Notice of Violation of IEEE Publication Principles

"Dynamic Embedding and Scheduling of Service Function Chains for Future SDN/NFV-Enabled Networks"
by Haotong Cao, Hongbo Zhu, and Longxiang Yang
in IEEE Access, March 2019, pp.39721-39730

After careful and considered review of the content and authorship of this paper by a duly constituted expert committee, this paper has been found to be in violation of IEEE's Publication Principles.

This paper copied original content from the paper cited below. The original content was copied without attribution (including appropriate references to the original author(s) and/or paper title) and without permission during the review process. One author (Haotong Cao) was responsible for the violation, and this violation was done without the knowledge of the other authors.

"On Dynamic Mapping and Scheduling of Service Function Chains in SDN/NFV-Enabled Networks"
by Junling Li, Weisen Shi, Peng Yang, and Xuemin (Sherman) Shen
Submitted to the IEEE International Conference on Communications Workshops (ICC Workshops), May 2019

Received March 8, 2019, accepted March 13, 2019, date of publication March 22, 2019, date of current version April 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2906874

Dynamic Embedding and Scheduling of Service Function Chains for Future SDN/NFV-Enabled Networks

HAOTONG CAO, (Student Member, IEEE), HONGBO ZHU, AND LONGXIANG YANG
Key Laboratory of Broadband Wireless Communication and Sensor Network Techniques, Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
Jiangsu Key Laboratory of Wireless Communications, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

Corresponding author: Longxiang Yang (yanglx@njupt.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC1314903, and in part by the National Natural Science Foundation of China under Grant 61372124 and Grant 61427801.

ABSTRACT Currently, software-defined networking (SDN) and network function virtualization (NFV) are the two most promising approaches for implementing network virtualization (NV). Traditional TCP/IP-based networks (e.g. Internet) will be transformed into SDN/NFV-enabled networks in the foreseeable future. Through virtualization, the slicing of heterogeneous underlying resources will provide an agile and customized virtual network (VN) services to end users. Hence, the virtual network function (VNF) embedding and scheduling are crucial to the VN service deployment in the SDN/NFV-enabled networks. In this paper, the dynamic VNF embedding and scheduling are jointly researched for enhancing the VN service provisioning. At first, the VNF embedding and scheduling are formulated by using the mixed integer linear programming (MILP) model, having the goals of minimizing the consumed underlying resources and providing QoS-guaranteed VN service. Subsequently, to remove the NP-hardness of the MILP model, a dynamic VNF embedding and scheduling algorithm is proposed. For instance, when a new VN service is requested, the VNFs, constituting the service function chain (SFC) of the VN service, will be embedded and scheduled by the proposed algorithm. If the resource and QoS requirements of the VNFs are not satisfied, a re-embedding and re-scheduling scheme will be triggered in order to optimize certain existing VNFs. The dynamic embedding and scheduling algorithm has flexible network function placement and improves the underlying resource utilization. Finally, the simulation results are illustrated to validate the proposed algorithm.

INDEX TERMS Future network, network virtualization, SDN, NFV, SFC, VN service, dynamic embedding and scheduling algorithm, QoS-guaranteed, MILP.

I. INTRODUCTION

Traditional TCP/IP based networks (e.g. Internet) have developed a lot over the past four decades. Meanwhile, the rigidity of TCP/IP based networks emerges along with its worldwide promotion [1]. In order to remove the rigidity, researchers have reached the consensus that network virtualization (NV) [2] will be one crucial attribute of the future network. Through virtualization, the architecture of future network will be more flexible, enabling to provide efficient resource allocation and customized network services. Software defined networking (SDN) [3] and network function virtualization (NFV) [4] are accepted as two promising approaches towards the virtualization. With respect to the SDN, it enables programmable and centralized network control, by separating the network control from data plane. With respect to the NFV, it can virtualize and realize network functions that run on dedicated network devices, also called middleboxes, in the form of software applications on high performance processors. Thus, traditional TCP/IP based networks will be transformed into SDN/NFV-enabled networks in the foreseeable future.

Considering the heterogeneous resources (CPU, storage, memory and communication bandwidth) of underlying

The associate editor coordinating the review of this manuscript and approving it for publication was Javed Iqbal.

substrate network (SN), how to efficiently manage, allocate and schedule the heterogeneous resources to implement different virtual network functions (VNFs) is the key problem in the SDN/NFV-enabled networks, known as NFV resource allocation problem [5]. In the literature, the resource allocation of VNFs is usually conducted in three stages: VNF composition, VNF embedding and VNF scheduling [5]. In this paper, we focus on the research of VNF embedding and VNF scheduling. With given various VNFs, VNF composition and resource information of the underlying SN, it is vital to embed the VNFs, chained and demanded, onto the underlying SN efficiently.

In the literature, VNF embedding and VNF scheduling have attracted significant attention from both academia and industry [4], [5]. However, most researchers focus on either VNF embedding [6]–[11] or VNF scheduling [12]–[15]. Few researchers concentrate on solving both VNF embedding and scheduling in a coordinated manner. In [16], the authors formulate the VNF embedding and VNF scheduling by using the integer linear programming (ILP) approach. Another heuristic algorithm is adopted to separately solve the VNF embedding and scheduling. In [17], the authors propose a greedy algorithm and a tabu search meta-heuristic algorithm to solve the VNF embedding and scheduling separately. However, these proposed algorithms are static. That is to say, the initial VNF embedding and scheduling assignment cannot be updated once they are executed. While in NV environments, VN service demands and underlying network conditions vary over time. Thus, dynamic VNF embedding and scheduling are required in order to increase the overall performance of VN services. In recent years, the dynamic VNF embedding has been studied in [7]–[9]. However, none of these research works consider the VNF scheduling after completing the VNF embedding. Consequently, the issue of dynamically re-adjust the initial VNF embedding and scheduling results has not been fully investigated yet.

Based on the above backgrounds, we aim at proposing a dynamic VNF embedding and scheduling algorithm. In the first place, we formulate the joint VNF embedding and scheduling problem, using the mixed integer linear programming (MILP) model. In the MILP model, CPU, node storage, communication bandwidth and VN quality of service (QoS) demand are formulated as virtual constraints. The recently emerging mobile applications, such as virtual reality and autonomous driving, are highly diversified in QoS demands. For example, virtual reality applications require gigabit per-user throughput. Thus, optimizing the QoS demand of VN service is very essential. In previous research [4], [5], VN QoS demand has not been considered. However, the MILP model cannot be directly solved within polynomial time. A dynamic algorithm is proposed instead. The dynamic algorithm is composed of two parts: one part is the greedy algorithm for initial VNF embedding and VNF scheduling, the other part is the QoS demand-triggered VNF re-embedding and re-scheduling for certain existing VNFs. With respect to the QoS demand-triggered scheme, it is designed in such a way that QoS sensitive VN services have higher priority in the VNF re-embedding and re-scheduling stage. In this paper, the VN execution delay is selected as the QoS demand of VN service. To validate the proposed dynamic algorithm, simulations are conducted. Simulation results reveal that the proposed dynamic algorithm is able to increase the VN service acceptance ratio by at least 15%. With respect to the substrate node (CPU) and link (communication bandwidth) utilization, they are improved by approximately 11% and 20%, respectively.

