Fig. 5: Impact on service capacity.

A. Dynamic task requests and unbalanced workload

The fixed deployment of edge servers is first evaluated. The ID of each BS has been labeled in Fig. 2. Fig. 4 shows the number of tasks assigned to the corresponding BS. To guarantee that the tasks are completed before the given task deadline, according to the default setup, each edge server can handle around 50 tasks at the same time at maximum. In general, the number of associated tasks shows a sharp difference between the daytime and night. After 2 AM, there are fewer than 20 tasks per BS in the park. The number remains relatively stable until 7 AM. After that, the number of tasks increases considerably in the morning and reaches a peak at 1 PM. Over the 24 hours, BSs near the park entrance, i.e., BS 1, BS 2 and BS 3, have a large number of tasks to deal with while BS 4, BS 5 and BS 7 have far fewer tasks assigned to them. Dynamic task requests and the nonuniform distribution of the crowds result in the unbalanced workload among the fixed BSs. Meanwhile, tasks will not be assigned to the BSs far away from them, considering the communication range and cost. This motivates us to dispatch mobile edge servers closer to the hot-spot areas to handle the dynamic demands.

B. Number of served tasks

Fig. 5 shows the total number of served tasks with the given task deadline, which is constrained by both the communication and the computation cost. At night, the number of tasks is much smaller than the maximum service capacity of the edge servers. All four schemes can satisfy all UE demands. However, during the day time, the tasks increase dramatically. Due to the flexible placement, mobile edge servers are dispatched closer to the crowds. With the same computation capacities, mobile dispatching can serve 59% and 31% more tasks during peak hours when compared with the fixed deployment and Spiral, respectively. The fixed deployment ignores the dynamic UE distribution and Spiral overlooks the hot-spot areas. Thus, the performance of the aforementioned schemes sacrifices during the daytime. In the hybrid scheme, mobile edge servers assist the fixed servers and help serve more tasks. On average, 96.2% tasks can be finished on time in the hybrid scheme, indicating the need to increase the server computation capacity or number of UAVs beyond the default setup as shown in Section VI-G. Moreover, the fluctuation of the number of served tasks can reflect network stability under different algorithms. When the demand exceeds supply, the standard deviation of the mobile dispatching scheme, Spiral, and the fixed deployment is 5.8, 13.8, and 16.1, respectively. No matter how the number of tasks changes and how UEs are distributed due to their mobility, the number of served tasks remain stable with a higher value. This indicates the robustness of the proposed scheme for dynamic networks.

C. Number of UAVs dispatched

Fig. 6 shows the number of mobile edge servers dispatched over 24 hours in the mobile dispatching scheme, the hybrid scheme, and Spiral. Generally speaking, the number of dispatched UAVs depends on real-time demands. Intuitively, more UAVs will be dispatched to serve the crowds with higher density. During the day time, UAVs are dispatched to support the hot-spot areas. At night, more UAVs are dispatched by Spiral as UEs are sparsely distributed in the theme park and Spiral starts from randomly selected UEs at the boundary. In contrast, only 1 or 2 UAVs are employed in the mobile dispatching scheme. Moreover, the hybrid scheme shows its ability to deal with varying tasks with fewer UAVs. As the fixed BSs can handle all the tasks at night, there is no need to dispatch extra UAVs after midnight until early morning when crowds form again. This means the hybrid scheme can leverage the benefits of existing fixed BSs, which, in turn, not only exploits the flexibility of UAVs but also saves power and cuts down the total UAV maintenance cost.

D. Utilization of the computation resources

As mentioned above, the fixed BSs can complete all the tasks quickly at night and there is no need to dispatch extra UAVs. Thus, the latency variation in the mobile dispatching scheme can be larger than that in the fixed deployment, as all demands queue up for processing with fewer mobile servers. This, however, increases server utilization at night. Therefore, the server utilization and the service fairness in terms of user-experienced latency are evaluated during the daytime with bursty tasks from 10 AM to 6 PM for a fair comparison between schemes.

