Service Home Identification of Multiple-Source IoT Applications in Edge Computing

Jing Li, Weifa Liang, *Senior Member, IEEE*, Wenzheng Xu, *Member, IEEE*, Zichuan Xu, *Member, IEEE*, Yuchen Li, and Xiaohua Jia, *Fellow, IEEE*

Abstract—The real-time communication requirement of the Internet of Things (IoT) applications promotes the convergence of IoT and Mobile Edge Computing (MEC). The MEC paradigm greatly shortens the IoT service delay by leveraging cloudlets (edge servers) of MEC in the proximity of IoT devices. Considering limited computing and storage resources in an MEC network, it is challenging to provide efficient IoT-enabled service provisioning in such a network. In this article, we study the service home identification problem of service provisioning for multi-source IoT applications in an MEC network, by identifying a service home (cloudlet) of each multi-source IoT application for its data processing, querying and storage. Each multi-source IoT application consists of multiple sources located at different geographical locations and each source uploads its data stream via a gateway (its nearby access point) to the MEC network and the uploaded data then is aggregated with the stream data of the other sources of the IoT application at the service home. We here focus on two novel service home identification problems: the service operational cost minimization problem with the aim to minimize the total service operational cost by admitting as many multi-source IoT applications as possible, and the online throughput maximization problem with the aim to maximize the number of multi-source IoT application requests admitted. We first show that both the problems are NP-hard. We then formulate an Integer Linear Programming (ILP) solution to the service operational cost minimization problem, and propose a randomized algorithm with high probability and a deterministic approximation algorithm respectively, at moderate resource capacity violations. We third develop an efficient heuristic algorithm for the problem without any resource violation. Furthermore, we deal with the online throughput maximization problem under an assumption that multi-source IoT application requests arrive one by one without the knowledge of future arrivals, for which we formulate an Integer Linear Programming (ILP) solution to its offline version, followed by devising an online algorithm with competitive ratio. We finally evaluate the performance of the proposed algorithms through experimental simulations. Simulation results demonstrate that the proposed algorithms are promising, and outperform their comparison counterparts.

Index Terms—Internet of Things (IoT), mobile edge computing (MEC), service provisioning for multi-source IoT applications, service operational cost minimization, randomized algorithm, online algorithm, resource allocation and optimization, dynamic IoT application request admissions, network slicing in edge computing

1 Introduction

The emergence of the Internet of Things (IoT) drives unprecedented opportunities for the evolution of connection and communication towards the next era of the networking ecosystem [9]. In the paradigm of IoT, human beings, machines and services are methodically interconnected via intelligent IoT devices and platforms to eventually realize the 'smart world' [10]. Moreover, IoT devices facilitate the propagation and exchange of massive data generated from the virtual world and the real world across geographical areas to achieve business purposes or personal benefits [2]. The number of interconnected IoT devices has been anticipated to reach 500 billion by 2030, in accordance with the report from Cisco [6]. The latency brought by unstable wireless communications and computation failures caused by limited resource on IoT devices however prevents users from experiencing high efficiency and seamless user experience. To address these shortcomings, Mobile Edge Computing (MEC) is a promising technology to enable delay-sensitive service provisioning for IoT applications, where cloudlets (edge servers) are co-located with the access points (APs) [13]. By providing the network service in the proximity of IoT devices, the IoT devices can upload their collected sensory data to the MEC network for processing with short service latency [1], [23]. The data collected from different sources (sensors) usually have distinct structures, including images, audio and videos [4]. Therefore, an IoT application is supposed to manipulate a growing number of IoT devices and integrate their uploaded data streams to improve the processing performance. To this end, multiple data streams from different sources of a multi-source IoT application are collected and aggregated at a service home (cloudlet) in an MEC network

- *Jing Li and Yuchen Li are with the School of Computing, Australian National University, Canberra, ACT 2601, Australia. E-mail: {jing.li5, yuchen.li}@anu.edu.au.*
- *Weifa Liang and Xiaohua Jia are with the Department of Computer Science, City University of Hong Kong, Hong Kong, China. E-mail: {weifa.liang, csjia}@cityu.edu.hk.*
- *Wenzheng Xu is with the College of Computer Science, Sichuan University, Chengdu 610017, China. E-mail: wenzheng.xu@scu.edu.cn.*
- *Zichuan Xu is with the School of Software, Dalian University of Technology, Dalian 116024, China. E-mail: z.xu@dlut.edu.cn.*

Manuscript received 9 February 2022; revised 16 May 2022; accepted 17 May 2022. Date of publication 24 May 2022; date of current version 10 April 2023.
The work of Weifa Liang was supported by the City University of Hong Kong under Grant 9380137/CS. The work of Wenzheng Xu was supported by NSFC under Grant 61602330. The work of Zichuan Xu was supported in part by the National Natural Science Foundation of China (NSFC) under Grants 62172068, 62172071, 61802048, and 61802047, and in part by Xinghai Scholar Program.
(Corresponding author: Weifa Liang.)
Digital Object Identifier no. 10.1109/TSC.2022.3176576

for further processing and storage. The processed data then can be transferred to a remote cloud for permanent storage and big data analytic purposes.

In this paper we consider a set of multi-source IoT applications with each consisting of multiple sources located at different geographical locations. In the real world, different types of IoT sensory devices are deployed to monitor different geographical areas (e.g., public park safety monitoring) in a metropolitan area. The data streams generated from IoT devices of different parks can then be aggregated within their subnetworks and uploaded to the MEC through their gateways (APs). Each source is a sink of a sensor subnetwork and the collected stream data from the subnetwork will be continuously uploaded to a *service home* - a cloudlet for further processing, querying and storage. Both computing and communication costs incurred by accommodating a multi-source IoT application at the MEC network are part of the service operational cost of such an application. Enabling effective service provisioning in an MEC network for multi-source IoT applications poses several challenges: One challenge is how to identify a service home for each multi-source IoT application such that the overall data transfer delay from different sources to its service home is minimized, thereby reducing the overall bandwidth consumption and communication cost of the application. On the other hand, considering that the computing capacity on each cloudlet in an MEC network is limited, different multi-source IoT applications compete limited cloudlet resources with each other for their service homes. How to allocate the limited cloudlet resources for different multi-source IoT application requests to host their service homes while minimizing the total service operational cost is another challenge. Finally, the service provisioning for each of the multi-source IoT applications usually is expected to be run in a long term, thus both computing and bandwidth resources allocated to the IoT application will last for a foreseeable future. In other words, the MEC network can be sliced into multiple virtual networks with each providing services to a multi-source IoT application. Then, how to perform efficient network slicing so that the profit of the network service provider can be maximized is the last challenge. In the rest of this paper, we will address the challenges and tackle service home identification problems.

The novelty of this paper lies in considering the home cloudlet identification for each multi-source IoT application, where each source of the IoT application can upload its stream data to the MEC through the AP under which it is covered, and the data from different sources then are aggregated at a cloudlet (the home of the IoT application) for further processing, or the location of the data will be transferred to the remote cloud for permanent storage or further analysis. Two novel service home identification problems in an MEC network are formulated for such multi-source IoT applications, and performance-guaranteed approximation and online algorithms are proposed.

The main contributions of this paper are presented as follows.

- We formulate two novel service home identification problems for multi-source IoT applications in an MEC network, by identifying a service home for each multi-source IoT application: the service operational cost minimization problem, and the online throughput maximization problem. We show that both the problems are NP-hard and inapproximate.
- We formulate an Integer Linear Programming (ILP) solution to the service operational cost minimization problem when the problem size is small; otherwise, we devise a randomized algorithm for it through relaxing the ILP to a Linear Programming (LP) and adopting randomization rounding technique [22], at the expense of a bounded resource violation. We also develop a deterministic approximation algorithm for the problem at a moderate resource violation. We also develop an efficient heuristic algorithm for the problem without any resource violation.
- We deal with the online throughput maximization problem of dynamic admissions of multi-source IoT application requests, through formulating an ILP solution to its offline version as a benchmark first. We then develop an online algorithm for it with a provable competitive ratio.
- We evaluate the performance of the proposed algorithms through experimental simulations. Simulation results show that the proposed algorithms are promising.

The rest of the paper is organized as follows. Section 2 summarizes the related work on service provisioning in MEC. Section 3 introduces notions, notations and the problem definitions. Section 4 formulates an ILP solution, and devises a randomized approximation algorithm, a deterministic approximation algorithm, and a heuristic algorithm for the service operational cost minimization problem. Section 5 formulates an ILP solution and develops an online algorithm for the problem of dynamic multi-source IoT application request admissions. Section 6 evaluates the proposed algorithms empirically, and Section 7 concludes the paper.

2 Related Work

With the development of networking technology, more and more IoT applications, exemplified as environmental monitoring, augmented maps and health monitoring, have attracted growing attention. Enormous efforts have been taken for the service provisioning of IoT applications under a wide variety of scenarios. For example, Baker *et al.* [2] offered a solution to the cooperation and integration of multiple clouds to greatly reduce data interchanges among geographically distributed IoT applications. They proposed efficient composition plans to alleviate energy consumption. Basu *et al.* [3] investigated the efficient scheduling of heterogeneous IoT applications in a cloud computing environment. They proposed an intelligent model based on a Genetic Algorithm and Ant Colony Optimization to minimize the total lifespan of tasks subject to the task dependencies and the load balancing requirements. Cai *et al.* [4] considered different IoT functions to build a novel framework for big data processing in cloud computing, and identified the challenges and opportunities in provisioning IoT big data services. These mentioned works are based on powerful clouds, not MEC environments.

MEC has emerged as a promising paradigm to support delay-sensitive IoT applications. There are several investigations of service provisioning for single-source IoT applications in MEC environments. For example, Elgendy *et al.* [5] studied the secure service provisioning in a mobile IoT network, and proposed an efficient heuristic algorithm to minimize the total energy consumption of IoT devices by compressing the data to be transmitted. Hu *et al.* [8] considered an ultra-dense MEC network and devised a dynamic IoT application request scheduling scheme. They also proposed efficient heuristic algorithms to minimize either response delay or energy consumption of mobile users. Li *et al.* [11], [12] considered the accumulative user satisfaction problem of using services provided by an MEC network for IoT service requests with delay requirements. They introduced a novel metric to measure user service satisfaction and devised efficient approximation and online algorithms for the problem under static and dynamic service request arrival settings. Liang *et al.* [15], [16] recently introduced a novel service reliability augmentation problem in MEC networks with the aim to maximize the accumulated service reliability of all users while ensuring fair allocations of limited resource to different users, and they devised randomized approximation algorithms for the problem by adopting the randomized rounding technique [22]. Liu *et al.* [17] considered the strategic routing path planning and efficient resource allocation for a vehicle-mounted edge server to maximize the number of IoT tasks completed while meeting their deadline requirements. Sun *et al.* [26] investigated dynamic resource caching in cloudlets to minimize the average delay of transmitting contents to their corresponding clients. They also considered how to determine a suitable resource caching strategy to maximize the total energy savings. Song *et al.* [25] investigated a dynamically distributed IoT task management problem in an MEC network, and delivered a Mixed-integer Linear Programming (MILP) solution for IoT task processing while fulfilling the Quality-of-Service (QoS) requirements of users. Xu *et al.* [31] considered the operational cost minimization problem for the implementation of IoT applications with Service Function Chain (SFC) requirements. They focused on IoT application placements in MEC, by proposing randomized and heuristic algorithms for the problem. Xu *et al.* [32] studied the age of information (AoI) minimization problem of IoT big data processing in MEC networks via online learning. Xia *et al.* [28] addressed a service caching problem for delay-sensitive IoT applications through caching the services from remote clouds to cloudlets in a multi-tiered MEC network. They first proposed two approximation algorithms in the case where it exists only a single type of service. They also designed an efficient heuristic algorithm for the problem in a general case. Xiong *et al.* [29] developed a Markov Decision Process (MDP) based approach for resource allocation in an IoT edge computing environment, with the aim of minimizing the total completion time and the average resource consumption. In this paper, we study an essentially different optimization problem - the service home identification problem of service provisioning in MEC networks for multi-source IoT applications, and we make use of the randomized rounding technique for solving this new problem.

