Capacitated Shortest Path Tour Problem Based Integer Linear Programming for Service Chaining and Function Placement in NFV Networks

Masahiro Sasabe, *Member, IEEE,* Takanori Hara

***Abstract*—Network functions virtualization (NFV) is a new paradigm to achieve flexible and agile network services by decoupling network functions from proprietary hardware and running them on generic hardware as virtual network functions (VNFs). In the NFV network, a network service can be modeled as a sequence of VNFs, called a service chain. Given a connection request (e.g, origin, destination, and a sequence of required functions), we have to solve both the service chaining and function placement problems to find an appropriate service path that optimizes the objective (e.g., minimization of the total path delay) while satisfying the service chain requirements. In this paper, focusing on the similarity between the service chaining problem and the shortest path tour problem (SPTP) and developing the novel network model called augmented network, we formulate capacitated SPTP-based integer linear programs (ILPs) for the service chaining and function placement. Through numerical results obtained by the existing solver, we show the proposed ILP for the service chaining can support 1.22–1.90 times as large-scale systems as the existing ILP. Furthermore, we also demonstrate that the proposed ILP for both the service chaining and function placement can shorten the total delay by 15.8% compared with that only for the service chaining. For further scalability, we propose a shortest-path-based heuristic algorithm to solve the ILPs and show the heuristic for service chaining and function placement can calculate the optimal solution with high accuracy in strongly polynomial time.**

***Index Terms*—Network functions virtualization (NFV), service chaining, function placement, integer linear programming (ILP), augmented network, capacitated shortest path tour problem (CSPTP)**

I. Introduction

NETWORK functions virtualization (NFV) can realize flexible and agile network services by decoupling network functions (e.g., routing, firewall, deep packet inspection, and load balancing) from dedicated hardware (e.g., appliances and middleboxes) and running them on generic hardware as virtual network functions (VNFs) [2]. In what follows, the terms VNF and function are used interchangeably. In addition to the flexibility and agility, NFV can also reduce both capital expenditures (CAPEX) and operating expenditures (OPEX) because VNFs can be built on the generic hardware and liberate operators from learning knowledge for the conventional dedicated hardware.

The resource allocation is one of the challenging issues in the NFV networks [3], [4]. We can view a certain network service as a sequence of VNFs, called a service chain (SC) or service function chain (SFC) [5]. Given a connection request with service chain requirements (e.g., an origin node, a destination node, and a sequence of functions), the *service chaining* problem aims to find an appropriate *service path*, which starts from the origin node and ends with the destination node while executing the corresponding VNFs at the intermediate nodes in the required order. The service chaining problem belongs to **NP**-hard [3]. Note that the service chain can have a more complex structure (i.e., directed graph) but we focus on the sequential type of service chain in this paper. Furthermore, the locations of functions in the NFV network also affect the effectiveness of the service path, and thus the *function placement* problem also arises [6].

One of the difficulties of the service chaining problem is how to guarantee to execute functions along the service path in the required order. Existing studies tackled this problem in various ways, e.g., advance enumeration of path candidates [7], utilizing an expanded network [8], and utilizing a layered graph [9], [10]. Thanks to these techniques, they formulated the service chaining problem as integer linear programs (ILPs) at the cost of introducing some complexity. More specifically, enumerating path candidates is more difficult than the simple source-destination path enumeration, which belongs to **#P**-complete class [11] and both the expanded network and the layered graph require to construct a hierarchical network by preparing copies (layers) of the original physical network in proportion to the number of required functions.

Recently, several studies coped with the ordering problem by focusing on the similarity between the shortest path tour problem (SPTP) [12]–[14] and the service chaining problem [15], [16]. The SPTP is a variant of the shortest path problem (SPP) and tries to find a shortest path from an origin node to a destination node such that the path must visit a sequence of disjoint node subsets $\mathcal{T}_1, \ldots, \mathcal{T}_K$ in this order. For example, $\mathcal{T}_k$ $(k = 1, \ldots, K)$ can be regarded as touristic spots on a certain tour. Bhat and Rouskas first showed the possibility of modeling the service chaining problem as the SPTP [15]. Gao and Rouskas proposed the SPTP-based heuristic algorithm for the online service chaining [16].

The service chaining problem is similar to the SPTP but it also has a different feature, i.e., constraints on node and link capacities. The constrained SPTP is a special case of the SPTP where the maximum number of times that an edge can

M. Ssasabe and T. Hara are with the Division of Information Science, Nara Institute of Science and Technology, Nara, Japan e-mail: m-sasabe@ieee.org, hara.takanori.hm8@is.naist.jp.

This paper is an expanded version of the paper that was presented at 6th IEEE International Conference on Network Softwarization (IEEE NetSoft 2020) [1].

Manuscript received April 19, 2005; revised August 26, 2015.

be traversed is limited to a predefined integer, e.g., one [17]. From the viewpoint of the service chaining problem, we can generalize the constrained SPTP to support constraints on both node and link capacities with real values. In this paper, we refer to this generalized version as a *capacitated SPTP (CSPTP)*.

Recently, Andrade and Saraiva proposed an ILP for the constrained SPTP [18]. In our previous work [1], combining this approach and a novel network model called *augmented network*, we showed that the service chaining problem can exactly be modeled as a CSPTP-based ILP, which minimizes the total delay of the service path under constraints on node and link capacities. To the best of our knowledge, this was the first work that exactly formulated the service chaining problem as the CSPTP-based ILP. Compared with the above-mentioned ILPs for the sequential-type service chaining [7]–[10], the CSPTP-based ILP can be modeled as much simpler formulation with a more compact network structure, which contributes to the scalability and extensibility of the formulation. In this paper, we further extend this ILP for both the service chaining and function placement, which can determine the appropriate service path as well as the appropriate number and locations of VNFs in the NFV networks. Through numerical results obtained by solving the proposed ILPs using the existing solver CPLEX 12.8 [19], we evaluate the proposed ILPs in terms of the scalability and effectiveness of the resource allocation.

The main contributions of this paper are as follows:

1) We reveal the exact relationship between the conventional SPTP and the NFV-related problems (i.e., service chaining and function placement) by developing two kinds of CSPTP-based ILPs, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, with the help of the combination of the ILP formulation of the constrained SPTP and the augmented network. $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ is the CSPTP-based ILP for the service chaining while $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ is that for both the service chaining and function placement.
2) We propose a new type of NFV network model called augmented network, which can connect the NFV-related problems to the conventional SPTP. The augmented network can be much smaller than the existing NFV network models (i.e., layered graph and expanded network) especially when the number of functions required in the service chain increases.
3) Through numerical results obtained by solving the proposed ILPs using the existing solver CPLEX, we will show $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ can support 1.22–1.90 as times large-scale systems as the existing ILP over the expanded network. More specifically, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ (resp. $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$) can support 440 and 1520 (resp. 420 and 1380) physical nodes within 1 and 10 execution time [s], respectively. We further demonstrate that $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ can reduce the total delay of the service path by 15.8% and the average physical node (resp. link) utilization by 7.3% (resp. 61.1%) compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$.
4) For further scalability, we propose a shortest-path-based heuristic algorithm to solve the proposed ILPs and show that the heuristic for both service chaining and function placement can calculate the optimal solution with high accuracy in strongly polynomial time.

The rest of the manuscript is organized as follows. Section II gives the related work. After introducing the system model in Section III, we develop the two kinds of CSPTP-based ILPs for the service chaining and function placement in NFV networks. In Section V, we give a simple greedy-based heuristic algorithm to solve the proposed ILPs. We demonstrate the scalability and effectiveness of the proposed CSPTP-ILPs and heuristic algorithm through numerical results in Section VI. Finally, Section VII gives the conclusions and future work.

II. Related Work

The resource allocation is one of the most challenging problems in NFV networks because it requires to deal with various types of requirements (e.g., service (function) chaining, function placement, scheduling, and demand provisioning) as well as their complex dependence. The recent survey on NFV networks can be found in [3], [4], [6]. In what follows, we introduce existing studies on the ILP formulation for the NFV-related problems (i.e., service chaining and function placement) and SPTP-related work.

Service chaining is one of the major NFV resource allocation problems, which aims to find an appropriate service path under the constraints of function locations and service chain requirements (e.g., processing/bandwidth demand and a sequence of functions). It is a kind of combinatorial optimization problems and there are multiple studies on the mathematical formulation of the service chaining problem [7]–[10], [20], [21]. Nguyen et al. formulated the service chaining problem as an ILP over an expanded network to minimize the blocking probability of service chain requests [8]. Huin et al. developed an ILP over a layered graph to minimize the total bandwidth usage [9]. Gupta et al. also formulated an ILP to minimize the total bandwidth usage [7]. Savi et al. modeled an ILP to minimize the number of active nodes by considering the processing-related costs, i.e., context switching costs and upscaling costs, in a VNF consolidation scenario where multiple VNFs were served at the same hardware [20]. Nejad et al. [21] aimed to maximize the total revenue. Hyodo et al. formulated an ILP to minimize the placement cost and bandwidth usage under the relaxation of VNF order and non-loop constraints [10].

Path selection-based service chaining aims to find an appropriate service path from given path candidates. D'Oro et al. applied the non-cooperative game theory to achieve service chaining a distributed manner [22]. They also proposed a two-stage Stackelberg game composed of leading servers and following users to achieve the distributed resource allocation and orchestration [23].

Function placement also plays an important role to minimize the total path delay as well as utilize the limited resource effectively. Bhamare et al. formulated an ILP for the VNF placement across geographically distributed clouds to minimize the total response time to the end-users in the network [24]. Li et al. formulated a facility location problem based ILP for the VNF placement to minimize the resource consumption [25].

Tomassilli et al. formulated the VNF placement under the SFC constraints as a set cover problem and proposed approximation algorithms to solve it [26].

There are also several studies that aim to solve both the service chaining and function placement. Sallam et al. first proposed a transformation of the network to calculate SFC-constrained shortest path candidates, then proposed two kinds of ILPs for SFC-constrained maximum flow and VNF placement [27]. Gouareb et al. modeled an ILP for both the VNF routing and placement across the physical nodes to minimize the edge-cloud latency [28]. Soualah et al. proposed an ILP for both the service chaining and function placement to minimize the resource usage when VNFs are shared across tenants [29].

Recently, some researchers noticed that the service chaining is similar to the conventional SPTP [15], [16], [30]. The SPTP is a variant of the SPP where some intermediate nodes (locations) should to be visited in a predefined order. Festa et al. achieved polynomial-time reduction of the SPTP to a classical SPP over a modified digraph and proposed heuristics to solve the SPTP [14]. Ferone et al. studied constrained SPTP, which is a variant of the SPTP without repeated links, proved that it belonged to the complexity class of **NP**-complete, and proposed a branch-and-bound based heuristic [17]. Ferone et al. further proposed a mathematical model and new branch-and-bound heuristics for the constrained SPTP [31]. Recently, Andrade and Saraiva developed an ILP for the constrained SPTP [18]. They also proposed a Lagrangian-based heuristic framework to solve the constrained SPTP [32]

As for the application of the SPTP in the NFV networks, Bhat and Rouskas showed that service chaining could be modeled as SPTP [15]. They proposed a heuristic algorithm to solve the SPTP and compared it with existing algorithms. Gao et al. proposed a SPTP-based online path-selection algorithm to minimize the maximum network congestion [16]. Focusing on the similarity between the service chaining and SPTP, Liu et al. tackled the scaling problem of service chaining with the help of the combination of the multistage graph and the max-flow min-cut theory [30].

To the best of our knowledge, this is the first work that exactly formulates CSPTP-based ILPs for the NFV-related problems (i.e., service chaining and function placement).

