Service Provisioning for Multi-source IoT Applications in Mobile Edge Computing

JING LI, Australian National University, Australia
WEIFA LIANG, City University of Hong Kong, P. R. China
ZICHUAN XU, Dalian University of Technology, P. R. China
XIAOHUA JIA, City University of Hong Kong, P. R. China
WANLEI ZHOU, City University of Macau, P. R. China

We are embracing an era of Internet of Things (IoT). The latency brought by unstable wireless networks caused by limited resources of IoT devices seriously impacts the quality of services of users, particularly the service delay they experienced. Mobile Edge Computing (MEC) technology provides promising solutions to delay-sensitive IoT applications, where cloudlets (edge servers) are co-located with wireless access points in the proximity of IoT devices. The service response latency for IoT applications can be significantly shortened due to that their data processing can be performed in a local MEC network. Meanwhile, most IoT applications usually impose Service Function Chain (SFC) enforcement on their data transmission, where each data packet from its source gateway of an IoT device to the destination (a cloudlet) of the IoT application must pass through each Virtual Network Function (VNF) in the SFC in an MEC network. However, little attention has been paid on such a service provisioning of multi-source IoT applications in an MEC network with SFC enforcement.

In this article, we study service provisioning in an MEC network for multi-source IoT applications with SFC requirements and aiming at minimizing the cost of such service provisioning, where each IoT application has multiple data streams from different sources to be uploaded to a location (cloudlet) in the MEC network for aggregation, processing, and storage purposes. To this end, we first formulate two novel optimization problems: the cost minimization problem of service provisioning for a single multi-source IoT application, and the service provisioning problem for a set of multi-source IoT applications, respectively, and show that both problems are NP-hard. Second, we propose a service provisioning framework in the MEC network for multi-source IoT applications that consists of uploading stream data from multiple sources of the IoT application to the MEC network, data stream aggregation and routing through the VNF instance placement and sharing, and workload balancing among cloudlets. Third, we devise an efficient algorithm for the cost minimization problem built upon the proposed service provisioning framework, and further extend the solution for the service provisioning problem of a set of multi-source IoT applications. We finally evaluate the performance of the proposed algorithms through experimental simulations. Simulation results demonstrate that the proposed algorithms are promising.

CCS Concepts: • **Computer systems organization → Sensor networks;**

Authors' addresses: J. Li, Australian National University, Canberra, ACT 2601, Australia; email: jing.li5@anu.edu.au; W. Liang and X. Jia, City University of Hong Kong, 83 Tat Chee Ave., Kowloon, Hong Kong, P. R. China; emails: weifa.liang@cityu.edu.hk, csjia@cityu.edu.hk; Z. Xu, Dalian University of Technology, Dalian, Liaoning, 116621, P. R. China; email: z.xu@dlut.edu.cn; W. Zhou, City University of Macau, Avenida Padre Tomás Pereira Taipa, Macao, P. R. China; email: wlzhou@cityu.mo.

1550-4859/2021/10-ART17 $15.00
https://doi.org/10.1145/3484200

Additional Key Words and Phrases: Service Function Chain (SFC), Network Function Virtualization (NFV), IoT service provisioning, IoT-driven service provision framework, operational cost minimization, algorithms for IoT service provisioning, workload balancing, Mobile Edge Computing (MEC), Virtual Network Function (VNF) instance placement and sharing, data stream routing and aggregation, dynamic programming, network slicing, distributed resource allocation and optimization

ACM Reference format:
Jing Li, Weifa Liang, Zichuan Xu, Xiaohua Jia, and Wanlei Zhou. 2021. Service Provisioning for Multi-source IoT Applications in Mobile Edge Computing. *ACM Trans. Sen. Netw.* 18, 2, Article 17 (October 2021), 25 pages.
https://doi.org/10.1145/3484200

1 INTRODUCTION

The **Internet of Things (IoT)** is paving the way for many new emerging technologies, such as smart grid, Industry 4.0, connected cars, smart cities, and so on. The number of connected devices is set to increase from 700M to 3.2B by 2023 [2]. The data collected from IoT devices need to be analyzed in real time such that the hidden patterns and insights can be extracted in almost no time. Conventional IoT devices usually transmit their data to remote clouds for storage and processing [17]. Due to the remoteness of clouds and increasingly congested core networks, this incurs prohibitive long transmission delays, thereby violating the real-time data processing requirements of many IoT applications. Thus, conventional clouds may not be suitable for delay-sensitive IoT applications [15, 18, 24].

With the fast development of 5G, **Mobile Edge Computing (MEC)** promises to greatly reduce data processing delays of IoT services, by deploying computing resource (e.g., cloudlets) in the proximity of IoT devices [3, 14]. Thus, IoT devices can directly forward their streaming sensory data to virtual services instantiated in **Virtual Machines (VMs)** in cloudlets of MEC networks. A typical multi-source IoT application usually collects sensory data from multiple sources (e.g., different IoT devices) located at different geographical locations, aggregates the streaming data at some intermediate nodes (cloudlets), and ultimately routes the aggregated data stream to a destination (a cloudlet) for further processing and storage. To ensure the security and privacy of data stream routing in the MEC network, various network service functions such as firewalls, intrusion detection systems, proxies, and load balancers may also be deployed. For example, IoT applications usually need a sequence of network service functions for data aggregation and filtering to guarantee the real-time in-network processing (e.g., summation, averaging, maximum or minimum) of IoT data streams. Such a sequence of network service functions is referred to as a **Service Function Chain (SFC)**. Conventional network functions are implemented by dedicated hardware—middleboxes, making their deployment and maintenance very expensive and not agile. **Network Function Virtualization (NFV)** [7, 8, 10] as a new technology promises to provide inexpensive and flexible network services, and implements the network service functions as software in VMs or containers in cloudlets. NFV makes virtual service provisioning become affordable, easy to adopt, and maintainable.

To enable service provisioning in an MEC network for multi-source IoT applications with SFC requirements, it poses significant challenges. First, the provisioning of an IoT service requires joint considerations of many complicated data processing procedures, such as uploading sensory data streams from multiple sources through different gateways to a single destination, data stream aggregation and routing in the MEC network, and the aggregated data stream is processed by the **Virtual Network Function (VNF)** instances of network functions in the SFC. To minimize computing and bandwidth resource usages, a routing tree rooted at the destination cloudlet and

spanning each source (access point) of an IoT application is built, instead of uploading the data stream of each source to the destination separately. The uploaded data streams are aggregated at non-leaf tree nodes, and appropriate numbers of VNF instances of service functions should also be deployed at the tree nodes of data streaming data processing. Building such a routing tree for a multi-source IoT application involves non-trivial interplays between VNF instance placement, aggregation node selections, and routing path selections. However, the computing and bandwidth resources in an MEC network are usually very precious and costly. The operational cost of service provisioning for multi-source IoT applications depends on not only various MEC resource consumptions but also balancing wordloads among cloudlets. How to minimize the operational cost of service provisioning while meeting the SFC requirements of IoT applications poses another challenge. The data streams of each IoT application are allowed to be merged or aggregated at intermediate nodes of the routing tree before reaching its destination. Also, VNFs for such data merging and aggregation usually can be shared among the data streams from different gateways of the IoT application. How to further reduce the operational cost of service provisioning of IoT applications through VNF sharing is the last challenge. In the rest of this article, we will address the aforementioned challenges.

The novelties of the work in this article lie in formulating two novel service provisioning problems for multi-source IoT applications in an MEC network, where each multi-source IoT application has multiple data streams from different sources to be uploaded to the MEC network for processing and storage, while each data stream must pass through the network functions in the SFC of the IoT application prior to reaching its destination. A service provisioning framework for such multi-source IoT applications is proposed through VNF instance placement and sharing, workload balancing among cloudlets, and in-network aggregation of data streams. Efficient algorithms for service provisioning of multi-source IoT applications, built upon the proposed framework, are also proposed. To the best of our knowledge, this is the first work on service provisioning for multi-source IoT applications in MEC networks with multiple critical constraints.

The main contributions of this article are given as follows. We consider the service provisioning in an MEC network for multi-source IoT applications with SFC requirements. We first formulate two optimization problems: the cost minimization problem of service provisioning for a single multi-source IoT application and the service provisioning problem for a set of multi-source IoT applications, respectively, and show that both problems are NP-hard. We then propose a novel service provisioning framework in an MEC network for a single multi-source IoT application, which consists of multiple data streams uploading from different subnetworks through wireless access points (gateway nodes), data stream in-network aggregation and routing, VNF instance placement and sharing, and workload balancing among cloudlets in the MEC network. Due to the NP-hardness of the two defined problems, we third devise efficient heuristic algorithms for them. We finally evaluate the performance of the proposed algorithms through experimental simulations. Experimental results demonstrate that the proposed algorithms are promising, and outperform their comparison counterparts.

The rest of the article is organized as follows. Section 2 summarizes the related work on service provisioning with SFC requirements in MEC. Section 3 introduces notions, notations, and the problem definitions. Section 4 shows that the defined problems are NP-hard. Section 5 proposes a novel service provisioning framework for multi-source IoT applications with SFC requirements. Section 6 devises an algorithm for the cost minimization problem of service provisioning for a single multi-source IoT application. Section 7 extends the proposed algorithm to the service provisioning of a set of multi-source IoT applications through network slicing. Section 8 evaluates the proposed algorithms empirically, and Section 9 concludes the article.

2 RELATED WORK

There are a few investigations of service provisioning of NFV-enabled network services for IoT applications in MEC networks. For example, Song et al. [25] considered the **Quality-of-Service (QoS)**-based task allocation in MEC for IoT applications by proposing efficient algorithms, however, they did not incorporate the SFC requirement into consideration. Yu et al. [31] studied the problem of IoT service provisioning with the objective to meet computing, bandwidth and QoS requirements of an IoT application. They however did not consider the processing of IoT data traffic with SFC enforcement. Mouradian et al. [22] proposed an architecture of NFV and SDN-based distributed IoT gateways for large-scale disaster management. They however did not focus on the operational cost minimization and the workload balancing. Xu et al. [30] studied the QoS-aware VNF placement of SFCs in MEC for one-source IoT applications. They considered the operational cost minimization problem for the implementation of IoT applications with SFC requirements, and concentrated on IoT application placement in MEC by proposing randomized and heuristic algorithms. Although we also deal with the operational cost minimization problem of service provisioning for multi-source IoT applications, the problem in this study is essentially different from the mentioned work [30] as follows. We here deal with multi-source IoT applications to minimize the operational cost of service provisioning, where data streams from different sources need to be uploaded, aggregated, and processed in the MEC, while the demanded number of NFV instances of each service function in the SFC needs to be placed in cloudlets, and the workload among cloudlets needs to be balanced. We aim to construct a data routing tree for each multi-source IoT application such that the operational cost is minimized, while the work in Reference [30] considered a single SFC chain placement, where there is no data stream aggregation, data routing tree construction, and workload balancing among cloudlets.