There are only a few studies of service provisioning for multi-source IoT applications in MEC networks. For example, Goudarzi *et al.* [6] proposed an IoT application placement algorithm with the aim of minimizing the execution time and the total energy consumption, considering the heterogeneity of multiple IoT applications and dependency among tasks. Yu *et al.* [34] studied the IoT application provisioning problem with each IoT application receiving data streams from multiple sources. They proposed approximation schemes for different cases to meet both bandwidth and delay requirements. They developed an efficient heuristic algorithm for the problem. Recently, Li *et al.* [13], [14] developed an efficient heuristic algorithm for the stream data aggregation problem of multi-source IoT applications. They focused on building a routing tree rooted at service home such that the data streams from multiple sources can be aggregated via the tree while enforcing the SFC requirement of the IoT application in each routing path from a leaf node (source) to the root node (service home) in the tree. They assumed that the service home of each multi-source IoT application in the MEC network is given.

Unlike the aforementioned studies, in this paper we study the service provisioning for multi-source IoT applications in an MEC environment. The work of this paper is complementary to the works in [13], [14], by identifying the service home for each multi-source IoT application in an MEC network such that the total operational service cost of accommodating all applications is minimized.

3 Preliminaries

In this section we first introduce the system model, we then give notions and notations, and we finally present problem definitions.

3.1 System Model

A Mobile Edge Computing (MEC) network is represented by an undirected graph $G = (V, E)$, where V is the set of nodes and E is the set of links between nodes. Each node $v \in V$ is a cloudlet (edge cloud) co-located with an access point (AP). Each cloudlet $v \in V$ has computing capacity C_v, and the cost of per unit computing resource in it is $cost_v$. Furthermore, an AP and its co-located cloudlet are connected by a high-speed optical cable, and the communication delay between them thus is negligible. Fig. 1 is an illustrative example of an MEC network that accommodates two multi-source IoT applications. We further assume that there is a hypervisor in the MEC network for the resource scheduling of the service home identification of each multi-source IoT application.

3.2 Problem Formulations

Given an MEC network $G = (V, E)$ and a collection of multi-source IoT applications $\mathcal{Q} = \{G_1, G_2, \ldots, G_K\}$ in which each IoT application G_k is a tuple $\langle (s_{k,1}, \rho_{k,1}), (s_{k,2}, \rho_{k,2}), \ldots, (s_{k,l_k}, \rho_{k,l_k}) \rangle$ with $l_k \geq 1$ different sources located at l_k different geographical locations of the MEC network. Each source (gateway) $s_{k,l}$ is the sink of a sensor subnetwork of the IoT application that continuously uploads the data stream from the IoT devices of the subnetworks to the MEC. The uploaded data streams from these sources will then be

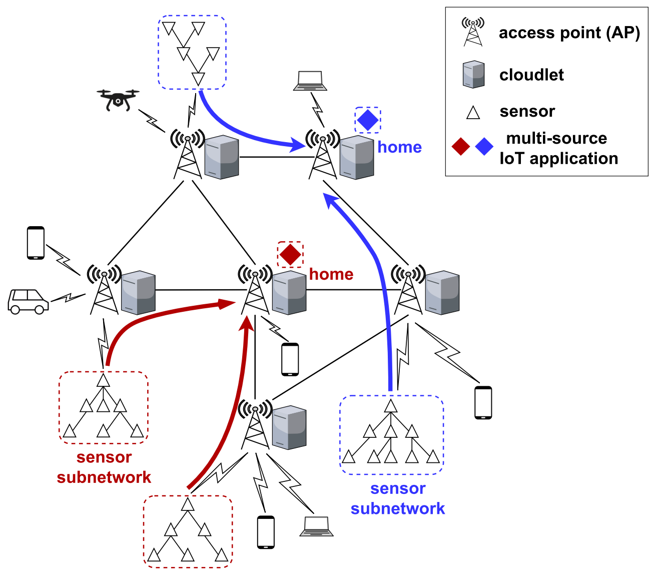


Fig. 1. An illustrative example of an MEC network that consists of 6 cloudlets. There are 2 multi-source IoT applications (indicated by red and blue colors, respectively), each consists of 2 sensor subnetworks, where the data streams from the 2 sources of each multi-source IoT application are routed to its service home (cloudlet).

routed to the service home (cloudlet) of the IoT application for further processing and storage, assuming that the stream data rate from each source $s_{k,l}$ of IoT application G_k is $\rho_{k,l}$ with $1 \leq l \leq l_k$.

For each IoT application G_k, if a cloudlet $v \in V$ has been identified to be its service home, then cloudlet v must have sufficient computing resource to accommodate the data streams from its l_k sources, while the demanded computing resource R_k at its service home for processing its data streams is

$$R_k = r_k(\rho_{k,1}, \rho_{k,2}, \ldots, \rho_{k,l_k}), \tag{1}$$

where $r_k(\cdot, \ldots, \cdot)$ is an aggregation function. For example, the value of function r_k can be proportional to the sum of data packet rates of different sources, or the average of all source packet rates. Notice that different multi-source IoT applications may have different aggregation functions.

The service operational cost of service provisioning for IoT application G_k thus consists of the computing cost (processing cost) at its service home - cloudlet v, and the communication cost that is the sum of the amounts of bandwidth consumed from each source (an AP) to its service home in the MEC network. Specifically, the service operational cost of G_k is defined as follows.

Let $L(v, u)$ be the shortest path in an MEC network G between AP nodes u and v, and the weight of each link in the path is the communication cost per packet data on the link. Then, *the service operational cost* of G_k by choosing cloudlet $v \in V$ as its service home is

$$c(k, v) = \sum_{l=1}^{l_k} \rho_{k,l} \cdot L(s_{k,l}, v) + R_k \cdot cost_v, \tag{2}$$

where $\rho_{k,l} \cdot L(s_{k,l}, v)$ is the communication cost by transferring the data stream from source $s_{k,l}$ to its service home v, while $R_k \cdot cost_v$ is the computing cost of all data streams of G_k at cloudlet v, and $cost_v$ is the unit computing cost of cloudlet $v \in V$.

In the following we define two service home identification problems of service provisioning for multi-source IoT applications.

Definition 1. *Given a set $\mathcal{Q} = \{G_1, G_2, \ldots, G_k\}$ of multi-source IoT application requests,* the service operational cost minimization problem *of service provisioning for the set $\mathcal{Q}$ of multi-source IoT applications in an MEC network $G = (V, E)$ is to identify a service home for each IoT application such that the total service operational cost is minimized, subject to resource capacities on cloudlets in G.*

The problem in Definition 1 considered a snapshot of service provisioning for multi-source IoT applications at a specific time point, where a set $\mathcal{Q}$ of multi-source IoT applications are given in advance. In reality, the MEC network as a public service provisioning platform accepts various IoT applications from different users at different time points dynamically. To cast such dynamic multi-source IoT request admissions, we define another service home identification problem of service provisioning for dynamic multi-source IoT application request admissions with the aim to maximize the network throughput as follows.

Definition 2. *Given an MEC network $G = (V, E)$ and a finite time horizon T, a sequence of multi-source IoT application requests that arrive one by one without the knowledge of future arrivals,* the online throughput maximization problem *of service provisioning for dynamic multi-source IoT application request admissions in G is to maximize the number of multi-source IoT application request admissions by identifying service homes to the admitted requests, subject to resource constraints on cloudlets in G.*

Theorem 1. *The service operational cost minimization problem of service provisioning in an MEC network $G(V, E)$ for multi-source IoT applications is NP-hard, and cannot be approximated within any factor unless NP = P.*

Proof. We first show the NP-hardness of the service operational cost minimization problem by reducing the minimum-cost generalized assignment problem (GAP) to a special case of the problem where the communication cost of data streams from different sources to the service home of a multi-source IoT application is neglected. The reduction is as follows. The minimum-cost GAP contains $|V|$ bins, and each bin $v \in V$ has a computing capacity C_v. There are $|\mathcal{Q}|$ items with each item G_k of size R_k (the accumulative computing resource demand of all data stream rates). The service operational cost associated with packing G_k to bin v is $c(k, v)$ as defined in Eq. (2). The minimum-cost GAP is to pack as many items as possible to the $|V|$ bins such that the total cost is minimized, subject to the computing capacity on each bin. It can be seen that this is a special case of the problem of concern in this paper, and a solution to the latter will deliver a solution to the minimum-cost GAP. It is well known that the minimum-cost GAP is NP-hard [21], so the service operational cost minimization problem is NP-hard, too.

We then show the inapproximability of the service operational cost minimization problem, by a gap-introducing reduction from the decision version of the well-known NP-complete problem - the bin packing problem (BPP) which is defined as follows. Given a set of $|Q|$ items, each of which has a different size $R_k \in [0, 1]$, and $|V|$ bins of size 1, the problem is to determine whether there is a proper packing such that all items can be packed into the $|V|$ bins.

Given a BPP instance, we construct an instance of the service operational cost minimization problem as follows. For each item G_k of size R_k in the BPP instance, we create a user request G_k, which demands the amount R_k of computing resource in the instance of the service operational cost minimization problem. For each bin $v \in V$ of size 1 in the BPP instance, we create a cloudlet v with a capacity of 1 in the corresponding instance of the service operational cost minimization problem. For each pair of a user request G_k and a cloudlet v in the instance of the service operational cost minimization problem, we set the operation cost as $c(k, v) = 1$. Therefore, each item corresponds to a user request, and each bin corresponds to a cloudlet. We now prove the inapproximability of the service operational cost minimization problem by a contradiction.

