Hindawi
Mobile Information Systems
Volume 2021, Article ID 5590963, 12 pages
https://doi.org/10.1155/2021/5590963

Research Article

QoS-Based Multicast Routing in Network Function Virtualization-Enabled Software-Defined Mobile Edge Computing Networks

Shimin Sun,[1] **Xinchao Zhang**,[1] **Wentian Huang**,[1] **Aixin Xu**,[1] **Xiaofan Wang**,[1] **and Li Han**[2]

[1]*Tianjin Key Laboratory of Autonomous Intelligent Technology and System, School of Computer Science and Technology, Tiangong University, Tianjin 300387, China*
[2]*National Engineering Laboratory for Computer Virus Prevention and Control Technology, School of Computer Science and Engineering, Tianjin University of Technology, Tianjin 300384, China*

Correspondence should be addressed to Li Han; hanli@tjut.edu.cn

Received 8 February 2021; Revised 27 March 2021; Accepted 2 April 2021; Published 15 April 2021

Academic Editor: Xiaoxian Yang

Mobile Edge Computing (MEC) technology brings the unprecedented computing capacity to the edge of mobile network. It provides the cloud and end user swift high-quality services with seamless integration of mobile network and Internet. With powerful capability, virtualized network functions can be allocated to MEC. In this paper, we study QoS guaranteed multicasting routing with Network Function Virtualization (NFV) in MEC. Specifically, data should pass through a service function chain before reaching destinations along a multicast tree with minimal computational cost and meeting QoS requirements. Furthermore, to overcome the problems of traditional IP multicast and software-defined multicasting approaches, we propose an implementable multicast mechanism that delivers data along multicast tree but uses unicast sessions. We finally evaluate the performance of the proposed mechanism based on experimental simulations. The results show that our mechanism outperforms others reported in the literature.

1. Introduction

In recent years, social networking and personal entertainment have made great success. In many popular scenarios, concurrent subscribers receive data from single or multiple servers, such as remote video conference, online education, and interactive gaming. This paradigm is deemed to be multicasting. Nevertheless, data from most current applications are distributed in a multicast pattern but use unicast sessions in practice. The replicated data put huge traffic pressure on server-side and backbone networks. Meanwhile, with specific requirements, data needs to pass various network functions before reaching destinations, such as Encoder, Network Address Translation (NAT), firewall, Intrusion Detection Systems (IDS), and proxies. The implementation of those functions using a software instance is called Network Function Virtualization (NFV) [1]. As a core functionality of Software-Defined Networking (SDN), NFV is a promising technique to replace the hardware network middle box. It implements network functions in software-based Virtual Machine (VM) instances. Each network function, named Virtual Network Function (VNF), can be deployed on commodity server and further on Mobile Edge Cloud (Computing) (MEC) [2, 3].

With the explosive growth of mobile terminals and application markets, application technology is progressing by leaps [4, 5]. The demands of computational resource on terminals promote the rapid development of MEC technology. MEC has emerged as the light cloud computing distributed at the edge of core network. It realizes data processing and storage capability in the vicinity of terminal devices. Allocating at the edge, MEC significantly reduces the response delay in some situations and enhances the capabilities of mobile users with ever-growing resource demands [6, 7].

A sequenced chain of VNFs comprises a service function chain. A service function chain can be flexibly composed of a series of heterogeneous VNF nodes from different MECs to process massive data flows. Different types of multicast services may require different combination and order of VNFs. To improve the efficiency of resource utilization of MECs, VNFs can be placed on different cloudlets. The implementation of NFV is usually combined with SDN [8], because SDN enables flexible management of VNF placement.

Provisioning multicasting service with a specific network function chain in software-defined MEC is a promising technique and poses many challenges. Firstly, most of the studies on multicasting in MEC or SDN are based on classic IP multicast [9–12], which is not negligible to face those basic issues [13]. The applicability is necessary to be concerned before designing multicasting mechanism. Secondly, comparing with self-contained centralized cloud networks, the computing and storage resource at cloudlets and link capacity is limited to accommodate VM-based VNFs. There should be a consultation mechanism on how to efficiently apply existing or create new VNF instances to minimize the operational cost and meanwhile satisfy the QoS requirements of applications [14]. Further, how to steer data traffic to pass a specific sequence of network functions within a reasonable cost should be deliberated [15].

In this paper, we address the aforementioned challenges by presenting an implementable multicast mechanism in software-defined hybrid MEC network, including cost modelling, data processing mechanism, and tree construction algorithm. The novelties and contributions of this paper are as follows:

(a) We present an implementable multicast mechanism using the SDN technique to overcome the dilemma of traditional IP multicast. Based on our mechanisms, legacy network devices and terminal users are unnecessarily aware of data distribution pattern. The server sends data in unicast sessions (not tunnelling) to each destination. Therefore, the presented mechanisms are not only limited to apply to MEC networks but also suitable for SDN/Internet hybrid networks.

(b) We study the problems of NFV-enabled QoS multicasting in MEC networks, exploring VNF instance placement and resource sharing. We strive for the minimum of accumulative computational cost while meeting the QoS requirements of applications. We further present an NFV-enabled QoS guaranteed multicast tree construction algorithm.

(c) We evaluate the performance of the proposed algorithms through experimental simulations. The results demonstrate that the proposed algorithms are promising and outperform compared algorithms.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the system model, notions and notations, and problem definitions. Section 4 develops an implementable SDN-based NFV-enabled QoS multicasting mechanism. Section 5 elaborates the detailed multicast tree maintenance algorithms. Section 6 evaluates the proposed algorithms empirically, and Section 7 concludes the paper.

2. Related Work

Network traffic engineering has regained much attention due to the opportunities introduced by SDN [9, 16, 17]. Many studies are devoted to finding the optimal placement of VNFs [14, 15, 18, 19]. However, most of them were primarily applied to unicast communication scenarios. For example, Golkarifard et al. [14] presented a dynamic VNF placement algorithm in 5G environment. Yu et al. [19] address the VNF placement to minimize cost and meet the delay demand of each unicast session.

Recently, many studies focus on multicast mechanisms in SDN [9, 20] and NFV-enabled multicasting [10, 21]. For instance, Chiang et al. [9] proposed a dynamic group management approach, Online Branch-aware Steiner Tree (OBST). OBST considers bandwidth utilization, tree scalability, and rerouting overhead. Alhussein et al. [21] presented a multicast service orchestration framework to deal with joint routing and VNF placement problem, while maximizing the throughput of the physical substrate and minimizing the provisioning cost. He et al. [10] took into account QoS constraints to build a reliable multicast tree with recovery nodes and passing a service chain. The overhead could be large for data recovery using response messages if the network is unstable. However, those proposals did not take into account the computation capacity of edge cloudlets, as well as the applicability in real hybrid Internet/MEC networks.

There are rare studies on multicasting in NFV-enabled MEC networks [11, 12]. Ma et al. [11] focused on the maximization of throughput. To minimize the cost, they presented an approximation algorithm and an online throughput maximization algorithm. Ren et al. [12] proposed a delay-aware NFV-enabled multicasting mechanism. They presented an approximation algorithm and a heuristic algorithm with and without delay constraint. Although those NFV-enabled multicasting algorithms aimed to minimize resource expense, they did not involve the combined constraints of QoS demands and computing capacity of cloudlets. Most of the above researches did not consider the basic implementation method of data distribution. Thus, it is an urgent problem to design an implementable multicasting approach before putting into effect the aforementioned algorithms.