The rest of this paper is organized as follows. Formal VNF problem formulation is presented in Section II. In Section III, the joint VNF embedding and VNF scheduling are formulated by the MILP model. The dynamic heuristic algorithm is detailed in Section IV. In Section V, simulation results are illustrated to validate the dynamic heuristic algorithm. Finally, the conclusion and future work are briefly talked.

II. FORMAL PROBLEM MODEL AND DYNAMIC VNF DESCRIPTION

In this section, two sub-sections are involved. One is about the SN and VN service models. The other is about the dynamic VNF embedding and scheduling description.

A. FORMAL SN AND VN SERVICE MODELS

1) SUBSTRATE NETWORK

In the area of virtualization research, the underlying SN is usually modeled as a weighted graph $G^S = \left(N^S, L^S\right)$. In $G^S = \left(N^S, L^S\right)$, N^S is the set of all substrate nodes (e.g. servers, switches). With respect to the L^S, it represents the set of all substrate links. In this paper, all substrate nodes are fully connected. The communication bandwidth of each substrate link is adequate to forward the data traffic that is assigned to it. For each substrate node, such as $m \in N^S$, it has CPU C_m^S and node storage S_m^S for hosting multiple VNFs. For each substrate link, such as $mn \in L^S$, it has communication bandwidth CB_{mn}^S to support data traffic transmission. In addition, for any substrate node m, the set of its neighboring nodes is denoted by NE_m. If one node $n \in NE_m$, a direct link mn directly connects node m to n, no intermediate nodes. Last but not least, while embedding the given VN service, different VNFs of the VN service cannot share the same substrate node a time. For a better understanding of the underlying SN, we plot the Fig. 1.

2) VIRTUAL NETWORK SERVICE

Similar to previous research [7]–[17], the requested VN services arrive, following the Poisson distribution. The arrival rate is set λ (constant, Section V). With respect to the jth VN service VN_j^S, it is composed of a sequence of $|F_i|$ VNFs, also called a service function chain (SFC). As a consequence, the VN service VN_j^S is denoted by $VN_j^S = (s_j \rightarrow f_{j1} \rightarrow f_{j2} \rightarrow \ldots \rightarrow f_{j|F_i|} \rightarrow d_j, De_j)$, where s_j and d_j are the source and destination of the traffic flow. $|F_i|$ records the number of VNFs in VN_j^S. De_j, selected as the VN QoS demand in

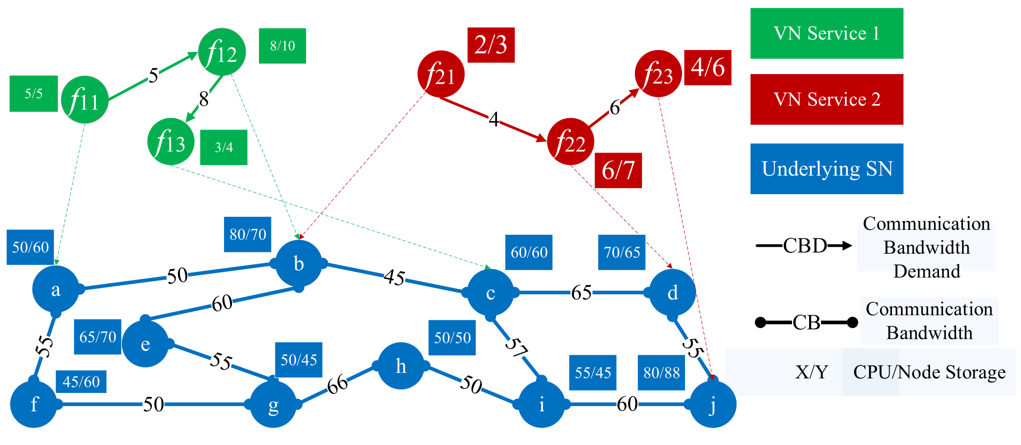

FIGURE 1. Underlying SN and two VN services.

this paper, represents the execution delay requirement of the VN_j^S. We also define t_j^a and t_j^c to denote the arrival time and completion time of VN_j^S. That is to say, the value of t_j^c minus t_j^a must not be higher than the value of De_j. Otherwise, the QoS demand of VN_j^S cannot be fulfilled. In addition, each VNF f_{jk} $(k = 1, \ldots, |F_i|)$ is associated with a set of selected substrate nodes in SN, labeled as $N(sel)$. Any substrate node in $N(sel)$ set can execute f_{jk}. We define the binary variable X_{jk}^m indicating whether the VNF f_{jk} can be embedded by substrate node m. If f_{jk} is embedded by substrate node m, the value of X_{jk}^m is 1. Otherwise, $X_{jk}^m = 0$. While conducting the VNF embedding, the CPU and storage demands of f_{jk} must be fulfilled by the embedded substrate node m. In addition, the execution time for VNF f_{jk} embedding onto substrate node m is denoted by Exe_{jk}^m. With respect to the communication bandwidth demand between joint VNFs f_{jk} and $f_{j(k+1)}$, it must be accommodated by the embedded substrate path $Path_{mn}^S$. Path splitting is not considered in this paper. For readers to understand the VN service, we plot another two VN services in Fig. 1.

B. DYNAMIC VNF EMBEDDING AND SCHEDULING DESCRIPTION

In the form of a sequence of VNFs, multiple VN services, with various resource and QoS requirements, are requested dynamically. One VN service is processed one time. The underlying SN has finite substrate resources, such as CPU, node storage and communication bandwidth. By conducting dynamic VNFs embedding and scheduling, we aim at finding the optimal embedding and scheduling assignments for all requested VN services. We try to maximize the VN services acceptance ratio and optimize the QoS performance of each VN.