III. System Model

In this section, we describe the system model considered in this manuscript, from the viewpoint of service chain request, augmented network, and service path. Table I summarizes the notations used in this paper. In Section IV, we will provide two kinds of ILPs: one is only for the service chaining, $\text{ILP}_{\text{SC}}^{\text{CSPTP}}$, and another is for both the service chaining and function placement, $\text{ILP}_{\text{SCFP}}^{\text{CSPTP}}$. In this section, for simplicity in explanation, we mainly focus on the service chaining and give the information related to both the service chaining and function placement if required.

A. Service Chain Request

TABLE I
Notations in the model.

Notation	Definition
G	Physical network $G = (\mathcal{V}, \mathcal{E})$
$\mathcal{V}$	Set of physical nodes, $V = \|\mathcal{V}\|$
$\mathcal{E}$	Set of physical links, $E = \|\mathcal{E}\|$
$\mathcal{F}$	Set of functions, $\mathcal{F} = \{f_1, \ldots, f_F\}$, $F = \|\mathcal{F}\|$
$\mathcal{F}_i$	Set of functions contained in physical node $i \in \mathcal{V}$
N_f	Number of physical nodes capable of function f
$\mathcal{V}_{\text{VNF}}$	Set of VNF-enabled physical nodes, $\mathcal{V}_{\text{VNF}} \subseteq \mathcal{V}$
c	Connection with origin o_c and destination d_c
$\mathcal{R}_c$	Sequence of functions $(f_{c,1}, \ldots, f_{c,K_c})$ required by c
$\mathcal{K}_c, \mathcal{K}_c^+$	$\mathcal{K}_c = \{1, \ldots, K_c\}, \mathcal{K}_c^+ = \{1, \ldots, K_c + 1\}$
$\widehat{\mathcal{V}}$	Set of imaginary nodes, $\widehat{\mathcal{V}} = \{\hat{v}_f\}_{f \in \mathcal{F}}$, where imaginary node $\hat{v}_f$ is responsible for function f
$\widehat{\mathcal{E}}^{\text{in}}$	Set of links incoming to imaginary nodes, $\widehat{\mathcal{E}}^{\text{in}} = \{(v, \hat{v}) \mid v \in \mathcal{V}, \hat{v}_f \in \widehat{\mathcal{V}}, f \in \mathcal{F}_v\}$
$\widehat{\mathcal{E}}^{\text{out}}$	Set of links outgoing from imaginary nodes, $\widehat{\mathcal{E}}^{\text{out}} = \{(\hat{v}, v) \mid \hat{v}_f \in \widehat{\mathcal{V}}, v \in \mathcal{V}, f \in \mathcal{F}_v\}$
G^+	Augmented network of G, $G^+ = (\mathcal{V}^+, \mathcal{E}^+)$
$\mathcal{V}^+$	Set of nodes in G^+, $\mathcal{V}^+ = \mathcal{V} \cup \widehat{\mathcal{V}}$, $V^+ = \|\mathcal{V}^+\|$
$\mathcal{V}_i^+$	Set of node i's neighbors in G^+
$\mathcal{E}^+$	Set of links in G^+, $\mathcal{E}^+ = \mathcal{E} \cup \widehat{\mathcal{E}}^{\text{in}} \cup \widehat{\mathcal{E}}^{\text{out}}$
$\mathcal{S}_c$	Service path for c, $(\mathcal{S}_{c,1}, \ldots, \mathcal{S}_{c,K_c+1})$
$\mathcal{S}_{c,k}$	kth sub service path with origin $a_{c,k}$ and destination $b_{c,k}$
b_c	Bandwidth requirement of c
p_c^{node}	Processing requirement of c for traversing a node
$p_{c,f_{c,k}}^{\text{func}}$	Processing requirement of $f_{c,k} \in \mathcal{R}_c$ at a node
$\boldsymbol{r}_c$	Service chain requirements for connection c $(o_c, d_c, \mathcal{R}_c, b_c, p_c^{\text{node}}, \{p_{c,f_{c,k}}^{\text{func}}\}_{k=1,\ldots,K_c})$
$B_{i,j}$	Residual bandwidth of link (i, j) at arrival of c
P_i	Residual processing capacity of node i at arrival of c at arrival of c
$d_{i,j}^{\text{link}}$	Propagation delay of link $(i, j) \in \mathcal{E}$
d_i^{node}	Traversal delay of node $i \in \mathcal{V}$
$d_{\hat{v}_f,v}^{\text{func}}$	Processing delay of function f of node $v \in \mathcal{V}$
$u_{i,j}$	Utilization of physical link (i, j) (derived variables)
u_i	Utilization of physical node i (derived variables)
$x_{i,j}^{c,k}$	Binary decision variables: 1: if link (i, j) is included in $\mathcal{S}_{c,k}$, 0: otherwise

We consider an NFV network where connection requests randomly arrive as in [8]. The orchestrator waits for C $(C = 1, \ldots, C_{\max})$ requests and solves the service chaining (and function placement) problem for the collected requests $\mathcal{C}$. The way of processing the request(s) can be categorized into three types, i.e., online, batch, and offline, depending on the size of $\mathcal{C}$, i.e., C. In case of the online processing $(C = 1)$, the orchestrator serves the connection request just after its arrival. If the orchestrator knows all the requests $(C = C_{\max})$ in advance, it solves the service chaining problem for them at once, and thus this results in the offline processing. In other cases, i.e., $C = 2, \ldots, C_{\max} - 1$, the orchestrator adopts the batch processing with size of C.

A connection $c \in \mathcal{C}$ has service chain requirements, $\boldsymbol{r}_c = (o_c, d_c, \mathcal{R}_c, b_c, p_c^{\text{node}}, \{p_{c,f_{c,k}}^{\text{func}}\}_{k=1,\ldots,K_c})$. o_c (resp. d_c) is a physical node that the connection c starts from (resp. ends with). $\mathcal{R}_c$ is a sequence (an ordered set) of K_c $(K_c > 0)$ functions $(f_{c,1}, \ldots, f_{c,K_c})$ with which the connection c will be served in this order. In general, the sequence of functions, $\mathcal{R}_c$, can be more complex processing order rather than the sequential order, e.g., inclusion of split and merge. In this paper, we mainly focus on the sequential order, which results in the simple yet novel ILP of the service chaining problem. As for the bandwidth/processing demand, we use the same

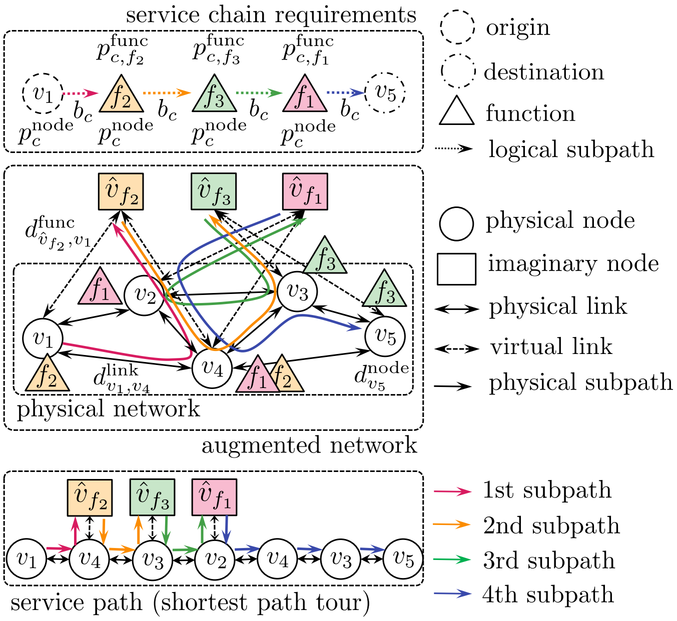


Fig. 1. Overview of service chaining: relationship between service chain requirements, augmented network, and service path ($C = 1$).

assumptions as in [8]. We consider that b_c ($b_c > 0$) is a fixed value, e.g., constant bit rate (CBR). Whenever the connection c passes through a physical node, it needs a processing capacity p^{node}_c ($p^{\text{node}}_c > 0$) to process the packets through the node. In addition, the connection c requires a processing capacity $p^{\text{func}}_{c,f_{c,k}}$ ($p^{\text{func}}_{c,f_{c,k}} > 0$) for executing each kth function $f_{c,k}$ in $\mathcal{R}_c$. An example of the service chain requirements for one connection c is shown in the top layer of Fig. 1 where $(o_c, d_c) = (v_1, v_5)$, $\mathcal{R}_c = (f_{c,1}, f_{c,2}, f_{c,3}) = (f_2, f_3, f_1)$, and bandwidth/processing requirements are given as b_c, p^{node}_c, and $\{p^{\text{func}}_{c,f}\}_{f \in \mathcal{R}_c}$, respectively.

Note that our model can also aggregate the same requests from multiple users into one connection c, as in [20]. In such cases, b_c, p^{node}_c, and $\{p^{\text{func}}_{c,f}\}_{f \in \mathcal{R}_c}$ are multiplied by the number of aggregated users in the connection c, respectively.

B. Augmented Network

We consider the NFV network relies on a physical network $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ (resp. $\mathcal{E}$) is the set of physical nodes (resp. links), as shown in the middle layer of Fig. 1. G is a directed graph where an arrow from node $i \in \mathcal{V}$ to $j \in \mathcal{V}$ expresses the physical link (i, j). For example, the bidirectional arrow between nodes v_1 and v_2 indicates that there are two physical links between them, i.e., (v_1, v_2) and (v_2, v_1), in Fig. 1. Each physical link $(i, j) \in \mathcal{E}$ (resp. each physical node $i \in \mathcal{V}$) has residual bandwidth $B_{i,j}$ (resp. residual processing capacity P_i) at the start of serving connections $c \in \mathcal{C}$.

The NFV network totally supports a set of F functions, $\mathcal{F} = \{f_1, \ldots, f_F\}$. In general, there are two kinds of physical nodes: VNF-enabled nodes ($\mathcal{V}_{\text{VNF}} \subseteq \mathcal{V}$) and normal nodes. Each VNF-enabled node $i \in \mathcal{V}_{\text{VNF}}$ is capable of (part of) F functions, $\mathcal{F}_i \subseteq \mathcal{F}$, while the normal nodes are conventional routers and switches only for data forwarding. In what follows, for simplicity, we assume that all the nodes are VNF-enable (i.e., $\mathcal{V}_{\text{VNF}} = \mathcal{V}$). From the viewpoint of each function $f \in \mathcal{F}$, f is assigned to N_f ($N_f > 0$) VNF-enabled nodes. In case of the service chaining, the location of each function f is fixed, and thus N_f is also constant. On the other hand, in case of both the service chaining and function placement, we aim to determine both the number N_f and locations of each function f according to the service chain requirements $\boldsymbol{r}_c$ of connections $c \in \mathcal{C}$.

Service chaining for the connection c with the requirements $\boldsymbol{r}_c = (o_c, d_c, \mathcal{R}_c, b_c, p^{\text{node}}_c, \{p^{\text{func}}_{c,f_{c,k}}\}_{k=1,\ldots,K_c})$ is finding an appropriate service path $\mathcal{S}_c$, which starts from o_c and ends with d_c while executing functions of $\mathcal{R}_c$ in the required order and passing through the corresponding physical nodes and links under the capacity constraints on both physical nodes and links. Inspired by the approach in [18], we can regard the service chaining as the capacitated SPTP (CSPTP). The conventional SPTP tries to find a shortest path from an origin node to a destination node with the constraint that the path should visit at least one node from each of K ($K > 0$) given disjoint node subsets $\mathcal{T}_1, \ldots, \mathcal{T}_K$ [14]. In our case, $\mathcal{T}_k = \{\hat{v}_{f_{c,k}}\}$ ($k \in \mathcal{K}_c$), where $\hat{v}_{f_{c,k}}$ is an *imaginary node* responsible for kth function $f_{c,k}$ in $\mathcal{R}_c$. For example, in Fig. 1, $\mathcal{T}_1 = \{\hat{v}_{f_2}\}, \mathcal{T}_2 = \{\hat{v}_{f_3}\}$, and $\mathcal{T}_3 = \{\hat{v}_{f_1}\}$. In addition, the service chaining should also satisfies capacity constraints on both physical nodes and links, and thus our problem can be regarded as the CSPTP.

