Throughput Maximization of NFV-Enabled Multicasting in Mobile Edge Cloud Networks

Yu Ma, Weifa Liang, *Senior Member, IEEE,* Jie Wu, *Fellow, IEEE,* and Zichuan Xu, *Member, IEEE*

Abstract—Mobile Edge Computing (MEC) reforms the cloud paradigm by bringing unprecedented computing capacity to the vicinity of end users at the mobile network edge. This provides end users with swift and powerful computing and storage capacities, energy efficiency, and mobility- and context-awareness support. Furthermore, Network Function Virtualization (NFV) is another promising technique that implements various network functions for many applications as pieces of software in servers or cloudlets in MEC networks. The provisioning of virtualized network services in MEC can improve user service experiences, simplify network service deployment, and ease network resource management. However, user requests arrive dynamically and different users demand different amounts of resources, while the resources in MEC are dynamically occupied or released by different services. It thus poses a significant challenge to optimize the performance of MEC through efficient computing and communication resource allocations to meet ever-growing resource demands of users. In this paper, we study NFV-enabled multicasting that is a fundamental routing problem in an MEC network, subject to resource capacities on both its cloudlets and links. Specifically, we first devise an approximation algorithm for the cost minimization problem of admitting a single NFV-enabled multicast request. We then develop an efficient algorithm for the throughput maximization problem for the admissions of a given set of NFV-enabled multicast requests. We thirdly devise an online algorithm with a provable competitive ratio for the online throughput maximization problem when NFV-enabled multicast requests arrive one by one without the knowledge of future request arrivals. We finally evaluate the performance of the proposed algorithms through experimental simulations. Simulation results demonstrate that the proposed algorithms are promising.

Index Terms—Mobile edge-cloud networks (MEC); distributed resource allocation and provisioning; NFV-enabled multicast requests; virtualized network function (VNF); VNF instance placement and sharing; service function chains (SFCs); throughput maximization; Steiner tree problems; online algorithms.

1 Introduction

Mobile devices, including smart phones and tablets, gain increasing popularity as communication tools of users for their business, social networking, and personal entertainment. However, the computing, storage and battery capacity of each of them is very limited, due to their portal size. Leveraging by rich computing and storage resources in clouds, mobile devices can offload some of their tasks to clouds for processing and storage, while the clouds usually are remote located from their end users. Thus, the response delay to user requests may not be tolerable for some real-time applications. Instead, a new network paradigm, Mobile Edge Computing (MEC) is emerged, which can provide cloud-computing capability at the edge of core network in the proximity of mobile users [1]. MEC can significantly shorten the response delay to user applications, ensure highly efficient network operation and service delivery, and improve user experience of using the services, which is an ideal platform to meet ever-growing resource demands of mobile users for their applications, by enhancing mobile device capabilities with a real-time manner [22]. On the other hand, Network Function Virtualization (NFV) [21], has been envisaged as another promising technique to the next-generation networking that enables fast service deployment and cost-effective yet error-free service provisioning in future communication networks. It replaces resource demanding service applications from expensive, dedicated hardware-middleboxes, by software implementation in generic servers or cloudlets, where each network function is virtualized as a virtualized network function instance that runs in a virtual machine in a cloudlet.

Although implementing network functions as VNF instances in MEC is a promising technology, admitting NFV-enabled multicast requests in MEC poses several challenges. Firstly, both computing and storage resources at cloudlets and communication resources at links in an MEC are very limited in comparison with its counterpart - the powerful centralized data center networks (clouds). It is of paramount importance to optimize the performance of an MEC network through judicious allocating its limited resources to meet user resource demands. Secondly, each NFV-enabled multicast request has a service function chain requirement, how to steer the data traffic of the request to go through each network function in its service function chain correctly? Thirdly, the service chain implementation may either share some existing network function instances with the other requests or instantiate new VNF instances. How to make such a decision to minimize the admission cost of the request? Finally, how to maximize the network throughput by admitting or rejecting each arrived request immediately if requests arrive one by one without the knowledge of future request arrivals? In this paper, we will address the

- *Y. Ma and W. Liang are with the Research School of Computer Science, The Australian National University, Canberra, ACT 2601, Australia E-mails: yu.ma@anu.edu.au, and wliang@cs.anu.edu.au*
- *J. Wu is with the Department of Computer and Information Sciences, Temple University, 1925N 12th Street, Philadelphia, PA 19122, USA E-mail: jiewu@temple.edu*
- *Z. Xu is with the School of Software, Dalian University of Technology, Dalian, 116020, China E-mail: z.xu@dlut.edu.cn*

aforementioned challenges.

The novelties of the work in this paper are as follows. We study NFV-enabled multicast request admissions in MEC, and formulate three novel optimization problems that explore VNF instance placement and sharing among different multicast requests. We strive for the finest tradeoff between the usages of computing and bandwidth resources to maximize the network throughput while minimizing the accumulative admission cost of admitted requests. We devise the very first approximation algorithm for a single NFV-enabled multicast request admission with the objective to minimize its admission cost, and an efficient heuristic algorithm for the admissions of a set of NFV-enabled multicast requests. Furthermore, we also consider the dynamic admissions of NFV-enabled multicast requests by developing an online algorithm with a provable competitive ratio for it. The key ingredients in the development of these proposed algorithms lie in (a) dynamically determining the use of existing VNF instances or instantiating new VNF instances for each request admission; and (b) determining the admission order of a given set of requests as admitted requests will heavily impact the admissions of future requests, due to the availability of the demanded resources and whether existing VNF instances can be shared by future requests.

The main contributions of this paper are summarized as follows. We study the NFV-enabled multicast request admissions in a mobile edge-cloud network with the aim to either minimize the request admission cost, or maximize the network throughput for a set of requests or a sequence of requests arriving one by one without the knowledge of future arrivals, subject to both computing and bandwidth resource capacities on cloudlets and links in the network. We first propose an approximation algorithm for the cost minimization problem of a single NFV-enabled multicast request admission. We then develop an efficient heuristic for a set of NFV-enabled multicast request admissions, by reducing the problem to the single NFV-enabled multicast request admission. We thirdly consider dynamic NFV-enabled multicast request admissions by devising an online algorithm with a provable competitive ratio. We finally evaluate the performance of the proposed algorithms through experimental simulations. The simulation results reveal that the proposed algorithms are very promising.

The rest of the paper is organized as follows. Section 2 conducts literature review. Section 3 introduces notions, notations, and problem definitions. Section 4 devises an approximation algorithm for the cost minimization problem of a single NFV-enabled multicast request admission. Section 5 develops an efficient heuristic for the throughput maximization problem of a group of NFV-enabled multicast request admissions. Section 6 devises an online algorithm for dynamic NFV-enabled multicast request admissions. Section 7 evaluates the proposed algorithms empirically, and Section 8 concludes the paper.

2 Related work

As a key-enabling technology of 5G, MEC networks have gained tremendous attentions by the research community recently. There are extensive studies of user unicast and multicast request admissions through resource provisioning in MEC networks [3], [5], [6], [8], [11], [12], [19]. For example, Jia *et al.* [9] 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. Ceselli *et al.* [3] focused on 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. Xia *et al.* [23] investigated opportunistic task offloading under link bandwidth, residual energy in mobile devices, and cloudlet computing capacity constraints.

All the aforementioned studies assumed that each task will be allocated with dedicated computing resource, and there is no consideration of utilizing existing VNF instances to serve new tasks. However, many requests usually demand the same type of services. If the VNF instance of a specified service has already been instantiated with sufficient residual processing capacity, the other tasks that request for the service can make use of the VNF instance. Several recent studies explored the placement and sharing of VNF instances [8], [12], [27]. For example, Jia *et al.* [12], [27] studied a novel task offloading problem in an MEC network, where each offloading task requests a network function service with a maximum tolerable delay requirement. They aimed at maximizing the number of requests admitted while minimizing their admission cost, for which they proposed an efficient online algorithm. He *et al.* [8] studied the joint service placement and request scheduling in order to optimally provision edge services while taking into account the demands of both sharable and non-sharable resources. They aim to maximize the network throughput, for which they showed that this joint optimization problem is NP-hard and then developed heuristic algorithms.

There are several studies of NFV-enabled multicasting in MEC environments [2], [12], [26]. For example, Zhang *et al.* [29] 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. In reality, it is not uncommon that both computing and bandwidth resources in MEC are limited, which need to be carefully allocated. Furthermore, they did not consider dynamic admissions of NFV-enabled multicast requests, which is much complicated compared with the problem of admitting a single or a set of given requests. Xu *et al.* [26] studied the cost minimization problem of admitting a single NFV-enabled multicast request, where the implementation of the service chain of each request will be consolidated into a single cloudlet. Xu *et al.* [28] recently considered the admissions of NFV-enabled multicast requests with QoS constraints in MEC by proposing approximation and heuristic algorithms for the problem. Ma *et al.* [17], [18] 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, the problem of NFV-enabled unicast request admissions in [13], [17], [18] is a special case of the problem of NFV-

enabled multicast request admissions where the destination set contains only one node. The essential differences of the study in this paper from these mentioned studies [12], [26], [27], [29] are (i) the VNF instances of the service chain of each NFV-enabled multicast request in this paper can be placed to multiple cloudlets, not just one cloudlet in the previous studies; and (ii) the sharing of existing VNF instances among different multicast requests has not been explored, this exploration makes the problem become more challenging. It is mentioned that this paper is an extended version of a conference paper in [16].

3 PRELIMINARIES

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

3.1 System model

We consider a mobile edge cloud (computing) network (MEC) in a metropolitan region that is modelled by an undirected graph $G = (V, E)$, where V is a set of *access points* (APs) located at different locations in a metropolitan region, e.g., shopping centers, airports, restaurants, bus stations, and hospitals. A cloudlet is co-located with each AP node $v \in V$ via a high-speed optical cable. This implies that the communication delay between them is negligible due to plenty of bandwidth on the cable. For simplicity, each AP node and its co-located cloudlet will be used interchangeably if no confusion arises. Each cloudlet has computing capacity C_v for implementing various virtualized network functions (VNFs). E is the set of links between APs. Each link $e \in E$ has a bandwidth capacity B_e. We assume that each AP node covers a certain area, in which each mobile user can access the MEC service wirelessly through the AP. In case a mobile user located at an overlapping coverage region of multiple APs, the mobile user can connect to its nearest AP or the AP with the strongest signal strength. Figure 1 is an example of an MEC network.

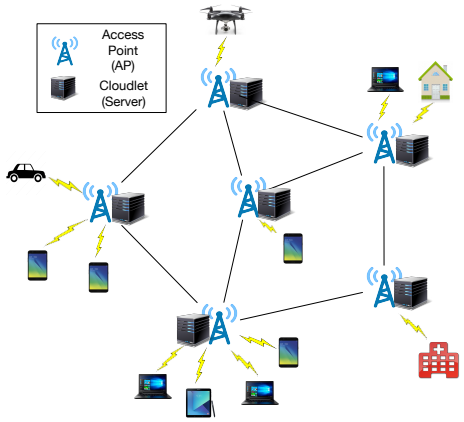


Fig. 1. An illustrative example of an MEC network consisting of 6 APs with each co-located with a cloudlet.