From Fig. 2 to Fig. 5, we present the description of dynamic VNF embedding and scheduling of two VN services, labeled as VN_1^S and VN_2^S. $VN_1^S = (f_{11} \rightarrow f_{12} \rightarrow f_{13}, De_1)$. $VN_2^S = (f_{21} \rightarrow f_{22} \rightarrow f_{23}, De_2)$. In Fig. 1, we also label the

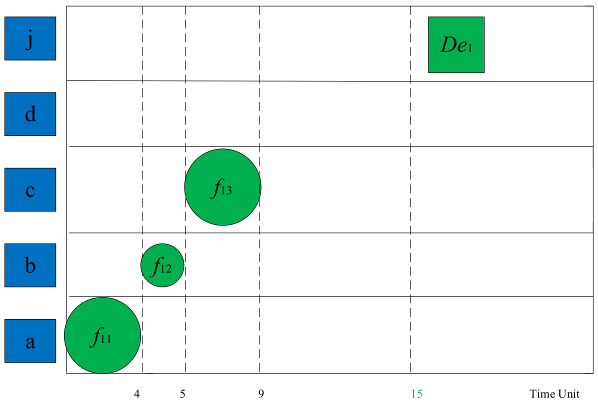

FIGURE 2. Initial VNF scheduling for VN service 1.

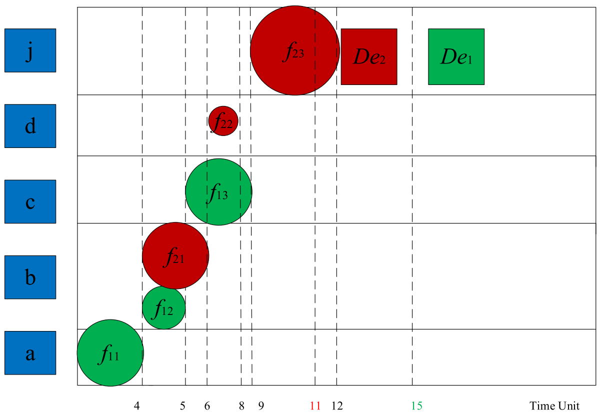

FIGURE 3. Initial VNF scheduling for VN service 2.

same substrate node to support different VNFs. For instance, substrate node b can support VNFs f_{12} (VN_1^S) and f_{21} (VN_2^S).

With respect to the VN_1^S, it arrives at time $t = 0$ with VN delay requirement $De_1 = 15$ time units. VN_2^S arrives at time $t = 4$ with a VN delay requirement of 7 time units. The initial VNF embedding for VN_1^S at $t = 0$ is shown

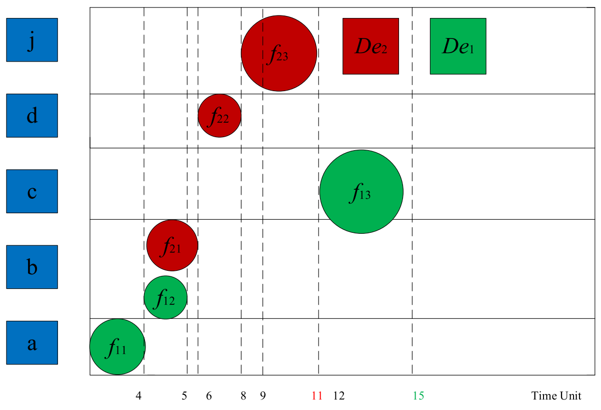

FIGURE 4. Dynamic VNF scheduling for two VN services.

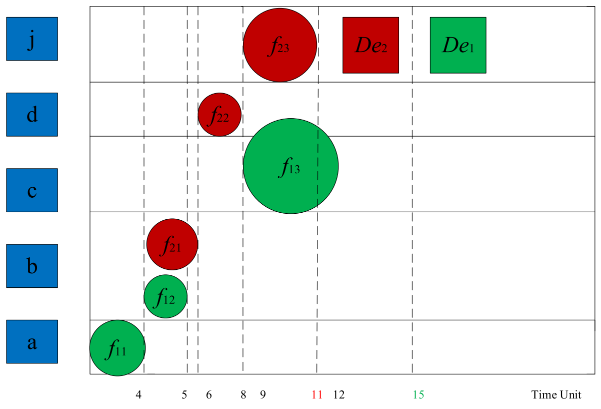

FIGURE 5. Dynamic VNF embedding and scheduling results.

in Fig. 2. VN_2^S arrives at $t = 4$ with initial VNF embedding results shown in Fig. 3. According to the VNF embedding results in Fig. 3, the embedding completion time of VN_2^S is at 12 time units, which violates the VN delay requirement. Therefore, VN_2^S will be rejected in the static VNF embedding and scheduling scheme [16], [17]. For dynamic VNF embedding and scheduling considered in this paper, when the initial scheduling for VN_2^S is failed, a delay-aware scheme is triggered to re-embed and re-schedule the existing VNFs. An example of VNF rescheduling scheme is illustrated in Fig. 4. After switching the processing sequence of f_{13} and f_{23}, the delay requirements of VN_1^S and VN_2^S can be satisfied simultaneously. Fig. 5 illustrates an example of VNF re-embedding, where f_{13} of VN_1^S is instantiated on substrate node c. In addition, f_{13} and f_{23} can both start processing at 8 time units. Additionally, the re-embedding reduces the completion time of VN_1^S, compared with that of re-schedule solution in Fig. 4.

Derived from dynamic VNF embedding and scheduling description, we can conclude that dynamic VNF embedding and scheduling can improve the acceptance of VN services and load the SN to its full capacity. For each VN service, the dynamic strategy enables to fulfill its resource demands and enhance its QoS performance.

III. MILP-BASED VNF EMBEDDING AND SCHEDULING FORMULATION

In this section, we formulate the VNF embedding and scheduling of a VN service, using the MILP model [5]. The MILP-based approach is able to achieve the optimal VNF embedding and scheduling, after much calculation [18].

In the first place, we define the binary variable $X_{jk}^m = 1$, indicating that substrate node m is assigned to accommodate the kth VNF of jth VN service. Otherwise, the value of X_{jk}^m is equal to 0. We further define a binary variable Y_{jk}^{mn}. If substrate link mn is adopted to forward data traffic for jth VN service, the value is 1. Otherwise, it is 0. In addition, Exe_{jk}^m is defined to record the execution time of the kth VNF of jth VN service embedding onto substrate node m.

A. OBJECT

The objective for the VN service VNF embedding and scheduling is to minimize the amount of consumed substrate resources, leaving more resource space for coming VN services and load SN to its full capacity:

$$\textbf{\textit{Min}}\ Cost_j \tag{1}$$

With respect to $Cost_j$, it records the amount of consumed substrate resources for accommodating the jth VN service. The $Cost_j$ is formulated in Expression 2.

$$Cost_j = \sum_{k=1, i\epsilon[1,|F_j|]}^{|F_j|} (\alpha \cdot C_{jk}^V + \beta \cdot S_{jk}^V + \gamma \cdot CB_{jki}^V) \tag{2}$$

where C_{jk}^V and S_{jk}^V refer to the required CPU and demanded node storage of VNF f_{jk}. CB_{ijk}^V refers to the required communication bandwidth of virtual link ki (no intermediate nodes). With respect to α, β and γ, they are weight factors, balancing different substrate resources.

