NFV-Enabled IoT Service Provisioning in Mobile Edge Clouds

Zichuan Xu, *Member, IEEE*, Wanli Gong, Qiufen Xia, *Member, IEEE*, Weifa Liang, *Senior Member, IEEE*, Omer F. Rana, *Senior Member, IEEE*, and Guowei Wu

Abstract—Conventional Internet of Things (IoT) applications involve data capture from various sensors in environments, and the captured data then is processed in remote clouds. However, some critical IoT applications (e.g., autonomous vehicles) require a much lower response latency and more secure guarantees than those offered by remote clouds today. Mobile edge clouds (MEC) supported by the network function virtualization (NFV) technique have been envisioned as an ideal platform for supporting such IoT applications. Specifically, MECs enable to handle IoT applications in edge networks to shorten network latency, and NFV enables agile and low-cost network functions to run in low-cost commodity servers as virtual machines (VMs). One fundamental problem for the provisioning of IoT applications in an NFV-enabled MEC is where to place virtualized network functions (VNFs) for IoT applications in the MEC, such that the operational cost of provisioning IoT applications is minimized. In this paper, we first address this fundamental problem, by considering a special case of the IoT application placement problem, where the IoT application and VNFs of each service request are consolidated into a single location (gateway or cloudlet), for which we propose an exact solution and an approximation algorithm with a provable approximation ratio. We then develop a heuristic algorithm that controls the resource violation ratios of edge clouds in the network. For the IoT application placement problem for IoT applications where their VNFs can be placed to multiple locations, we propose an efficient heuristic that jointly places the IoT application and its VNFs. We finally study the performance of the proposed algorithms by simulations and implementations in a real test-bed, Experimental results show that the performance of the proposed algorithms outperform their counterparts by at least 10 percent.

Index Terms—Mobile edge clouds, network function virtualization, Internet of Things (IoT) application, approximation algorithms, VNF placement, algorithm design

1 Introduction

Internet of Things (IoT) as the interconnection of intelligent devices and management platforms that collectively enable the "smart world" has been envisioned as an enabling technology that can transfer various aspects of people's daily lives from wellness and health monitoring to smart utility meters [7], [38], [43], [44], [61]. Existing IoT platforms usually offload the generated data from heterogeneous IoT nodes to their applications in remote data centers for storage and analysis, through network transmission services leased from network service providers. To ensure the security and privacy of the offloaded data, network service providers install various hardware-based network functions (middleboxes) in gateways or switches of their operational network infrastructures. This however may degrade the Quality of Service (QoS) requirements of IoT applications due to a prohibitive transmission delay from the IoT nodes to remote data centers. Furthermore, these installed hardware-based network functions are expensive and inflexible to manage, thereby preventing new services for fast deployments and significantly increasing development cycles of new IoT applications [15], [16].

Mobile edge cloud (MEC) computing and network function virtualization (NFV) technologies enhance the QoS of users [15], [16], [17], [58], [59], by moving computing resources into mobile access networks within the proximity of IoT nodes. Further, the adoption of NFV can reduce the usage cost of network functions by shifting the implementations of network functions from hardware to software that run in Virtual Machines (VMs). Also, distributing intelligence throughout the mobile edge cloud enables real-time analytics and business intelligence to be possible, through the provisioning of a collection of virtualized network functions (VNFs) such as gateways, mobile core, deep packet inspection (DPI), security, routing, and traffic.

In this paper, we aim to develop algorithmic techniques to provide efficient, secure and agile data processing services for IoT applications in an MEC, through finding strategic locations in the MEC for both IoT applications and VNFs placements. One challenge is how to offload the data of IoT devices to their IoT applications. Usually, this is

- Z. Xu, W. Gong, and G. Wu are with the School of Software, and the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian University of Technology, Dalian 116024, China. E-mail: {z.xu, wgwdut}@dlut.edu.cn, 651767079@qq.com.
- Q. Xia is with the International School of Information Science and Engineering, and the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian University of Technology, Dalian 116024, China. E-mail: qiufenxia@dlut.edu.cn.
- W. Liang is with the Research School of Computer Science, Australian National University, Canberra, ACT 2601, Australia. E-mail: wliang@cs.anu.edu.au.
- O.F. Rana is with the Cardiff University, CF10 3AT Cardiff, United Kingdom. E-mail: RanaOF@cardiff.ac.uk.

Manuscript received 8 Apr. 2019; revised 16 Jan. 2020; accepted 28 Jan. 2020. Date of publication 10 Feb. 2020; date of current version 2 Apr. 2021. (Corresponding author: Qiufen Xia.) Digital Object Identifier no. 10.1109/TMC.2020.2972530

addressed by including gateways to bridge communication networks and IoT devices. Traditional gateways however are hardware based, and only perform built-in functions. In contrast, we consider a scenario where each gateway is attached to a physical server and VMs are instantiated in the server to implement gateway functions and the other network functions. An example of such gateways is the smart IoT gateway powered by Intel Movidius Vision Accelerator [32], [34]. Another challenge is how to jointly find appropriate locations for IoT applications and VNFs in the MEC, such that the cost of offloading data from IoT nodes to their services is minimized, while the QoS requirements of these applications are met.

Existing studies on NFV in literature often assume that the locations of IoT applications are fixed and given as a priori [6], [11], [22], [25], [27], [30], [36], [40], [51]. Many of them considered the placement of consolidated VNFs of a network service or chaining a sequence of VNFs into different locations. They assumed that the data traffic of user requests need to be transferred from their sources to destinations, and the data traffic is processed by the VNFs in the specified order before reaching their destinations. In contrast, IoT applications are the destinations of IoT data transmission, while these destinations are not given and need to be strategically identified by developing algorithms. Therefore, existing solutions for user data transfers in MEC thus cannot be directly applied to our problem.

To the best of our knowledge, we are the first to consider NFV-enabled IoT application provisioning in an MEC, by considering the joint placement of both IoT applications, VNFs and traffic routing finding.

The main contributions of this paper are as follows.

- We formulate the IoT application placement problem in an MEC with the aim to minimize the implementation cost of NFV-enabled requests, subject to the computing resource capacity constraints of each gateway node and each cloudlet.
- We consider a special case of the IoT application placement problem in an MEC with the IoT application and VNF demanded by each request being consolidated into a single cloudlet/gateway, we propose two approximation algorithms with provable approximation ratios for them with and without network bandwidth capacity constraints.
- We propose an efficient heuristic for the IoT application placement problem in an MEC, by assuming the IoT application and VNF requested can be placed into different locations.
- We evaluate the performance of the proposed algorithms through simulations and experiments in a real test-bed. Experimental results demonstrate that the performance of the proposed algorithms outperform existing solutions.

The remainder of the paper is arranged as follows. Section 2 will survey the state-of-the-art on this topic, and detail the difference of this work from previous studies of NFV-enabled IoT application provisioning. Section 3 will introduce the system model, notations and problem definition. Section 4 will propose two approximation algorithms with provable approximation ratios for the problem with and without network bandwidth capacity constraints. Section 5 will propose an efficient heuristic for the IoT application placement problem, by allowing the IoT application and its VNF can be placed to multiple locations. Section 6 will provide experimental results on the performance of the proposed algorithms. Section 7 will give the implementation of the proposed algorithms in a real test-bed, and Section 8 concludes the paper.

2 Related Work

With the advancement of networking and 5G communication technologies, IoT has been emerging as an enabling technology that can bring remarkable transformation in almost every domain of human life. The rise of various IoT applications has notably increased the potential security attacks. IoT applications thus adopt different VNFs to guarantee the secure, flexible, agile, and low-cost data processing. On the other hand, although IoT applications have experienced great success in data center networks (clouds), the long access delays of these IoT applications in remote clouds are too high to meet the real-time response demands of many industrial IoT applications. IoT application providers are embracing mobile edge clouds to reduce service access delays, thereby meeting the delay requirements of users. In the following, we first summarize the research advances on NFV. We then describe the efforts in the use of NFV in MEC. We finally elaborate on recent advances in NFV-enabled IoT applications.

Recently, notable efforts have been addressed to provide VNFs for various network services, aiming to reduce the capital and operational costs of network service providers while enhancing the service security [6], [11], [22], [25], [27], [30], [36], [40], [51]. In particular, the techniques of SDN and NFV are expected to enable network service providers to use software to set up, configure, and manage network services in their networks automatically and dynamically. There are studies focusing on the placement of VNFs [13], [40], traffic steering given placed network functions [46], joint traffic steering and VNF placement [30], programming abstractions for NFV [27], and dynamic network function chaining [53]. For example, Sekar *et al.* [51] designed an architecture for the deployment of consolidated middleboxes with the aim to minimize network provisioning cost and to improve network imbalance. Qazi *et al.* developed SIMPLE [46] that enforces high-level routing policies for middlebox-specific traffic, they however did not consider virtualization or dynamic network function placements. Fayazbakhsh *et al.* proposed FlowTags [18] for flow scheduling in a network in the presence of dynamic modifications performed by middleboxes. Gember *et al.* [22] designed and implemented an orchestration layer for virtual middlebox, to enable dynamic, efficient, and scalable provisioning of middleboxes. Martins *et al.* [40] introduced a platform to improve network performance, by revising existing virtualization technologies to support the deployment of modular, virtual middleboxes on lightweight VMs. Gupta *et al.* [27] studied the wide-area traffic delivery in Internet exchange points, by implementing a new programming abstractions to create and run new applications and guaranteeing the scalability of the system in terms of both rule-table size and computational overhead. Cheng *et al.* [11] devised a description-language to help service providers to develop instances of network functions and proposed an exact for the service instantiation problem. Qu *et al.* [48] studied the problem of delay-aware scheduling and resource optimization with NFV in a virtual network. Wang *et al.* [53] studied the problem of dynamic network function composition, and proposed a distributed algorithm, using Markov approximation method for the problem. Huang *et al.* [30] studied the problem of jointly routing user requests and placing their required network functions to some servers in a data center, with the aim to maximize the network throughput while meeting various capacity constraints of the network and the end-to-end delay requirement of each user requests.

There are several studies of investigating the provisioning of NFV-enabled networks in data centers and mobile edge clouds [25], [36], [56], [60]. For example, Li *et al.* [36] aimed to provide real-time guarantees for user requests in a data center. Gu *et al.* [25] investigated dynamic service chaining in an NFV market of a single data center, by devising efficient and truthful auction mechanisms and assuming some of the instantiated network functions can be reused by later requests. In particular, Xu *et al.* [56] formulated a task offloading problem in a mobile edge-cloud network, where each offloaded task requests a specific network function with a tolerable delay. Efficient heuristics and an online algorithm with a competitive ratio are devised. Yang *et al.* [60] addressed the questions on where and when to allocate resources as well as how many resources to be allocated among NFV-enabled mobile edge clouds, such that both the low latency requirements of mobile services and cost efficiency are achieved. These studies however assumed that either the locations of applications are given or do not consider the joint placement of VNFs and applications.

