Online Service Function Chain Placement for Cost-effectiveness and Network Congestion Control

Xiaojun Shang, Zhenhua Liu, Yuanyuan Yang
Stony Brook University, Stony Brook, NY 11794

Abstract—The emerging network function virtualization is migrating traditional middleboxes, e.g., firewalls, load balancers, proxies, from dedicated hardware to virtual network functions (VNFs) running on commercial servers defined as network points of presence (N-PoPs). VNFs further chain up for more complex network services called service function chains (SFCs). SFCs introduce new flexibility and scalability which greatly reduce expenses and rolling out time of network services. However, chasing the lowest cost may lead to congestion on popular N-PoPs and links, thus resulting in performance degradation or violation of service-level agreements. To address this problem, we propose a novel scheme that reduces the operating cost and controls network congestion at the same time. It does so by placing VNFs and routing flows among them jointly. Given the problem is NP-hard, we design an approximation algorithm named candidate path selection (CPS) with a theoretical performance guarantee. We then consider cases when SFC demands fluctuate frequently. We propose an online candidate path selection (OCPS) algorithm to handle such cases considering the VNF migration cost. OCPS is designed to preserve good performance under various migration costs and prediction errors. Extensive simulation results highlight that CPS and OCPS algorithms perform better than baselines and comparably to the optimal solution.

Index Terms—cloud computing, virtual network function, service function chain, network congestion, load balancing

1 Introduction

The development of virtual network functions (VNFs) moves network services, e.g., Routers, Firewalls, NATs, from specialized hardware to commercial servers. Clusters of such servers supporting VNFs are called network points of presence (N-PoPs) [1]. For some complex network services, corresponding VNFs are chained together to provide services as a whole called service function chains (SFCs). SFCs may operate on multiple N-PoPs connected by a communication network and can be easily installed, deleted, scaled up and down, updated, or even migrated among N-PoPs. Such features largely reduce the operating cost and rolling out time of network services.

ETSI standard [2] and realized VNF platforms [3], [4] indicate that different VNFs may demand different resources of servers and some may require specific licenses to operate. Therefore, the operating cost of a VNF is affected by the N-PoP picked to support it. This is especially the case for hybrid N-PoPs consisted of both commercial servers and dedicated hardware [5]. Thus, it is not surprising to see some previous work proposes to place VNFs over N-PoPs with the lowest cost to reduce the operating expenditure.

However, the operating cost of VNFs is not the only consideration when determining the placement and routing of SFCs. In particular, the load on each N-PoP and network link should be carefully balanced to eliminate possible congestion. Simply using constraints to limit the maximum load on each N-PoP and link, as most existing work does [6], [7], is not enough. Consider a simple example with two N-PoPs and two VNFs where each VNF requests for half of the resources of one N-PoP. Clearly, placing one VNF on an N-PoP each may bring much better performance compared to placing both VNFs on the same N-PoP, even though both strategies satisfy the constraints. In other words, placing VNFs on N-PoPs with lower operating costs may incur congestion. Therefore, a thoughtful tradeoff between reducing the operating cost and minimizing N-PoP/network congestion is in great need.

It is also worth noting that the placement of SFCs involves not only placement of VNFs but also routing of flows among VNFs. Jointly optimizing VNF placement and flow routing is crucial when considering congestion of both N-PoPs and links. For instance, there exist N-PoPs with sufficient VNF resources but limited bandwidth on some of its connecting links. In this way, placing VNFs on these N-PoPs without jointly considering flow routing can easily aggregate large congestion on their connecting links with limited bandwidth. Existing papers concerning network congestion either focus on only one aspect [3], [8]–[10] or treat VNF placement and flow routing as two separate components [11], which may lead to sub-optimal solutions. Recent work [12] considers a predefined set of routes, which reduces the complexity at the cost of optimality.

With above-mentioned conclusions, the goal of our paper is to *jointly optimize the SFC placement and routing to minimize the operating cost and congestion*. The problem formulated based on this goal is challenging first because the indivisibility of VNFs leads to integer constraints. In addition, jointly placing VNFs and routing corresponding flows incur multiple sets of variables and make the problem even harder. Related work [8] proves the NP-hardness of jointly placing and routing SFCs when balancing the load on links, thus proposing heuristic algorithms to solve it. In this paper, we propose an approximation algorithm named candidate path selection (CPS) to solve the formulated problem in polynomial time with a theoretical performance guarantee to the optimum.

CPS preserves good performance with constant or slow-changing SFC demands. Whenever a significant demand change happens, the algorithm is applied again to update the placement and routing of SFCs. However, when SFC demands fluctuate very fast, the costs of redeploying SFCs, e.g., migrating VNFs, will become comparable to or even higher than the operating costs. In these cases, it is inappropriate to redo CPS upon a demand change without considering the migration cost. In preceding research fields such as virtual network embedding (VNE), VM migration is an exorbitant procedure with high cost and large latency. Fortunately, some realized software-based VNF platforms such as ClickOS make the cost of VNF migration more affordable [4]. As a result, it is possible to pay moderate migration costs to reduce the operating cost and congestion when such migrations bring larger reductions. In this way, the problem becomes an online optimization, and we extend our algorithm into an online version called online candidate path selection (OCPS) to handle the lack of future information. OCPS is designed to perform comparably to the offline optimal solution under different migration costs and prediction errors.

Our main contributions in this paper are summarized as follows.

- We formulate an optimization problem to minimize the operating cost and congestion of SFCs by jointly placing VNFs and routing flows on these SFCs. The problem bridges the operating cost and the network congestion which are separately considered in existing work.
- As the formulated problem is NP-hard, we propose a CPS algorithm to both reduce the computational complexity and achieve an approximation ratio of $\mathcal{O}(\log(|V|))$ to the optimal solution, where $|V|$ is the number of N-PoPs in the network.
- When SFC demands change rapidly, we extend CPS into an OCPS algorithm by optimizing over a time horizon with VNF migrations allowed between consecutive time slots and migration costs added to the objective function. OCPS is specifically designed to handle different scales of VNF migration cost and prediction errors while performing comparably to the offline optimal solution.
- We perform extensive simulations to validate that the CPS algorithm achieves comparable performance to the optimal solution, which outperforms the baselines significantly. We also show that the improvement of the OCPS algorithm is consistent under different VNF migration costs and various prediction errors. When the demands of SFCs follow real traces with fast fluctuation, our OCPS algorithm is shown to maintain comparable performance to the offline optimum.

The remainder of this paper is organized as follows. Section 2 proposes the SFC placement problem and the CPS algorithm. Section 3 presents the online SFC placement problem and the OCPS algorithm designed for fast demand fluctuation. Section 4 shows the performance evaluation of our algorithms. Section 5 briefly reviews the related work and Section 6 concludes this paper.

2 SFC Placement with Slow Demand Fluctuation

One important advantage of SFCs is that their VNFs and corresponding flows can be flexibly placed on different N-PoPs and connecting links in the network to maximize various benefits. Therefore, SFC deployment involves both the placement of VNFs and the routing of flows. In Fig. 1, we present a simplified example of SFC placement. The detailed placement is driven by different factors, e.g., operating cost, network congestion, demand fluctuation, security.

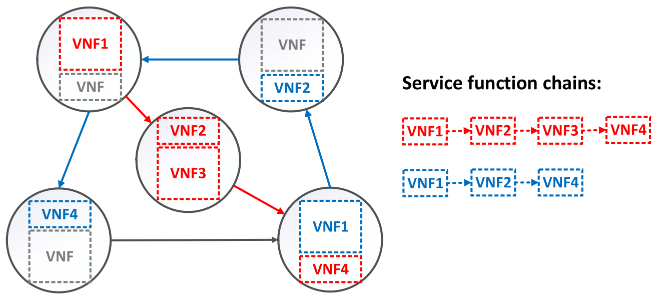


Fig. 1. An example of SFC placement. The red and blue SFCs are deployed onto five N-PoPs. The sizes of VNFs represent different operating costs of VNFs. The colored arrows denote directions of routing between different N-PoPs. Routing paths between two VNFs may consist of multiple N-PoPs and physical links.

In this section, we focus on two major and correlated objectives of the SFC placement problem, i.e., the operating cost and the network congestion. As mentioned in Section 1, pursuing low operating costs may lead to aggregation of load on some N-PoPs and links and thus cause congestion. On the contrary, eliminating congestion requires distributing workload into the network evenly which moves some VNFs away from N-PoPs with the lowest operating cost. This natural tradeoff makes it important to consider the operating cost and the congestion jointly. By doing so, we can save operating costs when computing resources and link bandwidth are sufficient. On the other hand, we can reduce latency and packet loss by avoiding severe congestion when resources or bandwidth are highly limited.

The optimization model in this paper is designed to minimize the combination of operating cost and congestion cost. It deals with VNF placement and flow routing jointly. All notations used in this model are listed in Table 1.

2.1 Optimization Model

Denote the set of SFCs in the network as set R. SFC r is an SFC in set R with demand l_r, the amount of traffic load on r. X_r is the set of VNFs on SFC r. We define two types of decision variables, $A^v_{r,x}$ and $f^e_{r,x}$, for the joint VNF placement and routing. $A^v_{r,x}$ is a binary variable which indicates whether VNF x on SFC r is placed at N-PoP $v \in V$, where V is the set of all N-PoPs in the network. $f^e_{r,x} \in [0, 1]$

TABLE 1
Notations

Notation	Definition
V	set of all N-PoPs in the network
E	set of all connecting links between N-PoPs
R	set of all SFCs to be deployed into the network
X_r	set of VNFs demanded by SFC r
x	x^{th} type of VNF in set X_r
$A^v_{r,x}$	$\in \{0,1\}$, indicating whether VNF x of SFC r is placed on N-PoP v
$f^e_{r,x}$	$\in [0,1]$, indicating the fraction of flow using link e from VNF x to the next VNF on SFC r
l_r	service demand of SFC r
$\mathcal{G}(A,L)$	operating cost of VNFs, determined by $A:\{A^v_{r,x}\}$ and $L:\{l_r\}$
$\alpha^v_{r,x}$	the unit operating cost of VNF x from SFC r on N-PoP v
λ^v	weight parameter of congestion on N-PoP v
μ^e	weight parameter of congestion on link e
β	cost parameter of the N-PoP congestion
γ	cost parameter of the link congestion
Y	maximal congestion on N-PoPs
Z	maximal congestion on links
$in(v)$	set of links going into N-PoP v
$out(v)$	set of links going out of N-PoP v
M^v	capacity of N-PoP v
N^e	bandwidth of link e
$vl^e_{r,x}$	virtual link: link e carrying flow from VNF x to $x+1$ on SFC r
vg_r	virtual graph consisting of virtual links for the same r
$\eta_{r,k}$	the k^{th} virtual path consisting of virtual links of r from source to sink
$Pr(\eta_{r,k})$	probability of virtual path $\eta_{r,k}$
$cp_{r,g}$	g^{th} candidate path merged by all $\eta_{r,k}$ with the same VNF placement
$Pr(cp_{r,g})$	probability of picking candidate path $cp_{r,g}$ as placement of r

denotes the fraction of flow routing from VNF x to the next VNF $x+1$ of SFC r on link $e \in E$, where E is the set of all links in the network.