B. CONSTRAINTS

The following constraints can be categorized into two parts. One part is for the VNF embedding of the VN service. The other part is for the VNF scheduling of the VN service.

Expressions 3 and 4 below are formulated to ensure the relationship between any VNF (e.g. f_{jk}) and any substrate node (e.g. m):

$$\forall f_{jk}\epsilon[1, |F_j|], \exists m\epsilon N^S, \quad X_{jk}^m = 1 \tag{3}$$
$$\forall m\epsilon N^S, \exists f_{jk}\epsilon[1, |F_j|], \quad X_{jk}^m \leq 1 \tag{4}$$

Expression 3 indicates that every VNF (e.g. f_{jk}, $k = 1, 2, \ldots, |F_j|$) in the given jth VN service must be embedded by one substrate node (e.g. node m). For Expression 4, every substrate node (e.g. node m) accommodates at most one VNF of the given VN service. Different VNFs of the jth VN service are not allowed to share the same substrate node.

With respect to the traffic flow constraints, we formulate Expression 5 below.

$$\forall f_{jk}\epsilon[1,|F_j|], \forall m, n\epsilon N^S, Y_{jk}^{mn} - Y_{jk}^{nm} = \begin{cases} 1, & \textit{source node} \\ 0, & \textit{else} \\ -1, & \textit{destination node} \end{cases} \quad (5)$$

where the top of Expression 5 ensures at least one substrate node m acts as the source node, with $Y_{jk}^{mn} = 1$. In the bottom of Expression 5, it ensures that at least one substrate node m serves as the destination node, with $Y_{jk}^{nm} = 1$. For any other substrate node (e.g. node n), it serves as the intermediate node.

With respect to substrate resource demands (CPU, node storage and communication bandwidth), they are formulated, ranging from Expressions 6 to 8.

$$\forall f_{jk}\epsilon[1,|F_j|], \quad \exists m\epsilon N^S, \ X_{jk}^m \cdot C_{jk}^V \le C_m^S \quad (6)$$
$$\forall f_{jk}\epsilon[1,|F_j|], \quad \exists m\epsilon N^S, \ X_{jk}^m \cdot S_{jk}^V \le S_m^S \quad (7)$$
$$\forall f_{jk}\epsilon[1,|F_j|], \quad \exists m\epsilon N^S, \ X_{jk}^m \cdot CB_{jki}^V \le CB_{mn}^S \quad (8)$$

Apart from the constraints for VNF embedding, we formulate the constraints for VNF scheduling of VN service, Expressions 9 and 10. At first, it is about the specified scheduling order of the VNFs ($VN_j^S = (f_{j1} \rightarrow f_{j2} \rightarrow \ldots \rightarrow f_{j|F_i|})$) in the VN service VN_j^S. It is formulated in Expression 9.

$$\forall f_{jk}\epsilon[1,|F_j|], \quad \exists m\epsilon N^S, \ X_{jk}^m \cdot Exe_{jk}^m \le t_{j,k+1}^S - t_{j,k}^S \quad (9)$$

Then, it is about the QoS guaranteed constraint of the VN service VN_j^S (Expression 10).

$$t_j^{start} + \sum_{k=1}^{|F_j|} X_{jk}^m \cdot Exe_{jk}^m - t_j^a \le De_j \quad (10)$$

where t_j^{start} denotes the embedding starting time of VN service VN_j^S. t_j^a denotes the arriving time of VN service VN_j^S.

As formulated above, the MILP-based VNF embedding and scheduling approach (Expressions 1-10, labeled as the ILP method in Section V) is an NP-hard combinatorial optimization problem whose optimal solution cannot be achieved within a polynomial time [18], [19]. For online and dynamic networking implementation, it is inefficient to solve the MILP frequently, as the long calculation time will incur considerable queuing delay for the VN services. To achieve fast and sub-optimal VN service provisioning, we propose an efficient heuristic algorithm for dynamic VNF embedding and scheduling.

IV. DYNAMIC HEURISTIC ALGORITHM FOR VNF EMBEDDING AND SCHEDULING

As VNF embedding and scheduling per VN service promises to run in dynamic networking environment, it is essential to develop the heuristic algorithm. The proposed heuristic algorithm for dynamic VNF embedding and scheduling will be divided into two stages. In the first stage, a greedy strategy is designed for initial VNF embedding and scheduling, upon the arrival of new requested VN services. Then, the proposed algorithm will verify whether the VN execution delay demand is satisfied or not. If the delay requirement is not satisfied, a delay-aware re-embedding and re-scheduling scheme is triggered, aiming at re-adjusting VNFs embedding and scheduling.

A. INITIAL VNF EMBEDDING AND SCHEDULING

With multiple VN services coming, we adopt the first-come-first-embed (FCFE) strategy [4], [5] for initial VNF embedding and scheduling. For instance, when the jth VN service VN_j^S comes, its VNFs ($f_{j1} \rightarrow f_{j2} \rightarrow \ldots \rightarrow f_{j|F_i|}$) will be embedded sequentially. For each VNF (e.g. f_{jk}), a set of candidate substrate nodes will be generated in the set $N(sel)^{f_{jk}}$. These candidate substrate nodes must have enough CPU and node storage for accommodating the VNF f_{jk}. The VN service will be rejected if there is no candidate substrate node in $N(sel)^{f_{jk}}$. Then, the candidate substrate nodes for the VNF f_{jk} will be stored according to their available starting times. For example, the time when f_{jk} needs to be processed if it is embedded onto its candidate node. The candidate node that provides the earliest starting time for f_{jk} will be chosen to accommodate the VNF f_{jk}. The starting time f_{jk} depends on both the completion time of its previous embedded VNF $f_{(j-1)k}$ and the earliest available start time of the candidate node when the f_{jk} is embedded. In addition, when conducting the f_{jk} embedding, the communication bandwidth between f_{jk} and $f_{j(k-1)}$ must not be more than that between their embedded substrate nodes. With fulfilling resource demands (CPU, node storage and bandwidth), the initial VNFs embedding and scheduling of the VN_j^S is completed. With respect to remaining VNFs in the VN_j^S, the embedding and scheduling strategy repeats.

B. DELAY-AWARE VNF RE-EMBEDDING AND RE-SCHEDULING ALGORITHM

As usual, the VNFs scheduling and the completion time t_j^c for VN_j^S can be obtained immediately after the successful initial VNF embedding and scheduling of VN_j^S. We will check whether the VN execution delay requirement of the processed service is satisfied. If De_j is not more than the value of ($t_j^a - t_j^c$), the embedded VN service VN_j^S will be admitted with corresponding CPU, storage and bandwidth resources being allocated. Otherwise, the delay-aware VNF re-embedding and re-scheduling scheme will be triggered immediately (See our **Algorithm 1**).