3.2 NFV-enabled multicast requests with service function chain requirements

Consider *an NFV-enabled multicast request* $r_j = (s_j, D_j, \rho_j, SFC_j)$ that transmits its data traffic from the source node $s_j \in V$ to the given set $D_j \subseteq V$ of destination nodes with a specified packet rate ρ_j. Each packet in the data traffic stream must pass through the sequence

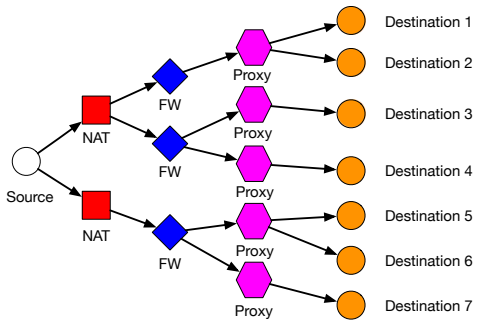


Fig. 2. An example of an NFV-enabled multicast request with a service function chain that consists of three network functions: Network Address Translation (NAT), Firewall (FW), and Proxy. Its data packet traffic flows from the source node $Source$ to a set of seven destination nodes. Each packet must pass through one VNF instance of each of the three network functions in the service function chain.

of network functions of its *service function chain* $SFC_j = \langle f_{j,1}, \ldots, f_{j,l}, \ldots, f_{j,L_j} \rangle$ before reaching each of the destinations, where L_j is the length of the service function chain. We assume that a unit packet rate of r_j requires bandwidth resource b_e in a link $e \in E$, thus, the total amount $\rho_j \cdot b_e$ of bandwidth required for request r_j in link e.

We assume that resources in cloudlets are virtualized, using container-based lightweight virtualization technologies, and thus can be allocated and shared flexibly. Each instance of a *virtualized network function* (VNF) is a virtual machine in a cloudlet. Without loss of generality, we assume that different types of VNFs among all service function chains of requests can be classified into K types. Denote by $f^{(k)}$ and $C(f^{(k)})$ the VNF of type k and the amount of computing resource consumed for its implementation in a cloudlet, respectively, $1 \leq k \leq K$. Suppose each VNF instance of $f^{(k)}$ has *a maximum processing capacity* $\mu^{(k)}$. Furthermore, if the residual processing capacity of an existing VNF instance is sufficient to process the data traffic of a newly admitted request, this VNF instance can be shared by the request. Otherwise, a new VNF instance for the request needs to be instantiated in a cloudlet with sufficient residual computing resource in order to admit the request.

To admit an NFV-enabled multicast request r_j, each packet of its data traffic is enforced to go through a VNF instance of each network function in its SFC_j prior to reaching each of the destinations in D_j. Denote by $T(j)$ *the pseudo-multicast tree* that transmits the data traffic of request r_j from the source s_j to the destinations in D_j, where *a pseudo-multicast tree* [25] in fact may be a graph, not a tree. A pseudo-multicast tree is a directed pseudo-steiner tree which starts from a source node and reaches each node in a destination set. However, due to the availability of some cloudlets in the MEC (i.e., be able to accommodate the VNF instances with sufficient resources), each cloudlet node and physical link of the network may appear multiple times in the pseudo-multicast tree. Figure 2 is an example to illustrate the admission of an NFV-enabled multicast request, where for each network function $f_{j,l}$ in the service function chain SFC_j, either an existing VNF instance (with sufficient residual processing capacity) is selected or a new VNF instance is instantiated in a cloudlet $f_{j,l}$ in each path from the source node s_j to each destination node in D_j, and these VNF instances can be placed at different cloudlets.

3.3 Admission of an NFV-enabled multicast request

The admission cost of an NFV-enabled multicast request in an MEC network is the sum of three constituent costs: the VNF instance processing cost for processing its data packets, the VNF instance instantiation cost for instantiating new VNF instances in cloudlets, and the bandwidth cost for routing its data traffic along links in its pseudo-multicast tree. Instantiating VNF instances at cloudlets consumes both computing and storage resources of the cloudlets, thus incurs the VNF instantiation cost. Denote by $c_{ins}(f^{(k)}, v)$ the instantiation cost a VNF instance of network function $f^{(k)}$ in a cloudlet v, and $\rho_j \cdot c_{proc}(f^{(k)}, v)$ the processing cost of data traffic of a request r_j at a VNF instance of $f^{(k)}$ at cloudlet v, where $c_{proc}(f^{(k)}, v)$ is the cost of processing a packet by a VNF instance $f^{(k)}$ at cloudlet v and ρ_j is the packet rate of r_j. Notice that the processing cost $c_{proc}(f^{(k)}, v)$ of a data packet of different VNF instances at different cloudlets may be significantly different, since different VNF instances consume different amounts of computing resources and different cloudlets have different amounts of energy consumptions. In addition, each packet of the data traffic of request r_j is routed along a pseudo-multicast tree $T(j)$ that incurs the communication cost $\rho_j \cdot \sum_{e \in T(j)} c_e$, where c_e is the unit transmission cost on link $e \in E$, and $\sum_{e \in T(j)} c_e$ is the cost of transferring a packet along the pseudo-multicast tree $T(j)$.

3.4 Problem definitions

In this paper, we consider three NFV-enabled multicast request admission problems in a mobile edge cloud network G, which are defined as follows.

Definition 1: Given an MEC network $G = (V, E)$ with a set V of cloudlets (or APs), each $v \in V$ has computing capacity C_v, let B_e be the bandwidth capacity of each link $e \in E$, assuming that the previous $j-1$ NFV-enabled multicast requests have been responded (admitted or rejected), consider an incoming NFV-enabled multicast request $r_j = (s_j, D_j, \rho_j, SFC_j)$, *the cost minimization problem* of admitting request r_j is to find a pseudo-multicast tree $T(j)$ in G to route its data traffic from the source node s_j to each destination node in D_j while each packet in the data traffic must pass through each VNF instance in the service function chain SFC_j, such that its admission cost is minimized, subject to computing and bandwidth capacities on both cloudlets and links of G.

Definition 2: Given an MEC network $G = (V, E)$ with a set V of cloudlets, each $v \in V$ has computing capacity C_v, and each link $e \in E$ has bandwidth capacity B_e. Let $R = \{r_j = (s_j, D_j, \rho_j, SFC_j) \mid 1 \leq j \leq |R|\}$ be a given set of NFV-enabled multicast requests, the *throughput maximization problem* in G is to maximize the number of requests in R admitted while minimizing the cost sum of their admissions, subject to computing and bandwidth capacities on both cloudlets and links of G.

Definition 3: Given an MEC network $G = (V, E)$ with a set V of cloudlets, each $v \in V$ has computing capacity C_v, and each link $e \in E$ has bandwidth capacity B_e. Let $r_1, r_2, \ldots, r_j$ be a sequence of NFV-enabled multicast requests that arrive one by one without the knowledge of future request arrivals, *the online throughput maximization problem* in G is to maximize the number of requests admitted, subject to computing and bandwidth capacities on both cloudlets and links of G.

3.5 NP hardness of problems

In the following we show that all the three defined optimization problems are NP-hard. Since the online version is a general version of the offline version of a set of request admissions. If the offline version is NP-hard, its online version is NP-hard too. We thus only show the NP-hardness of the first two problems as follows.

Theorem 1. The cost minimization problem in G is NP-hard.

Proof: We show the NP-hardness of the cost minimization problem by a reduction from an NP-hard problem - *the directed Steiner tree problem* that is defined as follows. Given a directed weighted graph $G = (V, A)$ with a set V of vertices and a set A of arcs, a specified node $s \in V$ as the root, and a set of destinations $D \subseteq V$, each arc $a \in A$ has a weight w_a, the objective is to find a minimum cost directed tree rooted at s and spanning all the vertices in D, i.e., there is a directed path from s to every vertex in D.

We show that an instance of the directed Steiner tree problem can be reduced to an instance of the cost minimization problem. Specifically, the root $s \in V$ corresponds to the source s_j of multicast request r_j, and a set of destinations $D \subseteq V$ corresponds to the set of destination nodes of r_j. The multicast request r_j has a service function chain that consists of only one network function. The VNF instance of the network function is deployed in the cloudlet co-located with the source AP s_j, and assume that the processing cost of the VNF instance is 0. The weight w_a on each arc $a \in A$ corresponds to the communication cost on the link. We aim to admit request r_j by routing its data traffic from the source s_j to each destination in D so that the admission cost is minimized. It can be seen that a solution to this cost minimization problem is a solution to the directed Steiner tree problem. The theorem thus holds. □

Theorem 2. The throughput maximization problem in G is NP-hard.

Proof: We prove the NP-hardness of the throughput maximization problem by a reduction from an NP-hard problem - *the knapsack problem* that is defined as follows. Given a bin with capacity B, a set $\mathcal{I}$ of items with each item $i \in \mathcal{I}$ having a specified size $s(i)$ and a profit $p(i)$, the problem is to pack a subset of items into the bin such that the total profit is maximized, subject to the bin capacity B.

We show that an instance of the knapsack problem can be reduced to an instance of the throughput maximization problem. Specifically, the bin B corresponds to a cloudlet with capacity B, a set $\mathcal{I}$ of items corresponds to a set of multicast requests to be admitted and processed by their service function chain instances at the cloudlet. Each multicast request $i \in \mathcal{I}$ has a service function chain with one virtualized network function. Each network function instance for a multicast request i demands computing resource $s(i)$, and the profit $p(i)$ received is 1 if it is admitted. We further assume that no VNF instance is pre-installed in the cloudlet, and each link $e \in E$ has unlimited bandwidth. We aim to admit a subset of requests by creating VNF instances for

them so that the number of multicast requests admitted is maximized, while the size of the cloudlet is bounded by its capacity B. It can be seen that a solution to this special throughput maximization problem is a solution to the knapsack problem. The theorem thus holds. □

3.6 Approximation and competitive ratios

A *γ-approximation algorithm* for a minimization problem P_1 is a polynomial time algorithm $\mathcal{A}$ that delivers an approximate solution for P_1 whose value is no more than γ times the optimal one for any instance of P_1 with $\gamma > 1$, where γ is termed as the approximation ratio of algorithm $\mathcal{A}$.

Let OPT and S be an optimal solution of the offline version of a maximization problem P_2 and the solution delivered by an online algorithm $\mathcal{A}'$ for the online version of P_2. The *competitive ratio* of the online algorithm $\mathcal{A}'$ is ξ if $\frac{S}{OPT} \geq \xi$ for any instance I of problem P_2 with $0 < \xi < 1$.

4 An Approximation Algorithm for the Cost Minimization Problem

In this section, we deal with the cost minimization problem of a single NFV-enabled multicast request admission. We first devise an approximation algorithm for the problem, and then analyze its performance.

4.1 Algorithm overview

Given an MEC $G = (V, E)$ and an NFV-enabled multicast request r_j, we aim to minimize the admission cost of the request by steering its data traffic from the source s_j to the set of destinations in D_j while each packet of the data traffic must pass through a sequence of network functions in its specified service function chain SFC_j. To tackle the problem, it poses three challenges. One is the resource availability in MEC. Whether request r_j should be admitted or not is determined by the availability of its demanded resources in G; the other is which cloudlets should be identified to implement which network functions of its SFC_j; and finally, whether new VNF instances will be instantiated or existing VNF instances can be shared for the implementation of SFC_j must be made dynamically. It is essential to address the aforementioned challenges in order to deliver a cost-efficient solution to the problem.

