On Dynamic Service Function Chain Deployment and Readjustment

Junjie Liu, Wei Lu, Fen Zhou, Ping Lu, and Zuqing Zhu, *Senior Member, IEEE*

Abstract—Network function virtualization (NFV) is a promising technology to decouple the network functions from dedicated hardware elements, leading to the significant cost reduction in network service provisioning. As more and more users are trying to access their services wherever and whenever, we expect the NFV-related service function chains (SFCs) to be dynamic and adaptive, i.e., they can be readjusted to adapt to the service requests' dynamics for better user experience. In this paper, we study how to optimize SFC deployment and readjustment in the dynamic situation. Specifically, we try to jointly optimize the deployment of new users' SFCs and the readjustment of in-service users' SFCs while considering the trade-off between resource consumption and operational overhead. We first formulate an integer linear programming (ILP) model to solve the problem exactly. Then, to reduce the time complexity, we design a column generation (CG) model for the optimization. Simulation results show that the proposed CG-based algorithm can approximate the performance of the ILP and outperform an existing benchmark in terms of the profit from service provisioning.

Index Terms—Network function virtualization (NFV), service function chain (SFC), dynamic support, column generation.

I. Introduction

TRADITIONALLY, service providers rely on middleboxes to realize the network functions that are needed as a part of a specific service [1]. However, this scheme is restricted by the drawbacks of dedicated hardware, which results in difficulties in deployment and maintenance, prolonged time to market, and high expenses, and thus it cannot adapt to the requirements of emerging applications related to cloud computing [2], [3] and big data [4]. To address these issues, network function virtualization (NFV) [5] was proposed to migrate network functions from expensive dedicated hardware to software-defined elements by leveraging IT resource virtualization, *i.e.,* processing traffic with virtual network functions (vNFs) [6]. Hence, service providers do not need to worry too much about the hardware environment during service provisioning.

Note that, with NFV, we can represent a network service with a series of connected vNFs, *i.e.,* formulating a service function chain (SFC) [7]. This could be challenging for network control and management (NC&M) when the constraints on IT and bandwidth resources need both to be addressed well [8]. Previously, researchers have considered the problem of vNF placement in [9] and [10], while the studies in [7], [11], and [12] have addressed how to deploy SFCs. Nevertheless, in order to fully explore the advantages of NFV, we have more challenges to work on. For instance, with the rapidly growth of mobile devices, more and more users are trying to access their services wherever and whenever [13]. Consequently, SFCs need to be readjusted to ensure service continuity when the users are moving, and vNFs might be added in or removed from SFCs to adapt to their demands. However, how to set up and readjust SFCs adaptively such that the dynamic nature of user demands can be properly addressed has not been fully explored yet.

For SFCs, the optimal deployment of vNFs will change when their users move [14]. Hence, a static SFC deployment scheme can make the paths among the users and their vNFs sub-optimal, which would not only cause unnecessary bandwidth consumption but also degrade user experience. Moreover, the static deployment approaches can not address the situation in which users can change their SFC patterns on-the-fly, since the consideration on vNFs reassignment is missing. On the other hand, if we readjust SFC deployment schemes too frequently, the overhead from vNF migrations could become a serious issue [15]. Therefore, for the dynamic SFC deployment and readjustment, we have to carefully balance the tradeoff between resource consumption and operational overhead.

This paper studies how to optimize vNFs deployment and reassignment to efficiently orchestrate SFCs in respond to dynamic user demands. Here, as we consider the service provisioning step after the formulation of SFCs, the order of vNFs in each SFC is predetermined. We also assume that an SFC is provisioned for one user and thus each SFC would take the one-dimensional chain topology. Since our network model allows multiple SFCs to share certain vNFs, more complex and two-dimensional vNF graphs are also addressed in our work. However, the SFCs can actually be more generic with more complex and flexible topologies [16], [17], which will be addressed in our future work. For the vNF deployment, we consider the cost of two parts, *i.e.,* IT and bandwidth resource

consumption. For the vNF reassignment, we consider the associated operational overhead. To solve the SFC provisioning problem with these considerations, we first formulate an integer linear programming (ILP) model to maximize the service provider's profit under the resource and operational overhead constraints. Then, to reduce the time complexity, we design a column generation (CG) [18], [19] model for the optimization to approximate the performance of the ILP. In summary, our main contributions are as follows.

- We consider a practical network scenario for dynamic SFC deployment and readjustment and formulate an ILP model to solve the optimization exactly.
- We design a CG model based on the feasible service provisioning schemes of the users and leverage it to reduce the time complexity of the optimization.

The rest of the paper is organized as follows. Section II provides a brief survey on the related work. In Section III, we describe the network model and formulate the problem of dynamic SFC deployment and readjustment. An ILP model is formulated in Section IV to solve the problem exactly, with formal complexity analysis. Section V discusses the CG model for the problem. The performance evaluation is presented in Section VI. Finally, Section VII summarizes the paper.

II. RELATED WORK

Since its inception, NFV has attracted intensive interests from both academia and industry. For comprehensive surveys on it, one is suggested to refer to [20] and [21]. Meanwhile, the standardization activities related to NFV are making important progress recently [5], and drafts on the use cases of NFV and the description of SFCs can be found in [22] and [23], respectively. The work in [24] and [25] investigated NFV from the perspective of system implementation. Specifically, Martins *et al.* considered how to realize NFV with ClickOS in [24], while Mamatas *et al.* [25] proposed an information-exchange-management-as-a-service facility as an extension to the standard NFV management framework, which can support information flow establishment, operation, and optimization.

The studies in [7], [9], [11], and [26]–[30] have investigated the problems of vNF or SFC deployment from the perspective of resource allocation. Cohen *et al.* [26] formulated a problem on vNF placement to minimize both the deployment and connection costs of vNFs, and proposed approximation algorithms that could solve it with performance guarantee. However, their studies did not consider the resource constraints of substrate nodes and links. Moens and De Turck [9] considered a hybrid scenario in which network services can be provisioned with both the hardware-specified elements and vNFs. A binary search algorithm was designed in [27] to minimize the number of deployed vNFs under the constraint of data-transfer latency. The work in [28] introduced a dynamic programming based heuristic called Viterbi to solve the problem of SFC deployment. Mehraghdam *et al.* [7] used a context-free language to formulate the problem of SFC deployment as a mixed integer quadratically constrained program model and use the Pareto set analysis to balance the tradeoffs among several optimization goals. Xia *et al.* [11] studied the problem of

vNF placement in optical inter-datacenter (inter-DC) networks and proposed a binary integer programming model to minimize the cost due to optical-electronic-optical conversions. The study in [30] developed a prediction scheme that can forecast NFV's future resource requirements, with which the service provider can spin up new resources more wisely and/or plan global availability more effectively. Nevertheless, none of the aforementioned studies considered to support dynamic user demands in vNF/SFC deployment. Note that, for the dynamic schemes, the problem of SFC deployment becomes much more complex since the readjustment of vNF/SFC deployment has to be taken into account and we need to balance the tradeoff between resource consumption and operational overhead.

