Lagrangian Heuristics for Capacitated Shortest Path Tour Problem Based Online Service Chaining

Takanori Hara, Masahiro Sasabe
Graduate School of Science and Technology, Nara Institute of Science and Technology,
8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan.
{hara, m-sasabe}@ieee.org

***Abstract*—Network functions virtualization (NFV) can flexibly deploy diverse network services by liberating network functions from traditional network appliances and executing them as virtual network functions (VNFs) on generic hardware. A certain network service can be represented by a service chain, which consists of VNFs in required order. The service chaining problem is finding a suitable service path from the origin to the destination such that the VNFs are executed at the intermediate nodes in the required order under the resource constraints, which belongs to the complexity class NP-hard. In our previous work, considering the similarity between the service chaining problem and the shortest path tour problem (SPTP), we formulated the service chaining as the capacitated SPTP (CSPTP) based ILP, where CSPTP is an extended version of the SPTP with the node and link capacity constraints. In this paper, to address both computational complexity and optimality of resource allocation, we propose Lagrangian heuristics to solve the CSPTP-based ILP especially for the online service chaining. Through simulation results, we show that the proposed algorithm almost achieves the optimal resource allocation with much smaller execution time compared with the existing solver, CPLEX.**

***Index Terms*—Network functions virtualization (NFV), service chaining, capacitated shortest path tour problem (CSPTP), integer linear programming (ILP), Lagrangian relaxation, subgradient algorithm.**

I. Introduction

Network functions virtualization (NFV) enables a communication network to provide more flexible network services by replacing the traditional network appliances (e.g., firewall, network address translation (NAT), and deep packet inspection) with inexpensive generic hardware (e.g., high volume server) and executing the network functions virtually on them, which are called virtual network functions (VNFs) [1]–[4]. A certain network service can be composed of VNFs in required order, which is called a *service chain* (SC) or service function chain (SFC) [5].

To realize a service chain from its requirements (e.g., an origin node, a destination node, a sequence of functions, and demand for bandwidth and processing), we need to solve a *service chaining problem*, which is a kind of combinational optimization problems. More specifically, the service chaining problem is finding a suitable service path from the origin to the destination such that the VNFs are executed at the intermediate node in the required order under the resource constraints. The service chaining problem is one of the resource allocation problems in the NFV network [2]–[4] and belongs to the complexity class **NP**-hard [6].

Recently, several studies pointed out that the service chaining [7]–[10] has the same aspect as the *shortest path tour problem* (SPTP) [11], [12]. The SPTP is a variant of the shortest path problem (SPP) and aims at finding the shortest path from an origin to a destination while traversing at least one node from given disjoint node subsets $\mathcal{T}_1, \ldots, \mathcal{T}_K$ in this order [13]. In [13], the author proved that the SPTP belongs to the complexity class **P**. Bhat and Rouskas first pointed out the similarity between service chaining and SPTP as well as proposed an algorithm for finding the shortest path tour [7].

In addition to the similarity, the service chaining problem also has a different aspect from the SPTP, i.e., constraints on both node and link capacities. As for the constraints on link capacities, several researchers extended the SPTP to constrained SPTP such that the path does not use an identical link more than once [14]–[16]. The constrained SPTP was proved to belong to the complexity class **NP**-complete [14]. The problem was formulated as an integer linear program (ILP) and Lagrangian heuristics was proposed to solve the problem [15], [16].

In our previous work [10], we modeled the service chaining problem as a *capacitated* SPTP (CSPTP), which is a general case of the constrained SPTP, where it supports more general constraints on both node and link capacities with real values. We exactly formulated the CSPTP-based ILP for the service chaining problem, with the help of the ILP formulation for the constrained SPTP [15] and a special network model called *augmented network*. The details about the augmented network and CSPTP-based ILP formulation will be given in Section III-B and Section III-D, respectively. The high computational complexity of CSPTP-based service chaining comes from the following characteristics: (1) allowing the use of identical links as many times as needed, (2) ensuring execution of VNFs in required order, and (3) meeting the constraints on both node and link capacities.

To cope with the tradeoff between optimality and computational complexity of service chaining, we propose a Lagrangian-based heuristic framework to effectively solve the CSPTP-based ILP especially for the online service chaining. The proposed framework is a combination of several existing techniques, i.e., the CSPTP [10], Lagrangian relaxation, the shortest path tour algorithm [7], and a subgradient algorithm, by taking advantage of each technique. The details of the proposed framework will be given in Section IV. Through

simulation results, we reveal fundamental characteristics of the proposed heuristics from the viewpoint of optimality of resource allocation and computational complexity.

The rest of the manuscript is organized as follows. Section II gives the related work. In Section III, we introduce the system model and CSPTP-based ILP for the online service chaining. To overcome the computational complexity, we propose the Lagrangian-based heuristic framework for solving the CSPTP-based ILP in Section IV. Section V shows the fundamental characteristics of the proposed algorithm. Finally, Section VI gives the conclusions and future work.

II. Related Work

The service chaining is one of the resource allocation problems in NFV networks, which tries to find a suitable service path that meets both resource constraints and service chain requirements, e.g., processing/bandwidth demand and sequential execution of functions [2]–[4]. To tackle this issue, there are many studies on efficiently solving the service chaining problem to achieve various objectives [10], [17]–[21]. In addition, many studies proposed heuristic algorithms for solving the service chaining problem from the viewpoint of practicability [9], [10], [18]–[21]. Sun et al. formulated an ILP for the energy-efficient and traffic-aware service chaining in multi-domain networks and proposed a heuristic algorithm [19]. Huin et al. formulated an ILP for the service chaining problem using a layered graph to minimize the total link utilization and proposed a heuristic algorithm [20]. Hyodo et al. formulated the function placement problem for the service chaining as an ILP by using a layered graph and proposed a heuristic algorithm by relaxing the visiting order and non-loop constraints [18]. In [21], the authors formulated an ILP for the service chaining problem using an expanded network to minimize the total network utilization and proposed the Lagrangian heuristic algorithms. Sallam et al. modeled the service chaining problem as a multi-commodity maximum flow problem using graph transformation [17]. Recent surveys on the service chaining problem can be found in [2]–[4].