Fig. 7 illustrates the CDF of server utilization during the daytime. At each time slot, a fairness index of the whole system is calculated based on (24) and then the CDF over all time slots is obtained. On average, the server utilization in the mobile dispatching scheme and the hybrid scheme is 98% and 93.8%, respectively. In the fixed deployment, the server utilization is 90.5%, which indicates one out of the ten VMs at each fixed server is idle. In the mobile dispatching scheme, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

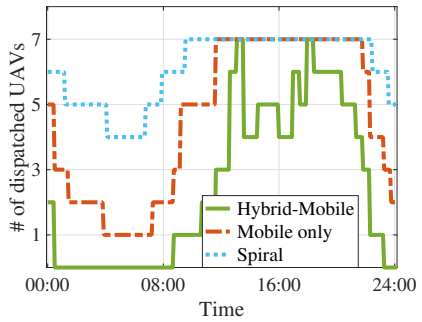


Fig. 6: Impact on the dispatched UAVs.

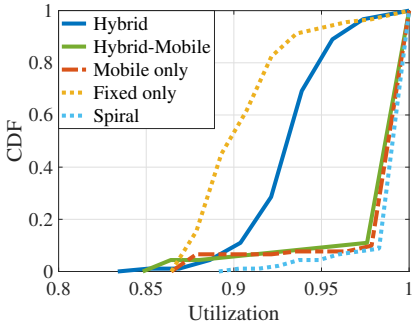


Fig. 7: Impact on latency fairness index.

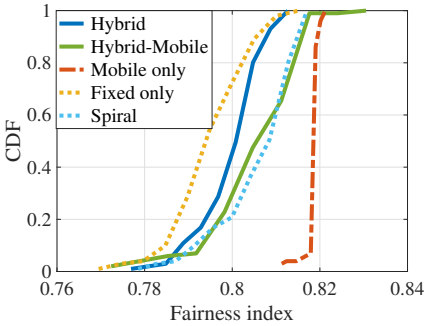


Fig. 8: Impact on server utilization.

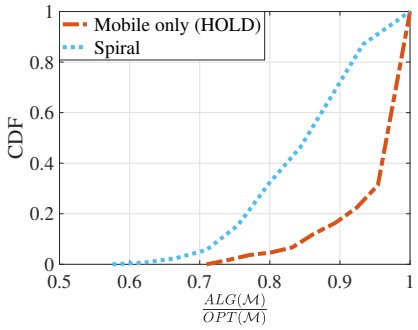


Fig. 9: Practical performance of different algorithms when compared with OPT.

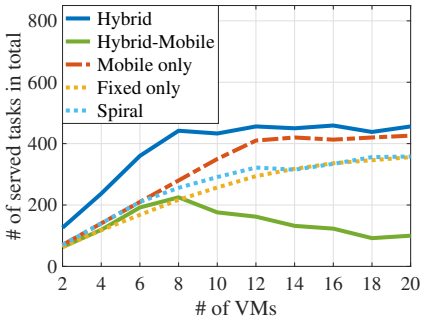


Fig. 10: Impact of server capacity on service capacity.

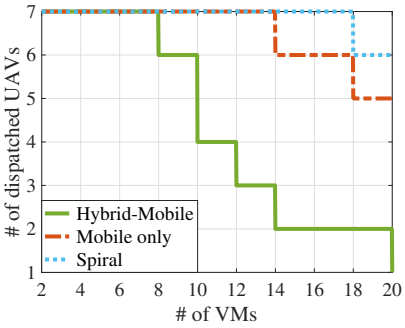


Fig. 11: Impact of server capacity on the number of dispatched UAVs.

probability of the server utilization higher than 90% increases by 80% when compared with the fixed deployment. Even faced with a large number of tasks, the fixed edge servers still cannot be fully utilized. The reason is that some tasks in the hot-spot areas either are refused by the overloaded BS or do not have the opportunity to connect to the idle BSs that are far away from them. Mobile edge servers overcome this limitation by dispatching UAVs in close proximity to UEs. Spiral also performs well during the daytime. Because Spiral and the mobile dispatching scheme both act in a best-effort manner by serving UEs as much as possible. But the difference is that the mobile dispatching scheme focuses on hot-spot areas with adjustable coverage radius while Spiral starts from the boundary with a constant radius.

