IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 19, 2019, accepted August 7, 2019, date of publication August 12, 2019, date of current version November 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934725

Virtual Network Function Placement for Service Chaining by Relaxing Visit Order and Non-Loop Constraints

NAOKI HYODO, (Student Member, IEEE), TAKEHIRO SATO, (Member, IEEE), RYOICHI SHINKUMA, (Senior Member, IEEE), AND EIJI OKI, (Fellow, IEEE)
Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan

Corresponding author: Naoki Hyodo (nhyodo@icn.cce.i.kyoto-u.ac.jp)

This work was supported in part by the JSPS KAKENHI, Japan, under Grant 15K00116 and Grant 18H03230.

ABSTRACT Network Function Virtualization (NFV) is a paradigm that virtualizes traditional network functions and instantiates Virtual Network Functions (VNFs) as software instances separate from hardware appliances. Service Chaining (SC), seen as one of the major NFV use cases, provides customized services to users by concatenating VNFs. A VNF placement model for SC that relaxes the visit order constraints of requested VNFs has been considered. Relaxing the VNF visit order constraints reduces the number of VNFs which need to be placed in the network. However, since the model does not permit any loop within an SC path, the efficiency of utilization of computation resources deteriorates in some topologies. This paper proposes a VNF placement model for SC which minimizes the cost for placing VNFs and utilizing link capacity while allowing both relaxation of VNF visit order constraints and configuration of SC paths including loops. The proposed model determines routes of requested SC paths, which can have loops, by introducing a logical layered network generated from an original physical network. This model is formulated as an Integer Linear Programming (ILP) problem. A heuristic algorithm is introduced for the case that the ILP problem is not tractable. Simulation results show that the proposed model provides SC paths with smaller cost compared to the conventional model.

INDEX TERMS Optimization, integer linear programming, Heuristic algorithms, network function virtualization, virtual network function, service chaining.

I. INTRODUCTION

Network functions such as firewalls and deep packet inspections are traditionally implemented as dedicated hardware devices, which achieve high reliability and performance. However, network operators suffer high deployment cost and a lot of troublesome configurations when they provide network services for users. When the network operators start a certain network service, it is necessary to introduce specific devices that provide network functions required by the network service. The office and rack spaces need for device installation must then be secured along with the power supply. Finally, the devices must be manually installed and set up. Every time a new service is introduced, it is necessary to reconfigure the existing system to satisfy the requirements of the new service.

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

Network Function Virtualization (NFV) is a concept of network architecture which decouples network functions from dedicated hardware to offer a solution for addressing the above difficulties and reducing capital and operating expenditures [1], [2]. NFV implements network functions as Virtual Network Functions (VNFs) running on commercial off-the-shelf (COTS) equipment, i.e., commodity servers [3], [4]. NFV provides the network operators with several key advantages. First, by virtualizing the network function and implementing it on relatively inexpensive servers, the cost for purchasing and installing dedicated hardware is reduced [5]. Second, the network operators can operate VNFs as a cloud when the VNFs are aggregated in a data center, which is effective in reducing energy consumption. Third, VNF addition, deletion, and alteration can be realized on demand simply by changing the programs on the servers, so users can receive customized services according to their needs. Finally, since the installation of dedicated hardware is

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

not required when introducing a new service, NFV reduces the time to bring the service to market.

Service Chaining (SC) is one of the main use cases of NFV [6], [7]. SC provides a set of VNFs concatenated in a chain form (service chain) to users as customized services. A path of SC for each user traverses an ordered list of VNFs in the network to comply with the user's service requirement.

In order to provide SC to users, the network operator needs to determine the placement of VNFs and the route of SC paths which satisfy the users' requirements, including the VNF visit order, computation resources for running the VNFs, bandwidth, and end-to-end latency. This is called the VNF placement problem [8]. VNF placement and path routing should be determined appropriately to efficiently utilize the computation resources of servers and distribute the load of the network; the goal is to increase the number of service chains that the network can accommodate.

The existing work [9] presented a VNF placement model that computes the allocation of VNFs and an SC path for each SC request simultaneously in consideration of the relaxation of VNF visit order. The relaxation of the VNF visit order constraints provides flexibility in the VNF placement and saves the number of VNF instances that need to be placed [10]. It has been shown that the order of some types of VNFs, such as a firewall, can be exchanged without dropping the performance of SC [9]. However, since the model presented in [9] does not permit any loop within the SC path, the efficiency of utilization of computation resources deteriorates in some topologies.

The relaxation of both VNF visit order and non-loop constraints within the SC path helps the network operator to use computation resources more efficiently. This is because allowing SC paths to create loops enables flexible route selection and promotes the sharing of VNF instances among multiple service chains, especially in a network having *cut vertices* [11]; a cut vertex is defined as a vertex whose deletion increases the number of connected components of the network. Several works on the VNF placement problem permit the creation of loops in SC routing [12], [13]. However, no study addresses VNF placement by relaxing both the VNF visit order and the non-loop constraint along an SC path.

This paper proposes a VNF placement model, which is referred to as Visit order and Routing constraint Relaxation (VoR-R) hereafter. This paper is an extended version of [14]. VoR-R determines the placement of VNF instances and the routes of SC paths under the conditions that some of the VNF visit order constraints are relaxed and creating loops along an SC path is allowed. We introduce a problem that considers the VNF placement in a logical layered network, which is generated by superposing topologies of an original physical network, to obtain SC paths with loops. This problem is formulated as an Integer Linear Programming (ILP) problem which minimizes the cost for placing VNF instances and utilizing link capacity for providing service chains. The number of feasible solutions of VNF placement obtained by the ILP approach increases by introducing a logical layered network. We develop a heuristic algorithm for VoR-R for the case that the ILP approach becomes intractable. In our heuristic algorithm, we introduce a VNF placement model called Visit order constraint Relaxation (Vo-R), which does not make any loop on an SC path and is more tractable than VoR-R. Our heuristic algorithm sets the initial VNF placement state to a solution obtained by solving Vo-R. The algorithm merges a pair of VNF instances of the same type on the network into one VNF instance to reduce the cost. The route of the SC path which uses the merged VNF instance is recalculated under the condition that the creation of loops is allowed. We evaluate the performances of VoR-R on several topologies, including such with cut vertices, in terms of the total cost for providing service chains and the computation time. Simulation results show that VoR-R provides SC paths with lower cost compared to the conventional model, especially in networks that have cut vertices.

The rest of the paper is organized as follows. Section II presents related works. Section III describes VoR-R including the ILP approach and the heuristic algorithm. Section IV evaluates the performance of VoR-R to show the effect of relaxing both VNF visit order and non-loop constraint at a time. Finally, Section V concludes this paper.

II. RELATED WORKS

A. TECHNIQUES FOR REALIZING SERVICE CHAINING

One of the approaches to realize SC is to utilize Software-Defined Networking (SDN) technology to set up SC paths [15], [16]. An SDN controller sets flow rules into a flow table of each SDN switch so that the switch checks packets and forwards them to a specific next switch. SC paths are set up in two flow management modes as follows.

- Reactive mode: This mode is utilized in the network where flow tables of SDN switches are too small to accommodate flow rules of all SC paths at a time. Flow rules are set on demand in the flow tables by the SDN controller. When a new SC request occurs and the first packet arrives at the edge switch of the SDN network, the switch checks whether there is any flow rule for that packet in the flow table. If any flow rule for the request does not exist, the SDN controller is notified of the arrival of the new SC request by the switch, and computes a suitable path for the SC request. The flow rule will expire after a predefined timeout and wiped out from the flow table.
- Proactive mode: The SDN controller sets flow rules required for SC in advance in the flow table of each switch. The switch forwards arriving packets according to the flow rule, so the proactive mode avoids the extra delay for processing new flows in the reactive mode. Since all required flow rules need to be set in switches, the proactive mode is not feasible if the flow tables of the switches have limited spaces. Therefore,

the proactive mode requires switches with large flow tables.

Both flow management modes have advantages and disadvantages. Since both modes are not mutually exclusive, it is common to set some flows proactively and the remaining flows reactively [17].