Assume that we have an approximation algorithm $\mathcal{A}$ with an approximation $\alpha > 1$ for the service operational cost minimization problem. By the above instance construction, if the BPP instance has a feasible solution, $\mathcal{A}$ must return a solution with a cost of $|Q|$ in the instance of the service operational cost minimization problem since there are $|Q|$ user requests and each operation cost is 1. Otherwise (i.e., the BPP instance's answer is no), $\mathcal{A}$ returns no solution since there is no feasible solution in the instance of the service operational cost minimization problem. Thus, $\mathcal{A}$ can be applied to answer the BPP in polynomial time by the above polynomial-time reduction. It is impossible unless NP = P since the BPP is NP-complete (i.e., a contradiction). Therefore, such an approximation algorithm $\mathcal{A}$ does not exist unless NP = P. Hence, the service operational cost minimization problem cannot be approximated within any factor unless NP = P. □

Theorem 2. *The online throughput maximization problem in an MEC network $G = (V, E)$ for multi-source IoT applications is NP-hard.*

Proof. Consider a special case of the problem, where the time horizon consists of only one time slot and the communication cost of data streams from different sources to the service home is ignored. We assume that the profit of packing any item into a bin is 1. Then, the special case of the problem of concern becomes a maximum-profit GAP while the latter is NP-hard [21], the online throughput maximization problem of service provisioning for multi-source IoT applications thus is NP-hard. □

4 Algorithms for the Service Operational Cost Minimization Problem

In this section, we first formulate an Integer Linear Programming (ILP) solution to the service operational cost minimization problem. We then devise a randomized algorithm and a deterministic approximation algorithm for the problem at the expense of bounded resource violations. We also develop a heuristic solution to the problem without resource violation. We finally analyze the approximation ratio of the randomized algorithm and the time complexity of the heuristic algorithm.

4.1 Integer Linear Programming

In the following, we formulate an ILP solution to the service operational cost minimization problem. Let $x_{k,v}$ be a binary decision variable, where $x_{k,v} = 1$ indicates that cloudlet v is identified as the service home of G_k, and $x_{k,v} = 0$ otherwise.

The ILP solution to the service operational cost minimization problem is then formulated as follows:

$$\text{Minimize} \qquad \sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} c(k, v) \cdot x_{k,v}, \tag{3}$$

subject to:

$$\text{Eq. } (1),\ (2),$$

$$\sum_{k=1}^{|\mathcal{Q}|} R_k \cdot x_{k,v} \le C_v, \quad \forall v \in V, \tag{4}$$

$$\sum_{v \in V} x_{k,v} = 1, \quad \forall G_k \in \mathcal{Q}, \tag{5}$$

$$x_{k,v} \in \{0, 1\}, \quad \forall G_k \in \mathcal{Q},\ \forall v \in V, \tag{6}$$

where Constraint (4) ensures that the accumulative computing resource consumption of different IoT applications on each cloudlet $v \in V$ is no more than its computing capacity C_v. Constraint (5) ensures that each multi-source IoT application has a service home (cloudlet) in the MEC network. Constraint (6) shows that $x_{k,v}$ is a binary decision variable indicating whether cloudlet v is identified as the service home of G_k.

4.2 Randomized Algorithm

We now devise a randomized algorithm based on the relaxation of the ILP solution, by relaxing the value range of binary variables $x_{k,v}$ in Eq. (6) to real numbers between 0 and 1 as follows:

$$x_{k,v} \in [0, 1], \quad \forall G_k \in \mathcal{Q},\ \forall v \in V. \tag{7}$$

The basic idea behind the proposed randomized algorithm is as follows. An optimal fractional solution of the corresponding Linear Programming (LP) relaxation of the ILP formulation can be obtained in polynomial time. An integral solution is then obtained by randomly rounding on the fractional solution obtained [22]. This integral solution becomes a feasible solution to the original problem with high probability.

The detailed randomized algorithm for the service operational cost minimization problem is given in `Algorithm` 1.

4.3 Approximation Algorithm

In the following, we devise a deterministic approximation algorithm for the service operational cost minimization problem. Although it is very likely that the randomized algorithm `Algorithm` 1 can deliver a feasible solution to the problem, Algorithm 1 is non-deterministic. This motivates us to develop a deterministic approximation algorithm for the problem that can deliver an optimal solution while the resource violation is guaranteed no more than twice the capacity of each cloudlet.

Algorithm 1. A Randomized Algorithm for the Service Operational Cost Minimization Problem

```
Input: An MEC network $G(V,E)$, $|V|$ cloudlets with each $v$ having capacity $C_v$, and a set $\mathcal{Q}$ of multi-source IoT applications.
Output: Minimize the total service operational cost by identifying a service home and the routing path from each of its sources to the service home in the MEC network for each multi-source IoT application.
1: Calculate the shortest paths between each pair of APs.
2: for each $G_k \in \mathcal{Q}$ do
3:     for each cloudlet $v \in V$ with $C_v \geq R_k$ do
4:         Calculate $c(k,v)$ by Eq. (2);
5:     end for ;
6: end for ;
7: Solve the relaxed version LP of the ILP in polynomial time;
8: Let $\widetilde{OPT}$ be the optimal solution of the LP and $\tilde{x}_{k,v}$ be the value of each variable $x_{k,v}$, where $\tilde{x}_{k,v} \in [0,1]$;
9: An integral solution $\hat{x}_{k,v}$ can be obtained, by the randomized rounding approach in [22]. That is, $\hat{x}_{k,v}$ is set to 1 with the probability of $\tilde{x}_{k,v}$; otherwise, $\hat{x}_{k,v}$ is set to 0; The choice is performed in an exclusive manner, with Constraint (5): $\forall v \in V$, exactly one of the variables $\hat{x}_{k,v}$ is set to 1, and the rest are set to be 0s. This random choice is made independently for all $v$s;
10: return A candidate integral solution $\hat{S}$ based on $\hat{x}_{k,v}$, which will be a feasible solution to the ILP with high probability.
```

We observe that the service operational cost minimization problem can be reduced to a minimum-cost generalized assignment problem (GAP) as follows. Each multi-source IoT application $G_k \in \mathcal{Q}$ has a corresponding item with size R_k by Eq. (1), and each cloudlet $v \in V$ corresponds to a bin with the capacity C_v. The cost of assigning item G_k to bin v is $c(k,v)$ by Eq. (2).

By applying the approximation technique due to Shmoys and Tardos [24], a deterministic approximation algorithm for the problem is developed as follows.

We first solve the Linear Programming (LP) - the relaxed version of the Integer Linear Programming (ILP) (3), and obtain $\widetilde{OPT}$, which is the optimal solution of the LP. Denote by $\tilde{x}_{k,v}$ the value of each variable $x_{k,v}$, with $\tilde{x}_{k,v} \in [0,1]$.

Following [24], we then construct a weighted bipartite graph $B = (\mathcal{Q}', W', E')$, where $\mathcal{Q}'$ and W' are two disjoint sets of nodes, and E' is the set of edges between nodes in $\mathcal{Q}'$ and W', respectively. Especially, $\mathcal{Q}' = \{G_k \mid 1 \leq k \leq |\mathcal{Q}|\}$, where $\mathcal{Q}$ is the set of multi-source IoT applications, and $W' = \{w_{v,j}, v \in V \mid 1 \leq j \leq \sigma_v\}$, where V is the set of cloudlets, and $\sigma_v = \lceil \sum_{G_k \in \mathcal{Q}} \tilde{x}_{k,v} \rceil$. σ_v nodes for cloudlet $v \in V$ then are created in W'.

We third sort the multi-source IoT applications in $\mathcal{Q}'$ in non-increasing order of their demanded computing resource R_k with $1 \leq k \leq |\mathcal{Q}|$. For notation simplicity, we assume that $R_1 \geq R_2 \geq \cdots \geq R_{|\mathcal{Q}|}$.

For each cloudlet $v \in V$, the derived nodes from $w_{v,1}, \ldots, w_{v,\sigma_v}$ are treated as bins with each having capacity 1, and each multi-source IoT application G_k with $\tilde{x}_{k,v} > 0$ is treated as an item with size of $\tilde{x}_{k,v}$ that has also been sorted in non-increasing order of demanded computing resource R_k.

The packing procedure proceeds as follows. Initially, we pack the sorted items into the first bin $w_{v,1}$ one by one until packing an item G_k causes the overflow (i.e., the bin capacity of $w_{v,1}$ is violated). We then partition item G_k into two disjoint parts: G_k^1 and G_k^2, such that packing G_k^1 into $w_{v,1}$ makes $w_{v,1}$ saturated, i.e., the capacity of $w_{v,1}$ is fully used. We then pack G_k^2 into the next bin $w_{v,2}$. This procedure continues until all items are packed.

If item G_k is packed (or partially packed) into bin $w_{v,j}$, an edge $e(G_k, w_{v,j})$ is added to the edge set E' with the edge cost of $c(k,v)$, i.e., the service operational cost of identifying v as the service home of G_k.

Having constructed the bipartite graph B, we finally find a minimum-cost maximum matching M in B, which exactly matches all multi-source IoT application nodes. For each edge $e(G_k, w_{v,j})$ in M, cloudlet v is identified as the service home of G_k. The detailed algorithm is given in Algorithm 2.

Algorithm 2. A Deterministic Approximation Algorithm for the Service Operational Cost Minimization Problem

```
Input: An MEC network $G = (V,E)$, and a set $\mathcal{Q}$ of multi-source IoT applications.
Output: Find a solution to the problem, i.e., identify the service home and the routing path from each of its sources to the service home in the MEC network for each multi-source IoT application.
1: for each $G_k \in \mathcal{Q}$ do
2:     for each cloudlet $v \in V$ with $C_v \geq R_k$ do
3:         calculate $c(k,v)$ by Eq. (2);
4:     end for ;
5: end for ;
6: Solve the relaxed version LP of the ILP in polynomial time;
7: Let $\widetilde{OPT}$ be the optimal solution of the LP and $\tilde{x}_{k,v}$ be the value of each variable $x_{k,v}$, where $\tilde{x}_{k,v} \in [0,1]$;
8: Construct a bipartite graph $B = (\mathcal{Q}', W', E')$, using the technique by Shmoys and Tardos [24].
9: Find a minimum-cost matching $M$ on $B$ which exactly matches all multi-source IoT application nodes in $B$, by invoking the Hungarian algorithm.
10: for each edge $e(G_k, w_{v,j}) \in M$ do
11:     Identify cloudlet $v$ as the service home of the multi-source IoT application $G_k$.
12: end for
```

4.4 Heuristic Algorithm

Although the randomized algorithm for the service operational cost minimization problem can deliver a feasible solution with high probability, the achieved solution is at the expense of the bounded computing resource capacity violation, so is the proposed deterministic approximation algorithm. In the following we develop an efficient heuristic algorithm without resource violation.

The algorithm proceeds iteratively. Within iteration p, a subset of multi-source IoT applications in $\mathcal{Q}$ with the minimum total operational cost is identified, through finding a minimum-cost maximum matching in an auxiliary bipartite graph $G_B^{(p)} = (X^{(p)}, Y^{(p)}; E_{XY}^{(p)})$, where $X^{(p)}$ is the subset of cloudlets with residual computing capacities at iteration p, and $Y^{(p)}$ is the set of yet-to-admitted IoT applications in $\mathcal{Q}$ at iteration p. Initially, $X^{(1)}$ is the set of cloudlets in V and $Y^{(1)} = \mathcal{Q}$. There is an edge in $E_{XY}^{(p)}$ between a cloudlet $v \in X^{(p)}$ and a multi-source IoT application $G_k \in Y^{(p)}$ with edge weight $c(k,v)$ if the residual computing capacity of cloudlet v is no less than the computing resource demand R_k of application G_k. Let M_p be a minimum-cost maximum matching in $G_B^{(p)}$ at iteration p. We deduct the amount of computing resource from the corresponding cloudlet for each matched edge in M_p, and the IoT applications in M_p are then removed. The next auxiliary bipartite graph then is constructed. This procedure continues until all IoT applications are admitted.