Closely related to the problem studied in this article are the NFV-enabled unicast and multicast problems. There are extensive studies of user unicast and multicast request admissions through resource provisioning and allocations in MEC networks [4, 5, 9, 11, 12, 20, 21, 23, 26, 29, 32]. For example, Jia et al. [11] considered the assignment of user requests to different cloudlets in a wireless metropolitan area network with the aim to minimize the maximum delay among offloaded tasks, by developing heuristics for the problem. Jia et al. [12] also studied workload balancing among cloudlets to reduce the maximum response delay of user requests. Xu et al. [29] devised approximation algorithms to efficiently offload user requests to different cloudlets under different conditions. Xia et al. [27] considered opportunistic task offloading under link bandwidth, mobile device energy, and cloudlet computing capacity constraints. Ma et al. [21] considered the profit maximization problem in MEC by dynamically admitting NFV-enabled unicast requests with QoS requirements, for which they developed an efficient heuristic and an online algorithm with a provable competitive ratio if the QoS requirement can be ignored. Although they considered the sharing of existing VNF instances among different unicast requests. It can be seen that the problem of NFV-enabled unicast request admissions in Reference [21] is a special case of the NFV-enabled multicast request admissions where the destination set contains only one node. Recently, there are several studies by extending the NFV-enabled unicast routing to NFV-enabled multicast routing in MEC environments. For example, Alhussein et al. [4] investigated the embedding of a multicast request with both computing and bandwidth resource requirements to a 5G core substrate network, by enabling multipath routing between two consecutive VNFs of an SFC. Ceselli et al. [5] considered the design optimization such as the VM placement and migration, and user request assignment, by formulating a **Mixed Integer Linear Programming (MILP)** solution and heuristic algorithms for the problem. Feng et al. [9] proposed an algorithm with a performance guarantee for placing VNFs in distributed cloud networks and routing service flows among the placed VNFs under the constraints of SFCs of the requests. Xu et al. [28] considered the cost minimization of admitting

a single NFV-enabled multicast request with the QoS requirement in MEC, where the implementation of the SFC of each request is consolidated to a single cloudlet. They aimed to minimize the admission cost by placing no more than constant numbers of VNF instances of the SFC of the request in different branches of the found pseudo-multicast tree for the request. Ma et al. [19, 20] studied admissions of NFV-enabled multicasting requests under both static and dynamic admission scenarios, by proposing approximation and online algorithms for the problems with provable performance guarantees. Soni et al. [26] proposed a scalable multicast group management scheme and a workload balancing method for the routing of best-effort traffic and bandwidth-guaranteed traffic. Zhang et al. [32] investigated the NFV-enabled multicasting problem in SDNs. They assumed that there are sufficient computing and bandwidth resources to accommodate all multicast requests, for which they provided a 2-approximation algorithm if only one server is deployed for implementing the SFC of each multicast request. The applicability of their method, unfortunately, is very limited, and cannot be extended for the general case where multiple servers are employed. Ren et al. [23] also considered request admissions of NFV-enabled multicasting under the end-to-end delay constraint by developing both approximation and heuristic algorithms for the problem. However, the above studies did not tackle the service provisioning problem for multi-source IoT applications, which is much more difficult compared with the NFV-enabled multicasting problem. Although there are some similarities between the IoT service provisioning problem in this article and the NFV-enabled multicasting problem in MEC [20, 23], due to the fact that they both aim to build a routing tree for their solutions, the main difference between them lies in the following. In the multicasting case, a packet from the source (the tree root) will be broadcast to all destinations (leaves) such that a VNF instance of each service function in the SFC is installed along the path from the root to each leaf. In contrast, in the routing tree for a multi-source IoT application, all leaves are data sources. The data streams will converge to the tree root, and the data streams from the children of a non-leaf node in the tree will be aggregated at the node. The number of VNF instances instantiated at a tree node in the path from a leaf to the tree root is determined by the accumulative volume of the data streams at the node while the accumulative volumes of data streams at different tree nodes are different. Thus, the construction of such a data routing tree for each multi-source IoT application is much challenging, due to that the volume of the data stream at each tree node is not fixed, which is determined by the number of children and the data volume of each of the children. Also, the VNF instance placement must meet certain restrictions while keeping the workload among involving cloudlets to be balanced. It must be mentioned that this article is an extension of the conference paper [16], where an efficient algorithm is proposed to deal with the cost minimization problem of service provisioning for a single multi-source IoT application. In this extension, we also deal with the service provisioning problem for a set of multi-source IoT applications, and propose an efficient algorithm for it as well.

3 PRELIMINARIES

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

3.1 System Model

An MEC network is represented as 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 an **Access Point (AP)**. Associated with each AP, there is a co-located cloudlet $v \in V$ (edge cloud) with computing capacity $C_v > 0$. The AP and its co-located cloudlet are connected through a high-speed optical cable, and the communication delay between each AP and its co-located cloudlet thus is negligible. Assume that the MEC network supports a set of services in which each service is associated with a set of VNFs.

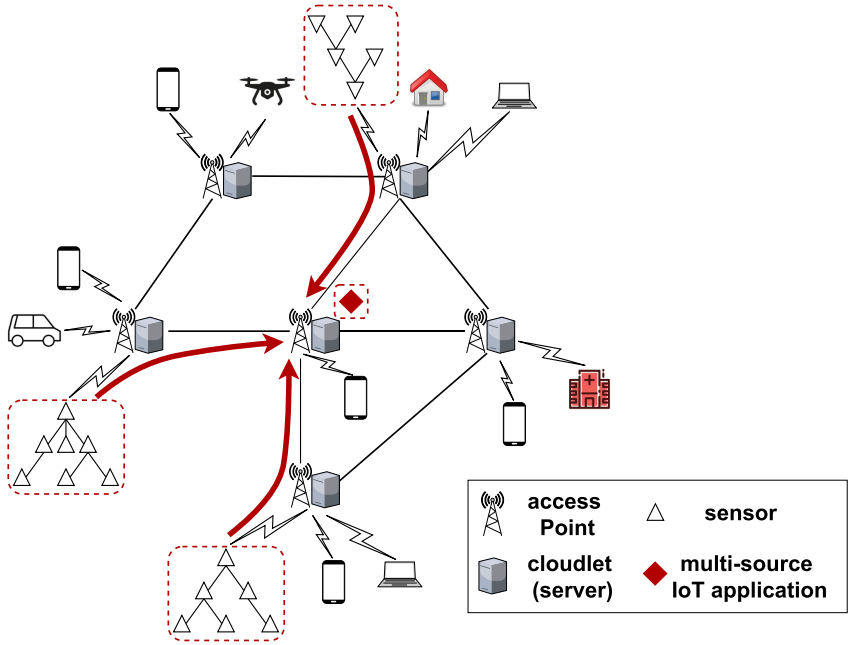


Fig. 1. An illustrative example of an MEC network that consists of six APs, and there are six cloudlets co-located with the APs. There is a destination cloudlet (service provisioning) for a multi-source IoT application, where the multi-source IoT application contains three data stream sources with each from a different sensor subnetwork of the IoT application.

Denote by $\mathcal{F} = \langle f_1, f_2, \ldots, f_{|\mathcal{F}|} \rangle$ the set of VNFs in the system, where the implementation of VNF $f_j \in \mathcal{F}$ consumes the amount c_{f_j} of computing resource of a cloudlet. One implementation of f_j is termed as one of VNF instances of the function and each such a VNF instance has a data processing capacity μ_{f_j}, where $1 \leq j \leq |\mathcal{F}|$. Figure 1 is an illustrative example of an MEC network.

3.2 Problem Definitions

Given an MEC network $G = (V, E)$, we consider a multi-source IoT application, which provides continuous surveillance to monitor different geographical areas (e.g., public parks) in a metropolitan region, and different types of IoT sensory devices are deployed in the monitoring areas. The data streams generated from these IoT devices can then be aggregated within their subnetworks and uploaded to the MEC through their gateways (APs). We further assume that there is a destination node (cloudlet) in the MEC for the aggregated data storage of each IoT application, where all updated streaming data from the gateways (APs) of the IoT application will be stored for user ad hoc queries. When the stream data from each gateway is routed to the destination node, the data stream may be merged or aggregated with the data streams from the other gateways of the application. However, each data stream from a gateway to the destination node must pass through the specified VNF instances in the SFC to ensure the security and privacy of the data stream and network performance. Thus, the VNF instances of each function in the SFC must be instantiated in the cloudlets of their routing paths and some of the VNF instances can be shared among the data streams from different gateways of the IoT application. For the sake of convenience, in the rest of our discussion, we assume that there are sufficient computing and bandwidth resources in the MEC to accommodate the VNF instances and data stream routing for the IoT application. In

summary, we aim to build a VNF network slicing for each multi-source IoT application such that the operational cost of implementing the application is minimized, in terms of the computing and communication resource consumptions, and the workload balancing among cloudlets involved. We refer to this IoT-driven optimization problem as *the cost minimization problem of service provisioning for a single multi-source IoT application in MEC*. An illustrative example of such an IoT application is given in Figure 1.

Definition 1. Given an MEC $G = (V, E)$, and a multi-source IoT application with a SFC requirement $\langle f_1, f_2, \ldots f_L \rangle$, in which there is a set $S = \{s_1, s_2, \ldots, s_K\} \subset V$ of sources and one destination $d \in V$ for the application. Each source s_i has a streaming data rate ρ_{s_i} with $1 \leq i \leq K$, *the cost minimization problem of service provisioning for a single multi-source IoT application* is to find a data routing tree T in G rooted at $d \in V$ and spanning all nodes in S, and place the demanded number of VNF instances of each f_j in the SFC with $1 \leq j \leq L$ at some nodes in T that meets the following conditions: (1) the operational cost of implementing the IoT application is minimized; (2) the data stream in the tree path from each s_i to the destination d must pass through the VNF instances of functions in the SFC in order; (3) there is a defined *data aggregation function* $g(v)$ at each non-leaf node v in T, and the value of $g(v)$ is determined by the values of its l children: $g(v_1), g(v_2), \ldots, g(v_l)$, where $v_1, \ldots, v_l$ are the children of node v. Particularly, $g(v) = \rho_v$ for each leaf node (source) v in T. For example, the value of function $g(\cdot)$ can be the sum of the data rates of all data streams of its children, or the average of the data rates of its children, depending on which types of IoT applications; and (4) the workload at each cloudlet (in terms of the total amount of computing resource consumed for hosting VNF instances of different functions) is as balanced as possible. The optimization objective is to minimize the operational cost of T, where the operational cost of implementing a multi-source IoT application consists of the computing cost, communication cost, and workload balancing cost that are defined as follows.

The computing cost: For each node v in the data routing tree T with the accumulative data steam $g(v)$, the computing cost at node v for accommodating the VNF instances of the multi-source IoT application is

$$C_{comp}(v) = \sum_{j=1}^{L} I(v, f_j) \cdot \left\lceil \frac{g(v)}{\mu_{f_j}} \right\rceil \cdot c_{f_j} \cdot c_{comp}, \tag{1}$$

where $I(v, f_j)$ is an indication function, which is 1 if there is any VNF instance of f_j in cloudlet v; otherwise 0, and the range of j is $1 \leq j \leq L$, and $v \in V$. $\lceil \frac{g(v)}{\mu_{f_j}} \rceil$ is the number of VNF instances of f_j required to process the accumulative data stream at node v, and c_{f_j} is the amount of computing resource of an instance of VNF f_j. The value of c_{comp} is the price of a unit computing resource consumption at any cloudlet $v \in V$. If node v is a leaf node in T, assuming that node v is source s_i, then $g(v) = \rho_{s_i}$; otherwise, assuming that v has l children $v_1, \ldots, v_l$, then $g(v) = g(g(v_1), \ldots, g(v_l))$, where $g(v)$ is an aggregation function at node v, e.g., $g(v)$ can be a summation function with $g(v) = \sum_{i=1}^{l} g(v_i)$.

The communication cost: The communication cost of transmitting a volume $\rho(u, v)$ of data on an edge (u, v) from node u to node v in G is

$$C_{comm}(u, v) = \rho(u, v) \cdot b_{comm}(u, v), \tag{2}$$

where $\rho(u, v)$ is the accumulative volume of data streams transmitted from node u to node m, and $b_{comm}(u, v)$ is the price of unit data transfer along edge (u, v).