Using Network Service Headers (NSH) [18], [19] is another approach to realize SC. An NSH is a data plane header format that is composed of an SC path identification, indication of location within an SC path, and an optional per-packet metadata. An NSH is typically inserted to the packet by switches and often on ingress into a network.

B. EXISTING STUDIES ABOUT SERVICE CHAINING

A number of studies of SC have appeared, some of which [9]–[13], [20], [21] addressed the VNF placement problem that follows from a combination of traffic routing (multicommodity flow problem) and deployment of VNF instances (location problem). Raayatpanah and Weise [21] presented a mathematical model that determines the number and location of VNFs deployed in the network and the optimal traffic flow path according to SC. The work in [21] sets the objective function to minimize the energy consumption of the servers which hold the VNFs.

Carpio *et al.* [20] presented a computation model to solve the VNF placement problem that aims at load balancing of traffic on links. As SC path candidates, multiple routes from the source node to the destination node are computed in advance by the k-shortest path algorithm [22]. Requested VNFs are placed on the obtained multiple path candidates, and then the candidate that offers the greatest dispersion of link load is selected as the SC path. The computation model presented in [20] does not permit SC with loops along a path since the VNFs are placed on paths selected by using the k-shortest path algorithm.

Fang *et al.* [12] examined the VNF placement problem on an elastic optical network connecting data centers. The work in [12] focuses on the fact that the traffic volume of a service chain may change before and after each VNF process [23]. The computation model presented in [12] improves bandwidth utilization efficiency in the elastic optical network by expanding or narrowing the frequency slots allocated to match the change in traffic volume. Paths between arbitrary nodes are computed by the k-shortest path algorithm and input to the computation model. Hmaity *et al.* [13] addressed the VNF placement problem of creating a backup service chain for a primary service chain in order to improve fault tolerance. The computation model presented in [13] determines VNF placement and path routing simultaneously. It is possible to obtain SC paths including loops in [12] and [13] by individually determining the routes of SC paths between each pair of nodes hosting VNFs. Neither study considers the relaxation of VNF visit order constraints.

Allybokus *et al.* [9] presented a VNF placement model that computes the allocation of VNFs and SC paths for each SC request simultaneously in consideration of the relaxation of VNF visit order. In this model, a ternary operator which presents the relative order between any pair of VNFs, including the case that there is no order constraints between the two VNFs, is introduced. The model presented in [9] does not permit any loop within the SC path.

Dwaraki and Wolf [24] designs an algorithm that determines a path that passes through suitable processing nodes in the pre-defined order to satisfy the given SC requirements. In order to transform the original VNF placement problem to a shortest path problem on a different graph, the network graph is transformed into a layered graph, based on their prior work in [25], by adding multiple layers to the graph in this algorithm. Our idea of considering a logical layered network in the ILP approach is based on this layered graph. In this paper, we introduce a logical layered network into the ILP problem so that the creation of loops along an SC path is allowed.

III. PROPOSED MODEL

A. PROBLEM DESCRIPTION

To consider an SC path with loops in VoR-R, we present a problem in which a logical layered network is introduced. We refer to links in the logical layered network as logical links hereinafter. We make multiple duplications of the graph of physical network, and refer to these copies as layers in the resulting graph. Every pair of logical nodes which correspond to the same physical node are connected by vertical logical links. We call the graph constructed by the above procedure as a logical layered network. The logical layered network has the same number of layers as the number of VNF types. We denote the logical node of each layer as (v, t), where v and t are the node identifier and the layer identifier, respectively. Only one type of VNF instances is allowed to be placed on the logical nodes of each layer. Although no loop across layers occurs on a path in the logical layered network, the creation of loops along the SC path is allowed on the corresponding physical network by overlaying each layer after determining a path by VoR-R.

The intuitive example of procedure from routing on the logical layered network (Fig. 1(a)) to the creation of service chain on the physical network (Fig. 1(b)) is shown as below. An SC request which has node 1 as a source node and node 6 as a destination node is presented in Fig. 1(a). The SC request requires VNFs in the order of VNF 1, 3, 4 and 2; all requested VNFs are visit order constrained. The SC path leaves source node (1, 1) and passes through node (2, 1) where VNF 1 is placed, and then moves to node (2, 3) through the logical link. After that it passes through node (3, 3) where VNF 3 is placed and passes to node (4, 3) through the logical link. After heading to node (1, 4) where VNF 4 is placed, it moves to node (1, 2) through the logical link. Finally, it passes through node (4, 2) where VNF 2 is placed and then reaches destination node (6, 2). The resulting path in the logical layered network is mapped to an SC path in the physical network by projecting each layer of the logical layered network back into

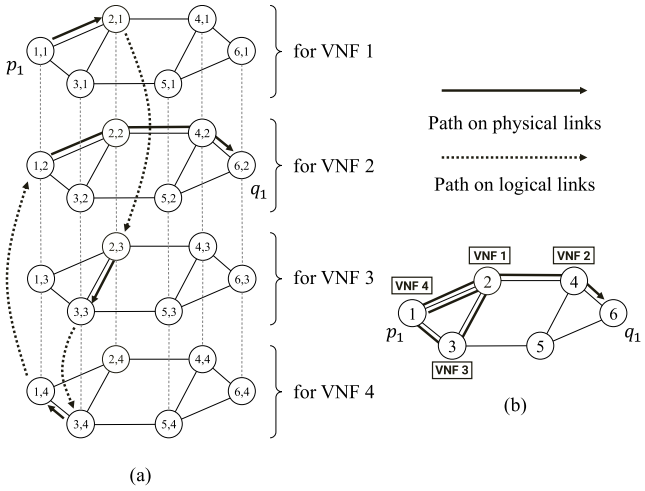


FIGURE 1. Example of creating service chain in VoR-R. (a) Logical layered network. (b) Corresponding physical network.

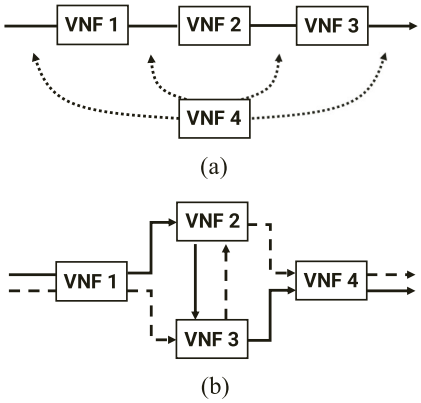


FIGURE 2. Relaxation of VNF visit order supported in VoR-R.

a single layer. Figure 1(b) presents a route of SC path in the physical network; note that the SC path creates a loop in the physical network.

We consider two kinds of VNF visit order relaxations in VoR-R. The first relaxation method supports VNFs without any visit order constraint. In Fig. 2(a), a service chain requests VNF 1, 2, 3 and 4, of which VNF 4 does not have any visit order constraint. When the required visit order for the other VNFs is VNF 1 $\rightarrow$ VNF 2 $\rightarrow$ VNF 3 , there are four possible VNF visit orders.

The second relaxation method supports a set of VNF types that has no visit order constraint among them. In Fig. 2(b), the required VNFs are also VNF 1, 2, 3 and 4 but no order constraint between VNF 2 and 3 is given. Thus visit orders, VNF 1 $\rightarrow$ VNF 2 $\rightarrow$ VNF 3 $\rightarrow$ VNF 4 and VNF 1 $\rightarrow$ VNF 3 $\rightarrow$ VNF 2 $\rightarrow$ VNF 4 , are allowed.