The basic idea behind the proposed approximation algorithm for the problem is reducing it to the directed multicast tree problem in an auxiliary, directed acyclic graph. If there is a multicast tree in the auxiliary graph rooted at the source s_j and spanning all destinations in D_j, then, request r_j can be admitted, otherwise, r_j should be rejected due to lack of sufficient resources to meet its resource demands. This claim will be shown later in algorithm analysis. A pseudo-multicast tree $T(j)$ in G [25] finally can be derived from the multicast tree $T'(j)$ in the auxiliary graph for the implementation of request r_j.

4.2 Approximation algorithm

Given an NFV-enabled multicast request r_j, we can either make use of existing network function instances as long as their residual processing capacities are sufficient to admit the request. Or if there is sufficient available computing resource in a cloudlet, a new instance for the requested type of network function can be instantiated in the cloudlet. Thus, there are multiple candidate instances for each network function $f_{j,l}$ in its service function chain SFC_j in G to be dynamically determined with $1 \leq l \leq L_j$.

Denote by $\lambda(j,l) = k$ the type of network function which is the lth network function $f_{j,l}$ in SFC_j of request r_j with $1 \leq k \leq K$ and $1 \leq l \leq |SFC_j|$, and denote by $F_v^{(k)}$ the set of VNF instances of type k instantiated in cloudlet v. Let μ_i^{re} be the residual processing capacity of VNF instance $i \in F_v^{(k)}$. Let C_v^{re} be the residual computing capacity of cloudlet $v \in V$. Denote by $N_{l,v}$ the set of VNF instances that can be employed as the lth network function $f_{j,l}$ in SFC_j in cloudlet v, including both existing network function instances with sufficient residual processing capacities, i.e., $\mu_i^{re} \geq \rho_j$ with $i \in F_v^{(\lambda(j,l))}$, as well as a new VNF instance i' to be created providing sufficient computing resource in cloudlet v, i.e, $C_v^{re} \geq C(f^{(\lambda(j,l))})$. Then, N_l is the set of VNF instances that can be employed as the lth network function $f_{j,l}$ in SFC_j among all cloudlets in V, i.e., $N_l = \cup_{v \in V} N_{l,v}$. We assume that the number of VNF instances of the same type in each cloudlet is a small constant. To this end, we construct an auxiliary, directed acyclic graph $G'_j = (V'_j, E'_j)$ for request r_j from G as follows.

Let G' be a subgraph graph of G after removing each link from G if the residual bandwidth of the link is less than $\rho_j \cdot b_e$. The node set V'_j of G'_j is the union on sets N_l of VNF instances with $1 \leq l \leq L_j$, with the source node s_j, the destination node set D_j of multicast request r_j, i.e., $V'_j = \cup_{l=1}^{L_j} N_l \cup \{s_j\} \cup D_j$. To ensure that the network functions of $SFC_j = \langle f_{j,1}, \ldots, f_{j,l}, \ldots, f_{j,L_j} \rangle$ are traversed in order, we add a directed edge from a node $x \in N_{l-1}$ to each node $y \in N_l$ with $2 \leq l \leq L_j$ if there is a shortest path in graph G' between x and y, and the weight $w(x,y)$ assigned to the directed edge is the sum of *the communication cost* along the shortest path in G' between the cloudlets implementing VNF instances x and y and *the processing and VNF instance instantiation cost* of network function y. Notice that if the VNF instance is an existing one, its instantiation cost is 0; and if the two network functions x and y reside in the same cloudlet, their communication cost is 0. We then add a directed edge from s_j to each node $y \in N_1$ if such a shortest path in G' exists, and the weight assigned to the edge is the sum of the communication cost along the shortest path and the processing and VNF instance instantiation cost of network function y. Also, we add a directed edge from each node $x \in N_L$ to a node $y \in D_j$, and set the communication cost along the shortest path from a cloudlet that implements network function x to the AP node y as its weight if such a shortest path in G' exists. Thus, $E'_j = \cup_{l=2}^{L_j} \{\langle x, y \rangle \mid x \in N_{l-1}, y \in N_l\} \cup \{\langle s_j, y \rangle \mid y \in N_1\} \cup \{\langle x, y \rangle \mid x \in N_{L_j}, y \in D_j\}$.

To ensure that a multicast request can be admitted without violating computing capacity of any cloudlet, it must be mentioned that we here adopt a conservative request admission strategy. That is, only if the residual computing capacity of a cloudlet is sufficient to accommodate all necessary VNF instance instantiating (any VNF instances in its SFC_j), it can be allowed to create new VNF instances for request r_j. Figure 3 shows the construction of graph G'_j for request r_j

after the first $j-1$ NFV-enabled multicast requests have been considered. Notice that if there is a shortest path between a VNF instance hosted in cloudlet $u \in V$ and another VNF instance hosted in cloudlet $v \in V$, there will be a shortest path between any pair of VNF instances hosted in these two cloudlets, respectively. For simplicity, we use an edge in the graph to represent a set of edges between each pair of VNF instances residing in the two cloudlets, respectively.

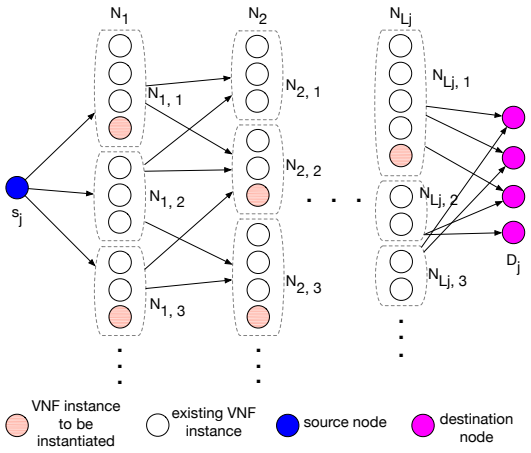


Fig. 3. The auxiliary directed acyclic graph G'_j for NFV-enabled multicast request r_j consists of $L_j + 2$ layers from left to right, where layer 0 is the source node s_j and layer $L_j + 1$ contains all destination nodes in D_j. Each layer l with $1 \leq l \leq L_j$, consists of the VNF instances of type $\lambda(j, l)$ that can be deployed to process the data traffic of request r_j in some of the cloudlets $v \in V$, and if there is sufficient residual computing resource in a cloudlet, a new VNF instance of that type can be instantiated in cloudlet v as well.

Having constructed graph G'_j, the cost minimization problem of the admission of request r_j is reduced to find a directed multicast tree $T'(j)$ in G'_j rooted at s_j and spanning all nodes in D_j, such that the weighted sum of the edges in $T'(j)$ is minimized. Notice that the cost $c(T'(j))$ is *the minimum admission cost* of request r_j in G. This is the classic directed Steiner tree problem, which is NP-hard. There is an approximate solution within $|D_j|^\epsilon$ times of the optimal one [4], where ϵ is a constant with $0 < \epsilon \leq 1$. The value choice of ϵ reflects a tradeoff between the solution accuracy and the running time to obtain the solution. If the multicast tree $T'(j)$ in G'_j rooted at s_j and spanning all destinations in D_j does exist, a pseudo-multicast tree $T(j)$ in G rooted at s_j and spanning all nodes in D_j can then be derived. Specifically, we replace each directed edge in the multicast tree $T'(j)$ by a set of edges in its corresponding shortest path in G. The detailed description of the algorithm for the cost minimization problem is given in `Algorithm` 1.

4.3 Algorithm analysis

In the following, we show the correctness of the proposed algorithm, `Algorithm` 1, and analyze its approximation ratio and time complexity.

Lemma 1. An NFV-enabled multicast request r_j is admissible in G if and only if there is a multicast tree $T'(j)$ in graph G'_j rooted at s_j and spanning all nodes in D_j.

Proof We first show that if there is a multicast tree $T'(j)$ in G' rooted at s_j and spanning all nodes in D_j, there is a feasible solution to the cost minimization problem for request r_j in G. It can be seen that G'_j contains L_j+2 layers with source s_j in layer 0 and all destination nodes in D_j in layer $L_j + 1$. Thus, for each destination node $d \in D_j$, there is a directed path in G'_j from s_j to d that goes through a node in each layer, which implies that each packet of request r_j will be processed by either an existing VNF or a newly instantiated VNF of the network function in that layer, and the segment of the routing path in G that corresponds a directed edge in G'_j has sufficient communication bandwidth to meet the requirement of data traffic of r_j in G. Thus, the solution delivered is a feasible solution.

Algorithm 1 Finding a minimum-cost pseudo-multicast tree in G for request r_j

Input: An MEC network $G = (V, E)$ with a set V of cloudlets. Assume that the first $j-1$ NFV-enabled multicast requests have been considered, and some VNF instances have been instantiated for the admissions of requests. Now consider an NFV-enabled multicast request $r_j = (s_j, D_j, \rho_j, SFC_j)$.
Output: Admit or reject request r_j, and if r_j is admitted, a pseudo-multicast tree $T(j)$ in G will be delivered.
1: A subgraph G' is obtained by removing all edges from G whose residual bandwidth is strictly less than $\rho_j \cdot b_e$;
2: Compute all pairs shortest paths in G' between each pair of AP nodes;
3: Construct the auxiliary directed acyclic graph $G'_j = (V'_j, E'_j)$ from G, and assign a weight on each edge in E'_j;
4: Find an approximate multicast tree $T'(j)$ in G'_j rooted at s_j and spanning all nodes in D_j, by applying the approximation algorithm on G'_j due to Charikar *et al.* [4];
5: **if** $T'(j)$ in G'_j exists **then**
6: A pseudo-multicast tree $T(j)$ in G is derived, by replacing each edge in $T'(j)$ with the edges of its corresponding shortest path in G;
7: If a selected VNF instance is to be instantiated, create a new VNF instance in its cloudlet;
8: Update residual resource capacities of links, cloudlets, and VNF instances in G;
9: **else**
10: Reject request r_j.
11: **end if**

We then show that if there does not exist a multicast tree $T'(j)$ in G'_j rooted at s_j and spanning all nodes in D_j, then request r_j is inadmissible and should be rejected, i.e., there is not sufficient resources in G to admit the request. Assume that a destination node $d \in D_j$ is not reachable from s_j (or d is not contained in $T'(j)$). Assume that node v_l in layer l is the smallest layer from which d is reachable in G'_j with $1 \leq l \leq L_j$, this implies that there is not any directed edge from any node in layer $l-1$ to node v_l in layer l. Following the construction of G'_j, there are three possibilities for absence of any such an edge: (1) among all cloudlets in G, either none of them has sufficient computing resource to instantiate a new VNF for $f_{j,l} \in SFC_j$; or (2) all existing VNF instances of $f_{l,j}$ in these cloudlets have less residual processing capacities for $f_{j,l}$; or (3) there is not any path in G' from any node in layer $l-1$ to a node (cloudlet) in layer l due to the lack of communication bandwidth to meet the bandwidth requirement of r_j. In other words, it is lack of sufficient resources in G to meet the resource demands of r_j, thus, it will be rejected.

Theorem 3. Given an MEC network $G = (V, E)$ with a set V of APs that each is attached a cloudlet, and an NFV-enabled multicast request $r_j = (s_j, D_j, \rho_j, SFC_j)$, there

is an approximation algorithm, Algorithm 1, for the cost minimization problem with an approximation ratio of $|D_j|^{\epsilon}$. The algorithm takes $O((L_j \cdot |V|)^{\frac{1}{\epsilon}}|D_j|^{\frac{2}{\epsilon}} + |V|^3)$ time, where L_j $(= |SFC_j|)$ is the length of SFC_j of request r_j, and ϵ is a constant with $0 < \epsilon \leq 1$.