For the deployment of multicasting in SDN, most of the studies endeavoured to implement or amend IP multicast mechanisms. That is, data dissemination is still based on multicast IP addresses and inevitably suffers from most of IP multicast problems. For instance, MultiFlow [22] developed an approach to parse IGMP messages by OpenFlow switches (OFSs), in order to perform group member management. OFM [23] developed multicast services from a clean-slate perspective. The limitation of MultiFlow and OFM is that OFSs need to have the capability of parsing group joining and departure messages. Locality-Aware Multicast Approach (LAMA) [24] reduced the computation cost of multicast tree construction by designing an election algorithm of Rendezvous Point (RP). The algorithm calculates the distances from RP to all sources and generates a minimum tree. The problem is that RP election algorithm requires additional message exchanging, which causes packet overhead and challenges the capacity of flow table. SDM [25] attempted to slice multicast tree by designing SDM domain consisting of pure SDN devices to distribute traffic using multicast address. A flow table entry was required for each group. Specific messages were elaborated to realize the functions of network entities, such as NL-SDM and Virtual Peer Instance. In brief, all those proposals use IP multicast mechanism or design new protocols to manage groups and parse multicast traffic.

3. Preliminaries

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

3.1. System Model. We consider an MEC network modelled by an undirected graph $G = (V, E)$ with a set V of OpenFlow-enabled access points (APs), OFSs, cloudlets, and a set E of links in the network. Each cloudlet is colocated with an AP node (or an OFS) v $(v \in V)$ connected via a high-speed optical cable. Communication delay between cloudlet and directly connected APs is negligible. Cloudlets have the computing and storage capacity to implement various VNFs based on VMs. There is at least one SDN controller taking the role of the brain of the network. SDN controller is responsible for network traffic steering, multicast group management, and placement of VNFs.

In Figure 1, we present an illustrative example of an NFV-enabled MEC network, in which some of the APs/OFSs are attached to cloudlets, while the rest nodes are not. VNF-enabled APs/OFSs provide VNF support on behalf of multicasting services. SDN controller is responsible for the management of APs/OFSs, while the source node produces streaming and provides the information of group members. All those entities connect to the Internet. All the links could be direct physical links or virtual links.

3.2. Cost Model of NFV-Enabled QoS Multicast Routing. For real-time service, QoS multicast routing is essential to provide high-quality service. In this paper, our study achieves the objective that is constructing multicast tree with NFV support and meeting QoS constraints. For QoS guarantee, we mainly consider the QoS service in two aspects: delay constraints and bandwidth constraints.

Delay constraints: it is assumed that P_k is a set of unicast routes from source S_k to destination D_k in the multicast graph. Path $p_m \in P_k$ denotes the path from S_k to a destination. The maximal delay is incurred by the largest end-to-end delay of p_m. Then, we can define the transmission delay d_k^t using

$$d_k^t = \max_{p_m \in P_k} \sum_{e \in p_m} d(u, v). \tag{1}$$

Without loss of generality, it is assumed that the processing delay of path p, denoted byd_k^p, is proportional to the data rate of the flow. The processing delay may be different, since the computational consumption of each VNF is disparate. Besides, the processing delay of VNF and legacy routers is also discrepant. Generally, the functionality implemented by hardware is much faster than implemented by software. Therefore, d_k^p can be deduced by the sum of processing delay of each VNF and legacy routers, denoted by

$$d_k^p = \sum_{f_{k,i} \in SC_k} \alpha_j \cdot r_k + \sum_{e \in E, e \notin SC_k} \beta \cdot r_k, \tag{2}$$

where α_j and β are given proportional factors for each VNF processing delay and legacy network device processing delay, respectively.

Finally, we obtain the maximum delay d_k of the multicast tree, depicted in equation (3). To meet the delay demand of service k, d_k should be lower than delay threshold D_k; that is, $d_k < D_k$.

$$d_k = d_k^t + d_k^p. \tag{3}$$

Bandwidth constraints: denote by b_p the available bandwidth of a path $p_m \in P_k$ from source S_k to destination D_k. Denote by b_m the minimum bandwidth of a multicast tree. Then, b_p and b_m can be calculated by equations (4) and (5)

$$b_p = \min_{e \in P_m} e(u, v), \tag{4}$$

$$b_m = \min_{p_m \in P_k} b_p. \tag{5}$$

Demand bandwidth B_k should be larger than b_m; that is, $b_m > B_k$.

Instantiating a new VNF instance consumes both computing and storage resources. The cost of setting up a new VNF is denoted by $c_{\text{ins}}(f_{k,i}, v)$. Unit data processing cost using an existing instance of function f_k in cloudlet v is indicated by$c_{\text{proc}}(f_{k,i}, v)$. Then, the processing cost of a service with data rate r_k is $r_k \cdot c_{\text{proc}}(f_{k,i}, v)$. Notice that, the placement of VNFs on different cloudlets may result in different cost. Let n_{ins} be the number of newly created instances for $f_{k,i}$ in cloudlet v, while n_{ext} is the number of existing instances to process data on behalf of $f_{k,i}$.

The total operational cost for k-th multicast service is represented by the sum of the cost of setting up new VNFs and the processing cost of k-th service with data rate r_k.

The cost of setting up new VNFs for the multicast service is$\sum_{f_{k,i} \in SC_k} n_{\text{ins}} \cdot c_{\text{ins}}(f_{k,i}, v)$. The processing cost of all VNFs is$\sum_{f_{k,i} \in SC_k} (n_{\text{ins}} + n_{\text{ext}}) \cdot r_k \cdot c_{\text{proc}}(f_{k,i}, v)$.

Then, the total cost to implement and run all VNFs can be specified as

$$C_k = \sum_{f_{k,i} \in SC_k} \left(n_{ins} \cdot c_{ins}\left(f_{k,i}, v\right) + \left(n_{ins} + n_{ext}\right) \cdot r_k \cdot c_{proc}\left(f_{k,i}, v\right) \right). \tag{6}$$