Note that imaginary nodes are not actual virtual nodes serving the corresponding functions but they play a key role in formulating the service chaining as the CSPTP-based ILP. (The detail of formulation will be given in Section IV.) We further introduce two sets of virtual links, $\widehat{\mathcal{E}}^{\text{in}}$ and $\widehat{\mathcal{E}}^{\text{out}}$. $\widehat{\mathcal{E}}^{\text{in}}$ is a set of links incoming to imaginary nodes, $\widehat{\mathcal{E}}^{\text{in}} = \{(v, \hat{v}_f) \mid v \in \mathcal{V}, \hat{v}_f \in \widehat{\mathcal{V}}, f \in \mathcal{F}_v\}$. On the other hand, $\widehat{\mathcal{E}}^{\text{out}}$ is a set of links outgoing from imaginary nodes, $\widehat{\mathcal{E}}^{\text{out}} = \{(\hat{v}_f, v) \mid \hat{v} \in \widehat{\mathcal{V}}, v \in \mathcal{V}, f \in \mathcal{F}_v\}$. In this model, selecting an outgoing virtual link $(\hat{v}_f, v)$ from an imaginary node $\hat{v}_f$ to a physical node v as a part of the service path can represent executing the function f at the physical node v. For example, in Fig. 1, the virtual link $(\hat{v}_{f_2}, v_1)$ indicates that physical node v_1 is capable of function f_2. Note that physical node v_4 is also capable of function f_2 in this example. Selecting the virtual link $(\hat{v}_{f_2}, v_1)$ (resp. $(\hat{v}_{f_2}, v_4)$) means that function f_2 will be executed at physical node v_1 (resp. v_4).

We call the finally obtained network the *augmented network* $G^+ = (\mathcal{V}^+, \mathcal{E}^+)$ where $\mathcal{V}^+ = \mathcal{V} \cup \widehat{\mathcal{V}}$ and $\mathcal{E}^+ = \mathcal{E} \cup \widehat{\mathcal{E}}^{\text{in}} \cup \widehat{\mathcal{E}}^{\text{out}}$. An example of the augmented network is shown in the middle layer of Fig. 1. For simplicity in description, the neighbors of node i in G^+ is defined as $\mathcal{V}^+_i \subseteq \mathcal{V}^+$.

At the last of this section, we clarify the difference between the augmented network and the existing network models: layered graph [9] and expanded network [8]. Fig. 2 illustrates the structure comparison among the three network models in case of the service chaining with $C = 1$. (The comparison in case of both the service chaining and function placement will be given in Section IV-B.) The layered graph, which is given in the right of Fig. 2, consists of $K_c + 1$ layers of the original physical network and multiple vertical links connecting two successive layers. A vertical link exists between the node

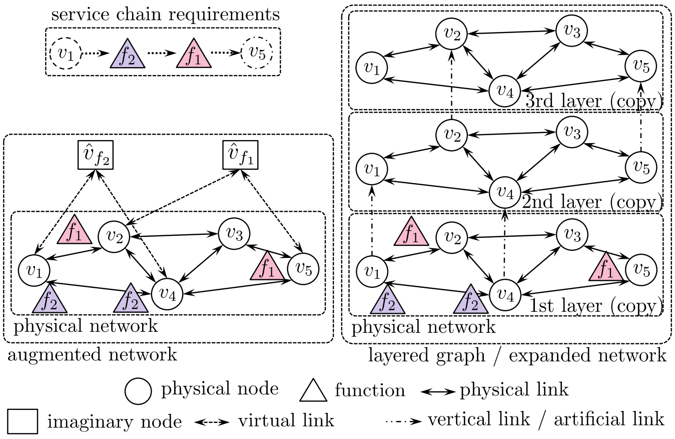


Fig. 2. Structure comparison among network models (service chaining case with $C = 1$).

TABLE II
SCALE COMPARISON AMONG NETWORK MODELS (SERVICE CHAINING CASE).

Network model	# of nodes	# of links
Layered graph	$\sum_{c\in\mathcal{C}}(K_c+1)V$	$\sum_{c\in\mathcal{C}}((K_c+1)E+K_cN)$
Expanded net.	$\sum_{c\in\mathcal{C}}(K_c+1)V$	$\sum_{c\in\mathcal{C}}((K_c+1)E+K_cN)$
Augmented net.	$V+\lvert\cup_{c\in\mathcal{C}}\mathcal{R}_c\rvert$	$E+2\lvert\cup_{c\in\mathcal{C}}\mathcal{R}_c\rvert N$

$i \in \mathcal{V}$ at kth layer and that at $(k+1)$th layer ($k = 1, \ldots, K_c$) only when the node i is capable of the kth function $f_{c,k}$ of the chain request c. The expanded network is an extension of the layered graph by adding the pruning process. The pruning process excludes some links from the network by considering the fact that each physical link (resp. node) should have the bandwidth (resp. processing) capacity for at least one traversal of the connection c. Note that the actual number of times that a certain link/node is used in the service path may be more than one. In other words, the pruned network may still have physical links (nodes) without sufficient capacity for supporting the service path. However, the frequency of link/node usage cannot be identified until calculating the service path, which makes the problem more complex. We will see an example of multiple usage of the same link in the service path. (Please see Section III-C.)

Table II presents the scale comparison among the network models in case of the service chaining. For simplicity, we assume that $N_f = N$ ($N > 0, f \in \mathcal{F}$). Note that the number of links in case of the expanded network is shown as its upper limit where all the nodes and links are available for the connection request. We can confirm that the augmented network becomes smaller than the layered graph and expanded network with increase of K_c. Furthermore, all the networks are basically constructed per requested connections $\mathcal{C}$ (i.e., in an *on-demand* manner). In case of the augmented network, however, we can also construct a *full* augmented network that consists of physical nodes $\mathcal{V}$, physical links $\mathcal{E}$, and all F imaginary nodes with the corresponding $2N$ incoming/outgoing virtual links. Since the full augmented network can support any kind of connection request $\mathcal{C}$, we can save both the construction time and memory space of the network at the risk of increasing computation complexity.

C. *Service Path*

According to the SPTP, the service path $\mathcal{S}_c$ for $\mathcal{R}_c = (f_{c,1}, \ldots, f_{c,K_c})$ with the origin o_c and destination d_c can be expressed by a sequence of $K_c + 1$ subpaths, $(\mathcal{S}_{c,1}, \ldots, \mathcal{S}_{c,K_c+1})$, where the kth subpath $\mathcal{S}_{c,k}$ has the origin node $a_{c,k}$ and destination node $b_{c,k}$, which are given as follows:

$$(a_{c,k}, b_{c,k}) = \begin{cases} (o_c, \hat{v}_{f_{c,1}}), & k = 1, \\ (\hat{v}_{f_{c,k-1}}, \hat{v}_{f_{c,k}}), & k = 2, \ldots, K_c, \\ (\hat{v}_{f_{c,K_c}}, d_c), & k = K_c + 1. \end{cases}$$

$\mathcal{S}_{c,k}$ starts from its origin node $a_{c,k}$ and ends with its destination node $b_{c,k}$ through appropriate physical and virtual links in G^+. (Finding an optimal service path $\mathcal{S}_c^*$ is our goal and will be discussed in Section IV.) Note that there is no loop in each subpath but loop(s) may occur in the whole path, that is, some links may be used multiple times, which makes the service chaining problem more complex. For example, the bottom layer of Fig. 1 shows an example of the service path $\mathcal{S}_c = (\mathcal{S}_{c,1}, \ldots, \mathcal{S}_{c,4})$ where $\mathcal{S}_{c,1} = ((v_1, v_4), (v_4, \hat{v}_{f_2}))$ (red arrow), $\mathcal{S}_{c,2} = ((\hat{v}_{f_2}, v_4), (v_4, v_3), (v_3, \hat{v}_{f_3}))$ (orange arrow), $\mathcal{S}_{c,3} = ((\hat{v}_{f_3}, v_3), (v_3, v_2), (v_2, \hat{v}_{f_1}))$ (green arrow), and $\mathcal{S}_{c,4} = ((\hat{v}_{f_1}, v_2), (v_2, v_4), (v_4, v_3), (v_3, v_5))$ (blue arrow). We can also confirm the service path $\mathcal{S}_c$ in the augmented network, as shown in the middle layer of Fig.1. In this case, each subpath (i.e., $\mathcal{S}_{c,1}, \mathcal{S}_{c,2}, \mathcal{S}_{c,3}$, and $\mathcal{S}_{c,4}$) does not have any loop while the whole service path $\mathcal{S}_c$ has a loop. (The physical link (v_4, v_3) is used twice.)

We consider that the optimality of service path $\mathcal{S}_c$ is evaluated by total delay consisting of processing delay at nodes and propagation delay at physical links included in $\mathcal{S}_c$. (The detail will be given in Section IV.) Each physical link $(i, j) \in \mathcal{E}$ has the propagation delay $d_{i,j}^{\text{link}}$ ($d_{i,j}^{\text{link}} > 0$). As mentioned above, the connection c requires packet processing at each physical node that it uses, and thus the corresponding processing delay at the physical node $i \in \mathcal{V}$ is given by d_i^{node} ($d_i^{\text{node}} > 0$). In $\mathcal{S}_c$, each function $f \in \mathcal{R}_c$ is executed at the corresponding physical node v with the processing delay of $d_{\hat{v}_f,v}^{\text{func}}$ ($d_{\hat{v}_f,v}^{\text{func}} > 0$).

IV. Modeling Service Chaining and Function Placement as Capacitated Shortest Path Tour Problem Based Integer Linear Program

In this section, we propose two kinds of CSPTP-based ILPs: one is only for the service chaining, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$, in Section IV-A, and another is for both the service chaining and function placement, $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, in Section IV-B. We further discuss the applicability of them to other NFV-related problems and the computational complexity of the proposed ILPs in Section IV-C and Section IV-D, respectively.