Most existing work on NFV focused on either communication or cloud networks, and there are a few studies on provisioning IoT applications that concentrate on either light weight virtualization techniques, architecture design [1], software-defined wireless sensor networks [7], [20], [38], or application provisioning [3], [4], [14], [29], [43], [44], [61]. For example, to enable VNFs running in IoT gateways, we need light weight implementation of VNFs. Several efficient methods and implementations of VNFs in IoT gateways have been proposed [4], [14], [29]. Yu *et al.* [61] studied the problem of IoT application provisioning, with the objective to meet computing, network bandwidth and QoS requirements of IoT applications. VNFs however are not enabled in the IoT applications. Recently, the design of novel NFV architectures has attracted much attention [9], [26], [43], [45], [49]. For example, Mouradian *et al.* [43] proposed an architecture of NFV-based distributed IoT gateways in a software-defined network (SDN) for large-scale disaster management. Joint VNF placement and IoT application placement however are not the focus of those studies. Based on those NFV architectures, novel methods and algorithms are needed to enable flexible and agile IoT services, such that the IoT services can be designed, managed and deployed in an efficient and effective way. This paper tries to fill this gap by proposing novel VNF provisioning for IoT in mobile edge clouds.

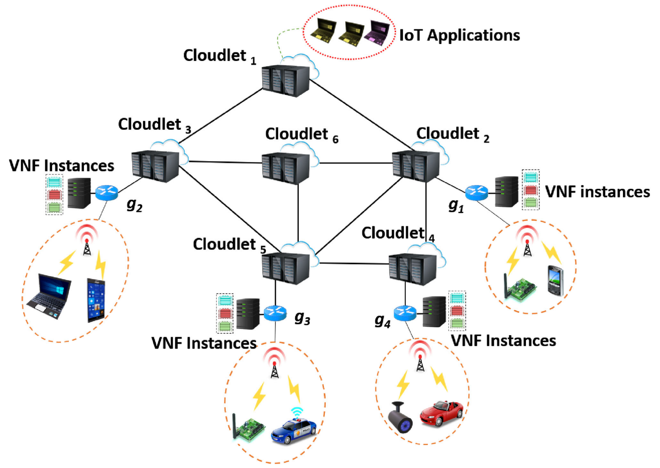


Fig. 1. An example of the mobile edge network with IoT applications.

3 Preliminaries

In this section we first define the system model and notations, and then precisely define the problems.

3.1 System Model

We consider a mobile edge cloud $G = (V \cup \mathcal{GW}, E)$, where V is the set of *cloudlets* with each having a number of servers, $\mathcal{GW}$ denotes the set of gateways with each being responsible for a set of IoT nodes (i.e., an IoT domain), and E is the set of wired links that interconnect gateways and cloudlets and the wireless links that interconnect IoT nodes and their gateways. Let v_i be a cloudlet in V, and each v_i has a capacitated computing resource to implement VNFs and IoT applications. Each link e has bandwidth capacity $B(e)$, and B_{unit} amount of bandwidth resource in it is assigned to transfer a unit amount of data traffic. Denote by g_k the kth gateway node in $\mathcal{GW}$. Each gateway node g_k usually has larger resource capacities than that of an IoT node. It therefore can host a certain set of light-weight VNFs to process the traffic of IoT nodes. In addition, each gateway is usually co-located with an Access Point (AP) that provides wireless access to IoT nodes [44].

We target IoT applications that need to consistently process data streams, such as environmental monitoring, building health monitoring, and smart plants applications. Such applications need not only consistent traffic routing from IoT nodes to their applications but also timely data processing. An IoT application and its VNF may be instantiated in a gateway or a cloudlet. For simplicity, we assume that Loc_m ($\in V \cup \mathcal{GW}$) represents a potential location for IoT applications and VNF instances. Denote by $C(Loc_m)$ the amount of computing resource available in cloudlet v_i or gateway node g_k to implement VNF instances and IoT applications. Let C_{unit} be the amount of computing resource needed to process a unit amount of data traffic. An example of IoT applications in the MEC is shown in Fig. 1.

3.2 NFV-Enabled Requests and IoT Applications

To guarantee that data is processed in a timely and secure way, IoT application providers usually deploy VNFs in gateway nodes or cloudlets within the proximity of IoT nodes. For example, a network function of data

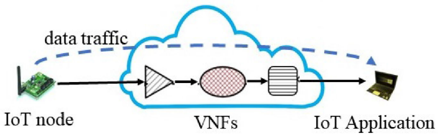


Fig. 2. An example of a NFV-enabled request r_l.

preprocessing may be deployed to perform stratified sampling of collected data. Also, an IDS may be deployed into one of the cloudlets to identify possible intrusions from malicious IoT nodes. Let n_l be an IoT node. To make sure that its data traffic is processed by an instance of VNF_l, it needs to send its data traffic to one of its nearby gateway nodes, and then the data traffic is forwarded to a cloudlet with VNF_l for processing. Its data traffic then will be analyzed by its IoT application.

Let sv_l be the IoT application requested by IoT node n_l. We assume that each IoT application sv_l is implemented in a VM of a cloudlet. The locations of IoT applications are vital for the performance of the IoT applications. For example, if an IoT application is located in a cloudlet far from its IoT node, its data traffic has to be routed to it via a longer path in the MEC. Since each link in the routing path has to reserve an amount of bandwidth for its data transfer, it consumes more bandwidth in general. Eventually, this may reduce the number of requests admitted, considering that the bandwidth resource in the MEC is limited. On the other hand, if an IoT application is located in a gateway node of the IoT domain where its IoT nodes reside, the gateway node may be overloaded making the other IoT nodes in the same domain wait for their traffic being processed.

Denote by r_l the NFV-enabled request by IoT node n_l. It requests to transfer an amount b_l of data from a source node n_l to its IoT application sv_l. Each NFV-enabled request thus is represented by $r_l = (n_l, sv_l; VNF_l, b_l)$. Fig. 2 shows an example of a NFV-enabled request.

3.3 Costs of Implementing IoT Applications and VNFs

IoT applications consume computing and bandwidth resources for the processing and transmission of traffic. We thus consider that the implementation cost of each request includes the *processing cost*, *transmission cost*, and *energy cost*. Notice that we consider a mobile edge cloud that is operated by an IoT service provider. The provider has its own IoT devices; it however may lease resources from cloud service providers. For the operation of its IoT devices, the energy cost is mainly due to the electricity cost to power them, while the processing and transmission costs are due to the usage of the resources leased from the cloud service provider. This means that all the three costs can be considered as monetary costs.

Recall that an amount of b_l data needs to be processed by VNF_l before being analyzed by its IoT application sv_l. The processing cost is proportional to the volume of data that will be processed. Let $c_{l,m}$ be the cost of processing unit volume of data traffic by VNF_l at location Loc_m (a cloudlet v_i or a gateway g_k). Similarly, denote by $c^{sv}_{l,m}$ the cost of processing unit volume of data traffic by IoT application sv_l in potential location Loc_m. Denote by $y_{l,m}$ the indicator variable indicating whether location Loc_m implements network function VNF_l of request r_l. Let $y'_{l,m}$ be an indicator variable showing whether potential location Loc_m hosts IoT application sv_l of r_l. The processing cost of r_l thus is

$$c_p(r_l) = \sum_{Loc_m \in V \cup \mathcal{GW}} b_l \cdot \big(y_{l,m} \cdot c_{l,m} + y'_{l,m} \cdot c^{sv}_{l,m}\big). \tag{1}$$

Before analyzing data b_l of request r_l, the data has to be transferred from its IoT node n_l to its application sv_l. The transmission cost of r_l is proportional to the volume that is transmitted along the path from the IoT node n_l of r_l to the location of its IoT application sv_l. It must be mentioned that this link can be a wireless link from the IoT node to its nearby gateway node and a wired link from the gateway to the cloudlet with its IoT application. If it is a wireless link, energy cost can be consumed due to the sending of data from the IoT node to its nearby gateway. Since the IoT node only collects data and transmits data, its energy consumption is mainly due to the data transmission. We thus consider this as a part of the transmission cost. Assuming that the bandwidth of the gateway of r_l is B_l, the achieved data rate W_l (bits per second) via the wireless channel of the gateway for r_l is

$$W_l = B_l \log_2\Big(1 + \frac{p_l \cdot h_l}{\sigma^2 + I_l}\Big), \tag{2}$$

where σ^2 is the noise power of mobile devices and I_l is the inter-cell interference power [10], h_l is the channel gain between the IoT node n_l and its gateway, and p_l is the transmission power of IoT node n_l. The energy cost spent in transmission thus is

$$c_{energy}(r_l) = w_e \cdot p_l \cdot (b_l / W_l), \tag{3}$$

where w_e is a given constant that captures the monetary cost of per unit of energy consumption.

After the data of r_l being forwarded to its nearby gateway, its further transmission in the MEC incurs costs as well. Let c_e be the cost of transmitting a unit volume of data traffic along edge $e \in E$. Assume that $z_{e,l}$ indicates whether the traffic of request r_l is transferred via link $e \in E$. Denote by $c_t(r_l)$ the traffic transmission cost of r_l, which can be calculated by

$$c_t(r_l) = c_{energy}(r_l) + \sum_{e \in E} z_{e,l} \cdot c_e \cdot b_l. \tag{4}$$

The total cost of implementing an NFV-enabled request is

$$c(r_l) = c_p(r_l) + c_t(r_l). \tag{5}$$

3.4 Problem Definition

Given a mobile edge cloud $G = (V \cup \mathcal{GW}, E)$, a historical trace of NFV-enabled requests $\mathcal{R}$, assuming that the capacities of the accumulative computing and bandwidth resources are larger than the total demand of requests in $\mathcal{R}$, *the IoT application placement problem in a mobile edge cloud* is to find appropriate locations for both IoT applications and the VNFs of the requests in G, such that the weighted sum of computing and bandwidth resource consumptions is