Let $\mathcal{P}$ be the number of iterations of the proposed algorithm. Then, the union $M = \cup_{p=1}^{\mathcal{P}} M_p$ of the found minimum-cost maximum matchings forms a solution to the problem, and the weighted sum $\sum_{p=1}^{\mathcal{P}} c(M_p)$ of all edges in M_p is the total service operational cost of all admitted IoT applications, where $c(M_p)$ is the weighted sum of all edges in M_p.

The detailed heuristic algorithm for the service operational cost minimization problem is given in Algorithm 3.

Algorithm 3. A Heuristic Algorithm for the Service Operational Cost Minimization Problem

```
Input: An MEC network $G(V,E)$, $|V|$ cloudlets with each $v$ having capacity $C_v$, and a set $\mathcal{Q}$ of multi-source IoT applications.
Output: Minimize the total service operational cost by identifying the service home and routing path from each of its sources to the service home in the MEC network for each multi-source IoT application.
1: Calculate the shortest paths between each pair of APs.
2: $M \leftarrow \emptyset$; $total\_cost \leftarrow 0$; $p \leftarrow 1$; /* the solution */
3: $X^{(p)} \leftarrow V$, $Y^{(p)} \leftarrow \mathcal{Q}$, and calculate $E_{XY}^{(p)}$;
4: while $E_{XY}^{(p)} \neq \emptyset$ do
5:     Construct graph $G^{(p)} = (X^{(p)}, Y^{(p)}; E_{XY}^{(p)})$;
6:     Find a minimum-cost maximum matching $M_p$ in $G^{(p)}$, by invoking the Hungarian algorithm;
7:     $M \leftarrow M \cup M_p$;
8:     $total\_cost \leftarrow total\_cost + c(M_p)$;
9:     Allocate cloudlet resources to the matched requests in $M_p$;
10:    Update the residual computing capacity of each cloudlet;
11:    $p \leftarrow p + 1$;
12:    Calculate $X^{(p)}$, $Y^{(p)}$ and $E_{XY}^{(p)}$;
13: end while
14: return $M$ corresponds to the assignment of IoT applications in $\mathcal{Q}$, while $total\_cost$ is the sum of their service costs.
```

4.5 Analysis of the Proposed Algorithms

The rest is to analyze the approximation ratio of Algorithm 1 and bound the computing resource violation on each cloudlet. The time complexity analysis of Algorithm 3 is also given.

Denote by Γ a given value, which is defined as follows.

$$\Gamma = \max\{R_k,\ c(k,v) \mid G_k \in \mathcal{Q}, v \in V\}, \tag{8}$$

where $\max\{c(k,v) \mid G_k \in \mathcal{Q}, v \in V\}$ is the maximum cost of hosting a multi-source IoT application among the $|V|$ cloudlets, and $\max\{R_k \mid G_k \in \mathcal{Q}\}$ is the maximum computing resource demand of a multi-source IoT application.

Theorem 3. *Given an MEC network $G = (V,E)$ and a set $\mathcal{Q}$ of multi-source IoT applications, there is a randomized algorithm, Algorithm 1, with high probability of $\min\{1 - \frac{1}{|\mathcal{Q}|}, 1 - \frac{1}{|V|}\}$, for the service operational cost minimization problem. The expected approximation ratio of Algorithm 1 is 2, and the computing resource consumption at any cloudlet is no more than twice its capacity, provided that $OPT \geq 3\Gamma \ln |\mathcal{Q}|$ and $\min\{C_v \mid v \in V\} \geq 6\Gamma \ln |V|$, where Γ is defined in Eq. (8), OPT is the optimal solution of the problem, and $\min\{C_v \mid v \in V\}$ is the minimum computing capacity of a cloudlet.*

Proof. We analyze the approximation ratio of the randomized algorithm, Algorithm 1, and show that the computing resource capacity violation of each cloudlet is upper bounded.

Let $\widetilde{OPT}$ and OPT be the optimal solutions of the LP and the ILP, respectively. The value of $\widetilde{OPT}$ is a lower bound on the optimal solution of the ILP as this is a minimization optimization problem, i.e., $\widetilde{OPT} \leq OPT$.

Let $\tilde{x}_{k,v}$ be the value of variable $x_{k,v}$ in the LP solution, which is within [0,1]. By Algorithm 1, the delivered integral solution $\hat{x}_{k,v}$ is set to 1 with the probability of $\tilde{x}_{k,v}$; otherwise, $\hat{x}_{k,v}$ is set to 0.

Denote by $y_{k,v} = \frac{c(k,v)}{\Gamma} \cdot \hat{x}_{k,v}$ a random variable derived from the integral solution $\hat{x}_{k,v}$. It can be seen that the value of $y_{k,v}$ is $\frac{c(k,v)}{\Gamma}$ with probability $\tilde{x}_{k,v}$; otherwise, $y_{k,v}$ is 0. Thus, the value range of $y_{k,v}$ is within [0,1], as $\frac{c(k,v)\cdot \hat{x}_{k,v}}{\Gamma} \leq \frac{c(k,v)\cdot \hat{x}_{k,v}}{\max\{c(k,v) | G_k \in \mathcal{Q}, v \in V\}} \leq 1$ and $\mathbb{E}[y_{k,v}] = \frac{c(k,v)}{\Gamma} \cdot \tilde{x}_{k,v}$. Then

$$\mathbb{E}\left[\sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} y_{k,v}\right] = \sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} \frac{c(k,v) \cdot \tilde{x}_{k,v}}{\Gamma} = \frac{\widetilde{OPT}}{\Gamma}. \tag{9}$$

Let $\beta > 0$ be a constant. As the solution delivered by Algorithm 1 is $\sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} c(k,v) \cdot \hat{x}_{k,v}$, by Chernoff Bounds [20], we have

$$\begin{aligned}
&\mathbf{Pr}\left[\sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} c(k,v) \cdot \hat{x}_{k,v} \geq (1+\beta) \cdot OPT\right] \\
\leq &\mathbf{Pr}\left[\sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} c(k,v) \cdot \hat{x}_{k,v} \geq (1+\beta) \cdot \widetilde{OPT}\right], \\
&\qquad\qquad \text{since } \widetilde{\mathrm{OPT}} \leq \mathrm{OPT} \\
= &\mathbf{Pr}\left[\sum_{k=1}^{|\mathcal{Q}|} \sum_{v \in V} y_{k,v} \geq (1+\beta) \cdot \frac{\widetilde{OPT}}{\Gamma}\right], \\
&\qquad \text{by yk,v} = \text{c(k,v) } \Gamma \cdot \hat{\text{x}}\text{k, v} \\
= &\mathbf{Pr}\left[\sum_{k=1}^{|\mathcal{Q}|} Y_k \geq (1+\beta) \cdot \frac{\widetilde{OPT}}{\Gamma}\right], \text{ let Yk=} \textstyle\sum \text{v} \in \text{V yk,v} \\
\leq &exp\left(\frac{-\beta^2 \cdot \widetilde{OPT}}{(2+\beta) \cdot \Gamma}\right), \quad \text{for all } \beta > 0,
\end{aligned} \tag{10}$$

where Inequality (10) is obtained by the Chernoff bound as variables $Y_1, Y_2, \ldots, Y_{|\mathcal{Q}|}$ are independent random variables.

We further assume that

$$exp\left(\frac{-\beta^2 \cdot \widetilde{OPT}}{3\Gamma}\right) \le \frac{1}{|\mathcal{Q}|} \quad \text{with } \beta > 0. \tag{11}$$

Then, when $0 < \beta \le 1$, Inequality (10) can be transformed as follows:

$$exp\left(\frac{-\beta^2 \cdot \widetilde{OPT}}{(2+\beta)\cdot\Gamma}\right) \le exp\left(\frac{-\beta^2 \cdot \widetilde{OPT}}{3\Gamma}\right) \le \frac{1}{|\mathcal{Q}|}. \tag{12}$$

We then have

$$\beta \ge \sqrt{\frac{3\Gamma \ln|\mathcal{Q}|}{\widetilde{OPT}}} \ge \sqrt{\frac{3\Gamma \ln|\mathcal{Q}|}{OPT}}. \tag{13}$$

Since $\beta \le 1$, we must have $OPT \ge 3\Gamma \ln|\mathcal{Q}|$ by Inequality (13). The approximation ratio of the randomized algorithm, Algorithm 1, is 2 with high probability $\min\{1-\frac{1}{|\mathcal{Q}|}, 1-\frac{1}{|V|}\}$, since $1+\beta \le 2$.

We finally analyze the computing resource capacity violation on each cloudlet $v \in V$ by Algorithm 1.

Similar to the definition of $y_{k,v}$, let $z_{k,v} = \frac{R_k}{\Gamma} \cdot \hat{x}_{k,v}$ be a random variable derived from the integral solution $x_{k,v}$. It can be seen that the value of $z_{k,v}$ is $\frac{R_k}{\Gamma}$ with probability of $\tilde{x}_{k,v}$; otherwise, $z_{k,v}$ is 0. The values of random variables $z_{k,v}$ are within [0,1] as $\frac{R_k \cdot \hat{x}_{k,v}}{\Gamma} \le \frac{R_k \cdot \hat{x}_{k,v}}{\max\{R_k | G_k \in \mathcal{Q}\}} \le 1$. And $\mathbb{E}[z_{k,v}] = \frac{R_k \cdot \tilde{x}_{k,v}}{\Gamma}$. Then, for a cloudlet v, we have

$$\mathbb{E}\left[\sum_{k=1}^{|\mathcal{Q}|} z_{k,v}\right] = \sum_{k=1}^{|\mathcal{Q}|} \frac{R_k \cdot \tilde{x}_{k,v}}{\Gamma} = \frac{\tilde{C}_v}{\Gamma}, \tag{14}$$

where $\tilde{C}_v$ is the amount of computing resource consumed at cloudlet v in the solution of the LP, and $\tilde{C}_v \le C_v$.