The work in [31]–[34] have previously addressed dynamic vNF/SFC deployment. Sun and Ansari [31] proposed a novel network architecture for cloudlet (*i.e.,* a mobility-enhanced small-scale cloud DC located at the edge of the Internet). Specifically, they assumed that each user communicated with a specific virtual machine (VM) called Avatar in the cloudlet, investigated how to realize live Avatar migration, and proposed a strategy to optimize the tradeoff between data-transfer latency and VM migration overhead. Note that, even though they considered the mobility feature of the users, their problem is fundamentally different from ours. Basically, each SFC consists of multiple connected VM/vNFs and the relation among them has to be considered in the deployment and readjustment of SFCs. Ghaznavi *et al.* [32] studied the problem of elastic vNF placement problem and tried to balance the trade-off between elastic overhead and resource consumption. However, they assumed that there was only one type of vNFs and did not consider SFC. Note that, deploying an SFC usually involves the orchestration of several vNFs in different types, which apparently makes the problem-solving more complex. Ayoubi *et al.* [33] designed an availability-aware resource allocation and reconfiguration framework for the elastic services in failure-prone DC networks. Although they also considered to reconfigure the embedding schemes of virtual network services when the users' requests change over time, their focus was on providing the highest availability improvement with the lowest reconfiguration cost. Hence, both their network model and optimization objective are different from ours. In [34], the migration scheme of vNF instances (vNFIs) in dynamic SFCs have been studied to minimize the costs from energy consumption and reconfiguration. The authors proposed algorithms to handle the VNFI placement, SFC routing, and VNFI migration in response to the change of user workload. Nevertheless, they did not consider the bandwidth consumption of SFCs when designing their algorithms, which makes their network model also different from ours.

Finally, we hope to point out that even though the problem of dynamic SFC deployment looks similar to the virtual network embedding (VNE) problem that has already been studied intensively before [35]–[39], they are fundamentally different. Basically, in VNE, the virtual networks' topologies would not change during the embedding, while in dynamic SFC deployment, the actual topology of an SFC can change with the vNF placement.

III. PROBLEM DESCRIPTION

This section describes the network model and formulates the problem of dynamic SFC deployment and readjustment.

A. Network Model

The network can be denoted as $G = (V, E)$, where V represents the set of nodes and E is the set of physical links. Here, we assume $V = V_d \bigcup V_s$, which means that there are two types of nodes in the network, *i.e.*, the DCs and the switches, and their sets are V_d and V_s, respectively. The switches can be used as the access points for mobile users and they also forward data traffic in the network, while the DCs contain the IT resources for vNF deployment. Each DC is locally attached to a switch.[1] For every link $e \in E$ between two adjacent switches, we use B_e to denote its bandwidth capacity. The IT resource capacity of a DC $v \in V_d$ is denoted as C_v. We use $\mathcal{P}$ to represent the set of vNF types that can be instantiated in the DCs, *e.g.*, network address translation (NAT) and firewall. For a p-th type vNF, we denote its IT resource requirement as c_p and assume that it can serve n_p users at most.

B. User Model

The user i's service request can be represented by a 5-tuple $\langle v_i, \overrightarrow{g_i}, s_i, \beta_i, r_i \rangle$, where $v_i \in V_s$ is the switch for its current access, $\overrightarrow{g_i}$ is a vector to denote the required SFC that consists of a sequence of specific vNFs, $s_i \in V_d$ is the source of the traffic flow, β_i is the bandwidth requirement on a connection between two adjacent vNFs in the SFC, and r_i represents the profit that the service provider gains after serving this request. Hence, to serve the request, the service provider needs to secure all the required vNFs in $\overrightarrow{g_i}$ by either deploying new ones or reusing the existing ones that are not fully loaded, find a proper routing path that goes through the required vNFs in correct order, and reserve enough bandwidth on the path to satisfy the bandwidth requirement.

C. Dynamic SFC Deployment and Readjustment

We consider a periodical service provisioning scenario in which the service provider can serve new requests or change the provisioning schemes of in-service requests at time instants with a fixed interval in between. Hence, at each service time, the service provider needs to handle two kinds of users. We use Γ_1 to represent the set of new users that are at the first time to access the network, while the in-service users are denoted with set Γ_2. We have $\Gamma = \Gamma_1 \cup \Gamma_2$ as the whole set of users to handle. In the service provisioning, the operational overhead of the service provider is mainly from allocating new users to their required vNFs and migrating the services of in-service users to vNFs at new locations. Hence, for each user in Γ, we count how many vNFs the service provider operates for it, and use the total number as its operational

[1]In this work, we assume that the bandwidth capacity of the link between a DC and its local switch would never be used up.
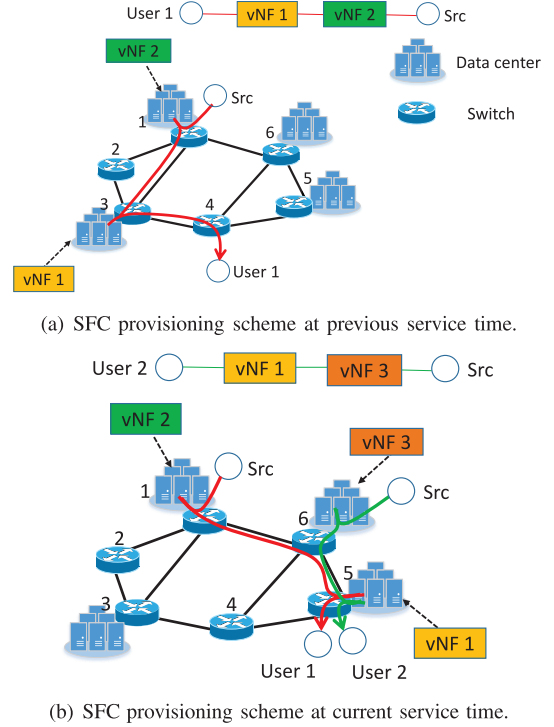

(a) SFC provisioning scheme at previous service time.

(b) SFC provisioning scheme at current service time.

Fig. 1. Example on dynamic SFC deployment and readjustment.

overhead. More specifically, for a new user in Γ_1, the operational overhead equals the number of required vNFs in its SFC, while for an in-service user in Γ_2, the operational overhead is the number of migrated and newly added vNFs since the last service time.

The objective of dynamic SFC deployment and readjustment is to maximize the service provider's profit, which is equal to the total profit from the served requests minus the total deployment cost. The major constraints are those on IT and bandwidth resources and operational overhead. It should be noted that certain requests may be blocked due to the lack of resources. To maximize the profit, the service provider needs to utilize both the IT and bandwidth resources properly, *i.e.*, deploying vNFs adaptively in the DCs, and setting up routing paths intelligently to connect the vNFs for formulating the required SFCs for users.

Fig. 1 shows an intuitive example on the dynamic SFC deployment. Here, for simplicity, we omit the wireless access points in the figure and assume that each switch can be used as an access point (*i.e.*, wired or wireless) for the users. Fig. 1(a) shows the SFC deployment scheme at the previous service time, where the in-service *User* 1 accesses the network from *Switch* 4 and takes an SFC as *vNF* 1→*vNF* 2. Hence, to set up the SFC for *User* 1, the service provider deploys a *vNF* 1 and a *vNF* 2 on *DCs* 3 and 1, respectively. Then, at the current service time, the service provider finds that a new *User* 2 joins at *Switch* 5 to request an SFC as *vNF* 1→*vNF* 3 and the in-service *User* 1 changes its access point to *Switch* 5 but its SFC stays unchanged. Hence, as shown in Fig. 1(b), the service provider decides to migrate the *vNF* 1 to *DC* 5 where it can be efficiently shared by *Users* 1 and 2, and deploys a *vNF* 3 on *DC* 6 for *User* 2.

IV. ILP FORMULATION

In this section, we formulate a path-based ILP model to solve the problem of dynamic SFC deployment and readjustment exactly.