TABLE 1
Symbols

Symbols	Meaning
$G = (V \cup \mathcal{GW}, E)$	a mobile edge cloud with a set V of switches, a set of gateway nodes, and a set E of links
v_i	a cloudlet in V
e	a link in E
$B(e)$	the capacity of the available bandwidth resource in link e
B_{unit}	the amount of bandwidth resource that is needed to transfer a unit amount of data
g_k	the kth gateway node in $\mathcal{GW}$
Loc_m	a location that can be a cloudlet in V or a gateway node in $\mathcal{GW}$
$C(Loc_m)$	the amount of computing resource that is available in cloudlet v_i or gateway node g_k
C_{unit}	the amount of computing resource that is needed to process a unit amount of data traffic
n_l	an IoT node
VNF_l	the VNF demanded by n_l to process its traffic
sv_l	the IoT application demanded by IoT node n_l
r_l	the NFV-enabled request by IoT node n_l
b_l	the amount of data that request r_l requests to transfer from n_l to sv_l
$c_{l,m}$	the cost of processing unit volume of data traffic by VNF_l at location Loc_m
$c^{sv}_{l,m}$	the cost of processing unit volume of data traffic by IoT application sv_l
$c_p(r_l)$	the cost of processing request r_l
$y'_{l,m}$	an indicator variable showing whether location Loc_m hosts IoT application sv_l
$z_{e,l}$	an indicator variable indicates whether the traffic of request r_l is transferred via link $e \in E$
$c_t(r_l)$	the traffic transmission cost of r_l
$\mathcal{R}$	a set of NFV-enabled requests
$y_{l,m}$	an indicator variable shows whether sv_l and VNF_l are consolidated in a single location Loc_m in the IoT application placement problem with consolidated IoT applications and VNFs
$q_{m,l}$	a binary indicator variable that shows whether the switch of each Loc_m is used to forward the traffic of r_l
$z_{e,l}$	a binary variable that indicates whether edge $e \in E$ is used to route the traffic of r_l
$\delta(v)$	the incident edges of switch node $v \in V$
p_{n_l,Loc_m}	the shortest path between IoT node n_l to potential location Loc_m
y^*	the optimal solution to **LP**
$y^*_{m,l}$	a partial assignment of request r_l to location Loc_m
c^*_l	the optimal cost of implementing request r_l due to the fractional solution obtained from the **LP**, i.e., y^*
ω^*_l	the maximum ratio of the partially assigned demand of requests to the capacities of a location in solution y^*
$\mathcal{L}_l$	the set of candidate locations for each request r_l
ϵ and η	constants with values in the range of $(0, 1]$
ϑ and ξ	scaling factors with their values greater than 1
$\hat{y}$	the solution obtained due to algorithm `Appro_Consolidated`
$Can_{vnf,l}$	the set of the pruned candidate set for VNF_l of r_l in algorithm `Heuristic`

minimized, subject to the computing resource constraint on each gateway or cloudlet, and the bandwidth resource constraint of each link in E. In other words, to construct an end-to-end IoT application, we need to provide efficient solutions for the IoT application placement problem that jointly (1) place IoT applications and VNFs, and (2) find a path from IoT devices to the placed VNFs and IoT applications to transfer gathered data.

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

4 Algorithms for the IoT Application Placement Problem With Consolidated IoT Applications and VNF Placements

We here propose exact and approximate solutions for a special IoT application placement problem in an MEC, where both the IoT application and the VNFs of each admitted request are consolidated into a single location.

4.1 Exact Solution

Since both the IoT application and VNF of each request r_l are consolidated to a single cloudlet or gateway, we use an indicator variable $y_{l,m}$ to denote whether sv_l and VNF_l are consolidated into location $Loc_m \in V \cup \mathcal{GW}$. Denote by $q_{m,l}$ a binary indicator variable that shows whether the switch of each Loc_m is used to forward the traffic of r_l. Let $\delta(v)$ denote the incident edges of switch node $v \in V$, respectively. Let $z_{e,l}$ be a binary variable that indicates whether edge $e \in E$ is used to route the traffic of r_l. ILP The objective of the proposed integer linear program (ILP) thus can be formulated as

$$\textbf{ILP :}\min \sum_{r_l \in R} \sum_{Loc_m \in V \cup \mathcal{GW}} b_l\Big(\cdot y_{l,m}\big(c_{l,m} + c^{sv}_{l,m}\big)\Big) + c_{energy}(r_l) + \sum_{e \in E} z_{e,l} c_e \ , \tag{6}$$

subject to the following constraints:

$$\sum_{Loc_m \in V \cup \mathcal{GW}} y_{l,m} = 1, \qquad \text{for each } r_l \in R \tag{7}$$

$$\sum_{r_l \in R} y_{l,m} \cdot b_l \cdot C_{unit} \le C(Loc_m), \quad \text{for each } Loc_m \tag{8}$$

$$\sum_{r_l \in R} z_{e,l} \cdot b_l \cdot B_{unit} \leq B(e), \quad \text{for each } e \in E \tag{9}$$

$$q_{m,l} \leq 1, \quad \text{for each } VNF_l \text{ and each } Loc_m \in V \cup \mathcal{GW} \tag{10}$$

$$\sum_{e \in \delta(Loc_m)} z_{e,l} \leq 2 \cdot q_{m,l}, \text{ for each } Loc_m \text{ and each } r_l \tag{11}$$

$$\sum_{e \in \delta(n_l)} z_{e,l} = 1, \quad \text{for } n_l \text{ of each } r_l \tag{12}$$

$$\sum_{e \in \delta(Loc_m)} z_{e,l} = y_{l,m}, \text{ for each potential location } Loc_m \tag{13}$$

$$y_{l,m}, z_{e,l}, q_{m,l} \in \{0, 1\}, \tag{14}$$

where Constraint (7) ensures that each request r_l has to be admitted and only one location is selected for its IoT application sv_l and VNF_l, because the consolidated VNF and IoT application of r_l cannot be split into different locations. Constraint (8) guarantees that the capacity of each potential location (a cloudlet or a gateway node) is not violated by its assigned IoT applications and VNF instances. Constraint (9) ensures that the bandwidth resource consumed by all the requests that use link e does not exceed its capacity $B(e)$, where the amount of bandwidth resource B_{unit} is assigned to process a unit amount of data. Constraint (10) says that the switch of each potential location may or may not forward the traffic of r_l. Constraint (11) shows that if the switch of a location routes the data b_l of r_l via its incident edges (i.e., $q_{m,l} = 1$), there are at most two of its incident edges that are used for data forwarding (one for incoming traffic and the other for outgoing traffic). Constraint (12) ensures that the data of each request r_l goes out its IoT node n_l, whereas Constraint (13) says that the final destination of the data is the placed VNF and IoT application when they are placed to Loc_m (i.e., $y_{l,m}$). Constraint (14) implies that indicator variables $y_{l,m}$, $z_{e,l}$, and $q_{m,l}$, can be either 1 or 0.

4.2 Approximation Algorithm for the Problem Without the Bandwidth Requirement

The proposed exact solution is not scalable for large problem sizes, due to the fact that the IoT application placement problem is NP-Hard. We thus propose an efficient and scalable approximate solution to the problem.

The basic idea behind the approximation algorithm is to formulate the problem without bandwidth requirement into an ILP. We then relax the ILP to a Linear Program (LP). Although there is a polynomial solution to the LP, the obtained 'fractional' solution is not a feasible solution to the original problem. It however provides a very good base of constructing an approximate solution whose performance is not far from the optimal one. To guarantee the performance of the approximate solution, we adopt an elegant filtering-and-rounding method [37]. The method filters out 'bad' fractional placements of the VNF and IoT application while preserving the good ones, during the rounding process based on the fractional solution. An approximate integral solution to the problem is finally produced.

ILP Formulation and Relaxation. Similar to *ILP* formulation in (15), we use an indicator variable $y_{l,m}$ to indicate whether sv_l and VNF_l are consolidated into location $Loc_m \in V \cup \mathcal{GW}$. Since we do not consider bandwidth capacity of each link, the data traffic of each request is transferred via a shortest path. Let p_{n_l,Loc_m} be the shortest path between IoT node n_l to potential location Loc_m. The objective of the problem thus can be formulated as

$$\begin{aligned} \textbf{ILP2 :} \min & \sum_{r_l \in R} c_{energy}(r_l) \\ & + \sum_{Loc_m \in V \cup \mathcal{GW}} b_l \Big(\cdot y_{l,m} (c_{l,m} + c^{sv}_{l,m}) + \sum_{e \in p_{n_l,Loc_m}} c_e \Big), \end{aligned} \tag{15}$$

subject to Constraints (7), (8), and

$$y_{l,m} \in \{0, 1\}. \tag{16}$$

The algorithm first relaxes Constraint (16), by considering $y_{l,m}$ as a real value in the range of $[0, 1]$. Let *LP* be the relaxed version of *ILP2*, and it has the same objective as *ILP2*, subject to Constraints (7), (8), and

$$0 \leq y_{l,m} \leq 1. \tag{17}$$

Let y^* be the optimal solution to the *LP*, which is a fractional solution. We can interpret a fractional $y^*_{m,l}$ as a partial assignment of request r_l to location Loc_m. Based on this fractional solution, we then round it to an integer solution, by adopting the filtering-and-rounding optimization technique. In the following we describe the filtering-and-bounding steps of the proposed algorithm for the IoT application placement problem in a MEC.

The Filtering Step. Request r_l is partially assigned to location Loc_m if $0 < y^*_{m,l} < 1$. If we assign request r_l to one of the locations with $y^*_{m,l} > 0$, we may get a high implementation cost or the capacity of a location may be violated too much, since the request can be split into multiple locations in solution y^*. To avoid such situations, we filter out those locations. Specifically, we fix parameters ϵ and η to control the filtering process. Let c^*_l be the current optimal cost of implementing request r_l, which can be calculated by

$$\begin{aligned} c^*_l = & \, c_{energy}(r_l) \\ & + \sum_{Loc_m \in V \cup \mathcal{GW}} y^*_{m,l} \cdot b_l \cdot \Big(c_{l,m} + c^{sv}_{l,m} + \sum_{e \in p_{n_l,Loc_m}} c_e \Big). \end{aligned}$$

Denote by ω^*_l the maximum ratio of the partially assigned demand of requests to the capacities of a location in the optimal solution to the *LP*, i.e.,

$$\omega^*_l = \arg \max_{Loc_m \in V \cup \mathcal{GW}} \frac{b_l \cdot C_{unit} \cdot y^*_{m,l}}{C(Loc_m)}. \tag{18}$$

To filter the locations that may incur high implementation costs or high violation of computing capacity of a cloudlet or data center, we denote by $\mathcal{L}_l$ the set of candidate locations for each request r_l, i.e.,

$$\mathcal{L}_l = \Big\{Loc_m \mid y^*_{m,l} > 0, \\ c_{energy}(r_l) + b_l \cdot (c_{l,m} + c^{sv}_{l,m} + \sum_{e \in p_{n_l, Loc_m}} c_e) \leq (1+\epsilon) \cdot c^*_l, \\ \frac{b_l \cdot C_{unit}}{C(Loc_m)} \leq (1+\eta) \cdot \omega^*_l \Big\}. \tag{19}$$

Using the mentioned filtering method, we now construct a feasible solution y' to the *LP*, such that the following conditions are met:

- *(1):* The cost of y' is not far from the optimal cost y^* to the *LP*;
- *(2):* The contribution to strengthening the capacity constraint is no greater than $(1+\eta) \cdot \omega^*_l$;
- *(3):* $y'_{l,m}$ implies that $c_{energy}(r_l) + b_l \cdot (c_{l,m} + c^{sv}_{l,m} + \sum_{e \in p_{n_l, Loc_m}} c_e) \leq (1+\epsilon) \cdot c^*_l$.