E. Service fairness in terms of latency

Fig. 8 shows the cumulative distribution function (CDF) of the service fairness index with the four schemes. The trend of service fairness over time is similar to that of utilization except for Spiral. In the fixed deployment, the service fairness deteriorates with bursty requests in certain areas, e.g., a huge crowd gathers at the entrance in the morning. This results in the overloaded BSs near the entrance and underloaded BSs along the outside road, thus leading to a high variation among task latencies. Similarly, Spiral places UAVs from the boundary to center and thus performs a bit better than the fixed deployment. The steep slope of the mobile dispatching scheme indicates that the service fairness concentrates on 0.82 over all time slots. This not only implies network stability under dynamic UE distributions but also shows that the mobile dispatching scheme outperforms other schemes by maintaining a better service fairness. The hybrid scheme increases service fairness when compared with the fixed deployment by introducing mobile edge servers. It can be seen that 60% of the time, service fairness is higher than 0.8 while 25% of the time in the fixed deployment. When compared with the mobile dispatching scheme where UAVs are determined to serve the hot-spot areas first, mobile servers in the hybrid scheme only assist the UEs that cannot be served by the fixed BSs on time. Thus, its performance is affected by the number of its served tasks and the corresponding distance between the tasks and the servers.

F. Practical performance of HOLD

Fig. 9 shows the practical performance of the proposed dispatching algorithm HOLD and Spiral when compared with the optimal solution OPT. OPT is obtained by solving (5) through the optimization tool Gurobi 8.1.1. The x-axis denotes the proportion of the number of tasks served by each algorithm and OPT, and the y-axis shows the CDF of the proportion over the time slots. In more than 80% of the time slots, the number of tasks served by HOLD can reach as many as 90% of OPT, which is bounded by $1 - e^{-1}$. However, Spiral can only achieve the same situation with 30% of the time slots. A larger gap between HOLD and OPT is caused by the algorithm termination condition, i.e., the dispatching algorithm stops when the number of the remaining tasks falls below a threshold. In the circumstance where the remaining tasks are sparsely distributed, dispatching extra mobile servers results in the deterioration of the server utilization.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2954798, IEEE Internet of Things Journal

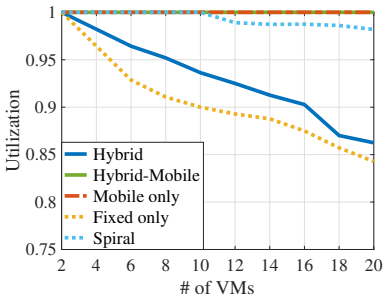


Fig. 12: Impact of server capacity on service fairness.

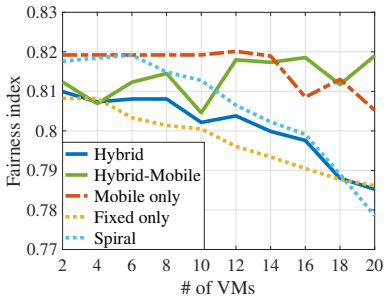


Fig. 13: Impact of server capacity on server utilization.

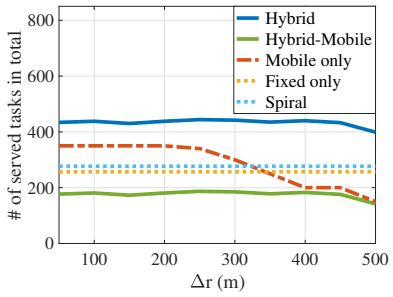


Fig. 14: Impact of radius step Δr on service capacity.