Several studies focused on the similarity between the service chaining problem and the SPTP [7]–[10]. The SPTP aims at finding the shortest path from an origin to a destination while visiting at least one node from given disjoint node subsets $\mathcal{T}_1, \ldots, \mathcal{T}_K$ in this order, which belongs to the complexity class **P** [13]. Bhat and Rouskas proposed the efficient algorithm for solving the SPTP [7]. However, this algorithm does not consider the load and capacity of each physical node and link. Liu et al. proposed the SPTP-based service chaining to minimize the transmission cost by using the multi-stage graph [9]. In [8], the authors proposed the SPTP-based online routing algorithm for service chaining to minimize maximum network utilization by using a potential function.

In our previous work [10], we exactly formulated the CSPTP-based ILP for service chaining using the augmented network. We also proposed a greedy-based heuristic algorithm applying sequential shortest path selection, which can drastically reduce the computational complexity at the sacrifice of the optimality of service chaining. As mentioned in Section I, the CSPTP is the generalized version of the constrained SPTP [14]–[16]. The constrained SPTP is a variant of the SPTP such that the path does not use an identical link more than once, which belongs to **NP**-complete [14]. As for the constrained SPTP, Saraiva and Andrade formulated it as an ILP and proposed a Lagrangian-based heuristic algorithm [15], [16]. Inspired by these approaches, we will formulate the Lagrangian dual problem of the CSPTP-based ILP for the online service chaining. To deal with the complexity of the CSPTP while keeping the optimality of the service chaining, we propose a Lagrangian heuristics for the service chaining by integrating the CSPTP [10], Lagrangian relaxation, the existing shortest path tour algorithm [7], and a subgradient algorithm.

The network models, e.g., layered graph, expanded network, and augmented network, also play an important role to achieve effective service chaining [10], [18], [20], [21]. Both the layered graph and expanded network construct a hierarchical network by layering the original physical networks according to the service chain requirement [18], [20], [21]. As a result, the number of layers increases with the number of required functions. In addition, they also require to customize (reconstruct) the network in response to the arrival of a new service chain request. In contrast to them, the augmented network is constructed by the extending the original physical network with imaginary nodes, each of which is responsible for the corresponding VNF and connected to physical node(s) capable of it via a virtual link. Because the augmented network can support arbitrary service chain requests by preparing the imaginary nodes for all functions $\mathcal{F}$, it can alleviate the overhead of the network reconstruction, which indicates that it is more suitable for the online service chaining. In this paper, we adopt the augmented network to solve the online service chaining.

III. System Model

We follow the system model considered in [10]. In this section, we briefly introduce it in terms of service chain request, network, service path, and CSPTP-based ILP for the online service chaining, respectively.

A. Service Chain Request

We consider the online service chaining where a new service chain request c arriving at the NFV network is immediately served by an NFV orchestrator. Each service chain request c has requirements $r_c = (o_c, d_c, \mathcal{R}_c, b_c, p_c^{\mathrm{node}}, \{p_{c,f_{c,k}}^{\mathrm{func}}\}_{k=1,\ldots,K_c})$. o_c (resp. d_c) denotes an origin (resp. a destination) node. $\mathcal{R}_c$ represents a sequence of K_c functions in the required order, i.e., $(f_{c,1}, \ldots, f_{c,K_c})$. b_c denotes the required throughput at a constant bit rate. p_c^{node} (resp. $p_{c,f_{c,k}}^{\mathrm{func}}$) is the processing capacity required for forwarding packets (resp. executing the kth function $f_{c,k}$) at a physical node. The top part of Fig. 1 shows an example of the service chain requirements for the service chain request c.

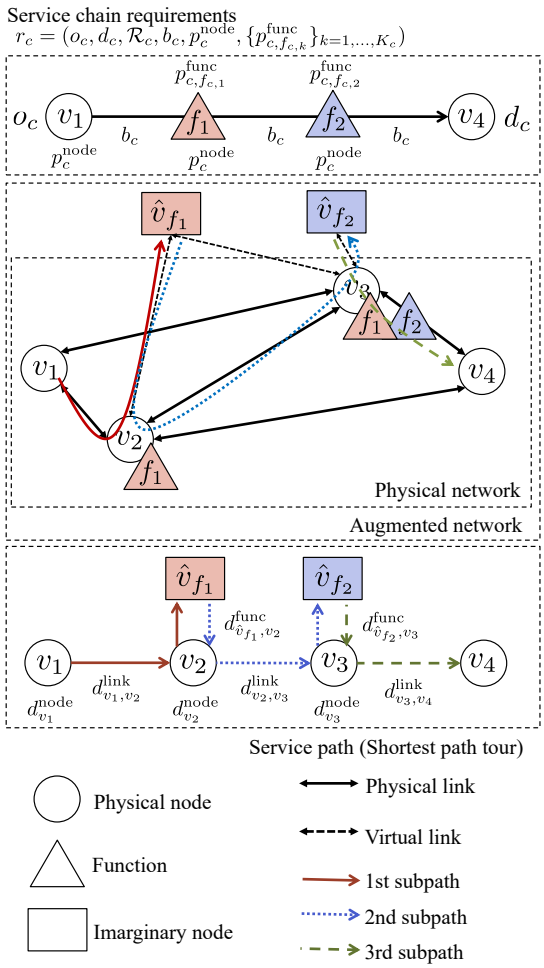

Fig. 1: Overview of service chaining.

B. Network

We consider a physical network as a directed graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ (resp. $\mathcal{E}$) denotes a set of physical nodes (resp. links). Each physical node $i \in \mathcal{V}$ (resp. each physical link $(i,j) \in \mathcal{E}$) has available processing capacity P_i (resp. available bandwidth $B_{i,j}$) at the beginning of serving the service chain request c. The NFV network supports a set of F distinct network functions, $\mathcal{F} = \{f_1, \ldots, f_F\}$. Note that the first (resp. second) function $f_{c,1}$ (resp. $f_{c,2}$) of service chain request c corresponds to f_1 (resp. f_2) in the example of Fig. 1. In general, the NFV network consists of two types of physical nodes, i.e., VNF-enabled nodes $\mathcal{V}_{\mathrm{VNF}}$ and normal one. Each VNF-enable node $i \in \mathcal{V}_{\mathrm{VNF}}$ can support one or more functions $\mathcal{F}_i \subseteq \mathcal{F}$ while the normal one corresponds to a conventional router or switch only for data forwarding. In this paper, we simply assume that all the physical nodes are VNF-enabled nodes, i.e., $\mathcal{V} = \mathcal{V}_{\mathrm{VNF}}$. We further assume that each function $f \in \mathcal{F}$ is supported by part of VNF-enabled nodes, i.e., $\mathcal{N}_f \subseteq \mathcal{V}_{\mathrm{VNF}}$, where $N_f = |\mathcal{N}_f|$.