B. INTEGER LINEAR PROGRAMMING FORMULATION

We formulate the problem described in Section III-A as an ILP problem. A physical network is modeled as bidirected graph $G = (V, E)$, where V represents a set of physical nodes and E represents a set of physical links. A logical layered network, which is generated from physical network $G = (V, E)$, is modeled as bidirected graph $G' = (W, J)$. W is a set of logical nodes, each node of which is represented as a tuple of (v, t), $v \in V$, $t \in T$, where T represents a set of layers of the logical layered network. Note that a set of VNF types provided in the network is also represented by T. J represents a set of logical links in the logical layered network. T_m represents a set of VNF types requested by service chain $m \in M$, where m and M are a unique index of an SC request and a set of SC requests, respectively. T_{mk} represents a set of VNF types requested by service chain $m \in M$ in the kth visit order, where $k \in K_m$ represents the visit order of VNFs excluding ones that do not have any visit order constraint. K_m is a set of identification numbers of VNF visit orders requested by service chain $m \in M$. N_m represents a set of VNFs that do not have any visit order constraint. Since T_m contains not only VNFs that have visit order constraints but also VNFs that do not have any visit order constraint, it is clear that the relationship $T_{mk} \subseteq T_m \subseteq T$ is always true.

We consider CPU cores as the computing resources. Each physical node $v \in V$ includes a physical switch which is locally attached to a set of servers that has c_v CPU cores in total, and instances of any VNF type can be deployed in the server. Each physical link $(v, v') \in E$ corresponds to a connection between two physical nodes with bandwidth capacity $b_{vtv't}$ and propagation delay $l_{vtv't}$. The link utilization cost, $\psi^{\mathrm{LINK}}_{vtv't}$, is imposed when an SC path passes through link $(v, v') \in E$. A sufficiently small cost is set for each logical link which connects different layers, $((v, t), (v, t')) \in J, t \neq t'$, to prevent unnecessary routing of an SC path such that the path passes through the same layer multiple times. We assume that a VNF instance placed on a node occupies one CPU core of the node. Therefore, c_v equals the maximum number of VNF instances that node $v \in V$ can accommodate. m_t represents the processing capacity of VNF $t \in T$, i.e. the maximum number of service chains that can share an instance of VNF t. ψ^{VNF}_t represents the cost for placing an instance of VNF $t \in T$ on a node.

An SC request is denoted as $f_m(p_m, q_m, T_m, b_m, l_m)$. $p_m = (p_{mv}, p_{mt})$ and $q_m = (q_{mv}, q_{mt})$ are the source and destination nodes, respectively. T_m consists of subsets $T_{m1}, T_{m2}, \cdots, T_{m|K_m|}$, and N_m. b_m is the requested bandwidth of service chain $m \in M$. l_m is the maximum latency between the source and destination nodes which service chain $m \in M$ tolerates.

The decision variables in the ILP problem are represented as follows. $x^m_{vtv't'}$ is the binary variable that equals one when the path of service chain $m \in M$ passes through logical link $((v, t), (v', t')) \in J$ and zero otherwise. F^m_{vt} is the binary variable that equals one when service chain $m \in M$ utilizes VNF $t \in T_m$ at physical node $v \in V$ and zero otherwise. h_{vt} is the integer variable which represents the number of instances of VNF $t \in T$ placed on physical node $v \in V$. U^m_{vt} is the integer variable which represents the *height* parameter

of node $(v, t) \in W$ on the path of service chain $m \in M$. The height parameter is incremented each time the SC path passes through a node.

The objective function to minimize is the total cost for VNF placement and link utilization while accommodating all service chain requests, as

Minimize:

$$\sum_{m\in M}\sum_{((v,t),(v',t'))\in J}\psi^{\mathrm{LINK}}_{vtv't'}b_m x^m_{vtv't'}+\sum_{v\in V}\sum_{t\in T}\psi^{\mathrm{VNF}}_t h_{vt}, \quad (1)$$

where $\psi^{\mathrm{LINK}}_{vtv't'}$ and ψ^{VNF}_t are the costs for utilizing logical link $((v, t), (v', t')) \in J$ and placing an instance of VNF $t \in T$, respectively, which should be set based on the actual situation by network operators. The first term, excluding $\psi^{\mathrm{LINK}}_{vtv't'}$, represents the sum of the required bandwidth capacities of all links passed by SC paths. The second term, excluding ψ^{VNF}_t, represents the total number of placed VNF instances. The efficiency of resource utilization in the servers improves if we reduce the second term.

$$\sum_{\substack{(v',t')\in W:((v,t),\\(v',t'))\in J}} x^m_{vtv't'} - \sum_{\substack{(v',t')\in W:((v',t'),\\(v,t))\in J}} x^m_{v't'vt} = 1,$$
$$\forall m \in M, \text{ if } (v, t) = (p_{mv}, p_{mt}) \quad (2)$$

$$\sum_{\substack{(v',t')\in W:((v,t),\\(v',t'))\in J}} x^m_{vtv't'} - \sum_{\substack{(v',t')\in W:((v',t'),\\(v,t))\in J}} x^m_{v't'vt} = 0,$$
$$\forall m \in M, (v, t)(\neq (p_{mv}, p_{mt}), (q_{mv}, q_{mt})) \in W \quad (3)$$

$$\sum_{m\in M}\sum_{t\in T} b_m x_{vtv't} \le b_{vtv't}, \forall((v, t), (v', t)) \in J \quad (4)$$

$$\sum_{((v,t),(v',t'))\in J} x_{vtv't} l_{vtv't} \le l_m, \quad \forall m \in M \quad (5)$$

Equations (2) and (3) express the conditions of flow conservation. Equation (2) defines the flow of SC request $m \in M$ at source node (p_{mv}, p_{mt}). The difference between the number of incoming traffic flows and that of outgoing traffic flows is one at the source node. Equation (3) defines the flow of $m \in M$ at intermediate node (v, t), where $(v, t) \neq (p_{mv}, p_{mt}), (q_{mv}, q_{mt})$. The number of incoming traffic flows at node (v, t) equals that of outgoing traffic flows. The utilization of (2) and (3) satisfies the condition of the flow of m at destination node (q_{mv}, q_{mt}) [26]:

$$\sum_{\substack{(v',t')\in W:((v,t),\\(v',t'))\in J}} x^m_{vtv't'} - \sum_{\substack{(v',t')\in W:((v',t'),\\(v,t))\in J}} x^m_{v't'vt} = -1,$$
$$\forall m \in M, \text{ if } (v, t) = (q_{mv}, q_{mt}). \quad (6)$$

Equations (4) and (5) ensure that the limitation of bandwidth capacity of each link and the requirement of maximum end-to-end latency of each SC request are not violated, respectively.

$$\sum_{v\in V} F^m_{vt} = \begin{cases} 1, & \text{if } t \in T_m \\ 0, & \text{otherwise} \end{cases} \quad \forall m \in M, \forall t \in T \quad (7)$$

$$F^m_{vt} \le \sum_{v\in V}\sum_{t'\in T} x^m_{vtv't'} + \sum_{v'\in V}\sum_{t\in T} x^m_{v't'vt},$$
$$\forall m \in M, \forall v \in V, \forall t \in T_m,$$
$$\forall((v, t), (v', t')), ((v', t'), (v, t)) \in J \quad (8)$$

$$\sum_{v\in V}\sum_{t\in T}\sum_{t'\in T} x^m_{vtvt'} \le |T_m| + 1,$$
$$\forall m \in M, \forall((v, t), (v, t')) \in J \quad (9)$$

$$\sum_{m\in M} F^m_{vt} b_m \le m_t h_{vt}, \quad \forall v \in V, \forall t \in T \quad (10)$$

$$\sum_{t\in T} h_{vt} \le c_v, \quad \forall v \in V \quad (11)$$

$$F^m_{vt} \le h_{vt}, \quad \forall m \in M, \forall v \in V, \forall t \in T \quad (12)$$

Equation (7) states that one node is selected for running VNF $t \in T_m$ that SC request $m \in M$ utilizes. Equation (8) ensures that there is a leaving or coming flow of SC request $m \in M$ on a node if m utilizes one or more VNF instances at that node. Equation (9) limits the number of logical links which service chain $m \in M$ can pass through. This equation eliminates the redundant feasible solutions and contributes to shortening the computation time for solving this ILP problem. Equation (10) ensures that a sufficient number of instances of VNF $t \in T$ are placed on physical node $v \in V$ so that node v accommodates all service chains that utilize VNF t at node v. Equation (11) states that node $v \in V$ cannot support more VNF instances than the number of CPU cores that v equips. Equation (12) states that no service chain can utilize VNF $t \in T$ at node $v \in V$ if no instances of VNF t are placed at node v. This equation also limits the range of feasible solutions and contributes to shortening the computation time.

$$U^m_{p_{mv}p_{mt}} = 0, \forall m \in M \quad (13)$$

$$U^m_{v't'} - U^m_{vt} \ge 1 - |T||V| \cdot (1 - x^m_{vtv't'}),$$
$$\forall m \in M, \forall((v, t), (v', t')) \in J \quad (14)$$

$$U^m_{vt^k} \le U^m_{v't^{k+1}} + |T||V| \cdot (2 - F^m_{vt^k} - F^m_{v't^{k+1}}),$$
$$\forall m \in M, \forall v, v' \in V, \forall t^k \in T_{mk},$$
$$\forall t^{k+1} \in T_{mk+1}, \forall k(\le |K_m| - 1) \in K_m \quad (15)$$

U^m_{vt} represents the height of the node [27] for each SC path, which makes it possible to perform SC while satisfying the VNF visit order constraints imposed by SC request m.