G. Impact of parameters

Based on the analysis of the limitations mentioned above, the impact of computation capacities, i.e., the number of VMs deployed in each edge server, and radius increment Δr, is further investigated. We select a specific time slot, 1 PM, to illustrate the impact of the parameters.

Fig. 10 shows the impact of the number of VMs per edge server on the service capacity. Here each VM has the same computation capacity as the default setup. It can be seen that the number of served tasks in the fixed deployment and Spiral continuously increases. This is because when the computation capacity increases, the computation latency of UEs reduces sharply with lower execution latency and queuing latency. Thus, with more VMs available, tasks can be executed as soon as possible and more tasks can be finished before the given deadline. In the hybrid scheme, the number of served tasks triples when the number of VMs per edge server increases from 2 to 6. 8 VMs per server can meet all the UE demands in the hybrid scheme while 14 VMs in mobile dispatching. After that, fewer UAVs are needed because of the increased computation capacity of the fixed BSs. Thus, in the hybrid scheme, the number of tasks served by mobile edge servers decreases.

Fig. 11 shows the impact of VMs on the number of dispatched UAVs needed to meet all demands. Equipped more computation capacities, each edge server can complete more tasks with the same task deadline. This results in fewer mobile edge servers to be dispatched for all demands. The number of dispatched UAVs first decreases with 8 VMs, 14 VMs and 18 VMs in the hybrid scheme, the mobile dispatching scheme and Spiral, respectively. This, in turn, shows that the number of served tasks per mobile edge server increases.

Fig. 12 and Fig. 13 show the impact of the number of VMs per edge server on server utilization and service fairness. Overall, the fairness index decreases with the increase of VMs. In the fixed deployment and Spiral, more tasks in the hot-spot areas queue up to be processed. This leads to a larger latency variation. In the mobile dispatching scheme, the service fairness jitters around 16 VMs per edge server. The reason is that with the same number of dispatched UAVs, higher computation capacities reduce the per task latency, thus leading to a slight increase in service fairness. The same reason applies to the hybrid scheme. In addition, servers are fully utilized when the demand exceeds the supply or with fewer UAVs to just meet the demands. That is, the server utilization reduces with extra VMs installed in the fixed BSs while the mobile dispatching scheme can fully utilize the resources by adjusting the number of dispatched UAVs.

Fig. 14 shows the impact of radius increment Δr on the number of served tasks. Δr affects the scale of identifying the hot-spot areas. A small Δr makes the mobile dispatching scheme more accurate, which, however, increases the computation complexity. A large Δr may fail to determine the most appropriate location of the edge server, thus leading to fewer served tasks. The fixed deployment and Spiral do not rely on Δr and thus are used as the baselines. With the increase of Δr, the performances of both the mobile dispatching and the hybrid scheme decrease. A large radius increment makes the mobile edge servers be located at the center of a large hot-spot area, which, however, exceed their computation capacities. This leads to the problems that the servers miss the smaller hot-spot areas inside and thus need to pay a larger communication cost.

VII. Conclusions and Future Work

In this paper, an online UAV-mounted mobile edge server dispatching scheme is proposed for mobile-to-mobile edge computing scenarios. The mobility of both UEs and edge servers are jointly considered to tackle the issues of nonuniformly distributed tasks and dynamic crowds. UAVs iteratively find their appropriate hover locations to serve the crowds. Simulations show that either the mobile dispatching alone or the hybrid dispatching scheme is superior to the fixed deployment. In the mobile dispatching scheme, the number of served tasks increases by 59% on average and the server utilization can reach 98% with the same computation capabilities. For future work, UAVs can act as a relay to fully utilize the computation resources of the fixed edge servers by interacting with the ground fixed BSs. Moreover, inter-UAV interference and collision avoidance need to be further addressed by considering the constraints of the minimum distance among UAVs if they have to use the same channel.

References

[1] P. Hao, L. Hu, J. Jiang, J. Hu, and X. Che, "Mobile edge provision with flexible deployment," *IEEE Trans. Serv. Comput.*, 2018.