In the first step, all VN services (e.g. the VN services with $t_j^a - t_j^c, j = 1, 2, \ldots, i-1$), requiring to be re-embedded, together with the new coming services (e.g. $VN_k^S, k \neq j$) will be added into the new VN services set S_U. Next, our proposed algorithm will be adopted to re-embed and re-schedule all VNFs in S_U set to admit all VN services, while ensuring the

Algorithm 1 Delay-Aware VNF Re-Embedding and Re-Scheduling Algorithm

```
Input: All VN services in the set $S_U$ and the underlying SN
Output: Re-embedded and re-scheduled results of VN services, e.g. $VN_k^S$
1:  Add all requiring re-embedding VN services to form up the set $S_U$;
2:  for $j = 1, 2, \ldots, i-1$ do
3:      if $t_j^c \geq t_j^a$ then
4:          Add VN service $S_j$ to VN set $S_U$;
5:          Calculate the Room value of $S_j$, $Ro(j)$, for re-embedding and re-scheduling;
6:      end if
7:  end for
8:  for $j = 1, 2, \ldots, |S_U|$ do
9:      if $Ro(j)$ has the smallest value among all Room values then
10:         $Ro(j)$ has the highest priority;
11:         Conduct the re-embedding and re-scheduling of $Ro(j)$;
12:         Until no VN services left.
13:     end if
14: end for
```

QoS requirements. We define and calculate the Room *Ro*(*j*) for any service VN_j^S as $Ro(j) = De(j)\text{-}(t_j^c - t_j^a)$. The Room of an existing service represents the sensitive delay degree of a service. A smaller value of *Ro*(*j*) implies that VN_j^S has limited time for VNF re-embedding and re-scheduling, which indicates VN_j^S has a higher priority in the VNF re-embedding and re-scheduling procedure. The proposed algorithm sorts all the existing services in S_U in ascending order of *Ro*(*j*). After that, the VNFs in the existing services will be re-embedded and re-scheduled sequentially. At this stage, incompleted VNFs for each service VN_j^S are re-embedded and re-scheduled using the greedy algorithm. If the delay requirements of all the existing services can be satisfied after the re-embedding and re-scheduling procedure, we will update the VNF embedding and scheduling results for all VN services in S_U. Otherwise, the VNF embedding and scheduling results will remain unchanged.

Two major advantages of our dynamic embedding and scheduling algorithm are: 1) allowing the switching of a VNF from one substrate node to another and the re-scheduling of VNF, which improves the resource utilization and reduces the VN service completion time; 2) having much shorter algorithm execution time compared with the MILP-based approach, detailed in Section III.

C. DISCUSSION OF ALGORITHM TIME COMPLEXITY

For the initial VNF embedding and scheduling of the given VN service, its time complexity lies in the construction of $|N(sel)^{f_{jk}}|$ for all VNFs in the VN service. The construction procedure has a time complexity of $O(|N(sel)^{f_{jk}}|log|N(sel)^{f_{jk}}|)$ [18]. With respect to the re-embedding and re-scheduling scheme, we first sort S_U incompleted services according to their Room values, which has a time complexity of $O(|S_U|log|S_U|)$. Then, for each VNF re-embedding and re-scheduling, the time complexity is $O(|N(sel)^{f_{jk}}|log|N(sel)^{f_{jk}}|)$. Hence, our proposed algorithm is a polynomial-time algorithm. As a consequence, we can evaluate our proposed algorithm in continuous time (Section V).

TABLE 1. SN and VN service settings for simulation.

Substrate Nodes of SN	60
CPU, Node Storage of SN	[50,100], Uniform Distribution
Communication Bandwidth of SN	[50,100], Uniform Distribution
VN Service Arrival Rate (λ)	[5, 15] per 1000 time units, Integer, Uniform Distribution
Number of VNFs per VN Service	[2,6], Uniform Distribution
CPU, Node Storage per VN Service	[1,20], Uniform Distribution
Communication Bandwidth per VN Service	[1,20], Uniform Distribution
VNF Embedding Time	[1,8], Uniform Distribution

V. SIMULATION RESULTS

We will illustrate the simulation results in this section. We aim at highlighting our proposed dynamic embedding and scheduling algorithm (sub-sections B and C). As no closely related algorithms are proposed in the literature, we compare our dynamic embedding and scheduling algorithm with its static counterpart. In addition, we conduct an extra simulation in order to highlight our dynamic algorithm optimality (sub-section D).

A. SIMULATION SETTINGS

For the simulation, the underlying SN is generated by the Waxman model approach [19]. For readers to reproduce our simulation, we list out SN parameter settings in Table 1. With respect to each VN service request, it is generated by the Waxman model approach, too. VNs are set to be demanded obeying the well-known Poisson distribution. We further set VN service arrival rate (λ) in Table 1. With respect to each VN service lifetime, we set an exponentially distributed lifetime of 1000 time units. In Table 1, we also provide parameter settings for each VN service request. In addition, the simulation lasts for 100000 time units that is seen as a continuous time event.

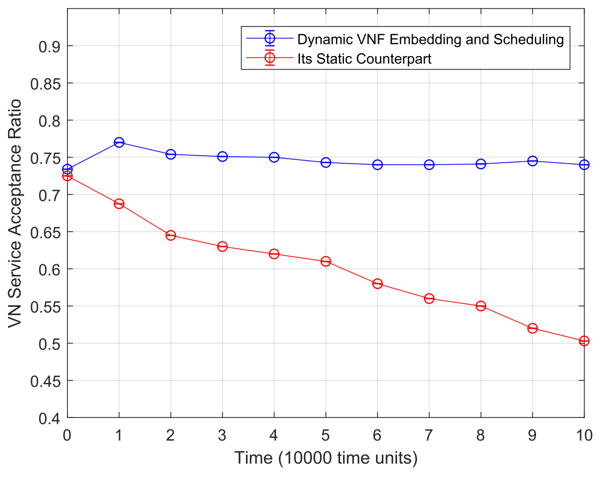

FIGURE 6. VN service acceptance ratio.

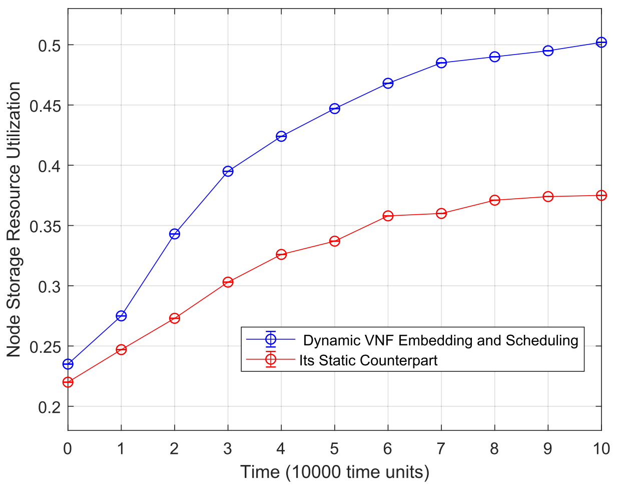

FIGURE 8. Node storage resource utilization.