Proof The solution obtained by the proposed algorithm Algorithm 1 is feasible, which has been shown by Lemma 1. In the following, we analyze the approximation ratio of the proposed algorithm. The admission cost of multicast request r_j is the sum of (i) the VNF instance processing cost; (ii) the VNF instance instantiation cost, and (iii) the communication bandwidth usage cost. Each packet of the data traffic of request r_j is transferred from the source node s_j to each destination node in D_j while passing through each VNF instance in its service function chain SFC_j. The sum of these three costs is assigned to each directed edge in E'_j. Thus, the cost of the minimum Steiner tree $T'(j)$ found in G'_j rooted at s_j and spanning all nodes in D_j, is the minimum admission cost of r_j in G. Following [4], the approximation ratio of the proposed algorithm for the cost minimization problem for a single multicast request admission is $|D_j|^{\epsilon}$, where ϵ is a constant with $0 < \epsilon \leq 1$.

We finally analyze the time complexity of Algorithm 1 as follows. Finding all pairs shortest paths in G' between each pair of AP nodes takes $O(|V|^3)$ time, by invoking the well-known Floyd-Warshall algorithm. The construction of the auxiliary directed acyclic graph G'_j for each request r_j takes $O(L_j \cdot |V|)$ time, since there are $L_j + 2$ layers in G'_j and each each layer contains $O(|V|)$ nodes, assuming that the number of VNF instances of the same type in each cloudlet is a small constant, i.e., $O(1)$. Recall that $L_j = |SFC_j|$. Finding an approximate multicast tree in G'_j for request r_j takes time $O(|V'_j|^{\frac{1}{\epsilon}}|D_j|^{\frac{2}{\epsilon}})$, by applying the $(|D_j|^{\epsilon})$-approximation algorithm due to Charikar *et al.* [4]. Thus, the running time of Algorithm 1 is $O((L_j \cdot |V|)^{\frac{1}{\epsilon}}|D_j|^{\frac{2}{\epsilon}} + |V|^3)$ where ϵ is a constant with $0 < \epsilon \leq 1$.

5 An efficient algorithm for the throughput maximization problem

In this section, we deal with the throughput maximization problem, by reducing the problem to the cost minimization problem in the previous section.

5.1 Algorithm

The proposed algorithm proceeds iteratively. Within each iteration, a request is admitted if its admission cost is the minimum one among non-admitted requests. This procedure continues until not any request can be admitted due to the lack of resources to accommodate any of them.

Specifically, denote by $\mathcal{U}$ the set of non-admitted requests, and $cost(r_j)$ the admission cost of request $r_j \in \mathcal{U}$ which is the cost of the multicast tree $T'(j)$ constructed in the previous section. Notice that if an NFV-enabled multicast request is rejected by Algorithm 1 due to the violation of either all computing capacities of cloudlets or bandwidth capacities of links, its admission cost $cost(r_j)$ is set as $+\infty$. The request then will be removed from the non-admitted request set $\mathcal{U}$ for good. This claim can be proven easily. With more and more request admissions, the resource utilization ratio in G increases. If a request cannot be admitted at the current iteration, there must be no sufficient resources to meet its demands, and it cannot be admitted in any iteration in future. The detailed algorithm for the throughput maximization problem is described in Algorithm 2.

Algorithm 2 An algorithm for the throughput maximization problem

Input: Given an MEC network $G = (V, E)$, each cloudlet $v \in V$ has computing capacity C_v and each link $e \in E$ has bandwidth capacity B_e, and a set of NFV-enabled multicast requests $R = \{r_j = (s_j, D_j, \rho_j, SFC_j)\}$.

Output: Maximize the network throughput while minimizing the total admission cost of admitted requests, by deploying VNF instances in cloudlets and routing data traffic along its pseudo-multicast tree in G for each admitted request.

$flag \leftarrow true$;
$admissionCost \leftarrow 0$; /* the cost of all multicast request admissions */
$A \leftarrow \emptyset$; /* the set of admitted multicast requests */
$\mathcal{U} \leftarrow R$; /* the set of non-admitted requests in R */
while $flag$ **do**
  $cost \leftarrow +\infty$; /* record the minimum admission cost */
  $r_{min} \leftarrow \emptyset$; /* record the multicast request with the minimum admission cost */
  for request $r_j \in \mathcal{U}$ **do**
    Calculate the admission cost $cost(r_j)$ of request r_j, by invoking Algorithm 1;
    if r_j is rejected **then**
      $cost(r_j) \leftarrow +\infty$; /* reject request r_j and its admission cost is $+\infty$ */
      $\mathcal{U} \leftarrow \mathcal{U} \setminus \{r_j\}$;
    end if
    if $cost(r_j) < cost$ **then**
      $cost \leftarrow cost(r_j)$; /* update local variable $cost$ */
      $r_{min} \leftarrow r_j$; /* update local variable r_{min} */
    end if
  end for
  if $cost \neq +\infty$ **then**
    $admissionCost \leftarrow admissionCost + cost$;
    $A \leftarrow A \cup \{r_j\}$;
    Create new VNF instances if required by checking the pseudo-multicast tree $T(j)$ of r_j;
    Update residual resource capacities of all involving VNF instances, cloudlets, and links in G;
  else
    $flag \leftarrow false$;
  end if
  if $flag$ **then**
    $\mathcal{U} \leftarrow \mathcal{U} \setminus \{r_{min}\}$;
  end if
end while
return $admissionCost$, A;

5.2 Time complexity of the proposed algorithm

We now analyze the time complexity of Algorithm 2 for the throughput maximization problem as follows.

Theorem 4. Given an MEC network $G = (V, E)$ with a set V of APs in which each AP v is attached a cloudlet of computing capacity C_v, and a set of NFV-enabled multicast requests $R = \{r_j = (s_j, D_j, \rho_j, SFC_j)\}$, there is an algorithm, Algorithm 2, for the throughput maximization problem, which takes $O(|R|^2 \cdot (L_{max} \cdot |V|)^{\frac{1}{\epsilon}}|D_{max}|^{\frac{2}{\epsilon}} + |V|^3)$ time, where L_{max} is the maximum length of all service function chains SFC_j of any request r_j with $1 \leq j \leq |R|$, $|D_{max}| = \max_{1 \leq j \leq |R|}\{|D_j|\}$, and ϵ is a constant with $0 < \epsilon \leq 1$.

Proof The time complexity of `Algorithm 2` is analyzed as follows. Within each iteration, each multicast request in the non-admitted request set $\mathcal{U}$ ($= O(|R|)$) is examined by invoking `Algorithm 1`. There are at most $O(|R|)$ iterations, thus the running time of `Algorithm 2` is $O(|R|^2 \cdot (L_{max} \cdot |V|)^{\frac{1}{\epsilon}}|D_{max}|^{\frac{2}{\epsilon}} + |V|^3)$. Notice that all pairs shortest paths in G between each pair of AP nodes is calculated only once.

6 An online algorithm for the online throughput maximization problem

In this section, we study the online throughput maximization problem, where NFV-enabled multicast requests arrive one by one without the knowledge of future request arrivals. We first propose an online algorithm for the problem, through building a novel cost model to capture dynamic resource consumptions in G and performing resource allocations for request admissions based on the built cost model. We then analyze the competitive ratio and time complexity of the proposed online algorithm.

6.1 The usage cost model of resources

The basic idea behind the proposed online algorithm is to regulate an online admission control policy to respond to each arrived NFV-enabled multicast request by either admitting or rejecting it, depending on the availability of its demanded resources and a given admission control policy. We still make use of the auxiliary directed acyclic graph G'_j as an important data structure for the online throughput maximization problem. The weight assigned to each edge in G'_j here however is different from its weight in the previous section that is defined as follows.

We here introduce a resource usage cost model to measure all different types of resource consumptions of each VNF instance (processing capacity), each cloudlet (computing resource), and each link (bandwidth resource) when admitting requests. Given the dynamics of resource demands of user requests and occupied resource releasing in the network, there is a need of a cost model to capture the dynamic consumptions of various resources in the network in order to assist the admissions of future requests and better utilize the resources. Intuitively, overloaded resources usually have higher probabilities to be violated by the resource demands of currently admitted requests, due to the high dynamics of resource consumptions. This eventually will affect the admissions of future requests. Therefore, if a specific type of resource has been highly utilized, it should be assigned a higher usage cost to reduce its usage in future; otherwise, it should be assigned a lower usage cost to encourage its usage in future.

The proposed online algorithm examines each incoming NFV-enabled multicast request one by one. When request r_j arrives, the resource availabilities of the VNF instances of network functions in its service chain, computing resources in cloudlets, and bandwidth resources in links will determine whether it is admissible. Recall that $F_v^{(k)}$ is the set of existing VNF instances of type k in cloudlet v. If there is sufficient computing resource in cloudlet v, a new VNF instance of type k can be instantiated at it. For the sake of convenience, assume that set $F_v^{(k)}$ contains the newly instantiated VNF instance of type k as well.

Denote by $\mu_{v,i}^{(k)}(j)$ the residual processing capacity of the VNF instance $i \in F_v^{(k)}$ of type k in cloudlet v when request r_j arrives with $\mu_{v,i}^{(k)}(0) = \mu^{(k)}$ initially. If request r_j is admitted and its packets is processed by the VNF instance i, then $\mu_{v,i}^{(k)}(j) = \mu_{v,i}^{(k)}(j-1) - \rho_j$, otherwise, its residual computing capacity does not change.

As for each network function in service function chain SFC_j of r_j, a new VNF instance of it can be instantiated or an existing VNF instance of it can be shared, a binary variable $x_v^{(\lambda(j,l))}$ is introduced for each network function $f_{j,l}$ in SFC_j with $1 \leq l \leq |SFC_j| = L_j$, where $x_v^{(\lambda(j,l))}$ is 1 if the lth VNF instance is newly instantiated in cloudlet v; otherwise 0. Then, denote by $C_v(j)$ the residual computing capacity at cloudlet $v \in V$ when request r_j arrives with $C_v(0) = C_v$ initially. If request r_j is admitted and some VNF instances are instantiated in cloudlet v, then $C_v(j) = C_v(j-1) - \sum_{l=1}^{L_j} C(f^{(\lambda(j,l))}) \cdot x_v^{(\lambda(j,l))}$. Similarly, denote by $B_e(j)$ the residual bandwidth in link $e \in E$ when request r_j arrives with $B_e(j) = B_e(j-1) - \rho_j \cdot b_e$ if request r_j is admitted and $B_e(0) = B_e$.

To capture the resource usage of request r_j, we use an exponential function to model the cost $W_{v,i}^{(k)}(j)$ of processing packets of r_j by the VNF instance $i \in F_v^{(k)}$ as follows,

$$W_{v,i}^{(k)}(j) = \mu^{(k)}(\alpha^{1-\frac{\mu_{v,i}^{(k)}(j)}{\mu^{(k)}}} - 1), \tag{1}$$

where α (> 1) is a tuning parameter to be decided later, and $1 - \frac{\mu_{v,i}^{(k)}(j)}{\mu^{(k)}}$ is the processing capacity utilization ratio in the VNF instance i when request r_j is considered. Similarly, the cost $W_v(j)$ of instantiating new VNF instances for request r_j at cloudlet $v \in V$ and the cost $W_e(j)$ of using bandwidth resource at link $e \in B$ are defined, respectively,

$$W_v(j) = C_v(\beta^{1-\frac{C_v(j)}{C_v}} - 1), \tag{2}$$

$$W_e(j) = B_e(\gamma^{1-\frac{B_e(j)}{B_e}} - 1), \tag{3}$$

where β (> 1) and γ (> 1) are tuning parameters to be decided later, and $1 - \frac{C_v(j)}{C_v}$ and $1 - \frac{B_e(j)}{B_e}$ are the resource utilization ratios in cloudlet v and link e, respectively, when request r_j is considered. In order to encourage the sharing of VNF instances among multicast requests, we assume that the cost of creating a new VNF instance is much higher than the cost of processing capacity usage, i.e., $\beta \gg \alpha$.