To handle the CSPTP, the augmented network $G^+ = (\mathcal{V}^+, \mathcal{E}^+)$ is constructed by extending the physical network G with imaginary nodes $\hat{\mathcal{V}}$ and virtual links $\hat{\mathcal{E}}^{\mathrm{in}} \cup \hat{\mathcal{E}}^{\mathrm{out}}$. Note that $\mathcal{V}^+ = \mathcal{V} \cup \hat{\mathcal{V}}$ and $\mathcal{E}^+ = \mathcal{E} \cup \hat{\mathcal{E}}^{\mathrm{in}} \cup \hat{\mathcal{E}}^{\mathrm{out}}$. An imaginary node $\hat{v}_{f_{c,k}} \in \hat{\mathcal{V}}$ is responsible for the execution of kth function $f_{c,k}$ and is connected to physical node(s) capable of $f_{c,k}$. These connected links are called virtual links. $\hat{\mathcal{E}}^{\mathrm{in}}$ (resp. $\hat{\mathcal{E}}^{\mathrm{out}}$) denotes a set of links incoming to (resp. outgoing from) an imaginary node $\hat{v}_f$. Note that $\hat{\mathcal{E}}^{\mathrm{in}} = \{(v, \hat{v}_f) \mid v \in \mathcal{N}_f, \hat{v}_f \in \hat{\mathcal{V}}, f \in \mathcal{F}_v\}$ (resp. $\hat{\mathcal{E}}^{\mathrm{out}} = \{(\hat{v}_f, v) \mid \hat{v}_f \in \hat{\mathcal{V}}, v \in \mathcal{N}_f, f \in \mathcal{F}_v\}$). The virtual link $(\hat{v}_{f_{c,k}}, v) \in \hat{\mathcal{E}}^{\mathrm{out}}$ indicates that the physical node $v \in \mathcal{N}_{f_{c,k}}$ supports the function $f_{c,k}$. Selecting the virtual link $(\hat{v}_{f_{c,k}}, v)$ as a part of the service path means that the function $f_{c,k}$ will be executed at the physical node v. We also define the set of neighbors of node i as $\mathcal{V}_i^+$.

In contrast to the existing network models, i.e., the layered graph and expanded network, we only need to construct one augmented network, which can support arbitrary service chain requests by preparing the imaginary nodes for all the functions $\mathcal{F}$, and thus we can alleviate the overhead of network construction. We present an example of the augmented network in the middle part of Fig. 1.

C. Service Path

The service path $\mathcal{S}_c$ with the origin o_c, destination d_c, and $R_c = (f_{c,1}, \ldots, f_{c,K_c})$ can be decomposed into a sequence of $K_c + 1$ subpaths, i.e., $(\mathcal{S}_{c,1}, \ldots, \mathcal{S}_{c,K_c+1})$. Here, we define $\mathcal{K}_c = \{1, \ldots, K_c\}$ and $\mathcal{K}_c^+ = \{1, \ldots, K_c + 1\}$. The origin node $\alpha_{c,k}$ and destination node $\beta_{c,k}$ of the kth subpath are given as follows:

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

Each subpath does not contain any loop while the entire service path may have loop(s). We cannot determine how many times one link will be used in the service path before calculating the service path itself, which is one of the reasons making the service chaining problem **NP**-complete. An example of the service path is presented at the bottom part of Fig. 1.

The optimality of a service path will be evaluated by total delay, which is expressed by the sum of propagation delay and processing delay, where each physical link $(i,j) \in \mathcal{E}$ has the propagation delay $d_{i,j}^{\mathrm{link}}$ and each physical node $v \in \mathcal{V}$ has the processing delay d_v^{node} for data forwarding. The physical node $v \in \mathcal{V}$ capable of function $f \in \mathcal{R}_c$ has the processing delay $d_{\hat{v}_f, v}^{\mathrm{func}}$ for executing the function f.

D. CSPTP-Based ILP for Online Service Chaining

With the help of the augmented network, the online service chaining problem can be formulated as the following ILP $\mathrm{Z}(\boldsymbol{x})$, where $\boldsymbol{x} = [x_{i,j}^{c,k}]$ $(k \in \mathcal{K}_c^+), (i,j) \in \mathcal{E}^+$ denotes the binary decision variables [10]:

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

$$\min_{\boldsymbol{x}} \quad \sum_{(i,j)\in\mathcal{E}^+} d_{i,j} \sum_{k\in\mathcal{K}_c^+} x_{i,j}^{c,k}, \tag{1}$$

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

$$\sum_{j\in\mathcal{V}_i^+} x_{i,j}^{c,k} - \sum_{j\in\mathcal{V}_i^+} x_{j,i}^{c,k} = \begin{cases} 1 & \text{if } i=\alpha_{c,k}, \\ -1 & \text{if } i=\beta_{c,k}, \\ 0 & \text{otherwise}, \end{cases} \qquad i\in\mathcal{V}^+, k\in\mathcal{K}_c^+, \tag{3}$$

$$x_{i,\hat{v}_{f_{c,k}}}^{c,k} = x_{\hat{v}_{f_{c,k}},i}^{c,k+1}, \qquad (i,\hat{v}_{f_{c,k}})\in\hat{\mathcal{E}}^{\text{in}}, (\hat{v}_{f_{c,k}},i)\in\hat{\mathcal{E}}^{\text{out}}, k\in\mathcal{K}_c, \tag{4}$$

$$x_{i,\hat{v}_{f_{c,m}}}^{c,k} = 0, \qquad (i,\hat{v}_{f_{c,m}})\in\hat{\mathcal{E}}^{\text{in}}, k\in\mathcal{K}_c^+, m\neq k, \tag{5}$$

$$b_c \sum_{k\in\mathcal{K}_c^+} x_{i,j}^{c,k} \le B_{i,j}, \qquad (i,j)\in\mathcal{E}, \tag{6}$$

$$p_c^{\text{node}} \sum_{(v,j)\in\mathcal{E}} \sum_{k\in\mathcal{K}_c^+} x_{v,j}^{c,k} + \sum_{(\hat{v}_f,v)\in\hat{\mathcal{E}}^{\text{out}}} p_{c,f}^{\text{func}} \sum_{k\in\mathcal{K}_c^+} x_{\hat{v}_f,v}^{c,k} \le P_v, \qquad v\in\mathcal{V}. \tag{7}$$