Notice that conditions (2) and (3) also mean that when

$$\frac{b_l \cdot C_{unit}}{C(Loc_m)} > (1+\eta) \cdot \omega^*_l, \tag{20}$$

and

$$c_{energy}(r_l) + b_l \cdot (c_{l,m} + c^{sv}_{l,m} + \sum_{e \in p_{n_l, Loc_m}} c_e) > (1+\epsilon) \cdot c^*_l, \tag{21}$$

we have $y'_{l,m} = 0$.

We now construct the feasible solution, by defining $y'_{m,l}$ based on $y^*_{m,l}$. Obviously, we have $\sum_{Loc_m \in V \cup \mathcal{GW}} y'_{m,l} = 1$. Considering only locations in $\mathcal{L}_l$ to be considered as the candidate locations for request r_l, we define $y'_{m,l}$ by

$$y'_{m,l} = \begin{cases} \frac{y^*_{m,l}}{\sum_{Loc_m \in \mathcal{L}_l} y^*_{m,l}} & \text{if } Loc_m \in \mathcal{L}_l \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

The Bounding Step. We round solution y' to a feasible solution to **ILP2** as follows.

We pick a unassigned request r_l with the smallest implementation cost c^*_l. For this request r_l, we assign it to a location $Loc_{\theta(l)}$ in $\mathcal{L}_l$ with the smallest implementation cost, that is, we round $y'_{l,\theta(l)}$ to 1. For each of all other requests $r_{l'}$, if $\mathcal{L}_l \cap \mathcal{L}_{l'} \neq \emptyset$ and assigning $r_{l'}$ to location $Loc_{\theta(l)}$ does not violate the computing capacity constraint of $Loc_{\theta(l)}$, round $y'_{l',\theta(l)}$ to 1. This procedure continues until all requests are assigned.

The detailed algorithm is given in Algorithm 1.

4.3 A Heuristic for the Problem With the Bandwidth Requirement

So far we have not considered the bandwidth requirement of the IoT application placement problem in an MEC $G = (V \cup \mathcal{GW}, E)$. Applying algorithm `Appro_Consolidated` directly to the IoT application placement problem may violate the bandwidth requirement of some links in the mobile edge cloud. We now propose an efficient heuristic that consider the bandwidth constraint for the problem with IoT application and VNF being consolidated together, based on the proposed approximation algorithm in `Appro_Consolidated`.

Algorithm 1. `Appro_Consolidated`

Input: $G = (V \cup \mathcal{GW}, E)$, a set of NFV-enabled requests $\mathcal{R}$, computing capacity $C(Loc_m)$ for each potential location (either a cloudlet $v_i \in V$ or gateway node $g_k \in \mathcal{GW}$).

Output: The assignment of the IoT application sv_l and VNF_l of each request r_l to a cloudlet or a gateway node.

1: Solve the **LP**.
2: Let y^* be the optimal solution due to **LP**;
3: Filter the solution based on the optimal solution y^* to the **LP**, as shown in Eq. (19);
4: Construct a feasible solution y' to the **LP**, according to Eq. (22);
5: Pick a unassigned request r_l with the smallest implementation cost c^*_l. For this request r_l, we assign it to a location $Loc_{\theta(l)}$ in $\mathcal{L}_l$ with the smallest implementation cost, that is, we round $y'_{l,\theta(l)}$ to 1;
6: For each of all other requests $r_{l'}$, if $\mathcal{L}_l \cap \mathcal{L}_{l'} \neq \emptyset$, round $y'_{l',\theta(l)}$ to 1;
7: Repeat steps from 6 to 8 until all requests are assigned;

The steps of the proposed heuristic algorithm are similar to those of algorithms `Appro_Consolidated`, except the bounding step. Specifically, we pick an unassigned request r_l with the smallest implementation cost c^*_l. For this request r_l, we assign it to a location $Loc_{\theta(l)}$ in $\mathcal{L}_l$ with the smallest implementation cost. That is, we round $y'_{l,\theta(l)}$ to 1. For each of all other requests $r_{l'}$, if $\mathcal{L}_l \cap \mathcal{L}_{l'} \neq \emptyset$, round $y'_{l',\theta(l)}$ to 1 if the location meet the following conditions:

- The amount of available computing resource in $Loc_{\theta(l)}$ is enough to implement request $r_{l'}$
- There exists a path from n_l to $Loc_{\theta(l)}$ that has at least an amount $\xi \cdot b_l \cdot B_{unit}$ of available bandwidth resource, where ξ is a scaling factor with $\xi \geq 1$.

It can be seen that a large value for scaling factor ξ means that more conservative the algorithm is, since more bandwidth resources are reserved for each request r_l.

The key to this algorithm is to find an appropriate value for ξ. A larger ξ means less resource violation, if multiple requests share the same bottleneck link. It however reserves more bandwidth resource demand than requested, and may lead to requests being rejected if ξ is too large. To find an appropriate value for ξ, we set a threshold Ξ for the maximum resource violation ratio of the edges in E. If the maximum resource violation ratio is larger than Ξ, we increase ξ by one, re-run the mentioned bounding step.

The detailed algorithm is presented in `Algorithm` 2.

4.4 Algorithm Analysis

We first show the feasibility of the solution delivered by algorithm `Appro_Consolidated` in Lemma 1.

Lemma 1. *Assuming that the IoT application and VNF of each NFV-enabled request are consolidated to a single location (either a cloudlet or a gateway node), Algorithm 1 delivers a feasible solution to the IoT application placement problem in an MEC with abundant bandwidth resource, while the computing capacity of each cloudlet or gateway node is violated by a maximum factor of $(1+\eta)$, where η is a parameter used in the filtering process of the algorithm.*

Algorithm 2. `Heu_Consolidated`

Input: $G = (V \cup \mathcal{GW}, E)$, a set of NFV-enabled requests $\mathcal{R}$, computing capacity $C(Loc_m)$ for each potential location (either a cloudlet $v_i \in V$ or gateway node $g_k \in \mathcal{GW}$).
Output: The assignment of the IoT application sv_l and VNF_l of each request r_l to a cloudlet or a gateway node.
1: Solve the LP;
2: $\xi \leftarrow 1$;
3: **while** $\xi \geq 1$ **do**
4: Solve the LP;
5: Let y^* be the optimal solution due to **LP**;
6: Filter the solution based on the optimal solution y^* to the **LP**, as shown in Eq. (19);
7: Construct a feasible solution y' to the **LP**, according to Eq. (22);
8: Pick a unassigned request r_l with the smallest implementation cost c_l^*. For this request r_l, we assign it to a location $Loc_{\theta(l)}$ in $\mathcal{L}_l$ with the smallest implementation cost if the location has enough computing resource demand, and there exist a path from n_l to Loc_m that has at least an amount of $\xi \cdot b_l \cdot B_{unit}$ of available bandwidth resource;
9: Repeat step 8 until no more requests can be admitted;
10: Calculate the maximum resource violation ratio RV of edges in E;
11: **if** $RV \leq \Xi$ **then**
12: **break**;

Proof. Showing the solution feasibility is to show that the IoT application sv_l and network function VNF_l are placed into a cloudlet or gateway node of the MEC G and the computing capacity constraints of each cloudlet and gateway nodes are met. Algorithm 1 relaxes the *ILP*. The solution obtained from the relaxed *LP* is a fractional solution, since the indicator variable $y_{l,m}$ is relaxed into a real value that is the range of $[0, 1]$. This means that the combination of IoT application sv_l and VNF_l of r_l can be split into multiple locations. This however is not a feasible solution because the problem assumes that the IoT application and VNF of each NFV-enabled request r_l is not splittable. It obviously is a lower bound for the problem, since allowing the IoT application sv_l and VNF_l being split into different locations creates more opportunities to find locations with lower implementation costs. To make this solution feasible, we adopt a filter-and-bound method. The solution obtained thus guarantees each request is assigned to a single location in $V \cup \mathcal{GW}$.

We then show the maximum resource violation ratios of the proposed algorithm. Recall that in the filter-and-bound process, we filter out some locations from the optimal solution y^* to the *LP*, by considering a location Loc_m as a candidate location for r_l only if $\frac{b_l \cdot C_{unit}}{C(Loc_m)} \leq (1+\eta) \cdot \omega_l^*$. This means that r_l's contribution to strengthening the computing capacity of Loc_m will not be greater than $(1+\eta) \cdot \omega_l^*$, i.e.,

$$\begin{aligned}\frac{b_l \cdot C_{unit}}{C(Loc_m)} &\leq (1+\eta) \cdot \arg\max_{Loc_m \in V \cup \mathcal{GW}} \frac{b_l \cdot C_{unit} \cdot y_{ml}^*}{C(Loc_m)} \\ &\leq (1+\eta) \cdot \arg\max_{Loc_m \in V \cup \mathcal{GW}} \frac{b_l \cdot C_{unit}}{C(Loc_m)}, \\ &\leq (1+\eta) \cdot \frac{b_{max} \cdot C_{unit}}{C_{min}}.\end{aligned} \tag{23}$$

In the worst case all the requests in R may be assigned to a single location. Without loss of generality, we assume that the computing capacity of each cloudlet is much higher than the demand of user requests, i.e., $b_{max} \cdot C_{unit} \ll C_{min}$. This can also be interpreted as $b_{max} \cdot C_{unit} \cdot |R| \leq C_{min}$. Then, the computing capacity of Loc_m may be violated by a ratio of

$$|R| \cdot \frac{b_l \cdot C_{unit}}{C(Loc_m)} \leq |R| \cdot (1+\eta) \cdot \frac{b_{max} \cdot C_{unit}}{C_{min}} \leq (1+\eta). \tag{24}$$

in the worst case. □

We finally show the approximation ratio and analyze the running time of algorithm `Appro_Consolidated` as stated by the following theorem.

Theorem 1. *Given an MEC $G = (V \cup \mathcal{GW}, E)$, a set of NFV-enabled requests $\mathcal{R}$ with each request $r_l = (n_l, sv_l; VNF_l, b_l)$ requesting transfer an amount b_l of data to its network function VNF_l for processing before being forwarded to its IoT application sv_l for further processing, assuming that the IoT application sv_l and network function VNF_l are consolidated into a single location (either a cloudlet or a gateway node), there is an approximation algorithm* `Appro_Consolidated` *that delivers a feasible solution with an approximation ratio of $3(1+\epsilon)$ to the IoT application placement problem.*

Proof. We showed that the obtained solution by algorithm `Appro_Consolidated` is feasible in Lemma 1. In this theorem, we analyze its approximation ratio. For clarity, we use OPT to denote the optimal solution to the IoT application placement problem in an MEC, which can be obtained from solving *ILP*. Let $\hat{y}$ be a solution obtained from algorithm `Appro_Consolidated`, and $c(\hat{y})$ be the cost due to solution $\hat{(y)}$. Likewise, we have $c(y^*)$ to denote the cost of the optimal solution to the *LP* and $c(y')$ to denote the cost of the constructed feasible solution to the *LP*.