Operating cost: Denote by $\mathcal{G}(A,L)$ as the operating cost of VNFs, where $A = \{A^v_{r,x}\}$ and $L = \{l_r\}$. $\mathcal{G}$ is a linear function determined by A and L which can be extended to different forms based on actual conditions. In the following sections of this paper, without loss of generality, we define an $\alpha^v_{r,x}$ as the operating cost of deploying one unit of VNF x from SFC r on N-PoP $v \in V$. Therefore, we restrict $\mathcal{G}(A,L)$ as $\sum\limits_{r\in R}\sum\limits_{x\in X_r}\sum\limits_{v\in V}\alpha^v_{r,x} l_r A^v_{r,x}$.

N-PoP congestion: For each N-PoP v, denote by M^v the capacity of N-PoP v. Then, the load on N-PoP v caused by VNFs running on it is $\sum\limits_{r\in R}\sum\limits_{x\in X_r}\frac{l_r}{M^v}A^v_{r,x}$. Since the diversity of N-PoPs in computational capacity, I/O bandwidth and location, different N-PoPs may cause different scales of congestion even with the same load. For instance, N-PoPs serving as hubs of the network or N-PoPs with slower CPUs will have relatively larger congestion to the same amount of load. Thus, We define a weight parameter λ^v which represents the scale of congestion caused by adding one unit of load onto N-PoP v. We then use y^v to represent the congestion on N-PoP v, which indicates how much latency and packet loss it may introduce into the network. It is clear that congestion of v is positively related to λ^v and the load on the N-PoP. In this paper, we consider $y^v = \lambda^v\sum\limits_{r\in R}\sum\limits_{x\in X_r}\frac{l_r}{M^v}A^v_{r,x}$. We further denote by Y, a non-negative variable, the maximum congestion on all N-PoPs. we have $Y = \max\{y^v | v \in V\}$. Since the total amount of VNFs to be deployed into the network is fixed, larger Y means that the network is more congested with unbalanced load on N-PoPs. To compare N-PoP congestion with the operating cost, we introduce β, the cost parameter of N-PoP congestion, to measure the negative impact of N-PoP congestion to the network. The cost of N-PoP congestion is thus denoted as βY.

Link congestion: Similarly, N^e is the bandwidth of physical link e and μ^e is the weight parameter of link e in causing congestion. We then have z^e, the congestion on link e, that $z^e = \mu^e\sum\limits_{r\in R}\sum\limits_{x\in X_r}\frac{l_r}{N^e}f^e_{r,x}$. Denote by Z the maximum congestion on links and by γ as the cost parameter of link congestion. We can measure the cost of link congestion by γZ.

The relative sizes of β and γ compared to $\mathcal{G}$ are determined by the actual requirements and conditions. For instance, β and γ should be relatively large if the network services require low latency (e.g., online games, live broadcast) or the N-PoPs have no waiting queues and drop packets whenever congested. With such definitions, the SFC placement model is formulated as follows.

$$\min \quad \mathcal{G}(A,L) + \beta Y + \gamma Z$$
$$s.t.$$
$$\sum_{v\in V} A^v_{r,x} = 1 \qquad r\in R, x\in X_r \tag{1}$$
$$\sum_{e\in in(v)} f^e_{r,x} - \sum_{e\in out(v)} f^e_{r,x} = A^v_{r,x+1} - A^v_{r,x}$$
$$r\in R, x\in X_r, v\in V \tag{2}$$
$$\lambda^v\sum_{r\in R}\sum_{x\in X_r}\frac{l_r}{M^v}A^v_{r,x} \le Y \qquad v\in V \tag{3}$$
$$\mu^e\sum_{r\in R}\sum_{x\in X_r}\frac{l_r}{N^e}f^e_{r,x} \le Z \qquad e\in E \tag{4}$$
$$A^v_{r,x}\in\{0,1\}, \quad f^e_{r,x}\in[0,1]$$
$$r\in R, x\in X_r, v\in V, e\in E \tag{5}$$

The objective function minimizes the combination of operating cost and network congestion cost. Constraint (1) requires that each VNF should only be placed once on one N-PoP. Constraint (2) makes sure that, after VNFs are placed, all flows are routed accordingly. If both or neither of VNF x and $x+1$ is placed on the same N-PoP v, i.e., $A^v_{r,x+1} - A^v_{r,x} = 0$, the in-flow and out-flow $f^e_{r,x}$ to node v should be the same. Otherwise, if one of the two VNFs is placed on node v, the difference between in-flow and out-flow should be ± 1. Constraint (3) and (4) mean that Y and Z are the maximum congestion over N-PoPs and links.

According to [8], the problem of jointly placing VNFs and routing flows to minimize link congestion is NP-hard. We point out that such a problem is a special case of our model when parameters α and β are zeros. Therefore, our SFC placement model also leads to an NP-hard problem. In the following section, we propose an approximation algorithm to solve the problem in polynomial time with both good performance and theoretical guarantee to the optimal solution. All notations used in the algorithm can be found in Table 1.

2.2 Candidate Path Selection Algorithm

In this section, we raise an algorithm with polynomial time complexity named Candidate Path Selection (CPS) to yield SFC placement and routing with an approximation ratio of $\mathcal{O}(\log|V|)$ to the optimal solution, where $|V|$ is the total number of N-PoPs.

CPS first relaxes the original ILP problem into a linear programming (LP) problem. This is done by relaxing integer variables $A^v_{r,x} \in \{0,1\}$ to fractional ones $A^v_{r,x} \in [0,1]$. It then solves the relaxed LP and gets the (fractional) solution $\{A^v_{r,x}\}$ and $\{f^e_{r,x}\}$. Define a virtual link $vl^e_{r,x}$ which represents the physical link e carrying the flow from VNF x to VNF $x+1$ on SFC r. Each virtual link $vl^e_{r,x}$ corresponds to one $f^e_{r,x}$. With the same r, all $vl^e_{r,x}$ with positive $f^e_{r,x}$ formulate a virtual graph vg_r. The algorithm then does the following process for each SFC r to place it into the network.

In the virtual graph vg_r, CPS selects the virtual link $vl^{e'}_{r,x'}$ with the smallest positive $f^e_{r,x}$. Then, from the starting point of the selected virtual link, CPS finds the adjacent former virtual link pointing upward to the source of the SFC r. The former virtual link satisfies the following conditions. First, it belongs to the virtual graph vg_r. Second, its ending point is the start point of $vl^{e'}_{r,x'}$. Third, the former virtual link has the largest x which is smaller or equal to x'. We call these conditions virtual path conditions (VPC). If multiple links satisfy VPC, CPS randomly picks one as the former virtual link. CPS keeps finding former virtual links until reaching the source of the SFC r. In a similar method, CPS finds latter virtual links until the sink of r and a virtual path $\eta_{r,k}$ is thus created for the SFC from source to sink. All $f^e_{r,x}$ of the virtual links along the selected virtual path are subtracted by $f^{e'}_{r,x'}$. In this way, the selected virtual link $vl^{e'}_{r,x'}$ is removed from the virtual graph. The value of $f^{e'}_{r,x'}$ equals to $Pr(\eta_{r,k})$.

Based on $\eta_{r,k}$, CPS further figures out the placement of each VNF because virtual links contain information of both VNF placement and flow routing. Repeat picking $\eta_{r,k}$ until no positive $f^e_{r,x}$ left for r. Since all virtual paths with the same VNF placement can be seen as one SFC placement with multi-path routing, the algorithm merges all virtual paths $\eta_{r,k}$ which place their VNFs at same N-PoPs but route flows with different links as one candidate path $cp_{r,g}$. The probability $Pr(cp_{r,g})$ is the summation of all $Pr(\eta_{r,k})$ from the merged paths. The distributing of flow on each routing path is weighted by $\frac{Pr(\eta_{r,k})}{Pr(cp_{r,g})}$. In the way, CPS gets each SFC r a candidate path set. Select $cp_{r,g}$ in the candidate path set as the actual SFC placement scheme according to the probability $Pr(cp_{r,g})$. This specific value of $Pr(cp_{r,g})$ is an important prerequisite for the proof of our theoretical performance guarantee which will be discussed in detail in the following sections.

The detailed algorithm is shown in Algorithm 1. Line 3 to 12 present the process to get candidate paths. With candidate paths found, Line 13 to 19 then show how to place VNFs. Line 20 to 24 further merge all candidate paths with the same VNF placement into one and select a candidate path for each SFC r. When the candidate path of SFC r is chosen, its VNF placement and flow routing are thus determined.