The workload balancing cost: As the workload W_v of each cloudlet $v \in V$ is the actual computing resource consumption of cloudlet v, it is well-known that a heavily loaded cloudlet usually

has a much longer processing delay, compared to a lightly loaded cloudlet. Therefore, a penalty cost of this delay should be taken into account, which can be expressed by *the workload utility* of the cloudlet as follows.

Denote by γ_v the workload utility of cloudlet v for an IoT application that is defined as

$$\gamma_v = \frac{W_v}{C_v} = \frac{\sum_{j=1}^{L} I(v, f_j) \cdot \left\lceil \frac{g(v)}{\mu_{f_j}} \right\rceil \cdot c_{f_j}}{C_v}, \tag{3}$$

where $W_v = \sum_{j=1}^{L} I(v, f_j) \cdot \lceil \frac{g(v)}{\mu_{f_j}} \rceil \cdot c_{f_j}$ is the computing resource consumption on cloudlet v, and C_v is the computing capacity of cloudlet v.

The workload balancing cost at each cloudlet v thus is defined as follows:

$$cost_{load}(v) = \beta_{load} \cdot (2^{\gamma_v} - 1), \tag{4}$$

where β_{load} is a given coefficient of the workload balancing cost with $\beta_{load} > 0$, and $0 \leq cost_{load}(v) \leq \beta_{load}$, since $0 \leq \gamma_v \leq 1$. Note that the workload balancing cost is modeled as an exponential function of the workload utility, as a workload balancing cost increases with the increase on the workload utility, while the increasing rate of the workload balancing cost is supposed to become faster with the increase of the workload utility.

The optimization objective of the cost minimization problem of service provisioning for a single multi-source IoT application thus is to find a data routing tree T in G such that its operational cost $c(T)$ is minimized, where

$$c(T) = \sum_{v \in V(T)} C_{comp}(v) + \sum_{(u,v) \in E(T)} C_{comm}(u, v) + \sum_{v \in V(T)} cost_{load}(v),$$

where $V(T)$ and $E(T)$ are the set of nodes and edges in the data routing tree T, respectively. Notice that the solution of the problem consists of the construction of a data routing tree T, the placement of VNF instances of each VNF f_j in the SFC at some cloudlet nodes in T, and the workload balancing among cloudlets involved.

So far, we have formulated a cost minimization problem of service provisioning for a single multi-source IoT application in an MEC network through performing NFV-enabled network slicing to accommodate the IoT service. However, the MEC network as a public service platform is expected to accommodate different IoT applications for different users while minimizing the accumulative operational cost of implementing the IoT applications. We thus define another service provisioning problem for a set of multi-source IoT applications in the MEC network as follows.

Definition 2. Given an MEC network $G = (V, E)$ and a set $\mathcal{I}$ of multi-source IoT applications with each having an SFC requirement, assume that the VNF instances of different IoT applications cannot be shared with each other due to security and privacy concerns. *The service provisioning problem of a set of multi-source IoT applications* is to implement all IoT applications in $\mathcal{I}$ by finding a data routing tree in G for each application in $\mathcal{I}$ such that the total operational cost of implementing all IoT applications in $\mathcal{I}$ is minimized, assuming that there are sufficient resources in G to accommodate all the IoT applications, i.e., the optimization objective is to minimize

$$\sum_{r_i \in \mathcal{I}} c(T_{r_i}), \tag{5}$$

where $r_i \in \mathcal{I}$ is an IoT application, T_{r_i} is the data routing tree in G for $r_i \in \mathcal{I}$, and $c(T_{r_i})$ is the operational cost of T_{r_i} in Equation (5) with $1 \leq i \leq |\mathcal{I}|$. In other words, the problem is to build $|\mathcal{I}|$ VNF-enabled network slicing in the MEC for all IoT applications in $\mathcal{I}$. In case the MEC network G does not have sufficient resources to accommodate all IoT applications, the problem then is to

Table 1. Table of Symbols

Notations	Descriptions
$G = (V, E)$	an MEC network with a set V of nodes and a set E of links between nodes, where each node is an AP co-located with a cloudlet
C_v	the computing capacity of cloudlet $v \in V$
$\mathcal{F} = \langle f_1, f_2, \ldots, f_{\lvert\mathcal{F}\rvert} \rangle$	the set of VNFs in the system
c_{f_j} and μ_{f_j}	the computing resource consumption and the processing capacity of VNF f_j
$S = \{s_1, s_2, \ldots, s_K\}$ and ρ_{s_i}	the set of sources of the IoT application and the streaming data rate from source s_i
K and L	the number of sources and the length of the SFC of the IoT application
d	the destination of the IoT application
T	the constructed data routing tree for the IoT application
$g(v)$	the (aggregated) streaming data rate at node v of the data routing tree T
$I(v, f_j)$	the indication function, which is 1 if there is any VNF instance of f_j in cloudlet v; otherwise 0
c_{comp}	the price of a unit computing resource consumption at any cloudlet $v \in V$ in the network
$C_{comp}(v)$	the computing cost at node v for accommodating the VNF instances of the multi-source IoT application
$\rho(u, v)$	the accumulative volume of data streams transmitted from node u to node v
$b_{comm}(u, v)$	the price of unit data transfer along edge (u, v)
$C_{comm}(u, v)$	the communication cost of transmitting a volume $\rho(u, v)$ of data on an edge (u, v) from node u to node v
W_v	the computing resource consumption on cloudlet v
γ_v	the workload utility of a cloudlet v
β_{load}	the coefficient of the workload balancing cost
$cost_{load}(v)$	the workload balancing cost at cloudlet v
$c(T)$	the operational cost of the IoT application by constructing the data routing tree T
$V(T)$ and $E(T)$	the set of nodes and edges in the constructed data routing tree T
$\mathcal{I}$	the set of multi-source IoT applications
$T(s)$	the routing tree rooted at the destination d and source s being the first joined in
$T(s, k)$	the partial routing tree of $T(s)$ by adding the first k source nodes into the tree with $1 \le k \le K$
$P_k(u, s')$	a shortest path in $G \setminus T(s, k-1)$ between a tree node u and a source node s' that has not been contained in $T(s, k-1)$ yet
$P_k^T(u, d)$	the unique tree path in $T(s, k-1)$ between node u and destination d
$w_{f_{j'}}$	the amount of computing resource consumed by the VNF instances of $f_{j'}$
$U_i = u_1, u_2, \ldots, u_i$	the subsequence of nodes $u_1, \ldots, u_p$ with $1 \le i \le p$
$Y_{j'} = y_1, y_2, \ldots, y_{j'}$	the subsequence of VNFs $y_1, \ldots, y_q$ with $1 \le j' \le q$
$B(i, j')$	the optimal cost of workload balancing by placing the VNF instances of service functions in $Y_{j'}$ to nodes in U_i
$\Delta(T(s, k-1), u, s')$	the cost gain by connecting source s' to tree $T(s, k-1)$ through the tree node u

admit as many IoT applications as possible while minimizing the accumulative operational cost of admitted IoT applications.

For the sake of convenience, the symbols used in this article are summarized in Table 1.

4 NP-HARDNESS OF THE DEFINED PROBLEMS

In this section, we show that the defined two problems are NP-hard.

Theorem 1. *The cost minimization problem of service provisioning for a single multi-source IoT application in an MEC network $G = (V, E)$ is NP-hard.*

Proof. We consider a special case of the problem of concern, where the workload balancing at each cloudlet will not be considered, and the data stream rates of all sources of an IoT application are identical, i.e., $\rho = \rho_1 = \rho_2 = \cdots = \rho_K$, the aggregate function $g(\cdot)$ at each node is the average of the data stream rates of its children, i.e., $g(v) = \frac{\sum_{i=1}^{l} g(v_i)}{l} = \rho$, where node v has l children $v_1, v_2, \ldots, v_l$, and all VNF instances of each service function in the SFC are instantiated at the destination node d of the IoT application. This special cost minimization problem of service provisioning for a multi-source IoT application then is equivalent to the Steiner tree problem in G that finds a Steiner tree rooted at the destination node d and spanning all source nodes in $S \subset V$ such that the weighted sum of edges in the tree is minimized. It is well known that the Steiner tree problem is NP-hard [6], the cost minimization problem of service provisioning for a single multi-source IoT application thus is NP-hard, too. □

Theorem 2. *The service provisioning problem of a set of multi-source IoT applications in an MEC network $G = (V, E)$ is NP-hard.*

Proof. When $|\mathcal{I}| = 1$, the problem becomes the cost minimization problem of service provisioning for a single multi-source IoT application in MEC, which is NP-hard by Theorem 1. Thus, the service provisioning problem of a set of multi-source IoT applications is NP-hard, too. □

5 A NOVEL FRAMEWORK FOR IOT-DRIVEN SERVICE PROVISIONING

In this section, we provide a generic service provisioning framework for multi-source IoT applications in an MEC network. We consider an IoT application that consists of multiple sources, where each source is a base station (an AP) of a sensor subnetwork. The streaming sensory data from the sensors in the subnetwork are collected at the source (the base station or the gateway) and will be uploaded to the MEC network for further processing and storage. We assume that there is a destination node d in the MEC for the aggregated data storage. To ensure the security and privacy of transferring data stream, each data stream from its source to the destination must pass through a specified SFC as the requirement of the IoT application. For example, in a metropolitan region, there are many public parks. Assuming that a sensor network is deployed for each park, the monitored data stream from each of the sensor networks will be uploaded to the MEC through its nearby AP, and the collected data will be finally stored at the destination node for storage and processing.

A naive approach for sensory data collection from sources to the destination node is to build a routing path in the MEC network from each source (an AP) to the destination node, and place the demanded number of VNF instances of each service function in the SFC along the cloudlets in the routing path to meet the data rate of the source. As this IoT-driven service in an MEC network is expected for a long run, which implies that the owner of the IoT application will pay its IoT service at the cost that is proportional to the resource occupied by the service. However, running IoT services through this naive method is not economical, because the data streams from different sources of an IoT application can be aggregated or merged through exploring temporal-spatial data correlations. A much less accumulative volume of data streams of these sources can be routed to the destination node, thereby reducing the communication cost. However, the certain number of VNF instances of each service function in the SFC for the data stream of each source must be instantiated in the nodes in the routing path. The number of VNF instances of each service function in the SFC for the aggregated data stream could be significantly reduced if the data stream

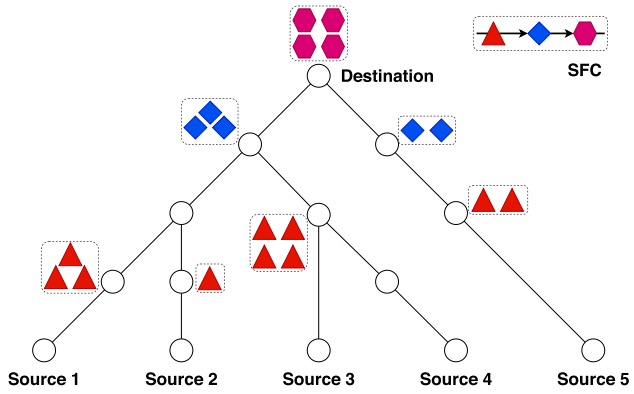


Fig. 2. An illustrative example of a data routing tree for a multi-source IoT application with three service functions in its SFC. There are five sources $s_1, s_2, \ldots, s_5$, and the data stream from each source to the destination node (the tree root) must pass through the VNF instances of each service function in the SFC.

aggregation at each intermediate node is performed, implying that less computing resource will be consumed.