A. *CSPTP-based ILP for Service Chaining*

Inspired by the ILP for the constrained SPTP in [18], we first formulate the CSPTP-based ILP for the service chaining,

$\text{ILP}^{\text{CSPTP}}_{\text{SC}}$, which has the binary decision variables $x^{c,k}_{i,j}$ $(c \in \mathcal{C}, (i,j) \in \mathcal{E}^+, k \in \mathcal{K}^+_c)$:

$$x^{c,k}_{i,j} = \begin{cases} 1, & \text{if physical/virtual link } (i,j) \text{ is used in } k\text{th} \\ & \text{subpath of service path for connection } c, \\ 0, & \text{otherwise.} \end{cases}$$

$$\min \sum_{c\in\mathcal{C}} \sum_{(i,j)\in\mathcal{E}^+} d_{i,j} \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{i,j} \tag{1}$$

$$\text{s.t. } x^{c,k}_{i,j} = \{0,1\}, \qquad (i,j) \in \mathcal{E}^+, c \in \mathcal{C}, k \in \mathcal{K}^+_c, \tag{2}$$

$$\sum_{j\in\mathcal{V}^+_{a_{c,k}}} x^{c,k}_{a_{c,k},j} = 1, \qquad c \in \mathcal{C}, k \in \mathcal{K}^+_c, \tag{3}$$

$$\sum_{j\in\mathcal{V}^+_{b_{c,k}}} x^{c,k}_{j,b_{c,k}} = 1, \qquad c \in \mathcal{C}, k \in \mathcal{K}^+_c, \tag{4}$$

$$\sum_{j\in\mathcal{V}^+_i} x^{c,k}_{j,i} = \sum_{j\in\mathcal{V}^+_i} x^{c,k}_{i,j}, \qquad i \in \mathcal{V} \setminus \{a_{c,k}, b_{c,k}\}, c \in \mathcal{C}, k \in \mathcal{K}^+_c, \tag{5}$$

$$x^{c,k}_{i,\hat{v}_{f_{c,k}}} = x^{c,k+1}_{\hat{v}_{f_{c,k}},i}, \qquad (i,\hat{v}_{f_{c,k}}) \in \widehat{\mathcal{E}}^{\text{in}}, (\hat{v}_{f_{c,k}}, i) \in \widehat{\mathcal{E}}^{\text{out}}, c \in \mathcal{C}, k \in \mathcal{K}_c, \tag{6}$$

$$x^{c,k}_{i,\hat{v}_{f_{c,m}}} = 0, \qquad (i,\hat{v}_{f_{c,m}}) \in \widehat{\mathcal{E}}^{\text{in}}, c \in \mathcal{C}, k \in \mathcal{K}^+_c, \hat{v}_{f_{c,m}} \neq b_{c,k}, \tag{7}$$

$$\sum_{c\in\mathcal{C}} \left(b_c \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{i,j} \right) \leq B_{i,j}, \qquad (i,j) \in \mathcal{E}, \tag{8}$$

$$\sum_{c\in\mathcal{C}} \left(p^{\text{node}}_c \sum_{(v,j)\in\mathcal{E}} \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{v,j} + \sum_{(\hat{v}_f,v)\in\widehat{\mathcal{E}}^{\text{out}}} p^{\text{func}}_{c,f} \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{\hat{v}_f,v} \right) \leq P_v, \qquad v \in \mathcal{V}. \tag{9}$$

The objective function (1) is the minimization of the total delay of all the service paths where $d_{i,j}$ is given as follows:

$$d_{i,j} = \begin{cases} d^{\text{node}}_i + d^{\text{link}}_{i,j}, & \text{if } (i,j) \in \mathcal{E}, \\ d^{\text{func}}_{i,j}, & \text{if } (i,j) \in \widehat{\mathcal{E}}^{\text{out}}, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

We first observe that the objective function (1) is the same as that of the SPTP. From the viewpoint of service chaining, (10) indicates that passing each physical link $(i,j) \in \mathcal{E}$ suffers both forwarding delay d^{node}_i and propagation delay $d^{\text{link}}_{i,j}$. The service path also suffers the execution delay of each function $i \in \mathcal{R}_c$ at the corresponding physical node j. As a result, the objective function (1) can be rewritten by

$$\sum_{c\in\mathcal{C}} \left(\sum_{(i,j)\in\mathcal{E}} (d^{\text{node}}_i + d^{\text{link}}_{i,j}) \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{i,j} + \sum_{(\hat{v}_f,v)\in\widehat{\mathcal{E}}^{\text{out}}} d^{\text{func}}_{\hat{v}_f,v} \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{\hat{v}_f,v} \right),$$

where the first (resp. second) term corresponds to the physical forwarding and propagation delay (resp. function execution delay). For example, in the bottom of Fig. 1, the horizontal (resp. vertical) arrows correspond to the physical (resp. virtual) links included in service path $\mathcal{S}_c$. Note that the objective function can be transformed into the minimum-cost flow problem by replacing $d_{i,j}$ with node/link utilization as in [8]:

$$d_{i,j} = \begin{cases} \dfrac{b_c}{B_{i,j}}, & \text{if } (i,j) \in \mathcal{E}, \\ \dfrac{p^{\text{func}}_{c,f(i)}}{P_i}, & \text{if } (i,j) \in \widehat{\mathcal{E}}^{\text{out}}, \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

where $f(i)$ represents the function for which the imaginary node i is responsible.

Constraints are given by (2)–(9). Constraint (2) represents the binary decision variables. Constraints (3)–(5) present the flow rules in each subpath $\mathcal{S}_{c,k}$ $(c, \in \mathcal{C}, k \in \mathcal{K}^+_c)$. Constraint (3) (resp. Constraint (4)) indicates that the origin (resp. destination) node $a_{c,k}$ (resp. $b_{c,k}$) of the connection c's subpath k has the outgoing (resp. incoming) flow. Constraint (5) is the flow conservation rule at each intermediate node i in the connection c's subpath k $(c, \in \mathcal{C}, k \in \mathcal{K}^+_c)$. For example, focusing on the 1st subpath in Fig. 1, we observe that the flow occurs at the physical node v_1 (i.e., $\sum_{j\in\mathcal{V}^+_{v_1}} x^{c,1}_{v_1,j} = 1$), goes through any physical node v (i.e,. $\sum_{j\in\mathcal{V}^+_v} x^{c,1}_{j,v} = \sum_{j\in\mathcal{V}^+_v} x^{c,1}_{v,j}$), and finally ends with the imaginary node $\hat{v}_{f_2}$ (i.e., $\sum_{j\in\mathcal{V}^+_{\hat{v}_{f_2}}} x^{c,1}_{j,\hat{v}_{f_2}} = 1$).

Constraint (6) guarantees the connectivity between two successive subpaths $\mathcal{S}_{c,k}$ and $\mathcal{S}_{c,k+1}$ of the connection c $(c, \in \mathcal{C}, k \in \mathcal{K}_c)$. More specifically, the connection c's $(k+1)$th subpath should start from the same physical node as the final physical node of the connection c's kth subpath. For example, focusing on the 1st and 2nd subpaths in Fig. 1, we observe that $x^{c,1}_{i,\hat{v}_{f_{c,1}}} = x^{c,2}_{\hat{v}_{f_{c,1}},i}$ $(\forall(i,\hat{v}_{f_{c,1}}) \in \widehat{\mathcal{E}}^{\text{in}}, \forall(\hat{v}_{f_{c,1}}, i) \in \widehat{\mathcal{E}}^{\text{out}})$ where $f_{c,1} = f_2$. Constraint (7) prohibits the imaginary node $\hat{v}_{f_{c,m}}$ from being used in the kth subpath $(m \neq k)$. For example, focusing on the 1st subpath in Fig. 1, we observe that $x^{c,1}_{i,\hat{v}_{f_{c,m}}} = 0$ $(m \neq 1, \forall(i,\hat{v}_{f_{c,m}}) \in \widehat{\mathcal{E}}^{\text{in}})$.

Constraint (8) gives the physical link capacity constraint where the total bandwidth consumption of the physical link $(i,j) \in \mathcal{E}$ should be equal or less than the residual bandwidth capacity $B_{i,j}$. The ratio of the left side to the right side of (8) is the utilization $u_{i,j}$ of the physical link (i,j). Similarly, constraint (9) shows the processing capacity constraint where the total processing load of the physical node $v \in \mathcal{V}$ should be equal or less than the processing capacity P_v. Note that the processing load consists of the traversal cost, $\sum_{c\in\mathcal{C}}(p^{\text{node}}_c \sum_{(v,j)\in\mathcal{E}} \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{v,j})$, and the processing cost, $\sum_{c\in\mathcal{C}}(\sum_{(\hat{v}_f,v)\in\widehat{\mathcal{E}}^{\text{out}}} p^{\text{func}}_{c,f} \sum_{k\in\mathcal{K}^+_c} x^{c,k}_{\hat{v}_f,v})$. The ratio of the left side to the right side of (9) is the utilization u_v of the physical node v.

B. CSPTP-based ILP for Service Chaining and Function Placement

The CSPTP-based ILP for the service chaining problem $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ in Section IV-A can easily be extended to an optimization problem, $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$, which considers not only the service chaining but also the function placement, in the following manner. Please first remember that selecting an outgoing virtual link $(\hat{v}_f, v)$ from an imaginary node $\hat{v}_f$ to a

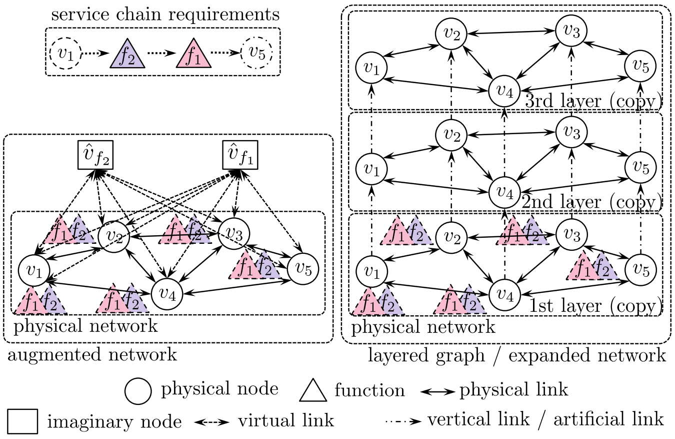


Fig. 3. Structure comparison among network models (service chaining and function placement case with $C = 1$).

TABLE III
SCALE COMPARISON AMONG NETWORK MODELS (SERVICE CHAINING AND FUNCTION PLACEMENT).

Network model	# of nodes	# of links
Layered graph	$\sum_{c\in\mathcal{C}}(K_c+1)V$	$\sum_{c\in\mathcal{C}}((K_c+1)E+K_cN)$
Expanded net.	$\sum_{c\in\mathcal{C}}(K_c+1)V$	$\sum_{c\in\mathcal{C}}((K_c+1)E+K_cN)$
Augmented net.	$V+\lvert\cup_{c\in\mathcal{C}}\mathcal{R}_c\rvert$	$E+2\lvert\cup_{c\in\mathcal{C}}\mathcal{R}_c\rvert N$

physical node v at kth subpath of connection c (i.e., $x^{c,k}_{\hat{v}_f,v}=1$) represents executing the function f at the physical node v, as mentioned in Section III-B. Moreover, virtual links only exist between imaginary nodes and physical nodes capable of the corresponding functions in the augmented network. In other words, connecting each function (imaginary node) to all VNF-enabled physical nodes via virtual links, i.e., $\widehat{\mathcal{E}}^{\rm in}=\{(v,\hat{v}_f)\mid v\in\mathcal{V}_{\rm VNF},\hat{v}_f\in\widehat{\mathcal{V}},f\in\mathcal{F}\},\widehat{\mathcal{E}}^{\rm out}=\{(\hat{v}_f,v)\mid\hat{v}\in\widehat{\mathcal{V}},v\in\mathcal{V}_{\rm VNF},f\in\mathcal{F}\}$, can consider all possibilities of function placement at the expense of the computation complexity. Once the ILP$^{\rm CSPTP}_{\rm SCFP}$ is solved, we can determine the actual location of each function $f\in\mathcal{F}$ on the corresponding VNF-enabled physical node $v\in\mathcal{V}_{\rm VNF}$ with $x^{c,k}_{\hat{v}_f,v}=1$ $(c\in\mathcal{C},k\in\mathcal{K}_c)$. For simplicity, we consider $\mathcal{V}_{\rm VNF}=\mathcal{V}$ in what follows.

Fig. 3 illustrates an example of the augmented network for both the service chaining and function placement with $C=1$, which is an expanded version of that only for the service chaining in Fig. 2. For comparison purpose, we also show the corresponding example of the layered graph and expanded network. Comparing Fig. 3 with Fig. 2, we can confirm that all the physical nodes are capable of all the VNFs (i.e., f_1 and f_2) and the corresponding virtual links (resp. vertical links/artificial links) exist in case of the augmented network (resp. layered graph and expanded network). Please note that all the functions at the physical layer are presented by dotted triangles, which means that they are candidates of their locations and the actual locations will be determined by solving the corresponding ILP. Table III shows the corresponding number of nodes and links in case of the augmented network and expanded network. Comparing Table II with Table III, we observe that the number of nodes is identical while the number of virtual links (resp. artificial links) increases in case of the augmented network (resp. expanded network).