We now analyze the complexity of Algorithm 1. For each SFC, Algorithm 1 continuously seeks for the virtual link with the smallest positive $f^e_{r,x}$ and reduces it to zero. Hence, there are at most $|X_r||E|$ iterations before all $f^e_{r,x}$ are reduced to zero. Meanwhile, in each iteration, Algorithm 1 finds a virtual path and subtracts the value of each link on the candidate path by the smallest positive $f^e_{r,x}$ in at most $|X_r||E|$ iterations. According to the determined virtual paths, Algorithm 1 also allocates VNFs on N-PoPs in $|X_r||V|$ iterations. Since in general, the number of VNF types in the network is a small constant, $|X_r|$ can be omitted in the expression of complexity. There are a total of $|R|$ SFCs, so the computational complexity of Algorithm 1 besides solving an LP problem is $\mathcal{O}(|R||E|^2)$. In the following section, We will prove the theoretical guarantee of Algorithm 1.

Algorithm 1 Candidate Path Selection Algorithm

```
Input: NFV network $(V,E)$, SFC demands: $\{l_r\}$
Output: set of routing paths selected for each SFC: $\{Path_r\}$; N-PoPs selected for allocation of VNFs: $\{nd_{r,x}\}$
1: solve the relaxed LP problem and get $\{A^v_{r,x}\}$, $\{f^e_{r,x}\}$
2: for each $r \in R$ do
3:     Define $vg_r$ as the virtual graph consisting of all $vl^e_{r,x}$ with positive $f^e_{r,x}$
4:     Define $\Omega_r$ as the set of candidate paths for $r$ at time $t$, $\Omega_r \leftarrow \emptyset$
5:     while $\exists\, vl^e_{r,x}$ with $f^e_{r,x} > 0$ do
6:         Select the virtual link $vl^{e'}_{r,x'}$ from $vg_r$ with $f^{e'}_{r,x'} = \min\{f^e_{r,x} | f^e_{r,x} > 0\}$
7:         Find a virtual path $\eta_{r,k}$ from $vg_r$ which contains the selected $vl^{e'}_{r,x'}$ according to VPC
8:         $Pr(\eta_{r,k})$ is the probability of picking $\eta_{r,k}$, $Pr(\eta_{r,k}) \leftarrow f^{e'}_{r,x'}$
9:         for each $vl^e_{r,x} \in \eta_{r,k}$ do
10:            $f^e_{r,x} \leftarrow f^e_{r,x} - f^{e'}_{r,x'}$
11:        end for
12:    end while
13:    Define $\{vn_{r,x,k}\}$ as the locations of VNFs on $\eta_{r,k}$
14:    for all N-PoP $u$ on $\eta_{r,k}$ do
15:        Define $vl^{e1}_{r,x1}$, $vl^{e2}_{r,x2}$ as adjacent virtual links connecting to N-PoP $u$
16:        for all $x \in (x1, x2]$ do
17:            $vn_{r,x} \leftarrow u$
18:        end for
19:    end for
20:    Due to multi-path routing, merge all $\eta_{r,k}$ with the same VNF placement $\{vn_{r,x,k}\}$ as the same candidate path $cp_{r,g}$ with VNF placement $\{cn_{r,x,g}\}$.
21:    $Pr(cp_{r,g}) \leftarrow \sum_{k \in merge(g)} Pr(\eta_{r,k})$.
22:    Distribute $\frac{Pr(\eta_{r,k})}{Pr(cp_{r,g})}$ of flow to routing path $k$.
23:    $\Omega_r \leftarrow \Omega_r \cap cp_{r,g}$
24:    Randomly select a $cp_{r,g}$ with probability $Pr(cp_{r,g})$, $Path_r \leftarrow cp_{r,g}$, $\{nd_{r,x}\} \leftarrow \{cn_{r,x,g}\}$
25: end for
```

2.3 Proof of the Approximation Ratio

As shown in Theorem 1, the approximation ratio between the CPS algorithm and the optimal solution is $\mathcal{O}(\log(|V|))$. In this theorem, we denote l_{max} and l_{min} as the maximal and the minimal values of l_r. Similarly, M_{max}, M_{min}, N_{max}, N_{min}, α_{max}, α_{min}, λ_{max}, λ_{min}, μ_{max} and μ_{min} are corresponding maximum and minimum of M^v, N^e, $\alpha^v_{r,x}$, λ^v and μ^e.

Theorem 1. *The total cost solved by Algorithm 1 guarantees with high probability an approximation ratio of $1 + d\ln(|V|)$ to the optimal solution of the ILP problem, where*

$$d = \max\{\frac{\alpha_{max}l_{max}}{|R||X|\alpha_{min}l_{min}}, \frac{2\lambda_{max}l_{max}M_{max}}{\lambda_{min}l_{min}M_{min}}, \frac{3\mu_{max}l_{max}N_{max}}{\mu_{min}l_{min}N_{min}}, \mathrm{e}^2\}.$$

In order to prove Theorem 1, we need Lemma 1 and Lemma 2 shown below.

Lemma 1. *Suppose $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m$ are binary independent random variables with $Pr\{\mathcal{X}_i = 1\} = p_i$ and $Pr\{\mathcal{X}_i = 0\} = 1-p_i$ for each $i \in \{1, 2, \ldots, m\}$. Let $\mathcal{Y} = \sum_i a_i\mathcal{X}_i$ with $a_i \geq 0$. h and $\mathcal{Z}$ are positive real numbers with $h \geq \max\{a_i | i \in [1, m]\}$ and $\mathcal{Z} \geq \mathbb{E}(\mathcal{Y})$. For any $\Delta \geq 0$, we have $Pr\{\mathcal{Y} \geq (1+\Delta)\mathcal{Z}\} \leq [\mathrm{e}^{\Delta}(1+\Delta)^{-\Delta-1}]^{\frac{\mathcal{Z}}{h}}$.*

The detailed proof of Lemma 1 can be found in [13].

Lemma 2. *For each SFC r, $\Theta^v_{r,x}$ is the set of candidate paths which allocate VNF x on node v. The sum of all probabilities of the candidate paths in $\Theta^v_{r,x}$ equals to the probability that v is chosen to deploy VNF x. Formally,*

$$\sum_{cp_{r,g} \in \Theta^v_{r,x}} Pr(cp_{r,g}) = A^v_{r,x}.$$

We can easily prove Lemma 2 using the strong correlation of the fractional solution $A^v_{r,x}$ and $f^e_{r,x}$. Though the probabilities of virtual paths in Algorithm 1 are determined by $f^e_{r,x}$, they also agree with $A^v_{r,x}$, the solution of the VNF placement. Therefore, when the deployment of virtual links is fixed, the allocation of VNFs is also determined. The detailed proof is omitted here due to page limitation.

Given Lemma 1 and 2, we now prove the distances between CPS and the optimal solution by seperately considering the operating cost, the N-PoP congestion cost and the link congestion cost. which leads to Lemma 3, 4 and 5.

Lemma 3. *The operating cost solved by Algorithm 1 guarantees with high probability an approximation ratio of $1 + a\ln(|V|)$ to H, the total operating cost from the LP solution, where*

$$a = \max\{\frac{\alpha_{max}l_{max}}{|R||X|\alpha_{min}l_{min}}, \mathrm{e}^2\}.$$

Proof. For each VNF placement, we define a random variable $\mathcal{X}^v_{r,x}$. If node v is selected to map VNF x of SFC r, $\mathcal{X}^v_{r,x} = 1$. Otherwise $\mathcal{X}^v_{r,x} = 0$. According to Lemma 2,

$$\mathbb{E}(\mathcal{X}^v_{r,x}) = \sum_{cp_{r,g} \in \Theta^v_{r,x}} Pr(cp_{r,g}) = A^v_{r,x}.$$

Next, consider random variable $\mathcal{Y} = \sum\limits_{r\in R}\sum\limits_{x\in X_r}\sum\limits_{v\in V} \alpha^v_x l_r \mathcal{X}^v_{r,x}$ which represents the VNF operating cost under the approximation algorithm. Hence,

$$\mathbb{E}(\mathcal{Y}) = \sum_{r\in R}\sum_{x\in X_r}\sum_{v\in V} \alpha^v_x l_r A^v_{r,x}.$$

which is also the operating cost of the relaxed LP problem denoted as H. Applying Lemma 1, we have

$$Pr(\mathcal{Y} \geq (1+\Delta)H) \leq [e^{\Delta}(1+\Delta)^{-\Delta-1}]^{\frac{H}{h}}$$
$$\leq [(\frac{\Delta}{e})^{-\Delta}]^{\frac{H}{h}}.$$

Recalling that $|V|$ is the number of nodes in the network, let $\Delta = a\ln(|V|)$ and $h = a_{max}l_{max}$. Since $a = \max\{\frac{\alpha_{max}l_{max}}{|R||X|\alpha_{min}l_{min}}, e^2\}$, we derive that

$$\ln(\frac{\Delta}{e})^{\Delta} = a\ln(|V|)(\ln(a) + \ln\ln(|V|) - 1) \geq a\ln(|V|)$$
$$\geq \frac{\alpha_{max}l_{max}}{|R||X|\alpha_{min}l_{min}}\ln(|V|).$$

And further, $(\frac{\gamma}{e})^{-\Delta} \leq |V|^{-\frac{\alpha_{max}l_{max}}{|R||X|\alpha_{min}l_{min}}}$. Clearly, $H \geq |R||X|\alpha_{min}l_{min}$. Combining all the solutions above, we get

$$Pr\{\mathcal{Y} \geq (1+\Delta)H\} \leq [|V|^{-\frac{\alpha_{max}l_{max}}{|T||R||X|\alpha_{min}l_{min}}}]^{\frac{H}{h}}$$
$$\leq [|V|^{-\frac{\alpha_{max}l_{max}}{|R||X|\alpha_{min}l_{min}}}]^{\frac{|R||X|\alpha_{min}l_{min}}{\alpha_{max}l_{max}}} = |V|^{-1}.$$

□

We now go on with Lemma 4. The proof is similar to that of Lemma 3. We only emphasize the differences.