To provide a cost-effective service in MEC for a multi-source IoT application, a data routing tree rooted at the destination node and spanning all source nodes in the MEC can be built to reduce both computing and communication costs through data aggregation and VNF instance instantiations at nodes in the routing paths. Each non-leaf node v in the tree except the destination has a parent node and a set of children nodes. There is an aggregation function $g(v)$ at v that aggregates the data streams from its children, e.g., a simple aggregation function is the sum of the data streams of its children. Also, less numbers of VNF instances of a VNF in the SFC can be instantiated at node v for its aggregated data processing. However, the data stream along the unique routing path in the tree from each source to the destination must pass through its demanded number of VNF instances in the SFC. Thus, both communication and computing resources for the IoT application can be shared among the data streams from different sources of the application through the data routing tree. Figure 2 is an illustrative example of a data routing tree for a multi-source IoT application.

6 ALGORITHM FOR THE COST MINIMIZATION PROBLEM OF SERVICE PROVISIONING FOR A SINGLE MULTI-SOURCE IOT APPLICATION

Since the cost minimization problem of service provisioning for a single multi-source IoT application in MEC is NP-hard, in this section, we first develop an efficient algorithm for it, based on the proposed service framework. We then analyze the properties of the solution delivered and the time complexity of the proposed algorithm.

6.1 Overview of The Proposed Algorithm

The proposed algorithm proceeds greedily. We assume that the destination (a cloudlet) of each multi-source IoT application is given in advance. Let S be the set of all sources of the IoT application. For each source node $s \in S$, a data routing tree $T(s)$ is constructed by setting the destination node d as the tree root, adding node $s \in S$ as the first source node to the tree, followed by adding the other source nodes to $T(s)$ greedily, one by one. Each time, a source node with the minimum cost increment compared with the previous cost of $T(s)$ will be chosen to add to the tree until all source nodes are added to the tree. Thus, a data routing tree $T(s_0)$ with the minimum operational cost is

chosen from the $|S|$ trees, which will be the solution to the problem. Specifically, we first choose a source node $s \in S$ as the very first node to add to the data routing tree $T(s)$. To this end, we find a shortest path P_1 in G from s to the destination d, and place the VNF instances of each function $f \in SFC$ to the nodes along path P_1 while balancing the workload of the nodes in the path (the detailed VNF instance placement will be discussed later), where the number of VNF instances of each service function in the SFC is determined by the data rate of source s. Let $S' = S \setminus \{s\}$ be the set of the rest source nodes that have not been added to $T(s)$ yet. The rest source nodes in S' then are added to tree $T(s)$ iteratively, one by one. We claim that the data stream along the tree path from each source node to the destination must pass through its demanded number of VNF instances of each service function only once, i.e., no VNF instances of a service function is wrongly placed, or placed in multiple nodes in the path.

Let $T(s,k)$ be a partial data routing tree of $T(s)$ by adding the first k source nodes into the tree with $1 \le k \le K$. When $k = 1$, the partial data routing tree $T(s,1)$ is a shortest path P_1 in G from s to d, where each edge $(v_i, v_{i+1}) \in P_1$ with $v_1 = s$, $v_{|P_1|+1} = d$, and v_{i+1} is the parent of v_i in the tree. The accumulative data stream on edge (v_i, v_{i+1}) in P_1 is $g(v_i)$, where $g(v)$ is an aggregate function at node v. Therefore, the communication cost on edge (v_i, v_{i+1}) is $b_{comm}(v_i, v_{i+1}) \cdot g(v_i)$, where $b_{comm}(v_i, v_{i+1})$ is the communication cost of one unit data transfer on edge (v_i, v_{i+1}). The number of VNF instances of each service function f_j will be deployed to some of the nodes along path P_1, i.e., the placement order of these VNF instances in P_1 from s to d is $f_1, f_2, \ldots, f_L$. We will deal with the VNF instance placement later, by incorporating the workload balancing among the cloudlets in P_1. It can be seen that the claim holds for this initial tree construction.

We assume that $T(s,k-1)$ has been constructed, and the first $(k-1)$ sources are added to the tree. We now construct $T(s,k)$ by adding a new source node s' to the tree.

Let $P_k(u,s')$ be a shortest path in $G \setminus T(s,k-1)$ between a tree node u and a source node s' that has not been contained in $T(s,k-1)$ yet. Let $P_k^T(u,d)$ be the unique tree path in $T(s,k-1)$ between node u and destination d. The VNF instances of functions of the subchain $\langle f_j, f_{j+1}, \ldots, f_L \rangle$ of the SFC are deployed in $P_k^T(u,d)$, i.e., no VNF instances of function $f_{j'}$ with $1 \le j' \le j-1$ are deployed in any node in $P_k^T(u,d)$. For the VNF instances of $f_{j'}$ placed at the node v in path $P_k^T(u,d)$ with $j \le j' \le L$, the accumulate data dream $g(v)$ at node v then is recomputed, by incorporating the data stream $\rho_{s'}$ from source s' through branch $P_k(u,s')$. Therefore the number of VNF instances of $f_{j'}$ at node v in path $P_k^T(u,d)$ is $\lceil \frac{g(v)}{\mu_{f_{j'}}} \rceil$.

For the data stream along a routing path from s' to d that consists of two segments $P_k^T(u,d)$ and $P_k(s',u)$, although the data stream along the tree path $P_k^T(u,d)$ has been processed by the VNF instances in the subchain $\langle f_j, \ldots, f_L \rangle$ in $P_k^T(u,d)$, the data stream along path $P_k(s',u)$ from node s' to node u has not been processed by the VNF instances of each service function in the subchain $\langle f_1, f_2, \ldots, f_{j-1} \rangle$ of the SFC. We thus need to place the number of VNF instances for each of them in the nodes in $P_k(s',u)$ in their specified order, where the rate of data stream on each edge in $P_k(s',u)$ is $\rho_{s'}$. In the following, we show how to place the VNF instances of each function in $\langle f_1, f_2, \ldots, f_{j-1} \rangle$ to cloudlets (nodes) in $P_k(s',u)$ to process the data stream while balancing the workload among the cloudlets, by developing a dynamic program solution for this workload-aware VNF instance placement problem.

6.2 Workload-Aware VNF Instance Placement in a Routing Path

We here consider the workload-aware VNF instance placement on the nodes in a routing path. We assume that there is a data stream from source s' with data rate $\rho_{s'}$, the number of VNF instances of each function $f_{j'}$ in the SFC needs to be placed to a node in path $P_k(u,s')$ in order, where $1 \le j' \le j-1$. We aim to place the VNF instances such that the sum of the workload balancing

costs among the nodes is minimized, by formulating the following workload-aware VNF instance placement problem.

Let $u_0, u_1, u_2, \ldots, u_p$ be the node sequence in path $P_k(s', u)$ starting from u with $u_0 = u$, $u_p = s'$ and $p \geq 1$, where the computing resource consumption and computing capacity of each cloudlet node $u_i \in V$ are W_{u_i} and C_{u_i}, respectively. *The workload utility* of node u_i is $\gamma_{u_i} = \frac{W_{u_i}}{C_{u_i}}$ by Equation (3).

There is a subchain $\langle f_{j-1}, f_{j-2}, \ldots, f_1 \rangle$ of the SFC for the data stream from source s' with data rate $\rho_{s'}$. The number of VNF instances of $f_{j'}$ to be placed along path $P_k(u, s')$ is $\lceil \frac{\rho_{s'}}{\mu_{f_{j'}}} \rceil$, where $\mu_{f_{j'}}$ is the data processing capacity of an VNF instance of function $f_{j'}$ in any cloudlet and $1 \leq j' \leq j-1$. The amount of computing resource consumed by the VNF instances of $f_{j'}$ is

$$w_{f_{j'}} = \left\lceil \frac{\rho_{s'}}{\mu_{f_{j'}}} \right\rceil \cdot c_{f_{j'}}. \tag{6}$$

Let $y_1, y_2, \ldots, y_q$ represent the function sequence $\langle f_{j-1}, f_{j-2}, \ldots, f_1 \rangle$ with $y_1 = f_{j-1}$ and $y_q = f_1$. Each $y_{j'}$ has a workload $w_{y_{j'}}$ by Equation (6) with $1 \leq j' \leq q$. *The workload-aware VNF instance placement problem* is to place the VNF instances of functions in the subchain $\langle y_1, y_2, \ldots, y_q \rangle$ to the nodes in path $u_1, u_2, \ldots, u_p$ such that the sum of the workload balancing cost is minimized, assuming that there is sufficient computing resource at each cloudlet u_i to accommodate all VNF instances of function $y_{j'}$ in the subchain with $1 \leq i \leq p$ and $1 \leq j' \leq q$.

We devise a dynamic programming algorithm for the workload-aware VNF instance placement problem as follows.

Let $U_i = u_1, u_2, \ldots, u_i$ be the subsequence of $u_1, \ldots, u_p$ and $Y_{j'} = y_1, y_2, \ldots, y_{j'}$ the subsequence of $y_1, \ldots, y_q$ with $1 \leq i \leq p$ and $1 \leq j' \leq q$.

Denote by $B(i, j')$ the optimal cost of workload balancing by placing the VNF instances of service functions in $Y_{j'}$ to nodes in U_i. Then,

$$B(i,j') = \begin{cases} \beta_{load} \cdot \left(2^{\gamma_{u_1} + \frac{\sum_{l=1}^{j'} w_{y_l}}{C_{u_1}}} - 1 \right) & i = 1, \\ \beta_{load} \cdot \left(\sum_{i'=1}^{i} (2^{\gamma_{u_{i'}}} - 1) + \min \left\{ 2^{\gamma_{u_{i'}} + \frac{w_{y_1}}{C_{u_{i'}}}} - 2^{\gamma_{u_{i'}}} | 1 \leq i' \leq i \right\} \right) & j' = 1, \\ \min \{ B(i-1, j') + \beta_{load} \cdot (2^{\gamma_{u_i}} - 1), \\ \quad B(i-1, j'-1) + \beta_{load} \cdot \left(2^{\gamma_{u_i} + \frac{w_{y_{j'}}}{C_{u_i}}} - 1 \right) \right\} & i > 1 \ \ \& \ \ j' > 1, \end{cases}$$

where γ_{u_i} is the workload utility of node u_i, $B(1, j') = \beta_{load} \cdot (2^{\gamma_{u_1} + \frac{\sum_{l=1}^{j'} w_{y_l}}{C_{u_1}}} - 1)$ implies that there is only one cloudlet u_1 and the VNF instances of the chain will be hosted by the cloudlet, while $B(i, 1) = \beta_{load} \cdot (\sum_{i'=1}^{i} (2^{\gamma_{u_{i'}}} - 1) + \min\{2^{\gamma_{u_{i'}} + \frac{w_{y_1}}{C_{u_{i'}}}} - 2^{\gamma_{u_{i'}}} | 1 \leq i' \leq i\})$ indicates that the VNF instances of y_1 is added to such a cloudlet that results in the minimum increase of the workload balancing cost among the cloudlets. $B(i, j') = B(i-1, j') + \beta_{load} \cdot (2^{\gamma_{u_i}} - 1)$ implies that the VNF instances of the subchain $Y_{j'}$ are placed to the first $(i-1)$ nodes in the node sequence U_i. $B(i, j') = B(i-1, j'-1) + \beta_{load} \cdot (2^{\gamma_{u_i} + \frac{w_{y_{j'}}}{C_{u_i}}} - 1)$ implies that the VNF instances of the subchain $Y_{j'-1}$ are placed to the nodes in the node sequence U_{i-1}, and the VNF instances of $y_{j'}$ are placed on node u_i.