C. Applicability to Other NFV Problems

In this paper, we mainly focus on clarifying the relationship between the NFV-related problems (i.e., service chaining and function placement) and the conventional SPTP. Someone may have concerns about the applicability of the proposed approach to other NFV problems with different objective functions and/or constraints. In ILP$^{\rm CSPTP}_{\rm SC}$, the SPTP-related constraints consist of three parts: 1) the flow rules in each subpath, given by constraints (3)–(5), 2) the connectivity between two successive subpaths, given by constraint (6), and 3) the prohibition of utilizing unnecessary imaginary node, given by constraint (7). Other constraints except the decision variables, i.e., (8) and (9), are the constraints on link and node capacities, which contribute to formulate the service chaining problem as the CSPTP.

In other words, keeping these essential constraints for the CSPTP, we still have some room to add/modify the objective function and/or constraints to support other objectives. For example, the utilization u_i of physical node $i\in\mathcal{V}$, which was derived in Section IV-A, can also be used to express the energy consumption proportional to the amount of resources being used, as in [33]. Since a server usually consumes power even in the idle state, reduction of the number of active physical nodes, which are included in at least one service path and serve forwarding and/or function execution, will also contribute to energy saving [20], [33]. In our case, the number of active physical nodes can be expressed by

$$\sum_{i\in\mathcal{V}} y_i,$$

where binary decision variables $y_i=\{0,1\}$ $(i\in\mathcal{V})$ should satisfy the following:

$$x^{c,k}_{i,j}\le y_i,\qquad i\in\mathcal{V},(i,j)\in\mathcal{E}^+,c\in\mathcal{C},k\in\mathcal{K}^+_c.\quad(12)$$

Constraint (12) indicates that if $x^{c,k}_{i,j}=1$, the physical node i is included in the kth subpath of the connection c, and thus it should be active (i.e., $y_i=1$). On the other hand, y_i can be zero only if the physical node i is not included in any service path $\mathcal{S}_c$ $(c\in\mathcal{C})$. The original objective function (1) can be extended to a multi-objective function, e.g., weighted sum of total path delay and the number of active nodes.

D. Computational Complexity

In this section, we discuss the computational complexity of the proposed ILPs, which can be regarded as a variant of SPTP, i.e., CSPTP. At first, it was proved that the SPTP belongs to the complexity class **P** [13]. Ferone et al. further proved that the constrained SPTP belongs to the complexity class **NP**-complete [17]. As mentioned in Section I, the capacitated SPTP is more complicated than the constrained SPTP because it has more general constraints on both node and link capacities with real values. Note that a special case of our problem with $b_c=1$ $(c\in\mathcal{C})$, $B_{i,j}=1$ $(i,j\in\mathcal{E})$, and $P_v=\infty$ $(v\in\mathcal{V})$ belongs to the constrained SPTP.

Algorithm 1 Greedy-based heuristic algorithm.

```
Input: set $\mathcal{C}$ of connections with requirements $\{\boldsymbol{r}_c\}_{c\in\mathcal{C}}$, physical network $G = (\mathcal{V}, \mathcal{E})$ with attribute information about bandwidth $\{B_{i,j}\}_{(i,j\in\mathcal{E})}$ and processing capacity $\{P_i\}_{i\in\mathcal{V}}$
Output: set $\mathcal{S}$ of service paths for all connections $\mathcal{C}$
 1: $\mathcal{S} \leftarrow \emptyset$
 2: for $c \in \mathcal{C}$ do
 3:     $\mathcal{S}_c \leftarrow \emptyset$, $\mathcal{S}_{c,0} \leftarrow \emptyset$
 4:     for $k = 1$; $k \leq K_c + 1$; $K$++ do
 5:         update_residual_network$(G, \boldsymbol{r}_c, \mathcal{S}_{c,k-1})$
 6:         $G^+ \leftarrow$ get_augmented_network$(G, f_{c,k})$
 7:         pruning$(G^+, \boldsymbol{r}_c)$
 8:         if $k = 1$ then          ▷ obtain origin of 1st subpath
 9:             $o_{c,k} \leftarrow o_c$
10:         else                      ▷ obtain origin of $k$th subpath
11:             $o_{c,k} \leftarrow$ get_origin_node$(\mathcal{S}_{c,k-1}, \hat{v}_{f_{c,k-1}})$
12:         if $k = K_c + 1$ then     ▷ obtain destination of last subpath
13:             $d_{c,k} \leftarrow d_c$
14:         else                      ▷ obtain destination of $k$th subpath
15:             $d_{c,k} \leftarrow \hat{v}_{f_{c,k}}$
16:         $\mathcal{S}_{c,k} \leftarrow$ get_shortest_path$(G^+, o_{c,k}, d_{c,k})$
17:         if $S_{c,k} == \emptyset$ then return $\emptyset$ ▷ no subpath is found
18:         concat$(S_c, S_{c,k})$    ▷ concatenate the subpaths into the service path
19:     $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}_c$
    return $\mathcal{S}$
```

In Section VI, we will demonstrate that the proposed ILPs can directly and speedily be solved by the existing solver, CPLEX, under a certain scale of systems (e.g., a thousand physical nodes). For further scalability, we also propose a greedy-based heuristic algorithm in Section V.

V. Heuristic Algorithm

As mentioned in Section III-C, a service path may traverse the same link multiple times, which makes the service chaining problem more complex in terms of the constraints on node and link capacities, and thus the simple shortest path algorithm, e.g., Dijkstra algorithm, cannot directly be applied. On the other hand, each subpath in a service path does not include such loop(s), which means that the capacity constraints can easily be guaranteed by considering the residual node/link capacities of the physical network and the processing/bandwidth requirements of the service path.

Considering these facts, we propose a simple greedy-based heuristic algorithm where we sequentially calculates the kth subpath $(k = 1, \ldots, K_c{+}1)$ for connection c using the shortest path algorithm under the capacity constraints. If more than one connections are requested $(C \geq 2)$, we also conduct the sequential calculation of each service path. Algorithm 1 gives the corresponding pseudo code.

Given the set $\mathcal{C}$ of connections with service chain requirements $\{\boldsymbol{r}_c\}_{c\in\mathcal{C}}$ and physical network $G = (\mathcal{V}, \mathcal{E})$ with attribute information about bandwidth $\{B_{i,j}\}_{(i,j\in\mathcal{E})}$ and processing capacity $\{P_i\}_{i\in\mathcal{V}}$, the orchestrator first initializes the set $\mathcal{S}$ of service paths for all the connections $\mathcal{C}$ to be empty. Then, it starts to find the service path $\mathcal{S}_c$ for the first connection c by sequentially calculating its subpath from the first $(k = 1)$ to the last $(k = K_c + 1)$. It first updates the residual capacities of nodes and links of the physical network G using update_residual_network$(G, \boldsymbol{r}_c, \mathcal{S}_{c,k-1})$ function (line 5). In this function, the processing capacities of nodes (resp. bandwidth of links) used in the last subpath $\mathcal{S}_{c,k-1}$ are reduced according to the processing (resp. bandwidth) requirements included in $\boldsymbol{r}_c$. Note that in case of $k = 1$, there is no last subpath, i.e., $\mathcal{S}_{c,0} = \emptyset$, and thus no update will occur. Next, the orchestrator creates an augmented network G^+ for kth subpath using get_augmented_network$(G, f_{c,k})$ function (line 6). It further conducts pruning$(G^+, \boldsymbol{r}_c)$ function to update the augmented network G^+ by removing nodes (resp. links) that cannot satisfy the processing (resp. bandwidth) requirement for kth subpath of connection c (line 7).

The orchestrator calculates the kth subpath from the origin node $o_{c,k}$ to the destination node $d_{c,k}$ using get_shortest_path$(G^+, o_{c,k}, d_{c,k})$ function (line 16). Here, $o_{c,k}$ and $d_{c,k}$ are appropriately assigned according to the value of k (lines 12–15). Here, get_origin_node$(\mathcal{S}_{c,k-1}, \hat{v}_{f_{c,k-1}})$ function gives the last physical node in the $(k-1)$th subpath $S_{c,k-1}$, which executes $(k-1)$th function $f_{c,k-1}$. If the orchestrator successfully finds the kth subpath $S_{c,k}$, it concatenates $S_{c,k}$ to the service path $\mathcal{S}_c$ using concat$(\mathcal{S}_c, \mathcal{S}_{c,k})$ function (line 18). Otherwise, it fails in service chaining, and thus it returns $\mathcal{S} = \emptyset$ (line 17). This process will be repeated $K_c + 1$ times to find the service path for each connection $c \in \mathcal{C}$. Please note that Algorithm 1 can also support both the service chaining and function placement by preparing the augmented network according to the procedure in Section IV-B.

Finally, we discuss the computational complexity of Algorithm 1. At first, we focus on the loop from line 4 to 18. One loop mainly consists of three processes: 1) management of augmented network in lines 5–7, which requires $O(V^+{+}E^+)$, 2) calculation of origin and destination in lines 8–15, which requires $O(V^+)$, and 3) calculation of the shortest path in line 16, which requires $O(E^+ + V^+ \log V^+)$ that is the computational complexity of Dijkstra algorithm implemented with a Fibonacci heap [34]. Therefore, the computational complexity of one loop is dominated by $O(E^+ + V^+ \log V^+)$ and at most $K_c{+}1$ loops will be required for each connection $c \in \mathcal{C}$. Since the loop for all connections $c \in \mathcal{C}$ is conducted from line 2 to 19, Algorithm 1 runs in strongly polynomial time of $O(C(K_c + 1)(E^+ + V^+ \log V^+))$.

VI. Numerical Results

In this section, we evaluate the effectiveness of the proposed CSPTP-based ILP for the service chaining ($\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SC}}$), that for both the service chaining and function placement ($\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$), the greedy-based heuristic approach for the service chaining ($\mathrm{Greedy}_{\mathrm{SC}}$), and that for both the service chaining and function placement ($\mathrm{Greedy}_{\mathrm{SCFP}}$). We used the existing solver CPLEX 12.8 to solve the ILPs. We also implemented Algorithm 1 in C++ programming language with the boost graph library [35] where `dijkstra_shortest_paths` function implemented with a Fibonacci heap is provided. In the calculation, we used the server with Intel Xeon E7-8895v3 (18 cores and 2.60 GHz) and 2 TB memory.

A. Evaluation Scenario

We use the physical network consisting of 200 physical nodes and physical links. The physical links are randomly generated between two arbitrary physical nodes at the probability of $\pi = 0.032$ as in [36]. As for the physical node (resp. link) capacity, we assume that each physical node $i \in \mathcal{V}$ (resp. physical link between physical nodes $i \in \mathcal{V}$ and $j \in \mathcal{V} \setminus \{i\}$) has the same capacity $P_i = 1.71$ (resp. $B_{i,j} = 1.14$). We set the physical link delay between physical nodes i and j, $d^{\text{link}}_{i,j}$, to be 10 [ms]. Each physical node v has the same traversal and processing delay, i.e., $d^{\text{node}}_v = 1$ [ms] and $d^{\text{func}}_{v_f,v} = 50$ [ms] $(v_f \in \mathcal{F}_v)$, respectively.

We consider a single-round resource allocation for online, batch, and offline processing. For each connection $c \in \mathcal{C}$, the origin node o_c and destination node d_c are randomly chosen from the physical nodes. Each connection c requires a set $\mathcal{R}_c$ of K_c functions, each of which is selected from the set of F functions. As mentioned in Section III-B, the number N_f of physical nodes capable of the function $f \in \mathcal{F}$ is fixed in case of the service chaining while that is adaptive in case of both the service chaining and function placement. The settings for $\mathcal{R}_c$, F, and N_f will be different in each scenario and explained later. We set the bandwidth requirement, processing requirement for traversing a node, and processing requirement of executing $f_{c,k} \in \mathcal{R}_c$ at a node as follows: $b_c = 0.1, p^{\text{node}}_c = 0.05$, and $p^{\text{func}}_{c,f_{c,k}} = 0.1$.