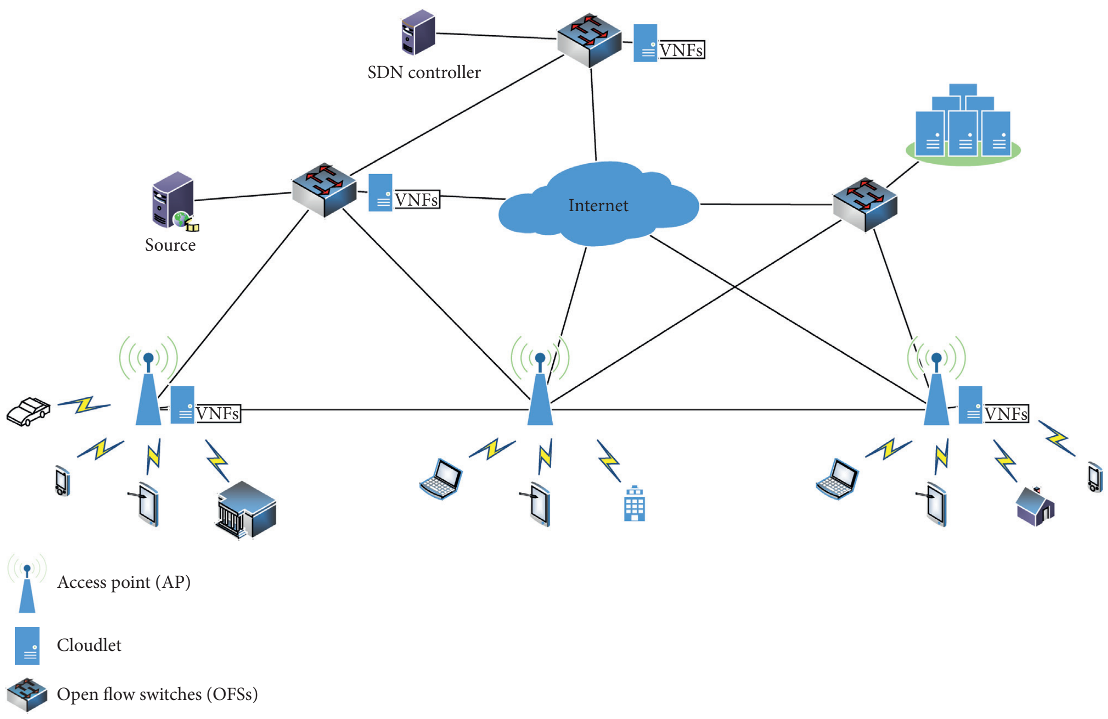

Figure 1: An illustrative example of an MEC network with VNFs and SDN.

For a service function chain $SC_k = (f_{k,1}, f_{k,2}, \ldots, f_{k,L_k})$ with m available cloudlets, the total cost can be represented by a matrix, as shown in equation (7). Where the set (CL_1, $CL_2, \ldots, CL_m$) is the cloudlets that have sufficient resource to implement VNF instances, the element is the operational cost of a VNF implementing on a cloudlet. Then, the computational cost of each cloudlet is the summation of each row, which has an upper bound threshold depending on the computation capability of each cloudlet.

$$C_k = \begin{array}{c} \\ CL_1 \\ CL_2 \\ \cdots \\ CL_m \end{array} \begin{array}{c} \begin{array}{cccc} f_{k,1} & f_{k,2} & \cdots & f_{k,L_k} \end{array} \\ \begin{pmatrix} a_1 & a_2 & \cdots & a_{L_k} \\ b_1 & b_2 & \cdots & b_{L_k} \\ \cdots & \cdots & \cdots & \cdots \\ m_1 & m_2 & \cdots & m_{L_k} \end{pmatrix} \end{array} \quad (7)$$

For clarity, the symbols that used in this paper are summarized in Table 1.

3.3. Problem Definitions

Problem 1. Most of the studies focus on the functional design of multicast routing and ignore the implementation of basic functionalities, in most cases, IP multicast. Because IP multicast is far from being widely deployed in current Internet architecture, puzzled by router capability, scalability, security, economic efficiency, and so on. Therefore, it is difficult to be implemented in large-scale networks.

The first challenge is to design an implementable multicast approach that can be facilely applied to real IP networks. In particular, with the development of new-type network technology, such as SDN and MEC, there emerge new thoughts and innovation opportunities to be put forward to novel multicasting algorithms. Therefore, we attempt to design an implementable multicast mechanism considering NFV and QoS requirements, which is compatible to SDN, MEC, and various Internet hybrid networks.

Problem 2. NFV-enable multicast routing with QoS guarantee problem is NP-hard.

Proof. NFV-enable multicast routing is a special case of classic QoS multicast routing without NFV support. It builds a multicast tree from source to destinations passing by a series of VNFs. For simplicity, we split the process of tree construction into two steps. First, the paths from source to the last VNF of a service chain are sorted by the cost of equation (6). Then, we construct the minimal multicast tree from each last VNF to destinations with QoS guarantees. Finally, the optimal multicast tree is generated with NFV-enabled and meeting QoS requirements. According to literature [11, 13, 26], multicast tree construction problem can be reduced to a Steiner tree problem. □

Problem 3. NFV-enable multicast routing problem in MEC network is to route the traffic to each destination bypassing either existing or newly created VNF instances. Such that, it requires that the operational cost is minimum, while satisfying the QoS requirements D_k and B_k, as well as the capacity constraint of each cloudlet. If the computational resource of each cloudlet is sufficient to implement all required VNFs, the problem is simplified into a QoS

Table 1: Symbols.

Symbols	Meaning
$G = (V, E)$	A software-defined Mobile Edge Cloud network with a set V of OpenFlow switches (OFSs) or APs and a set E of links
v and e	An OFS node or an AP node v $(v \in V)$ and a link e $(e \in E)$
C_v	The available computing capacity of the cloudlet attached to node v
$M_k = (S_k, D_k, SC_k, b_k, d_k)$	The k-th multicast service with source node S_k, the set of destinations D_k, the service chain SC_k that consists of a sequence of VNFs, bandwidth requirement b_k, and delay limitation d_k
L_k	The number of VNFs in SC_k
$SC_k = (f_{k,1}, f_{k,2}, \ldots, f_{k,L_k})$	A service chain for k-th multicast service consisting of L_k-th functions $(f_{k,1}, f_{k,2}, \cdots, f_{k,L_k})$
$c_{f,\ m}$	Computational cost to implement a VNF f on VM m
$d_{u,v}$	Delay of link $e(u,\ v)$
$b_{u,v}$	Available bandwidth of link $e(u,\ v)$
r_k	Data rate of k-th multicast service
P_k	A set of unicast route from source S_k to each destination from D_k
p_m	A route from source S_k to a destination
d_k^t and d_k^p	Transmission delay and processing delay, respectively
d_k	The maximum delay of the multicast tree
b_p and b_m	The minimum bandwidth of a path and a multicast tree, respectively
D_k and B_k	QoS requirement of delay and bandwidth, respectively
$c_{\text{ins}}(f_{k,i}, v)$	The cost of instantiating a new VNF instance i on node v for k-th multicast service
$c_{\text{proc}}(f_{k,i}, v)$	The cost of using an existing VNF instance i on node v for k-th multicast service
C_k	The total operational cost for multicast service k
$CL = (CL_1,\ CL_2,\ \ldots,\ CL_m)$	A set of cloudlets that have sufficient resource to implement VNF instances
$VNFL = (VNF_1,\ VNF_2,\ \ldots,\ VNF_n)$	VNF that exists or can be created for a multicast service

constraints multicast routing problem that is to find a QoS guaranteed minimal spanning tree, while the VNFs can be placed on any cloudlet on the paths from source to each destination.

4. Design of Implementable SDN-Based Multicasting Routing Architecture

4.1. System Architecture. The proposed mechanism is built on top of overlay SDN architecture with three layered planes. Functionalities are designed as application modules in application plane, routing decision distributed by SDN controller in control plane [27, 28] and executed by OFSs in data plane. Unlike other studies, we involve servers as a participant for multicast group management. Then, it is much easier to manage group members from the point of both source and SDN controller. The roles and responsibilities of the main entities are described as follows. Parts of the content of this section have been presented in our previous work [29].

Receiver. End host that initiates service request in a unicast session to a source node. Commonly, it also receives data from this unicast session. Thus, it is transparent to multicast data distribution mechanism.

Source. Server that provides data distribution service. In this proposal, it acts not only as a resource server but also as a participant for group management. It is responsible for maintaining the unicast session information with each receiver and meanwhile keeping the information of receivers updated with SDN controller.

SDN Controller. The brain of the whole system. It allocates network resource, manages multicast group, constructs and maintains multicast tree, distributes decisions where to place VNFs, and so on.