ALGORITHM 1: Algorithm for the workload-aware VNF instance placement problem

Input: A node sequence $u_1, u_2, \ldots, u_p$, a function sequence $y_1, y_2, \ldots y_q$, and the date rate $\rho_{s'}$ of a source s', assuming that the current workload W_{u_i} and capacity C_{u_i} of each cloudlet node u_i is given.

Output: the number of VNF instances of each function y_j is placed to a node in the node sequence such that the function order does not change and the sum of the workload balancing cost among the nodes is minimized.

for $i \leftarrow 1$ to p **do**
  for $j' \leftarrow 1$ to q **do**
    if $i = 1$ **then**
      $B(i, j') \leftarrow \beta_{load} \cdot \left(2^{\gamma_{u_1} + \frac{\sum_{l=1}^{j'} w_{y_l}}{C_{u_1}}} - 1 \right)$;
    else
      if $j' = 1$ **then**
        $B(i, j') \leftarrow \beta_{load} \cdot \left(\sum_{i'=1}^{i} (2^{\gamma_{u_{i'}}} - 1) + \min \left\{ 2^{\gamma_{u_{i'}} + \frac{w_{y_1}}{C_{u_{i'}}}} - 2^{\gamma_{u_{i'}}} \,|\, 1 \le i' \le i \right\} \right)$;
      else
        $B(i, j') \leftarrow \min \left\{ B(i-1, j') + \beta_{load} \cdot (2^{\gamma_{u_i}} - 1),\ B(i-1, j'-1) + \beta_{load} \cdot \left(2^{\gamma_{u_i} + \frac{w_{y_{j'}}}{C_{u_i}}} - 1 \right) \right\}$;
      end if;
    end if;
  end for;
end for;
return the solution $B(p, q)$.

The value of the solution to the workload-aware VNF instance placement problem thus is $B(p, q)$. The time complexity of the proposed algorithm is $O(p \cdot q) = O(|V| \cdot L)$ as $p \le |V|$ and $q \le L$, where L is the length of the SFC. The detailed algorithm is given in `Algorithm 1`.

It can be seen that the data stream from source s' to the destination node d passes through the demanded VNF instances of each function f_j in the SFC with $1 \le j \le L$, while the data streams of the first $(k-1)$ sources in tree $T(s, k)$ pass through their demanded numbers of VNF instances of each service function in the SFC, following the proposed tree construction.

The cost gain $\Delta(T(s, k-1), u, s')$, by connecting source s' to tree $T(s, k-1)$ through the tree node u, is the sum of the increased computing cost, communication cost and workload balancing cost by adding source s' to tree $T(s, k-1)$.

The detailed algorithm for the construction of a potential partial routing tree, by connecting source s' to tree $T(s, k-1)$ through the tree node u, is given in `Algorithm 2`.

6.3 Algorithm for the Cost Minimization Problem of Service Provisioning for a Single Multi-source IoT Application

In the following, we propose an efficient algorithm for the cost minimization problem of service provisioning for a single multi-source IoT application, by making use of `Algorithm 2` as its subroutine. Specifically, we first construct $|S|$ data routing trees, where S is the set of sources of the IoT application. Each data routing tree $T(s)$ is constructed by setting the destination node d as the tree root and source $s \in S$ as the first node added to the tree. Then each of the rest source nodes in $S \setminus \{s\}$ is added to $T(s)$ iteratively. In each iteration, the source node with the minimum cost increment compared with the previous cost of $T(s)$ will be chosen to add to the tree until all source nodes are added, by invoking `Algorithm 2`. Having constructed $|S|$ data routing trees, the data routing tree $T(s_0)$ with the minimum operational cost is chosen as the solution to the problem. The detailed algorithm is presented in `Algorithm 3`.

ALGORITHM 2: A potential routing tree construction by connecting source s' to tree $T(s, k-1)$ through the tree node u

Input: An MEC network $G = (V, E)$ and a multi-source IoT application with an $SFC = \langle f_1, f_2, \ldots, f_L \rangle$, a destination $d \in V$, K sources $s_1, s_2 \ldots, s_K$, and each source s_i has a data stream rate ρ_{s_i} with $1 \le i \le K$.

Output: A potential partial routing tree rooted at d and spanning k source nodes with the first joining in source s, and assuming that the first $(k-1)$ sources are added already and the partial routing tree is $T(s, k-1)$, a routing tree is constructed with the minimum cost where source s' is joining to the tree through a path $P_k(s', u)$ and u is the tree node.

1: **if** $k = 1$ **then**
2: Find a shortest path P_1 in G from s' to d;
3: Place the demanded number of VNF instances of f_j to the nodes along path P_1 with $1 \le j \le L$ by invoking `Algorithm 1`;
4: **else**
5: Update the number of VNF instances of each function f_j at each node v in the tree path $P_k^T(u, d)$ from u to d, considering to add an extra $\rho_{s'}$ data stream to each edge along path $P_k^T(u, d)$;
6: /* assuming f_{j-1} is the first service function in the subchain from f_1 to f_L whose VNF instances do not appear in a node in $P_k^T(u, d)$; */
7: Place the demanded number of VNF instances of each function $f_{j'}$ with $1 \le j' \le j-1$ to the nodes of a shortest path $P_k(s', u)$ in $G \setminus T(s, k-1)$ from s' to u' with the data rate $\rho_{s'}$ by invoking `Algorithm 1`, where (u', u) is the tree edge with endpoints u' and u, and u' is a child of u in the tree, by balancing the workload among the nodes in the path;
8: $\Delta(T(s, k-1), u, s') \leftarrow c(T(s, k-1) \cup P_k(s', u)) - c(T(s, k-1))$; /* $T(s, k-1) \cup P_k(s', u)$ is the partial routing tree by connecting source s' to tree $T(s, k-1)$ through the tree node u */
9: **end if**;
10: **return** the constructed partial routing tree $T(s, k-1) \cup P_k(s', u)$ and the increased cost $\Delta(T(s, k-1), u, s')$.

6.4 Analysis of The Proposed Algorithm

The rest is to address some important properties of the built data routing tree and the time complexity of the proposed algorithm. Recall that $f_1, f_2, \ldots f_L$ are the service functions in the SFC of the IoT application.

Lemma 1. *Let $e_1, e_2, \ldots, e_q$ be the edges on the routing path in the data routing tree T from a source node s to the destination d, and let $v_1, v_2, \ldots, v_{q+1}$ be the corresponding nodes in the path with $v_1 = s$ and $v_{q+1} = d$. We have (i) the VNF instances of each function $f_j \in SFC$ are installed in the nodes in its specified order with $1 \le j \le L$, i.e., if the VNF instances of f_j are deployed in node v_i, then the VNF instances of $f_{j'}$ cannot be deployed to any node $v_{i'}$ with $j' > j$ and $i' < i$. We term this as the VNF deployment order by the SFC; and (ii) the VNF instances of any service function $f \in SFC$ cannot appear in multiple nodes in any routing path in the tree from a source to the destination, i.e., it is prohibited that VNF instances of function f are deployed into two different nodes v_i and v_j in the routing path with $i \ne j$. Otherwise, the data stream data will pass through the VNF instances of f twice from its source to the destination.*

Proof. Property (i) holds because the construction of the data routing tree by `Algorithm 3` follows the function dependency of the SFC.

By Property (i) of tree T, if some VNF instances of a function f_j are instantiated at node v_i, then any of its other VNF instances cannot be deployed to other nodes in the subtree rooted at v_i; otherwise, Property (ii) is violated, i.e., the data stream from a descendent leaf node of v_i will pass through the VNF instances of the service function twice at two different cloudlets (at that node and at v_i), and this is prohibited by the SFC requirement of the IoT application. □

Lemma 2. *Let $T(s, k)$ be a constructed partial data routing tree that contains the first $(k-1)$ source nodes by `Algorithm 2`, where source s is the first one added to the tree, for all k with $1 \le k \le K$. Then,*

ALGORITHM 3: Heuristic algorithm for the cost minimization problem of service provisioning for a single multi-source IoT application in MEC

Input: An MEC network $G = (V, E)$ and an IoT application with an $SFC = \langle f_1, f_2, \ldots, f_L \rangle$ and K sources $s_1, s_2 \ldots, s_K$ and a destination $d \in V$, each source s_i has a data rate ρ_{s_i} with $1 \leq i \leq K$.

Output: A data routing tree rooted at d and spanning all source nodes, and different numbers of VNF instances of the SFC are placed in the nodes of the tree such that the total operational cost outing of the IoT application is minimized.

$cost \leftarrow \infty$; /* the operational cost of the solution */
$T \leftarrow \emptyset$; /* the data routing tree for the IoT application */
for $s \in S$ **do**
  /* The construction of a routing tree $T(s)$ with the minimum cost*/
  Find a shortest path P_1 in G from s to d;
  Construct the partial routing tree $T(s, 1)$ through placing the VNF instances of service function in the SFC along the nodes in P_1 with the data rate ρ_s, by calling `Algorithm 1`;
  $k \leftarrow 1$;
  $S' \leftarrow S \setminus \{s\}$;
  while $S' \neq \emptyset$ **do**
    $k \leftarrow k + 1$
    for each $s' \in S'$ **do**
      for each $u \in T(s)$ **do**
        Construct the partial routing tree $T(s, k-1) \cup P_k(s', u)$ by connecting source s' to tree $T(s, k-1)$ through the tree node u along the path $P_k(s', u)$, by calling `Algorithm 2`;
        Compute the total cost gain $\Delta(T(s, k-1), u, s')$ if path $P_k(s', u)$ is added to $T(s, k-1)$;
      end for;
    end for;
    $u_0, s'_0 \leftarrow \min \arg_{u \in T(s), s' \in S'} \{\Delta(T(s, k-1), u, s')\}$;
    $T(s, k) \leftarrow T(s, k-1) \cup P_k(s'_0, u_0)$; /* adding a source s'_0 with the minimum cost gain to the data routing tree */
    $S' \leftarrow S' \setminus \{s'_0\}$;
  end while;
  $T(s) \leftarrow T(s, K)$;
  if $c(T(s)) < cost$ **then**
    $cost \leftarrow c(T(s))$;
    $s_0 \leftarrow s$;
  end if;
end for;
return the solution $T(s_0)$ with the total operational cost $c(T(s_0))$.

the data stream along the tree path from each of the sources to the destination will pass through the demanded number of VNF instances of each service function in the SFC, and the solution delivered is feasible.