In the evaluation, we mainly focus on the following three points: 1) the scalability of $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ itself through comparison with the existing ILP over the expanded network ($\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$) [8], 2) the quantitative evaluation of performance improvement and computational overhead of $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$ through comparison with $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$, and 3) the performance difference between the CSPTP-based ILPs and the greedy-based heuristic approaches.

As for the scalability, we evaluate the computational complexity in terms of the execution time, which is the actual time required to solve (resp. execute) the ILP (resp. heuristic algorithm). Since the execution time and the objective value may change even under the same number of physical nodes, due to the difference of the topological structure, we show the average of 200 independent numerical experiments with 95% confidence interval in what follows. Note that CPLEX supports the parallel optimization and we set the number of threads to be 32.

As for the service chaining, we compare $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$, $\text{Greedy}_{\text{SC}}$, and $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$. Here, we select $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$ for comparison to reveal the effectiveness of the ILP formulation over the augmented network because there is no SPTP-based ILP formulation other than the proposed ILP, $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$, as mentioned in Section I and $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$ has a similar structure to $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ except the network structure (i.e., expanded network). As for $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$, the objective function was originally designed as the minimization of the link and node utilization to alleviate the blocking probability of connection requests [8]. In what follows, focusing on the computation complexity of the ILP itself, we slightly replace the objective function of $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$ with that of $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$, i.e., the minimization of the total delay given by (1).

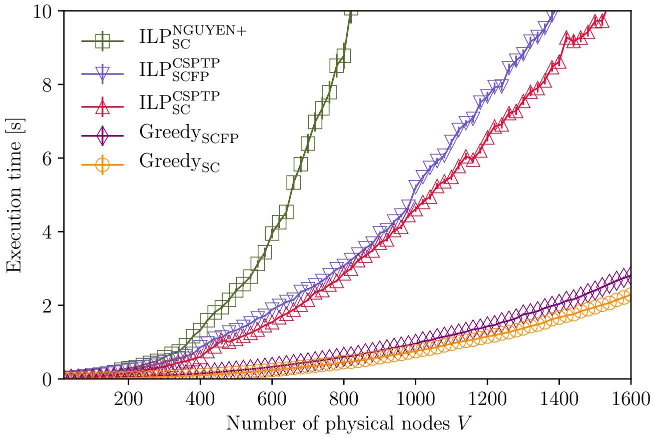


Fig. 4. Impact of the number of physical nodes on execution time.

As for the effectiveness of both the service chaining and function placement, we evaluate $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$ and $\text{Greedy}_{\text{SCFP}}$ in terms of the utilization of physical nodes/links, which are given in Section IV, as well as the total delay of all the service paths (i.e., objective value).

B. Scalability

1) Scalability to Number of Physical Nodes: Fig. 4 presents how the execution time increases with the number of physical nodes, V, where $C = 1$, $K_c = 5$, and $F = 20$. Note that K_c functions in $\mathcal{R}_c$ are randomly chosen from F functions such that $f_{c,k} \neq f_{c,m}$ $(k \neq m)$. In addition, $N_f = N = 5$ $(f \in \mathcal{F})$ for the service chaining problems. We first focus on the ILPs only for the service chaining, i.e., $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ and $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$. We observe that the execution time of all the ILPs exponentially grows with increase of V, but the increasing rate of $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ is much smaller than that of $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$. As a result, $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ can support 1.22–1.90 as times large-scale systems as $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$. For example, $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ (resp. $\text{ILP}^{\text{NGUYEN+}}_{\text{SC}}$) can solve the service chaining problem with $V = 1380$ (resp. $V = 800$) within the execution time limit of 10 [s]. As mentioned in Section III-B, the augmented network is more compact than the expanded network, which contributes to the scalability.

Next, we compare the ILP for both the service chaining and function placement, i.e., $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$, with $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$. We observe that $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$ (resp. $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$) can support 420 and 1380 (resp. 440 and 1520) physical nodes within 1 and 10 execution time [s], respectively. Although $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$ has a more complex structure than $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$, we observe that the increase of the execution time of $\text{ILP}^{\text{CSPTP}}_{\text{SCFP}}$ is suppressed by 16.6% than that of $\text{ILP}^{\text{CSPTP}}_{\text{SC}}$ when $V = 1380$.

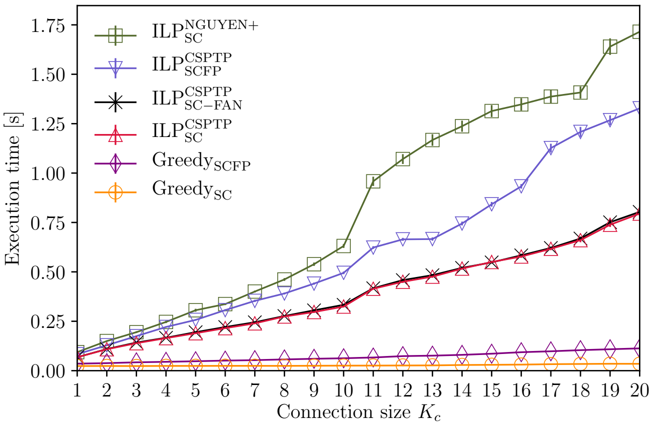


Fig. 5. Impact of connection size K_c on execution time.

Finally, we focus on the greedy-based heuristic approaches, i.e., $\mathrm{Greedy}_{\mathrm{SC}}$ and $\mathrm{Greedy}_{\mathrm{SCFP}}$. As mentioned in Section V, the execution time of both the greedy-based heuristic approaches follows $O(C(K_c+1)(E^+ + V^+ \log V^+))$, and thus $\mathrm{Greedy}_{\mathrm{SC}}$ (resp. $\mathrm{Greedy}_{\mathrm{SCFP}}$) decreases the execution time by 79.3% (resp. 79.9%), compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ (resp. $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$) when $V = 1520$ (resp. $V = 1380$). Please note that the greedy-based heuristic approaches may not be able to achieve the optimal solution. We will discuss the optimality of the greedy-based heuristic approaches in Section VI-C.

2) Scalability to Connection Size: Fig. 5 shows the impact of the connection size K_c on the execution time when $V = 200, C = 1$, and $F = 20$. Note that $N_f = N = 5$ $(f \in \mathcal{F})$ for the service chaining problems. In this evaluation, we set $\forall f \in \mathcal{R}_c$ to be different each other. As mentioned in Section III-B, the augmented network can be constructed as the full version. Therefore, we further show the results of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ over the full augmented network ($\mathrm{ILP}_{\mathrm{SC-FAN}}^{\mathrm{CSPTP}}$), in addition to those of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$, and $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{NGUYEN+}}$. We first observe that the execution time almost linearly grows with increase of K_c, regardless of the ILP formulation. However, we also confirm that the execution time of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ is always smaller than that of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{NGUYEN+}}$. Specifically, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ can reduce the execution time by 27.5% ($K_c = 1$) and 53.7% ($K_c = F$) compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{NGUYEN+}}$.

We also observe that the performance difference between $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SC-FAN}}^{\mathrm{CSPTP}}$ is limited, which indicates that $\mathrm{ILP}_{\mathrm{SC-FAN}}^{\mathrm{CSPTP}}$ can deal with any kinds of connection requests with the pre-constructed full augmented network while suppressing the increase of the execution time.

Next, we focus on the performance difference between $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$. Note that N_f $(f \in \mathcal{K}_c)$ is fixed to be $N = 5$ in case of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ while that is adjusted according to the demand in case of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$. Since C is set to be one in this evaluation, N_f $(f \in \mathcal{K}_c)$ also becomes one in case of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$. We observe that the increasing rate of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ is larger than that of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$. This result mainly stems from the different size of the augmented network between $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$. The number of the imaginary links of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ (resp. $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$) increases by $2V$ (resp. $2N$) per connection size from Table III (resp. Table II).

TABLE IV
SERVICE CHAIN REQUIREMENTS [9], [20] (NAT: NETWORK ADDRESS TRANSLATOR, FW: FIREWALL, TM: TRAFFIC MONITOR, WOC: WAN OPTIMIZATION CONTROLLER, IDPS: INTRUSION DETECTION PREVENTION SYSTEM, AND VOC: VIDEO OPTIMIZATION CONTROLLER).

Service	Sequence of functions	Demand	b_c
Web service	NAT-FW-TM-WOC-IDPS	18.2%	100 Kbps
VoIP	NAT-FW-TM-FW-NAT	11.8%	64 Kbps
Video streaming	NAT-FW-TM-VOC-IDPS	69.9%	4 Mbps
Online gaming	NAT-FW-VOC-WOC-IDPS	0.1%	4 Mbps

TABLE V
PROCESSING REQUIREMENTS PER CONNECTION (10 USERS) FOR THE VNFS.

Function type	$p_{c,f,k}^{\mathrm{func}}$
NAT	0.0092
FW	0.009
TM	0.133
WOC	0.054
IDPS	0.107
VOC	0.054

Finally, comparing the execution time of the greedy-based heuristic approaches with that of the CSPTP-based ILPs, we observe that $\mathrm{Greedy}_{\mathrm{SC}}$ (resp. $\mathrm{Greedy}_{\mathrm{SCFP}}$) decreases the execution time by 65.5% and 95.6% (resp. 57.4% and 91.5%) compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ (resp. $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$) in case of $K_c = 1$ and $K_c = F$, respectively.

3) Scalability to Number of Connections: In this section, we focus on the impact of the number of connections, C, on the computation complexity. We use the physical network with 200 nodes ($V = 200$) and set the capacity of each physical link (i, j) to be $B_{i,j} = 1$ [Gbps] $(i, j \in \mathcal{V}, i \neq j)$. As for the service demand, we use the more practical scenario in Table IV, which is given in [9], [20]. There are six function types ($F = 6$) and four service types, each of which consists of five functions ($K_c = 5$). For each connection $c \in \mathcal{C}$, we select one of the services according to the demand distribution. Every connection c serves ten aggregated users and the resulting processing requirements per connection for the functions are given in Table V. We also set p_c^{node} per connection to be 0.005. As for the physical node capacity, we assume that each physical node $i \in \mathcal{V}$ has the same capacity $P_i = 1$. Note that $N_f = N = 5$ $(f \in \mathcal{F})$ for the service chaining problems.

Fig. 6 depicts the impact of C on the execution time for $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{NGUYEN+}}$, $\mathrm{Greedy}_{\mathrm{SC}}$, and $\mathrm{Greedy}_{\mathrm{SCFP}}$. We first confirm that the execution time of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ is always smaller than that of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{NGUYEN+}}$. Comparing the results of $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, we observe that $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ initially has lower complexity than $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ but the relationship becomes reverted when $C \geq 19$. This is because $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ suffers scarcity of VNFs, due to the fixed value of $N_f = 5$ $(f \in \mathcal{K}_c, c \in \mathcal{C})$, while $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ adjusts N_f according to its demand. Table VI presents the mean and standard deviation of N_f $(f \in \mathcal{F})$ in case of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ when $C = 20$. We confirm

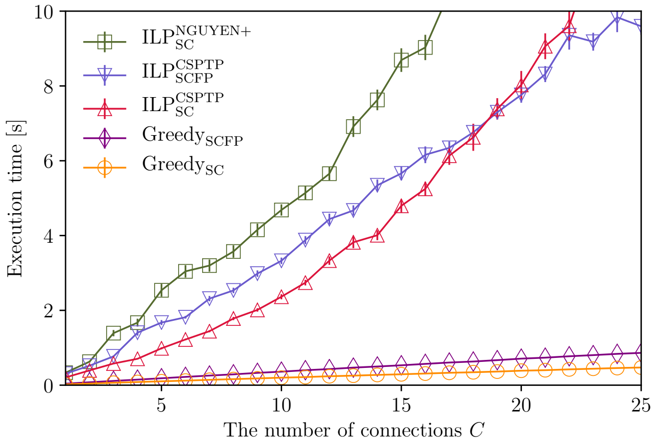


Fig. 6. Impact of the number of connections on the execution time.