OpenFlow Switch and OpenFlow-enabled AP. In the virtualized overlay network, OFS is the network entities managed and controlled by SDN controller. They connect to cloudlets, Internet, and each other. OFS is the actor of data forwarding strictly following the rules from controllers. More importantly, it monitors the status of direct links and counts the statistics of processed packets.

Legacy router. Except OFSs, there are numerous legacy routers in the current dominant Internet. They are lacking programmability and out of the control of controller. They are used to connect OFSs. When building virtualized overlay network, legacy routers are ignored since they do not participate in any multicast-related operation.

4.2. Processing of Multicast Service. The procedure of multicast service is illustrated in Figure 2. It is assumed that source node S accesses the Internet by an OFS and provides only a single streaming service. Before initiating the steaming, source needs to complete the authentication to controller. A multicast address is assigned to S to indicate the multicast service.

Initially, S does not stream and waits for service request. After that, host R_1 issues the first service request to S. S replies a response message to R_1 encapsulating an authentication request. The authentication can be completed based

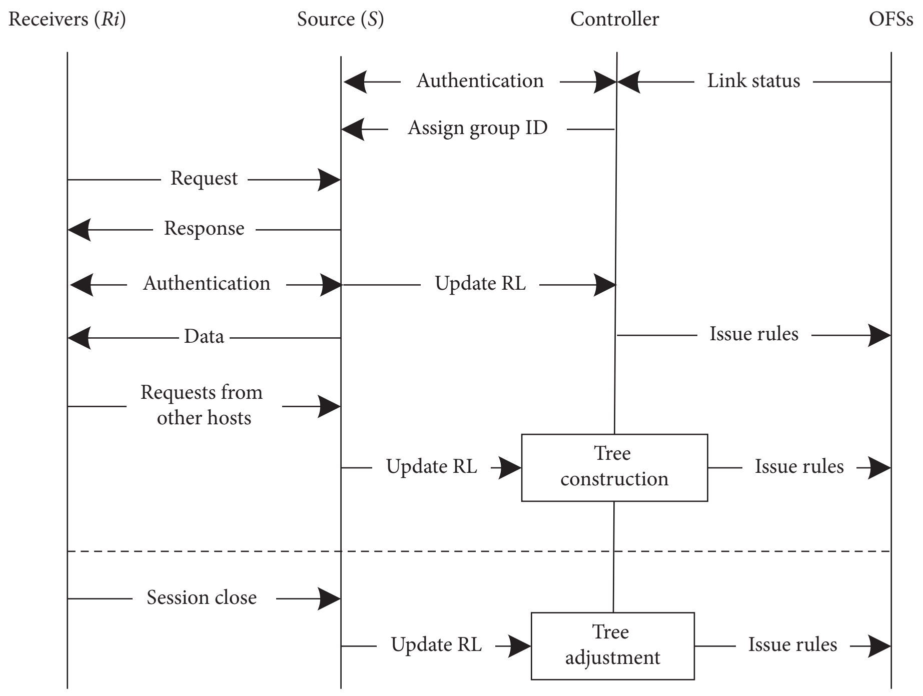

Figure 2: Brief illustration of functional flowchart.

on frequently used password-based authentication, which is out of the scope of this paper and will not be further extended. After achieving authentication, S creates a unicast session with R_1. It fetches the session information from the exchanged packets and stores it to the receiver list. Since R_1 is the first receiver of the service, S issues data to R_1 through the unicast session along the default routing path. OFSs on the path may report the arrival of new packets to the controller and request for admission of access.

After that, other hosts may request the service. S maintains the receiver list dynamically and updates synchronously with the controller. As decision-maker, SDN controller constructs and adjusts the multicasts tree along with the joining and departure of receivers. It issues rules to the corresponding OFSs on behalf of data dissemination for each receiver. Data manipulation mechanism will be presented in Section 5.

When receiver leaving, the corresponding session will be closed. The information of the receiver is removed from the receiver list. Tree adjustment algorithm initiates if necessary.

5. Joint Unicast and Multicast Routing Algorithm for NFV-Enabled QoS Guaranteed Network

5.1. Data Manipulation to Realize Packet Distribution on Multicast Tree. For a better understanding of the proposed mechanism, we first introduce how to steer data delivered along constructed multicast tree ignoring VNFs. To illustrate the basic operation, we present a brief example in Figure 3. There, triangle icons denote the server; circular icons and grey squares with sequence numbers inside indicate OFSs and receivers, respectively.

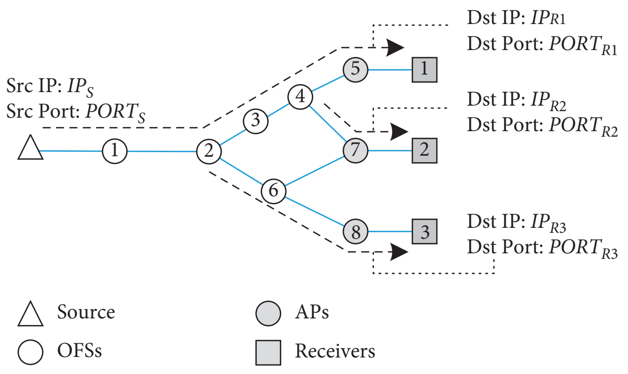

Figure 3: Basic operations of multicast data distribution.

It is assumed that R_1 is the first host that sends a request to the source. The server sends data packets to R_1 directly in unicast session. Afterwards, R_2 requests the same source. Now, a branch is necessary to distribute data to R_2 with minimum cost. Therefore, an OFS is selected as the operation node for data bifurcation. Following the tree construction mechanism presented in Section 5.2, the fork node selected is OFS_4 in common situation. OFS_4 replicates the flow and replaces the destination of the replicated flow with the IP/Port pair of R_2, which is from $<IP_{R1}, PORT_{R1}>$ to $<IP_{R2}, PORT_{R2}>$. Similarly, OFS_2 is selected as the operation node for R_3, which transfers duplicated packets from $<IP_{R1}, PORT_{R1}>$ to $<IP_{R3}, PORT_{R3}>$.

In current networks, there are numerous legacy routers, and some OFSs may be capacity limited to be an operation node. In fact, the operation node is not necessary to be a physically branching node. Wrapping route is allowed to find a usable operation node. For example in Figure 3, the desired operation node for R_3 is OFS_2. However, if OFS_2 is unable to serve as the operation node, other OFSs, such as OFS_1, OFS_3, OFS_7, or even OFS_4, could undertake the task of data manipulation for R_3.

In this section, we described how data are distributed along a multicast tree but in unicast sessions, without involving NFV capability, QoS, and MEC capacity constraints. This is the implementable method and can be extensively applied to NFV-enabled SDN MEC networks.

5.2. NFV-Enabled QoS Guaranteed Multicast Tree Construction Algorithms. In this section, we elaborate how to solve VNF placement problems during multicast tree construction. Based on the introduction of NFV and notions in Section 3.2, we consider the cost matrix defined in equation (7) and aim to calculate the computational cost of each possible route from source to a last VNF in a specific sequence of service chain, that is, with a set $VNFL = (VNF_1, VNF_2, \ldots, VNF_n)$ of VNFs and a service chain $SC_k = (f_{k,1}, f_{k,2}, \ldots, f_{k,L_k})$, to find all the paths from source to the node of the last service function f_{k,L_k} passing the service chain of SC_k and to calculate the cost using equation (6).