We compare the costs of solutions y' and y^*. Specifically

$$\begin{aligned}c(y') &= \sum_{r_l \in R} c_{energy}(r_l) + \sum_{Loc_m \in V \cup \mathcal{GW}} y'_{l,m} \cdot b_l \Big(c_{l,m} \\ &\quad + c_{l,m}^{sv} + \sum_{e \in p_{n_l, Loc_m}} c_e \Big) \\ &\leq \sum_{r_l \in R} (1+\epsilon) \cdot c_l^* \sum_{Loc_m \in V \cup \mathcal{GW}} y'_{l,m} \\ &\leq (1+\epsilon) c(y^*).\end{aligned} \tag{25}$$

In the rounding step, we assign the request r_l with the least implementation cost in solution y^* to the location that incurs the least implementation cost in $\mathcal{L}_l$. For each of other requests, e.g., $r_{l'} \in \mathcal{L}_l \cap \mathcal{L}_{l'}$, we increased its assignment cost to that in location Loc_m for r_l. Let $c(r_l, Loc_m)$ be the implementation cost of r_l in location Loc_m. Assuming that the cost satisfies the triangle inequality, we have

$$\begin{aligned}c(r_{l'}) &= c(r_{l'}, Loc_{\theta(l)}) \\ &\leq c(r_{l'}, Loc_m) + c(r_l, Loc_m) + c(r_l, Loc_{\theta(l)}) \\ &\leq (1+\epsilon) c_{l'}^* + (1+\epsilon) c_l^* + (1+\epsilon) c_l^* \\ &\leq 3 \cdot (1+\epsilon) c_{l'}^*.\end{aligned} \tag{26}$$

In summary, we have

$$c((\hat{y})) \leq 3(1+\epsilon)c(y^*) \leq 3(1+\epsilon)OPT, \tag{27}$$

because the *LP* is a relaxed version of the *ILP2*. □

Theorem 2. *Given an MEC $G = (V \cup \mathcal{GW}, E)$, a set of NFV-enabled requests $\mathcal{R}$ with each request $r_l = (n_l, sv_l; VNF_l, b_l)$ requesting transfer an amount b_l of data to its network function VNF_l for processing before being forwarded to its IoT application sv_l for further processing, assuming that the IoT application sv_l and network function VNF_l are consolidated into a single location (either a cloudlet or a gateway node), there is a heuristic* `Heu_Consolidated` *that delivers a feasible solution to the IoT application placement problem while the bandwidth resource of an edge is violated by at most Ξ times.*

The proof of the solution feasibility of algorithm `Heu_Consolidated` is similar to the one in Lemma 1, and the violation ratio of resource is obvious, omitted.

5 Heuristic Algorithms for the IoT Application Placement Problem

So far, we have assumed that the IoT application and VNF instance of each request are consolidated into a single cloudlet or a gateway node. In this section we propose an efficient heuristic for the IoT application placement problem in an MEC without this assumption.

5.1 Basic Idea

The basic idea of the proposed heuristic is to first place the VNFs of all requests, by using a similar ILP in algorithm *Appro_Consolidated*. Specifically, another ILP is formulated to place the VNFs of requests without considering the IoT application placements, by adopting a similar program as *ILP2*. The formulated program is then relaxed into an LP that can be solved in polynomial time. However, the obtained solution may be infeasible, since each request may be 'partially' assigned to multiple locations. Consider such locations as the candidate locations for the VNFs of all requests. Based on the candidate locations of the VNFs, we place each IoT application sv_l of request r_l into a location that could achieve the minimum implementation cost of the request.

5.2 Algorithm

Recall that in *ILP2* $y_{l,m}$ is an indicator variable that shows whether VNF_l and sv_l of request r_l is consolidated into a potential location $Loc_m \in V \cup \mathcal{GW}$. In contrast, since VNF_l and sv_l are not necessarily consolidated into a single location, we assume that $y_{l,m}$ indicates whether VNF_l of request r_l is placed into a potential location $Loc_m \in V \cup \mathcal{GW}$. We refer to this modified version of *ILP2* as *ILP3*.

Similar to algorithm *Appro_Consolidated*, we obtain a fractional solution that corresponds the locations of VNFs of requests, by solving the relaxed version of *ILP3*. Specifically, we relax Constraint (16) of *ILP3* into a set of real values with each being in the range of $[0, 1]$. This relaxed program can be solved in polynomial time. The obtained solution may split each VNF_l into a number of locations with positive values for $y_{l,m}$, i.e., locations in $\{Loc_m \mid y_{l,m} > 0\}$. Since VNF_l cannot be split into multiple locations, we use locations in $\{Loc_m \mid y_{l,m} > 0\}$ as the candidate locations for VNFs of all requests in $\mathcal{R}$. Notice that a candidate location may not be able to host VNF_l as *ILP2* does not consider bandwidth capacity constraints on links and the path from the IoT node n_l to the location may not have enough bandwidth resource available. We thus prune set $\{Loc_m \mid y_{l,m} > 0\}$ by removing the locations that could not met the computing resource demand of r_l and its bandwidth resource demand in the paths from IoT nodes to the locations. Let $Can_{vnf,l}$ be the set of the pruned candidate set for VNF_l of request r_l.

We proceed by finding locations for the IoT application sv_l of each request r_l, based on the candidate set $Can_{vnf,l}$ of each request r_l. Specifically, we first rank the requests in $\mathcal{R}$ into an increasing order of their traffic. For the ranked list $\mathcal{R}$, we then admit the requests one by one. For each request r_l, we find a pair of locations with one location Loc_a in $Can_{vnf,l}$ for VNF_l and the other location Loc_b in $V \cup \mathcal{GW}$ for IoT application sv_l, such that (1) Loc_b have enough computing resource for VNF_l and sv_l; (2) The path from Loc_a to Loc_b has enough bandwidth resource to transfer an amount b_l of data traffic of r_l. The proposed heuristic is given in Algorithm 3.

Algorithm 3. `Heuristic`

Input: $G = (V \cup \mathcal{GW}, E)$, a set of NFV-enabled requests $\mathcal{R}$, computing capacity $C(Loc_m)$ for each potential location (either a cloudlet $v_i \in V$ or gateway node $g_k \in \mathcal{GW}$).

Output: The assignment of the IoT application sv_l and VNF_l of each request r_l to a cloudlet or a gateway node.

/*Stage 1: Placement of VNFs of the requests in $\mathcal{R}$*/
Solve the relaxed program of **ILP3**);
For each request r_l, consider locations in $\{Loc_m \mid y_{l,m} > 0\}$ as the candidate locations for VNF_l;
Let $Can_{vnf,l}$ be the set of the pruned candidate set for VNF_l of request r_l;
Removing the locations that could not met the computing resource demand of r_l and its bandwidth resource demand in the paths from IoT nodes to the locations;
/*Stage 2: Placement of VNFs of the requests in $\mathcal{R}$*/
Rank the requests in $\mathcal{R}$ into an increasing order of their traffic.
for each request $r_l \in \mathcal{R}$ **do**
    Find a pair of locations with one location Loc_a in $Can_{vnf,l}$ for VNF_l and the other location Loc_b in $V \cup \mathcal{GW}$ for IoT application sv_l, such that the computing resource demand of sv_l and its bandwidth resource from Loc_a to Loc_b is met;

We now analyze the performance of the proposed heuristic in the following theorem.

Theorem 3. *Given an MEC $G = (V \cup \mathcal{GW}, E)$, a set of NFV-enabled requests $\mathcal{R}$ with each request $r_l = (n_l, sv_l; VNF_l, b_l)$ requesting to transfer an amount b_l of data to its network function VNF_l for processing before reaching its IoT application sv_l for further processing, algorithm* `Heuristic` *delivers a feasible solution to the IoT application placement problem in MEC G.*

Proof. We show the solution feasibility of the proposed heuristic by showing that the computing capacities of each cloudlet v_i and gateway node g_k are met. We also show that the bandwidth capacity on each link $e \in E$ is met. Although we consider the computing capacity constraint on each potential location (either a cloudlet or gateway node) in *ILP3*, the computing capacity may be violated since we relaxed the *ILP3* and the obtained solution may place a single VNF_l into multiple locations. If one of the such locations is selected to implement VNF_l, it may not have sufficient computing resource for it. We thus remove the candidate locations that do not meet the computing resource demand of r_l. Also, in stage 2 of algorithm `Heuristic`, we only consider the locations that can meet the computing resource demand of sv_l. The computing resource thus can be met. Similarly, we can see from algorithm `Heuristic` that the bandwidth capacity of each link is met as well. □

6 Simulation

In this section we evaluate the performance of the proposed algorithms by simulations.

6.1 Experiment Settings

We consider mobile edge cloud networks with the network size being varied from 10 to 100 cloudlets, where each network topology is generated using GT-ITM [24]. The number of IoT gateways in the network is set to 10 percent of the network size, and they are randomly co-located with cloudlets in the network edge. There is an AP node in each of the IoT gateway. We also use real network topologies, i.e., GÉANT [23] and an ISP network from [35]. The computing capacity of cloudlet varies from 40,000 to 120,000 MHz [31] with around tens of servers. The bandwidth capacity of each link varied randomly in the range of $[20, 100]$ Mbps. Five types of network functions, i.e., Firewall, Proxy, NAT, IDS, and Load Balancing, are considered, and their computing demands are adopted from [28], [40]. The data of each NFV-enabled request is randomly drawn from $[20, 200]$ Megabytes. The number of NFV-enabled requests for each network is the double of its size. For example, there are 400 NFV-enabled requests for each network with size 200. The running time of each algorithm is obtained based on a machine with a 3.70 GHz Intel i7 Hexa-core CPU and 16 GiB RAM. Unless otherwise specified, these parameters will be adopted in the default setting. The result of each figure is the average values of 15 different runs of the proposed algorithms.