By Algorithm 1, the computing resource consumption on cloudlet v is $\sum_{k=1}^{|\mathcal{Q}|} R_k \cdot \hat{x}_{k,v}$. Let $\beta_1 > 0$ be a constant. Since there are $|V|$ cloudlets, the probability of the computing resource capacity violation on any cloudlet is

$$\begin{aligned}
&\mathbf{Pr}\left[\bigvee_{v\in V}\sum_{k=1}^{|\mathcal{Q}|} R_k \cdot \hat{x}_{k,v} \ge (1+\beta_1)\cdot C_v\right] \\
\le&\mathbf{Pr}\left[\bigvee_{v\in V}\sum_{k=1}^{|\mathcal{Q}|} R_k \cdot \hat{x}_{k,v} \ge (1+\beta_1)\cdot \tilde{C}_v\right], \text{since } \tilde{\mathrm{C}}\mathrm{v} \le \mathrm{Cv} \\
\le&\sum_{v\in V}\mathbf{Pr}\left[\sum_{k=1}^{|\mathcal{Q}|} z_{k,v} \ge (1+\beta_1)\cdot \frac{\tilde{C}_v}{\Gamma}\right], \\
&\text{by zk,v} = \text{Rk} \ \ \Gamma \cdot \hat{\mathrm{x}}\text{k, v and the Union Bound Inequality} \\
\le&|V|\cdot exp\left(\frac{-\beta_1^2\cdot \tilde{C}_v}{(2+\beta_1)\cdot\Gamma}\right), \quad \text{for all } \beta_1 > 0, \\
&\text{by Eq. (14) and Chernoff Bounds.}
\end{aligned} \tag{15}$$

We set $\beta_1 \le 1$ and let

$$exp\left(\frac{-\beta_1^2\cdot \tilde{C}_v}{(2+\beta_1)\cdot\Gamma}\right) \le \frac{1}{|V|^2}. \tag{16}$$

With $0 < \beta \le 1$, Eq. (16) is transformed as follows:

$$exp\left(\frac{-\beta_1^2\cdot \tilde{C}_v}{3\Gamma}\right) \le \frac{1}{|V|^2}. \tag{17}$$

Then, we have

$$\beta_1 \ge \sqrt{\frac{6\Gamma \ln|V|}{\tilde{C}_v}} \ge \sqrt{\frac{6\Gamma \ln|V|}{C_v}}, \text{ by } \tilde{\mathrm{C}}\mathrm{v} \le \mathrm{Cv}. \tag{18}$$

Since $\beta_1 \le 1$, we must have $C_v \ge 6\Gamma \ln|V|$ by Ineq. (18). To ensure that $C_v \ge 6\Gamma \ln|V|$ for any cloudlet $v \in V$, we must have

$$\min\{C_v \mid v \in V\} \ge 6\Gamma \ln|V|. \tag{19}$$

We thus have

$$\begin{aligned}
&\mathbf{Pr}\left[\bigvee_{v\in V}\sum_{k=1}^{|\mathcal{Q}|} R_k \cdot x_{k,v} \ge (1+\beta_1)\cdot C_v\right] \\
\le&|V|\cdot\frac{1}{|V|^2} = \frac{1}{|V|} \quad \text{, by Eq. (15) and (16).}
\end{aligned} \tag{20}$$

Therefore, the computing resource consumption at any cloudlet is no more than twice its capacity, with high probability $\min\{1-\frac{1}{|\mathcal{Q}|}, 1-\frac{1}{|V|}\}$, since $1+\beta_1 \le 2$.

The theorem then follows. □

Theorem 4. *Given an MEC network $G(V,E)$ and a set $\mathcal{Q}$ of multi-source IoT applications, there is a deterministic approximation algorithm, Algorithm 2 for the service operational cost minimization problem. The cost of the solution delivered by Algorithm 2 is no more than that of the optimal solution while the amount of computing resource consumed at any cloudlet is no more than twice its capacity [24]. The time complexity of Algorithm 2 is $O(|\mathcal{Q}|^3 \cdot |V|^3)$.*

Proof. See the detailed analysis in [24], omitted. □

Theorem 5. *Given a mobile edge computing network $G = (V,E)$ and a set $\mathcal{Q}$ of multi-source IoT applications with each having multiple sources, there is an algorithm, Algorithm 3, for the service operational cost minimization problem in G, which is $O(|\mathcal{Q}| \cdot (|\mathcal{Q}|+|V|)^3)$ time.*

Proof. We first show that the solution delivered by Algorithm 3 is feasible. As none of the constraints is violated, it can be verified that the solution indeed is a feasible solution.

We then analyze the time complexity of Algorithm 3. It takes $O(|V|^3)$ time to find a shortest path between each pair of APs, while it takes $O((|\mathcal{Q}|+|V|)^3)$ time to find a minimum-cost maximum matching in the constructed bipartite graph. In each iteration, at least one multi-source IoT application is admitted, therefore, there are at most $|\mathcal{Q}|$ iterations in the proposed algorithm. The time complexity of Algorithm 3 thus is $O(|\mathcal{Q}| \cdot (|\mathcal{Q}|+|V|)^3)$. □

5 Online Algorithm for the Online Throughput Maximization Problem

In this section, we deal with dynamic admissions of multi-source IoT application requests. We assume that IoT application requests arrive into the system one by one without the knowledge of future arrivals. For each incoming request, we need to determine whether the request is admitted or rejected immediately. If it is admitted, where is its service home? We will study the throughput maximization problem of service provisioning for dynamic multi-source IoT application request admissions. We first formulate an Integer Linear Programming (ILP) solution to an offline version of the problem that all requests in the monitoring period are given in advance. We then devise an online algorithm for the problem with a guaranteed competitive ratio.

5.1 Integer Linear Programming

We here formulate an ILP solution to the offline version of the online throughput maximization problem, where all multi-source IoT applications Q are given in advance. Recall that $x_{k,v}$ is a binary decision variable, where $x_{k,v}=1$ indicates that cloudlet v is identified as the service home of G_k, and $x_{k,v}=0$ otherwise. The ILP solution to the offline version of the problem is then formulated as follows:

$$\text{Maximize} \quad \sum_{k=1}^{|\mathcal{Q}|}\sum_{v\in V} x_{k,v}, \tag{21}$$

subject to:

$$\text{Eq. } (1),\ (2),\ (4)$$

$$\sum_{v\in V} x_{k,v} \le 1, \quad \forall G_k \in \mathcal{Q}, \tag{22}$$

$$x_{k,v}\in\{0,1\}, \quad \forall G_k\in\mathcal{Q},\ \forall v\in V, \tag{23}$$

where Constraint (22) ensures that each multi-source IoT application G_k is either rejected (i.e., $\sum_{v\in V} x_{k,v}=0$), or admitted by identifying its service home (cloudlet) in the MEC network (i.e., $\sum_{v\in V} x_{k,v}=1$). Constraint (23) shows that $x_{k,v}$ is a binary decision variable indicating whether cloudlet v is identified as the service home of G_k.

5.2 Online Algorithm

To devise an online algorithm with a guaranteed competitive ratio for the online throughput maximization problem, we here adopt an admission control policy. Denote by $C_v(k)$ the residual computing capacity of cloudlet $v\in V$ when IoT application G_k arrives. Initially, $C_v(1)$ is the computing capacity of cloudlet v, i.e., $C_v(1)=C_v$. If cloudlet v has been identified as the service home of G_k, then $C_v(k+1)=C_v(k)-R_k$.

Given the dynamics of user request arrivals and the network resource utilization, there is a need of a cost model to capture the dynamic consumption of computing resource in the network to better guide future request admissions. Intuitively, overloaded resources usually have higher probabilities to be violated by the resource demands of currently admitted requests. This eventually will affect the admissions of future requests (being occurred by currently running requests). To encourage utilizing the underloaded resources while restricting the use of overloaded resources, an exponential function is adopted to model the usage cost of computing resource $\omega_v(k)$ in cloudlets as follows:

$$\omega_v(k) = C_v(\alpha^{1-\frac{C_v(k)}{C_v}}-1), \tag{24}$$

where $1-\frac{C_v(k)}{C_v}$ is the computing resource utilization ratio of cloudlet v, α is a turning parameter demonstrating the sensitivity of the computing resource usage ratio at the cloudlet v with $\alpha>1$, and the value of α will be determined later.

The *normalized usage cost* $\psi_v(k)$ of cloudlet v as the service home of multi-source IoT application G_k then is

$$\psi_v(k)=\frac{\omega_v(k)}{C_v}=\alpha^{1-\frac{C_v(k)}{C_v}}-1. \tag{25}$$

When IoT application G_k arrives, we first identify the set of candidate cloudlets with sufficient residual computing capacity for G_k. If no such a cloudlet is available, the IoT application will be rejected. Otherwise, the candidate cloudlet with the minimum normalized usage cost by Eq. (25) will be identified. If there are multiple such candidates, only the candidate cloudlet with the least unit cost of computing resource is chosen.

Algorithm 4. Online Algorithm for the Online Throughput Maximization Problem

Input: An MEC network $G=(V,E)$ and a sequence of multi-source IoT applications arriving one by one without the knowledge of future arrivals.
Output: Maximize the number of multi-source IoT application requests admitted for a given time horizon T.

```
1:  $\mathbb{A} \leftarrow \emptyset$; /* the solution */
2:  while a multi-source IoT application request $G_k$ arrives do
3:      $\mathcal{U}_k \leftarrow \emptyset$; /* the set of candidate cloudlets for $G_k$ */
4:      for each cloudlet $v\in V$ with $C_v \ge R_k$ do
5:          $\mathcal{U}_k \leftarrow \mathcal{U}_k \cup \{v\}$;
6:          Calculate the normalized usage cost $\psi_v(k)$ of cloudlet $v$
            by Eq. (25);
7:      end for ;
8:      if $\mathcal{U}_k$ is empty then
9:          Reject $G_k$;
10:     else
11:         Identify the cloudlet $v'$ from $\mathcal{U}_k$ with the minimum nor-
            malized usage cost $\psi_{v'}(k)$, by Eq. (25);
12:         if multiple candidate cloudlets have the same mini-
            mum normalized usage cost then
13:             Identify the candidate cloudlet $v'$ with the least unit
                cost of computing resource from them;
14:         end if
15:         if $\psi_{v'}(k) > |V|$ then
16:             Reject $G_k$;
17:         else
18:             Admit $G_k$;
19:             $\mathbb{A} \leftarrow \mathbb{A} \cup \{G_k\}$;
20:         end if ;
21:     end if ;
22: end while
23: return Solution $\mathbb{A}$ to the problem.
```

Assume that cloudlet $v\in V$ has been identified as the service home of multi-source IoT application G_k. If the computing resource demand of G_k is quite large, it still can be rejected. We thus propose an admission control policy to examine the admission of each request G_k. That is, if the normalized usage cost of cloudlet v is greater than a given threshold, i.e., $\psi_v(k) > |V|$, request G_k will be rejected.

The detailed online algorithm for the online throughput maximization problem is given in `Algorithm` 4.

5.3 Analysis of the Online Algorithm

The rest is to analyze the competitive ratio and time complexity of Algorithm 4 by the following lemmas and theorem.