Parameters:

- $G(V, E)$: Network topology, where $V = V_d \bigcup V_s$ and V_d and V_s are for the sets of DCs and switches, respectively.
- C_v: IT resource capacity of a DC $v \in V_d$.
- B_e: Bandwidth resource capacity of a link $e \in E$.
- Π: Set of pre-calculated paths in $G(V, E)$.
  - $\Pi_{u,v}$: Set of K shortest paths between u and v in $G(V, E)$, where $u, v \in V$ and $\Pi_{u,v} \subset \Pi$.
  - π: A path in Π, $s(\pi)$ and $d(\pi)$ represent the source and destination of π, respectively, and $len(\pi)$ denotes the hop-count of π.
  - $\delta_{e,\pi}$: A boolean constant to indicate whether a path π uses a link e. If yes, we have $\delta_{e,\pi} = 1$, and 0 otherwise.
- H_{max}: Maximum operational overhead that the service provider can take at each service time.
- $\mathcal{P}$: Set of supported vNF types in the network.
  - n_p: Maximum number of users that a p-th type vNF can serve ($p \in \mathcal{P}$).
  - c_p: IT resource consumption of a p-th type vNF.
- Γ: Set of users to handle at the current service time, which includes both the new ones in Γ_1 and the in-service ones in Γ_2. We use $|\Gamma|$ to denote the total number of users.
- $\langle v_i, \overrightarrow{g_i}, s_i, \beta_i, r_i \rangle$: A 5-tuple to represent the service request of a user i at current service time.
  - v_i: Access switch of a user i, $v_i \in V_s$.
  - $\overrightarrow{g_i}$: Required SFC of a user i, which is a vector that consists of a sequence of specific vNFs.
    * L_i: Number of vNFs in the SFC of a user i. For convenience, we treat the access switch of a user as a dummy vNF that consumes 0 IT resource, i.e., the first vNF in each user's SFC is its access switch while the other ones are real vNFs. Hence, we have $L_i = |\overrightarrow{g_i}| + 1$.
    * $g_{i,j,p}$: A boolean constant that equals 1 if the j-th vNF on the SFC of a user i is a p-th type vNF, and 0 otherwise. Since the first vNF is the user's access switch, we have $g_{i,1,p} = 0$, $\forall i, p$. We also assume that there are no duplicated vNFs in a user's SFC, and thus if $j_1 \neq j_2$, we have $g_{i,j_1,p} + g_{i,j_2,p} \leqslant 1$, $\forall i, p$.
  - s_i: The source of the required traffic flow,[2] $s_i \in V_d$.
  - β_i: Bandwidth requirement on a connection between two adjacent vNFs in the SFC of a user i.
  - r_i: Profit that the service provider gains after provisioning the SFC of a user i.
- $\langle d_{i,j,p}, a_{i,j,v} \rangle$: A 2-tuple to represent the SFC provisioning scheme of an in-service user i before current service time.
  - $d_{i,j,p}$: A boolean constant to represent the required SFC of an in-service user i before current service time. It equals 1 if the j-th vNF on the SFC of a user

i is a p-th type vNF, and 0 otherwise. It corresponds to parameter $g_{i,j,p}$ described above for the current service time. It should be noted that $d_{i,j,p}$ might not equal $g_{i,j,p}$ since vNFs might be added in or deleted from an SFC due to the dynamic operation.
  - $a_{i,j,v}$: A boolean constant to indicate that for an in-service user i, whether the j-th vNF in its SFC was deployed on node v before current service time. If yes, we have $a_{i,j,v} = 1$, and 0 otherwise. This parameter corresponds to variable $z_{i,j,v}$ below for current service time.
  - $f_{i,v,p}$: For convenience, we integrate $d_{i,j,p}$ and $a_{i,j,v}$ to obtain another parameter $f_{i,v,p}$ to describe whether an in-service user i used the p-th vNF on DC v before the current service time, i.e., $f_{i,v,p} = \sum_j a_{i,j,v} \cdot d_{i,j,p}$.

Variables:

- x_i: A boolean variable that equals 1 if the SFC of a user i is served at current service time, and 0 otherwise.
- $y_{v,p}$: An integer variable to indicate the number of p-th type vNFs that are deployed on a DC $v \in V_d$.
- $z_{i,j,v}$: A boolean variable that equals 1 if the j-th vNF of a user i is deployed on a DC $v \in V_d$, and 0 otherwise.
- $w_{i,j,\pi}$: A boolean variable that equals 1 if for a user i, the connection from its j-th vNF to $(j+1)$-th vNF (i.e., $j \in [1, L_i - 1]$) uses the path π, and 0 otherwise.

Constraints:

1) *vNF Assignment and Readjustment Constraints:*

$$\sum_{v \in V_d} z_{i,j,v} = x_i, \quad \forall j \in [2, L_i], \ \forall i \in \Gamma. \tag{1}$$

Eq. (1) ensures that if a user i is served at the current service time, the required vNFs in its SFC are all realized on the DCs.

$$z_{i,1,v_i} = x_i, \quad \forall i \in \Gamma, \tag{2}$$
$$z_{i,L_i,s_i} = x_i, \quad \forall i \in \Gamma. \tag{3}$$

Eqs. (2)-(3) ensure that if a user i is served at the current service time, its first vNF is a dummy one that is allocated on its access switch while its last vNF represents the source of the traffic flow s_i.

$$\sum_i \sum_j g_{i,j,p} \cdot z_{i,j,v} \leq n_p \cdot y_{v,p}, \quad \forall v \in V_d, \ \forall p \in \mathcal{P}. \tag{4}$$

Eq. (4) ensures that the number of users that use any type of vNFs deployed in a DC v cannot exceed their total capacity.

$$\sum_p c_p \cdot y_{v,p} \leq C_v, \quad \forall v \in V_d. \tag{5}$$

Eq. (5) ensures that the vNFs deployed on a DC v cannot use more IT resources than its IT resource capacity.

$$\sum_{v \in V_d} \left\{ \sum_{i \in \Gamma_1} \sum_{j=2}^{L_i} z_{i,j,v} + \sum_{i \in \Gamma_2} \sum_{j=2}^{L_i} \left[\sum_p \left(1 - f_{i,v,p}\right) \cdot g_{i,j,p} \right] \cdot z_{i,j,v} \right\}$$
$$\leq H_{max}. \tag{6}$$

Eq. (6) ensures that the service provider's operational overhead at the current service time would not exceed the preset upper-limit H_{max}. Here, the first term corresponds to the overhead to handle the new users while the second one is for the overhead

---

[2]In this work, we assume that for each user's SFC, the source of the traffic flow s_i is fixed and will not move during the service provisioning.

to process the in-service ones. We use $\sum_p (1 - f_{i,v,p}) \cdot g_{i,j,p}$ to determine whether the j-th vNF on the SFC of a user i has been readjusted or not at the current service time.

2) *Routing and Bandwidth Allocation Constraints:*

$$\sum_i \sum_j \sum_{\pi \in \Pi} \delta_{e,\pi} \cdot \beta_i \cdot w_{i,j,\pi} \leq B_e, \quad \forall e \in E. \tag{7}$$

Eq. (7) ensures that the total bandwidth consumption on each link $e \in E$ would not exceed its bandwidth capacity.

$$\sum_\pi w_{i,j,\pi} = x_i, \quad \forall j \in [1, L_i - 1], \ \forall i \in \Gamma. \tag{8}$$

Eq. (8) ensures that for each served user i at the current service time, each of its connections between two adjacent vNFs takes one and only one path in the network.

$$z_{i,j,v} = \sum_{s(\pi)=v} w_{i,j,\pi}, \quad \forall i \in \Gamma, \ \forall v \in V, \ \forall j \in [1, L_i - 1], \tag{9}$$

$$z_{i,(j+1),v} = \sum_{d(\pi)=v} w_{i,j,\pi}, \quad \forall i \in \Gamma, \ \forall v \in V, \ \forall j \in [1, L_i - 1]. \tag{10}$$

Eqs. (9) and (10) ensure that if a path π is selected to carry the connection between the j-th vNF and the $(j+1)$-th vNF of a user i, the j-th and $(j+1)$-th vNFs are deployed on $s(\pi)$ and $d(\pi)$, respectively.