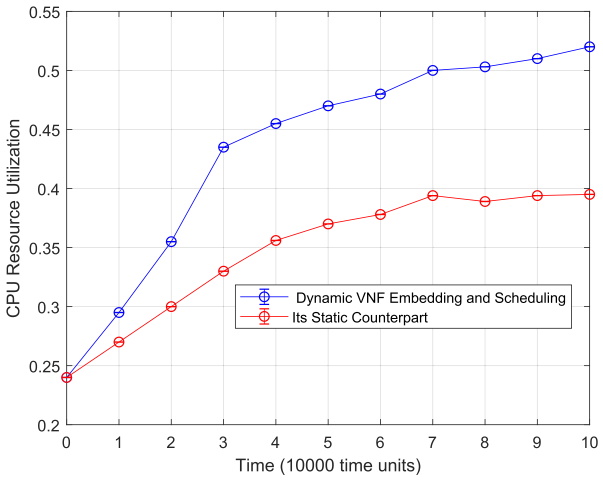

FIGURE 7. CPU resource utilization.

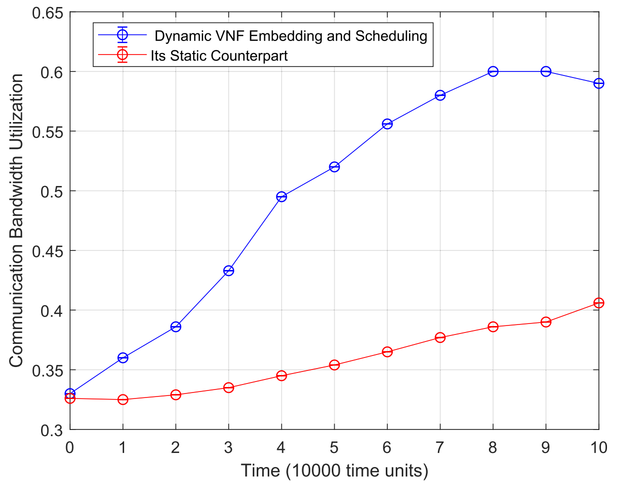

FIGURE 9. Communication bandwidth utilization.

B. PERFORMANCE COMPARISON FOR FIXED VN SERVICE ARRIVAL RATE

In this sub-section, we aim at illustrating the efficiency of our dynamic VNF embedding and scheduling algorithm in terms of the VN service acceptance ratio, substrate node (CPU, node storage) and link (communication bandwidth) resource utilization. The simulation is carried out for embedding 500 VN services, with VN service arrival rate 5 per 1000 time units.

The VN service acceptance performance comparison between the static and dynamic embedding and scheduling algorithms is plotted in Fig. 6. The curves of two algorithms indicate that the VN service acceptance ratios of the two algorithms are similar at the early stage. However, with the simulation time expanding, the dynamic VNF embedding and scheduling algorithm achieves higher VN service acceptance ratio than the static counterpart. On average, the proposed dynamic algorithm admits 15% more services than the static solution. Similar advantage can be observed from the substrate CPU resource utilization comparison and node storage utilization comparison in Fig. 7 and Fig. 8. Two algorithms perform similarly at the early stage of simulation. However, the dynamic VNF embedding and scheduling algorithm achieves approximately 11% higher CPU resource utilization than its static counterpart. To the node storage utilization, our dynamic algorithm is 10% higher than that of the static counterpart, since 50000 time unit. With respect to the link communication bandwidth performance, it is plotted in Fig. 9. Derived from the simulation results, the dynamic VNF embedding and scheduling algorithm achieves approximately 20% higher than that of the static counterpart.

C. PERFORMANCE COMPARISON FOR VARIED VN SERVICE ARRIVAL RATE

In Fig. 10, we plot the performance comparison of the VN service acceptance ratio of the two algorithms with varied VN service arrival rates ranging from [5], [15] services per 1000 time unit. It can be concluded that the proposed dynamic algorithm outperforms the static counterpart. It is owing to the fact that our proposed algorithm allows

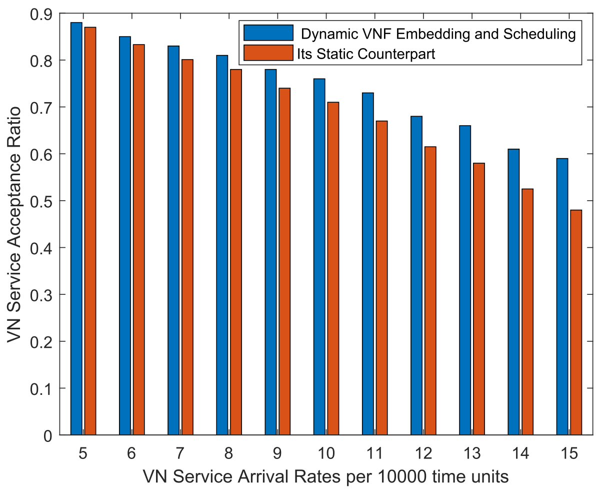

FIGURE 10. VN service acceptance ratio with varied arrival rate.

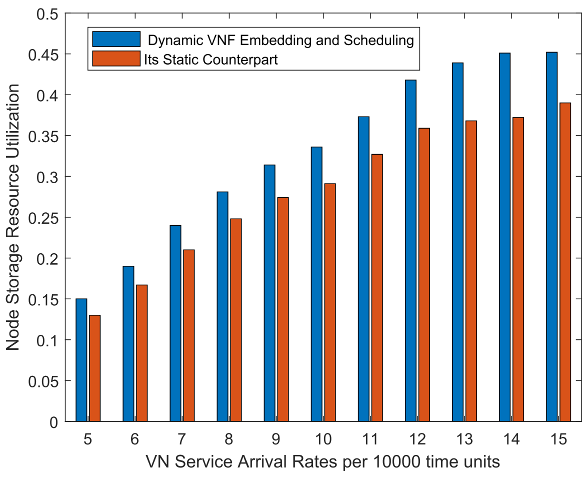

FIGURE 12. Node storage utilization with varied arrival rate.