Lemma 4. *The node congestion solved by Algorithm 1 guarantees with high probability an approximation ratio of $1 + b\ln(|V|)$ to J, the node congestion cost from the LP solution, where*

$$b = \max\{\frac{2\lambda_{max}l_{max}M_{max}}{\lambda_{min}l_{min}M_{min}}, \mathrm{e}^2\}.$$

Proof. With the same definition of $X^v_{r,x}$, we now consider the random variable $\mathcal{Y}^v = \lambda^v \sum\limits_{r\in R}\sum\limits_{x\in X_r} \frac{l_r}{M^v}\mathcal{X}^v_{r,x}$, which represents the congestion on node v from Algorithm 1. Hence,

$$\mathbb{E}(\mathcal{Y}^v) = \lambda^v \sum_{r\in R}\sum_{x\in X_r} \frac{l_r}{M^v} A^v_{r,x}.$$

According to constraint 3, we have $\mathbb{E}(\mathcal{Y}^v) \leq J$. Applying Lemma 1, we have $Pr(\mathcal{Y}^v) \geq (1+\Delta)J) \leq [(\frac{\Delta}{e})^{-\Delta}]^{\frac{J}{h}}$. Similarly, $\Delta = b\ln(|V|)$, $h = \frac{\lambda_{max}l_{max}}{M_{min}}$ and $b = \max\{\frac{2\lambda_{max}l_{max}M_{max}}{\lambda_{min}l_{min}M_{min}}, e^2\}$, we have $(\frac{\Delta}{e})^{-\Delta} \leq |V|^{-\frac{\lambda_{max}l_{max}M_{max}}{\lambda_{min}l_{min}M_{min}}}$.

Within our application scenarios, we always have $|R||X| \geq |V|$. In this way $J \geq \frac{|R||X|}{|V|}\frac{\lambda_{min}l_{min}}{M_{max}} \geq \frac{\lambda_{min}l_{min}}{M_{max}}$. Combining all conclusions above, we have

$$Pr\{\mathcal{Y}^v \geq (1+\Delta)J\} \leq [|V|^{-\frac{2\lambda_{max}l_{max}M_{max}}{\lambda_{min}l_{min}M_{min}}}]^{\frac{J}{h}}$$
$$\leq [|V|^{-\frac{2\lambda_{max}l_{max}M_{max}}{|T|\lambda_{min}l_{min}M_{min}}}]^{\frac{|T|\lambda_{min}l_{min}M_{min}}{\lambda_{max}l_{max}M_{max}}} = |V|^{-2}.$$

According to the Boole's inequality,

$$Pr\{\exists\, v \in V : \mathcal{Y}^v \geq (1+\Delta)J\}$$
$$\leq |V|Pr\{\mathcal{Y}^v \geq (1+\Delta)J\} \leq |V||V|^{-2} = |V|^{-1}.$$

□

We now show Lemma 5 and the corresponding proof.

Lemma 5. *The link congestion solved by Algorithm 1 guarantees with high probability an approximation ratio of $1 + c\ln(|V|)$ to K, the link congestion cost from the LP solution, where*

$$c = \max\{\frac{3\mu_{max}l_{max}N_{max}}{\mu_{min}l_{min}N_{min}}, \mathrm{e}^2\}.$$

Proof. We now consider $\mathcal{Y}^e$, the congestion on link e from Algorithm 1. In this proof, $\mathcal{X}^e_{r,x} = 1$ represents that there is a flow on link e from VNF i to the next VNF on SFC r. Due to the multi-path routing, we have

$$\mathcal{Y}^e \leq \mu^e \sum_{r\in R}\sum_{x\in X_r} \frac{l_r}{N^e}\mathcal{X}^e_{r,x}.$$

Then, we have

$$\mathbb{E}(\mathcal{Y}^e) \leq \mu^e \sum_{r\in R}\sum_{x\in X_r} \frac{l_r}{N^e} f^e_{r,x} \leq K.$$

In this way, Lemma 1 is still applicable. Following a similar proof of Lemma 4, we can get

$$\begin{aligned} &Pr\{\exists\, e \in E : \mathcal{Y}^e \leq |V|^2 Pr\{\mathcal{Y}^e \geq (1+\Delta)K\} \\ &\leq |V|^2|V|^{-3} = |V|^{-1}. \end{aligned}$$

□

With Lemmas 3, 4 and 5, we can finally prove Theorem 1 as follows.

Proof. We define a positive number d, where $d = \max\{a, b, c, e^2\}$. We also represent the overall cost solved by Algorithm 1 as $Cost(ALG_1)$ and the optimal cost of the ILP problem as OPT. With Lemma 3, 4 and 5, we conclude that

$$Cost(ALG_1) \leq (1 + d\ln(|V|))(J + H + K)$$

with high probability. Meanwhile, $J+H+K$ is the solution of the LP and serves as the lower pound of the optimal solution of the ILP that $J + H + K \leq OPT$. In this way, we have

$$Cost(ALG_1) \leq (1 + d\ln(|V|))OPT.$$

□

With the proof, we conclude that the cost of Algorithm 1 possesses a bounded approximation ratio of $\mathcal{O}(\log(|V|))$ to the optimal solution with high probability.

3 Online SFC Placement with Fast Demand Fluctuation

When SFC demands fluctuate very fast, the migration cost of a VNF is comparable to its operating cost. This is because that, when a VNF is migrated into another N-PoP within a short time period, its operating cost is also cut into a smaller scale. Thus, simply repeating CPS in Section 2 whenever a demand changes may introduce large migration cost and offset the benefit. According to practical VNF platforms [3] [4], the cost of migrating a VNF can be much lower than that of migration a VM. So, we can reduce the total cost by selectively migrating some VNFs at each time slot.

Migrating VNFs according to the change of SFC demands makes the problem in Section 2 an online SFC placement and routing problem seeking for the minimized total cost of VNF operating, congestion and VNF migration over the entire time span T. In this section, we first formulate a model leading to the online SFC placement problem. We then design a promising online algorithm to solve the problem under different migration cost and various prediction errors, which are major obstacles to good performance.

3.1 Online Optimization Model

To formulate the online model, we first introduce a dimension t and define the service demand of SFC r at time t as $l_r(t)$. Therefore, the fluctuation of demands is represented by different $l_r(t)$ at different time t. At any time slot t, future information is absent, which means that $l_r(\tau)$, the actual demand of SFC r at future time $\tau > t$, is unknown at time t. Similar to the offline model, denote by $A^v_{r,x}(t)$ and $f^e_{r,x}(t)$ the online decision variables at time t for the VNF placement and flow routing. Denote by $\mathcal{G}[A(t), L(t)]$ the operating cost of VNFs, where $A(t) = \{A^v_{r,x}(t)\}$ and $L(t) = \{l_r(t)\}$. Define $Y(t)$ and $Z(t)$ as maximal congestion costs on N-PoPs and links at time t, respectively.

Migration cost: The migration cost in the objective function is a major difference between the online and offline models. Denote by $\delta||A(t) - A(t-1)||$ the cost of migrating VNFs at time t, where $\delta \in \mathbb{R}^+$ and $||\cdot||$ can be any norm in $\mathbb{R}^n$. In this paper, without loss of generality, we define $\delta_{r,x}$ as the migration cost of VNF x on SFC r and assume that migrating a VNF in and out costs the same. In this way, the migration cost at time t is $\sum\limits_{r\in R}\sum\limits_{x\in X_r}\sum\limits_{v\in V}\delta_{r,x}|A^v_{r,x}(t) - A^v_{r,x}(t-1)|$.

The relative values of $\mathcal{G}$ and $\delta_{r,x}$ are affected by the length of a time slot and the type of a VNF. If VNF demands change violently within short time periods, or a particular type of VNFs has large migration overhead, $\delta_{r,x}$ should be relatively large.

The online SFC placement model is thus formulated as follows.

$$\begin{aligned}
\min \quad & \sum_{t\in T}\{\mathcal{G}[A(t), L(t)] + \beta Y(t) + \gamma Z(t) \\
& +\delta||A(t) - A(t-1)||\} \\
s.t. \quad & \\
& \sum_{v\in V} A^v_{r,x}(t) = 1 \\
& t \in T, r \in R, x \in X_r && (6) \\
& \sum_{e1\in in(v)} f^{e1}_{r,x}(t) - \sum_{e2\in out(v)} f^{e2}_{r,x}(t) = \\
& A^v_{r,x+1}(t) - A^v_{r,x}(t) \\
& t \in T, r \in R, x \in X_r, v \in V && (7) \\
& \lambda^v \sum_{r\in R}\sum_{x\in X_r}\frac{l_r(t)}{M^v}A^v_{r,x}(t) \leq Y(t) \\
& t \in T, v \in V && (8) \\
& \mu^e \sum_{r\in R}\sum_{x\in X_r}\frac{l_r(t)}{N^e}f^e_{r,x}(t) \leq Z(t) \\
& t \in T, e \in E && (9) \\
& A^v_{r,x}(t) \in \{0,1\}, \quad f^e_{r,x}(t) \in [0,1] \\
& t \in T, r \in R, x \in X_r, v \in V, e \in E && (10)
\end{aligned}$$

The formulated model leads to an online integer optimization problem, which is a general case of the NP-hard problem we discussed in Section 2. Thus, the online problem has both high complexity and unknown future information. In this paper, we break through such difficulties using a two-step online candidate path selection (OCPS) algorithm.

In the first step of the OCPS algorithm, it relaxes the online integer optimization problem into an online convex optimization (OCO) problem ($A^v_{r,x}(t)$ from $\{0,1\}$ to $[0,1]$). It then applies the committed horizon control (CHC) method proposed in our previous work [14] to solve the relaxed problem at each time slot. CHC adjusts the proportion of preceding data and predictions utilized to achieve solutions close to the offline optimal even with the presence of prediction errors. In the second step, OCPS applies an online candidate selection method which gets each SFC r a set of candidate paths based on both the solution of CHC and the placement of SFC r at time $t-1$. It then picks one candidate path as the placement of r for time t according to a novel criterion.

In the following sections, we first illustrate the CHC method and then present the online candidate path selection method. In the end, we show how the entire OCPS algorithm works combining both methods.