We then define *the normalized usage cost* $\omega_{v,i}^{(k)}(j)$ of each VNF instance $i \in F_v^{(k)}$ in cloudlet v for request r_j as follows,

$$\omega_{v,i}^{(k)}(j) = W_{v,i}^{(j)}(j)/\mu^{(k)} = \alpha^{1-\frac{\mu_{v,i}^{(k)}(j)}{\mu^{(k)}}} - 1. \tag{4}$$

Similarly, the normalized usage costs $\omega_v(j)$ at each cloudlet $v \in V$ and $\omega_e(j)$ at each link $e \in E$ for request r_j are defined as follows,

$$\omega_v(j) = W_v(j)/C_v = \beta^{1-\frac{C_v(j)}{C_v}} - 1, \tag{5}$$

$$\omega_e(j) = W_e(j)/B_e = \gamma^{1-\frac{B_e(j)}{B_e}} - 1. \tag{6}$$

Having defined the usage costs of different resources

in G, now consider the current incoming NFV-enabled multicast request r_j, we construct an auxiliary graph $G'_j = (V'_j, E'_j)$ which is almost identical to the one for the cost minimization problem. The difference lies in the weight assignment of edges in G'_j. Specifically, here the weight assigned to each directed edge in E'_j is the sum of the three normalized constituent usage costs defined in (4), (5), and (6), respectively. That is, each edge $(x, y) \in E'_j$ has a weight

$$w(x,y) = \omega_{v,y}^{(\lambda(j,l))}(j) + \omega_v(j) + \sum_{e \in P(u,v)} \omega_e(j), \qquad (7)$$

assuming that x is a VNF instance in level $l-1$ deployed in cloudlet u, y is a VNF instance in level l deployed in cloudlet v, $P(u,v)$ is a shortest path in G between cloudlets u and v.

To avoid admitting requests that consume too much resources, thereby undermining the performance of the MEC, we adopt the following admission control policy. If (i) the sum of normalized usage costs of the VNF instances in its service function chain is greater than a given threshold σ_1, i.e., $\sum_{v \in V} \sum_{l=1}^{L_j} \sum_{i \in F_v^{(\lambda(j,l))}} \omega_{v,i}^{(\lambda(j,l))}(j) > \sigma_1$, where $L_j = |SFC_j|$; or (ii) the sum of normalized usage costs of its VNF instantiations is greater than another given threshold σ_2, $\sum_{v \in V} \omega_v(j) > \sigma_2$; or (iii) the sum of normalized usage costs of its bandwidth in links is greater than the third threshold σ_3, $\sum_{e \in E} \omega_e(j) > \sigma_3$, request r_j will be rejected, where $\sigma_1 = \sigma_2 = \sigma_3 = n$, and $n = |V|$. The detailed algorithm for the online throughput maximization problem is given in `Algorithm` 3.

Algorithm 3 Online algorithm for the online throughput maximization problem

Input: An MEC network $G = (V, E)$ with a set V of APs, each $v \in V$ is attached a cloudlet with computing capacity C_v, a sequence of NFV-enabled multicast requests $r_j = (s_j, D_j, \rho_j, SFC_j)$ arriving one by one without the knowledge of future arrivals.

Output: Maximize the network throughput by admitting or rejecting each arrived request r_j immediately. If r_j admitted, a pseudo-multicast tree $T(j)$ for r_j in G from source node s_j to a set of destination nodes in D_j will be delivered.

while request r_j arrives **do**
  A subgraph G' of G is constructed by removing each edge with residual bandwidth capacity less than $\rho_j \cdot b_e$;
  Construct the auxiliary graph $G'_j = (V'_j, E'_j)$ for request r_j, assign a weight to each edge in E'_j according to Eq. (7);
  Find an approximate multicast tree $T'(j)$ in G'_j rooted at s_j and spanning all nodes in D_j, by applying the approximation algorithm on G'_j due to Charikar *et al.* [4];
  if $T'(j)$ does not exist **then**
    Reject request r_j;
  else
    Determine whether r_j will be accepted by the admission control policy;
    if r_j is admissible **then**
      A pseudo-multicast tree $T(j)$ in G is derived from $T'(j)$, by replacing each edge in $T'(j)$ by the edges in its corresponding shortest path in G;
      If a VNF instance in a cloudlet is to be instantiated, create the new VNF instance;
      Update residual resource capacities of VNF instances, links and cloudlets in G;
    end if
  end if
end while

6.2 Algorithm analysis

We now analyze the competitive ratio and time complexity of the proposed online algorithm, `Algorithm` 3. We first show the upper bound on the total cost of admitted requests. We then provide a lower bound on the cost of a rejected request by `Algorithm` 3 but admitted by an optimal offline algorithm. We finally derive the competitive ratio of `Algorithm` 3.

Lemma 2. Given an MEC network $G = (V, E)$, with each cloudlet $v \in V$ has computing capacity C_v and a set E of links that each link $e \in E$ has bandwidth capacity B_e, denote by $\mathcal{A}(j)$ the set of NFV-enabled multicast requests admitted by the algorithm, `Algorithm` 3, until the arrival of request r_j. Then, the cost sums of VNF instances, cloudlets, and links when multicast request r_j arrives are

$$\sum_{v \in V} \sum_{l=1}^{L_j} \sum_{i \in F_v^{(\lambda(j,l))}} W_{v,i}^{(\lambda(j,l))}(j) \le 2n \log \alpha \cdot \mathbb{B}(j), \qquad (8)$$

$$\sum_{v \in V} W_v(j) \le 2nL_{max} \log \beta \cdot |\mathcal{A}(j)| \cdot C(f_{max}), \qquad (9)$$

$$\sum_{e \in E} W_e(j) \le 2n \log \gamma \cdot \mathbb{B}(j), \qquad (10)$$

respectively, provided that the maximum length L_{max} of any service function chain is no greater than n, i.e., $L_{max} = \max_{1 \le j' \le j}\{|SFC_{j'}|\} \le n$, and $\rho_{j'} \le \frac{\min_{1 \le l \le L_{j'}}\{\mu^{(\lambda(j',l))}\}}{\log \alpha}$, $\sum_{l=1}^{L_{j'}} C(f^{(\lambda(j',l))}) \cdot x_v^{(\lambda(j',l))} \le \frac{\min_{v \in V} C_v}{\log \beta}$, $\rho_{j'} \cdot b_e \le \frac{\min_{e \in E} B_e}{\log \gamma}$ with $1 \le j' \le j$, where $\sum_{l=1}^{L_{j'}} C(f^{(\lambda(j',l))}) \cdot x_v^{(\lambda(j',l))}$ is the computing resource being occupied by newly instantiated VNF instances in cloudlet v for request $r_{j'}$, $\mathbb{B}(j)$ is the accumulative bandwidth resource being occupied by the admitted requests, i.e., $\mathbb{B}(j) = \sum_{r_{j'} \in \mathcal{A}(j)} \rho_{j'} \cdot b_e$, and $C(f_{max})$ is the maximum computing resource required among all VNF instance types, i.e., $C(f_{max}) = \max_{1 \le k \le K}\{C(f^{(k)})\}$.

Proof Consider a request $r_{j'} \in \mathcal{A}(j)$ admitted by `Algorithm` 3. For any VNF instance $i \in F_v^{(k)}$, we have

$$\begin{aligned}
&W_{v,i}^{(k)}(j'+1) - W_{v,i}^{(k)}(j') \\
&= \mu^{(k)}(\alpha^{1-\frac{\mu_{v,i}^{(k)}(j'+1)}{\mu^{(k)}}} - 1) - \mu^{(k)}(\alpha^{1-\frac{\mu_{v,i}^{(k)}(j')}{\mu^{(k)}}} - 1) \\
&= \mu^{(k)}\alpha^{1-\frac{\mu_{v,i}^{(k)}(j')}{\mu^{(k)}}}(\alpha^{\frac{\mu_{v,i}^{(k)}(j') - \mu_{v,i}^{(k)}(j'+1)}{\mu^{(k)}}} - 1) \\
&= \mu^{(k)}\alpha^{1-\frac{\mu_{v,i}^{(k)}(j')}{\mu^{(k)}}}(\alpha^{\frac{\rho_{j'}}{\mu^{(k)}}} - 1) \\
&= \mu^{(k)}\alpha^{1-\frac{\mu_{v,i}^{(k)}(j')}{\mu^{(k)}}}(2^{\frac{\rho_{j'}}{\mu^{(k)}}\log\alpha} - 1) \\
&\le \mu^{(k)}\alpha^{1-\frac{\mu_{v,i}^{(k)}(j')}{\mu^{(k)}}} \cdot \frac{\rho_{j'}}{\mu^{(k)}} \cdot \log\alpha \qquad (11) \\
&= \alpha^{1-\frac{\mu_{v,i}^{(k)}(j')}{\mu^{(k)}}} \cdot \rho_{j'} \cdot \log\alpha, \qquad (12)
\end{aligned}$$

where Ineq. (11) holds due to that $2^a - 1 \le a$ for $0 \le a \le 1$.

Similarly, for any cloudlet $v \in V$, we have $W_v(j'+1) - W_v(j') \le \beta^{1-\frac{C_v(j')}{C_v}}(\sum_{l=1}^{L_{j'}} C(f^{(\lambda(j',l))}) \cdot x_v^{(\lambda(j',l))}) \log\beta$ and

for any link $e \in E$, we have $W_e(j'+1) - W_e(j') \leq \gamma^{1-\frac{B_e(j')}{B_e}} \cdot \rho_{j'} \cdot b_e \cdot \log \gamma$.