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

$$d_{i,j} = \begin{cases} d_i^{\text{node}} + d_{i,j}^{\text{link}}, & \text{if } (i,j)\in\mathcal{E}, \\ d_{i,j}^{\text{func}}, & \text{if } (i,j)\in\hat{\mathcal{E}}^{\text{out}}, \\ 0, & \text{otherwise.} \end{cases}$$

The CSPTP-related constraints are given by (2)–(7). Constraints (2)–(5) are revelant to realization of the SPTP. Constraint (2) defines the domain of the binary decision variables. Constraint (3) is the standard flow conservation equations. Constraint (4) guarantees the connectivity between kth and $(k+1)$th subpaths of service chain request c. Constraint (5) prohibits the service path from traversing the imaginary node $\hat{v}_{f_{c,m}}$ in kth subpath ($m \neq k$). Constraint (6) (resp. (7)) gives the constraint on physical link capacity (resp. processing capacity of the physical node).

IV. Lagrangian Heuristics for CSPTP-based Online Service Chaining

To address the computational complexity, we propose Lagrangian heuristics to solve the CSPTP-based ILP for the online service chaining.

A. Overview

Algorithm 1 presents the proposed algorithm to solve the CSPTP-based ILP for the online service chaining. The proposed algorithm is based on the Lagrangian heuristics framework, which consists of three techniques, i.e., Lagrangian relaxation, shortest path tour algorithm [7], and subgradient algorithm. We first formulate the Lagrangian dual problem for the CSPTP-based ILP $\mathrm{Z}(\boldsymbol{x})$ using the Lagrangian relaxation, which transforms the CSTP into SPTP. (The details will be shown in Section IV-B.) Next, to solve the SPTP efficiently, we apply the existing algorithm for finding the shortest path tour [7], which will be described in Section IV-C. To handle the original capacity constraints, we also adopt the subgradient algorithm, which will be explained in Section IV-D.

Algorithm 1 Proposed algorithm.

Require: Optimality tolerance δ, maximum number $T_{\max}$ of subgradient iteration, and weighting parameter ω.
Ensure: Optimal solution $\boldsymbol{x}^*$.

$\boldsymbol{\mu}^{(\tau)} \leftarrow \mathbf{0},\ \boldsymbol{\gamma}^{(\tau)} \leftarrow \mathbf{0},\ \tau \leftarrow 0$
do
  $\boldsymbol{x}^* \leftarrow$ FIND_SHORTEST_PATH_TOUR$(\Phi(\boldsymbol{\mu}^{(\tau)}, \boldsymbol{\gamma}^{(\tau)}))$
  if $\boldsymbol{x}^*$ is infeasible **then**
    return None
  $\boldsymbol{h}^{(\tau)}, \boldsymbol{g}^{(\tau)} \leftarrow$ SUBGRADIENT$(\Phi(\boldsymbol{\mu}^{(\tau)}, \boldsymbol{\gamma}^{(\tau)}))$
  $\theta_{\boldsymbol{\mu}}^{(\tau)} \leftarrow \omega/(\sqrt{\tau}||\boldsymbol{h}^{(\boldsymbol{\tau})}||),\ \theta_{\boldsymbol{\gamma}}^{(\tau)} \leftarrow \omega/(\sqrt{\tau}||\boldsymbol{g}^{(\boldsymbol{\tau})}||)$
  $\boldsymbol{\mu}^{(\tau+1)} \leftarrow \max\{0, \boldsymbol{\mu}^{(\tau)} + \theta_{\boldsymbol{\mu}}^{(\tau)} \boldsymbol{h}^{(\tau)}\}$
  $\boldsymbol{\gamma}^{(\tau+1)} \leftarrow \max\{0, \boldsymbol{\gamma}^{(\tau)} + \theta_{\boldsymbol{\gamma}}^{(\tau)} \boldsymbol{g}^{(\tau)}\}$
  $\tau \leftarrow \tau + 1$
while STOP_CONDITION()
return $\boldsymbol{x}^*$

B. Lagrangian Relaxation

As mentioned in Section I, the constrained SPTP, which only considers the constraints on link capacities, belongs to **NP**-complete [14]. Since the CSPTP considers the constraints on both node and link capacities, i.e., (6) and (7), it is also expected to belong to **NP**-complete. Considering the fact that the SPTP belongs to the complexity class **P** [13], we transform the CSPTP-based ILP $\mathrm{Z}(\boldsymbol{x})$ into the SPTP-based ILP, i.e., Lagrangian problem, thanks to Lagrangian relaxation. We construct the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\gamma})$ by relaxing (6) and (7) with multipliers $\boldsymbol{\gamma} = (\gamma_{i,j})_{(i,j)\in\mathcal{E}}, \boldsymbol{\mu} = (\mu_v)_{v\in\mathcal{V}}$ ($\gamma_{i,j} \ge 0, \mu_v \ge 0$):

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\gamma}) = & \sum_{(i,j)\in\mathcal{E}} \sum_{k\in\mathcal{K}_c^+} (d_{i,j} + \mu_i p_c^{\text{node}} + \gamma_{i,j} b_c) x_{i,j}^{c,k} \\ & + \sum_{(\hat{v}_f,v)\in\hat{\mathcal{E}}^{\text{out}}} \sum_{k\in\mathcal{K}_c^+} (d_{\hat{v}_f,v}^{\text{func}} + \mu_v p_{c,f}^{\text{func}}) x_{\hat{v}_f,v}^{c,k} \\ & - \Big(\sum_{(i,j)\in\mathcal{E}} \gamma_{i,j} B_{i,j} + \sum_{i\in\mathcal{V}} \mu_i P_i \Big). \end{aligned}$$