Objective: The optimization objective is to maximize the service provider's profit, which is the total profit from the served requests minus the total deployment cost. The deployment cost can be described as follows.

$$\mathbb{C} = \rho_1 \cdot \sum_{v \in V_d} \sum_p c_p \cdot y_{v,p}$$
$$+ \rho_2 \cdot \sum_i \sum_{j=1}^{L_i - 1} \sum_\pi \beta_i \cdot len(\pi) \cdot w_{i,j,\pi}, \tag{11}$$

where ρ_1 and ρ_2 are the positive coefficients to quantify the unit costs of IT and bandwidth resource consumptions, respectively. The profit can be calculated as

$$\mathbb{R} = \sum_i r_i \cdot x_i, \tag{12}$$

and we finalize the optimization objective as

$$Maximize \quad \mathbb{R} - \mathbb{C}. \tag{13}$$

Theorem 1: The optimization described by the aforementioned ILP model for dynamic SFC deployment and readjustment is $\mathcal{NP}$-hard.

Proof: Firstly, we apply the restrictions that $B_e = +\infty$ and $\rho_2 = 0$, which means that the bandwidth constraint can be ignored in the optimization. Secondly, we apply the restriction that both the number of supported vNF types in the network and the number of vNFs in the SFC of each user are 1. Lastly, we apply the restriction that $H_{max} = +\infty$. Then, the problem is transformed into the capacitated facility location problem, which is known to be $\mathcal{NP}$-hard [40]. Hence, as the restricted case of the optimization is the general case of a known $\mathcal{NP}$-hard problem, we prove that the problem of dynamic SFC deployment and readjustment is $\mathcal{NP}$-hard as well [41]. ∎

V. COLUMN GENERATION BASED APPROACH

In this section, we leverage column generation (CG) [18], [19] to optimize the dynamic SFC deployment and readjustment and develop a time-efficient algorithm based on it.

A. General Procedure of CG-Based Approach

Note that, a solution to our problem is just a combination of certain feasible SFC provisioning schemes of the users in Γ. Therefore, if we denote a feasible SFC provisioning scheme of a user as a column c, we can leverage the idea of CG to obtain the set C that includes the proper columns to construct a near-optimal solution. It can be seen that *Algorithm* 1 follows the general operational principle of CG, *i.e.,* decomposing the original problem into a master problem and a pricing problem and then solving them in iterations to obtain the near-optimal solution to the original problem. *Lines* 1-7 are for the initialization. Here, *Line* 1 defines all the notes (*i.e.,* constants) needed to represent a column c, *i.e.,* a feasible SFC provisioning scheme of a user. For the master and pricing problems, *Lines* 2-3 formulate two ILP models, as ILP-MP and ILP-PP, respectively. In *Line* 4, the set C is used to record the columns that we get in the iterations. Note that, as we will explain later, a column c for which all the parameters in the notes are 0 corresponds to the SFC provisioning scheme that none of the users would be served, which is trivially a feasible solution to the original problem. Hence, we use it as the initial solution in *Line* 5 and insert it in C in *Line* 6. Then, *Lines* 7-20 just use the general procedure of CG to solve the problem. The pricing problem's outcome (*i.e.,* $\mathbb{Q}$ in *Line* 11) determines whether an improvement can still be made with more iterations. Specifically, $\mathbb{Q} \geq 0$ means that the objective of the original problem cannot be reduced anymore, and thus *Line* 13 would break the while-loop under such circumstance.

B. Problem Models for Column Generation

We design a CG model that uses the following notes to represent a column c, *i.e.,* a feasible SFC provisioning scheme of a user based on the current network status.

Notes:
- $m_{i,c}$: If c is a provisioning scheme of user i, we have $m_{i,c} = 1$ and 0 otherwise.
- $a_{c,v,p}$: If c allocates a p-th type vNF on DC v, we have $a_{c,v,p} = 1$ and 0 otherwise.
- h_c: The operational overhead that the service provider would take to serve the user i that is selected in c.
- $b_{c,e}$: The bandwidth consumption on each used link e for the provisioning scheme in c.

1) Master Problem: Basically, in order to limit the computational complexity, we design the master problem (*i.e.,* ILP-MP) to perform the optimization based on the solution space that only contains the current obtained columns, *i.e.,* C.

Algorithm 1: General Procedure of CG-Based Approach

1 define the notes to represent a column c;
2 formulate the ILP model for master problem (ILP-MP);
3 formulate the ILP model for pricing problem (ILP-PP);
4 $C = \emptyset$;
5 generate a column c for ILP-MP that all the parameters in the notes are 0;
6 insert c into C to obtain the initial solution;
7 use C to construct the LP relaxation of ILP-MP;
8 **while** *TRUE* **do**
9   solve the LP relaxation of ILP-MP to get the values of primal and dual variables;
10   construct ILP-PP with the results from the LP relaxation of ILP-MP;
11   solve ILP-PP to get its optimization objective $\mathbb{Q}$;
12   **if** $\mathbb{Q} \geq 0$ **then**
13   **break**;
14   **end**
15   generate a new column c with the results of ILP-PP;
16   insert c into C;
17   use C to update the LP relaxation of ILP-MP;
18 **end**
19 use C to construct ILP-MP;
20 solve ILP-MP to obtain the final solution;

Variables:
- λ_c: A boolean variable to indicate whether a column $c \in C$ is selected. If yes, we have $\lambda_c = 1$, and 0 otherwise.
- $y_{v,p}$: An integer variable to indicate the number of p-th type vNFs that are deployed on a DC $v \in V_d$.

Objective: For convenience, we add a minus sign to the objective in Eq. (13) to obtain the equivalent form of minimization. As we only consider the provisioning schemes included in C in the master problem, the objective of the dynamic SFC deployment and readjustment is transformed into

$$Minimize \sum_{c \in C} \lambda_c \cdot \left(\rho_2 \cdot \sum_e b_{c,e} - \sum_i r_i \cdot m_{i,c} \right) + \rho_1 \cdot \sum_{v \in V_d} \sum_p c_p \cdot y_{v,p}. \tag{14}$$

Constraints: The constraints get changed as follows. Note that, we also list the corresponding dual variables here in "()", which indicate the reduced cost on the objective in Eq. (14). For these inequations with "$\leq$", we use negative dual variables to ensure that the values of the dual variables are not smaller than 0.

$$\sum_c h_c \cdot \lambda_c \leq H_{max}, (-\varphi), \tag{15}$$

$$\sum_p c_p \cdot y_{v,p} \leq C_v, \quad \forall v \in V_d, \quad (-\tau_v), \tag{16}$$

$$\sum_c b_{c,e} \cdot \lambda_c \leq B_e, \quad \forall e \in E, \quad (-\alpha_e). \tag{17}$$

Eqs. (15)-(17) ensure that the constraints on operational overhead, IT resources and bandwidth resources are satisfied.

$$\sum_c m_{i,c} \cdot \lambda_c \leq 1, \quad \forall i \in \Gamma, (-\gamma_i), \tag{18}$$

$$\sum_c \lambda_c \leq |\Gamma|, (-\eta). \tag{19}$$

Eqs. (18), (19) ensure that for each user i, at most one column can be selected.

$$\sum_c a_{c,v,p} \cdot \lambda_c \leq n_p \cdot y_{v,p}, \quad \forall v \in V_d, \ \forall p \in \mathcal{P}, (-\chi_{v,p}). \tag{20}$$

Eq. (20) ensures that for each p-th type vNF on a DC v, the number of users that use it cannot exceed its capacity.