After sorting out the paths from the source to the last function nodes, we then construct trees from each last function node to the set of destinations, which is the Steiner tree problem with NP-hard. How to build a Steiner tree will not be extended in this paper; readers can refer to [30, 31] for more information. We calculate the minimal spanning trees meeting QoS constraints. Subsequently, a set of multicast trees are generated from each function node of f_{k,L_k} to the set of destinations.

Finally, we get the set of paths from the source to the node of the last service function f_{k,L_k} and a set of multicast trees from each last function node to the destination set. Based on the information, we can find the optimal multicast tree from the source node to the destination set with minimal cost and meeting the QoS requirements while passing required VNFs. Notice that, the cost of links from the last function node to the set of destinations is ignored, because we hold the opinion that the cost of deploying VNFs is much larger than that of utilizing a link in the case of QoS guaranteed. Algorithm 1 presents the pseudocode of the NFV-enabled QoS guaranteed multicast tree construction algorithm.

6. Methodology, Implementation, and Evaluation

It is a great challenge to design an SDN/NFV/MEC-enabled multicast routing mechanism. Likewise, it is hard to implement the proposal in a single simulation environment. Therefore, to experiment our proposal, we first set up an SDN environment in Mininet [32], and we use NS2 simulator to build the same network as the comparison. In this section, we evaluate the performance of the proposed mechanism and investigate the impact of the main parameters.

We denote our proposal by SDUM. Many of the existing multicast approaches explicitly or implicitly apply or improve IP multicast to suit specific network scenarios. However, the performance of basic data distribution efficiency is commonly no better than native IP multicast. In this paper, we chose SDM [25] and PIM-SM [30] as the comparison. SDM is a pure SDN-based multicast approach, while PIM-SM is a classical IP multicast routing protocol suitable for multicast in sparse networks. Both of them are in line with the cases of our proposal. Multicast tree rooted from the source to destinations was constructed according to Steiner tree algorithm. Thus, the tree topology was the same for all three methods.

6.1. Simulation Setup. We implemented SDUM on top of the Ryu controller [33] based on OpenFlow protocol version 1.3. The network topology was constructed using Mininet version 2.2.2. Due to functional limitations, it is difficult to implement the PIM-SM protocol in Mininet. Therefore, we deployed the same network with the exact same link status using NS2, which was able to simulate IP multicast protocols. The testbed was allocated in a desktop PC with Ubuntu Desktop 16.04. Each simulation time was set to 60 seconds for each scenario.

We built the networks consisting of 50 to 250 OFSs. Network topology was generated using the tool GT-ITM [34]. Source node was attached to an OFS and generated 1 Mbps constant UDP streaming.

6.2. Evaluation and Analysis. Since the examined approaches were all based on the Steiner tree algorithm, they conducted actually the same multicast tree. Therefore, it is meaningless to compare tree construction efficiency, data delivery latency, or bandwidth consumption with each other. Instead, it is significant to evaluate the number of control messages, signalling overhead ratios, and utilization of flow table entries or routing table entries installed to OFSs or routers, respectively, on the multicast tree to enable data transmission. We also evaluated the impact of the number of OFSs in a hybrid network. Finally, we present a qualitative comparison of typical multicast approaches.

6.2.1. Impact of Network Scale. In the experiments, we increased the network scale from 50 nodes to 250 nodes with 30 randomly selected destinations. The results (Figures 4(a)–4(c)) show that the total cost on three aspects significantly increased with the growing of network size. It is mainly because that multicast tree grows up along with more links and exchanging control messages when the network is scaling up.

The number of control messages reveals the load of protocols or approaches imposed on the network. For PIM-SM, control messages are exchanged among routers to build and maintain multicast tree, such as IGMP and RP election messages. For SDM and SDUM, control messages are mainly OpenFlow messages exchanged among OFSs and controller, including *PacketOut* messages for rule installation, *PacketIn* messages for rule query, and so on. Except OpenFlow messages, SDM generates control messages for domain

```
    Input: $G = (V, E)$, $M_k$, $D_k$, $B_k$, $c_{f,m}$, $SC_k$, $CL$
    Output: A QoS guaranteed multicast tree with minimal cost
(1) Calculate the cost $C_{kj}$ of path $j$ from source $S_i$ to VNF node $v (v \in CL)$ that implementing $f_{k,L_k}$;
(2) Get a set of path costs $C_k = (C_{k1}, C_{k2}, \ldots, C_{kn})$;
(3) for $v \in CL$, do
(4) Construct a Steiner tree from each $v$ to destination set;
(5) With QoS requirement constraints, generate a set of multicast trees $T$;
(6) end
(7) Select the multicast tree $T_{opt}$ with the minimum cost of $C_k$ of $v$, which meets the QoS requirements
(8) Return $T_{opt}$
```