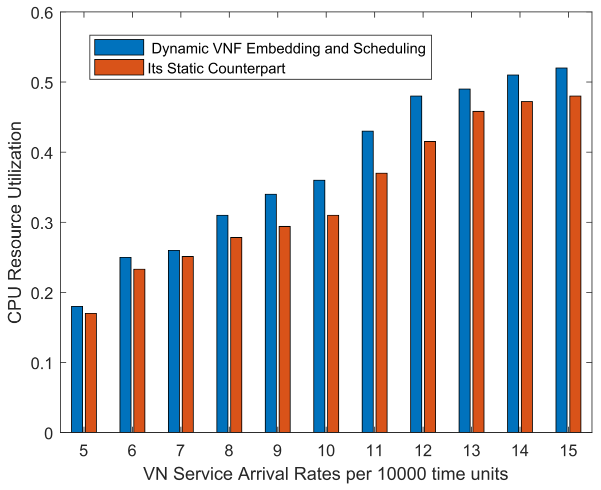

FIGURE 11. CPU resource utilization with varied arrival rate.

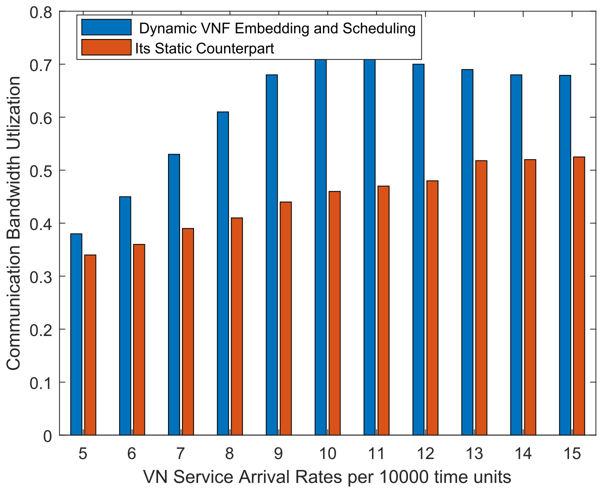

FIGURE 13. Communication bandwidth utilization with varied arrival rate.

re-embedding and re-scheduling of the existing VNFs in the underlying SN. Thus, our algorithm increases the chances of accommodating more proposed VN services. In addition, our dynamic embedding and scheduling algorithm optimizes the allocation and utilization of the physical resources. However, it can also be observed that the performance gap between the two compared algorithms is enlarged, with VN service arrival rate extending. It is owing to the fact that a lower VN service arrival rate will contribute to lower underlying SN loading. Therefore, more underlying resources will leave for accommodating new coming VN services.

With respect to Fig. 11 and Fig. 12, they are the performance comparisons of the CPU resource utilization and node storage utilization of the two algorithms. Derived from Fig. 11 and Fig. 12, we can reach the conclusion that CPU utilization and node storage utilization of two algorithms increase with accommodating more and more VN services. In addition, our dynamic VNF embedding and scheduling algorithm outperforms the static counterpart. The reasons for this behavior has been talked in above sub-section B. With respect to the Fig. 13, we plot the performance comparison of the communication bandwidth utilization of the two algorithms with varied VN service arrival rates. Similar to above figures, our dynamic VNF embedding and scheduling algorithm outperforms the static counterpart. However, an exception needs to be concerned that the bandwidth performance gap between the two compared algorithms is not just enlarged as VN service arrival rate increases. The gap is firstly enlarged. Then, since 10 VN services point, the gap decreases. The cause of this behavior indicates that there are no infinite communication bandwidth resources in the whole SN. In order to accommodate more VN services, our dynamic VNF embedding and scheduling algorithm will adjust the loaded links to their full capacities. Consequently, the utilization will decrease to a stable state. To the static counterpart, its greedy manner will lead to link bottleneck, though link utilization increases. In the end, the bandwidth utilization gap will decrease.

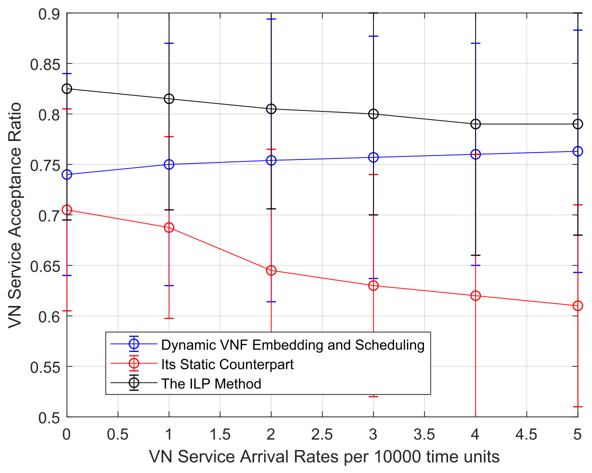

FIGURE 14. VN service acceptance ratio.

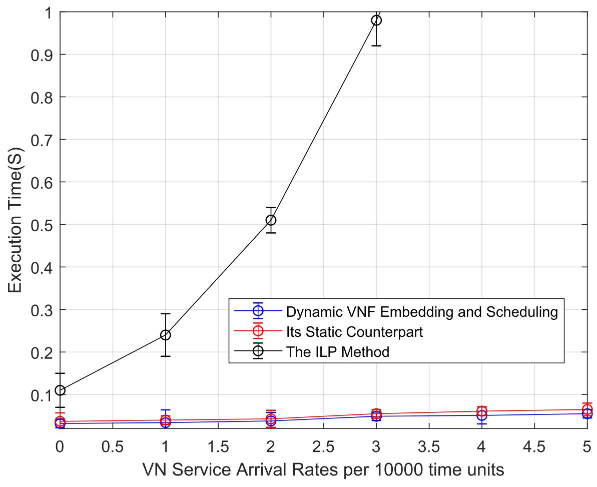

FIGURE 15. VN service execution time.

D. OPTIMALITY COMPARISON FOR OUR DYNAMIC ALGORITHM

In SFC research, the (M)ILP method is widely accepted as the approach to achieve the optimal and exact VNF embedding per VN service. Therefore, we conduct an extra performance comparison in this sub-section, aiming at highlighting our dynamic algorithm optimality. Considering the fact that the (M)ILP method is NP-hard, the (M)ILP method is time-consuming for accommodating VN service. Hence, the SN scale is limited to 30 nodes for simulation. The VN service arrival rate is set ranging from [0, 5], too. Other simulation settings are same to what are presented in Table 1.

We plot the VN service acceptance ratio (Fig. 14) and VN service execution time (Fig. 15) with varied VN service arrival rates. They are typical for evaluating the optimality. Other simulation results, such as CPU utilization, they are not illustrated.

Derived from Fig. 14, we can get our main conclusion. First and foremost, it is about the optimality. By comparison, our dynamic algorithm cannot behave as well as the ILP method. It is owing to the nature of the exact algorithm and the heuristic algorithm. However, with more and more VN services being proposed, both algorithms will come to stable states. An apparent gap between our dynamic algorithm and the ILP method exists. Though the gap is not large, our dynamic algorithm needs to be improved. In addition, the gap indicates that more and more QoS demand-triggered dynamic algorithms can be proposed in the future SFC research.