We then calculate the cost sum of all VNF instances when admitting request $r_{j'}$. The difference of the cost sum of VNF instances before and after admitting request $r_{j'}$ is

$$\begin{aligned}
&\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(\lambda(j',l))}} W_{v,i}^{(k)}(j'+1) - W_{v,i}^{(k)}(j') \\
&= \sum_{l=1}^{L_{j'}} W_{v,i}^{(k)}(j'+1) - W_{v,i}^{(k)}(j') \qquad (13)\\
&\leq \sum_{l=1}^{L_{j'}} \alpha^{1-\frac{\mu_{v,i}^{(\lambda(j',l))}(j')}{\mu^{(\lambda(j',l))}}} \cdot \rho_{j'} \cdot \log \alpha, \quad \text{by Ineq. (12)} \\
&= \rho_{j'} \cdot \log \alpha \sum_{l=1}^{L_{j'}} \alpha^{1-\frac{\mu_{v,i}^{(\lambda(j',l))}(j')}{\mu^{(\lambda(j',l))}}}, \\
&= \rho_{j'} \cdot \log \alpha \left(\sum_{l=1}^{L_{j'}} (\alpha^{1-\frac{\mu_{v,i}^{(\lambda(j',l))}(j')}{\mu^{(\lambda(j',l))}}} - 1) + \sum_{l=1}^{L_{j'}} 1 \right) \\
&= \rho_{j'} \cdot \log \alpha \left(\sum_{l=1}^{L_{j'}} \omega_{v,i}^{(k)}(j') + L_{j'} \right) \leq 2n\rho_{j'} \cdot \log \alpha \qquad (14)
\end{aligned}$$

Ineq. (12) holds since $\sum_{i=1}^{n} A_i \cdot B_i \leq \sum_{i=1}^{n} A_i \cdot \sum_{i=1}^{n} B_i$, for all $A_i \geq 0$ and $B_i \geq 0$. Eq. (13) holds due to that for each network function $f_{j',l}$, only one VNF instance is employed to process data traffic of request $r_{j'}$. Ineq. (14) holds due to the fact that if request $r_{j'}$ is admitted, the admission control policy is met, i.e., $\sum_{l=1}^{L_{j'}} \omega_{v,i}^{(\lambda(j',l))}(j') = \sum_{l=1}^{L_{j'}} \alpha^{1-\frac{\mu_{v,i}^{(\lambda(j',l))}(j')}{\mu^{(\lambda(j',l))}}} - 1 \leq \sigma_1 = n$, and the length of service function chain of request $r_{j'}$ is less than the number of APs, i.e., $|SFC_{j'}| = L_{j'} \leq L_{max} \leq n$.

Similarly, the difference of the cost sum of cloudlets before and after admitting request $r_{j'}$ is $\sum_{v\in V} W_v(j'+1) - W_v(j') \leq 2nL_{j'} \cdot C(f_{max}) \cdot \log \beta$, where $C(f_{max})$ is the maximum computing resource consumption of any VNF instance $f^{(k)}$, $1 \leq k \leq K$ in the MEC. And the difference of the cost sum of links before and after admitting request $r_{j'}$ is $\sum_{e\in E} W_e(j'+1) - W_e(j') \leq 2n\rho_{j'} \cdot b_e \cdot \log \gamma$.

The cost sum of VNF instances for request admissions when r_j arrives thus is

$$\begin{aligned}
&\sum_{v\in V}\sum_{l=1}^{L_j}\sum_{i\in F_v^{(k)}} W_{v,i}^{(k)}(j) \\
&= \sum_{j'=1}^{j-1}\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(k)}} W_{v,i}^{(k)}(j'+1) - W_{v,i}^{(k)}(j') \qquad (15)\\
&= \sum_{r_{j'}\in\mathcal{A}(j)}\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(\lambda(j',l))}} (W_{v,i}^{(k)}(j'+1) - W_{v,i}^{(k)}(j')) \\
&\leq \sum_{r_{j'}\in\mathcal{A}(j)} 2n\rho_{j'} \cdot \log \alpha, \quad \text{by Ineq. (14)} \\
&= 2n \log \alpha \sum_{r_{j'}\in\mathcal{A}(j)} \rho_{j'} = 2n \log \alpha \cdot \mathbb{B}(j),
\end{aligned}$$

where Eq. (15) follows from the fact that if a request is not admitted, none of the processing capacity of any VNF instance will be consumed.

Similarly, the cost sum of cloudlets for request admissions when r_j arrives is $\sum_{v\in V} W_v(j) \leq 2nL_{max} \log \beta \cdot |\mathcal{A}(j)| \cdot C(f_{max})$, and the cost sum of links for request admissions when r_j arrives is $\sum_{e\in E} W_e(j) \leq 2n \log \gamma \cdot \mathbb{B}(j)$.

We now provide a lower bound on the weight of a rejected request by `Algorithm` 3 but admitted by an optimal offline algorithm denoted by OPT. Before we proceed, we choose appropriate values for α, β, and γ prior to the arrival of any request r_j and VNF instance k, $1 \leq k \leq K$ as follows.

$$2n+2 \leq \alpha \leq \min_{1\leq k\leq K}\{2^{\frac{\mu^{(k)}}{\rho_j}}\} \qquad (16)$$

$$2n+2 \leq \beta \leq \min_{1\leq k\leq K}\min_{v\in V}\{2^{\frac{C_v}{C(f^{(k)})}}\} \qquad (17)$$

$$2n+2 \leq \gamma \leq \min_{e\in E}\{2^{\frac{B_e}{\rho_j \cdot b_e}}\} \qquad (18)$$

Lemma 3. Let $\mathcal{T}(j)$ be the set of requests that are rejected by `Algorithm` 3 but admitted by the optimal offline algorithm OPT prior to the arrival of request r_j. Then, for any request $r_{j'} \in \mathcal{T}(j)$, we have

$$\begin{aligned}
&\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(\lambda(j',l))}} \omega_{v,i}^{(\lambda(j',l))}(j') + \sum_{v\in V}\omega_v(j') + \sum_{e\in E}\omega_e(j') \\
&> \min\{\sigma_1, \sigma_2, \sigma_3\} = n.
\end{aligned}$$

Proof Consider a request $r_{j'}$ that is admitted by the optimal offline algorithm OPT yet rejected by `Algorithm` 3. A request r'_j will be rejected by `Algorithm` 3 by one of the four cases: (1) at least one VNF instance does not have sufficient processing capacity to admit request $r_{j'}$; (2) there is no sufficient computation resource in cloudlets to create new VNF instances for request $r_{j'}$ as required; (3) there is no sufficient bandwidth in G for routing its data traffic; or (4) the sum of normalized usage costs is too high, in other words, the admission control policy is not met.

Case (1). At least one VNF instance i' of type k' in cloudlet v' does not have sufficient processing capacity to process data traffic of request $r_{j'}$, i.e., $\mu_{v',i'}^{(k')}(j') < \rho_{j'}$. We then have

$$\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(\lambda(j',l))}} \omega_{v,i}^{(\lambda(j',l))}(j') \geq \omega_{v',i'}^{(k')}(j') \qquad (19)$$

$$\begin{aligned}
&= \alpha^{1-\frac{\mu_{v',i'}^{(k')}(j')}{\mu^{(k')}}} - 1 > \alpha^{1-\frac{\rho_{j'}}{\mu^{(k')}}} - 1, \quad \text{since } \mu_{v',i'}^{(k')}(j') < \rho_{j'} \\
&\geq \alpha^{1-\frac{1}{\log \alpha}} - 1 = \frac{\alpha}{2} - 1 \geq n, \quad \text{by Ineq. (16)}
\end{aligned}$$

Case (2). At least one cloudlet $v' \in V$ does not have sufficient capacity to create a new instance for a VNF of type k' in $SFC_{j'}$ as required, i.e., $C_{v'}(j') < C(f^{(k')})$. Similarly, we have $\sum_{v\in V} \omega_v(j') \geq \omega_{v'}(j') \geq \frac{\beta}{2} - 1 \geq n$.

Case (3). If request $r_{j'}$ is rejected, then there is an edge $e' \in E$ that does not have sufficient residual bandwidth to accommodate the request. This implies that $B_{e'}(j') < \rho_{j'} \cdot b_e$. Therefore, the normalized cost sum of E is greater than σ_3, i.e., $\sum_{e\in E} \omega_e(j') \geq \omega_{e'}(j') \geq \frac{\gamma}{2} - 1 \geq n$.

Case (4). Although there are sufficient resources to admit request $r_{j'}$, $r_{j'}$ is rejected by `Algorithm` 3 due to not meeting the admission control policy. That is

$$\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(\lambda(j',l))}}\omega_{v,i}^{(\lambda(j',l))}(j')+\sum_{v\in V}\omega_v(j')+\sum_{e\in E}\omega_e(j') > \min\{\sigma_1,\sigma_2,\sigma_3\}=n. \quad (20)$$

Lemma 3 thus follows.

We finally analyze the competitive ratio of `Algorithm` 3.

Theorem 5. Given an MEC network $G=(V,E)$ with a set V of APs in which each $v\in V$ is attached a cloudlet with computing capacity C_v, each link $e\in E$ has bandwidth capacity B_e, there is an online algorithm, `Algorithm` 3, with competitive ratio of $O(\log n)$ for the online throughput maximization problem, and the algorithm takes $O((L_j\cdot|V|)^{\frac{1}{\epsilon}}|D_j|^{\frac{2}{\epsilon}})$ time to admit each request r_j where $n=|V|$, $L_j=|SFC_j|$, and ϵ is a constant with $0<\epsilon\leq 1$.

Proof Denote by D_{max} and ρ_{max} the maximum cardinality of destination set $D_{j'}$ and the maximum packet rate of request $r_{j'}$ among all requests respectively, prior to the arrival of request r_j, i.e., $D_{max}=\max_{1\leq j'\leq j}\{D_{j'}\}$, and $\rho_{max}=\max_{1\leq j'\leq j}\{\rho_{j'}\}$. We first analyze the competitive ratio of the proposed online algorithm. We here abuse the notation OPT to denote the optimal offline algorithm OPT and the number of requests admitted by it. Let $\mathcal{A}(j)$ be the set of admitted requests when request r_j arrives, we have

$$\frac{n}{D_{max}^{\epsilon}}(OPT-|\mathcal{A}(j)|)\leq\frac{n}{D_{max}^{\epsilon}}\sum_{r_{j'}\in\mathcal{T}(j)}1$$

$$\leq\sum_{r_{j'}\in\mathcal{T}(j)}n \quad (21)$$

$$\leq\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{v\in V}\sum_{l=1}^{L_{j'}}\sum_{i\in F_v^{(\lambda(j',l))}}\omega_{v,i}^{(\lambda(j',l))}(j')+\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{v\in V}\omega_v(j')+\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{e\in E}\omega_e(j')$$

$$\leq\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{v\in V}\sum_{l=1}^{L_j}\sum_{i\in F_v^{(\lambda(j,l))}}\omega_{v,i}^{(\lambda(j,l))}(j)+\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{v\in V}\omega_v(j)+\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{e\in E}\omega_e(j), \quad (22)$$

$$=\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{v\in V}\sum_{l=1}^{L_j}\sum_{i\in F_v^{(\lambda(j,l))}}\frac{W_{v,i}^{(\lambda(j,l))}(j)}{\mu^{(\lambda(j,l))}}+\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{v\in V}\frac{W_v(j)}{C_v}+\sum_{r_{j'}\in\mathcal{T}(j)}\sum_{e\in E}\frac{W_e(j)}{B_e}$$

$$=\sum_{v\in V}\sum_{l=1}^{L_j}\sum_{i\in F_v^{(\lambda(j,l))}}W_{v,i}^{(\lambda(j,l))}(j)\sum_{r_{j'}\in\mathcal{T}(j)}\frac{1}{\mu^{(\lambda(j,l))}}+\sum_{v\in V}W_v(j)\sum_{r_{j'}\in\mathcal{T}(j)}\frac{1}{C_v}+\sum_{e\in E}W_e(j)\sum_{r_{j'}\in\mathcal{T}(j)}\frac{1}{B_e} \quad (23)$$

$$\leq\sum_{v\in V}\sum_{l=1}^{L_j}\sum_{i\in F_v^{(\lambda(j,l))}}W_{v,i}^{(\lambda(j,l))}(j)+\sum_{v\in V}W_v(j)+\sum_{e\in E}W_e(j) \quad (24)$$

$$\leq 2n\mathbb{B}(j)\log\alpha+2nL_{max}C(f_{max})\log\beta\cdot|\mathcal{A}(j)|+2n\mathbb{B}(j)\log\gamma$$

$$\leq 2n|\mathcal{A}(j)|\big(\rho_{max}\log\alpha+L_{max}\cdot C(f_{max})\log\beta+\rho_{max}\cdot b_e\log\gamma\big).$$

Ineq. (21) holds since $D_{max}\geq 1$, and $0<\epsilon\leq 1$, thus $D_{max}^{\epsilon}\geq 1$. Ineq. (22) holds since the resource utilization ratio does not decrease and thus the usage cost of each VNF instance, each cloudlet, and each link does not decrease with more request admissions. Ineq. (23) holds because $\sum_{i=1}^{m}\sum_{j=1}^{n}A_i\cdot B_j\leq\sum_{i=1}^{m}A_i\cdot\sum_{j=1}^{n}B_j$, for all $A_i\geq 0$ and $B_j\geq 0$. Ineq. (24) holds because all algorithms, including the optimal offline algorithm OPT, the accumulated usage of resources in any VNF instance, cloudlet and link is no greater than its capacity.