[2] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *IEEE IFIP Networking*, 2015, pp. 1–9.

[3] X. Sun and N. Ansari, "PRIMAL: Profit maximization avatar placement for mobile edge computing," in *IEEE ICC*, 2016, pp. 1–6.

[4] C. Li, L. Toni, J. Zou, H. Xiong, and P. Frossard, "QoE-driven mobile edge caching placement for adaptive video streaming," *IEEE Trans. Multimedia*, vol. 20, no. 4, pp. 965–984, 2018.

[5] H. Yin, X. Zhang, H. H. Liu, Y. Luo, C. Tian, S. Zhao, and F. Li, "Edge provisioning with flexible server placement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 4, pp. 1031–1045, 2017.

[6] Y. Cao, C. Long, T. Jiang, and S. Mao, "Share communication and computation resources on mobile devices: A social awareness perspective," *IEEE Wirel. Commun.*, vol. 23, no. 4, pp. 52–59, 2016.

[7] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Trans. Wirel. Commun.*, vol. 17, no. 3, pp. 2109–2121, 2018.

[8] N. Cheng, W. Xu, W. Shi, Y. Zhou, N. Lu, H. Zhou, and X. Shen, "Air-ground integrated mobile edge networks: Architecture, challenges, and opportunities," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 26–32, 2018.

[9] P. Oettershagen, A. Melzer, T. Mantel, K. Rudin, T. Stastny, B. Wawrzacz, T. Hinzmann, K. Alexis, and R. Siegwart, "Perpetual flight with a small solar-powered UAV: Flight results, performance analysis and model validation," in *IEEE AeroConf*, 2016, pp. 1–8.

[10] C. Papachristos and A. Tzes, "The power-tethered uav-ugv team: A collaborative strategy for navigation in partially-mapped environments," in *IEEE MCCA*, 2014, pp. 1153–1158.

[11] M. Moradi, K. Sundaresan, E. Chai, S. Rangarajan, and Z. M. Mao, "SkyCore: Moving core to the edge for untethered and reliable UAV-based LTE networks," in *ACM MobiCom*, 2018, pp. 35–49.

[12] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2049–2063, 2018.

[13] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, 2018.

[14] F. Cheng, S. Zhang, Z. Li, Y. Chen, N. Zhao, F. R. Yu, and V. C. M. Leung, "UAV trajectory optimization for data offloading at the edge of multiple cells," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6732–6736, 2018.

[15] D. K. Friesen and M. A. Langston, "Variable sized bin packing," *SIAM J. Comput.*, vol. 15, no. 1, pp. 222–230, 1986.

[16] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *J. Parallel Distrib. Comput.*, 2018.

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

[18] A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1818–1831, 2017.

[19] "Mobile Tencent Analytics," https://data.qq.com/article?id=3479.

[20] A. A. Khuwaja, Y. Chen, N. Zhao, M.-S. Alouini, and P. Dobbins, "A survey of channel modeling for UAV communications," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 4, pp. 2804–2821, 2018.

[21] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 36–42, 2016.

[22] T. Teixeira, G. Dublon, and A. Savvides, "A survey of human-sensing: Methods for detecting presence, count, location, track, and identity," *ACM Comput. Surv.*, vol. 5, no. 1, pp. 59–69, 2010.

[23] Y. Yu, G. Lin, I. H. Jiang, and C. Chiang, "Machine-learning-based hotspot detection using topological classification and critical feature extraction," *IEEE Trans. Comput-Aided Des. Integr. Circuits Syst.*, vol. 34, no. 3, pp. 460–470, 2015.

[24] E. G. Coffman Jr, J. Csirik, G. Galambos, S. Martello, and D. Vigo, "Bin packing approximation algorithms: Survey and classification," *Handbook of combinatorial optimization*, pp. 455–531, 2013.