With respect to Fig. 15, we can find that the ILP method is very time-consuming. Hence, it cannot be promoted to VN service implementation in the real networking environment. In the real networking environment, each VN service, requested by user, comes dynamically. The waiting time and execution time of the VN service is limited. Though the optimality of our dynamic algorithm is not as good as the ILP method, it can deploy the requested VN service in limited time.

VI. CONCLUSION AND FUTURE WORK

In order to coordinate the VNF embedding and scheduling of VN service, a dynamic heuristic algorithm is proposed in this paper. The goals of this dynamic algorithm are to minimize the consumed underlying resources and provide QoS-guaranteed VN service for future SDN/NFV-enabled networks. Especially, a QoS demand-triggered VNF re-embedding and re-scheduling scheme is presented and incorporated into the proposed algorithm. In previous works, optimizing the QoS demand of VN service has not been fully considered yet. Simulation results vividly show that the VN service acceptance ratio of the proposed dynamic embedding algorithm is at least 15% higher than the static counterpart. In addition, the substrate node (CPU, node storage) and link utilization (communication bandwidth) of the dynamic algorithm are higher than the static counterpart.

In general, this paper is regarded as our preliminary research on SFC embedding and scheduling. In the future, we will propose many other dynamic and efficient algorithms for VNF embedding and scheduling of SFC. In addition, we intend to incorporate the link propagation delay into the QoS demand of VN service while conducting the VNF embedding and scheduling.

ACKNOWLEDGMENT

Some preliminary results of this paper [20] will be presented in the IEEE International Conference on Communications Workshops, May 20–24, 2019, Shanghai, China.

REFERENCES

[1] A. S. Tanenbaum and D. Wetherall, *Computer Networks*, 5th ed. London, U.K: Pearson Education, 2011.

[2] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2017–2022," white paper, 2016. [Online]. Available: https://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/white-paper-listing.html

[3] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1617–1634, 3rd Quart., 2014.

[4] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 236–262, 1st Quart., 2016.

[5] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[6] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint VNF placement and CPU allocation in 5G," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2018, pp. 1943–1951.

[7] H. Cao, L. Yang, and H. Zhu, "Novel node-ranking approach and multiple topology attributes-based embedding algorithm for single-domain virtual network embedding," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 108–120, Feb. 2018.

[8] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 3, pp. 543–553, Sep. 2017.

[9] H. Cao, Y. Zhu, G. Zheng, and L. Yang, "A novel optimal mapping algorithm with less computational complexity for virtual network embedding," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 356–371, Mar. 2018.

[10] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2008–2025, Aug. 2017. doi: 10.1109/TNET.2017.2668470.

[11] D. Dietrich, A. Rizk, and P. Papadimitriou, "Multi-provider virtual network embedding with limited information disclosure," *IEEE Trans. Netw. Service Manag.*, vol. 12, no. 2, pp. 188–201, Jun. 2015.

[12] X. Zhong, Y. Wang, and X. Qiu, "Service function chain orchestration across multiple clouds," *China Commun.*, vol. 15, no. 10, pp. 99–116, Oct. 2018.

[13] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Trans. Commun.*, vol. 64, no. 9, pp. 3746–3758, Sep. 2016.

[14] P.-W. Chi, Y.-C. Huang, and C.-L. Lei, "Efficient NFV deployment in data center networks," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2015, pp. 5290–5295.

[15] H. Chen *et al.*, "Mosc: A method to assign the outsourcing of service function chain across multiple clouds," *Comput. Netw.*, vol. 133, pp. 166–182, Mar. 2018.

[16] H. A. Alameddine, L. Qu, and C. Assi, "Scheduling service function chains for ultra-low latency network services," in *Proc. 13th Int. Conf. Netw. Service Manage.*, Nov. 2017, pp. 1–9.

[17] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *Proc. 1st IEEE Conf. Netw. Softwarization (NetSoft)*, Apr. 2015, pp. 1–9.

[18] T. H. Cormen, C. Stein, R. Rivest, and C. Leiserson, *Introduction to Algorithms*, 2nd ed. New York, NY, USA: McGraw-Hill,, 2001.

[19] B. M. Waxman, "Routing of multipoint connections," *IEEE J. Sel. Areas Commun.*, vol. JSAC-6, no. 9, pp. 1617–1622, Dec. 1988.

[20] H. Cao, Y. Zhang, and L. Yang, "Dynamic embedding and scheduling of virtual network service for future networks," in *Proc. IEEE Int. Conf. Commun. (ICC) Workshops*, May 2019, pp. 1–6.

HAOTONG CAO (S'17) received the B.S. degree in communication engineering from the Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China, in 2015, where he is currently pursuing the Ph.D. degree. He was a Visiting Scholar with Loughborough University, U.K., in 2017. His research interests include next generation networks, wireless communication theory, and resource allocation in wired and wireless networks. He has served as a TPC member of multiple IEEE conferences, such as the IEEE INFOCOM, the IEEE ICC, the IEEE Globecom, and the IEEE WCNC. He has received the 2018 Postgraduate National Scholarship of China. He was a recipient of the Best Paper Award from the WiSATS 2019. He is also serving as a Reviewer of multiple academic journals, such as the IEEE Internet of Things Journal, the IEEE/ACM Transactions on Networking, the IEEE Transactions on Network and Service Management, and *Computer Networks* (Elsevier). He has published multiple IEEE Transactions/Journal/Magazine papers, since 2016.

HONGBO ZHU received the B.S. degree in communications engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 1982, and the Ph.D. degree in information and communications engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 1996. He was the Vice-President of the Nanjing University of Posts and Telecommunications, where he is currently a Professor. He is also the Head of the Coordination Innovative Center of IoT Technology and Application, Jiangsu, which is the first governmental authorized Coordination Innovative Center of IoT in China. He also serves as a referee or expert in multiple national organizations and committees. He has authored and co-authored over 300 technical papers published in various journals and conferences. He is currently leading a Big Group and multiple funds on IoT and wireless communications, with the current focus on architecture and enabling technologies for the Internet of Things. His research interests include the Internet of Things, mobile communications, and wireless communication theory.

LONGXIANG YANG is currently with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China, where he is also a Full Professor and a Doctoral Supervisor. He is also the Vice Dean of the College of Telecommunications and Information Engineering, NJUPT. He has fulfilled multiple National Natural Science Foundation projects of China. He has authored and co-authored over 200 technical papers published in various journals and conferences. His research interests include cooperative communication, network coding, wireless communication theory, key technologies of 5G mobile communication systems, ubiquitous networks, and the Internet of Things.