Proof. The claim can be shown by induction. Following `Algorithm 2`, when $k = 1$, the tree is a shortest path from s to d and the number of VNF instances of each service function in the SFC are placed to the nodes in the path. Then the claim clearly holds. We assume that the first $(k-1)$ sources have been added to the data routing tree and the claim holds for the data stream from each of the sources. Now, we show the claim still holds when adding the kth source into the tree. Following the algorithm, the number of VNF instances of each service function in the SFC is increased or instantiated accordingly to meet the data rate of the data stream from the kth source. The claim still holds. □

Theorem 3. *Given an MEC network $G = (V, E)$ and an IoT application that consists of K sources $s_1, \ldots, s_K$ and a destination $d \in V$, where each source gateway $s_i \in V$ has a data rate ρ_{s_i} with $1 \leq i \leq K$, and there is an SFC requirement $\langle f_1, \ldots, f_L \rangle$ for the IoT application and a defined data aggregation function $g(\cdot)$, the cost minimization problem of service provisioning for a single*

multi-source IoT application in G is to find a data routing tree T in G for implementing the IoT application such that the total operational cost is minimized. There is an efficient algorithm, `Algorithm 3`*, which delivers a feasible solution for the problem. The algorithm takes $O(|V|^2 \cdot K^3 \cdot L)$ time.*

Proof. We first show that the solution delivered is feasible by Lemma 1 and Lemma 2. That is, the data stream from each source $s \in S$ to the destination d must pass through the demanded number of VNF instances of each function in the SFC prior to the destination. Let a data routing tree $T(s)$ rooted at d is chosen as the solution of the problem, the construction of $T(s)$ proceeds iteratively. Within each iteration, a source node joins in the tree. Let $T(s,k)$ be the partial data routing tree of $T(s)$ that contains the first k source nodes. Clearly, for tree $T(s,1)$, the data stream ρ_s is routed along a shortest path (i.e., $T(s,1)$) in G from s to d, and the demanded number of VNF instances of each service function in the SFC is instantiated in the nodes along the path. Assume that all data streams from the sources in the partial data routing tree $T(s,k-1)$ meet the condition: their demanded numbers of VNF instances of each service function in the SFC are placed in nodes of $T(s,k-1)$ already. When we consider the kth source s' joining in $T(s,k-1)$ to form tree $T(s,k)$, assume that node u is the tree node at which a branch from s' to u is attached to $T(s,k-1)$. It can be seen that the data stream with data rate $\rho_{s'}$ along the path $P_k(s',u) \cup P_k^T(u,d)$ from s' to d must pass through the demanded VNF instances of each service function in the SFC, following the construction of $T(s,k)$ by `Algorithm 2`. The solution delivered by `Algorithm 3` thus is feasible.

We then analyze the time complexity of the proposed algorithm, `Algorithm 3`. There are K potential data routing trees to be constructed. To construct a data routing tree that contains a source s initially, another source node with the minimum cost increment will be chosen to add to the tree, this procedure continues until all source nodes are added to the tree. To calculate the cost increment of adding a source s' to the partial data routing tree, we first find the shortest paths between source s' and each node in the partial data routing tree and that takes $O(|V|^2)$ time, and the VNF instance placements along the tree paths by invoking the dynamic programming algorithm—`Algorithm 1` that takes $O(|V|^2 \cdot L)$ time. Thus, the running time of `Algorithm 3` is $O(|V|^2 \cdot K^3 \cdot L)$. □

Remarks. The proposed algorithm assumed that all VNF instances of any IoT application can be hosted by a single cloudlet, this assumption is reasonable in practice as the resource demands by a single IoT application should be far less than the amount of resource a cloudlet can provide. However, it is not uncommon that the resources in MEC are very limited, the proposed algorithm can be easily modified for this resource restriction case. That is, if the residual resource of a cloudlet is less than the resource demands of an IoT application, that cloudlet will not involve resource allocation for the IoT application by hosting any of the VNF instances of the IoT application (i.e., it cannot accommodate any VNF instance in the data routing tree for the IoT application). The cloudlet, however, is still part of the data routing tree for the IoT application to maintain the network connectivity, i.e., there may be routing paths passing through the cloudlet.

7 AN ALGORITHM FOR THE SERVICE PROVISIONING PROBLEM OF A SET OF MULTI-SOURCE IOT APPLICATIONS

In this section, we consider the service provisioning for a set of multi-source IoT applications in an MEC network, where each IoT application can share the VNF instances of data streams from its different sources, but it is prohibited to share VNF instances among different IoT applications due to their security and privacy concerns. In other words, we aim to perform VNF-enabled network slicing in the MEC to provide services for different IoT applications such that the total operational cost is minimized.

We implement the multi-source IoT applications in $\mathcal{I}$ one by one iteratively. Each time, one application with the minimum operational cost is chosen from the remaining applications by invoking Algorithm 3. This procedure continues until all IoT applications in $\mathcal{I}$ are implemented. The detailed algorithm is given as Algorithm 4.

ALGORITHM 4: An algorithm for the service provisioning of a set of multi-source IoT applications

Input: An MEC network $G = (V, E)$ and a set $\mathcal{I}$ of IoT applications with an $SFC = \langle f_{i,1}, f_{i,2}, \ldots, f_{i,L_i} \rangle$ of each application $r_i \in \mathcal{I}$ and K_i sources $s_{i,1}, s_{i,2}, \ldots, s_{i,K_i}$ and a destination $d_i \in V$, each source $s_{i,j}$ has a data rate $\rho_{i,j}$ with $1 \le j \le K_i$.

Output: A solution of implementing all IoT applications such that the total cost is minimized, where the solution of each IoT application r_i is a data routing tree with K_i sources.

$total\_cost \leftarrow 0$; /* the value of the solution */
$\mathcal{T} \leftarrow \emptyset$ /* the set of data routing trees for the IoT applications in $\mathcal{I}$ */
$\mathcal{U} \leftarrow \mathcal{I}$; /* the set of IoT applications that have not been implemented in the MEC yet */
while $\mathcal{U} \neq \emptyset$ **do**
  for each IoT application $r_i \in \mathcal{U}$ **do**
    Find a data routing tree T_{r_i} for it, and compute the total operational cost $c(T_{r_i})$ of tree T_{r_i}, by calling Algorithm 3;
  end for;
  Let $r_{i_0} = \text{argmin}_{r_i \in \mathcal{U}} \{c(T_{r_i})\}$ /* $T_{r_{i_0}}$ is a data routing tree with the minimum cost among the applications in $\mathcal{U}$ */;
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{T_{r_{i_0}}\}$;
  Update all computing resource in cloudlets for implementing r_{i_0}, and calculate the residual computing resource of cloudlets in the MEC;
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \{r_{i_0}\}$;
  $total\_cost \leftarrow total\_cost + c(T_{r_{i_0}})$;
end while;
return The solution $\mathcal{T}$.

Theorem 4. *Given an MEC network $G = (V, E)$, each cloudlet $v \in V$ has computing capacity C_v, a set $\mathcal{I}$ of multi-source IoT applications with each application $r_i \in \mathcal{I}$ having K_i sources and an SFC_i with length of L_i, and each source $s_{i,j}$ of r_i has a data rate $\rho_{i,j}$ with $1 \le j \le K_i$. There is an efficient algorithm for the service provisioning problem of a set of multi-source IoT applications,* Algorithm 4, *which takes $O(|V|^2 \cdot K_{max}^3 \cdot L_{max} \cdot |\mathcal{I}|^2)$ time, where $K_{max} = \max\{K_i \mid r_i \in \mathcal{I}\}$ and $L_{max} = \max\{L_i \mid r_i \in \mathcal{I}\}$ are the maximum number of sources and the maximum length of all SFCs among IoT applications in $\mathcal{I}$, respectively.*

Proof. Because Algorithm 4 invokes Algorithm 3 iteratively, the feasibility of the solution delivered by Algorithm 4 for the service provisioning problem of a set of multi-source IoT applications can be shown by the feasibility of the solution delivered by Algorithm 3 for the cost minimization problem, which has been proved in Theorem 3, omitted.

The time complexity of Algorithm 4 is $O(|V|^2 \cdot K_{max}^3 \cdot L_{max} \cdot |\mathcal{I}|^2)$, as an application with the minimum operational cost from the remaining applications is chosen at each iteration by invoking Algorithm 3, while the time complexity of Algorithm 3 is $O(|V|^2 \cdot K_{max}^3 \cdot L_{max})$ by Theorem 3, and the number of the invoking of Algorithm 3 is $O(\mathcal{I}^2)$. □

8 PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed algorithms for the service provisioning for IoT applications in an MEC network. We also investigate the impact of important

parameters—the number L of service function in a service chain, the number K of sources, and the workload balancing coefficient β_{load}, on the performance of the proposed algorithms.

8.1 Environment Settings

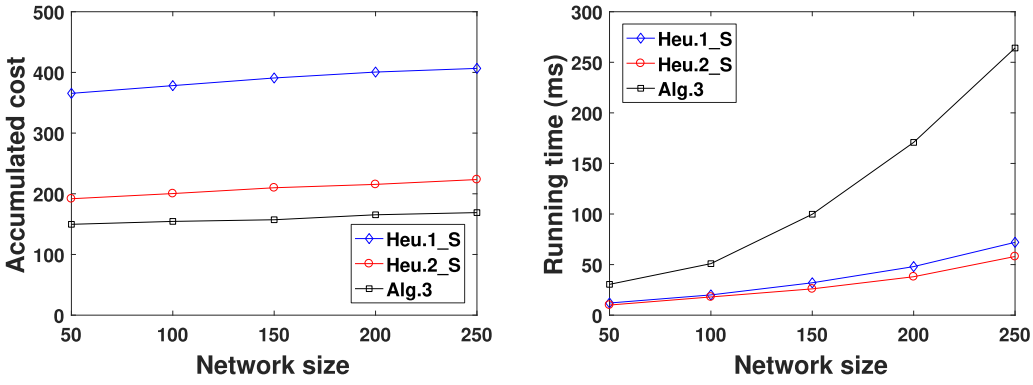
We consider an MEC network that consists of 100 APs, with each having a co-located cloudlet. We generate the topologies of MEC networks using GT-ITM [1]. The capacity of each cloudlet is set from 30,000 to 60,000 MHz [13, 23]. We assume that there are 10 types of VNFs, and the computing resource consumption of each VNF instance ranges from 20 to 100 MHz [20]. We further assume that the processing rate of each VNF instance varies from 30 to 80 packet per millisecond, where each data packet is of size 64 KB [20]. The number of different types of SFCs is set as 20, and the length of each varies from 3 to 7 [21]. For each multi-source IoT application request, the number of sources is set from 4 to 8, and the data rate of each source ranges from 2 to 10 packet per millisecond [21]. The communication cost of each link is randomly drawn from \$0.05 to \$0.4 per data packet [20]. The computing resource consumption cost is set at \$0.1 per MHz [28]. The coefficient of the workload balancing cost (i.e., β_{load}) is set at 2,000. The aggregation function of each multi-source IoT application is randomly chosen from the set $\{summation, averaging, maximum, minimum\}$. The value in each figure is the mean of the results out of 20 MEC network instances of the same size. The actual running time of each algorithm is based on a desktop with a 3.60 GHz Intel 8-Core i7-7700 CPU and 16 GB RAM. Unless otherwise specified, these parameters will be adopted in the default setting.

We evaluate the proposed algorithm `Algorithm 3` (referred to as `Alg.3`) against two baseline algorithms: `Heu.1_S` and `Heu.2_S`, for the cost minimization problem of service provisioning for a single multi-source IoT application. In algorithm `Heu.1_S`, we first find the shortest path between each source and the destination node, and the VNF instances of each service function in the SFC will be instantiated along the routing path. In this heuristic, neither computing resource (VNF instances) nor bandwidth resource is shared among the data streams of different sources for each IoT application. On contrary, algorithm `Heu.2_S` consolidates all the VNFs in the SFC of each request into a single cloudlet, and the VNFs are shared among the data streams from all sources. Especially, algorithm `Heu.2_S` first finds a shortest path P_1 from a randomly selected source s to the destination node of the request, and the cloudlet v with the least workload utility by Equation (3) on the shortest path is chosen to accommodate the VNFs. Based on path P_1, the data routing tree T for the request then is constructed by including the shortest path from each remaining source $s' \in S \setminus \{s\}$ to the chosen cloudlet v, one by one, and the data streams are then aggregated at each non-leaf node of the data routing tree T.