Lemma 1. *Given an MEC network $G = (V, E)$ and a time horizon T, a sequence of multi-source IoT application requests arrives one by one without the knowledge of future arrivals. Denote by $\mathcal{A}(k)$ the set of multi-source IoT applications admitted by* Algorithm *4 prior to the arrival of multi-source IoT application G_k. Then, the sum of the resource usage cost of all cloudlets is*

$$\sum_{v\in V}\omega_v(k) \le 2\cdot|V|\cdot\log_2\alpha\cdot\sum_{G_{k'}\in\mathcal{A}(k)}R_{k'}. \tag{26}$$

Proof. If $G_{k'}$ is rejected, the usage cost of all cloudlets do not change. Otherwise ($G_{k'}$ is admitted and allocated to cloudlet v'), we have $C_{v'}(k'+1) = C_{v'}(k') - R_{k'}$. Then

$$\begin{aligned}
&\omega_{v'}(k'+1) - \omega_{v'}(k')\\
&= C_{v'}\cdot\left(\alpha^{1-\frac{C_{v'}(k'+1)}{C_{v'}}} - 1\right) - C_{v'}\cdot\left(\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}} - 1\right)\\
&= C_{v'}\cdot\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}}\cdot\left(\alpha^{\frac{C_{v'}(k')-C_{v'}(k'+1)}{C_{v'}}} - 1\right)\\
&\le C_{v'}\cdot\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}}\cdot\left(\alpha^{\frac{R_{k'}}{C_{v'}}} - 1\right)\\
&= C_{v'}\cdot\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}}\cdot\left(2^{\frac{R_{k'}}{C_{v'}}\cdot\log_2\alpha} - 1\right)\\
&\le C_{v'}\cdot\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}}\cdot\frac{R_{k'}}{C_{v'}}\cdot\log_2\alpha. &(27)\\
&= R_{k'}\cdot\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}}\cdot\log_2\alpha, &(28)
\end{aligned}$$

where Ineq. (27) holds because $2^x - 1 \le x$ with $0 \le x \le 1$.

After allocating multi-source IoT application $G_{k'}$ to cloudlet v', the usage cost of cloudlet v' changes, while the usage costs of other cloudlets do not change at all. Thus, the difference in the sums of the usage costs of all cloudlets before and after admitting $G_{k'}$ is

$$\begin{aligned}
&\sum_{v\in V}(\omega_v(k'+1) - \omega_v(k')) = \omega_{v'}(k'+1) - \omega_{v'}(k')\\
&\le R_{k'}\cdot\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}}\cdot\log_2\alpha, \text{ by Ineq. (28)}\\
&= \log_2\alpha\cdot R_{k'}\cdot((\alpha^{1-\frac{C_{v'}(k')}{C_{v'}}} - 1) + 1)\\
&= \log_2\alpha\cdot R_{k'}\cdot(\psi_{v'}(k') + 1), \text{ by Eq. (25)}\\
&\le \log_2\alpha\cdot R_{k'}\cdot(|V| + 1) &(29)\\
&\le 2\cdot\log_2\alpha\cdot|V|\cdot R_{k'}, &(30)
\end{aligned}$$

where Ineq. (29) holds because IoT application $G_{k'}$ is admitted by the admission control policy.

The sum of usage costs of all cloudlets prior to the arrival of request G_k thus is

$$\begin{aligned}
\sum_{v\in V}\omega_v(k) &= \sum_{k'=1}^{k-1}\sum_{v\in V}(\omega_v(k'+1) - \omega_v(k'))\\
&= \sum_{G_{k'}\in\mathcal{A}(k)}\sum_{v\in V}(\omega_v(k'+1) - \omega_v(k'))\\
&\le \sum_{G_{k'}\in\mathcal{A}(k)}(2\cdot\log_2\alpha\cdot|V|\cdot R_{k'}), \text{ by Ineq. (30)}\\
&= 2\cdot|V|\cdot\log_2\alpha\cdot\sum_{G_{k'}\in\mathcal{A}(k)}R_{k'}. &(31)
\end{aligned}$$

Hence, the lemma follows. □

Lemma 2. *Given an MEC network $G = (V, E)$ and a time horizon T, a sequence of multi-source IoT application requests arrives one by one without the knowledge of future arrivals. Denote by $\mathcal{B}(k)$ the set of multi-source IoT application requests admitted by the optimal solution but rejected by* Algorithm *4 prior to the arrival of request G_k. Denote by $v^*_{k'}$ the cloudlet in the optimal solution to which multi-source IoT application $G_{k'} \in \mathcal{B}(k)$ is allocated. Then, for multi-source IoT application $G_{k'} \in \mathcal{B}(k)$*

$$\psi_{v^*_{k'}}(k') > |V|, \tag{32}$$

when $2|V| + 2 \le \alpha \le 2^{\frac{C_{min}}{R_{max}}}$, $C_{min} = \min_{v\in V}\{C_v\}$ is the minimum computing resource capacity of a cloudlet, $R_{max} = \max_{G_k\in\mathcal{Q}}\{R_k\}$ is the maximum computing resource demand of a multi-source IoT application, and $\mathcal{Q}$ is the set of requests arrived in time horizon T.

Proof. We prove the lemma by distinguishing the rejection of IoT application $G_{k'}$ into two cases: Case 1. There is no sufficient residual computing resource in cloudlet $v^*_{k'}$ for IoT application $G_{k'}$ by Algorithm 4. Case 2. There is sufficient residual computing resource in cloudlet $v^*_{k'}$ for IoT application $G_{k'}$ in Algorithm 4, and multi-source IoT application $G_{k'}$ is allocated to cloudlet $v_{k'}$ ($v_{k'}$ could be $v^*_{k'}$). However, the admission control policy in Algorithm 4 is not met.

Case 1. There is no sufficient residual computing resource in cloudlet $v^*_{k'}$ for $G_{k'}$ by Algorithm 4, i.e., $C_{v^*_{k'}}(k') < R_{k'}$. Then

$$\begin{aligned}
\psi_{v^*_{k'}}(k') &= \alpha^{1-\frac{C_{v^*_{k'}}(k')}{C_{v^*_{k'}}}} - 1 > \alpha^{1-\frac{R_{k'}}{C_{v^*_{k'}}}} - 1\\
&\ge \alpha^{1-\frac{1}{\log_2\alpha}} - 1, \quad \text{by } \alpha{\le}2\ \text{CminRmax} \le 2\ \text{Cvk}'\text{*Rk}'\\
&= \frac{\alpha}{2} - 1 \ge |V|, \quad \text{by } \alpha{\ge}2|\text{V}| + 2. &(33)
\end{aligned}$$

Case 2. As multi-source IoT application $G_{k'}$ is assigned to $v_{k'}$ with the minimum normalized usage cost by Eq. (25) in Algorithm 4, then

$$\psi_{v^*_{k'}}(k') \ge \psi_{v_{k'}}(k'), \tag{34}$$

where $v^*_{k'}$ is the cloudlet in the optimal solution to which multi-source IoT application $G_{k'} \in \mathcal{B}(i)$ is allocated.

Because the admission control policy is violated if cloudlet $v_{k'}$ is identified as the service home of $G_{k'}$, i.e., $\psi_{v_{k'}}(k') > |V|$, we have

$$\psi_{v^*_{k'}}(k') > |V|. \tag{35}$$

Hence, the lemma follows. □

Theorem 6. *Given an MEC network $G = (V, E)$, a time horizon T, and a sequence of multi-source IoT application requests arriving one by one without the knowledge of future arrivals, there is an online algorithm for the online throughput maximization problem,* Algorithm *4, with a competitive ratio of $O(R_{max}\cdot\log|V|)$, which takes $O(|V|)$ time to admit each IoT application request when $\alpha = 2|V| + 2$, where V is the set of* *cloudlets in G and R_{max} the maximum computing resource demand by a single request.*

Proof. Denote by $\mathcal{P}_{opt}(k)$ the set of admitted multi-source IoT application requests by the optimal solution prior to the arrival of IoT application G_k. Recall that $\mathcal{A}(k)$ is the set of multi-source IoT applications admitted by `Algorithm` 4 prior to the arrival of G_k, and $\mathcal{B}(k)$ is the set of IoT applications admitted by the optimal solution but rejected by `Algorithm` 4 prior to the arrival of G_k. Denote by $\mathcal{B}(k,v)$ the set of IoT applications in $\mathcal{B}(k)$ which are assigned to cloudlet v by the optimal solution, i.e., $\mathcal{B}(k,v)=\{G_{k'} \mid G_{k'}\in\mathcal{B}(k), v^*_{k'}=v\}$. It can be seen that $\mathcal{B}(k)=\cup_{v\in V}\mathcal{B}(k,v)$. We then have

$$
\begin{aligned}
|V|\cdot(|\mathcal{P}_{opt}(k)|-|\mathcal{A}(k)|) &\le |V|\cdot|\mathcal{B}(k)| = \sum_{G_{k'}\in\mathcal{B}(k)}|V| \\
&< \sum_{G_{k'}\in\mathcal{B}(k)}\psi_{v^*_{k'}}(k'), \quad \text{by Lemma } 2 \\
&\le \sum_{G_{k'}\in\mathcal{B}(k)}\psi_{v^*_{k'}}(k) \qquad (36)\\
&= \sum_{G_{k'}\in\mathcal{B}(k)}\frac{\omega_{v^*_{k'}}(k)}{C_{v^*_{k'}}} = \sum_{v\in V}\sum_{G_{k'}\in\mathcal{B}(k,v)}\frac{\omega_v(k)}{C_v} \qquad (37)\\
&\le \sum_{v\in V}\omega_v(k)\sum_{G_{k'}\in\mathcal{B}(k,v)}\frac{1}{C_v} \qquad (38)\\
&\le \sum_{v\in V}\omega_v(k) \qquad (39)\\
&\le 2\cdot|V|\cdot\log_2\alpha\cdot\sum_{G_{k'}\in\mathcal{A}(k)}R_{k'}, \quad \text{by Lemma } 1\\
&\le 2\cdot|V|\cdot\log_2\alpha\cdot|\mathcal{A}(k)|\cdot R_{max}. \qquad (40)
\end{aligned}
$$

Ineq. (36) holds because the computing resource utilization of a cloudlet does not decrease. Eq. (37) holds as $\mathcal{B}(k)=\cup_{v\in V}\mathcal{B}(k,v)$. Ineq. (38) holds because $\sum_{i=1}^{p}\sum_{j=1}^{q}A_iB_j\le\sum_{i=1}^{p}A_i\sum_{j=1}^{q}B_j$, with $A_i\ge 0$ and $B_j\ge 0$. Ineq. (39) holds as the computing resource consumption of each cloudlet is no greater than its capacity, and the computing resource consumption $R_{k'}$ of each IoT application $G_{k'}$ is assumed to be no less than 1, i.e., $R_{k'}\ge 1$, therefore, each cloudlet v is able to accommodate at most $\lfloor C_v\rfloor$ multi-source IoT applications. And Ineq. (40) holds because R_{max} is the maximum computing resource demand of a multi-source IoT application. Thus

$$|\mathcal{P}_{opt}(k)|-|\mathcal{A}(k)| < 2\cdot\log_2\alpha\cdot|\mathcal{A}(k)|\cdot R_{max}. \qquad (41)$$

Then

$$
\begin{aligned}
\frac{|\mathcal{P}_{opt}(k)|}{|\mathcal{A}(k)|} &= \frac{|\mathcal{P}_{opt}(k)|-|\mathcal{A}(k)|}{|\mathcal{A}(k)|}+1\\
&< \frac{2\cdot\log_2\alpha\cdot|\mathcal{A}(k)|\cdot R_{max}}{|\mathcal{A}(k)|}+1\\
&= 2\cdot\log_2\alpha\cdot R_{max}+1\\
&= O(R_{max}\cdot\log|V|), \quad \text{when } \alpha = 2|\mathrm{V}|+2.
\end{aligned}
$$

To examine the admission of a multi-source IoT application G_k, the time complexity of `Algorithm` 4 is dominated by calculating the normalized usage cost of all cloudlets with enough residual computing resource for G_k, which is $O(|V|)$ time. Hence, the theorem follows. □

6 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithms for the service operational cost minimization problem and the online throughput maximization problem, respectively. We also investigate the impact of important parameters on the performance of the proposed algorithms.