TABLE VI
MEAN AND STANDARD DEVIATION OF N_f AMONG THE 200 INDEPENDENT NUMERICAL EXPERIMENTS ($C = 20$).

Function type	mean	std.
NAT	20.24	1.54
FW	19.82	1.28
TM	18.91	1.02
WOC	3.67	1.81
IDPS	16.72	1.52
VOC	13.32	2.02

that all the functions except WOC require more than five VNFs. Since $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ can appropriately determine not only N_f ($f \in \mathcal{F}$) but also the locations of functions, we will further examine the effectiveness of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ in Section VI-C1.

Finally, we focus on the performance difference between the greedy-based heuristic approaches and the proposed ILPs. We observe that the greedy-based heuristic approaches are much scalable than the ILPs. In particular, $\mathrm{Greedy}_{\mathrm{SC}}$ (resp. $\mathrm{Greedy}_{\mathrm{SCFP}}$) decreases the execution time by 89.6–96.4% (resp. 86.9–91.0%) compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ (resp. $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$).

4) Impact of Objective Function: Finally, we examine how the difference of the objective function affects the computation complexity. Figs. 7a, 7b, and 7c are the results in case of the objective function used in [8] (i.e., the minimization of node and link utilization), each of which corresponds to Figs. 4, 5, and 6, respectively. Note that the CSPTP-based ILP can also support the minimization of node and link utilization by using $d_{i,j}$ in (11). We observe that the results in case of the minimization of node and link utilization are similar to those in case of the minimization of total delay.

C. Effectiveness of Service Chaining and Function Placement

Next, we focus on the effectiveness of the service chaining and function placement in terms of the allocated number of physical nodes capable of functions, the total delay, and the utilization of physical nodes and links. In this section, we use $V = 200$, the service chain requirements in Table IV, and the processing requirements in Table V, which results in $F = 6$ and $K_c = 5$.

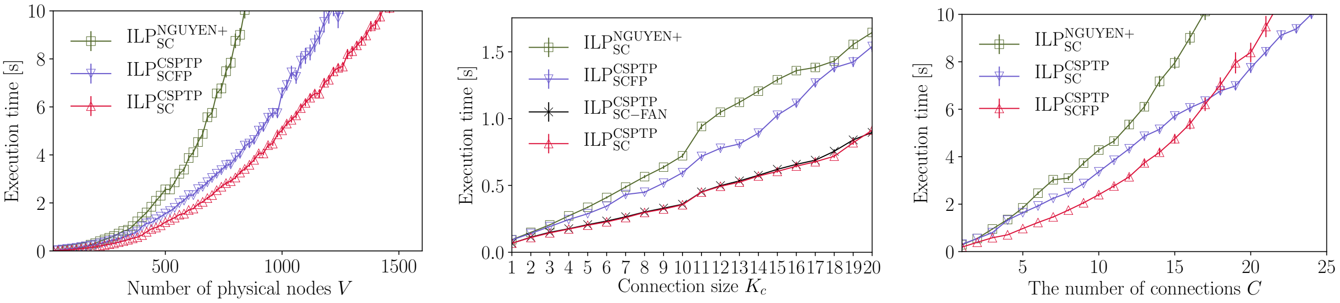
1) Allocated Number of Physical Nodes Capable of Functions: As mentioned in Section 1, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ considers N_f ($f \in \mathcal{F}$) to be constant while $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ adjusts both the number N_f and locations of physical nodes capable of functions. Fig. 8 illustrates how $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ adjusts the number N_f of physical nodes capable of each function $f \in \mathcal{F}$ when the number C of connections changes. We show the demand for each function $f \in \mathcal{F}$, which is calculated by Table IV, in the legend and the corresponding supply (i.e., the percentage range of N_f) in the right side. We confirm that $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ can appropriately adjust N_f for each function $f \in \mathcal{F}$ according to the corresponding demand.

2) Total Delay of All Service Paths: Next, we focus on the value of the objective function (1), i.e., total delay of all the service paths. Fig. 9 illustrates the impact of the number of connections, C, on the total delay of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$, $\mathrm{Greedy}_{\mathrm{SC}}$, and $\mathrm{Greedy}_{\mathrm{SCFP}}$. To clarify the impact of locations of physical nodes capable of function $f \in \mathcal{F}$, we first obtain the optimal number N_f^* and locations of physical nodes capable of function f by solving $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, and then we solve $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ under the constraint of $N_f = N_f^*$. We observe that $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ can reduce the total delay by 15.8%, compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ when $C = 20$. Since we consider homogeneous physical nodes and links in this evaluation (See Section VI-A), this result indicates that the improvement of the total delay is mainly achieved by the decrease of the hop count of the service path. In particular, $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ can reduce the average hop count of the service paths by 61.3%, compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ when $C = 20$.

We also observe that $\mathrm{Greedy}_{\mathrm{SC}}$ worsens the total delay with increase of C compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$. This is because $\mathrm{Greedy}_{\mathrm{SC}}$ sequentially calculates service paths for connections $c \in \mathcal{C}$ and their internal subpaths. In case of both the service chaining and function placement, $\mathrm{Greedy}_{\mathrm{SCFP}}$ can achieve almost the same performance as $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ because of the flexible function placement.

3) Utilization of Physical Nodes and Links: Next, we further examine the utilization of physical nodes and links. Fig. 10 (resp. Fig. 11) shows the average utilization of physical nodes (resp. links) for $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$, $\mathrm{Greedy}_{\mathrm{SC}}$, and $\mathrm{Greedy}_{\mathrm{SCFP}}$. Note that the setting of N_f for $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ is the same as that in Section VI-C2. We observe that $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ can reduce the average utilization of physical nodes (resp. links) by 7.3% (resp. 61.1%), compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ when $C = 20$. Recall that both $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$ and $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ support the same N_f for each function $f \in \mathcal{F}$ in this evaluation. In case of $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$, each function is appropriately assigned to physical nodes according to the sources and destinations of service requests, which realizes low latency, short hop, and low average node/link utilization.

Finally, we focus on the performance difference between the greedy-based heuristic approaches and the CSPTP-based ILPs. Because of the sequential calculation and limited function locations, $\mathrm{Greedy}_{\mathrm{SC}}$ increases the average utilization of physical node (resp. link) by 3.5% (resp. 30.1%) compared with $\mathrm{ILP}_{\mathrm{SC}}^{\mathrm{CSPTP}}$ when $C = 20$. On the other hand, $\mathrm{Greedy}_{\mathrm{SCFP}}$ achieves almost the same performance as $\mathrm{ILP}_{\mathrm{SCFP}}^{\mathrm{CSPTP}}$.



(a) Impact of the number of physical nodes on execution time.

(b) Impact of connection size K_c on execution time.

(c) Impact of the number of connections on execution time.

Fig. 7. Impact of objective function on computation complexity (minimization of node and link utilization).

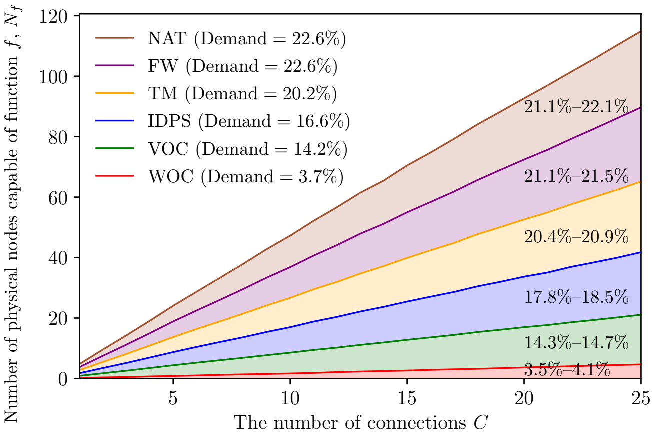


Fig. 8. Impact of the number of connections on the number of physical nodes capable of each function f.

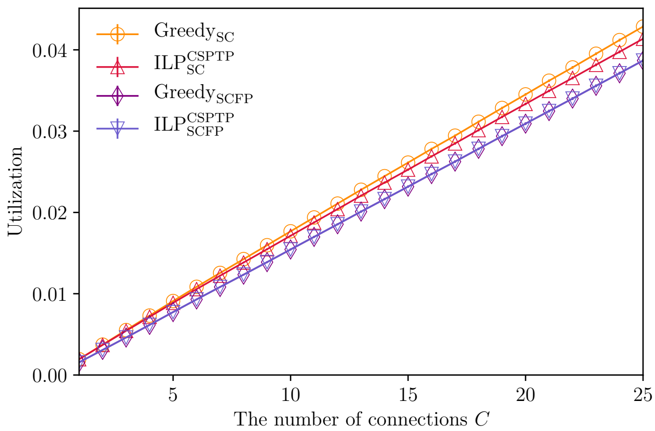


Fig. 10. The comparison of average utilization of physical nodes between ILP$^{\text{CSPTP}}_{\text{SCFP}}$ and ILP$^{\text{CSPTP}}_{\text{SC}}$.

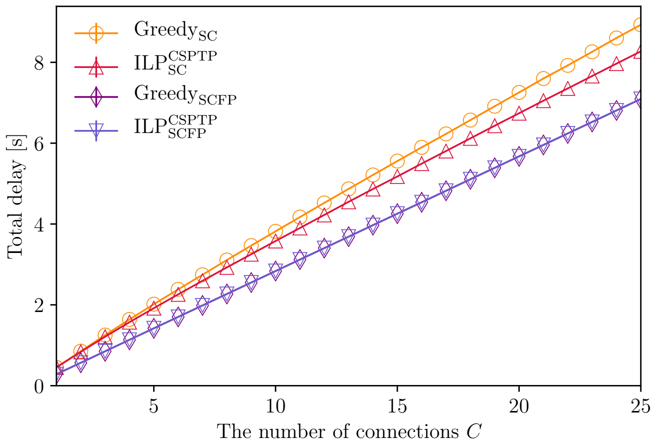


Fig. 9. Impact of the number of connections on the total delay.

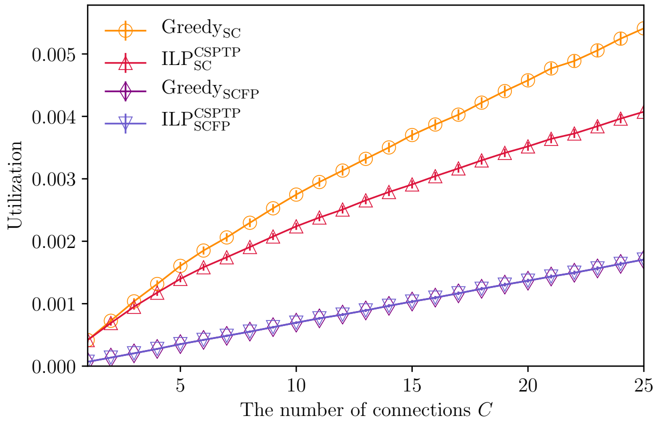
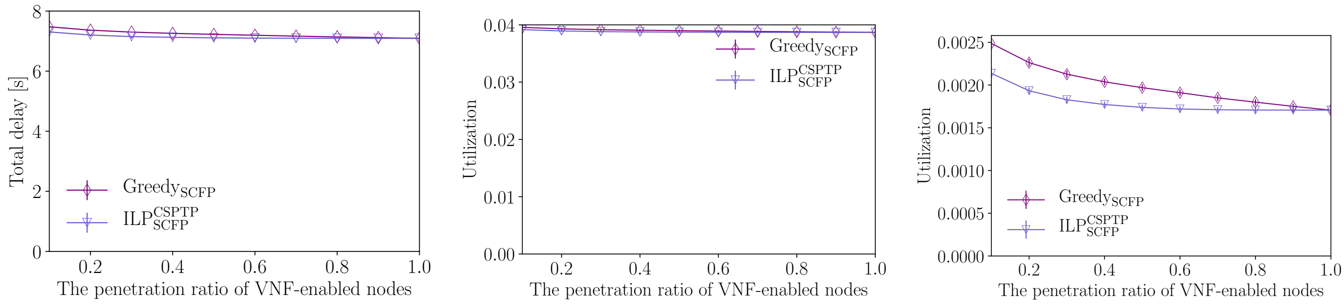


Fig. 11. The comparison of average utilization of physical links between ILP$^{\text{CSPTP}}_{\text{SCFP}}$ and ILP$^{\text{CSPTP}}_{\text{SC}}$.