We then evaluate the proposed algorithm `Algorithm 4` (referred to as `Alg.4`) against algorithms `Heu.1_M` and `Heu.2_M`, for the service provisioning problem of a set of multi-source IoT applications. Algorithms `Heu.1_M` and `Heu.2_M` are built upon algorithms `Heu.1_S` and `Heu.2_S`, respectively. Especially, algorithm `Heu.1_M` processes IoT applications one by one by algorithm `Heu.1_S`, and picks them for consideration randomly. Algorithm `Heu.2_M` is similarly developed, omitted.

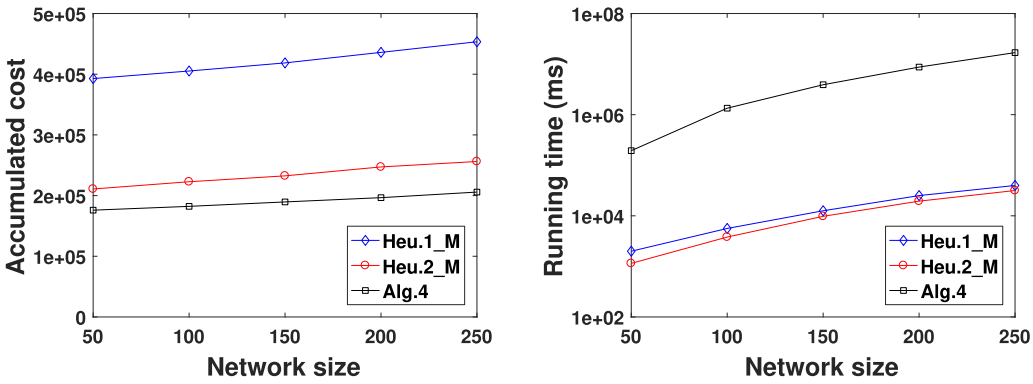
8.2 Performance Evaluation of Different Algorithms

We first studied the performance of `Alg.3` against algorithms `Heu.1_S` and `Heu.2_S`, respectively, by varying the network size from 50 to 250. Figures 3(a) and 3(b) plot the total cost and running time of different algorithms. It can be seen from Figure 3(a) that when the network size is 250, the total cost achieved by `Alg.3` is 41.6% of that by algorithm `Heu.1_S`, and 75.6% of that by algorithm `Heu.2_S`. This is because `Alg.3` establishes an efficient data routing tree that jointly considers VNF instance



(a) The total cost (b) The running time

Fig. 3. Performance of different algorithms for the cost minimization problem of service provisioning for a single multi-source IoT application, by varying the network size from 50 to 250.



(a) The total cost (b) The running time

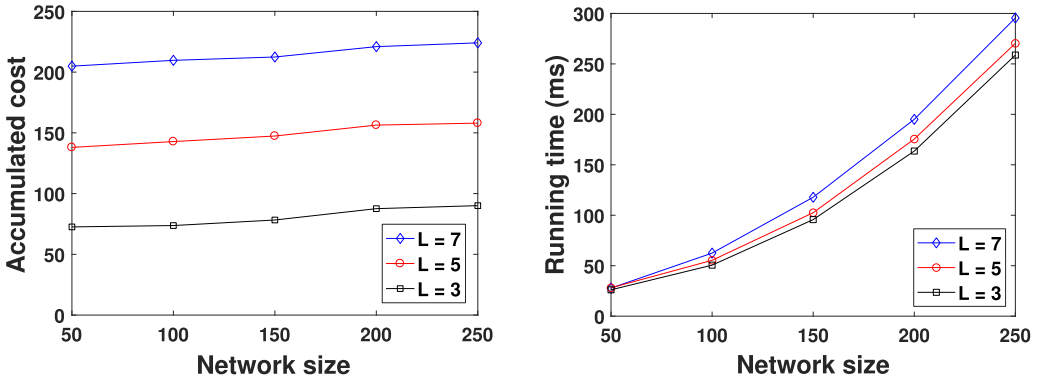
Fig. 4. Performance of different algorithms for the service provisioning problem of a set of multi-source IoT applications, by varying the network size from 50 to 250.

sharing and communication path sharing among different data streams of each IoT application, and the workload balancing among cloudlets.

We then investigated the performance of `Alg.4` against algorithms `Heu.1_M` and `Heu.2_M`, by varying the network size from 50 to 250, with 1,000 IoT applications. Figure 4(a) and Figure 4(b) depict the total cost and running time of different algorithms. It can be seen from Figure 4(a) that when the network size is 250, the total cost achieved by algorithm `Alg.4` is 45.3% of that by algorithm `Heu.1_M`, and 80.3% of that by algorithm `Heu.2_M`. The rationale behind this is that with the increase on the network size, algorithm `Alg.4` outperforms algorithms `Heu.1_M` and `Heu.2_M` not only in resource sharing but also in workload balancing among cloudlets.

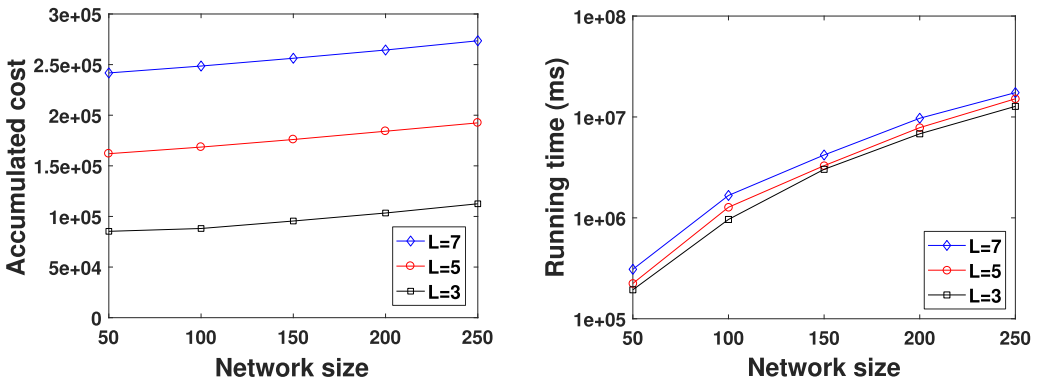
8.3 Impact of Important Parameters on The Performance of The Proposed Algorithms

We finally investigated the impact of important parameters on the performance of the proposed algorithms for the defined problems including the length L of SFCs, the number K of sources, and the coefficient β_{load} of the workload balancing cost, respectively. We started by studying the



(a) The total cost (b) The running time

Fig. 5. The impact of length L of the SFC on the performance of Alg.3.

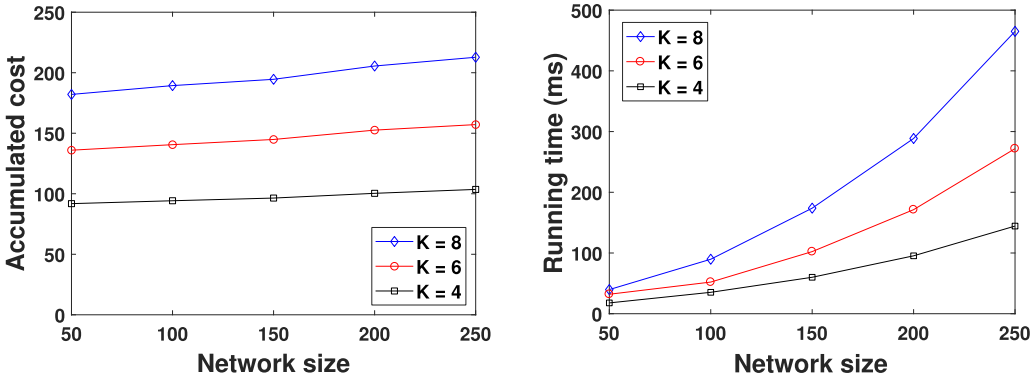


(a) The total cost (b) The running time

Fig. 6. The impact of length L of the SFC on the performance of Alg.4.

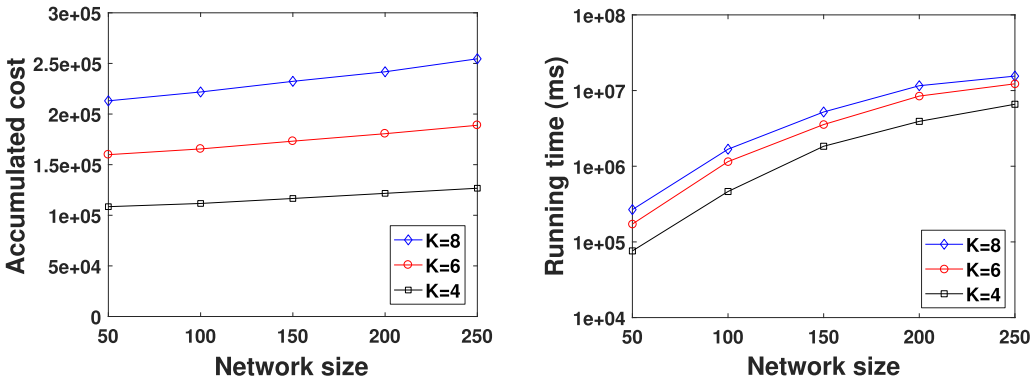
impact of the length L of SFCs on the proposed algorithms. Figures 5(a) and 5(b) demonstrates the total cost and running time of the solution delivered by algorithm Alg.3 for different network sizes when $L = 3$, 5, and 7, respectively. Figures 6(a) and 6(b) depict the total cost and running time of the solution delivered by algorithm Alg.4 with 1,000 IoT applications for different network sizes when $L = 3$, 5, and 7, respectively. It can be seen from Figure 5(a) that with the network size of 250, the total cost of the solution delivered by algorithm Alg.3 when $L = 3$ is 40.2% of itself when $L = 7$. This can be justified that with a small value of L, the IoT applications consumes less computing resource, thus, both the computing cost and the workload balancing cost decrease. The similar performance behaviors can be observed in Figure 6(a), too.

We then investigated the impact of the number K of sources on the proposed algorithms. Figures 7(a) and 7(b) plot the total cost and running time of the solution delivered by algorithm Alg.3 for different network sizes when $K = 4$, 6, and 8, respectively. Figures 8(a) and 8(b) show the total cost and running time of the solution delivered by algorithm Alg.3 with 1,000 IoT applications for different network sizes when $K = 4$, 6, and 8. It can be seen from Figure 7(a) that with the network size of 250, the total cost of the solution delivered by algorithm Alg.3 when $K = 4$ is 48.7% of itself when $K = 8$. This can be justified that with a small value of K, the IoT applications



(a) The total cost (b) The running time

Fig. 7. The impact of the number K of sources on the performance of Alg.3.

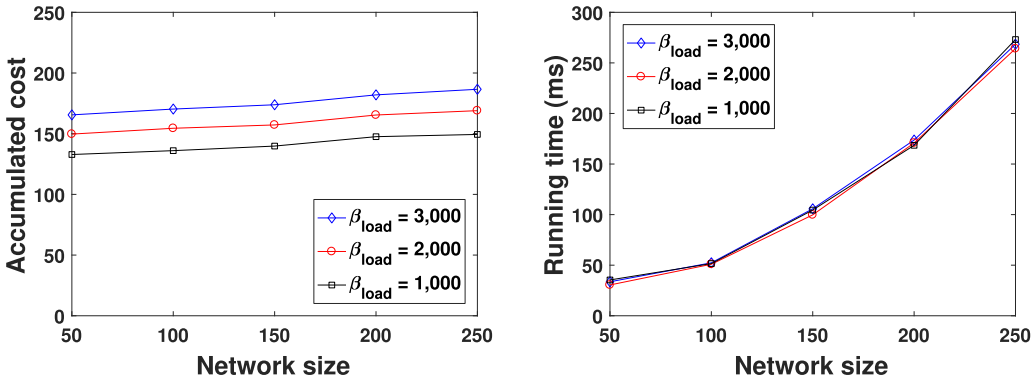


(a) The total cost (b) The running time

Fig. 8. The impact of the number K of sources on the performance of Alg.4.