Algorithm 1: NFV-enabled QoS guaranteed multicast tree construction

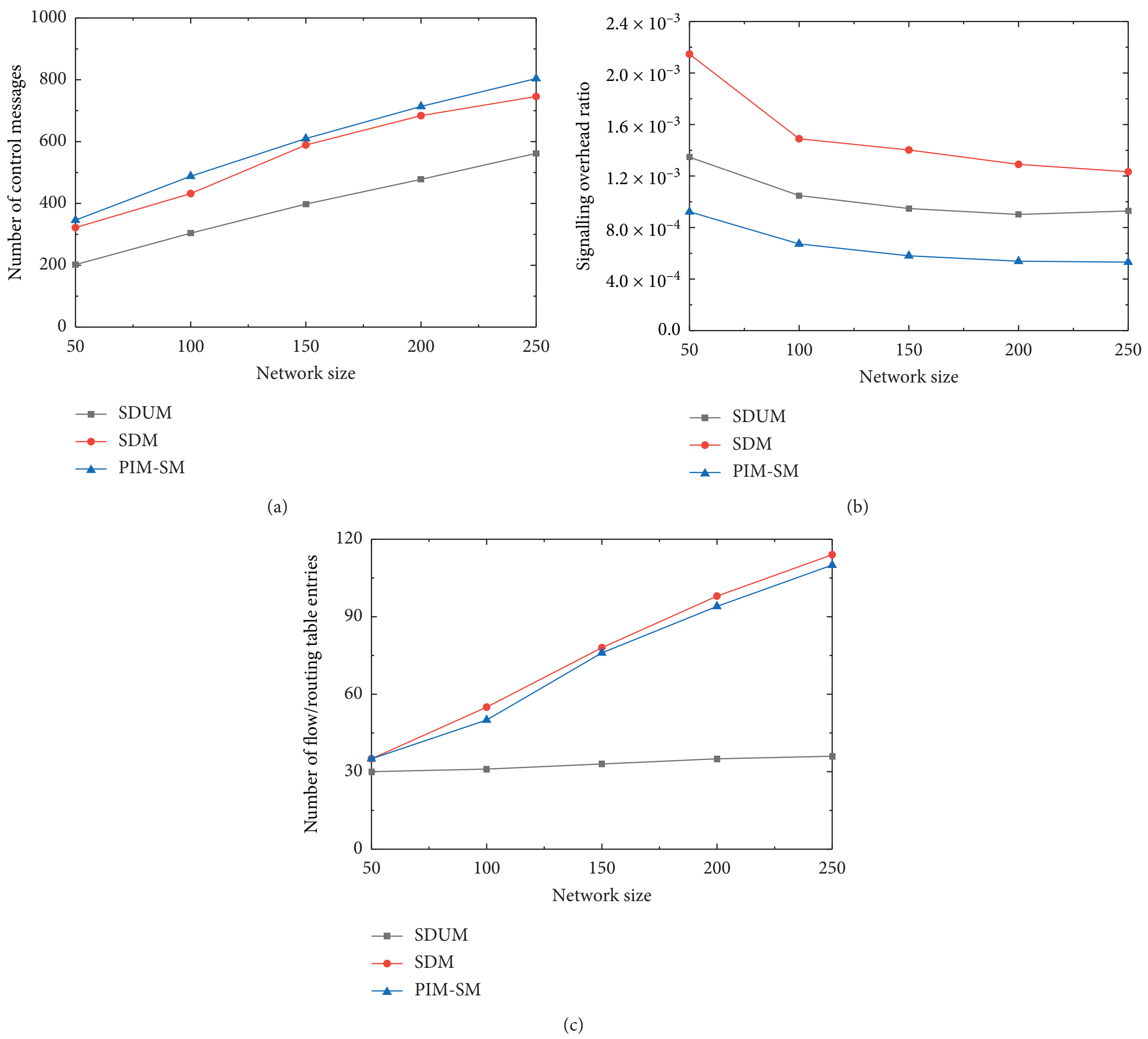

Figure 4: The impact of network size on control overhead. (a) The number of control messages. (b) Signalling overhead ratio. (c) The number of flow/routing entries.

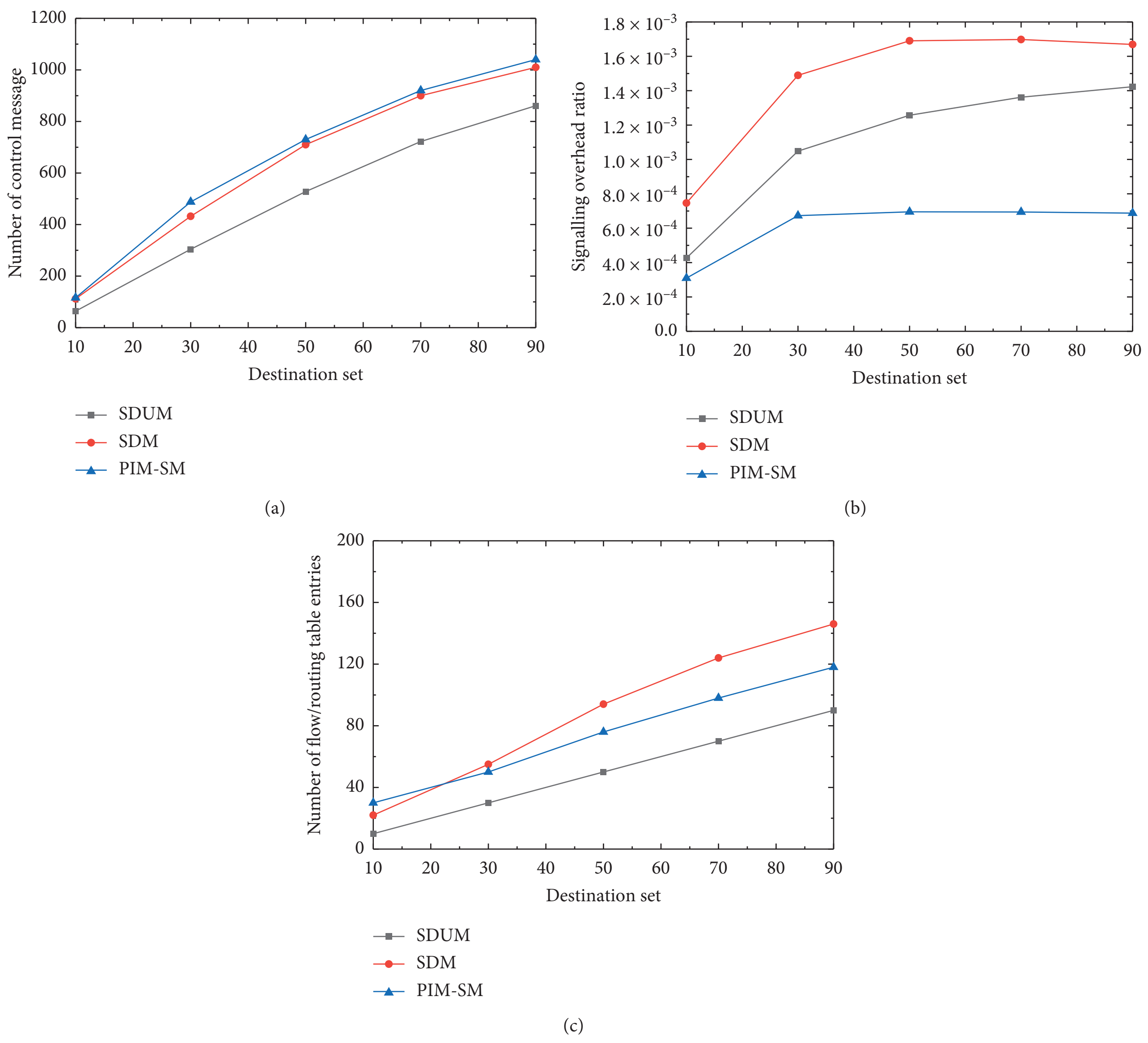

FIGURE 5: The impact of group size on control overhead. (a) The number of control messages. (b) Signalling overhead ratio. (c) The number of flow/routing entries.

management, virtual peer operation, host joining and leaving, and so on. From the results in Figure 4(a), we can conclude that SDUM injects much less control messages than SDM and PIM-SM.

Signalling overhead ratio denotes the bitrate proportion of multicast-related control messages to the total data rate per link. We emulated a constant 1 Mbps UDP traffic as the streaming. Depending on the basic format of control messages, PIM-SM control messages are around 70 bytes while control messages in SDN are around 200 bytes. The number of the control messages may not reveal the real signalling overhead due to different packet size. We can see that PIM-SM is the overwhelming approach among them, as depicted in Figure 4(b).

It is also necessary to measure the utilization of flow (routing) table entries, because it is the main reason that limits the globe deployment of IP multicast. Similarly, the volume of flow table is also restricted by the price of memory (TCAM). As a contrast, PIM-SM requires one table entry for each group on every router that may process multicast traffic. For SDM and SDUM, it is necessary to install rules to flow table to realize packet duplication, address translation, and data forwarding. The results in Figure 4(c) show that SDM performs worse since the occupation of flow table increase sharply. SDUM incurs a horizontal line, which is because the number of rules installed to OFSs depends on the number of destinations (30), while SDUM and PIM-SM are sensitive to the size of the network and the multicast tree.