3.2 Committed Horizon Control

Solving OCO problems with switching costs has drawn much research attention. Two widely accepted algorithms are receding horizon control (RHC) [15] and averaging fixed horizon control [16]. RHC utilizes predictions of future demands to solve the online problem while AFHC combines both previous data and predictions. Both algorithms have their own expertise. While AFHC has a smaller theoretical bound, RHC performs better in many practical cases. However, when considering the problem to place SFCs with fast demand fluctuation, neither of the two algorithms may be efficient enough. First, as mentioned in the previous section, the migration cost δ may vary drastically due to different changing frequency of SFC demands. We will show in Section 4 that RHC and AFHC surpass each other with different δ. So, a more generalized algorithm is needed to deal with different migration costs of VNFs. Second, it is very hard to accurately predict the upcoming SFC demands. The presence of prediction errors makes it necessary for an algorithm that can preserve a theoretical guarantee coping with various scales of prediction errors.

CHC proposed in our previous work [14] is a good candidate in solving the particular OCO problem in this paper with various δ and prediction errors. CHC utilizes a novel mechanism called the commitment level to balance the proportion of past data and predictions in solving an OCO problem. This mechanism enables CHC to combine the advantages of both RHC and AFHC. When the commitment level is 1, CHC degenerates to RHC which only uses prediction data. On the contrary, When the commitment level equals the window size of prediction, CHC turns to AFHC which heavily depends on preceding data. Thus, by tuning the commitment level, CHC can adapt to different migration costs and types of prediction errors to get better performances compared with RHC and AFHC. Detailed CHC applied in our algorithm is summarized as follows.

At each time slot $t-1$, CHC predicts the future SFC demands in a window of size w. The detailed prediction can be done using methods such as machine learning, which is beyond the concern of this paper. In Section 4, we will discuss different types of prediction errors added to perfect predictions to evaluate the performance of our scheme over different prediction methods and scenes.

For a commitment level c, where $1 \le c \le w$, CHC need to decide $A(t)$ and $f(t)$ for the upcoming time slot t at time $t-1$. We denote the future SFC demands predicted at time $t-1$ in window $[t, t+w-1]$ by $\{l^*_r(\tau) | \tau \in [t, t+w-1]\}$. CHC then solves the OCO problem over the time span $[t, t+w-1]$ using the predicted data. This is a small-scale LP problem which can be solved in polynomial time. We use $\{A^{*v}_{r,x}(\tau) | \tau \in [t, t+w-1]\}$ and $\{f^{*e}_{r,x}(\tau) | \tau \in [t, t+w-1]\}$ to represent the solutions. We denote the portion of solutions for time t by sets $\{\mathcal{A}^{t-1,v}_{r,x} | \mathcal{A}^{t-1,v}_{r,x} = A^{*v}_{r,x}(t)\}$ and $\{\mathcal{F}^{t-1,e}_{r,x} | \mathcal{F}^{t-1,e}_{r,x} = f^e_{r,x}(t)\}$. The header $t-1$ on $\mathcal{A}^{t-1,v}_{r,x}$ means the variable is solved by the predictions made at time $t-1$. CHC then get $\mathcal{A}^{t-2,v}_{r,x}$ and $\mathcal{F}^{t-2,e}_{r,x}$ with the same procedure but the window of solving the convex problem becomes $[t-1, t+w-2]$. This process is repeated using predictions made at time τ until getting sets $\{\mathcal{A}^{\tau,v}_{r,x} | \tau \in [t-c, t-1]\}$ and $\{\mathcal{F}^{\tau,e}_{r,x} | \tau \in [t-c, t-1]\}$. The only difference for each step is that all $\{l^*_r(\tau) | \tau \le t-1\}$ in the predictions are replaced by real data $l_r(\tau)$ since they are already known at time $t-1$. According to CHC, the decision variables at time t are thus determined as $A^v_{r,x}(t) = \frac{1}{c}\sum_{\tau=t-c}^{t-1} \mathcal{A}^{\tau,v}_{r,x}$ and $f^v_{r,x}(t) = \frac{1}{c}\sum_{\tau=t-c}^{t-1} \mathcal{F}^{\tau,e}_{r,x}$. The detailed choice of commitment level c is determined by the scale of migration cost and the type of prediction errors, which will be discussed in detail in Section 4.

3.3 Online Candidate Path Selection Method

With fractional results from CHC at each time slot, we need to get integral solutions from them for directions of SFC placement and routing. However, applying the randomized rounding (RR) method in Algorithm 1 directly is no longer appropriate when the migration cost of VNFs is taken into consideration. Furthermore, the competitive ratio between the output and the offline optimal solution is not bounded with counterexamples. The reason is that the expected migration cost by applying randomized rounding can be larger than that of the fractional result from the CHC algorithm. In other words, there exist situations when

$$\mathbb{E}[\sum_{t\in T}\sum_{r\in R}\sum_{x\in X_r} \delta_{r,x}|\mathcal{X}^v_{r,x}(t) - \mathcal{X}^v_{r,x}(t-1)|]$$

$$> \sum_{t\in T}\sum_{r\in R}\sum_{x\in X_r} \delta_{r,x}|A^v_{r,x}(t) - A^v_{r,x}(t-1)|.$$

In this inequality, we denote by $\mathcal{X}^v_{r,x}(t)$ the binary variable determining VNF placement from the randomized rounding method in Algorithm 1 and by $A^v_{r,x}(t)$ the corresponding relaxed variable in CHC. As we can see, the kernel of the proof of Theorem 1 does not stand anymore because the migration cost can be arbitrarily larger than the CHC result. For a clearer explanation, we propose the following counterexample. Suppose we have $A^{v_1}_{r,x}(t) = A^{v_1}_{r,x}(t-1) = 0.5$ and $A^{v_2}_{r,x}(t) = A^{v_2}_{r,x}(t-1) = 0.5$. So the migration cost of placing VNF x on SFC r within $[t-1, t]$ is 0 in CHC. However, when applying the randomized rounding method, there is

50% possibility that VNF x on SFC r is placed at different N-PoPs in $[t-1,t]$ (once at v_1 and the other time at v_2). Then, the migration cost is positive and the ratio between the cost of randomized rounding and that of CHC is infinite.

To tackle such a drawback, we design the online candidate path selection method on the basis of Algorithm 1. The general idea of the online method is that, when it is profitable for total cost reduction, it migrates SFCs at time t by following Algorithm 1. When the migration of an SFC is not profitable with large migration cost and little reduction of operating cost and congestion, the method does not migrate the SFC and keeps the same deployment as that in time $t-1$.

The key part of our online candidate path selection is to find a criterion determining whether an SFC should be migrated or not. Standing at time $t-1$, we define a parameter $\Pi_r(t)$ guiding whether we migrate SFC r between $t-1$ and t. The smaller $\Pi_r(t)$ is, we are more likely to migrate r for total cost reduction. Otherwise, we tend to leave r at the same place as that in $t-1$. Denote the total expected migration cost at t is $\mathcal{E}_t$ when Algorithm 1 is applied based on the result of CHC. We then have $\mathcal{E}_t = \mathbb{E}[\sum_{r\in R}\sum_{x\in X_r} \delta_{r,x}|\mathcal{X}^v_{r,x}(t) - \mathcal{X}^v_{r,x}(t-1)|]$. Denote the expected migration cost of SFC r at time t is $\mathcal{E}_{t,r}$. We have $\mathcal{E}_{t,r} = \mathbb{E}[\sum_{x\in X_r} \delta_{r,x}|\mathcal{X}^v_{r,x}(t) - \mathcal{X}^v_{r,x}(t-1)|]$. For the fractional result of CHC, we define the total cost at t as C_t and the migration cost for SFC r at t as $C_{t,r}$. We then define $\Pi_r(t) = \frac{\mathcal{E}_t}{C_t+\sigma}\frac{|\mathcal{E}_{t,r}-C_{t,r}|}{\mathcal{E}_{t,r}+\sigma}$, where σ is a small positive number.

Here, $\frac{\mathcal{E}_t}{C_t+\sigma}$ is the ratio between the expected migration cost of Algorithm 1 and the total cost in the fractional solution of CHC at time t. If $\frac{\mathcal{E}_t}{C_t+\sigma}$ is large, the expected migration cost at t is comparable to the total cost in CHC. In this condition, SFC migration should be done carefully to avoid unnecessary migration cost. $\frac{|\mathcal{E}_{t,r}-C_{t,r}|}{\mathcal{E}_{t,r}+\sigma}$ indicates the gap between the expected migration cost for placing SFC r in Algorithm 1 and the migration cost of r in CHC at time t. Large $\frac{|\mathcal{E}_{t,r}-C_{t,r}|}{\mathcal{E}_{t,r}+\sigma}$ means that there is a big difference between these two cost (often caused by the rounding manner of Algorithm 1). In this case, the migration of SFC r at time t in Algorithm 1 cannot decrease the overall cost efficiently. It is clear that $\Pi_r(t)$ is large only when both conditions hold. In this way, SFCs with big $\Pi_r(t)$ should not be migrated between $t-1$ and t.

In the online method, We use $Path_r(t-1)$ to denote the deployment of SFC r at $t-1$. We then create a new set of candidate paths $\Omega'_r(t)$ which contains both $Path_r(t-1)$ and candidate paths for SFC r at t solved by Algorithm 1. This means $\Omega'_r(t) = \Omega_r(t) \cup Path_r(t-1)$, where $\Omega_r(t)$ contains candidate paths $cp_{r,g}(t)$ from Algorithm 1. We define $\frac{\Pi_r(t)}{\Pi_r(t)+1}$ as the probability of choosing $Path_r(t-1)$ as deployment of SFC r at time t. Therefore, the probability of choosing one candidate path from the rest of set $\Omega'_r(t)$ is $Pr(cp'_{r,g}(t))$ and

$$Pr(cp'_{r,g}(t)) = \frac{1}{\Pi_r(t)+1}Pr(cp_{r,g}(t)).$$

With the candidate path sets and probabilities, the online candidate path selection method then selects one candidate path for each SFC r based on its probability as the placement of r at time t.

With CHC and the online candidate path selection method presented, we now illustrate the OCPS algorithm containing both of them in the following section.