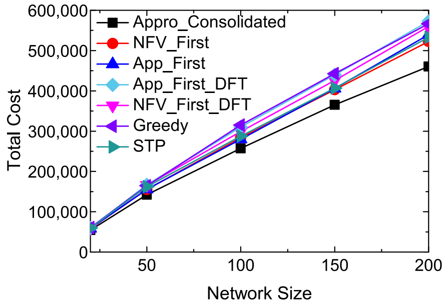
We compare the performance of the proposed algorithms against the following state-of-the-art algorithms:

- *NFV first heuristic:* We first place the VNFs of requests greedily, and given the locations of placed VNFs we then place IoT applications nearby. For simplicity, we refer to this algorithm as `NFV_First`.
- *Application first heuristic:* We first select locations for the IoT applications of requests, and then choose locations for its VNFs in-between IoT applications and IoT nodes, which is referred to as algorithm `App_First`.
- *NFV and decreasing first fit:* This heuristic is motivated by traditional decreasing first fit for bin packing. That is, the heuristic the requests into a decreasing order of their amounts of data. Similar to algorithm `NFV_First`, we first place VNFs and then IoT applications of the requests in the ranked order. We refer this algorithm as `NFV_First_DFT`.
- *Application and decreasing first fit:* Similar to algorithm `NFV_First_DFT`, the requests are ranked into a decreasing order of their amounts of data. We then place IoT applications before placing VNFs of the requests in the ranked order. This algorithm is referred to as `App_First_DFT`.
- *Greedy:* The next benchmark we use is the greedy algorithm in [8], which first places VNFs and IoT applications and then assign the requests to the placed VNFs and IoT applications.
- *Shortest-path-based algorithms:* We also compare the performance of our algorithms with those based on constructing auxiliary graphs and shortest paths, similar to the studies in [30], [57]. However, the problems studies in the paper is totally different with those in [30]. For example, they either do not consider bandwidth capacities or ignore We thus only adopt the design rationale behind those studies. Specifically, we add a virtual sink node to the network G, and connect each cloudlet/gateway to the virtual sink node. We then find a shortest path from the IoT node of each node to the virtual sink node. We refer this benchmark as `STP`.

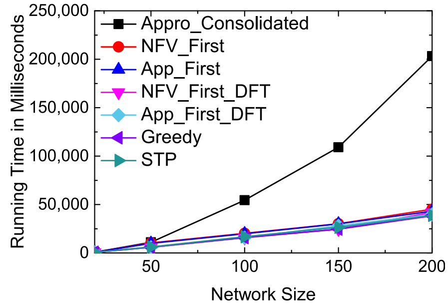
6.2 Performance of Algorithms `Appro_Consolidated` and `Heu_Consolidated`

We first evaluate the performance of algorithm `Appro_Consolidated` against that of algorithms `NFV_First`, `App_First`, `NFV_First_DFT` and `App_First_DFT` in terms of the total cost of implementing all requests and running time, in different networks with their sizes varying from 20 to 200. The results are shown in Fig. 3. From Fig. 3a, we can see that algorithm `Appro_Consolidated` has a much lower total cost than algorithms `NFV_First`, `App_First`, `NFV_First_DFT` and `App_First_DFT`. For example, when the network size is 200, algorithm `Appro_Consolidated` has around 15 percent less total cost than algorithms `NFV_First` and `App_First`. The reason is that algorithm `Appro_Consolidated` jointly finds locations for IoT applications and their VNFs. This significantly increases the probability of placing IoT applications and VNFs to locations that are close to their users, thereby reducing the transmission costs of implementing IoT applications and VNF instances. In addition, we can see that the total costs by the three algorithms are increasing with the growth of the network size. The rationale behind is that networks with larger sizes have higher probability of placing IoT applications and VNFs in locations that are far from the sources of requests. From Fig. 3b, we can see that the running time of algorithm `Appro_Consolidated` is slightly higher than algorithms `NFV_First` and `App_First`.

We then investigate the performance of algorithms `Appro_Consolidated`, `NFV_First`, `App_First`, `NFV_First_DFT` and `App_First_DFT` in terms of total cost in

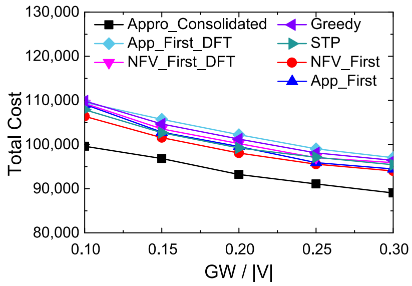


(a) Total cost ($).

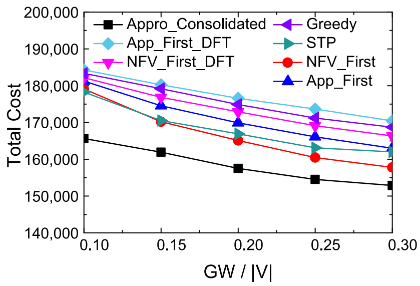


(b) Running time (milliseconds).

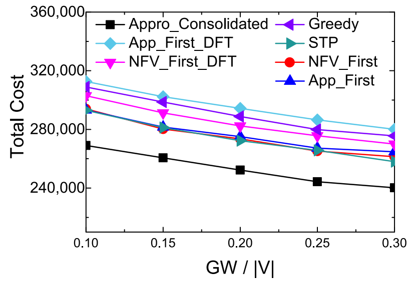
Fig. 3. The performance of algorithms `Appro_Consolidated`, `NFV_First`, and `App_First` in different networks.



(a) Total cost ($) in network GÉANT.

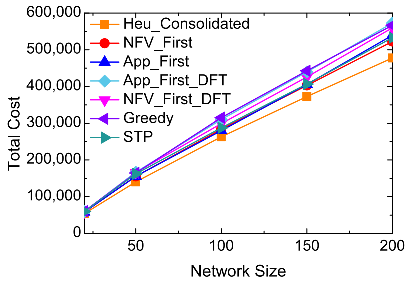


(b) Total cost ($) in network AS4755.

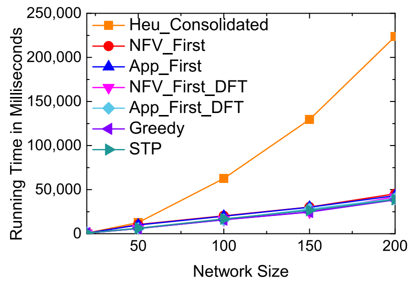


(c) Total cost ($) in network AS1755.

Fig. 4. The performance of algorithms `Appro_Consolidated`, `NFV_First`, and `App_First` in real networks GÉANT, AS4755, and AS1755.



(a) Total cost ($).

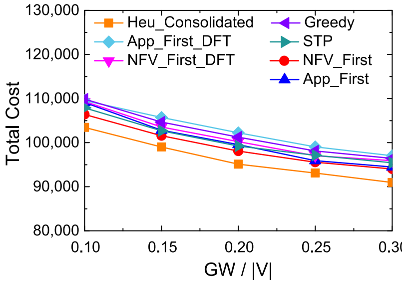


(b) Running time (milliseconds).

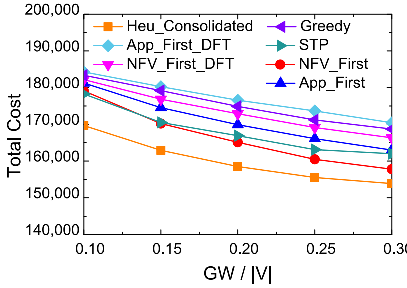
Fig. 5. The performance of algorithms `Heu_Consolidated`, `NFV_First`, and `App_First` in different networks.

real networks GÉANT, AS4755 and AS1755, by varying the ratio of $|\mathcal{GW}|/|V|$ from 0.1 to 0.3 with an increase of 0.05. Fig. 4 shows the results, from which we can see that algorithm `Appro_Consolidated` consistently delivers a lower cost in different networks of different ratios of $|\mathcal{GW}|/|V|$. For example, in Fig. 4a, algorithm `Appro_Consolidated` has approximately 10 percent less total cost than algorithms `NFV_First` and `App_First`. The reason is that algorithm `Appro_Consolidated` jointly places IoT applications and VNF instances of each request. However, other benchmarks either place IoT applications and VNFs separately, which could lead to suboptimal solutions. We can also see from Fig. 4a that the total cost is decreasing with the growth of ratio $|\mathcal{GW}|/|V|$. This is because a higher value for $|\mathcal{GW}|/|V|$ means more gateways for the network, which allows more user requests being implemented in the gateways that are closer to users than cloudlets. The transmission costs thus can be reduced significantly. Similar performance of algorithms `Appro_Consolidated` and `NFV_First` and `App_First` in networks AS4755 and AS1755 can be found in Figs. 4b and 4c. Furthermore, we can see that the performance gap between the algorithms is smaller than networks AS4755 and AS1755, the reason is that networks AS4755 and AS1755 have larger sizes and the impact of joint application and VNF placement is enlarged.

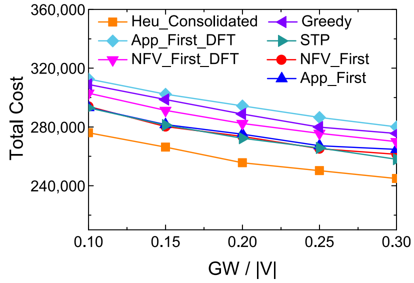
We third investigate the performance of algorithms `Heu_Consolidated`, `NFV_First`, `App_First`, `NFV_First_DFT` and `App_First_DFT` in terms of total cost and running time in different networks with their sizes being varied from 20 to 200. Fig. 5a shows the total costs achieved by the three algorithms `Heu_Consolidated`, `NFV_First`, and `App_First`, from which it can be seen that the total cost by algorithm `Heu_Consolidated` is much lower than those of algorithms `NFV_First` and `App_First`. For instance, when the network size is 200, algorithm `Heu_Consolidated` has around 10 percent lower cost than algorithm `App_First`. The reason is that algorithms `NFV_First` and `App_First` do not jointly place IoT applications and VNFs. It can also be seen from Fig. 5a that the performance gap between algorithm `Heu_Consolidated` and the other two is increasing



(a) Total cost ($) in network GÉANT.

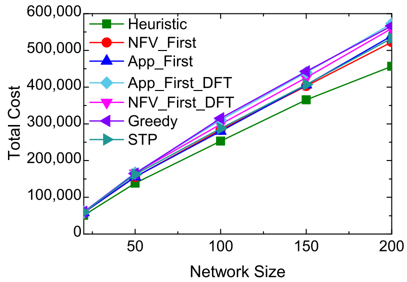


(b) Total cost ($) in network AS4755.

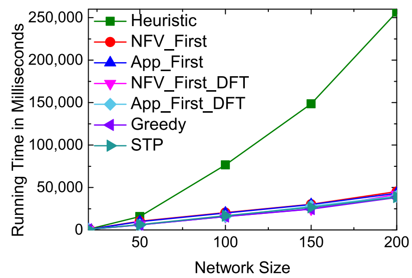


(c) Total cost ($) in network AS1755.