6.2.2. Impact of Group Size. We further measured the impact of group size. We set the network size to 200 nodes, with the number of destinations switched from 10 to 90. A constant 1 Mbps UDP traffic was injected as the streaming to the

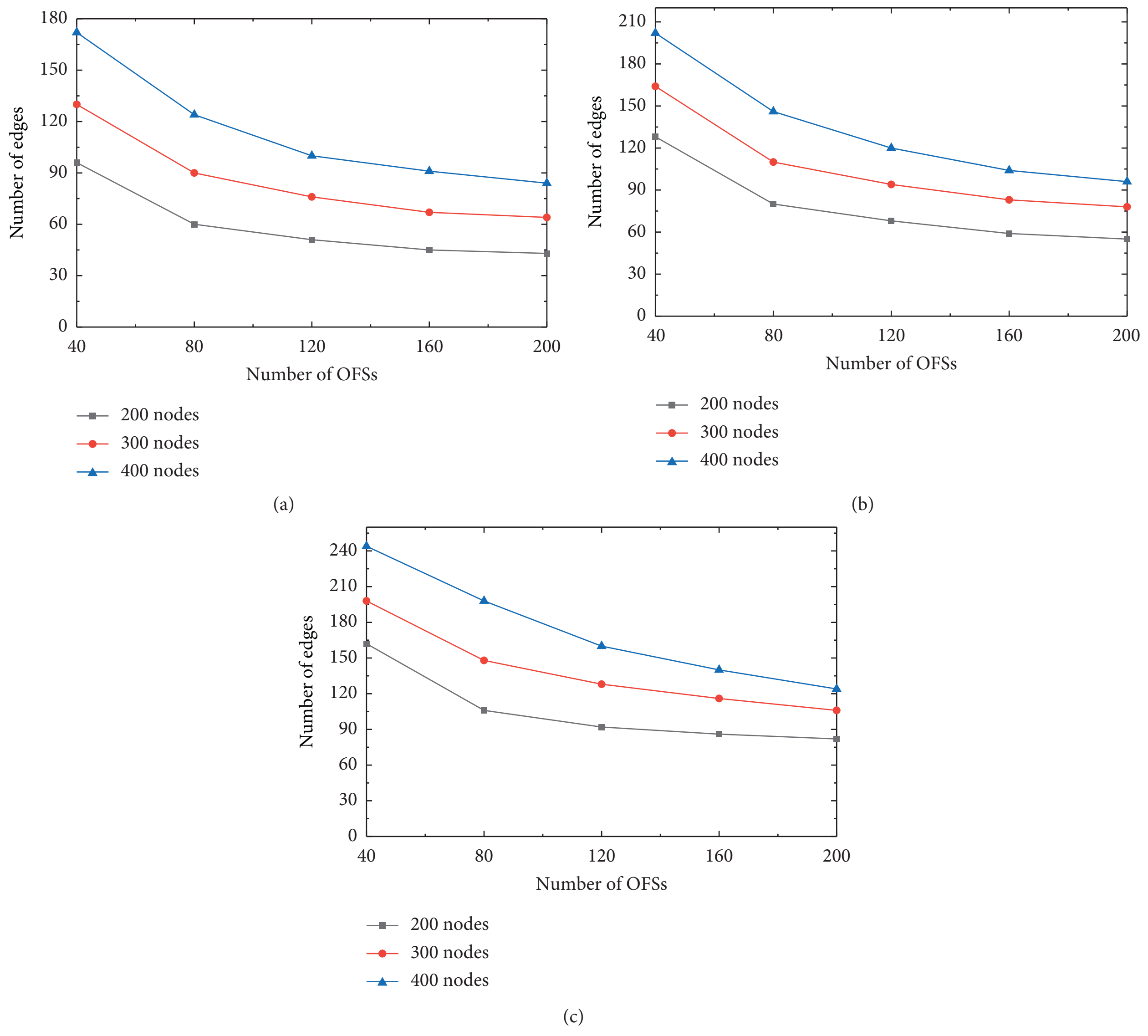

Figure 6: The impact of varying number of OpenFlow switches (OFSs) in hybrid network.

network by source node. Although the number of control messages increases for all three methods, the performance of SDM is close to IP multicast while SDUM consumes much less control messages (Figure 5(a)). For signalling overhead (Figure 5(b)), the number of edges of the multicast tree is also rising with the growing of destinations. Therefore, the signalling overhead per link becomes gently with the variation of destinations. In these experiments, the performance of PIM-SM is overwhelmed, because the packet size in PIM-SM is much smaller than OpenFlow messages. For utilization of flow (routing) table entries (Figure 5(c)), SDUM performs outstandingly than both others. The number of flow table entries grows linearly correlated to the number of receivers. For SDM, the number of rules installed greatly relies on network topology and destination location. It is necessary to install rules not only on access OFS of each receiver but also on specific core OFSs. For PIM-SM, only the legacy routers on the multicast tree are required to insert an entry to the routing table, which is necessary for parsing multicast addresses.

From the above experiments, we can conclude that SDM shows a worse scalability performance with the growing of network size and the increasing number of destinations. The performance of PIM-SM is more correlated to the size of multicast tree, while SDUM has strong correlation with the number of destinations. Therefore, PIM-SM is suitable for sparse network with limited groups, while SDUM is better to be deployed for large networks with a small quantity of receivers in each group.

6.2.3. Impact of the Number of OFSs in Hybrid Network. As we mentioned, our design is suitable for hybrid networks. The number of OFSs in the network affects the performance of multicast tree construction. In the simulation, we assumed that OFSs distributed in the network randomly. Legacy routers connected with OFSs and formed the network together. Therefore, instead of counting the edges of the virtualized network, we should count the physical edges

Table 2: Qualitative comparison.

	PIM-SM	SDM	SDUM
Incremental implementation	All routers in the network need to install multicast protocol	No legacy router allowed in SDM domain	Support hybrid networks; only operation nodes are OpenFlow demand
Scalability	Low (distributed routers of multiple ISPs result in difficulty to use same multicast mechanism)	Low (in the current evolution network, it is hard to build a pure large-scale SDN)	High (virtualized network on top of hybrid network is possible)
ISP control	Low (distributed legacy routers are hard to be managed)	High (centralized control of OFSs in SDM domain)	High (centralized control of fork OFSs distributed in the network)
Access control	Some specific protocols are necessary, for example, IGMP-AC and SIGMP	No concern	Source, receiver, and on-the-fly traffic management and control

of the multicast tree including legacy router-connected links. We present the experiments with different number of OFSs from 40 to 200 in networks with a total of 200, 300, and 400 nodes. We counted the number of edges that formed the multicast tree. For the simplicity of the simulation, we assumed that a single OFS at most attaches to one receiver.

According to the results demonstrated in Figure 6, it is obvious that the number of edges of a multicast tree has a strong correlation with the number of OFSs in hybrid network. With the raising of network size, the number of edges also increases. For a fixed network size, with the increasing of the number of OFSs, the number of edges decreases. That is because, in our mechanism, one receiver corresponds to a specific operation node (OFS). We can conclude that our mechanism can generate minimal multicast tree if most of the network devices are OFSs.

6.2.4. Qualitative Comparison. It is also meaningful to make a qualitative comparison about the classic multicasting approaches. As shown in Table 2, the main concern is the incremental implementation, which is essential whether it can be popularized or not. To implement multicasting service, PIM-SM requires all routers support multicasting function, while SDM requires that all devices are OpenFlow enabled in SDM domain. As a comparison, SDUM requires that only the operation nodes are OpenFlow enabled without any other function demands. For scalability consideration, we need concern about how many devices are required to support specific functions. To implement PIM-SM, all routers need to install the multicast protocol. For SDM, pure SDN domain is required and thus inadequate to be applied to hybrid networks. SDUM is suitable for hybrid networks with distributed OFSs in the network and implementable in large complex networks. For ISP control and access control, it is hard to manage group members using IP multicasting protocols due to the distribution management feature of group joining and leaving. However, attribute to the centralized management manner of SDUM, it can easily achieve source, receiver, and on-the-fly data traffic management and control.