3.4 Online Candidate Path Selection Algorithm

OCPS shown in Algorithm 2 is the improved version of Algorithm 1 to deal with fast SFC demand changes under various VNF migration cost and prediction errors. At any time slot t, Algorithm 2 first applies CHC to get fractional solution $\{A^v_{r,x}(t)\}$ and $\{f^e_{r,x}(t)\}$ for each SFC r. Based on the fractional solution, the algorithm then achieves a set of candidate paths for SFC r and the probabilities to pick them using the same method proposed in Algorithm 1. Algorithm 2 further adds the deployment of SFC r in time slot $t-1$ into the candidate path set to denote the possibility that the SFC may not be migrated during $[t-1,t]$. The algorithm then calculates a criterion $\Pi_r(t)$ representing whether an SFC should be migrated or not. With $\Pi_r(t)$, the probability of picking each candidate path is adjusted accordingly. Finally, Algorithm 2 randomly selects a candidate path as the mapping of SFC r at time t based on the new probabilities. The computational complexity of Algorithm 2 besides solving a convex problem is the same as that of Algorithm 1 as $\mathcal{O}(|R||E|^2)$ for each time slot.

Algorithm 2 Online Candidate Path Selection Algorithm

```
Input: NFV network $(V,E)$; SFC demands: $\{l_r(\tau)|\tau \in [t-w+1, t-1]\}$; predictions: $\{l^*_r(\tau)|\tau \in [t, t+w-1]\}$; Placement of SFCs at $t-1$: $\{Path_r(t-1)\}$
Output: set of routing paths selected for each SFC at $t$: $\{Path_r(t)\}$; N-PoPs selected for placement of VNFs: $\{nd^v_{r,x}(t)\}$
1: Apply CHC with a proper commitment level and get fractional solution: $\{A^v_{r,x}(t)\}$, $\{f^e_{r,x}(t)\}$
2: if t = 0 then
3:     Apply Algorithm 1 to get $\{Path_r(0)\}$ and $\{nd^v_{r,x}(0)\}$.
4: else
5:     Apply Algorithm 1 to get $\Omega_r(t)$.
6:     for each $r \in R$ do
7:         $\Pi_r(t) = \frac{\mathcal{E}_t}{C_t+\sigma}\frac{|\mathcal{E}_{t,r}-C_{t,r}|}{\mathcal{E}_{t,r}+\sigma}$
8:         $\Omega'_r(t)$ is the set of candidate paths for SFC $r$ at time $t$, $\Omega'_r(t) \leftarrow \Omega_r(t) \cup Path_r(t-1)$.
9:         $Pr(Path_r(t-1)) = \frac{\Pi_r(t)}{\Pi_r(t)+1}$
10:        for all $cp'_{r,g}(t) \in \Omega'_r(t)/Path(t-1)$ do
11:            $Pr(cp'_{r,g}(t)) = \frac{1}{\Pi_r(t)+1}Pr(cp_{r,g}(t))$
12:        end for
13:        Randomly select a $cp'_{r,g}(t)$ with probability $Pr(cp'_{r,g}(t))$, $Path_r(t) \leftarrow cp'_{r,g}(t)$.
14:        Get the placement of VNFs $\{nd^v_{r,x}(t)\}$ using the same method as Algorithm 1.
15:    end for
16: end if
```

It is worth noting that, when the expected migration cost at t is small compared with the total cost in CHC, $\Pi_r(t)$ is much smaller than 1. Thus, the probability of choosing $Path_r(t-1)$ is negligible and $Pr(cp'_{r,g}(t))$ converges to $Pr(cp_{r,g}(t))$. Under such conditions, we can prove that the sum of operating and congestion cost is bounded by $\log(|V|)$ times of that in the CHC solution using a proof similar to Lemma 3. With the proven competitive ratio of CHC to the offline optimum in [14], we can derive that the sum of the operating and congestion cost from Algorithm 2 is also bounded to the offline optimal operating and congestion cost. Since the operating cost and the congestion cost take the majority of the total cost and both are proved bounded, we can conclude that the overall cost is also bounded to the offline optimum.

4 Evaluation

In this section, we first present the setup of simulations and then evaluate the performance of our SFC placement algorithms.

4.1 Simulation Setup

In this paper, we implement our simulations using Python and solve the linear programming problem with PuLP [17]. For the simulation setup, we refer to basic settings of the VNF platform on commercial servers in [4]. We consider each N-PoP as one server with available CPUs for VNFs from 0.06 to 6 cores. we also assign a random number uniformly distributed in [1, 10] as λ to represent the diversity of N-PoPs in causing congestion. We use the utilization of CPU resources to represent the demand of an SFC (e.g., a VNF takes up 0.6 core of a CPU). All detailed CPU utilization of SFCs comes from the CPU utilization of VMs in real traces from the Azure data.

The paper [4] does not involve information about links or networks connecting these servers and there exists a variety of possible network conditions in actual cases. In this way, we connect these servers with links of bandwidth uniformly distributed between 2% of the bandwidth to the maximum bandwidth of the server's NIC port. The connection rate of each network is randomly picked between 0.3 to 0.8. we then assign each link a weight parameter uniformly distributed in [1, 10] to represent the diversity of links in causing congestion. In this way, we can simulate the uncertainty of networks. We run our algorithms on a large number of these randomly generated networks for the generalized conclusion. In each network, there exist 20 to 30 servers with 40 to 80 SFCs. Values of parameters β, γ and δ used in the following simulations are all normalized by the average value of α. Therefore, we can analyze the relations among the operating cost, the congestion cost and the migration cost when applying our algorithms without knowing their absolute values.

4.2 Evaluation of the CPS Algorithm

In this section, we present simulation results of the CPS algorithm (Algorithm 1). For demands of SFCs, we randomly pick one time slot of CPU utilization of VMs from the real trace and assign them to the SFCs.

The first advantage of our algorithm is that it considers the operating cost and the network congestion simultaneously for overall cost reduction. In this way, Algorithm 1 prevents heavy network congestion when pursuing low operating cost. We verify this advantage in Fig. 2. The effect of reducing congestion is affected by relative values of congestion parameters β and γ which reflect service requirements and network conditions. We get the maximum congestion on N-PoPs and links in multiple simulations while increasing β and γ and present results in Fig. 2 (a). We observe that congestion on N-PoPs decreases significantly when β and γ increase. This indicates that when congestion on N-PoPs weighs more, our strategy further reduces the congestion for cost reduction. Moreover, we find that γ, the congestion parameter of links, also affects the congestion on N-PoPs. If γ increases together with β, congestion on N-PoPs is further reduced. We can draw similar conclusions from results of congestion on links. Moreover, since load on links have no integral constraint with multi-path routing, the link congestion is sharply reduced as long as $\gamma > 0$.

Fig. 2 (b) shows the effect of total cost reduction of our algorithm while considering different types of congestion. All costs in the figure are normalized by costs of placing SFCs ignoring congestion. According to the figure, it is obvious that Algorithm 1 can reduce the total cost as long as congestion matters with positive β and γ. In addition, the larger damage caused by congestion to the total profit, the better performance Algorithm 1 will have. Moreover, considering congestion on Both N-PoPs and links leads to the largest cost reduction.

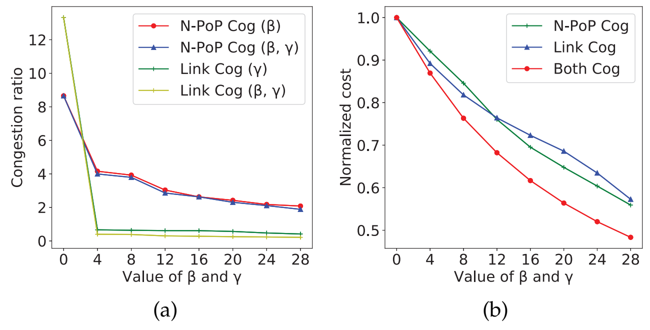


Fig. 2. (a) presents average maximum congestion on N-PoPs and links when β and γ increase. The line marked N-PoP Cog (β) denotes the congestion on N-PoPs when β changes with $\gamma = 0$. The line marked N-PoP Cog (β, γ) denotes the congestion on N-PoPs when both β and γ increase. Other lines have similar definitions. (b) shows the total cost with increasing β and γ when different types of congestion are considered. All costs are normalized by costs of placing SFCs without considering the congestion. The line marked Both Cog means the total cost of SFC placement when congestion on both N-PoPs and links is considered.

The second advantage of our algorithm is that it jointly considers the VNF placement and flow routing for global optimization. In Fig. 3, we compare our algorithm with an SFC placement algorithm named k-shortest paths. This algorithm first considers VNF placement to reduce the cost of VNF operating and congestion on N-PoP. It then manages the routing between adjacent VNFs by finding the first k shortest paths and distributing flows evenly on these paths. Fig. 3 (a) shows the congestion cost of Algorithm 1 and the k-shortest paths algorithm with different k. We only consider k no larger than 5, because there are often less than 5 routing paths between N-PoPs in the majority of networks and it is inefficient to manage multiple routing paths with little amount of flow. The results of algorithms are normalized by results of the LP. Fig. 3 (b) shows the corresponding total cost of two algorithms. From both figures, we observe that the performance of k-shortest paths improves with larger k but the improvement decreases as k grows. Furthermore, our algorithm always has lower congestion cost and total cost than the k-shortest paths algorithm even when k is large. This indicates the advantage of considering placement and routing jointly in our algorithm.

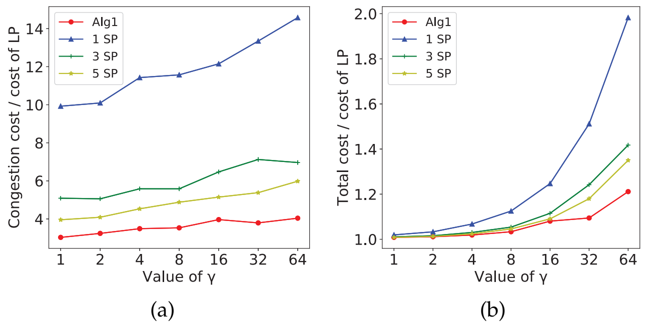


Fig. 3. (a) shows the congestion costs on links normalized by results from the LP when γ increases with different SFC placement strategies. (b) shows the corresponding result when counting the total cost. For all the simulations, $\beta = 10$. Lines marked with k SP represent results of the k-shortest paths algorithm with different k.