Equation (13) sets the height of the source node of service chain $m \in M$ to zero. Equation (14) ensures that U^m_{vt} is incremented each time the path of service chain $m \in M$ passes through a node. Specifically, $U^m_{v't'} \ge U^m_{vt}$ is satisfied when $1 - x^m_{vtv't'}$ on the right side of (14) equals zero, that is, the SC path passes through logical link $((v, t), (v', t')) \in J$. Equation (15) states the relation of heights of two nodes on the path of service chain $m \in M$, one of which has a VNF that must be visited prior to the VNF in the other node. $U^m_{v't^{k+1}}$ is greater than or equal to $U^m_{vt^k}$ when $2 - F^m_{vt^k} - F^m_{v't^{k+1}}$ on the right side of (15) equals zero, that is, instances of VNF t^k and VNF t^{k+1} are placed on node v and node v', respectively. This allows SC request m to pass through VNFs with visit order constraints, in the order specified by m.

Algorithm 1 Merging VNF (MV) Algorithm

Input: Network topology, set of requests M, set of the types of VNF T and requested VNFs of each request with visit order
Output: Service chains' routes, VNF placement and the total cost

Set the initial VNF placement state to a solution obtained by solving Vo-R.
Define $T_{\text{merge}} \leftarrow T$.
while $T_{\text{merge}} \neq \varnothing$ **do**
  Randomly select one type of VNF from T_{merge} with uniform distribution.
  Make a list of all pairs of VNF instances of the selected type that can be merged.
  Define T_{pair} as the set which consists of the listed pairs of VNF instances.
  while true **do**
    if $T_{\text{pair}} = \varnothing$ **then**
      Delete the selected type of VNF from T_{merge}.
      break
    Randomly select one pair of VNF instances from T_{pair} with uniform distribution.
    The merged VNF is allocated to the node providing the smallest total cost among all the nodes (the temporal VNF placement state).
    if The total cost of the temporal state is smaller than that of the current state **then**
      Adopt the temporal state.
    else Remove the selected pair from T_{pair}.

C. HEURISTIC ALGORITHM

Since the logical layered network takes into account a large number of logical links and nodes duplicated from the original physical network, the ILP approach can be intractable. In this section, we present a heuristic algorithm, namely a merging VNF (MV) algorithm that reduces the total cost by merging a pair of VNF instances placed on the network.

The MV algorithm is illustrated in Algorithm 1. A solution obtained by Vo-R, which is a VNF placement model for SC that calculates paths without any loop and requires shorter computation time than VoR-R, is set as the initial VNF placement state of the algorithm; we observed that using a solution of Vo-R as an initial VNF placement state is suitable to obtain a solution of VoR-R. Details of Vo-R are described in Appendix. Set T_{merge} is initialized as a set of types of VNF at the beginning of the algorithm. A VNF type whose instances are to be merged is randomly selected from T_{merge} with uniform distribution. The MV algorithm makes a list of all pairs of VNF instances that can be merged among the VNF instances whose type is selected, and generates set T_{pair} which is a set of listed pairs. It is possible to merge pairs of VNF instances only when the sum of the number of chains using these VNF instances is smaller than the maximum number of chain that can share one VNF instance. If there are multiple pairs of mergeable VNF instances, one pair is randomly selected from T_{pair} with uniform distribution. If there is no VNF instance pair that can be merged, the selected type of VNF instance is deleted from T_{merge}, and the type of VNF to be merged is selected at random again.

The algorithm searches the placement of merged VNF instance which provides the smallest cost. In this procedure, the algorithm examines all nodes in the network for placing the merged VNF instance. The route of SC path in each case is calculated by applying Dijkstra's shortest path algorithm to each pair of adjacent VNFs on the path, while maintaining the passing order of requested VNFs.

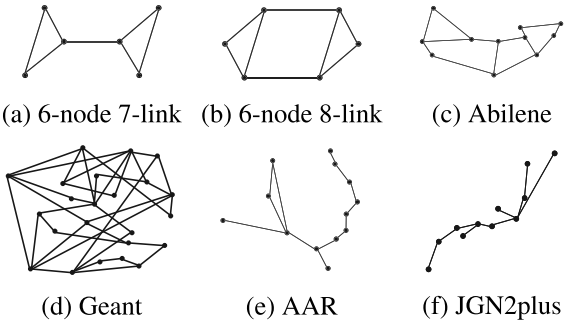
The merged VNF instance is allocated to the node providing the smallest total cost among all the nodes; we call this state as the temporal VNF placement state. If the total cost of the temporal state is smaller than that of the current VNF placement state, the temporal state is adopted, and the type of VNF to be merged is randomly selected again. Otherwise, the selected pair is deleted from T_{pair} and another pair is selected randomly from T_{pair} again. The above series of operations is iterated until T_{merge} becomes empty set. That is, the MV algorithm terminates when cost is not reduced even if any pair of VNF instances on the network are merged. The MV algorithm terminates also when there is no pair of VNF instances that can be merged.

IV. PERFORMANCE EVALUATION

We evaluate the performance of VoR-R to show the effect of relaxing both VNF visit order and non-loop constraint at a time, in terms of the cost and the computation time. We compare them of VoR-R by using the ILP approach and the heuristic MV algorithm with those of Vo-R by using ILP approach. We describe a formulation of a problem in Vo-R which minimizes the cost as an ILP problem in Appendix. In this evaluation, the initial VNF placement state of the MV algorithm is obtained by solving the ILP problem of Vo-R.

A. SIMULATION ENVIRONMENT

The networks used in simulation, which are 6-node 7-link, 6-node 8-link, Abilene, Geant, Australia's Academic and Research (AAR), and Japan's Gigabit Network 2 plus (JGN2plus) networks, are shown in Fig. 3 [28], [29]. The 6-node 7-link and 6-node 8-link networks are used to observe the impact of network topologies on the performance



(a) 6-node 7-link (b) 6-node 8-link (c) Abilene

(d) Geant (e) AAR (f) JGN2plus

FIGURE 3. Networks used in simulation.

of VoR-R and Vo-R. The 6-node 7-link network shown in Fig. 3(a) consists of seven bidirected links removing one link from the 6-node 8-link network shown in Fig. 3(b); the 6-node 7-link network has cut vertices. Since any node, including cut vertices, can be passed only once when the loop is not permitted, the number of possible routes of an SC path is lower as compared with when the creation of loop is allowed. The Abilene, Geant, AAR, and JGN2plus networks are used to compare the performance of VoR-R with the MV algorithm and that of Vo-R with ILP, and also to observe the effect of network topologies. AAR and JGN2plus networks have cut vertices that a long SC path must pass through.