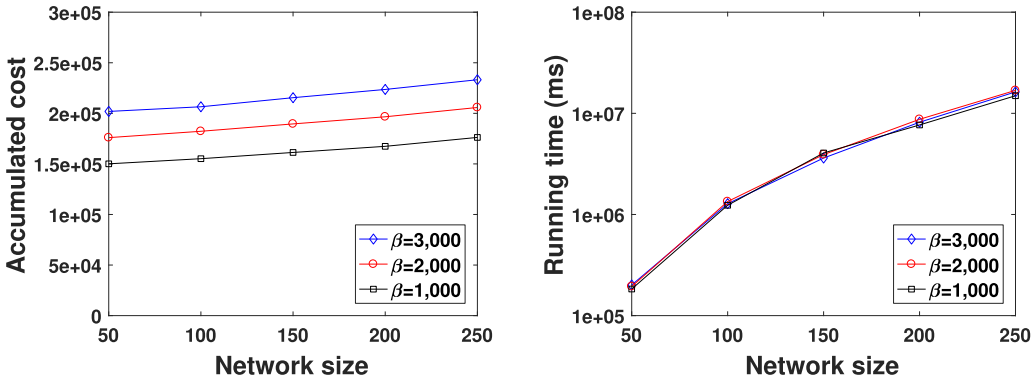
consumes not only less computing resource but also less data rates. Thus, the communication cost, the computing cost, and the workload balancing cost decrease. The similar performance behaviors can be observed in Figure 8(a), too.

We finally evaluated the impact of the coefficient β_{load} of the workload balancing cost on the proposed algorithms. Figures 9(a) and 9(b) show the total cost and running time of algorithm Alg.3 for different network sizes when β_{load} = 1,000, 2,000, and 3,000, respectively. Figures 10(a) and 10(b) depict the total cost and running time of algorithm Alg.3 with 1,000 IoT applications for different network sizes when β_{load} = 1,000, 2,000, and 3,000, respectively. From Figure 9(a), the total cost of algorithm Alg.3 when β_{load} = 1,000 is 80.4% of itself when β_{load} = 3,000. This is because more cost is caused to guarantee the workload balancing among the cloudlets in the MEC network, with the increase on the value of β_{load}. It can also be seen from Figure 9(a) that the performance gap of algorithm Alg.3 from β_{load} = 2,000 to β_{load} = 3,000 is 90.1% of itself from β_{load} = 1,000 to β_{load} = 2,000. The rationale behind this is that a lager β_{load} will lead to a more balanced workload among the cloudlets, thus, the total workload utility (W_v/C_v) of each cloudlet is reduced. The similar performance behaviors can be observed in Figure 10(a), too.



(a) The total cost (b) The running time

Fig. 9. The impact of β_{load} on the performance of Alg.3.



(a) The total cost (b) The running time

Fig. 10. The impact of β_{load} on the performance of Alg.4.

9 CONCLUSION

In this article, we studied the service provisioning in an MEC network for multi-source IoT applications with SFC requirements. We formulated two novel service provisioning problems for multi-source IoT applications: the cost minimization problem of service provisioning for a single multi-source IoT application, and the service provisioning problem of a set of multi-source IoT applications, respectively, and showed that both the problems are NP-hard. We then proposed a novel service provisioning framework in an MEC network for multi-source IoT applications, which includes uploading stream data from multiple sources to the MEC network, in-network data stream aggregation and routing, VNF instance placement and sharing, and workload balancing in the MEC network. We then devised efficient algorithms for the defined problems based upon the built framework. We finally evaluated the performance of the proposed algorithms through experimental simulations. Experimental results demonstrate that the proposed algorithms are promising.

REFERENCES

[1] Georgia Tech. 2019. GT-ITM. Retrieved from http://www.cc.gatech.edu/projects/gtitm/.

[2] Cisco. 2020. Cisco annual internet report (2018–2023) white paper. Retrieved from https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

[3] Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. 2017. Mobile edge computing: A survey. *IEEE Internet Things J.* 5, 1 (2017), 450–465.
[4] Omar Alhussein, Phu Thinh Do, Junling Li, Qiang Ye, Weisen Shi, Weihua Zhuang, Xuemin Shen, Xu Li, and Jaya Rao. 2018. Joint VNF placement and multicast traffic routing in 5G core networks. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM'18)*. IEEE, 1–6.
[5] Alberto Ceselli, Marco Premoli, and Stefano Secci. 2017. Mobile edge cloud network design optimization. *IEEE/ACM Trans. Netw.* 25, 3 (2017), 1818–1831.
[6] Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. 1999. Approximation algorithms for directed Steiner problems. *J. Algor.* 33, 1 (1999), 73–91.
[7] Rami Cohen, Liane Lewin-Eytan, Joseph Seffi Naor, and Danny Raz. 2014. On the effect of forwarding table size on SDN network utilization. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'14)*. IEEE, 1734–1742.
[8] Rami Cohen, Liane Lewin-Eytan, Joseph Seffi Naor, and Danny Raz. 2015. Near optimal placement of virtual network functions. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'15)*. IEEE, 1346–1354.
[9] Hao Feng, Jaime Llorca, Antonia M. Tulino, Danny Raz, and Andreas F. Molisch. 2017. Approximation algorithms for the NFV service distribution problem. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'17)*. IEEE, 1–9.
[10] Andrey Gushchin, Anwar Walid, and Ao Tang. 2015. Scalable routing in SDN-enabled networks with consolidated middleboxes. In *Proceedings of the ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization*. 55–60.
[11] Mike Jia, Jiannong Cao, and Weifa Liang. 2015. Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. *IEEE Trans. Cloud Comput.* 5, 4 (2015), 725–737.
[12] Mike Jia, Weifa Liang, Zichuan Xu, and Meitian Huang. 2016. Cloudlet load balancing in wireless metropolitan area networks. In *Proceedings of the 35th Annual IEEE International Conference on Computer Communications (INFOCOM'16)*. IEEE, 1–9.
[13] Jing Li, Weifa Liang, Meitian Huang, and Xiaohua Jia. 2020. Reliability-aware network service provisioning in mobile edge-cloud networks. *IEEE Trans. Parallel Distrib. Syst.* 31, 7 (2020), 1545–1558.
[14] Jing Li, Weifa Liang, and Yu Ma. 2021. Robust service provisioning with service function chain requirements in mobile edge computing. *IEEE Trans. Netw. Service Manage.* 18, 2 (2021), 2138–2153.
[15] Jing Li, Weifa Liang, Wenzheng Xu, Zichuan Xu, and Jin Zhao. 2020. Maximizing the quality of user experience of using services in edge computing for delay-sensitive IoT applications. In *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'20)*. 113–121.
[16] Jing Li, Weifa Liang, Zichuan Xu, and Wanlei Zhou. 2020. Service provisioning for IoT applications with multiple sources in mobile edge computing. In *Proceedings of the IEEE 45th Conference on Local Computer Networks (LCN'20)*. IEEE, 42–53.
[17] Yuchen Li, Weifa Liang, Wenzheng Xu, and Xiaohua Jia. 2020. Data collection of IoT devices using an energy-constrained UAV. In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS'20)*. IEEE, 644–653.
[18] Yuchen Li, Weifa Liang, Wenzheng Xu, Zichuan Xu, Xiaohua Jia, Yinlong Xu, and Haibin Kan. 2021. Data collection maximization in IoT-sensor networks via an energy-constrained UAV. *IEEE Trans. Mobile Comput.* (2021). https://doi.org/10.1109/TMC.2021.3084972
[19] Yu Ma, Weifa Liang, and Jie Wu. 2019. Online NFV-enabled multicasting in mobile edge cloud networks. In *Proceedings of the IEEE 39th International Conference on Distributed Computing Systems (ICDCS'19)*. IEEE, 821–830.
[20] Yu Ma, Weifa Liang, Jie Wu, and Zichuan Xu. 2019. Throughput maximization of NFV-enabled multicasting in mobile edge cloud networks. *IEEE Trans. Parallel Distrib. Syst.* 31, 2 (2019), 393–407.
[21] Yu Ma, Weifa Liang, Zichuan Xu, and Song Guo. 2018. Profit maximization for admitting requests with network function services in distributed clouds. *IEEE Trans. Parallel Distrib. Syst.* 30, 5 (2018), 1143–1157.
[22] Carla Mouradian, Narjes Tahghigh Jahromi, and Roch H. Glitho. 2018. NFV and SDN-based distributed IoT gateway for large-scale disaster management. *IEEE Internet Things J.* 5, 5 (2018), 4119–4131.
[23] Haozhe Ren, Zichuan Xu, Weifa Liang, Qiufen Xia, Pan Zhou, Omer F. Rana, Alex Galis, and Guowei Wu. 2020. Efficient algorithms for delay-aware NFV-enabled multicasting in mobile edge clouds with resource sharing. *IEEE Trans. Parallel Distrib. Syst.* 31, 9 (2020), 2050–2066.
[24] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge computing: Vision and challenges. *IEEE Internet Things J.* 3, 5 (2016), 637–646.
[25] Yaozhong Song, Stephen S. Yau, Ruozhou Yu, Xiang Zhang, and Guoliang Xue. 2017. An approach to QoS-based task distribution in edge computing networks for IoT applications. In *Proceedings of the IEEE International Conference on Edge Computing (EDGE'17)*. IEEE, 32–39.

[26] Hardik Soni, Walid Dabbous, Thierry Turletti, and Hitoshi Asaeda. 2017. NFV-based scalable guaranteed-bandwidth multicast service for software defined ISP networks. *IEEE Trans. Netw. Service Manage.* 14, 4 (2017), 1157–1170.
[27] Qiufen Xia, Weifa Liang, and Wenzheng Xu. 2013. Throughput maximization for online request admissions in mobile cloudlets. In *Proceedings of the 38th Annual IEEE Conference on Local Computer Networks*. IEEE, 589–596.
[28] Zichuan Xu, Weifa Liang, Meitian Huang, Mike Jia, Song Guo, and Alex Galis. 2018. Efficient NFV-enabled multicasting in SDNs. *IEEE Trans. Commun.* 67, 3 (2018), 2052–2070.
[29] Zichuan Xu, Weifa Liang, Wenzheng Xu, Mike Jia, and Song Guo. 2015. Efficient algorithms for capacitated cloudlet placements. *IEEE Trans. Parallel Distrib. Syst.* 27, 10 (2015), 2866–2880.
[30] Zichuan Xu, Zhiheng Zhang, Weifa Liang, Qiufen Xia, Omer Rana, and Guowei Wu. 2020. QoS-aware VNF placement and service chaining for IoT applications in multi-tier mobile edge networks. *ACM Trans. Sensor Netw.* 16, 3 (2020), 1–27.
[31] Ruozhou Yu, Guoliang Xue, and Xiang Zhang. 2018. Application provisioning in fog computing-enabled internet-of-things: A network perspective. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'18)*. IEEE, 783–791.
[32] Sai Qian Zhang, Qi Zhang, Hadi Bannazadeh, and Alberto Leon-Garcia. 2015. Network function virtualization enabled multicast routing on SDN. In *Proceedings of the IEEE International Conference on Communications (ICC'15)*. IEEE, 5595–5601.

Received February 2021; revised May 2021; accepted August 2021