4) Impact of the Number of VNF-enabled Nodes: So far we assume that all the physical nodes are VNF-enabled (i.e., $\mathcal{V}_{\text{VNF}} = \mathcal{V}$). In the transition period from the conventional network to the NFV network, VNF-enabled would be deployed gradually. In this section, we evaluate the impact of the penetration ratio ρ of the number of VNF-enabled nodes to total number of physical nodes on the system performance. In such a scenario, Greedy$_{\text{SCFP}}$ may not be able to achieve the optimal service chaining and function placement, due to the limited locations of VNF-enabled nodes. Figs. 12a, 12b, and 12c illustrate the impact of ρ on the total delay, node utilization, and link utilization, respectively. We set $C = 25$



(a) Total delay.

(b) Average utilization of physical nodes.

(c) Average utilization of physical links.

Fig. 12. Impact of penetration ratio ρ of VNF-enabled nodes on performance of $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$ and that of $\mathrm{Greedy}_{\mathrm{SCFP}}$ ($C = 25$).

and randomly choose ρV physical nodes as VNF-enabled nodes. We also focus on the performance difference between $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$ and $\mathrm{Greedy}_{\mathrm{SCFP}}$. As we expected, we confirm that the performance of $\mathrm{Greedy}_{\mathrm{SCFP}}$ becomes worse than that of $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$ with decrease of ρ. Note that we also observe that the performance degradation of $\mathrm{Greedy}_{\mathrm{SCFP}}$ compared with $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$ is not so large, i.e., 2.3% in total delay, 1.0% in node utilization, and 17.0% in link utilization.

VII. Conclusions

In this paper, we have formulated two kinds of the simple yet novel integer linear programs (ILPs) (i.e., $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SC}}$ for the service chaining and $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$ for both the service chaining and function placement) in network function virtualization (NFV) networks, with the help of the following two key ideas. One is focusing on the similarity between the service chaining problem and shortest path tour problem (SPTP) with node/link capacity constraints, i.e., capacitated SPTP (CSPTP). Another is the development of the new network model called augmented network. Through numerical results obtained by solving them using the existing solver CPLEX, we have shown that $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SC}}$ can support 1.22–1.90 times as large-scale systems as the existing ILP over the expanded network, $\mathrm{ILP}^{\mathrm{NGUYEN+}}_{\mathrm{SC}}$. We have further demonstrated that $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SCFP}}$ can reduce the total delay of all service paths by 15.8% and the average physical node (resp. link) utilization by 7.3% (resp. 61.1%) compared with $\mathrm{ILP}^{\mathrm{CSPTP}}_{\mathrm{SC}}$. For further scalability, we have proposed a shortest-path-based heuristic algorithm to solve the ILPs and have shown the heuristic for the service chaining and function placement, $\mathrm{Greedy}_{\mathrm{SCFP}}$, can obtain the optimal solution with high accuracy in in strongly polynomial time.

In future work, we plan the further extension of the formulation to deal with more complicated NFV-related constraints (e.g., context switching costs and upscaling costs caused by multi-core processing and implementations [20]).

Acknowledgment

This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI (C) under Grant 19K11942, Japan.

References

[1] M. Sasabe and T. Hara, "Shortest Path Tour Problem Based Integer Linear Programming for Service Chaining in NFV Networks," in *Proc. of 6th IEEE Conference on Network Softwarization (NetSoft)*, Jun. 2020, pp. 114–121.

[2] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network Function Virtualization: Challenges and Opportunities for Innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, Feb. 2015.

[3] J. G. Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[4] B. Yi, X. Wang, K. Li, S. k. Das, and M. Huang, "A Comprehensive Survey of Network Function Virtualization," *Computer Networks*, vol. 133, pp. 212–262, Mar. 2018.

[5] J. Halpern and C. Pignataro, "Service Function Chaining (SFC) Architecture," Tech. Rep. RFC7665, Oct. 2015.

[6] S. Demirci and S. Sagiroglu, "Optimal Placement of Virtual Network Functions in Software Defined Networks: A Survey," *Journal of Network and Computer Applications*, vol. 147, pp. 102 424: 1–20, Dec. 2019.

[7] A. Gupta, B. Jaumard, M. Tornatore, and B. Mukherjee, "A Scalable Approach for Service Chain Mapping with Multiple SC Instances in a Wide-Area Network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 529–541, Mar. 2018.

[8] T. Nguyen, A. Girard, C. Rosenberg, and S. Fdida, "Routing via Functions in Virtual Networks: The Curse of Choices," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp. 1192–1205, Jun. 2019.

[9] N. Huin, B. Jaumard, and F. Giroire, "Optimal Network Service Chain Provisioning," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1320–1333, Jun. 2018.

[10] N. Hyodo, T. Sato, R. Shinkuma, and E. Oki, "Virtual Network Function Placement for Service Chaining by Relaxing Visit Order and Non-Loop Constraints," *IEEE Access*, pp. 1–12, Aug. 2019.

[11] L. G. Valiant, "The Complexity of Enumeration and Reliability Problems," *SIAM Journal on Computing*, vol. 8, no. 3, pp. 410–421, Aug. 1979.

[12] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed., 2000.

[13] P. Festa, "Complexity Analysis and Optimization of the Shortest Path Tour Problem," *Optimization Letters*, vol. 6, no. 1, pp. 163–175, Jan. 2012.

[14] P. Festa, F. Guerriero, D. Laganà, and R. Musmanno, "Solving the Shortest Path Tour Problem," *European Journal of Operational Research*, vol. 230, no. 3, pp. 464–474, Nov. 2013.

[15] S. Bhat and G. N. Rouskas, "Service-Concatenation Routing with Applications to Network Functions Virtualization," in *Proc. of 26th International Conference on Computer Communication and Networks (ICCCN)*, Jul. 2017, pp. 1–9.

[16] L. Gao and G. N. Rouskas, "On Congestion Minimization for Service Chain Routing Problems," in *Proc. of IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–6.

[17] D. Ferone, P. Festa, F. Guerriero, and D. Laganà, "The Constrained Shortest Path Tour Problem," *Computers & Operations Research*, vol. 74, pp. 64–77, Oct. 2016.

[18] R. C. de Andrade and R. D. Saraiva, "An Integer Linear Programming Model for the Constrained Shortest Path Tour Problem," *Electronic Notes in Discrete Mathematics*, vol. 69, pp. 141–148, Aug. 2018.

[19] ILOG, "IBM ILOG CPLEX Optimizer," https://www.ibm.com/products/ilog-cplex-optimization-studio, 2020, Accessed 30 Sept. 2020.

[20] M. Savi, M. Tornatore, and G. Verticale, "Impact of Processing-Resource Sharing on the Placement of Chained Virtual Network Functions," *IEEE Transactions on Cloud Computing*, pp. 1–14, 2019.

[21] M. A. T. Nejad, S. Parsaeefard, M. A. Maddah-Ali, T. Mahmoodi, and B. H. Khalaj, "vSPACE: VNF Simultaneous Placement, Admission Control and Embedding," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 542–557, Mar. 2018.

[22] S. D'Oro, L. Galluccio, S. Palazzo, and G. Schembra, "Exploiting Congestion Games to Achieve Distributed Service Chaining in NFV Networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 407–420, Feb. 2017.

[23] S. D'Oro, L. Galluccio, S. Palazzo, and G. Schembra, "A Game Theoretic Approach for Distributed Resource Allocation and Orchestration of Softwarized Networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 3, pp. 721–735, Mar. 2017.

[24] D. Bhamare, M. Samaka, A. Erbad, R. Jain, L. Gupta, and H. A. Chan, "Optimal Virtual Network Function Placement in Multi-Cloud Service Function Chaining Architecture," *Computer Communications*, vol. 102, pp. 1–16, Apr. 2017.

[25] D. Li, P. Hong, K. Xue, and J. Pei, "Virtual Network Function Placement and Resource Optimization in NFV and Edge Computing Enabled Networks," *Computer Networks*, vol. 152, pp. 12–24, Apr. 2019.

[26] A. Tomassilli, F. Giroire, N. Huin, and S. Pérennes, "Provably Efficient Algorithms for Placement of Service Function Chains with Ordering Constraints," in *Proc. of IEEE INFOCOM 2018*, Apr. 2018, pp. 774–782.

[27] G. Sallam, G. R. Gupta, B. Li, and B. Ji, "Shortest Path and Maximum Flow Problems Under Service Function Chaining Constraints," in *Proc. of IEEE INFOCOM 2018*, Apr. 2018, pp. 2132–2140.

[28] R. Gouareb, V. Friderikos, and A.-H. Aghvami, "Virtual Network Functions Routing and Placement for Edge Cloud Latency Minimization," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2346–2357, Oct. 2018.

[29] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, "Online and Batch Algorithms for VNFs Placement and Chaining," *Computer Networks*, vol. 158, pp. 98–113, Jul. 2019.

[30] F. Liu, P. Li, and S. Gao, "Optimized Service Function Path Scaling in SDN/NFV Networks," in *Proc. of the 5th International Conference on Systems, Control and Communications*, Dec. 2019, pp. 27–32.

[31] D. Ferone, P. Festa, and F. Guerriero, "An Efficient Exact Approach for the Constrained Shortest Path Tour Problem," *Optimization Methods and Software*, pp. 1–20, Jan. 2019.

[32] R. D. Saraiva and R. C. de Andrade, "Constrained Shortest Path Tour Problem: Models, Valid Inequalities, and Lagrangian Heuristics," *International Transactions in Operational Research*, vol. 2020, pp. 1–40, Mar. 2020.

[33] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating Virtualized Network Functions," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 725–739, Dec. 2016.

[34] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 6th ed., 2018.

[35] Boost, "Boost Graph Library," https://www.boost.org/, 2020, Accessed 30 Sept. 2020.

[36] V. Batagelj and U. Brandes, "Efficient Generation of Large Random Networks," *Physical Review E*, vol. 71, no. 3, pp. 036 113:1–5, Mar. 2005.

Masahiro Sasabe (M'06) received the B.S., M.E., and Ph.D. degrees from Osaka University, Japan, in 2001, 2003, and 2006, respectively. He was an Assistant Professor with the Cybermedia Center, Osaka University from 2004 to 2007, an Assistant Professor of Graduate School of Engineering, Osaka University from 2007 to 2014. He is currently an Associate Professor of Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan. His research interests include P2P/NFV networking, game-theoretic approaches, and network optimization. Dr. Sasabe is a member of IEEE and IEICE.

Takanori Hara received M.Eng. degree from Nara Institute of Science and Technology, Japan, in 2018. He is currently working towards the Ph.D degree with Nara Institute of Science and Technology, Japan. He is also a Research Fellow of the Japan Society for the Promotion of Science (DC2). His research interests include route planning, NFV networking, and game-theoretic approaches.