The bandwidth capacity and the propagation delay of each physical link, $b_{vtv't}$ and $l_{vtv't}$, are set to 100 Gbit/s and 10 ms, respectively. The maximum end-to-end latency that each service chain tolerates, l_m, is set to 200 ms. The requested bandwidth of each service chain, b_m, is set to 500 Mbit/s. Each physical node in the networks used in this simulation equips 20 CPU cores, i.e. c_v is set to 20. Each CPU core has the same processing performance of 5 Gbit/s, i.e. m_t is $\frac{5\ \text{Gbit/s}}{500\ \text{Mbit/s}} = 10$. In VoR-R, the cost for passing through a logical link which corresponds to a physical link, $\psi^{\text{LINK}}_{vtv't}$, is set to one. The cost for passing through a logical link which connects different layers, $\psi^{\text{LINK}}_{vtvt'}$, is set to a sufficiently small value. In Vo-R, the cost for passing through a physical link, η_{ij}, is set to one. In both VoR-R and Vo-R, the cost for placing a VNF instance of type $t \in T$ at a node, ψ^{VNF}_t, is set to ten in the simulation of the 6-node 7-link and 6-node 8-link networks. On the Abilene, Geant, AAR, and JGN2plus networks, in addition to the evaluation under the above cost condition, we also carry out the evaluation under the following cost condition;

$$\psi^{\text{LINK}}_{vtv't} = \frac{\psi^{\text{VNF}}_t}{\epsilon + 1}, \quad \forall((v,t),(v',t)) \in J, t \in T, \tag{16}$$

$$\eta_{ij} = \frac{\psi^{\text{VNF}}_t}{\epsilon' + 1}, \quad \forall(i,j) \in E, t \in T, \tag{17}$$

where ϵ and ϵ' are the maximum values in the second terms of (1) and (18), respectively. The cost condition of (16) and (17) minimizes the number of placed VNFs.

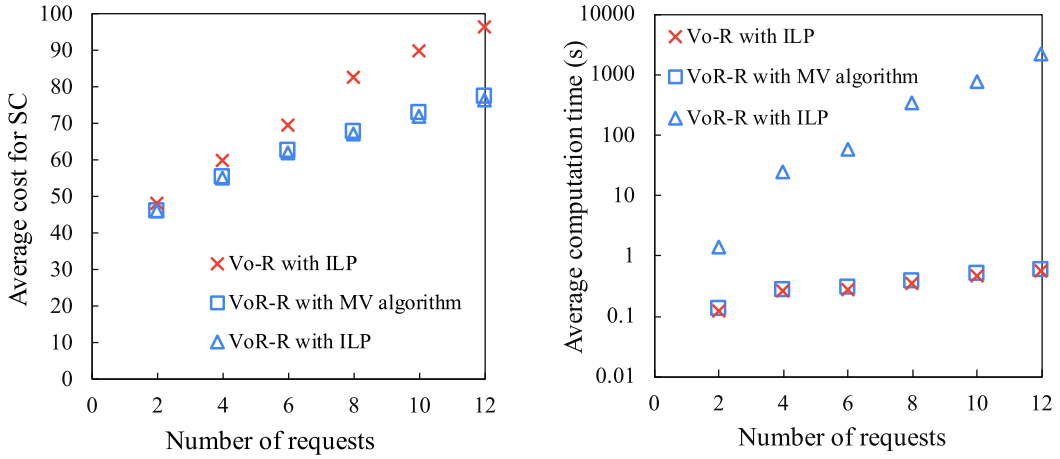
The source and destination nodes of each service chain are randomly selected from all physical nodes in V. On AAR and JGN2plus networks, evaluation is performed within the range up to 100 requests. On the other networks, since the computation time is large, the evaluation is performed within the smaller range of requests compared to AAR and JGN2plus networks. In the simulation of all networks, each service chain requests $1 \leq |T_m| \leq 4$ VNFs. VNFs that each service chain requests are randomly chosen. The number of types of VNFs that can be requested by service chains is $|T| = 5$ in simulations on the 6-node 7-link and 6-node 8-link networks, and it is $|T| = 10$ in simulations on the Abilene, Geant, AAR, and JGN2plus networks. $|K_m|$ is randomly selected among 1 to $|T_m|$. Each of the requested VNFs is randomly put in one of the sets of $T_{mk}, k \in K_m$ or N_m.

100 scenarios are generated for each number of requests on the networks excluding the Geant network, and the average value is obtained for each metrics. 10 scenarios are generated for each number of requests on the Geant network since the computation time increases compared to other networks due to its network size. Since the MV algorithm has randomness, we choose 200 different random seeds in one scenario and solve the MV algorithm for each seed. We choose the best solution among the solutions of all seeds as the solution of the scenario. We use a hardware platform with Intel Core i7-7700 3.60 GHz 4-core CPU and 32 GB RAM to solve the ILP problems and the heuristic MV algorithm. The ILP problems are solved by using CPLEX®Interactive Optimizer 12.7.1.0.

B. SIMULATION RESULTS

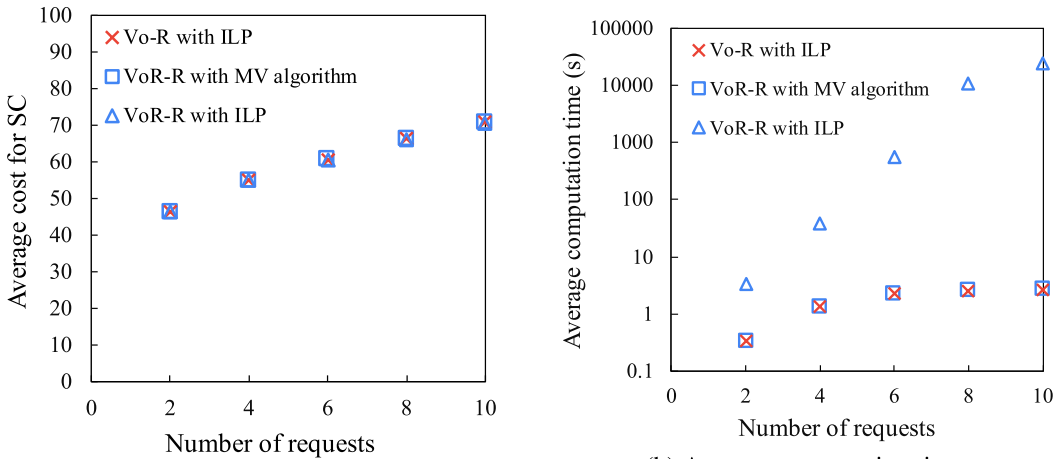
Figure 4 compares the average cost and the computation time of VoR-R and Vo-R on the 6-node 7-link network. The VoR-R achieves lower average cost than that in Vo-R with ILP on the 6-node 7-link network within a range of the number of requests we examined. Compared to Vo-R with ILP, the cost of VoR-R with ILP on the 6-node 7-link network is reduced by up to 20.7%, and that of VoR-R with MV algorithm is reduced by up to 19.8%. The optimality gap between the cost of VoR-R with ILP and that of VoR-R with MV algorithm is at most 1.13%. In terms of the computation time, while Vo-R with ILP and VoR-R with MV algorithm compute the solution in less than 1 second in the measurement range, the computation time for solving VoR-R with ILP reaches 2200 seconds when the number of requests is 12. Figure 5 compares the average cost and the computation time of VoR-R and Vo-R on the 6-node 8-link network. Compared to Vo-R with ILP, the cost of VoR-R with ILP in the 6-node 8-link network is reduced by up to 0.4%, and that of VoR-R with MV algorithm is not reduced; VoR-R has a higher cost reduction rate in the 6-node 7-link network compared to the 6-node 8-link network. It can be observed that VoR-R provides service chains at lower cost compared to Vo-R, especially in networks that have cut vertices.

Figures 6 to 9 show the results of average cost and average computation time, respectively, of the Abilene, Geant, AAR, and JGN2plus networks. Figures 6 and 7 are the results under the simulation environment that the cost for passing through link and the cost for placing a VNF instance are balanced.