The Lagrangian problem $\Phi(\boldsymbol{\mu}, \boldsymbol{\gamma})$ is defined as

$$\begin{aligned} \Phi(\boldsymbol{\mu}, \boldsymbol{\gamma}) = \min_{\boldsymbol{x}} \quad & \mathcal{L}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\gamma}) \\ \text{s.t.} \quad & (2)\text{–}(5), \end{aligned}$$

where all the constraints (2)–(5) are the SPTP-related ones. For any $\boldsymbol{\mu} \ge \mathbf{0}$ and $\boldsymbol{\gamma} \ge \mathbf{0}$, $\Phi(\boldsymbol{\mu}, \boldsymbol{\gamma})$ gives the lower bound of the optimal objective value of the original CSPTP-based ILP, i.e., $\Phi(\boldsymbol{\mu}, \boldsymbol{\gamma}) \le \mathrm{Z}(\boldsymbol{x})$. Therefore, finding $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ that maximize $\Phi(\boldsymbol{\mu}, \boldsymbol{\gamma})$ approaches the minimization of $\mathrm{Z}(\boldsymbol{x})$, which can be achieved by solving the following Lagrangian dual problem:

$$\max_{\boldsymbol{\gamma}\ge\mathbf{0}, \boldsymbol{\mu}\ge\mathbf{0}} \quad \Phi(\boldsymbol{\mu}, \boldsymbol{\gamma}). \tag{8}$$

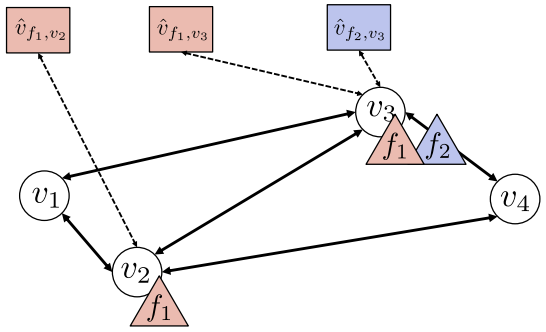

Fig. 2: An example of transformation of the augmented network in Fig. 1.

C. Finding Shortest Path Tour

To find the optimal $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ such that $\Phi(\boldsymbol{\mu},\boldsymbol{\gamma})$ is maximized, we need to solve the Lagrangian problem $\Phi(\boldsymbol{\mu},\boldsymbol{\gamma})$ repeatedly. For certain $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$, $\Phi(\boldsymbol{\mu},\boldsymbol{\gamma})$ belongs to an SPTP because the objective function $\mathcal{L}(\boldsymbol{x},\boldsymbol{\mu},\boldsymbol{\gamma})$ consists of the total path cost (delay) minus a constant $\left(\sum_{(i,j)\in\mathcal{E}}\gamma_{i,j}B_{i,j}+\sum_{i\in\mathcal{V}}\mu_i P_i\right)$ where the link delay $d_{i,j}$ is redefined as the following cost:

$$d_{i,j}=\begin{cases} d_i^{\text{node}}+d_{i,j}^{\text{link}}+\mu_i p_c^{\text{node}}+\gamma_{i,j}b_c, & \text{if } (i,j)\in\mathcal{E},\\ d_{i,j}^{\text{func}}+\mu_i p_{c,f}^{\text{func}}, & \text{if } (i,j)\in\hat{\mathcal{E}}^{\text{out}},\\ 0, & \text{otherwise.}\end{cases}$$

To efficiently solve the SPTP, we apply the shortest path tour algorithm [7], which first calculates the minimum cost for each subpath from $k=1$ to $k=K_c+1$ and then conducts the Dijkstra algorithm to obtain the shortest path.

To apply this algorithm, we need to slightly modify the augmented network to guarantee the connectivity between kth and $k+1$th subpaths ($k=1,\ldots,K_c$). More specifically, the set of imaginary nodes, $\hat{\mathcal{V}}$, is updated as $\{\hat{v}_{f,i}\}_{f\in\mathcal{F},i\in\mathcal{N}_f}$ where each imaginary node is prepared for the pair of function and physical node capable of the function, i.e., $\{\hat{v}_{f,i}\}_{f\in\mathcal{F},i\in\mathcal{N}_f}$. The corresponding set of incoming virtual links, $\hat{\mathcal{E}}^{\text{in}}$, and that of outgoing virtual links, $\hat{\mathcal{E}}^{\text{out}}$, are also updated as $\{(i,\hat{v}_{f,i})\}_{f\in\mathcal{F},i\in\mathcal{N}_f}$ and $\{(\hat{v}_{f,i},i)\}_{f\in\mathcal{F},i\in\mathcal{N}_f}$, respectively. The example of transformation of the augmented network in Fig.1 is shown in Fig. 2.

D. Subgradient Algorithm

The last task is finding appropriate $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ to maximize $\Phi(\boldsymbol{\mu},\boldsymbol{\gamma})$, i.e., solving the Lagrangian dual problem. We adopt the subgradient algorithm [22] to solve it. The subgradient algorithm first initializes parameters (i.e., τ, $\boldsymbol{\mu}$, and $\boldsymbol{\gamma}$) (line 1). Then, it solves the Lagrangian problem $\Phi(\boldsymbol{\mu},\boldsymbol{\gamma})$ using FIND_SHORTEST_PATH_TOUR($\Phi(\boldsymbol{\mu},\boldsymbol{\gamma})$) function [7] (line 3). If it fails to solve the problem, the algorithm stops and no solution is found (lines 4 and 5). Otherwise, it calculates the *subgradient* $\boldsymbol{h}^{(\tau)}$ (resp. $\boldsymbol{g}^{(\tau)}$) of the corresponding variable $\boldsymbol{\mu}$ (resp. $\boldsymbol{\gamma}$) through SUBGRADIENT($\Phi(\boldsymbol{\mu}^{(\tau)},\boldsymbol{\gamma}^{(\tau)})$) function (line 6). Here, the update rules of the subgradient $\boldsymbol{h}^{(\tau)}$ and $\boldsymbol{g}^{(\tau)}$ are given as follows:

$$\begin{aligned} h_i^{(\tau)} &= \Big(p_c^{\text{node}}\sum_{k\in\mathcal{K}_c^+}\sum_{(i,j)\in\mathcal{E}}x_{i,j}^{c,k}\\ &\qquad +p_{c,f}^{\text{func}}\sum_{k\in\mathcal{K}_c}\sum_{(j,i)\in\hat{\mathcal{E}}^{\text{out}}}x_{j,i}^{c,k}\Big)-P_i.\\ g_{i,j}^{(\tau)} &= b_c\sum_{k\in\mathcal{K}_c^+}x_{i,j}^{c,k}-B_{i,j}, \end{aligned}$$

where, $h_i^{(\tau)}$ (resp. $g_{i,j}^{(\tau)}$) denotes the ith (resp. (i,j)th) element of $\boldsymbol{h}^{(\tau)}$ (resp. $\boldsymbol{g}^{(\tau)}$). Based on the subgradient and step size (i.e., $\theta_{\boldsymbol{\mu}}^{(\tau)}$ and $\theta_{\boldsymbol{\gamma}}^{(\tau)}$), it updates $\boldsymbol{\mu}^{(\tau+1)}$ and $\boldsymbol{\gamma}^{(\tau+1)}$ (lines 8–9). The update rules of $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ are given as follows:

$$\boldsymbol{\mu}^{(\tau+1)}=\max\{0,\boldsymbol{\mu}^{(\tau)}+\theta_{\boldsymbol{\mu}}^{(\tau)}\boldsymbol{h}^{(\tau)}\},\tag{9}$$

$$\boldsymbol{\gamma}^{(\tau+1)}=\max\{0,\boldsymbol{\gamma}^{(\tau)}+\theta_{\boldsymbol{\gamma}}^{(\tau)}\boldsymbol{g}^{(\tau)}\},\tag{10}$$

$$\theta_{\boldsymbol{\mu}}^{(\tau)}=\omega/(\sqrt{\tau}||\boldsymbol{h}^{(\tau)}||),$$

$$\theta_{\boldsymbol{\gamma}}^{(\tau)}=\omega/(\sqrt{\tau}||\boldsymbol{g}^{(\tau)}||),$$

where ω is a weighting parameter. If $h_i^{(\tau)}$ (resp. $g_{i,j}^{(\tau)}$) is less than or equal to zero, (7) (resp. (6)) holds, and thus $\mu_i^{(\tau)}$ (resp. $\gamma_{i,j}^{(\tau)}$) will be reduced as in (9) (resp. (10)). Otherwise, $\mu_i^{(\tau+1)}$ (resp. $\gamma_{i,j}^{(\tau+1)}$) will be increased to strengthen the penalty for violating (7) (resp. (6)). This procedure is repeated by gradually decreasing the step size $\theta_{\boldsymbol{\mu}}^{(\tau)}$ and $\theta_{\boldsymbol{\gamma}}^{(\tau)}$ (line 7) until satisfying the stop conditions given by STOP_CONDITION() function (lines 2–11).

The algorithm will stop when one of the following three conditions is satisfied. The first condition is that the solution of $\Phi(\boldsymbol{\mu}^{(\tau)},\boldsymbol{\gamma}^{(\tau)})$, $\boldsymbol{x}^*$, satisfies the original CSPTP-related constraints at the first iteration. The second condition is that $\boldsymbol{x}^*$ satisfies the CSPTP-related constraints and the relative improvement ratio of the objective value between τth and $(\tau+1)$th iteration, i.e., $(\mathcal{L}^{(\tau+1)}-\mathcal{L}^{(\tau)})/\mathcal{L}^{(\tau)}$, is less than or equal to the optimality tolerance δ. The last condition is that the number τ of iterations reaches a predefined threshold $T_{\max}$.

E. Computational Complexity

Finally, we discuss the computational complexity of the proposed algorithm. The proposed algorithm will iterate the while loop (lines 2–11) at most $T_{\max}$. In each loop, the shortest path tour algorithm in line 3 becomes bottleneck. The computational complexity of the shortest path tour algorithm is given by $\mathcal{O}((K_c+1)V^+)+\mathcal{O}((K_c+1)E^+\log V^+)$, where the first term is related to the calculation of the minimum cost for each subpath from $k=1$ to $k=K_c+1$ and the second one is required to conduct the Dijkstra algorithm to obtain the shortest path [7]. As a result, the computational complexity of the proposed algorithm becomes $\mathcal{O}(T_{\max}(K_c+1)V^+)+\mathcal{O}(T_{\max}(K_c+1)E^+\log V^+)$.

TABLE I: Service chain demand and requirements [20], [23] (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 II: Processing requirements for VNFs per service chain request from 5 aggregated users [23].

Function type	$p^{\mathrm{func}}_{c,f,k}$
FW	0.0045
IDPS	0.0535
NAT	0.0046
TM	0.0665
VOC	0.027
WOC	0.027

V. Numerical Results

In this section, we evaluate the practicality of the proposed algorithm from the viewpoint of the computational complexity and performance of service chaining. In the calculation, we used the server with Intel Core i9-9900K 8 core and 64 GB memory.

A. Evaluation Scenario

We use the physical network composed of 200 physical nodes and physical links between two arbitrary physical nodes, each of which is randomly generated according to the probability $\pi = 0.032$ as in [24]. We assume that the capacity of each physical node i (resp. physical link between physical nodes i and j ($i, j \in \mathcal{V}, i \neq j$) is identical, i.e., $P_i = 1$ (resp. $B_{i,j} = 1$ [Gbps]). The propagation delay of each physical link between two physical nodes i and j is also identical, i.e., $d^{\mathrm{link}}_{i,j} = 10$ [ms]. The traversal and processing delay of each physical node v are set to be $d^{\mathrm{node}}_{v} = 1$ [ms] and $d^{\mathrm{func}}_{\hat{v}_f,v} = 50$ [ms] ($(\hat{v}_f, v) \in \hat{\mathcal{E}}^{\mathrm{out}}$), respectively.

We assume the service chain demand and requirements in Table I. As a result, the NFV network supports six function types ($F = 6$) and four service types, each of which is composed of five functions ($K_c = 5$). Each function $f \in \mathcal{F}$ is assigned to different $N_f = 5$ physical nodes, which are randomly chosen from $\mathcal{V}$. We select each service chain request c according to the demand distribution in Table I. Each service chain request c serves five aggregated users and has the processing requirements per service chain request, where each function required by the service chain request c has the processing requirement $p^{\mathrm{func}}_{c,f,k}$ and the data forwarding requires $p^{\mathrm{node}}_{c} = 0.0025$. The origin o_c and destination d_c of service chain request c are randomly selected from $\mathcal{V}$ such that $o_c \neq d_c$.

We implemented a simulator for the online service chaining in C++ programming language. In the simulation, we assume the following queuing model. A request for a new service chain request c arrives at the system following a Poisson process of parameter λ ($\lambda > 0$). The new request will be added to the end of the queue with infinite buffer. The NFV orchestrator tries to find a suitable service path for each service chain request in a first come first served (FCFS) manner. Note that the service time, which is required for service chaining, may vary depending on the service chain requirements and remaining network capacity. If the NFV orchestrator succeeds in finding the service path for service chain request c, the corresponding service path will be established with the service path delay and occupy the allocated physical nodes and links during the connection holding time, which is set be 10 [s]. The simulation time is set to be 100 [s].