6.1 Environment Settings

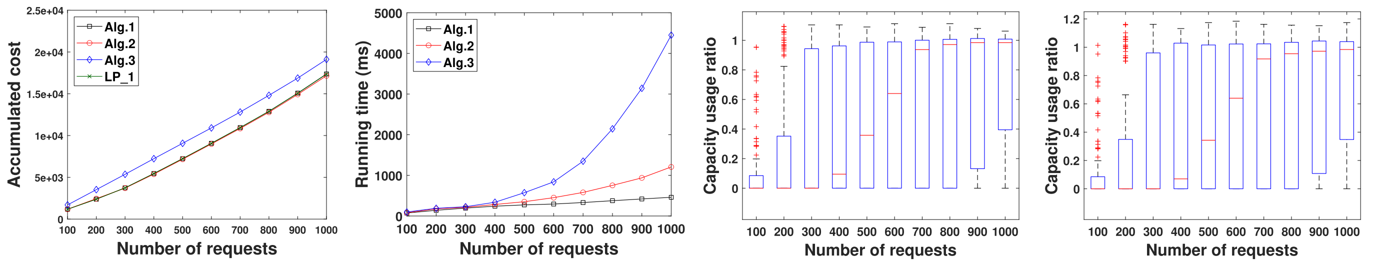
We consider an MEC network $G=(V,E)$ consisting of 100 APs, and each AP is co-located with a cloudlet. The topologies of the MEC networks are generated by GT-ITM [7]. The capacity of each cloudlet is set from 5,000 MHz to 15,000 MHz [30]. To generate a multi-source IoT application request, a set of APs is randomly selected as the source set of the request, and each source is a sink of a sensor subnetwork, while the number of sources in the source set is randomly drawn between 4 and 8 [34], and a rate of data stream of each source ranges from 1 Mbp to 5 Mbps [34]. We assume that the uploaded data streams from all sources of a multi-source IoT application request G_k are aggregated at its service home with a compression ratio λ_k randomly drawn from 0.1 to 0.5 [27], i.e., the volume of the aggregated data stream $\rho_k=\lambda_k\cdot\sum_{l=1}^{l_k}\rho_{k,l}$. The number of required CPU cycles to compute 1 bit task at its service home is randomly drawn from the interval [100, 300] cycles/bit [33]. The communication cost of each link ranges from \$0.1 to \$0.4 per MB [18], [19]. The computing resource consumption cost is randomly drawn from \$0.01 to \$0.03 per MHz [30]. The value in each figure is the mean of the results out of 30 MEC instances with the same size. The actual running time of each algorithm is obtained on a desktop with a 3.60 GHz Intel 8-Core i7 CPU and 16 GB RAM. Unless specified, these parameters are adopted by default.

We refer to the proposed algorithms `Algorithms` 1, 2 and 3 for the service operational cost minimization problem as `Alg.1`, `Alg.2` and `Alg.3`, respectively. To evaluate the performance of algorithms `Alg.1`, `Alg.2` and `Alg.3` for the service operational cost minimization problem, we refer to the Linear Programming solution (3) to the service operational cost minimization problem as `LP_1`, which is a lower bound on the optimal solution of the problem.

Similarly, we refer to the proposed algorithm `Algorithm` 4 for the online throughput maximization problem as `Alg.4`. To evaluate the performance of algorithm `Alg.4` for the online throughput maximization problem, we consider the Linear Programming solution (21) of the offline version of the problem as `LP_2`, which is an upper bound on the optimal solution of the problem. We also consider a comparison greedy algorithm `GDY` that admits each incoming request by allocating it to a cloudlet with sufficient computing resource randomly. An incoming request is rejected if there is no cloudlet with sufficient computing resource for its admission.

6.2 Algorithm Performance Evaluation for the Service Operational Cost Minimization Problem

We first studied the performance of algorithms `Alg.1`, `Alg.2` and `Alg.3` for the service operational cost minimization



(a) The total cost (b) The running time (c) The capacity usage ratios of cloudlets by algorithm `Alg.1` (d) The capacity usage ratios of cloudlets by algorithm `Alg.2`

Fig. 2. Algorithm performance for the service operational cost minimization problem by varying the number of requests.

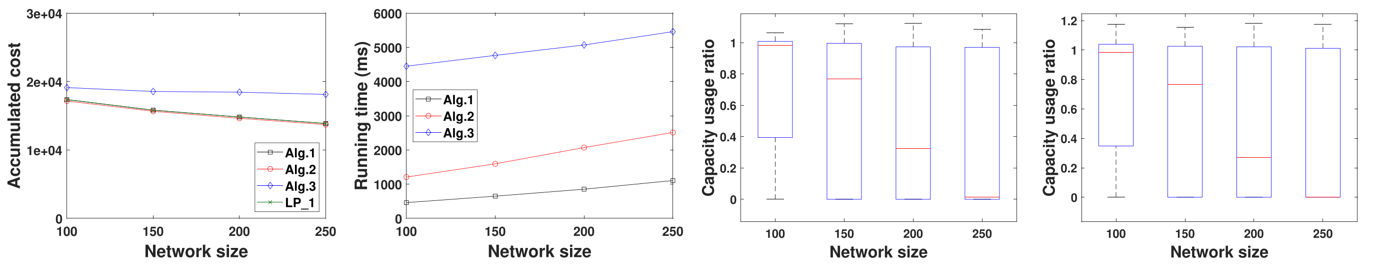
problem against algorithm `LP_1`, by varying the number of requests from 100 to 1,000. To characterize the capacity violations by algorithm `Alg.1` and `Alg.2`, we define the capacity usage ratio of a cloudlet v as $\frac{W_v}{C_v}$, where W_v is the consumed computing resource of cloudlet v, and C_v is the computing capacity of cloudlet v. Fig. 2 depicts the accumulated operational cost and running time of different algorithms, and the capacity usage ratios of cloudlets by algorithm `Alg.1` and `Alg.2`, respectively. It can be seen from Fig. 2a that the performance of algorithms `Alg.1` and `Alg.2` are similar to that of the algorithm `LP_1`, and when the number of requests reaches 1,000, the accumulated operational cost of algorithm `Alg.3` is 10.2% higher than that of algorithm `LP_1`. Fig. 2b demonstrates the running time curves of algorithms `Alg.1`, `Alg.2` and `Alg.3`, respectively. From Fig. 2b, it can be seen that algorithm `Alg.1` consumes the least running time while `Alg.3` consumes the most running time. Note that the capacities of cloudlets could be violated by adopting algorithms `Alg.1` and `Alg.2`. Figs. 2c and 2d plot the minimum, the average, and the maximum capacity usage ratios of cloudlets by algorithms `Alg.1` and `Alg.2`, respectively. From Fig. 2c, the computing capacity of each cloudlet is violated by no more than 11.2% of its actual capacity by adopting algorithm `Alg.1`. In Fig. 2d, the computing capacity of each cloudlet is violated by no more than 18.4% of its actual capacity by algorithm `Alg.2`. Thus, it can be seen from Fig. 2 that the performance of the proposed algorithms `Alg.1`, `Alg.2` and `Alg.3` are promising, compared with the lower bound of the original problem solution - the solution `LP_1`. The rationale behind this is that the proposed algorithms `Alg.1`, `Alg.2` and `Alg.3` are able to efficiently identify a service home for each IoT application request to minimize the total service operational cost.

We then evaluated the impact of network size on the performance of algorithms `Alg.1`, `Alg.2`, and `Alg.3`, by varying network size from 100 to 250, while fixing the number of multi-source IoT application requests at 1,000. Fig. 3 plots the accumulated operational costs, the running time of different algorithms, and the capacity usage ratios of different cloudlets by algorithm `Alg.1` and `Alg.2`, respectively. It can be seen from Fig. 3a that the accumulated operational cost by algorithm `Alg.1` when the network size is 250 is 79.8% of that by itself when the network size is 100, while the accumulated operational cost by algorithm `Alg.2` when the network size is 250 is 80.1% of that by itself when the network size 100. Also, the accumulated operational cost by algorithm `Alg.3` when the network size is 250 is 94.7% of that by itself when the network size 100. This is because there will be more computing resource in an MEC network with a large network size. For each multi-source IoT application request G_k, algorithms `Alg.1`, `Alg.2` and `Alg.3` can identify the cloudlet as the service home of G_k with relatively cheap computing cost and communication cost from its sources to its destination.

6.3 Algorithm Performance Evaluation for the Online Throughput Maximization Problem

We first investigated the performance of algorithm `Alg.4` against algorithms `GDY` and `LP_2`, by varying network size from 50 to 250, while fixing the number of incoming requests at 10,000. Fig. 4 depicts the number of admitted IoT application requests and the running time of different algorithms. It can be seen from Fig. 4a that when the network size reaches 250, algorithm `Alg.4` admits 19.2% more IoT application requests than that by algorithm `GDY`, while the performance of algorithm `Alg.4` is 83.1% of the upper bound of the original problem solution - the solution `LP_2`. This is because algorithm `Alg.4` adopts an efficient admission control policy to admit incoming requests.

We then studied the impact of the number of sources l_k per multi-source IoT application on the performance of



(a) The total cost (b) The running time (c) The capacity usage ratios of cloudlets by algorithm `Alg.1` (d) The capacity usage ratios of cloudlets by algorithm `Alg.2`

Fig. 3. Impact of network size on the algorithm performance for the service operational cost minimization problem.

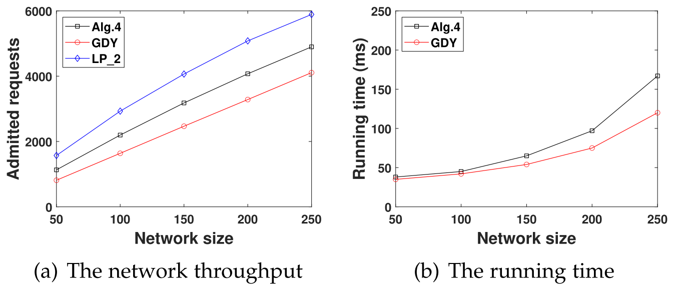


Fig. 4. Performance of different algorithms for the online throughput maximization problem, by varying the network size.