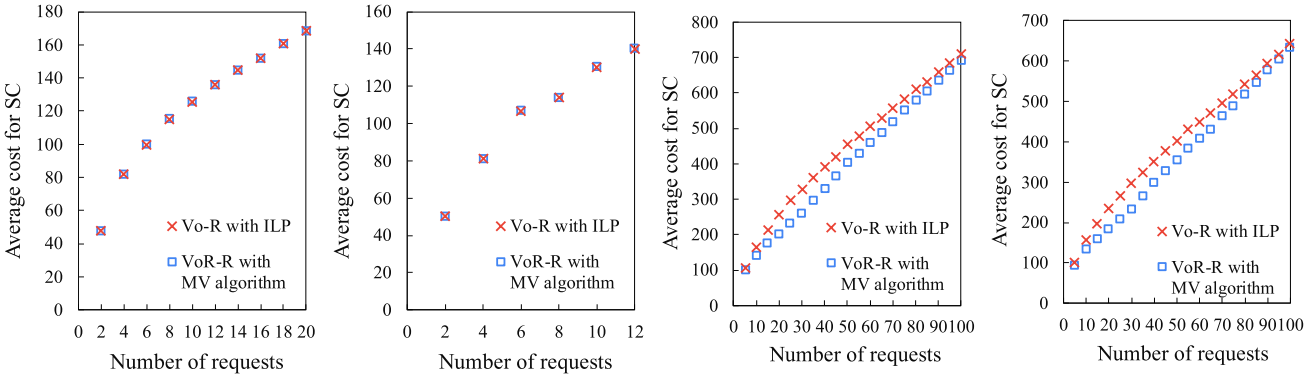
(a) Average cost for serving SC. (b) Average computation time.

FIGURE 4. Comparisons of average cost and computation time in the 6-node 7-link network.



(a) Average cost for serving SC. (b) Average computation time.

FIGURE 5. Comparisons of average cost and computation time in the 6-node 8-link network.



(a) Abilene (b) Geant (c) AAR (d) JGN2plus

FIGURE 6. Average cost for serving SC when link cost and VNF placement cost are balanced.

Figures 8 and 9 are the results under the simulation environment that the reduction of VNF placement cost is prioritized. VoR-R with MV algorithm achieves higher cost reduction rates than Vo-R with ILP in AAR and JGN2plus networks, especially in the simulation environment that the reduction of VNF placement cost is prioritized. Both networks have cut

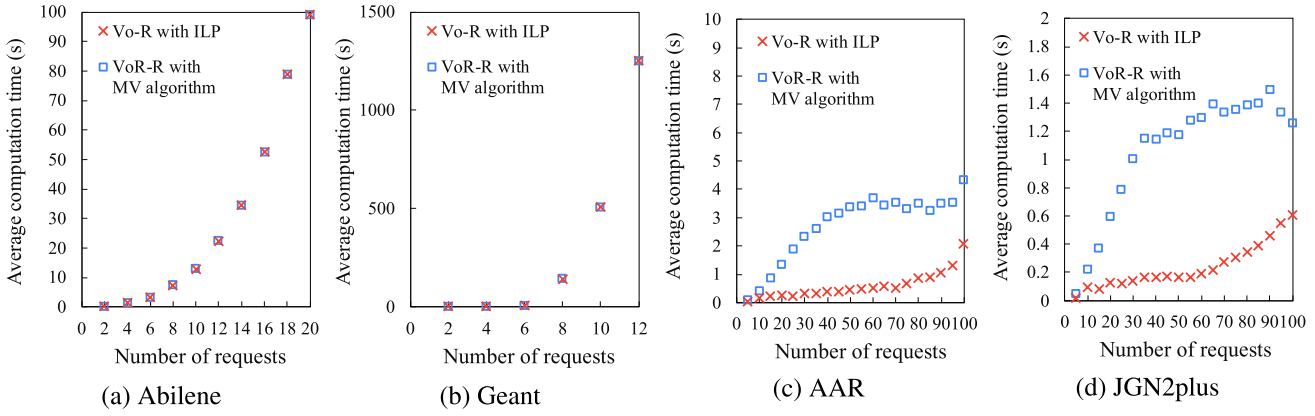


FIGURE 7. Average computation time when link cost and VNF placement cost are balanced.

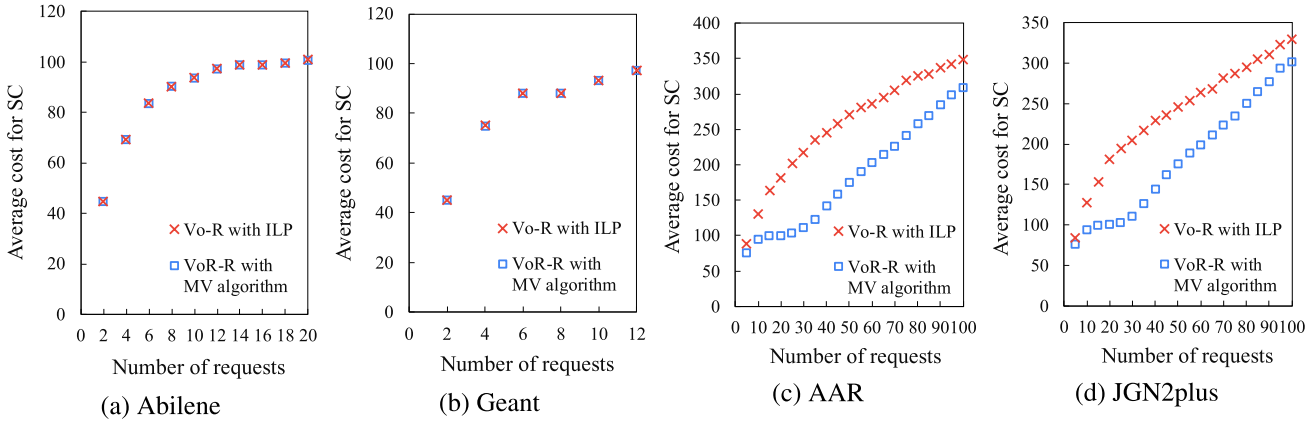


FIGURE 8. Average cost for serving SC when reduction of VNF placement cost is prioritized.

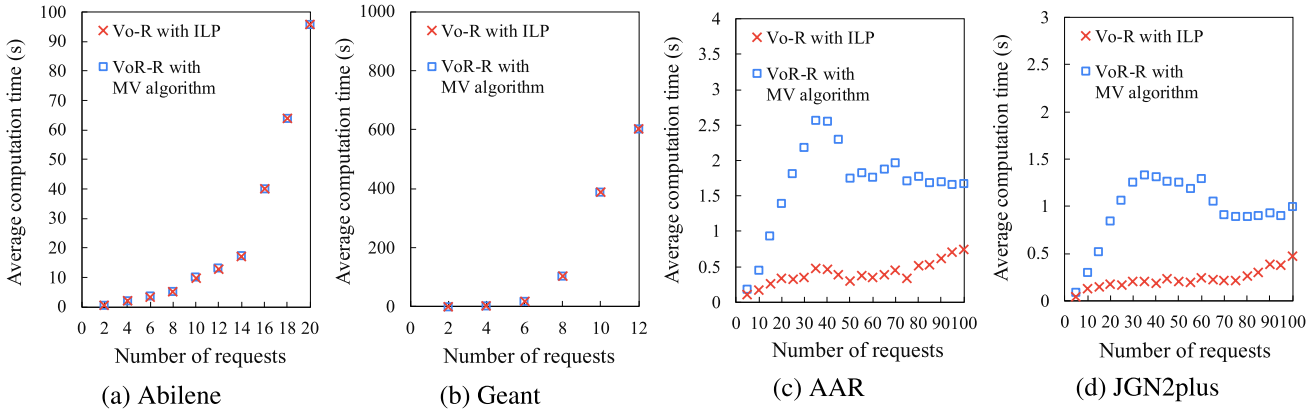


FIGURE 9. Average computation time when reduction of VNF placement cost is prioritized.