Fig. 6. The performance of algorithms `Heu_Consolidated`, `NFV_First`, and `App_First` in real networks GÉANT, AS4755, and AS1755.

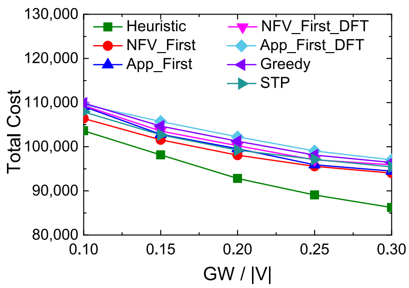


(a) Total cost ($).

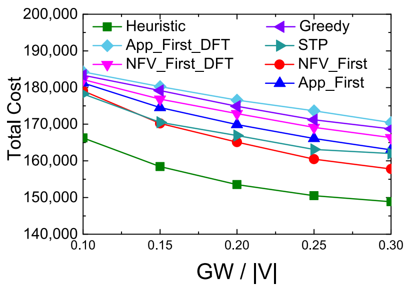


(b) Running time (milliseconds).

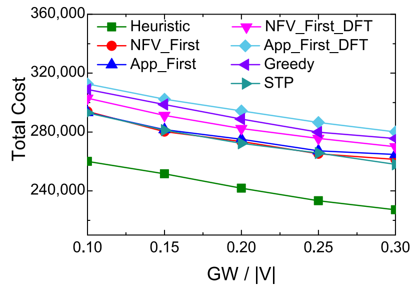
Fig. 7. The performance of algorithms `Heuristic`, `NFV_First`, and `App_First` in different networks.



(a) Total cost ($) in network GÉANT.



(b) Total cost ($) in network AS4755.



(c) Total cost ($) in network AS1755.

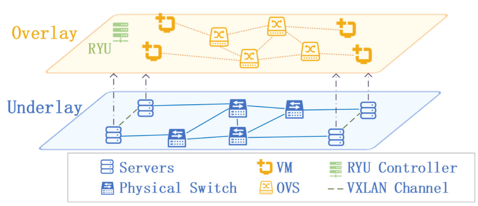
Fig. 8. The performance of algorithms `Heuristic`, `NFV_First`, and `App_First` in real networks GÉANT, AS4755, and AS1755.

with the growth of network sizes. This is because larger networks can increase the probability of placing IoT applications and VNFs in further distances, thereby enlarging the impact of joint placement of IoT applications and VNFs on the system performance. The running times of algorithms `Heu_Consolidated`, `NFV_First` and `App_First` are shown in Fig. 5b.

We finally study the performance of algorithms `Heu_Consolidated`, `NFV_First`, `App_First`, `NFV_First_DFT` and `App_First_DFT` in real networks GÉANT, AS4755, and AS1755 in terms of the total cost of implementing all requests, by varying the ratio $|\mathcal{GW}|/|V|$ from 0.1 to 0.3 with an increasing step of 0.05. From Fig. 6a, we can see that the total cost by algorithm `Heu_Consolidated` is lower than algorithms `NFV_First` and `App_First`. This performance gap in increasing with the growth of ratio $|\mathcal{GW}|/|V|$. The reason behind is that a higher value for $|\mathcal{GW}|/|V|$ means more gateways are deployed in the network. This makes algorithms `NFV_First` and `App_First` have a higher probability of separating IoT applications and VNFs via longer transmission paths, because gateways usually do not have enough capacity to serve IoT applications. Similar performance of the three algorithms in networks AS4755, and AS1755 can be seen in Figs. 6b and 6c.

6.3 Performance of Algorithm `Heuristic`

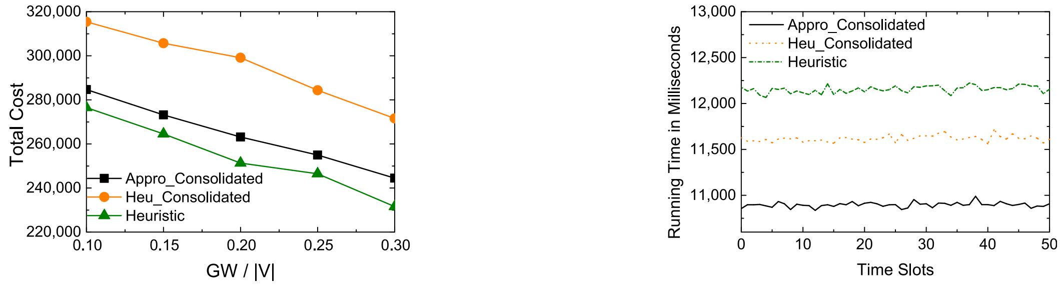
We continue evaluating the performance of algorithms `Heuristic` against that of algorithms `NFV_First`, `App_First`, `NFV_First_DFT` and `App_First_DFT` in networks with sizes being varied from 20 to 200. From Fig. 7a, we can see that the total cost by algorithm `Heuristic` is much lower than algorithms `NFV_First` and `App_First`.



(a) Overlay and underlay of the test-bed.

(b) Physical switches.

Fig. 9. The test-bed settings.



(a) Total cost ($) in the test-bed. (b) Running times (milliseconds).

Fig. 10. The performance of algorithms `Appro_Consolidated`, `Heu_Consolidated`, and `Heuristic` in a real test-bed.

The performance gap is also increasing with the growth of the network size. The running times of the three algorithms are shown in Fig. 7b, from which we can see algorithms `NFV_First` and `App_First` have similar running times that are lower than algorithm `Heuristic`. Similar performance on the total cost for the three algorithms in real networks GÉANT, AS4755, and AS1755 can be seen in Fig. 8.

7 Experiments in a Real Test-Bed

We now implement the proposed algorithms in a real test-bed and investigate the performance of the proposed algorithm in real environments.

7.1 Test-Bed Settings

We evaluate the performance of the proposed algorithms in a real test-bed with five hardware switches, as shown in Fig. 9b. To testify the scalability of the algorithms, we adopt a two-layered network architecture: an underlay and an overlay, as illustrated in Fig. 9a. The underlay is a network that interconnects five H3C S5560-30S-EI switches, and five servers with each having an i7-8700 CPU and 16G RAM. Based on this underlay, an overlay with an AS1755 topology is built based on VXLAN and Open vSwitch (OVS) [47]. Its OVS nodes and VMs are controlled by a Ryu [50] controller. The proposed algorithms are implemented as Ryu applications. All the other settings are the same as the simulations of the previous section.

7.2 Performance Results

We study the performance of the proposed algorithms `Appro_Consolidated`, `Heu_Consolidated`, and `Heuristic`. Specifically, we consider a dynamic scenario with the time being equally divided into time slots. A set of requests are ready for assignment in the beginning of each time slot. We then invoke the proposed algorithms in the beginning of each time slot. Fig. 10 illustrates the results of total costs and running times within a time horizon of 50 time slots. From the results, we can see that algorithms `Appro_Consolidated`, `Heu_Consolidated` have higher costs than that of algorithm `Heuristic`. The rationale behind is that algorithms `Appro_Consolidated`, `Heu_Consolidated` consolidate IoT application and its VNF together, and this may exclude some cloudlets with less computing resource and low costs. Furthermore, we can see that algorithm `Appro_Consolidated` has a higher total cost than algorithm `Heu_Consolidated`. This is because algorithm `Appro_Consolidated` does not consider the bandwidth capacity constraint of links. It thus admits more requests with higher costs. For the running times, algorithm `Heuristic` is the highest while algorithm `Appro_Consolidated` is the lowest.

8 Conclusion

In this paper, we studied the IoT application placement problem in a NFV-enabled MEC. We first considered a special case of the problem where both the IoT application and its VNF of each request are placed into a single cloudlet, for which we proposed an exact solution and an approximate solution with a provable approximation ratio without the bandwidth resource constraint. We also developed an efficient heuristic for the special case of the problem with the bandwidth resource constraint. Furthermore, we proposed an efficient heuristic for the IoT application placement problem that jointly places IoT applications and VNFs. We finally investigated the performance of the proposed algorithms by simulations and experiments in a real test-bed. Experimental results show that the performance of the proposed algorithms outperform their counterparts.

Acknowledgments

The authors would like to thank the three anonymous referees and the associate editor for their expertise comments and constructive suggestions, which have helped us improve the quality and presentation of the paper greatly. The work of Zichuan Xu, Qiufen Xia, and Guowei Wu was supported in part by the National Natural Science Foundation of China (Grant No. 61802048, 61802047, 61772113, 61872053), the fundamental research funds for the central universities in China (Grant No. DUT17RC(3)061, DUT17RC(3)070, DUT19RC(4) 035, DUT19GJ204), and the Xinghai Scholar Program in the Dalian University of Technology, China.

References

[1] T. Ahmed, A. Alleg, and N. Marie-Magdelaine, "An architecture framework for virtualization of IoT Network," in *Proc. IEEE Conf. Netw. Softwarization*, 2019, pp. 183–187.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.

[3] H. Alameddine, C. Assi, M. Tushar, and J. Yu, "Low-latency service schedule orchestration in NFV-based networks," in *Proc. IEEE Conf. Netw. Softwarization*, 2019, pp. 378–386.

[4] B. R. Al-Kaseen and H. S. Al-Raweshidy, "SD-NFV as an energy efficient approach for M2M networks using cloud-based 6LoWPAN testbed," *IEEE Internet Things*, vol. 4, no. 5, pp. 1787–1797, Oct. 2017.