Recall that $\mathcal{A}(j)$ is the set of requests admitted by `Algorithm` 3, and $\mathcal{T}(j)$ is the set of requests rejected by `Algorithm` 3 but accepted by the optimal offline algorithm OPT. We have $\frac{OPT-|\mathcal{A}(j)|}{|\mathcal{A}(j)|}\leq 2D_{max}^{\epsilon}(\rho_{max}\log\alpha+L_{max}\cdot C(f_{max})\log\beta+\rho_{max}\cdot b_e\log\gamma)$. Thus, we have $\frac{OPT}{|\mathcal{A}(j)|}\leq 2D_{max}^{\epsilon}(\rho_{max}\log\alpha+L_{max}\cdot C(f_{max})\log\beta+\rho_{max}\cdot b_e\log\gamma)+1=O(\log n)$ when $\alpha=\beta=\gamma=O(n)$.

We finally analyze the time complexity of `Algorithm` 2. The construction of auxiliary graph G'_j takes $O(L_j\cdot|V|+|E|)$ time. It takes $O(|V'_j|^{\frac{1}{\epsilon}}|D_j|^{\frac{2}{\epsilon}})$ time to find an approximate Steiner tree in G'_j for each request r_j, by invoking the approximation algorithm in [4]. `Algorithm` 2 therefore takes time $O((L_j\cdot|V|)^{\frac{1}{\epsilon}}|D_j|^{\frac{2}{\epsilon}})$ for each request admission, where ϵ is a constant with $0<\epsilon\leq 1$.

7 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithms for the admissions of NFV-enabled multicasting requests through experimental simulations. We also investigate the impact of important parameters on the performance of the proposed algorithms.

7.1 Experiment settings

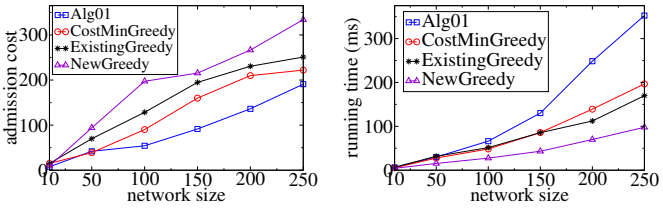
We consider an MEC network $G=(V,E)$ consisting of from 10 to 250 APs (cloudlets). All network topologies are generated by the tool GT-ITM [7]. The computing capacity of each cloudlet is set in the range from $2,000\ MHz$ to $5,000\ MHz$ [12], while the bandwidth capacity of each link varies from $2,000\ Mbps$ to $20,000\ Mbps$ [14]. The number of different types of network functions K is set at 30. The computing resource demand of each network function is set from $300\ MHz$ to $600\ MHz$ randomly, and their processing rate is also randomly drawn from 50 to 100 data packets per millisecond [20]. Recall that the admission cost of an NFV-enabled multicast request consists of three components: the VNF instance processing cost, the VNF instance instantiation cost, and the bandwidth usage cost, where the instantiation cost of a VNF instance in a cloudlet is randomly drawn in the interval $[0.50, 2.0]$, while the processing cost of per packet by a VNF instance is a random value drawn from $[0.01, 0.1]$ [24]. The routing cost per data packet along a link is a value drawn randomly from the interval $[0.01, 0.1]$. To generate request r_j, one AP node in V is randomly selected as its source s_j, and a set of AP nodes in V are randomly chosen as its destination set D_j. The data packet rate is drawn from 2 to 10 packets per

millisecond [15], where each data packet is of size $64KB$. The length of its service function chain is set from 5 to 20, and each network function is randomly drawn from the K types. The value in each figure is the mean of the results out of 30 MEC instances of the same size. The running time of an algorithm is obtained on a machine with 4.0GHz Intel i7 Quad-core CPU and 32GB RAM. Unless otherwise specified, these parameters will be adopted in the default setting.

In the following, we first evaluate the performance of `Algorithm` 1 for the minimum cost problem against three baseline heuristics `CostMinGreedy`, `ExistingGreedy`, and `NewGreedy`. Algorithm `CostMinGreedy` considers network functions in the service function chain one by one, it always chooses the cloudlet with the the minimum admission cost (including the processing cost, instance instantiation cost, and routing cost) for the next network function. Algorithm `ExistingGreedy` considers network functions one by one and tries to admit the request by existing VNF instances with the minimum admission cost as long as there is a VNF instance with sufficient residual processing capacity, while algorithm `NewGreedy` always aims to instantiate a new VNF instance for the request providing sufficient computation resource in a cloudlet. We then evaluate the performance of `Algorithm` 2 against a baseline heuristic `RandomSelect` for the throughput maximization problem, where algorithm `RandomSelect` randomly chooses an unexamined request, and randomly selects available VNF instances for its service function chain. If the request is admitted, the residual capacities of cloudlets, and links in the network are updated. This process continues until all requests are examined. We finally evaluate the performance of `Algorithm` 3 against a benchmark `OnlineLinear` for the online throughput maximization problem, where for each arrived request, algorithm `OnlineLinear` first excludes those VNF instances, cloudlets and links that do not have sufficient residual resources to accommodate the admission of the request from the consideration, it then assigns a cost to each VNF instance, each cloudlet, and each link, and constructs an auxiliary directed acyclic graph for the request. It finally finds a multicast tree rooted at the source node and spanning all destination nodes for the request.

7.2 Performance evaluation of algorithms

We first investigate the performance of `Algorithm` 1 against that of three baseline heuristics `CostMinGreedy`, `ExistingGreedy`, and `NewGreedy`, for the cost minimization problem of a single NFV-enabled request admission, by varying the network size from 10 to 250. Fig. 4 illustrates the admission cost and running time of the four mentioned algorithms. From Fig. 4 (a), we can see that `Algorithm` 1 achieves a much lower admission cost than those three benchmarks. Specifically, `Algorithm` 1 is only 43.1%, 24.0%, and 14.4% of the admission costs of algorithms `NewGreedy`, `ExistingGreedy`, and `CostMinGreedy`, respectively, when the network size is 250. The reason behind is that `Algorithm` 1 jointly considers the placement of VNF instances and data traffic routing for a request admission, it also makes a smart decision between using an existing VNF instance or creating a new VNF instance. Fig. 4 (b) plots the running time curves of the four comparison algorithms. It can be seen that algorithm `NewGreedy` achieves the least running time, as it gives priority to create new VNF instances in cloudlets, while `Algorithm` 1 takes the most running time due to the fact that it strives for finding a multicast tree with the least cost while passing through VNFs in its service function chain at the same time.



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

Fig. 4. Performance of `Algorithm` 1, `CostMinGreedy`, `ExistingGreedy`, and `NewGreedy`, by varying the network size.

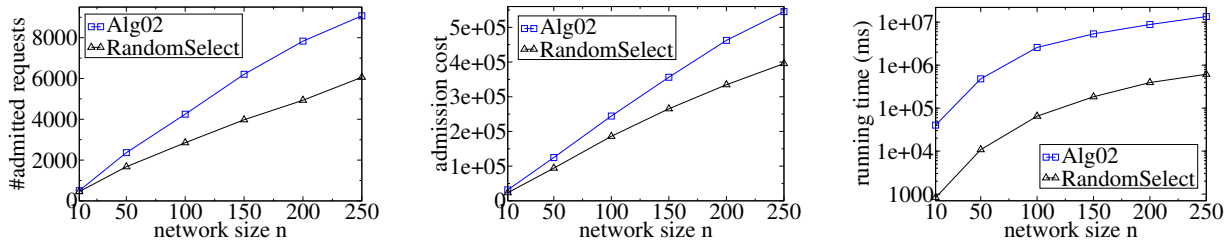
We then study the performance of `Algorithm` 2 against a heuristic `RandomSelect` for the throughput maximization problem, by varying the network size from 10 to 250 for a set of $12,000$ NFV-enabled multicast requests. Fig. 5 plots the performance curves of the two algorithms. It can be seen from Fig. 5 (a) that `Algorithm` 2 outperforms the benchmark `RandomSelect` in all cases, and their performance gap becomes larger and larger with the increase on network size. Specifically, the network throughout achieved by `Algorithm` 2 is 14.7% and 29.6% higher than that by algorithm `RandomSelect` and the admission cost by `Algorithm` 2 is 10.4% and 21.1% higher than that by algorithm `RandomSelect`, when the network size is set at 50 and 250, respectively. Fig. 5 (b) depicts the running times of the two mentioned algorithms. It can be seen that `Algorithm` 2 takes a longer time than that of algorithm `RandomSelect` for finding a more accurate solution.

7.3 Performance evaluation of the online algorithm

We now evaluate the performance of `Algorithm` 3 for the online throughput maximization problem, by varying the network size from 10 to 250 for a sequence of $10,000$ requests. Fig. 6 plots the performance curves of different algorithms, from which we can see that `Algorithm` 3 outperforms the baseline algorithm `OnlineLinear` in all cases, and `Algorithm` 3 can admit 38.6% more requests than that by algorithm `OnlineLinear` when the network size is 200. Fig. 6 (c) shows the running time of the two comparison algorithms.

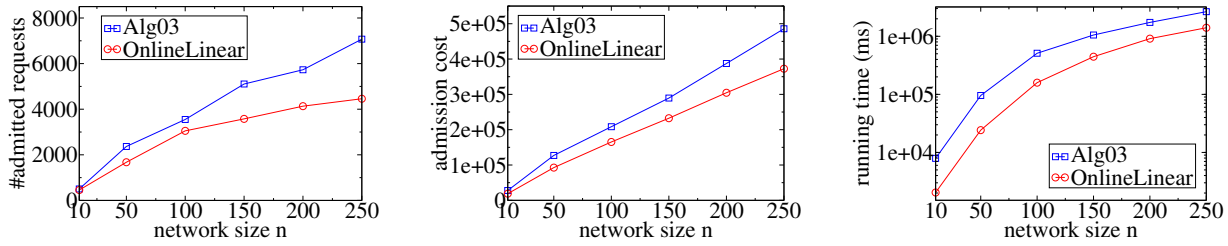
We then investigate the scalability of `Algorithm` 3, by varying the number of cloudlets from 200 to 1,600 for a sequence of $40,000$ NFV-enabled multicast requests while setting the length of request service function chains between 15 and 20. Fig. 7 shows that its performance for a large-scale network is similar to it for moderate-size networks. That is, `Algorithm` 3 achieves a much higher network throughput than that algorithm `OnlineLinear` achieves in all cases, while the increase on the admission cost is minimal.