2) Pricing Problem: As explained in *Algorithm* 1, by solving the pricing problem (*i.e.,* ILP-PP) in each iteration, we try to reduce the objective in Eq. (14). If this can be done, the objective $\mathbb{Q}$ of the pricing problem is negative, and we generate a new column with the results of ILP-PP and include it in C. Otherwise, CG cannot improve the quality of the solution to the original problem anymore and we terminate the iterations. The ILP-PP is formulated as follows.

Variables:
- $x_i, z_{i,j,v}, w_{i,j,\pi}$: Variables that have the same definitions as those in Section IV.

The relations between these variables and the notes $m_{i,c}, a_{c,v,p}, h_c$ and $b_{c,e}$ defined for a specific column c are as follows.

$$m_{i,c} = x_i, \quad \forall i \in \Gamma, \tag{21}$$

which is because in each iteration, we only consider one column that corresponds to the provisioning scheme of a user.

$$h_c = \sum_{i \in \Gamma_1} \sum_{j=2}^{L_i} \sum_{v \in V_d} z_{i,j,v}$$
$$+ \sum_{i \in \Gamma_2} \sum_{j=2}^{L_i} \sum_{v \in V_d} \left[\sum_p (1 - f_{i,v,p}) \cdot g_{i,j,p} \right] \cdot z_{i,j,v}, \tag{22}$$

which is because h_c denotes the total operational overhead.

$$a_{c,v,p} = \sum_i \sum_j g_{i,j,p} \cdot z_{i,j,v}, \quad \forall v \in V_d, \ \forall p \in \mathcal{P}, \tag{23}$$

where the item $g_{i,j,p} \cdot z_{i,j,v}$ indicates whether the j-th vNF on the SFC of a user i is assigned to a p-th type vNF on the DC v. Hence, with a summation, we can determine whether a p-th type vNF on the DC v is used in the column c.

$$b_{c,e} = \sum_i \sum_j \sum_\pi \beta_i \cdot \delta_{e,\pi} \cdot w_{i,j,\pi}, \quad \forall e \in E, \tag{24}$$

which gets the bandwidth usage on link e. Note that, based on the relation between the primal and dual problems [42], the reduced cost of λ_c has the expression as

$$\sum_i m_{i,c} \cdot (\gamma_i - r_i) + \sum_e b_{c,e} \cdot (\rho_2 + \alpha_e) + \sum_{v \in V_d} \sum_p \chi_{v,p} \cdot a_{c,v,p}$$
$$+ \varphi \cdot h_c + \eta. \tag{25}$$

Objective: Then, we put Eqs. (21)-(24) into Eq. (25) and get the objective of the pricing problem as

$$Minimize \quad \mathbb{Q} = \sum_{i,j,\pi} \zeta_{i,j,\pi} \cdot w_{i,j,\pi} + \sum_{i \in \Gamma_1, j, v} \tilde{\zeta}_{i,j,v} \cdot z_{i,j,v}$$
$$+ \sum_{i \in \Gamma_2, j, v} \hat{\zeta}_{i,j,v} \cdot z_{i,j,v} + \sum_i \zeta_i \cdot x_i + \eta, \quad (26)$$

where we use the following notations to simplify the objective:

$$\begin{cases} \zeta_{i,j,\pi} = \beta_i \cdot \sum_e (\rho_2 + \alpha_e) \cdot \delta_{e,\pi}, \\ \tilde{\zeta}_{i,j,v} = \varphi + \sum_p \chi_{v,p} \cdot g_{i,j,p}, \\ \hat{\zeta}_{i,j,v} = \sum_p [(1 - f_{i,v,p}) \cdot \varphi + \chi_{v,p}] \cdot g_{i,j,p}, \\ \zeta_i = \gamma_i - r_i. \end{cases} \quad (27)$$

Constraints: The ILP-PP inherits the constraints in Eqs. (1)-(3) and (8)-(10) defined in Section IV. Moreover, we introduce the following constraints to expedite the solving of the ILP-PP.

$$\sum_i x_i = 1. \quad (28)$$

Eq. (28) ensures that the pricing problem only consider the service provisioning of one user.

$$\sum_i \sum_j \left(\sum_p c_p \cdot g_{i,j,p} \right) \cdot z_{i,j,v} \leq C_v, \quad \forall u \in V_d,$$
$$\sum_i \sum_j \sum_\pi \delta_{e,\pi} \cdot \beta_i \cdot w_{i,j,\pi} \leq B_e, \quad \forall e \in E. \quad (29)$$

Eq. (29) ensures that in each obtained column, the capacity constraints on IT and bandwidth resources are satisfied, *i.e.,* the obtained column is a feasible solution. Hence, the ILP-MP can be constructed correctly in *Line* 19 of *Algorithm* 1.

3) Accelerating the Solving of Pricing Problem: It should be noted that in *Algorithm* 1, the pricing problem to be solved in *Line* 11 is still an ILP, which consumes most of the time used for solving the problem with the CG procedure. Meanwhile, according to the basic idea of CG, we actually do not need to obtain the minimum solution of the pricing problem in each iteration. Hence, we can design a heuristic to accelerate the problem-solving. *Algorithm* 2 shows the detailed procedure of the heuristic. To incorporate *Algorithm* 2 in our CG model, we use the procedure in *Algorithm* 3 to replace *Lines* 10-15 in *Algorithm* 1. Basically, in each iteration, we first try to achieve a reduced cost with *Algorithm* 3, and will only try to solve the ILP-PP when *Algorithm* 3 cannot provide a column with negative reduced cost anymore (*i.e.,* $\mathbb{Q} \geq 0$).

VI. PERFORMANCE EVALUATION AND DISCUSSION

In this section, we use numerical simulations to evaluate the performance of the proposed approaches for dynamic SFC deployment and readjustment problem.

A. Simulation Setup

The simulations use two network topologies, *i.e.,* a small six-node one as shown in Fig. 1 and a practical NSFNET

Algorithm 2: Heuristic to Solve ILP-PP

1 assign the weight of each link $e \in E$ as $\rho_2 + \alpha_e$ according to Eq. (27);
2 calculate the weighted shortest paths between all the node pairs in the network;
3 **for** *each user $i \in \Gamma$* **do**
4   **for** *each vNF $j \in [2, L_i]$* **do**
5   use the greedy strategy to find the DC v for this vNF such that $\mathbb{Q}$ in Eq. (26) is minimized except the L_i-th vNF which is fixed;
6   connect the $(j-1)$-th and j-th vNFs of user i with the weighted shortest path between them;
7   calculate the incremental cost as $Q_{i,j}$;
8   **end**
9   get the overall cost of user i as $Q_i = \sum_j Q_{i,j}$;
10 **end**
11 $\mathbb{Q} = \min_{i \in \Gamma}(Q_i)$;

Algorithm 3: Sub-Procedure to Accelerate CG

1 construct pricing problem with the results from the LP relaxation of ILP-MP;
2 solve the pricing problem with *Algorithm* 2 to get $\mathbb{Q}$;
3 **if** $\mathbb{Q} \geq 0$ **then**
4   construct ILP-PP with the results from the LP relaxation of ILP-MP;
5   solve ILP-PP to get its optimization objective $\mathbb{Q}$;
6   **if** $\mathbb{Q} \geq 0$ **then**
7   **break**;
8   **end**
9 **end**
10 generate a new column c with results of pricing problem;

---

topology [43]. Besides, in order to reduce the time complexity, we use the 5-shortest paths between two nodes in the NSFNET topology to replace the total paths when using the proposed-CG model. Here, we assume that the DCs are all light-weighted ones (*i.e.,* cloudlet) and they are randomly distributed in the topologies, which means not all the switches have local DCs. The DCs' IT resource capacities are uniformly distributed within [30, 35] units,[3] while the bandwidth capacities of each physical links are randomly selected within [25, 30] units. Note that, the upper-limit on the service provider's operational overhead (*i.e.,* H_{max}) is determined based on the network scale and traffic load. Specifically, it is set within [30, 60] for each service time. We treat the costs of IT and bandwidth resources equally in the optimization, and thus the simulations use $\rho_1 = \rho_2 = 1$ in the objective.