With results above, the CPS algorithm is verified to be effective in reducing the operating cost and the congestion cost while jointly placing VNFs and routing flows. We then present simulation results of the OCPS algorithm dealing with fast scaling demands.

4.3 Evaluation of the OCPS Algorithm

In this section, we evaluate the performance of the OCPS algorithm (Algorithm 2) using SFC demands from the real trace. Each time slot in the real trace represents 5 minutes and the total time span T contains 60 time slots. Each result in figures below is the average value of 20 independent simulations.

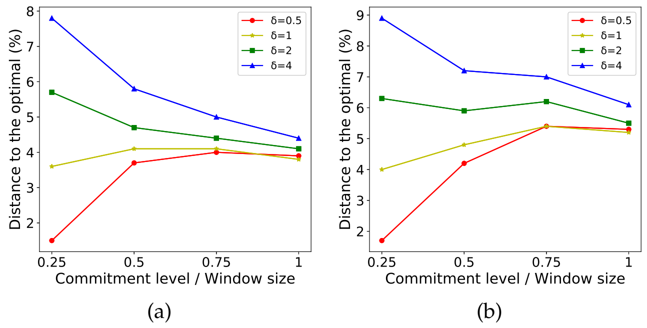


Fig. 4. The performance of CHC with different migration costs and prediction errors. (a) shows the distance between CHC and the offline optimal solution with different commitment levels when there exist uniformly distributed errors in predictions. Each line represents a CHC algorithm with a particular migration cost δ. The results are normalized by the value of optimal offline solutions. (b) shows the corresponding results when predictions are affected by heavy-tailed prediction errors.

In Fig. 4, we first present the performance of the CHC algorithms with different average migration costs δ when the commitment level increases to the window size. It is worth noting that when the ratio between the commitment level and the window size is small, CHC converges to RHC. When the ratio increases to 1, CHC will, on the contrary, become AFHC. So, Fig. 4 is actually comparing CHC with RHC and AFHC. Fig. 4 (a) shows the comparison with uniformly distributed prediction errors. The mean of errors is 5% of the SFC demand. It is clear that when the average migration cost is rather small, CHC with a lower commitment level will have lower overall cost and closer to the offline optimal solution. On the contrary, when the average migration cost is large, CHC algorithms with higher commitment levels will have better performances. Fig. 4 (b) shows similar simulations to (a). The only difference is that errors added to the perfect prediction follows a heavy-tailed distribution. That is, 5% of the errors are on average 10 times larger than the rest. The mean of the errors is still 5% of the SFC demand. We get a similar conclusion from Fig. 4 (b). Thus, we can conclude that CHC is more suitable for the specific OCO problem in this paper, since we can adjust the commitment level of CHC based on different migration costs and types of prediction errors for the best performance.

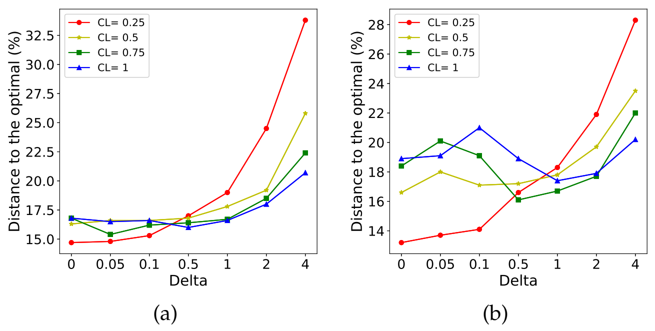


Fig. 5. The performance of OCPS with increasing migration cost and different prediction noises. (a) shows the distance between OCPS and the optimal solution when there exist uniformly distributed prediction errors. Each line corresponds to an OCPS algorithm with a different commitment level. The results are normalized by the value of optimal solutions. (b) shows corresponding results when predictions are affected by heavy-tailed prediction errors.

When considering which commitment level to choose for a particular scenario, we need to consider performances of CHC and the online candidate path selection method jointly. Fig. 5 presents the performance of the two-step OCPS algorithm with different commitment levels and increasing δ. Prediction errors added are the same as those in Fig. 4. Fig. 5 (a) shows that, under randomly distributed errors, OCPS algorithms with lower commitment levels work better when the average migration cost δ is small. OCPS algorithms with higher commitment levels take the lead when δ is relatively large. Fig. 5 (b) shows similar simulation results with heavy-tailed prediction errors. However, the discipline of OCPS under heavy-tailed prediction errors does not follow that lower commitment level leads to better performance with smaller migration costs. We propose that this is caused by the interaction between the heavy-tailed errors and the random rounding feature of the Online Candidate Path Selection Method. However, even with these extreme cases with heavy-tailed errors, we can still apply methods such as machine learning or large-scale simulation to find out which commitment level can achieve the best performance for a particular migration cost δ, which will be our direction in the further work.

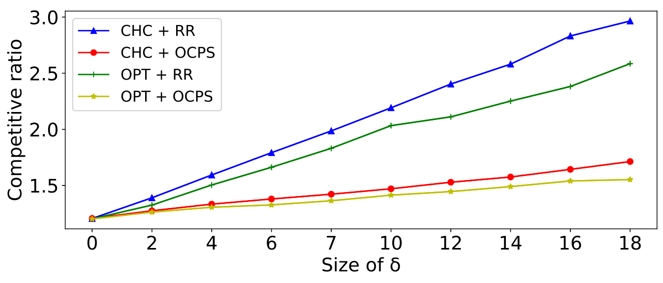


Fig. 6. Performance of the online candidate path selection method. Lines marked by CHC means the fractional solution is achieved by CHC with unknown future information, while OPT means the solution comes from offline optimal. Lines marked by OCPS means the integral solution is achieved using the online candidate path selection method, while RR means the solution comes from the randomized rounding method.

The online candidate path selection method in Section 3.3 is one of the main contributions in this paper to get integral results from CHC solutions, we use Fig. 6 to illustrate its improvement compared to directly applying the randomized rounding method. The results are normalized by the offline optimal solution. It is clear that applying the online candidate path selection method to the fractional output can significantly reduce the overall cost with or without knowing the upcoming SFC demands. Thus, it outperforms the traditional randomized rounding method for the online SFC placement problem in this paper.

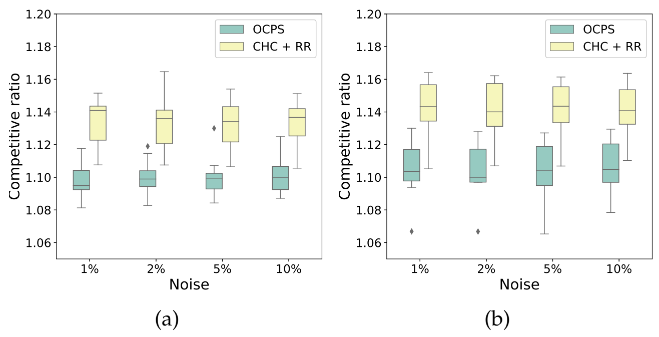


Fig. 7. The performance of the OCPS algorithm under different prediction errors. (a) shows the performance of OCPS under an increasing amount of uniformly distributed prediction errors (denoted by OCPS). It is compared with the algorithm with CHC and traditional randomized rounding method (denoted by RR). (b) shows similar simulations under an increasing amount of heavy-tailed prediction errors.

In the end, we show the ability of the OCPS algorithm in coping with different scale of prediction errors using Fig. 7. Here, we chose $\delta = 1$ and the commitment level is half of the window size. Boxes in Fig. 7 (a) conclude competitive ratios of the OCPS algorithm in 20 independent simulations with different scales of uniformly distributed prediction errors from 1% to 10% of the SFC demand. The baseline is the combination of the CHC algorithm and the randomized rounding method. We observe that, when prediction errors scale up, the performance of OCPS is rather stable with a slight deterioration. In addition, it is always superior to the baseline with the randomized rounding method. A similar conclusion can be drawn from Fig. 7 (b) with heavy-tailed errors. Thus, we can claim that the OCPS algorithm can tolerate different types and scales of prediction errors and preserves performance close to the optimal solution.

With evaluations above, we conclude that the OCPS algorithm works better than baselines and comparably to the offline optimal under a variety of migration costs and prediction errors. Thus, it is a good candidate in solving the online SFC placement problem in this paper.

5 Related Work

In recent years, methods have been proposed for deploying SFCs with different optimization objectives, e.g., [18]–[21]. For the purpose of reducing VNF costs, Moens et al. [5] propose a model called VNF-P for VNF resource allocation which focuses on a hybrid scenario where VNFs are provided by both dedicated physical hardware and virtualized service instances. Luo et al. [22] design an online scaling algorithm which balances the opening and operating costs of VNFs according to time-varying traffic demand. Shang et al. propose a self-adapting scheme to reduce the SFC backup cost over both the edge and the cloud in [23]. However, the above papers have not included server or network congestion into consideration.

To deal with congestion and latency problems, Sekar et al. propose a model exploring consolidation opportunities to reduce network provisioning cost and load skew in [3]. Ma et al. [8] design a novel scheme to place interdependent middleboxes considering their traffic changing effects. Carpio et al. propose heuristic algorithms to balance the load on links using replication in [9]. Ye et al. [24] establish an end-to-end packet delay modeling for embedded VNF chains in 5G networks. Zheng et al. [25] raise a 2-approximation algorithm for the latency optimization problem in Hybrid SFC composition and Embedding. Jin et al. [26] propose a two-stage VNF deployment scheme to optimize the resource utilization of both edge servers and physical links under the latency limitations. Nevertheless, the above work does not consider an online scheme that reduces the operating cost and controls network congestion at the same time when placing SFCs.