[25] J. Csirik, "Two simple algorithms for bin covering," *Acta Cybernetica*, vol. 14, no. 1, pp. 13–25, 1999.

[26] D. S. Hochbaum and A. Pathria, "Analysis of the greedy approach in problems of maximum K-coverage," *Nav. Res. Logist.*, vol. 45, no. 6, pp. 615–627, 1998.

[27] X. Cao, J. Xu, and R. Zhang, "Mobile edge computing for cellular-connected UAV: Computation offloading and trajectory optimization," in *IEEE SPAWC*, 2018, pp. 1–5.

[28] K. Liu, J. Peng, H. Li, X. Zhang, and W. Liu, "Multi-device task offloading with time-constraints for energy efficiency in mobile cloud computing," *Futur. Gener. Comp. Syst.*, vol. 64, pp. 1–14, 2016.

[29] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, 2016.

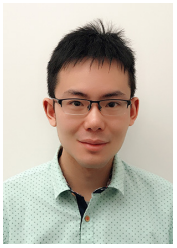
[30] Z. Xiao, P. Xia, and X.-G. Xia, "Enabling UAV cellular with millimeter-wave communication: Potentials and approaches," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 66–73, 2016.

[31] R. K. Jain, D. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination," *Eastern Research Laboratory, Digital Equipment Corporation*, pp. 2–7, 1984.

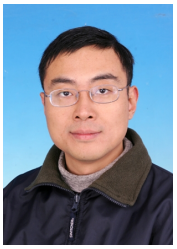
[32] J. Lyu, Y. Zeng, R. Zhang, and T. J. Lim, "Placement optimization of UAV-mounted mobile base stations," *IEEE Commun. Lett.*, vol. 21, no. 3, pp. 604–607, 2016.

[33] J. Elzinga and D. W. Hearn, "Geometrical solutions for some minimax location problems," *Transp. Sci.*, vol. 6, no. 4, pp. 379–394, 1972.

Jingrong Wang (S'18) is currently a Ph.D. student at the University of Toronto. She received her Bachelor's degree in the School of Electronic and Information Engineering from Beijing Jiaotong University in 2017, and M.Sc degree in Computer Science from the University of Victoria in 2019. Her research interests cover wireless communications, mobile edge computing, and machine learning.



Kaiyang Liu (S'14–M'19) received his Ph.D. degree in the School of Information Science and Engineering, Central South University in 2019. During 2016–2018, he was a research assistant at the University of Victoria, Canada, with Prof. Jianping Pan. His current research areas include networked systems, distributed systems and cloud/edge computing, with a special focus on the analysis and optimization of the data-intensive services. One of his papers is one of the three IEEE LCN 2018 Best Paper Award candidates.



Jianping Pan (S'96–M'98–SM'08) is currently a professor of computer science at the University of Victoria, Victoria, British Columbia, Canada. He received his Bachelor's and PhD degrees in computer science from Southeast University, Nanjing, Jiangsu, China, and he did his postdoctoral research at the University of Waterloo, Waterloo, Ontario, Canada. He also worked at Fujitsu Labs and NTT Labs. His area of specialization is computer networks and distributed systems, and his current research interests include protocols for advanced networking, performance analysis of networked systems, and applied network security. He received the IEICE Best Paper Award in 2009, the Telecommunications Advancement Foundation's Telesys Award in 2010, the WCSP 2011 Best Paper Award, the IEEE Globecom 2011 Best Paper Award, the JSPS Invitation Fellowship in 2012, the IEEE ICC 2013 Best Paper Award, and the NSERC DAS Award in 2016, and is a coauthor of one of the three IEEE LCN 2018 Best Paper Award candidates, and has been serving on the technical program committees of major computer communications and networking conferences including IEEE INFOCOM, ICC, Globecom, WCNC and CCNC. He was the Ad Hoc and Sensor Networking Symposium Co-Chair of IEEE Globecom 2012 and an Associate Editor of IEEE Transactions on Vehicular Technology. He is a senior member of the ACM.