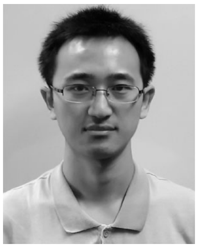
[5] Amazon Web Services, "Inc. Amazon EC2 instance configuration," 2020. [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-ec2-config.html
[6] J. W. Anderson, R. Braud, R. Kapoor, G. Porter, and A. Vahdat, "xOMB: Extensible open middleboxes with commodity servers," in *Proc. ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, 2012, pp. 49–60.
[7] A. Anadiotis, L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SD-WISE: A software-defined wireless sensor network," *Comput. Netw.*, vol. 159, pp. 84–95, 2019.
[8] D. Bhamare, M. Samaka, A. Erbad, R. Jain, L. Gupta, and H. A. Chan, "Optimal virtual network function placement in multi-cloud service function chaining architecture," *Comput. Commun.*, no. 102, pp. 1–16, 2017.
[9] N. Bizanis and F. A. Kuipers, "SDN and virtualization solutions for the Internet of Things: A survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.
[10] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.
[11] G. Cheng, H. Chen, H. Hu, Z. Wang, and J. Lan, "Enabling network function combination via service chain instantiation," *Comput. Netw.*, vol. 92, pp. 396–407, 2015.
[12] J. Chuzhoy and J. Naor, "Covering problems with hard capacities," *SIAM J. Comput.*, vol. 36, no. 2, pp. 498–515, 2006.
[13] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The dynamic placement of virtual network functions," in *Proc. Netw. Operations Manage. Symp.*, 2014, pp. 1–9.
[14] R. Cziva and D. P. Pezaros, "Container network functions: Bringing NFV to the network edge," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 24–31, Jun. 2017.
[15] ETSI Network Function Virtualization, 2018. Accessed: Sep. 2019. [Online]. Available: https://www.etsi.org/images/files/ETSITechnologyLeaflets/NetworkFunctionsVirtualization.pdf.
[16] White Paper on NFV priorities for 5G, 2017. Accessed: Sep. 2019. [Online]. Available: https://portal.etsi.org/NFV/NFV_White_Paper_5G.pdf
[17] Multi-access Edge Computing (ETSI), 2015. [Online]. Available: https://www.etsi.org/technologies/multi-access-edge-computing
[18] S. K. Fayazbakhsh, L. Chiang, V. Sekar, M. Yu, and J. C. Mogul, "Enforcing network-wide policies in the presence of dynamic middlebox actions using FlowTags," in *Proc. 11th USENIX Symp. Netw. Syst. Des. Implementation*, 2014, pp. 543–546.
[19] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1997.
[20] A. Gante, M. Aslan, and A. Matrawy, "Smart wireless sensor network management based on software-defined networking," in *Proc. 27th Biennial Symp. Commun.*, 2014, pp. 71–75.
[21] N. Garg and J. Könemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," in *Proc. 39th Annu. Symp. Found. Comput. Sci.*, 1998, pp. 300–309.
[22] A. Gember-Jacobson *et al.*, "Stratos: A Network-aware orchestration layer for middleboxes in the cloud," *CoRR*, vol. abs/1305.0209, 2013.
[23] GÉANT, 2000. [Online]. Available: http://www.geant.net
[24] 1996. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/
[25] S. Gu, Z. Li, C. Wu, and C. Huang, "An efficient auction mechanism for service chains in the NFV market," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.
[26] Y. Guo, H. Zhu, and L. Yang, "Service-oriented network virtualization architecture for Internet of Things," *China Commun.*, vol. 13, no. 9, pp. 163–172, 2016.
[27] A. Gupta *et al.*, "SDX: A software defined internet exchange," in *Proc. ACM Conf. SIGCOMM*, 2014, pp. 551–562.
[28] A. Gushchin, A. Walid, and A. Tang, "Scalable routing in SDN-enabled networks with consolidated middleboxes," in *Proc. ACM SIGCOMM Workshop Hot Topics Middleboxes Netw. Function Virtualization*, 2015, pp. 55–60.
[29] N. Herbaut, D. Negru, G. Xilouris, and Y. Chen, "Migrating to a NFV-based home gateway: Introducing a surrogate vNF approach," in *Proc. 6th Int. Conf. Netw. Future*, 2015, pp. 1–7.
[30] M. Huang, W. Liang, Z. Xu, M. Jia, and S. Guo, "Throughput maximization in software-defined networks with consolidated middleboxes," in *Proc. IEEE 41st Conf. Local Comput. Netw.*, 2016, pp. 298–306.
[31] Hewlett-Packard Development Company, "L.P. Servers for enterprise bladeSystem, rack & tower and hyperscale." 2019. Accessed: Mar. 2019. [Online]. Available: http://www8.hp.com/us/en/products/servers/
[32] Hikvision gateways powered by Intel, 2019. [Online]. Available: https://www.intel.cn/content/www/cn/zh/analytics/artificial-intelligence/intel-power-hikvision-create-deep-eyes-global-video-camera.html
[33] IHS, 2015. [Online]. Available: http://www.infonetics.com/pr/2015/1H15-Service-Provider-Capex.asp
[34] Intel Movidius Vision Accelerator, 2019. [Online]. Available: https://software.intel.com/en-us/iot/hardware/vision-accelerator-movidius-vpu
[35] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.
[36] Y. Li, L. T. X. Phan, and B. T. Loo, "Network functions virtualization with soft real-time guarantees," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.
[37] J. Lin and J. Vitter, "Approximation algorithms for geo- metric median problems," *Inf. Process. Lett.*, vol. 44, pp. 245–249, 1992.
[38] T. Luo, H. Tan, and T. Q. S. Quek, "Sensor OpenFlow: Enabling software-defined wireless sensor networks," *IEEE Commun. Lett.*, vol. 16, no. 11, pp. 1896–1899, Nov. 2012.
[39] L. Mamatas, S. Clayman, and A. Galis, "Information exchange management as a service for network function virtualization environments," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 564–577, Sep. 2016.
[40] J. Martins *et al.*, "ClickOS and the art of network function virtualization," in *Proc. 11th USENIX Conf. Netw. Syst. Des. Implementation*, 2014, pp. 459–473.
[41] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[42] Microsoft, "Plan network requirements for Skype for business," 2015. [Online]. Available: https://technet.microsoft.com/en-us/library/gg425841.aspx
[43] C. Mouradian, N. T. Jahromi, and R. H. Glitho, "NFV and SDN-based distributed IoT gateway for large-scale disaster management," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 4119–4131, Oct. 2018.
[44] R. Morabitoa, R. Petrolob, V. Loscríc, and N. Mitton, "LEGIoT: A lightweight edge gateway for the Internet of Things," *Future Gener. Comput. Syst.*, vol. 81, pp. 1–15, 2018.
[45] M. Ojo, D. Adami, and S. Giordano, "A SDN-IoT architecture with NFV implementation," in *Proc. IEEE Globecom Workshops*, 2016, pp. 1–6.
[46] Z. A. Qazi, C. C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, "SIMPLE-fying middlebox policy enforcement using SDN," in *Proc. ACM SIGCOMM Conf.*, 2013, pp. 27–38.
[47] Open vSwtich, 2016. [Online]. Available: https://www.openvswitch.org
[48] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Trans. Commun.*, vol. 64, no. 9, pp. 3746–3758, Sep. 2016.
[49] J. Ren, H. Guo, C. Xu, and Y. Zhang, "Serving at the edge: A scalable IoT architecture based on transparent computing," *IEEE Netw.*, vol. 31, no. 5, pp. 96–105, Aug. 3, 2017.
[50] Ryu SDN Controller, 2017. [Online]. Available: https://osrg.github.io/ryu/
[51] V. Sekar *et al.*, "Design and implementation of a consolidated middlebox architecture," in *Proc. 9th USENIX Conf. Netw. Syst. Des. Implementation*, 2012, Art. no. 24.
[52] V. Shrivastava, P. Zerfos, K. Lee, H. Jamjoom, Y. Liu, and S. Banerjee, "Application-aware virtual machine migration in data centers," in *Proc. IEEE INFOCOM*, 2011, pp. 66–70.
[53] P. Wang, J. Lan, X. Zhang, Y. Hu, and S. Chen, "Dynamic function composition for network service chain: Model and optimization," *Comput. Netw.*, vol. 92, pp. 408–418, 2015.
[54] Z. Xu and W. Liang, "Minimizing the operational cost of data centers via geographical electricity price diversity," in *Proc. 6th IEEE Int. Conf. Cloud Comput.*, 2013, pp. 99–106.
[55] Z. Xu and W. Liang, "Operational cost minimization for distributed data centers through exploring electricity price diversity," *Comput. Netw.*, vol. 83, pp. 59–75, 2015.
[56] Z. Xu, W. Liang, M. Jia, M. Huang, and G. Mao, "Task offloading with network function services in a mobile edge-cloud network," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2672–2685, Nov. 2019.
[57] Z. Xu, W. Liang, A. Galis, Y. Ma, Q. Xia, and W. Xu, "Throughput optimization for admitting NFV-enabled requests in cloud networks," *Comput. Netw.*, vol. 143, pp. 15–29, 2018.

[58] Z. Xu *et al*., "NFV-enabled multicasting in mobile edge clouds with resource sharing," in *Proc. 48th Int. Conf. Parallel Process.*, 2019, pp. 1–10.

[59] Z. Xu, L. Zhou, S. Chau, W. Liang, Q. Xia, and P. Zhou, "Collaborate or separate? Distributed service caching in mobile edge clouds," in *Proc IEEE INFOCOM*, 2020.

[60] B. Yang, W. Chai, Z. Xu, K. Katsaros, and G. Pavlou, "Cost-efficient NFV-Enabled mobile edge-cloud for low latency mobile applications," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 475–488, Mar. 2018.

[61] R. Yu, G. Xue, and X. Zhang, "Application provisioning in FOG computing-enabled Internet-of-Things: A network perspective," in *Proc. IEEE INFOCOM*, 2018, pp. 783–791.



Zichuan Xu (Member, IEEE) received the BSc and ME degrees in computer science from the Dalian University of Technology, Dalian, China, in 2008 and 2011, respectively, and the PhD degree in computer science from the Australian National University, Canberra, Australia, 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 an associate professor with the School of Software, 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.

Wanli Gong received the BSc degree in software engineering from the Dalian University of Technology, Dalian, China, in 2018. He is currently working toward the master's degree at the Dalian University of Technology, Dalian, China. His research interests include mobile cloud computing, Internet of Things, and network function virtualization.

Qiufen Xia (Member, IEEE) received the BSc and ME degrees in computer science from the Dalian University of Technology, Dalian, China, in 2009 and 2012, respectively, and the PhD degree in computer science from the Australian National University, Canberra, Australia, in 2017. She is currently a lecturer with the Dalian University of Technology. Her research interests include mobile cloud computing, query evaluation, big data analytics, big data management in distributed clouds, and cloud computing.

Weifa Liang (Senior Member, IEEE) received the BSc degree in computer science from Wuhan University, Wuhan, China, in 1984, the ME degree in computer science from the University of Science and Technology of China, Hefei, China, in 1989, and the PhD degree in computer science from the Australian National University, Canberra, Australia, in 1998. He is currently a full orofessor with the Research School of Computer Science, Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, cloud computing, software-defined networking, design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory.

Omer F. Rana (Senior Member, IEEE) received the BS degree in information systems engineering from the Imperial College of Science, Technology and Medicine, London, U.K. the MS degree in microelectronics systems design from the University of Southampton, Southampton, United Kingdom, and the PhD degree in neural computing and parallel architectures from the Imperial College of Science, Technology and Medicine, London. He is currently a professor of performance engineering with Cardiff University, Cardiff, U.K. His current research interests include problem solving environments for computational science and commercial computing, data analysis and management for large-scale computing, and scalability in high performance agent systems.

Guowei Wu received the PhD degree from Harbin Engineering University, Harbin, China, in 2003. He is currently a professor with the School of Software, Dalian University of Technology(DUT) in China. His research interests include embedded real-time system, cyber-physical systems, and smart edge computing. He has published more than 100 journal and conference papers.