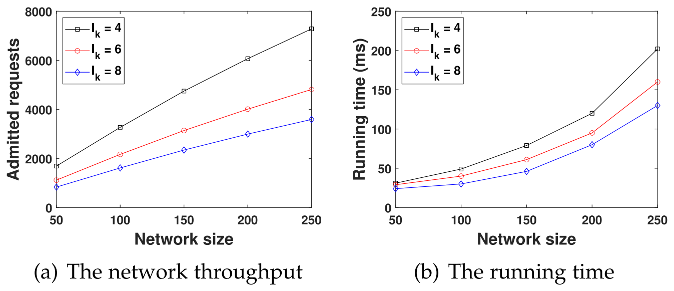


Fig. 5. Impact of the number of sources l_k on the performance of algorithm Alg.4 for the online throughput maximization problem.

algorithm Alg.4 for the online throughput maximization problem when $l_k = 4$, 6 and 8, by varying the network size from 50 to 250, while fixing the number of incoming requests at 10,000. Fig. 5 depicts the number of admitted requests and the running time of algorithm Alg.4. It can be seen from Fig. 5a that the performance achieved by algorithm Alg.4 when $l_k = 4$ is 49.2% of that by itself when $l_k = 8$. This can be justified that a smaller value of l_k leads to smaller computing resource consumption, and more requests can be admitted. It can be seen from Fig. 5b that algorithm Alg.4 with $l_k = 4$ consumes the most running time, due to the fact that it deals with the admission of the most requests.

7 Conclusion

In this paper, we studied the service home identification problem in edge computing for multi-source IoT applications. We first formulated the service operational cost minimization problem, and devised randomized and deterministic approximation algorithm for the problem with moderate resource violations. We then devised an efficient heuristic algorithm for the problem without resource violation. We also investigated the online throughput maximization problem by dynamically admitting requests of multi-source IoT applications, for which we proposed an online algorithm with a provable competitive ratio. We finally evaluated the performance of the proposed algorithms through experimental simulations. Experimental results demonstrated that the proposed algorithms are promising.

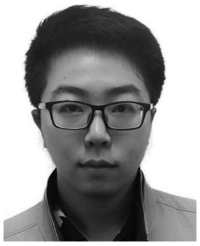
Acknowledgments

We appreciate the three anonymous referees for their constructive comments and valuable suggestions, which helped us improve the quality and presentation of the article greatly. Particularly we would like to thank one referee for the inapproximability proof of Theorem 1.

References

[1] C. Avasalcai, C. Tsigkanos, and S. Dustdar, "Resource management for latency-sensitive IoT applications with satisfiability," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3074188.
[2] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," *J. Netw. Comput. Appl.*, vol. 89, pp. 96–108, 2017.
[3] S. Basu *et al.*, "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment," *Future Gener. Comput. Syst.*, vol. 88, pp. 254–261, 2018.
[4] H. Cai, B. Xu, L. Jiang, and A. V. Vasilakos, "IoT-based big data storage systems in cloud computing: Perspectives and challenges," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 75–87, Feb. 2017.
[5] I. A. Elgendy, W. -Z. Zhang, Y. Zeng, H. He, Y. -C. Tian, and Y. Yang, "Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 4, pp. 2410–2422, Dec. 2020.
[6] M. Goudarzi, H. Wu, M. Palaniswami, and R. Buyya, "An application placement technique for concurrent IoT applications in edge and fog computing environments," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1298–1311, Apr. 2021.
[7] GT-ITM, 2019. [Online]. Available: https://www.cc.gatech.edu/projects/
[8] S. Hu and G. Li, "Dynamic request scheduling optimization in mobile edge computing for IoT applications," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1426–1437, Feb. 2020.
[9] D. Kimovski, N. Mehran, C. E. Kerth, and R. Prodan, "Mobility-aware IoT applications placement in the cloud edge continuum," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3094322.
[10] K. Lei, M. Du, J. Huang, and T. Jin, "Groupchain: Towards a scalable public blockchain in fog computing of IoT services computing," *IEEE Trans. Serv. Comput.*, vol. 13, no. 2, pp. 252–262, Mar./Apr. 2020.
[11] J. Li, W. Liang, W. Xu, Z. Xu, and J. Zhao, "Maximizing the quality of user experience of using services in edge computing for delay-sensitive IoT applications," in *Proc. 23rd Int. ACM Conf. Model. Anal. Simul. Wireless Mobile Syst.*, 2020, pp. 113–121.
[12] J. Li *et al.*, "Maximizing user service satisfaction for delay-sensitive IoT applications in edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 5, pp. 1199–1212, May 2022.
[13] J. Li, W. Liang, Z. Xu, and W. Zhou, "Provisioning virtual services in mobile edge computing for IoT applications with multiple sources," in *Proc. IEEE 45th Conf. Local Comput. Netw.*, 2020, pp. 42–53.
[14] J. Li, W. Liang, Z. Xu, X. Jia, and W. Zhou, "Service provisioning for multi-source IoT applications in mobile edge computing," *ACM Trans. Sensor Netw.*, vol. 18, no. 2, pp. 17:1–17:25, May 2022.
[15] W. Liang, Y. Ma, W. Xu, Z. Xu, X. Jia, and W. Zhou, "Request reliability augmentation with service function chain requirements in mobile edge computing," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2021.3081681.
[16] W. Liang, Y. Ma, W. Xu, X. Jia, and S. Chau, "Reliability augmentation of requests with service function chain requirements in mobile edge-cloud networks," in *Proc 49th Int. Conf. Parallel Process.*, 2020, Art. no. 74.
[17] Y. Liu, Y. Li, Y. Niu, and D. Jin, "Joint optimization of path planning and resource allocation in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 9, pp. 2129–2144, Sep. 2020.
[18] Y. Ma, W. Liang, and J. Wu, "Online NFV-enabled multicasting in mobile edge cloud networks," in *Proc. 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 821–830.
[19] Y. Ma, W. Liang, J. Wu, and Z. Xu, "Throughput maximization of NFV-enabled multicasting in mobile edge cloud networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 2, pp. 393–407, Feb. 2020.
[20] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

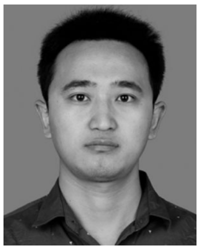
[21] R. M. Nauss, "Solving the generalized assignment problem: An optimizing and heuristic approach," *Informs J. Comput.*, vol. 15, no. 3, pp. 249–266, 2003.
[22] P. Raghavan and C. D. Thompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.
[23] H. Sami, A. Mourad, H. Otrok, and J. Bentahar, "Demand-driven deep reinforcement learning for scalable fog and service placement," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3075988.
[24] D. Shomys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, pp. 461–474, 1993.
[25] Y. Song, S. S. Yau, R. Yu, X. Zhang, and G. Xue, "An approach to QoS-based task distribution in edge computing networks for IoT applications," in *Proc. Int. Conf. Edge Comput.*, 2017, pp. 32–39.
[26] X. Sun and N. Ansari, "Dynamic resource caching in the IoT application layer for smart cities," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 606–613, Apr. 2018.
[27] P. Wang, C. Yao, Z. Zheng, G. Sun, and L. Song, "Joint task assignment, transmission, and computing resource allocation in multilayer mobile edge computing systems," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2872–2884, Apr. 2019.
[28] Q. Xia, W. Ren, Z. Xu, X. Wang, and W. Liang, "When edge caching meets a budget: Near optimal service delivery in multi-tiered edge clouds," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3091462.
[29] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in IoT edge computing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1133–1146, Jun. 2020.
[30] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis, "Efficient NFV-enabled multicasting in SDNs," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2052–2070, Mar. 2019.
[31] Z. Xu, W. Gong, Q. Xia, W. Liang, O. F. Rana, and G. Wu, "NFV-enabled IoT service provisioning in mobile edge clouds," *IEEE Trans. Mobile Comput.*, vol. 20, no. 5, pp. 1892–1906, May 2021.
[32] Z. Xu *et al.*, "Schedule or wait: Age-minimization for IoT big data processing in MEC via online learning," in *Proc IEEE Int. Conf. Comput. Commun.*, 2022.
[33] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.
[34] R. Yu, G. Xue, and X. Zhang, "Application provisioning in FOG computing-enabled Internet-of-Things: A network perspective," in *Proc. IEEE Conf. Comput. Commun.*, 2018, pp. 783–791.



Jing Li received the BSc (first class honours) degree in computer science from the Australian National University, in 2018. He is currently working toward the PhD degree in the School of Computing, Australian National University. His research interests include mobile edge computing, network function virtualization, and combinatorial optimization.

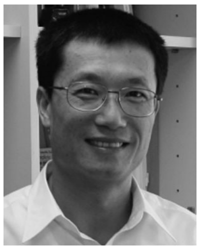
Weifa Liang (Senior Member, IEEE) received the BSc degree from Wuhan University, China, in 1984, the ME degree from the University of Science and Technology of China, in 1989, and the PhD degree from the Australian National University, in 1998, all in computer science. He is a professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he was a professor with the Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, mobile edge computing (MEC), network function virtualization (NFV), Internet of Things, software-defined networking (SDN), design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He currently serves as an associate editor in the editorial board of the *IEEE Transactions on Communications*.

Wenzheng Xu (Member, IEEE) received the BSc, ME, and PhD degrees in computer science from Sun Yat-Sen University, Guangzhou, China, in 2008, 2010, and 2015, respectively. He is currently an Associate Professor with Sichuan University, China. Also, he was a visitor at both the Australian National University, Australia and the Chinese University of Hong Kong, Hong Kong. His research interests include wireless ad hoc and sensor networks, mobile computing, approximation algorithms, combinatorial optimization, online social networks, and graph theory.



Zichuan Xu (Member, IEEE) received the BSc and ME degrees in computer science from the Dalian University of Technology, China, in 2008 and 2011, respectively, and the PhD degree in computer science from the Australian National University, in 2016. From 2016 to 2017, he was a research associate with the Department of Electronic and Electrical Engineering, University College London, U.K. He is currently a professor with the School of Software, Dalian University of Technology. He is also a "Xinghai Scholar' in Dalian University of Technology. His research interests include mobile edge computing, cloud computing, network function virtualization, software-defined networking, Internet of Things, algorithmic game theory, and optimization problems.

Yuchen Li received the BSc (first class honours) degree in computer science from the Australian National University, in 2018. He is currently working toward the PhD degree in the School of Computing, Australian National University. His research interests include the Internet of Things, mobile edge computing, and algorithm design.



Xiaohua Jia (Fellow, IEEE) received the BSc and ME degrees from the University of Science and Technology of China, in 1984 and 1987, respectively, and the DSc degree in information science from the University of Tokyo, in 1991. He is currently a chair professor with the Department of Computer Science, City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks, wireless sensor networks, and mobile wireless networks. He is an editor of the *IEEE Transactions on Parallel and Distributed Systems* (2006–2009), *Journal of World Wide Web*, *Wireless Networks*, *Journal of Combinatorial Optimization*, and so on. He is the general chair of ACM MobiHoc 2008, TPC co-chair of IEEE MASS 2009, area-chair of IEEE INFOCOM 2010, TPC co-chair of IEEE GlobeCom 2010, Ad Hoc and Sensor Networking Symposium, and Panel co-chair of IEEE INFOCOM 2011.