The network with the six-node topology can support 4 vNF types, while the one with the NSFNET topology can support 5 vNF types. For each vNF type, the IT resource consumption

---

[3]Note that, in order to normalize the related costs and make the simulation scenarios more generic, we use "units" instead of specific metrics, *e.g.,* CPU cycles, Mbps, to quantify IT and bandwidth resources in the simulations.

of a vNF is within [5, 8] units and each can serve 3 to 5 users at most. The number of vNFs requested on the SFC by each user is assumed to be uniformly distributed within [2, 4] and [2, 5] for the six-node topology and the NSFNET topology, respectively. Meanwhile, the bandwidth requirement on links is within [3, 5] units. To obtain sufficient statistical accuracy, we obtain each data point by averaging the results from 20 independent simulations. The simulations are carried out on a computer with 3.4 GHz Inter Core i3-3240 and 8 GB RAM, and we use MATLAB 2013a with the CPLEX toolbox to solve the ILP and CG models.

In order to compare the proposed algorithms to an existing benchmark, we modify the algorithm developed in [28] to obtain the benchmark algorithm (*i.e.,* Viterbi). Specifically, Viterbi computes the provisioning cost by constructing a multi-stage graph for each user, sorts the users according to their costs in descending order, and then serves them one by one using the logic of the Viterbi algorithm in [28].

B. Effects of Overhead-Restricted Policy

Note that, to limit the service provider's operational overhead, we apply a restriction on it with H_{max}. We first demonstrate that it is necessary to do so by comparing the proposed overhead-restricted approach with one that does have the restriction of H_{max} (*i.e.,* an overhead-unrestricted approach). Specifically, we use the six-node topology and consider the provisioning at a service time for different numbers of users, some of which might have changed their access switches or SFCs. The ILP model in Section IV is used to obtain the optimal solutions for the overhead-restricted/unrestricted approaches. Basically, for the overhead-unrestricted approach, the ILP sets $H_{max} = +\infty$ to ignore the limitation on operational overhead while orchestrating the SFCs.

Fig. 2 shows the simulation results. The results on service provider's profit are plotted in Fig. 2(a). It can be seen that the service provider may obtain about 25% more profit with the overhead-unrestricted approach. This is because when the restriction of H_{max} is relaxed, the service provider would have more freedom to deploy the SFCs with reduced resource costs. Meanwhile, it is interesting to notice that when the number of user requests is 7, the two approaches obtain equal profit. This is due to the fact that when the number of user requests is relatively small, the network system can afford the operational overhead to optimally deploy each user request's SFC even when the restriction of H_{max} is considered. When the number of user requests becomes larger, the operations of vNF deployment/reassignment should be carefully considered to maintain an acceptable operational overhead. This analysis can be verified with the results in Fig. 2(b), which are the operational overheads of the overhead-restricted/unrestricted approaches. We can see that the operational overhead of the overhead-unrestricted approach is significantly higher than that of the overhead-restricted one, when the number of user requests is 28 or more. This reveals the fact that the operational overhead may become a serious issue for the overhead-unrestricted approach. Hence, we should try to balance the tradeoff
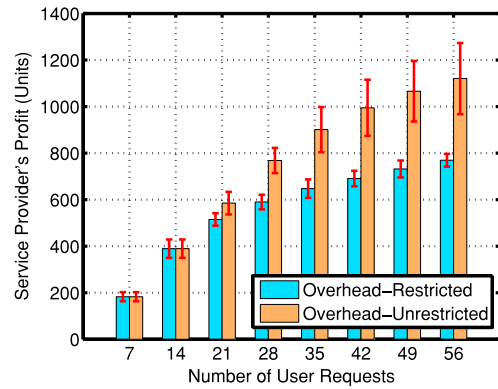
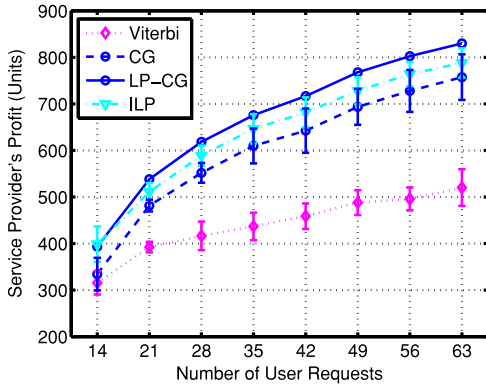(a) Service Provider's Profit
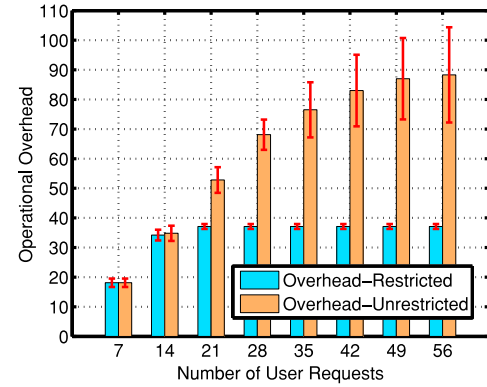
(b) Operational Overhead

Fig. 2. Comparison of overhead-restricted and overhead-unrestricted approaches.

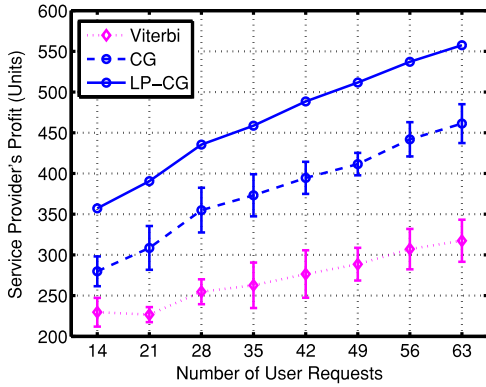between resource consumption and operational overhead in dynamic SFC deployment and readjustment.

C. One-Time Operation

Next, to evaluate the performance of our proposed CG-based algorithm, we first conduct simulations on one-time operation, which means that we only consider the SFC provisioning at a service time. Here, we assume that $\frac{3}{7}$ of the users are new ones while the rest of them are in-service ones that may have changed their access switches or SFCs since the last service time. We simulate the algorithms in the two topologies, except for the ILP in NSFNET since the time complexity is too high.

Fig. 3 shows the results on the service provider's profit. In Fig. 3(a), we also plot the linear-programming (LP) bound of the CG model (LP-CG), which is the LP relaxation solution of the original problem. We observe that the ILP's solution always lies in between those of LP-CG and CG. This is because the LP relaxation solution only provides an upper-bound on the service provider's profit but might not be a feasible solution to the original problem. Hence, we can use it as a baseline to evaluate the algorithms' performance, when the ILP is intractable (*e.g.,* in the NSFNET topology). Meanwhile, we find that when the number of user requests is less than 14, the algorithms perform similarly, but as the number of user requests increases, the CG's advantage over Viterbi in service provider's profit becomes more and more significant.

(a) Six-node topology

(b) NSFNET topology

Fig. 3.   Results on service provider's profit in one-time operation.