For comparison purpose, we evaluate the CSPTP-based ILP for the online service chaining and a simple greedy-based heuristic algorithm for the service chaining [10]. To solve the CSPTP-based ILP, we used the existing solver CPLEX 12.8 [25] with the parallel optimization parameter (i.e., the number of threads) of 32. The greedy-based heuristic algorithm (*Greedy SC*) divides the service path into $K_c + 1$ subpaths and sequentially tries to find a shortest path for each subpath from the beginning ($k = 1$) to the end ($k = K_c + 1$). The computational complexity of the greedy-based heuristic algorithm is given by $\mathcal{O}((K_c+1)E^+ \log V^+)$ [10]. As for the proposed algorithm, through the preliminary experiments, we set the parameters to be $\delta = 0.05$, $\omega = 100$, and $T_{\max} = 3$.

As for the effectiveness of service chaining, in addition to the objective function, i.e., *average total delay of service paths* among all accepted requests, we evaluate the *acceptance ratio* that is the ratio of the number of service chain requests successfully served by the NFV orchestrator to the total number of service chain requests arriving at the NFV network. Note that the remaining service chain requests in the queue at the simulation end are rejected. In terms of the computational complexity, we adopt the *execution time*, which is the average of times required to calculate service paths for all the service chain requests served by the NFV orchestrator. In context of queueing theory, the execution time can be interpreted as the service time. To measure how busy the system is, we use the *traffic intensity* $\rho = \lambda/\mu$ where μ is the average service rate. In what follows, the simulation results are the average of 50 independent simulation experiments.

B. Optimality of Service Chaining

In this section, we first focus on the performance limit of resource allocation under an ideal case where the execution time is ignored. This assumes that a new incoming service chain request is served immediately and the queue is always empty. Fig. 3 depicts the transition of total delay when changing the arrival rate λ. We observe that the total delay increases with the arrival rate λ, regardless of the schemes. This is because the increase of λ leads to more resource consumption, and thus the new service chain request may be forced to establish a longer service path, due to the capacity constraint. However, we confirm that the proposed algorithm

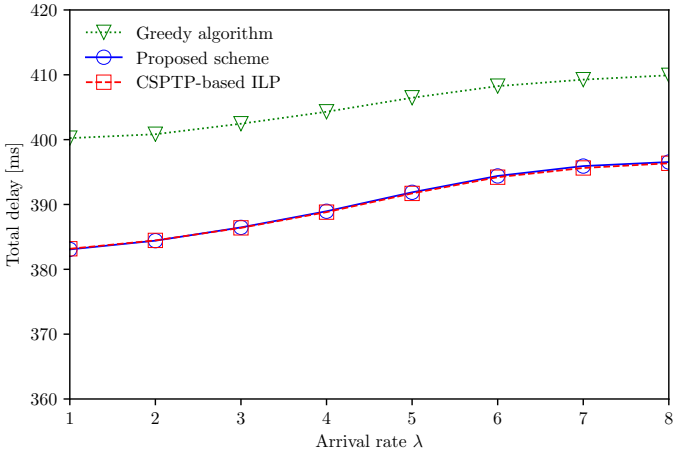

Fig. 3: Impact of arrival rate λ on total delay (ideal case).

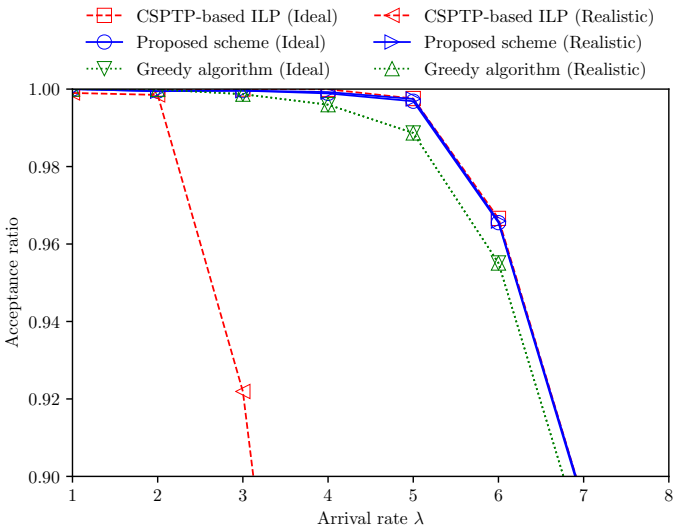

Fig. 4: Impact of arrival rate λ on acceptance ratio.

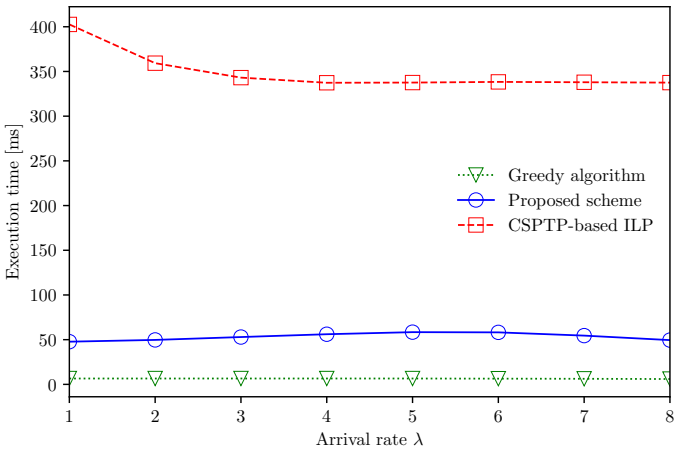

Fig. 5: Impact of arrival rate λ on execution time (realistic case).

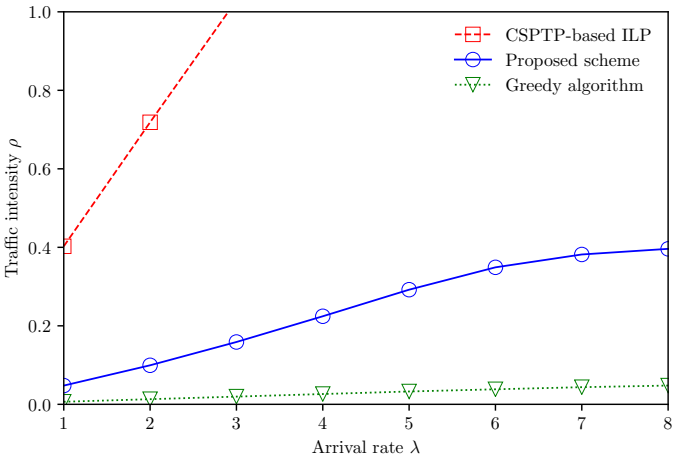

Fig. 6: Impact of arrival rate λ on traffic intensity ρ (realistic case).

exhibits almost the same performance as the CSPTP-based ILP and improves the total delay by 13.3–17.2 [ms] compared with the Greedy SC.