We finally study the impact of the admission control variables σ_1, σ_2, and σ_3 on the performance of `Algorithm` 3. Fig. 8 plots the performance curves of `Algorithm` 3 with and without adopting the admission control policy, from which it can be seen that less numbers of requests can be admitted if no admission control policy is adopted. When the network size is 100, `Algorithm` 3 admits 40.4% more requests than itself without adopting



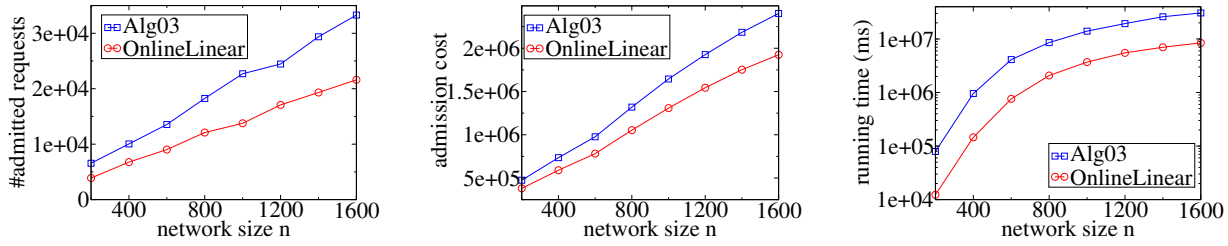
(a) The throughput (b) The admission cost (c) The running time

Fig. 5. Performance of `Algorithm` 2, and `RandomSelect`, by varying the network size from 10 to 250.



(a) The network throughput (b) The admission cost (c) The running time

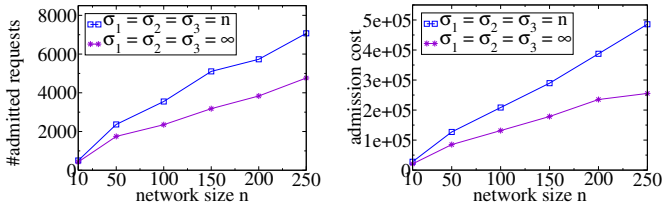
Fig. 6. Performance of `Algorithm` 3 and `OnlineLinear` by varying the network size from 10 to 250.



(a) The network throughput (b) The admission cost (c) The running time

Fig. 7. Scalability performance of `Algorithm` 3 and `OnlineLinear` by varying the network size from 200 to 1,600.

the admission control policy. Furthermore, the performance gap of `Algorithm` 3 with and without the admission control policy becomes larger and larger with the increase in network size. This is due to that in large networks, the size of the destination set of each request can be very large, and the distance between the source node and a destination node of the request can be quite long, thus consuming much more bandwidth resource for routing the data traffic of the request, while `Algorithm` 3 is able to reject those requests with large admission costs, thereby enabling to admit more requests in future to achieve a larger throughput. Fig. 8 (b) shows that the admission costs of `Algorithm` 3 with and without the admission control policy.



(a) The network throughput (b) The admission cost

Fig. 8. Impact of the admission control policy on the performance of `Algorithm` 3.

8 Conclusion

In this paper, we studied NFV-enabled multicast request admissions in a mobile edge cloud network, by formulating three novel optimization problems. We first proposed an approximation algorithm with approximation ratio for the cost minimization problem of a single multicast request admission. We then proposed an efficient algorithm for the throughput maximization problem of admitting a given set of multicast requests by reducing the problem to the cost minimization problem. We also studied the online throughput maximization problem where NFV-enabled multicast requests arrive one by one without the knowledge of future arrivals, for which we devised an online algorithm with a provable competitive ratio. We finally evaluated the performance of the proposed algorithms through experimental simulations. Simulation results demonstrate that the proposed algorithms are very promising, and exhibits better performance compared with their counterparts.

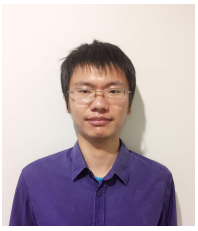
Acknowledgement

We would like to thank the four 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 is supported by the National Natural Science Foundation of China (Grant No. 61802048, 61802047), the fundamental research funds for the central universities in China (Grant No. DUT17RC(3)061, DUT17RC(3)070), and the Xinghai Scholar Program in Dalian University of Technology, China.

References

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie. Mobile edge computing: a survey. *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450 – 465, 2018.

[2] O. Alhussein, P. T. Do, J. Li, Q. Ye, W. Shi, W. Zhuang, X. Shen, X. Li, and J. Rao. Joint VNF placement and multicast traffic routing in 5G core networks. *Proc. of Globecom*, IEEE, 2018.
[3] A. Ceselli, M. Premoli, and S. Secci. Mobile edge cloud network design optimization. *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1818 – 1831, 2017.
[4] M. Charikar, C. Chekuri, T.-Y. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li. Approximation algorithms for directed Steiner problems. *J. Algorithms*, vol. 33, no. 1, pp. 73 – 91, Elsevier, 1998.
[5] M. Chen and Y. Hao. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, IEEE, 2018.
[6] H. Feng, J. Llorca, A. M. Tulino, D. Raz, and A. F. Molish. Approximation algorithms for the nfv service distribution problem. *Proc. of INFOCOM*, IEEE, 2017.
[7] GT-ITM. http://www.cc.gatech.edu/projects/gtitm/, 2019.
[8] T. He, H. Khamfroush, S. Wang, T. La Porta, and S. Stein. It's hard to share: joint service placement and request scheduling in edge clouds with sharable and non-sharable resources. *Proc. of ICDCS*, IEEE, 2018.
[9] M. Jia, J. Cao, and W. Liang. Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 725 – 737, 2017.
[10] M. Jia, W. Liang, M. Huang, Z. Xu, and Y. Ma. Throughput maximization of NFV-enabled unicasting in Software-Defined Networks. *Proc of Globecom'17*, IEEE, Dec, 2017.
[11] M. Jia, W. Liang, Z. Xu, and M. Huang. Cloudlet load balancing in wireless metropolitan area networks. *Proc. of INFOCOM*, IEEE, 2016.
[12] M. Jia, W. Liang, and Z. Xu. QoS-aware task offloading in distributed cloudlets with virtual network function services. *Proc. of MSWiM*, ACM, 2017.
[13] M. Jia, W. Liang, M. Huang, Z. Xu, and Y. Ma. Routing cost minimization and throughput maximization of NFV-enabled unicasting in Software-Defined Networks. *IEEE Transactions on Network and Service Management*, vol.15, no.2, pp. 732 – 745, 2018.
[14] S. Knight. S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *IEEE J. on Selected Areas in Communications*, vol. 29, pp. 1765 – 1775, IEEE, 2011.
[15] Y. Li, L. T. X. Phan, and B. T. Loo. Network functions virtualization with soft real-time guarantees. *Proc. of INFOCOM*, IEEE, 2016.
[16] Y. Ma, W. Liang, and J. Wu. Online NFV-enabled multicasting in mobile edge cloud networks. *Proc of ICDCS'19*, IEEE, July, 2019.
[17] Y. Ma, W. Liang, and Z. Xu. Online revenue maximization in NFV-enabled SDNs. *Proc of ICC'18*, IEEE, May, 2018.
[18] Y. Ma, W. Liang, Z. Xu, and S. Guo. Profit maximization for admitting requests with network function services in distributed clouds. *IEEE Transactions on Parallel and Distributed Systems*, Vol.30, No. 5, pp. 1143 – 1157, 2019.
[19] Y. Mao, C. You, J. Zhang, K, Huang, and K. Letaief. A survey on mobile edge computing: the communication perspective. *IEEE Commun. Surv. Tutor.*, vol. 19, pp. 2322 – 2358, 2017.
[20] J. Martins, J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, and F. Huici. ClickOS and the art of network function virtualization. *Proc. of NSDI 14, USENIX*, 2014.
[21] S. V. Rossem, W. Tavernier, B. Sonkoly, D. Colle, J. Czentye, M. Pickavet, and P. Demeester. Deploying elastic routing capability in an SDN/NFV-enabled environment. *2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pp. 22 – 24, Nov. 2015.
[22] S. Yi, C. Li, and Q. Li. A survey of fog computing: concepts, applications and issues *Proc. of Workshop on Mobile Big Data'15*, pp. 37 – 42, ACM, 2015.
[23] Q. Xia, W. Liang, and W. Xu. Throughput maximization for online request admissions in mobile cloudlets. *Proc. of 38th Annual IEEE Conference on Local Computer Networks* (LCN'13), IEEE, 2013.
[24] Z. Xu, W. Liang, A. Galis, and Y. Ma. Throughput maximization and resource optimization in NFV-enabled networks. *Proc. of ICC'17*, IEEE, 2017.
[25] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis. Approximation and online algorithms for NFV-enabled multicasting in SDNs. *Proc. of 37th Intl Conf on Distributed Computing Systems* (ICDCS'17), IEEE, 2017.
[26] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis. Efficient NFV-enabled multicasting in SDNs. *IEEE Transactions on Communications*, vol. 67, no. 3, pp. 2052 – 2070, 2019.
[27] Z. Xu, W. Liang, M. Jia, M. Huang, and G. Mao. Task offloading with network function services in a mobile edge-cloud network. To appear in *IEEE Transactions on Mobile Computing*, 2018. https://ieeexplore.ieee.org/document/8502709.
[28] Z. Xu, Y. Zhang, W. Liang, Q. Xia, O. Rana, A. Galis, G. Wu, and P. Zhou. NFV-enabled multicasting in mobile edge colouds with resource sharing. *Proc. of ICPP'19*, ACM, August, 2019.
[29] S. Q. Zhang, Q. Zhang, H. Bannazadeh, and A. L. Garcia. Routing algorithms for network function virtualization enabled multicast topology on SDN. *IEEE Transaction on Network and Service Management*, vol. 12, no. 4, pp. 580 – 594, 2015.



Yu Ma received his BSc degree with the first class Honours in Computer Science at the Australian National University in 2015. He is currently a PhD candidate in the Research School of Computer Science at the Australian National University. His research interests include Software Defined Networking, Internet of Things (IoT), and Social Networking.

Weifa Liang (M'99–SM'01) received the PhD degree from the Australian National University in 1998, the ME degree from the University of Science and Technology of China in 1989, and the BSc degree from Wuhan University, China in 1984, all in Computer Science. He is currently a Professor in the Research School of Computer Science at the Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, Mobile Edge Computing (MEC), Network Function Virtualization (NFV), Software-Defined Networking (SDN), design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He is a senior member of the IEEE.

Jie Wu (F'09) is the chair and a Laura H. Carnell professor in the Department of Computer and Information Sciences at Temple University. He is also an Intellectual Ventures endowed visiting chair professor with the National Laboratory for Information Science and Technology, Tsinghua University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He serves on several editorial boards, including the IEEE Transactions on Service Computing and the the Journal of Parallel and Distributed Computing. He was general co-chair/chair of IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, and ACM MobiHoc 2014, as well as program co-chair of IEEE INFOCOM 2011 and CCF CNCC 2013.

Zichuan Xu (M'17) received his PhD degree from the Australian National University in 2016, ME degree and BSc degree from Dalian University of Technology in China in 2011 and 2008, all in Computer Science. He was a Research Associate at Department of Electronic and Electrical Engineering, University College London, UK. He currently is an Associate Professor in School of Software at Dalian University of Technology, China. His research interests include cloud computing, software-defined networking, network function virtualization, wireless sensor networks, algorithmic game theory, and optimization problems.