7. Conclusion

In this paper, we presented an implementable multicast routing mechanism for NFV-enabled software-defined MEC networks. We first investigated the system mode with problem statements and analysed the capacity constraints of cloudlets and QoS constraints of service. Then, we proposed an implementable unicast and multicast jointed routing mechanism, which enabled us to deliver data in unicast sessions along a minimal multicast tree. Thus, it can provide a generic multicast service in network layer for SDN-based MEC networks. The evaluation results indicate an evident improvement in terms of control message and signalling overhead ratio, utilization of flow/routing table, number of OFSs in hybrid network, and qualitative comparison. For future work, we will focus on mechanism optimization and experiments in various scenarios.

Data Availability

The data required to reproduce these findings cannot be shared at this time as the data also form part of an ongoing study.

Disclosure

This is an extended version of a paper entitled “Software Defined Unicast/Multicast Jointed Routing for Real-Time Data Distribution” [28] published in the proceedings of EAI Chinacom 2020.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grants nos. 61802281, 61702366, and 61972456); the Natural Science Foundation of Tianjin (Grants nos. 19JCYBJC15800, 18JCQNJC70300, and 18JCTPJC61900); the State Key Laboratory of ASIC and System (Fudan University) (Grant no. 2020KF012); the Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology (Grant no. TJUT-KLICNST-K20160002); the Key Laboratory of Applied Mathematics of Fujian Province University (Grant no. SX201904); and the Science and Technology Development Fund of Tianjin Education Commission for Higher Education (Grants nos. 2017KJ090 and 2018KJ215).

References

[1] K. Kaur, V. Mangat, and K. kumar, "A comprehensive survey of service function chain provisioning approaches in SDN and NFV architecture," *Computer Science Review*, vol. 38, Article ID 100298, 2020.

[2] 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.

[3] Q. Duan, S. Wang, and N. Ansari, "Convergence of networking and Cloud/Edge Computing: status, challenges, and opportunities," *IEEE Network*, vol. 34, no. 6, pp. 148–155, 2020.

[4] H. Gao, L. Kuang, Y. Yin et al., "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *ACM/Springer Mobile Networks and Applications (MONET)*, vol. 25, no. 4, pp. 1233–1248, 2020.

[5] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *ACM/Springer Mobile Networks and Applications (MONET)*, vol. 25, no. 4, pp. 1233–1248, 2020.

[6] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2020.

[7] H. Gao, W. Huang, and Y. Duan, "The cloud-edge-based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, vol. 21, no. 1, 2021.

[8] L. Yang, B. Ng, W. K. G. Seah et al., "A survey on network forwarding in Software-Defined Network," *Journal of Network and Computer Applications*, vol. 176, Article ID 102947, 2021.

[9] S. Chiang, J. Kuo, S. Shen et al., "Online multicast traffic engineering for software-defined networks," in *Proceedings of the IEEE 2018 Conference on Computer Communications*, pp. 414–422, Honolulu, USA, 2018.

[10] S. He, K. Xie, X. Zhou et al., "Multi-source reliable multicast routing with QoS constraints of NFV in edge computing," *Electronics*, vol. 8, no. 10, Article ID 1106, 2019.

[11] Y. Ma, W. Liang, J. Wu, and Z. Xu, "Throughput maximization of NFV-enabled multicasting in mobile edge cloud networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 2, pp. 393–407, 2020.

[12] H. Ren, Z. Xu, W. Liang et al., "Efficient Algorithms for delay-aware NFV-enabled multicasting in mobile edge clouds with resource sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 9, pp. 2050–2066, 2020.

[13] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, no. 1, pp. 78–88, 2000.

[14] M. Golkarifard, C. F. Chiasserini, F. Malandrino et al., "Dynamic VNF placement, resource allocation and traffic routing in 5G," *Computer Networks*, vol. 188, Article ID 107830, 2021.

[15] D. Qi, S. Shen, and G. Wang, "Towards an efficient VNF placement in network function virtualization," *Computer Communications*, vol. 138, pp. 81–89, 2019.

[16] Y. Guo, Z. Wang, X. Yin, X. Shi, and J. Wu, "Traffic engineering in hybrid SDN networks with multiple traffic matrices," *Computer Networks*, vol. 126, pp. 187–199, 2017.

[17] N. Rikhtegar, M. Keshtgari, O. Bushehrian et al., "BiTE: a dynamic bi-level traffic engineering model for load balancing and energy efficiency in data center networks," *Applied Intelligence*, 2021.

[18] Z. Xu, W. Liang, M. Jia, M. Huang, and G. Mao, "Task offloading with network function requirements in a mobile edge-cloud network," *IEEE Transactions on Mobile Computing*, vol. 18, no. 11, pp. 2672–2685, 2019.

[19] 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.

[20] Z. AlSaeed, I. Ahmad, and I. Hussain, "Multicasting in software defined networks: a comprehensive survey," *Journal of Network and Computer Applications*, vol. 104, pp. 61–77, 2018.

[21] O. Alhussein, P. T. Do, Q. Ye et al., "A virtual network customization framework for multicast services in NFV-enabled core networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1025–1039, 2020.

[22] L. Bondan, L. F. Müller, and M. Kist, "Multiflow: multicast clean-slate with anticipated route calculation on OpenFlow programmable networks," *Journal of Applied Computing Research*, vol. 2, no. 2, pp. 68–74, 2012.

[23] Y. Yu, Q. Zhen, L. Xin et al., "OFM: a novel multicast mechanism based on Openflow," *Advances in Information Sciences Service Sciences*, vol. 4, no. 9, pp. 278–286, 2012.

[24] Y.-D. Lin, Y.-C. Lai, H.-Y. Teng, C.-C. Liao, and Y.-C. Kao, "Scalable multicasting with multiple shared trees in software defined networking," *Journal of Network and Computer Applications*, vol. 78, pp. 125–133, 2017.

[25] J. Rückert, J. Blendin, and D. Hausheer, "Software-defined multicast for over-the-top and overlay-based live streaming in ISP networks," *Journal of Network and Systems Management*, vol. 23, no. 2, pp. 280–308, 2015.

[26] M. Charikar, J. Naor, and B. Schieber, "Resource optimization in QoS multicast routing of real-time multimedia," *Proceedings of IEEE INFOCOM*, IEEE, pp. 1518–1527, 2000.

[27] A. V. Priya and N. Radhika, "Performance comparison of SDN OpenFlow controllers," *International Journal of Computer Aided Engineering and Technology*, vol. 11, no. 4/5, pp. 467–479, 2019.

[28] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: survey, taxonomy, challenges," *IEEE Communication Survery Tutorial*, vol. 20, no. 1, pp. 333–354, 2018.

[29] S. Sun, W. Huang, X. Zhang et al., "Software defined unicast/multicast jointed routing for real-time data distribution," *Communications and Networking*, vol. 352, pp. 226–243, 2021.

[30] B. Fenner, M. Handleyin, H. Holbrook et al., "Protocol independent multicast-sparse mode (PIM-SM): protocol specification (revised)," *IETF RFC*, vol. 7761, 2016.

[31] T. Koch and A. Martin, "Solving Steiner tree problems in graphs to optimality," *Networks*, vol. 32, no. 3, pp. 207–232, 2015.

[32] M. J. Mišić and S. R. Gajin, "Simulation of software defined networks in Mininet environment," in *Proceedings of the 22nd Telecommunications Forum Telfor (TELFOR)*, pp. 1055–1058, IEEE, Belgrade, Serbia, 2014.

[33] S. Asadollahi, B. Goswami, and M. Sameer, "Ryu controller's scalability experiment on software defined networks," in *Proceedings of the 2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)*, IEEE, Bangalore, India, 2018.

[34] GT-ITM: https://www.cc.gatech.edu/projects/gtitm/.