vertices that a long SC path must pass through. In Vo-R, this cut vertex can be passed only once, and the variation of an SC path is restricted. VoR-R solves this problem by allowing SC paths to make loops, which leads to further sharing of VNFs between SC paths and the reduction in the number of placed VNFs. In many cases, the number of hops of an SC path increases by merging VNF instances in VoR-R with MV algorithm. Therefore, the cost reduction rate is higher under the condition that the reduction of VNF placement cost is prioritized (Fig. 8) than that of the condition that link cost and VNF placement cost are balanced (Fig. 6). It can be observed that the cost reduction rate is maximized when the number of requests reaches a certain value, and the cost reduction rate decreases as it gets larger, on the

AAR and JGN2plus networks. For instance, on the AAR network, the cost reduction rate to Vo-R with ILP becomes the maximum when the number of requests is 25. It can be said that when the number of requests is around 25, the number of SC paths sharing a VNF instance is small, and there is a high possibility of merging a pair of VNF instances on the network. As the number of requests increases more than 25, the cost reduction rate decreases following the reduction of the number of pairs of mergeable VNFs. The Abilene and Geant network, as shown in Fig. 3, do not have any cut vertices like the 6-node 8-link network. Hence, the cost reduction rate is smaller than that obtained on AAR and JGN2plus networks. Figs. 7 and 9 show that the computation time of VoR-R with MV algorithm increases compared to that of Vo-R with ILP. The increase in computation time depends on the number of random seeds chosen when we solve VoR-R with MV algorithm.

V. CONCLUSION

This paper proposed a VNF placement model for SC named VoR-R, which relaxes both VNF visit order and non-loop constraint within SC paths. A problem was presented in VoR-R, which considers the VNF placement in a logical layered network, in order to obtain SC paths with loops in an original physical network. This problem was formulated as an ILP problem. For the case that the ILP approach becomes intractable, we developed a heuristic algorithm named the MV algorithm that merges a pair of VNF instances placed on the network to reduce the cost. Simulation results showed that VoR-R provides service chains with lower cost compared to Vo-R, which does not allow the creation of loops within an SC path. It was observed that the cost reduction effect of VoR-R becomes large in networks that have cut vertices.

APPENDIX
VISIT ORDER CONSTRAINT RELAXATION

In this appendix, we describe Vo-R, a VNF placement model for SC that relaxes VNF visit order and does not make loops on an SC path. We formulate a problem in Vo-R as an ILP problem. The main difference with the ILP approach in VoR-R is that Vo-R does not introduce logical layered network. The network is modeled as bidirected graph $G = (V, E)$. The model parameters and the decision variables used in the ILP problem is shown in TABLE 1. The definition of the sets used in the ILP problem is the same as that of VoR-R (see Section III-B).

The objective function is to minimize the total cost for VNF placement and link utilization while accommodating all service chain requests, as

Minimize:

$$\sum_{m\in M}\sum_{(i,j)\in E}\eta_{ij}b_m x_{ij}^m + \sum_{v\in V}\sum_{t\in T}\psi_t^{\mathrm{VNF}}h_{vt} \tag{18}$$

where η_{ij} and ψ_t^{VNF} are the costs for passing link (i, j) and placing an instance of VNF t, respectively. The first term, excluding η_{ij}, represents the sum of the required bandwidth capacities of all links passed by SC paths. The second term, excluding ψ_t^{VNF}, represents the total number of placed VNF instances.

TABLE 1. Parameters and decision variables description.

Parameters	Meaning
p_m	source node of service chain $m \in M$
q_m	destination node of service chain $m \in M$
b_m	requested bandwidth of service chain $m \in M$
c_v	maximum number of VNF instances that can be placed on node $v \in V$
m_t	maximum number of service chains that can share VNF t
b_{ij}	bandwidth capacity of link $(i, j) \in E$
l_{ij}	latency of link $(i, j) \in E$
l_m	maximum tolerated latency for service chain $m \in M$
η_{ij}	cost of traffic flow on $(i, j) \in E$
ψ_t^{VNF}	cost for running VNF t
Decision variables	**Meaning**
x_{ij}^m	1 if service chain $m \in M$ is using link $(i, j) \in E$
F_{vt}^m	1 if VNF $t \in T$ from service chain $m \in M$ is allocated in node $v \in V$
h_{vt}	number of VNF $t \in T$ located on node $v \in V$
U_v^m	*height* parameter increases as the SC path of $m \in M$ passes through the node

$$\sum_{j:(i,j)\in E} x_{ij}^m - \sum_{j:(j,i)\in E} x_{ji}^m = 1, \quad \forall m \in M, \text{if } i = p_m \tag{19}$$

$$\sum_{j:(i,j)\in E} x_{ij}^m - \sum_{j:(j,i)\in E} x_{ji}^m = 0, \quad \forall m \in M, \; i(\neq p_m, q_m) \in V \tag{20}$$

$$\sum_{m\in M} b_m x_{ij}^m \leq b_{ij}, \quad \forall (i,j) \in E \tag{21}$$

$$\sum_{(i,j)\in E} x_{ij}^m l_{ij} \leq l_m, \quad \forall m \in M \tag{22}$$

Equations (19) and (20) express the conditions of flow conservation. The utilization of (19) and (20) satisfies the below (23), which presents the condition of the flow of m at destination node q_m:

$$\sum_{j:(i,j)\in E} x_{ij}^m - \sum_{j:(j,i)\in E} x_{ji}^m = -1, \quad \forall m \in M, \text{if } i = q_m. \tag{23}$$

Equations (21) and (22) ensure that the limitation of bandwidth capacity of each link and the requirement of maximum end-to-end latency of each SC request are not violated, respectively.

$$\sum_{v\in V} F_{vt}^m = \begin{cases} 1, & \text{if } t \in T_m \\ 0, & \text{otherwise} \end{cases} \quad \forall m \in M, \forall t \in T \tag{24}$$

$$F_{vt}^m \leq \sum_{j\in V}(x_{vj}^m + x_{jv}^m), \quad \forall m \in M, \forall (v,j),(j,v) \in E, \forall t \in T_m \tag{25}$$

$$\sum_{m\in M} F_{vt}^m b_m \leq m_t h_{vt}, \quad \forall t \in T, \forall v \in V \tag{26}$$

$$\sum_{t \in T} h_{vt} \leq c_v, \quad \forall v \in V \tag{27}$$

$$F^m_{vt} \leq h_{vt}, \quad \forall m \in M, \forall v \in V, \forall t \in T \tag{28}$$

Equation (24) states that one node is selected for running VNF $t \in T_m$ that SC request $m \in M$ utilizes. Equation (25) ensures that there is a leaving or coming flow of SC request $m \in M$ on a node if m utilizes one or more VNF instances at that node. Equation (26) ensures that a sufficient number of instances of VNF $t \in T$ are placed on node $v \in V$ so that node v accommodates all service chains that utilize VNF t at node v. Equation (27) states that node $v \in V$ cannot support more VNF instances than the number of CPU cores that v equips. Equation (28) states that no service chain can utilize VNF $t \in T$ at node $v \in V$ if no instances of VNF t are placed at node v.

$$U^m_{p_m} = 0, \quad \forall m \in M \tag{29}$$

$$U^m_j - U^m_i \geq 1 - |V| \cdot (1 - x^m_{ij}), \quad \forall m \in M, \forall (i,j) \in E \tag{30}$$

$$U^m_i \leq U^m_j + |V| \cdot (2 - F^m_{it} - F^m_{jt'}), \quad \forall m \in M, \forall i,j \in V, \forall t \in T_{mk}, \forall t' \in T_{mk+1}, \forall k (\leq k_m - 1) \in K_m \tag{31}$$

U^m_v represents the height of the node for each SC path. Equation (29) sets the height of the source node of service chain $m \in M$ to zero. Equation (30) ensures that U^m_{vt} is incremented each time the path of service chain $m \in M$ passes through a node. Equation (31) states the relation of heights of two nodes on the path of service chain $m \in M$, one of which has a VNF that must be visited prior to the VNF in the other node.

REFERENCES

[1] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Commun. Mag.*, vol. 53, no. 2, pp. 90–97, Feb. 2015.