TABLE I
COMPUTATION TIME PER REQUEST WITH
SIX-NODE TOPOLOGY (SECONDS)

# of User Requests	Viterbi	CG	ILP
14	0.0068	0.15	0.071
21	0.0073	0.18	0.12
28	0.0079	0.20	0.27
35	0.0087	0.24	0.37
42	0.090	0.28	0.92
49	0.093	0.30	1.45
56	0.011	0.34	2.86
63	0.015	0.39	5.52

TABLE II
COMPUTATION TIME PER REQUEST WITH
NSFNET TOPOLOGY (SECONDS)

# of User Requests	Viterbi	CG
14	0.053	0.51
21	0.055	1.35
28	0.056	1.58
35	0.058	2.05
42	0.063	2.52
49	0.072	3.07
56	0.077	4.23
63	0.084	6.28

This is because as a heuristic algorithm, Viterbi can be easily trapped by local optima when the problem's scale becomes larger and thus it cannot follow the increasing trend of the ILP as CG does. Note that, the superior performance of the ILP and CG on service provider's profit is actually obtained at the cost of increased time complexity. The performance of Viterbi and CG is also compared with simulations using the NSFNET topology, and the results are shown in Fig. 3(b), which exhibit the similar trend as that in Fig. 3(a). This suggests that CG still performs well in a network whose topology is practical and larger than the six-node one.

Tables I and II list the computation time per request of the algorithms when using the six-node and NSFNET topologies, respectively. Since Viterbi does not use the iterative method, it runs significantly faster than CG and ILP, which confirms our analysis that CG obtains higher profits for the service provider at the cost of increased time complexity. Therefore, when the performance gap on service provider's profit between Viterbi and CG is not large (*e.g.,* the number of user requests is 14 in Figs. 3(a) and 3(b)), we should just use Viterbi. It is interesting to notice that the computation time of ILP is less than or comparable to CG when the number of user requests is 28 or less in Table I. This is because CPLEX can solve ILP in one shot while CG approximates the optimal solution with iterations. However, when the number of user requests keeps increasing, the computation time of the ILP increases much faster than that of CG and becomes one magnitude longer when there

are 56 user requests to handle. The results in Table II show the same trend as those in Table I. Since the NSFNET topology is larger, CG needs more computation time to converge. This time, however, the computation time of CG might be too long for practical network operation, especially in a dynamic network environment. Fortunately, the running time of CG can still be reduced from two perspectives. Firstly, by switching the software platform from MATLAB to C/C++, we can make the implementation of CG more time-efficient. Secondly, the computation time could be reduced if we use a more powerful computing platform other than a personal computer, as in the common case of a practical NC&M system.
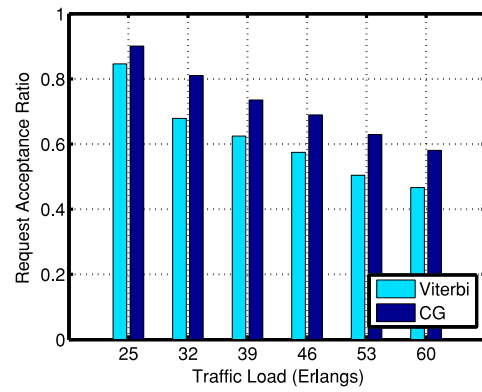
D. Dynamic Operation

Finally, we consider the scenario in which dynamic operations are performed over a period of time and at each service time, new users can join in and in-service users can change their SFCs. The simulations use the NSFNET topology and the dynamic user requests are generated by leveraging the self-similar traffic model in [44]. To evaluate the effect of the restriction on operational overhead, we consider the low ($H_{max} = 35$) and high ($H_{max} = 60$) overhead scenarios.
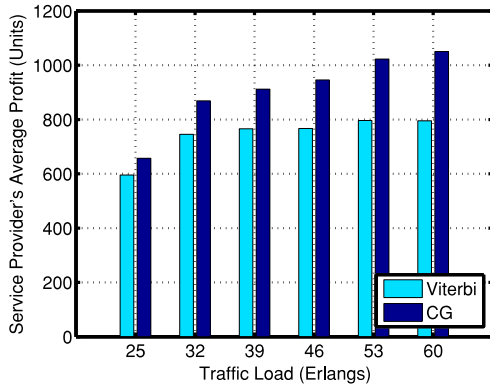
Fig. 4 shows the results on service provider's average profit per service time, which indicates that our proposed CG-based approach can constantly outperform Viterbi for both scenarios. When the traffic load increases, the service provider's profit also increases but its momentum toward increasing becomes less. Note that, in the high overhead scenario, since there are more freedom for the algorithms to readjust the SFC deployments, the service provider can get higher profit than in the
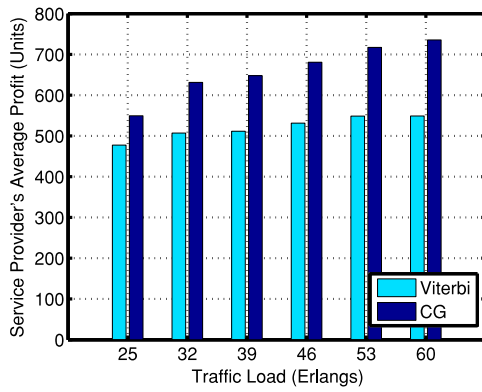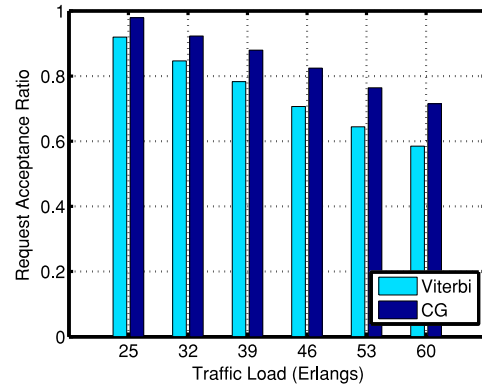
(a) Low overhead scenario

(b) High overhead scenario

Fig. 4. Results on service provider's average profit in dynamic operation.

(a) Low overhead scenario

(b) High overhead scenario

Fig. 5. Results on request acceptance ratio in dynamic operation.

TABLE III
AVERAGE PROFIT RATIO BETWEEN CG AND VITERBI

Traffic Load (Erlangs)	25	32	39	46	53	60
Low Overhead	1.15	1.25	1.27	1.28	1.31	1.34
High Overhead	1.10	1.17	1.19	1.23	1.28	1.32

low overhead scenario. Then, we compare the algorithms in terms of request acceptance ratio, which is defined as the ratio of total served time to total requested time of the user requests. As indicated in Fig. 5, the request acceptance ratios from both algorithms decrease with the traffic load. This is because the network system becomes more saturated when the traffic load is higher. Due to its superior performance, CG constantly accepts more user requests than Viterbi, which suggests that in dynamic operation, CG not only serves each user request with less resource costs but also serve more requests successfully. The results on the ratio of service provider's average profit per service time between CG and Viterbi are summarized in Table III. It can be seen that when the traffic load is the same, the performance gain of CG over Viterbi is higher in the low overhead scenario. This suggests that when the restriction on operational overhead is tighter, CG's advantage over Viterbi for dynamic SFC deployment and readjustment becomes more significant, which further verifies the effectiveness of our proposed algorithm. Meanwhile, we would like to point out that the superior performance of CG is obtained at the cost of increased time complexity.

VII. CONCLUSION

In this paper, we studied the problem of SFC service provisioning by taking the dynamic nature of user requests into consideration. We first formulated a path-based ILP model to solve the problem exactly. Then, to reduce the time complexity, we designed a CG model, developed an approximation algorithm based on it and proposed an effective heuristic to further accelerate the problem-solving. Simulation results showed that our CG-based approach significantly outperformed the benchmark algorithm in terms of the service provider's profit and request acceptance ratio.

REFERENCES

[1] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, and G. Shi, "Design and implementation of a consolidated middlebox architecture," in *Proc. NSDI*, San Jose, CA, USA, Apr. 2012, pp. 323–336.

[2] P. Lu, Q. Sun, K. Wu, and Z. Zhu, "Distributed online hybrid cloud management for profit-driven multimedia cloud computing," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1297–1308, Aug. 2015.