There have been several quality schemes dealing with the online pattern of VNF demands. Zhang et al. fuse online learning and online optimization to provide a proactive VNF provision with Multi-timescale cloud resources in [27]. The paper mainly focuses on online scaling and distribution of VNFs without detailed placement and routing. Fei et al. [11] also utilize the prediction from online learning to place and route SFCs. Yet, this paper process VNF placement and flow routing separately and leave space for further improvement with joint placement and routing. Guo et al. [12] propose an adaptive online algorithm to place and route VNFs whenever a new request comes. However, they reduce the complexity by predefining sets of routing paths at the cost of optimality. The network congestion can be further reduced with more choices of VNF placement and flow routing. Valls et al. [28] raise an online scheme that maximizes the analytics performance of data analytic services using multi-grade VNF chains while minimizing the data transfer and processing costs. Like other schemes mentioned above, VNF migration costs and prediction errors during the online process are not taken into consideration in their work. Therefore, the online SFC placement and routing scheme in our paper is novel which considers the cost of migrating VNFs between different time slots and remains robust with diverse prediction errors.

6 Conclusion and Future Work

In this paper, we study the problem of jointly optimizing the SFC placement and flow routing for cost-effectiveness and congestion control in VNF networks. We first propose an approximation algorithm to solve the offline problem with slow demand fluctuation. For the online version of the problem with fast demand fluctuation, we further propose an online algorithm that can handle diverse VNF migration costs and prediction errors while achieving comparable performances to the offline optimal solution. Extensive simulations validate the effectiveness of both algorithms.

When it comes to migrating VNFs among N-PoPs, traditional backup mechanisms with fixed numbers and locations of backups may fail to provide a sufficient and cost-effective reliability guarantee. This problem is faced by most online SFC deployment schemes. Therefore, one future direction is to develop online backup algorithms for both static and dynamic backups to ensure the availability of SFCs with minimal cost and environmental impacts. Meanwhile, with the rapid development of 5G and edge computing, an increasing number of SFCs are being deployed on the edge or a hybrid cloud-edge system. Optimizing the placement of SFCs in such environments is also an exciting future direction.

Acknowledgments

This research work was supported in part by the U.S. National Science Foundation under grant numbers CCF-1526162 and CCF-1717731.

References

[1] 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 *IFIP/IEEE IM 2015-IFIP/IEEE International Symposium on Integrated Network Management*. IFIP/IEEE, 2015, pp. 98–106.

[2] G. ETSI, "Network functions virtualisation (nfv): Architectural framework," *ETsI Gs NFV*, vol. 2, no. 2, 2013.

[3] V. Sekar, N. Egi, e. S. Ratnasamy, M. K. Reiter, and G. Shi, "Design and implementation of a consolidated middlebox architecture," in *USENIX NSDI 2012-USENIX Symposium on Networked Systems Design and Implementation*. USENIX, 2012, pp. 323–336.

[4] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, and F. Huici, "Clickos and the art of network function virtualization," in *USENIX NSDI 2014-USENIX Symposium on Networked Systems Design and Implementation*. USENIX, 2014, pp. 459–473.

[5] H. Moens and F. De Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *IEEE CNSM 2014-IEEE International Conference on Network and Service Management and Workshop*. IEEE, 2014, pp. 418–423.

[6] H. Feng, J. Llorca, A. M. Tulino, D. Raz, and A. F. Molisch, "Approximation algorithms for the nfv service distribution problem," in *IEEE INFOCOM 2017-IEEE International Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[7] J.-J. Kuo, S.-H. Shen, H.-Y. Kang, D.-N. Yang, M.-J. Tsai, and W.-T. Chen, "Service chain embedding with maximum flow in software defined network and application to the next-generation cellular network architecture," in *IEEE INFOCOM 2017-IEEE International Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[8] W. Ma, O. Sandoval, J. Beltran, D. Pan, and N. Pissinou, "Traffic aware placement of interdependent NFV middleboxes," in *IEEE INFOCOM 2017-IEEE International Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[9] F. Carpio, S. Dhahri, and A. Jukan, "VNF placement with replication for load balancing in NFV networks," in *IEEE ICC 2017-IEEE International Conference on Communications*. IEEE, 2017, pp. 1–6.

[10] X. Shang, Z. Li, and Y. Yang, "Placement of highly available virtual network functions through local rerouting," in *IEEE MASS 2018-IEEE International Conference on Mobile Ad Hoc and Sensor Systems*. IEEE, 2018, pp. 80–88.

[11] X. Fei, F. Liu, H. Xu, and H. Jin, "Adaptive vnf scaling and flow routing with proactive demand prediction," in *IEEE INFOCOM 2018-IEEE International Conference on Computer Communications*. IEEE, 2018, pp. 486–494.

[12] L. Guo, J. Pang, and A. Walid, "Joint placement and routing of network function chains in data centers," in *IEEE INFOCOM 2018-IEEE International Conference on Computer Communications*. IEEE, 2018, pp. 612–620.

[13] K. Han, Z. Hu, J. Luo, and L. Xiang, "Rush: Routing and scheduling for hybrid data center networks," in *IEEE INFOCOM 2015-IEEE International Conference on Computer Communications*. IEEE, 2015, pp. 415–423.

[14] N. Chen, J. Comden, Z. Liu, A. Gandhi, and A. Wierman, "Using predictions in online optimization: Looking forward with an eye on the past," *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 1, pp. 193–206, 2016.

[15] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 35, no. 7, pp. 814–824, 1990.

[16] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *IEEE IGCC 2012-IEEE International Green Computing Conference*. IEEE, 2012, pp. 1–10.

[17] S. Mitchell, M. OSullivan, and I. Dunning, "Pulp: a linear programming toolkit for python," *The University of Auckland, Auckland, New Zealand*, 2011.

[18] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *IEEE CloudNet 2014-IEEE International Conference on Cloud Networking*. IEEE, 2014, pp. 7–13.

[19] Y. Sang, B. Ji, G. R. Gupta, X. Du, and L. Ye, "Provably efficient algorithms for joint placement and allocation of virtual network functions," in *IEEE INFOCOM 2017-IEEE International Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[20] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint VNF placement and cpu allocation in 5g," in *IEEE INFOCOM 2018-IEEE International Conference on Computer Communications*. IEEE, 2018, pp. 1943–1951.

[21] X. Shang, Y. Liu, Y. Mao, Z. Liu, and Y. Yang, "Greening reliability of virtual network functions via online optimization," in *IEEE/ACM IWQoS 2020-IEEE/ACM International Symposium on Quality of Service*. IEEE, 2020, pp. 1–10.

[22] Z. Luo and C. Wu, "An online algorithm for VNF service chain scaling in datacenters," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1061–1073, 2020.

[23] X. Shang, Y. Huang, Z. Liu, and Y. Yang, "Reducing the service function chain backup cost over the edge and cloud by a self-adapting scheme," in *IEEE INFOCOM 2020-IEEE International Conference on Computer Communications*. IEEE, 2020, pp. 2096–2105.

[24] Q. Ye, W. Zhuang, X. Li, and J. Rao, "End-to-end delay modeling for embedded VNF chains in 5g core networks," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 692–704, 2018.

[25] D. Zheng, C. Peng, X. Liao, L. Tian, G. Luo, and X. Cao, "Towards latency optimization in hybrid service function chain composition and embedding," in *IEEE INFOCOM 2020-IEEE International Conference on Computer Communications*. IEEE, 2020, pp. 1539–1548.

[26] P. Jin, X. Fei, Q. Zhang, F. Liu, and B. Li, "Latency-aware VNF chain deployment with efficient resource reuse at network edge," in *IEEE INFOCOM 2020-IEEE International Conference on Computer Communications*. IEEE, 2020, pp. 267–276.

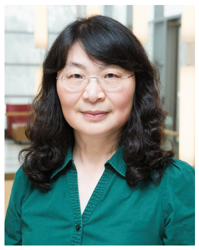
[27] X. Zhang, C. Wu, Z. Li, and F. C. Lau, "Proactive vnf provisioning with multi-timescale cloud resources: Fusing online learning and online optimization," in *IEEE INFOCOM 2017-IEEE International Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[28] V. Valls, G. Iosifidis, G. de Mel, and L. Tassiulas, "Online network flow optimization for multi-grade service chains," in *IEEE INFO-*

COM 2020-IEEE International Conference on Computer Communications. IEEE, 2020, pp. 1329–1338.

Xiaojun Shang received his B. Eng. degree in Information Science and Electronic Engineering from Zhejiang University, Hangzhou, China, and M.S. degree in Electronic Engineering from Columbia University, New York, USA. He is now pursuing his Ph.D. degree in Computer Engineering at Stony Brook University. His research interests are in cloud computing and data center networks, with focus on placement and routing of virtual network functions and resilience of service function chains.

Zhenhua Liu is currently assistant professor in the Department of Applied Mathematics and Statistics, also affiliated with Department of Computer Science and Smart Energy Technology Cluster, since August 2014. During the year 2014-2015, he is on leave for the ITRI-Rosenfeld Fellowship in the Energy and Environmental Technology Division at Lawrence Berkeley National Laboratory. Dr. Liu received his Ph.D. degree in Computer Science at the California Institute of Technology, where he was co-advised by Prof. Adam Wierman and Prof. Steven Low. Before Caltech, he received an M.S. degree of Computer Science Technology in 2009 and a B.E. degree of Measurement control in 2006, both from Tsinghua University with honor, as well as a B.S. degree of Economics from Peking University in 2009.



Yuanyuan Yang received the BEng and MS degrees in computer science and engineering from Tsinghua University, Beijing, China, and the MSE and Ph.D. degrees in computer science from Johns Hopkins University, Baltimore, Maryland. She is a SUNY Distinguished Professor of computer engineering and computer science at Stony Brook University, New York, and is currently on leave at the National Science Foundation as a Program Director. Her research interests include edge computing, data center networks, cloud computing and wireless networks. She has published more than 460 papers in major journals and refereed conference proceedings and holds seven US patents in these areas. She is currently the Editor-in-Chief for IEEE Transactions on Cloud Computing and an Associate Editor for IEEE Transactions on Parallel and Distributed Systems and ACM Computing Surveys. She has served as an Associate Editor-in-Chief for IEEE Transactions on Cloud Computing, Associate Editor-in-Chief and Associated Editor for IEEE Transactions on Computers, and Associate Editor for IEEE Transactions on Parallel and Distributed Systems. She has also served as a general chair, program chair, or vice chair for several major conferences and a program committee member for numerous conferences. She is an IEEE Fellow.