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

[3] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[4] T. Taleb, M. Bagaa, and A. Ksentini, "User mobility-aware virtual network function placement for virtual 5G network infrastructure," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 3879–3884.

[5] T. Lin, Z. Zhou, M. Tornatore, and B. Mukherjee, "Optimal network function virtualization realizing end-to-end requests," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2015, pp. 1–6.

[6] J. Halpern and C. Pignataro, *Service Function Chaining (SFC) Architecture*, document RFC 7665, Oct. 2015.

[7] W. John, K. Pentikousis, G. Agapiou, E. Jacob, M. Kind, A. Manzalini, F. Risso, D. Staessens, R. Steinert, and C. Meirosu, "Research directions in network service chaining," in *Proc. IEEE SDN Future Netw. Services*, Nov. 2013, pp. 1–7.

[8] X. Li and C. Qian, "The virtual network function placement problem," in *Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2015, pp. 69–70.

[9] Z. Allybokus, N. Perrot, J. Leguay, L. Maggi, and E. Gourdin, "Virtual function placement for service chaining with partial orders and anti-affinity rules," Aug. 2017, *arXiv:1705.10554*. [Online]. Available: https://arxiv.org/abs/1705.10554

[10] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *Proc. IEEE 4th Int. Conf. Cloud Netw. (CloudNet)*, Oct. 2015, pp. 171–177.

[11] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*, vol. 8. Cambridge, U.K.: Cambridge Univ. Press, 1994.

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

[13] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, "Virtual network function placement for resilient service chain provisioning," in *Proc. 8th Int. Workshop Resilient Netw. Design Model.*, Sep. 2016, pp. 245–252.

[14] N. Hyodo, T. Sato, R. Shinkuma, and E. Oki, "Virtual network function placement model for service chaining to relax visit order and routing constraints," in *Proc. IEEE 7th Int. Conf. Cloud Netw.*, Oct. 2018, pp. 1–3.

[15] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

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

[17] W. Braun and M. Menth, "Software-defined networking using OpenFlow: Protocols, applications and architectural design choices," *Future Internet*, vol. 6, no. 2, pp. 302–336, 2014.

[18] P. Quinn, U. Elzur, and C. Pignataro, *Network Service Header (NSH)*, document RFC 8300, Jan. 2018.

[19] P. Quinn and J. Guichard, "Service function chaining: Creating a service plane via network service headers," *Computer*, vol. 47, no. 11, pp. 38–44, Nov. 2014.

[20] F. Carpio, S. Dhahri, and A. Jukan, "VNF placement with replication for loac balancing in NFV networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[21] M. A. Raayatpanah and T. Weise, "Virtual network function placement for service function chaining with minimum energy consumption," in *Proc. IEEE Int. Conf. Comput. Commun. Eng. Technol. (CCET)*, Aug 2018, pp. 198–202.

[22] D. Eppstein, "Finding the K shortest paths," *SIAM J. Comput.*, vol. 28, no. 2, pp. 652–673, 1999.

[23] W. Ma, C. Medina, and D. Pan, "Traffic-aware placement of NFV middleboxes," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2015, pp. 1–6.

[24] A. Dwaraki and T. Wolf, "Adaptive service-chain routing for virtual network functions in software-defined networks," in *Proc. ACM Workshop Hot Topics Middleboxes Netw. Funct. Virtualization*, Aug. 2016, pp. 32–37.

[25] S. Choi, J. Turner, and T. Wolf, "Configuring sessions in programmable networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, vol. 1, Apr. 2001, pp. 60–66.

[26] E. Oki, *Linear Programming and Algorithms for Communication Networks*. Boca Raton, FL, USA: CRC Press, Aug. 2012.

[27] N. Charbonneau and V. M. Vokkarane, "Static routing and wavelength assignment for multicast advance reservation in all-optical wavelength-routed WDM networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 1–14, Feb. 2012.

[28] M. Roughan, "A case study of the accuracy of SNMP measurements," *J. Elect. Comput. Eng.*, vol. 2010, Feb. 2010, Art. no. 812979.

[29] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.

NAOKI HYODO (S'18) received the B.E. degree from the Undergraduate School of Electrical and Electronic Engineering, Kyoto University, Kyoto, Japan, in 2018, where he is currently pursuing the master's degree with the Graduate School of Informatics. His research interests include network function virtualization, resource allocation, modeling, optimization, and algorithms.

TAKEHIRO SATO (S'12–M'16) received the B.E., M.E., and Ph.D. degrees in engineering from Keio University, Japan, in 2010, 2011, and 2016, respectively.

From 2011 to 2012, he was a Research Assistant with Keio University Global COE Program, High-level Global Cooperation for Leading-edge Platform on Access Spaces by Ministry of Education, Culture, Sports, Science and Technology, Japan. From 2012 to 2015, he was a Research Fellow of the Japan Society for the Promotion of Science. From 2016 to 2017, he was a Research Associate with the Graduate School of Science and Technology, Keio University. He is currently an Assistant Professor with the Graduate School of Informatics, Kyoto University, Japan. His research interests include communication protocols and network architectures for the next-generation optical networks. He is also a member of IEICE.

RYOICHI SHINKUMA (S'02–M'03–SM'19) received the B.E., M.E., and Ph.D. degrees in communications engineering from Osaka University, Japan, in 2000, 2001, and 2003, respectively.

He was a Visiting Scholar with the Wireless Information Network Laboratory, Rutgers, and the State University of New Jersey, USA, from 2008 to 2009. In 2003, he joined the Faculty of Communications and Computer Engineering, Graduate School of Informatics, Kyoto University, Japan, where he is currently an Associate Professor. His research interest is mainly cooperation in heterogeneous networks. He received the Young Researchers' Award from IEICE, in 2006, the Young Scientist Award from Ericsson, Japan, in 2007, the Telecom System Technology Award from the Telecommunications Advancement Foundation, in 2016, and the Best Tutorial Paper Award from the IEICE Communications Society, in 2019. He was the Chairperson of the Mobile Network and Applications Technical Committee of the IEICE Communications Society, from 2017 to 2019.

EIJI OKI (M'95–SM'05–F'13) received the B.E. and M.E. degrees in instrumentation engineering and the Ph.D. degree in electrical engineering from Keio University, Yokohama, Japan, in 1991, 1993, and 1999, respectively.

From 1993 to 2018, he was with the Nippon Telegraph and Telephone Corporation (NTT) Communication Switching Laboratories, Tokyo, Japan, where he was involved in researching network design and control, traffic control methods, and switching systems. From 2000 to 2001, he was a Visiting Scholar with the Polytechnic Institute of New York University, Brooklyn, NY, USA, where he was involved in designing switch/router systems. From 2008 to 2017, he was with The University of Electro-Communications, Tokyo. He joined Kyoto University, Kyoto, Japan, in 2017, where he is currently a Professor. He has been active in the standardization of the path computation element and GMPLS in the IETF. He has authored over ten IETF RFCs. He has authored/coauthored five books, *Broadband Packet Switching Technologies* (Wiley, 2001), *GMPLS Technologies* (CRC Press, 2005), *Advanced Internet Protocols, Services, and Applications* (Wiley, 2012), *Linear Programming and Algorithms for Communication Networks* (CRC Press, 2012), and *Routing and Wavelength Assignment for WDM-based Optical Networks* (Springer, 2016).

Dr. Oki is a Fellow of IEICE. He was a recipient of several prestigious awards, including the 1999 IEICE Excellent Paper Award, the 2001 IEEE Communications Society Asia-Pacific Outstanding Young Researcher Award, the 2010 Telecom System Technology Prize by the Telecommunications Advanced Foundation, the IEEE HPSR 2012 Outstanding Paper Award, the IEEE HPSR 2014 Best Paper Award Finalist, the First Runner Up, and the 2015 IEICE Achievement Award, the IEEE Globecom 2015 Best Paper Award, the 2016 Fabio Neri Best Paper Award, the Runner up, and the 2018 Excellent Paper Award of Information and Communication Technology on Convergence.