[3] K. Wu, P. Lu, and Z. Zhu, "Distributed online scheduling and routing of multicast-oriented tasks for profit-driven cloud computing," *IEEE Commun. Lett.*, vol. 20, no. 4, pp. 684–687, Apr. 2016.

[4] P. Lu, L. Zhang, X. Liu, J. Yao, and Z. Zhu, "Highly efficient data migration and backup for big data applications in elastic optical inter-data-center networks," *IEEE Netw.*, vol. 29, no. 5, pp. 36–42, Sep./Oct. 2015.

[5] M. Chiosi *et al.* (2012). *Network Functions Virtualization*. [Online]. Available: https://portal.etsi.org/nfv/nfv_white_paper.pdf

[6] W. Fang, M. Zeng, X. Liu, W. Lu, and Z. Zhu, "Joint spectrum and IT resource allocation for efficient vNF service chaining in inter-datacenter elastic optical networks," *IEEE Commun. Lett.*, vol. 20, no. 8, pp. 1539–1542, Aug. 2016.

[7] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proc. CloudNet*, Oct. 2014, pp. 7–13.

[8] W. Fang, M. Lu, X. Liu, L. Gong, and Z. Zhu, "Joint defragmentation of optical spectrum and IT resources in elastic optical datacenter interconnections," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 7, no. 4, pp. 314–324, Apr. 2015.

[9] H. Moens and F. De Turck, "Customizable function chains: Managing service chain variability in hybrid NFV networks," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 4, pp. 711–724, Dec. 2016.

[10] M. Zeng, W. Fang, J. J. P. C. Rodrigues, and Z. Zhu, "Orchestrating multicast-oriented NFV trees in inter-DC elastic optical networks," in *Proc. ICC*, Kuala Lumpur, Malaysia, Jun. 2016, pp. 1–6.

[11] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for NFV chaining in packet/optical datacenters," *J. Lightw. Technol.*, vol. 33, no. 8, pp. 1565–1570, Apr. 15, 2015.

[12] Q. Sun, P. Lu, W. Lu, and Z. Zhu, "Forecast-assisted NFV service chain deployment based on affiliation-aware vNF placement," in *Proc. GLOBECOM*, Washington, DC, USA, Dec. 2016, pp. 1–6.

[13] T. Taleb, K. Samdanis, and A. Ksentini, "Supporting highly mobile users in cost-effective decentralized mobile operator networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 7, pp. 3381–3396, Sep. 2014.

[14] M. Abu-Lebdeh, J. Sahoo, R. Glitho, and C. W. Tchouati, "Cloudifying the 3GPP IP multimedia subsystem for 4G and beyond: A survey," *IEEE Commun. Mag.*, vol. 54, no. 1, pp. 91–97, Jan. 2016.

[15] W. Cerroni and F. Callegati, "Live migration of virtual network functions in cloud-based edge networks," in *Proc. ICC*, Sydney, NSW, Australia, Jun. 2014, pp. 2963–2968.

[16] M. Zeng, W. Fang, and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *J. Lightw. Technol.*, vol. 34, no. 14, pp. 3330–3341, Jul. 15, 2016.

[17] Y. Wang, P. Lu, W. Lu, and Z. Zhu, "Cost-efficient virtual network function graph (vNFG) provisioning in multidomain elastic optical networks," *J. Lightw. Technol.*, vol. 35, no. 13, pp. 2712–2723, Jul. 1, 2017.

[18] J. Desrosiers and M. E. Lübbecke, *A Primer in Column Generation*. New York, NY, USA: Springer, 2005.

[19] A. Jarray and A. Karmouch, "Decomposition approaches for virtual network embedding with one-shot node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 23, no. 3, pp. 1012–1025, Jun. 2015.

[20] R. Mijumbi *et al.*, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 236–262, 1st Quart., 2016.

[21] 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.

[22] "Network functions virtualisation (NFV): Use cases," ETSI, Sophia Antipolis, France, Tech. Rep., Oct. 2013. [Online]. Available: http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf

[23] "Service function chaining problem statement," Tech. Rep., Apr. 2015. [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc7498.txt.pdf

[24] J. Martins *et al.*, "ClickOS and the art of network function virtualization," in *Proc. NSDI*, Seattle, WA, USA, Apr. 2014, pp. 459–473.

[25] L. Mamatas, S. Clayman, and A. Galis, "Information exchange management as a service for network function virtualization environments," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 564–577, Sep. 2016.

[26] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *Proc. INFOCOM*, Hong Kong, Apr. 2015, pp. 1346–1354.

[27] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," in *Proc. IFIP/IEEE IM*, Ottawa, ON, Canada, May 2015, pp. 98–106.

[28] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 4, pp. 725–739, Dec. 2016.

[29] X. Chen, L. Sun, Z. Zhu, H. Lu, and S. J. B. Yoo, "On efficient incentive-driven VNF service chain provisioning with mixed-strategy gaming in broker-based EO-IDCNs," in *Proc. OFC*, Los Angeles, CA, USA, Mar. 2017, pp. 1–3.

[30] R. Mijumbi *et al.*, "Topology-aware prediction of virtual network function resource requirements," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 1, pp. 106–120, Mar. 2017.

[31] X. Sun and N. Ansari, "PRIMAL: Profit maximization avatar placement for mobile edge computing," in *Proc. ICC*, Kuala Lumpur, Malaysia, Jun. 2016, pp. 1–6.

[32] M. Ghaznavi *et al.*, "Elastic virtual network function placement," in *Proc. CloudNet*, Niagara Falls, ON, Canada, Oct. 2015, pp. 255–260.

[33] S. Ayoubi, Y. Zhang, and C. Assi, "A reliable embedding framework for elastic virtualized services in the cloud," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 489–503, Sep. 2016.

[34] 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.*, to be published.

[35] M. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 206–219, Feb. 2012.

[36] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightw. Technol.*, vol. 32, no. 3, pp. 450–460, Feb. 1, 2014.

[37] L. Gong, Y. Wen, Z. Zhu, and T. Lee, "Toward profit-seeking virtual network embedding algorithm via global resource capacity," in *Proc. INFOCOM*, Toronto, ON, Canada, Apr. 2014, pp. 1–9.

[38] H. Jiang, Y. Wang, L. Gong, and Z. Zhu, "Availability-aware survivable virtual network embedding in optical datacenter networks," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 7, no. 12, pp. 1160–1171, Dec. 2015.

[39] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding LC-VNE algorithms towards integrated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 24, no. 6, pp. 3648–3661, Dec. 2016.

[40] R. M. Nauss, "An improved algorithm for the capacitated facility location problem," *J. Oper. Res. Soc.*, vol. 29, no. 12, pp. 1195–1201, Dec. 1978.

[41] M. Gary and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman, 1979.

[42] D. Luenberger and Y. Ye, *Linear and Nonlinear Programming*. New York, NY, USA: Springer, 1984.

[43] X. Chen, S. Zhu, L. Jiang, and Z. Zhu, "On spectrum efficient failure-independent path protection p-cycle design in elastic optical networks," *J. Lightw. Technol.*, vol. 33, no. 17, pp. 3719–3729, Sep. 1, 2015.

[44] M. S. Taqqu, W. Willinger, and R. Sherman, "Proof of a fundamental result in self-similar traffic modeling," *SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 2, pp. 5–23, Apr. 1997.

Junjie Liu, photograph and biography not available at the time of publication.

Wei Lu, photograph and biography not available at the time of publication.

Fen Zhou, photograph and biography not available at the time of publication.

Ping Lu, photograph and biography not available at the time of publication.

Zuqing Zhu, photograph and biography not available at the time of publication.