Fig. 4 illustrates the impact of the arrival rate λ on the acceptance ratio in the ideal case. Simply speaking, resource allocation with low network utilization will contribute to high acceptance ratio. In our problem, the objective function is the minimization of total delay of service paths, which somewhat contributes to reducing the network utilization by shortening each service path in terms of its hop count. Focusing on the ideal case, we observe that the three schemes show similar tendency: 1) The acceptance ratio first keeps almost one in the range of $\lambda = [1, \hat{\lambda}]$ and 2) it gradually decreases with increase of λ. The value of $\hat{\lambda}$ is, however, different among the schemes. The Greedy SC has $\hat{\lambda} = 4$ and shows the worst performance among them. On the other hand, the CSPTP-based ILP achieves the optimal result. We confirm that the proposed algorithm is competitive with the CSPTP-based ILP. In particular, the performance degradation of the proposed algorithm is suppressed by 0.11% compared with that of the CSPTP-baed ILP.

C. Tradeoff between Optimality of Service Chaining and Solution Complexity

In actual systems, the execution time is not negligible and prone to vary depending on the schemes. Fig. 5 illustrates the impact of arrival rate λ on the execution time. We first observe that the proposed algorithm and Greedy SC show much smaller execution time than the CSPTP-based ILP, thanks to the low computational complexity. Note that the decrease of execution time in the CSPTP-based ILP is caused by the deterioration of the acceptance ratio in Fig. 4.

Next, we focus on how the execution time affects the performance of resource allocation. Increase of the execution time also increases the number of waiting service chain requests in the queue, and thus the acceptance ratio will drop. Fig. 4 illustrates how λ affects the acceptance ratio in the realistic case where the actual execution time is considered. Note that the acceptance ratio can be degraded by the following two

factors, i.e., resource depletion or long execution time. We first observe that the acceptance ratio of the CSPTP-based ILP drastically degrades at $\lambda = 3$ in the realistic case, due to the complexity factor, which can also be confirmed from the relationship between the arrival rate λ and traffic intensity ρ, as shown in Fig. 6. This result is caused by the heavy load of the system where the traffic intensity ρ becomes higher than one. On the other hand, the Greedy SC cannot achieve the optimal acceptance ratio, due to the resource depletion caused by inefficient resource allocation. In contrast to these schemes, the proposed algorithm can achieve almost the optimal acceptance ratio.

VI. Conclusion

In this paper, we have proposed the Lagrangian heuristics framework to solve the CSPTP-based ILP for the online service chaining in the NFV network, so as to address both computational complexity and optimality of service path. In particular, we have integrated multiple existing techniques (i.e., Lagrangian relaxation, shortest path tour algorithm, and subgradient algorithm) into one solution. We have first formulated the Lagrangian dual problem for the CSPTP-based ILP with the help of the Lagrangian relaxation, which reduces the computational complexity by transforming the CSPTP into the SPTP. Next, we have solved the Lagrangian dual problem by applying the existing SPTP algorithm and the subgradient algorithm.

Through the simulation results, we have confirmed that with the increase of arrival rate the CSPTP-based ILP with the existing solver CPLEX degrades the performance due to its complexity while the Greedy SC cannot perform the high acceptance ratio because of the resource depletion by the inefficient resource allocation. On the contrary, we have shown that the proposed algorithm can achieve almost the optimal resource allocation with much smaller execution time compared with the CSPTP-based ILP. In future work, we plan to extend the proposed algorithm for both service chaining and function placement.

Acknowledgment

This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI C under Grant 19K11942 Japan and the JSPS Research Activity Start-up under Grant 21K21288 Japan.

References

[1] 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.

[2] D. Bhamare, R. Jain, M. Samaka, and A. Erbad, "A Survey on Service Function Chaining," *Journal of Network and Computer Applications*, vol. 75, pp. 138–155, Nov. 2016.

[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] B. Awerbuch, Y. Azar, and A. Epstein, "The Price of Routing Unsplittable Flow," in *Proc. of STOC*, vol. 42, 2005, pp. 160–177.

[7] S. Bhat and G. N. Rouskas, "Service-Concatenation Routing with Applications to Network Functions Virtualization," in *Proc. of ICCCN*, Jul. 2017, pp. 1–9.

[8] L. Gao and G. N. Rouskas, "Congestion Minimization for Service Chain Routing Problems With Path Length Considerations," *IEEE/ACM Transactions on Networking*, pp. 1–14, 2020.

[9] F. Liu, X. Chen, W. An, Y. Peng, J. Cao, and Y. Zhang, "Minimizing Transmission Cost for Multiple Service Function Chains in SDN/NFV Networks," in *Proc. of IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct. 2017, pp. 1–6.

[10] M. Sasabe and T. Hara, "Capacitated Shortest Path Tour Problem Based Integer Linear Programming for Service Chaining and Function Placement in NFV Networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 104–117, Mar. 2021.

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

[12] 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.

[13] P. Festa, "The Shortest Path Tour Problem: Problem Definition, Modeling, and Optimization," in *Proc. of INOC*, 2009, pp. 1–7.

[14] 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.

[15] 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.

[16] 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. 28, no. 1, pp. 222–261, Jan. 2021.

[17] 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 Conference on Computer Communications*, Apr. 2018, pp. 2132–2140.

[18] 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*, vol. 7, pp. 165 399–165 410, 2019.

[19] G. Sun, Y. Li, H. Yu, A. V. Vasilakos, X. Du, and M. Guizani, "Energy-Efficient and Traffic-Aware Service Function Chaining Orchestration in Multi-Domain Networks," *Future Generation Computer Systems*, vol. 91, pp. 347–360, Feb. 2019.

[20] 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.

[21] 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.

[22] A. Schrijver, *Theory of Linear and Integer Programming*, reprinted ed. Chichester: Wiley, 2000.

[23] 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.

[24] 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.

[25] ILOG, "IBM ILOG CPLEX Optimizer," https://www.ibm.com/products/ilog-cplex-optimization-studio, 2021, Accessed 14 Apr. 2021.