A Scalable Approach for Service Chain (SC) Mapping with Multiple SC Instances in a Wide-Area Network

Abhishek Gupta, Brigitte Jaumard, Massimo Tornatore, and Biswanath Mukherjee

arXiv:1709.04772v1 [cs.NI] 13 Sep 2017

Abstract—Network Function Virtualization (NFV) aims to simplify deployment of network services by running Virtual Network Functions (VNFs) on commercial off-the-shelf servers. Service deployment involves placement of VNFs and in-sequence routing of traffic flows through VNFs comprising a Service Chain (SC). The joint VNF placement and traffic routing is called SC mapping. In a Wide-Area Network (WAN), a situation may arise where several traffic flows, generated by many distributed node pairs, require the same SC; then, a single instance (or occurrence) of that SC might not be enough. SC mapping with multiple SC instances for the same SC turns out to be a very complex problem, since the sequential traversal of VNFs has to be maintained while accounting for traffic flows in various directions.

Our study is the first to deal with the problem of SC mapping with multiple SC instances to minimize network resource consumption. We first propose an Integer Linear Program (ILP) to solve this problem. Since ILP does not scale to large networks, we develop a column-generation-based ILP (CG-ILP) model. However, we find that exact mathematical modeling of the problem results in quadratic constraints in our CG-ILP. The quadratic constraints are made linear but even the scalability of CG-ILP is limited. Hence, we also propose a two-phase column-generation-based approach to get results over large network topologies within reasonable computational times. Using such an approach, we observe that an appropriate choice of only a small set of SC instances can lead to a solution very close to the minimum bandwidth consumption. Further, this approach also helps us to analyze the effects of number of VNF replicas and number of NFV nodes on bandwidth consumption when deploying these minimum number of SC instances.

I. INTRODUCTION

Traditionally, communication networks have deployed network services through proprietary hardware appliances (e.g., network functions such as firewalls, NAT, etc.) which are statically configured. With rapid evolution of applications, networks require agile and scalable service deployment.

Network Function Virtualization (NFV) [1] offers a solution for an agile service deployment. NFV envisions traditional hardware functionality as software modules called Virtual Network Functions (VNFs). VNFs can be run on commercial-off-the-shelf hardware such as servers and switches in datacenters (DCs), making service deployment agile and scalable.

A. Gupta, M. Tornatore, and B. Mukherjee are with the University of California, Davis, USA. E-mail: {abgupta, mtornatore, bmukherjee}@ucdavis.edu; B. Jaumard is with Concordia University, Canada. E-mail: bjaumard@cse.concordia.ca; M. Tornatore is also with Politecnico di Milano, Italy. E-mail: massimo.tornatore@polimi.it

When several network functions are configured to provide a service, we have a "Service Chain". The term "service chain" is used "to describe the deployment of such functions, and the network operator's process of specifying an ordered list of service functions that should be applied to a deterministic set of traffic flows" [2]. So, a "Service Chain" (SC) specifies a set of network functions configured in a specific order. With NFV, we can form SCs where VNFs are configured in a specific sequence that minimizes the bandwidth usage in the network (an example is discussed below).

Unfortunately, since VNFs in a single SC may need to be traversed by several distinct traffic flows (i.e., flows requested by multiple geographically-distributed node pairs) in a specific sequence, it becomes difficult to improve network resource utilization. For example, consider Figs. 1(a) and 1(b), where three traffic requests r_1 (from node 4 to 13), r_2 (from node 6 to 3), and r_3 (from node 14 to 1) demand SC c_1 composed of VNF1, VNF2, and VNF3 (to be traversed in this order VNF1 → VNF2 → VNF3). In Fig. 1(a), if we consider only one mapping occurrence (or instance) for SC c_1, then some traffic flows (in our example, r_3 and r_2) will be ineffectively routed over long paths. Instead, as shown in Fig. 1(b), if we use two SC instances for the same SC, we can improve network resource utilization, at the expense of a larger number of VNFs to be deployed (or replicated) in the network to serve the same SC. This results in a more complex problem when, in a Wide-Area Network (WAN), a large number of distributed node pairs generate traffic flows, creating heavy traffic demands. Our objective in this work is to reduce the network resource consumption for a WAN with heavy traffic demands.

So the question is: how many SC instances for the same SC are required for optimal network resource utilization?

A possible (trivial) solution to the problem of SC mapping in case of multiple node pairs requiring the same SC is to use one single instance that would most likely lead to host SCs at a single node (e.g., a DC) which is centrally located in the network. However, traffic flows may have to take long paths to reach the node hosting the SC, which will result in a high network resource consumption.

The other extreme case would be to use a distinct SC mapping per each node pair (in other words, the number of SC instances is equal to the number of traffic node pairs). Now, we can achieve optimal network resource utilization as each node pair will use an SC effectively mapped along a shortest path
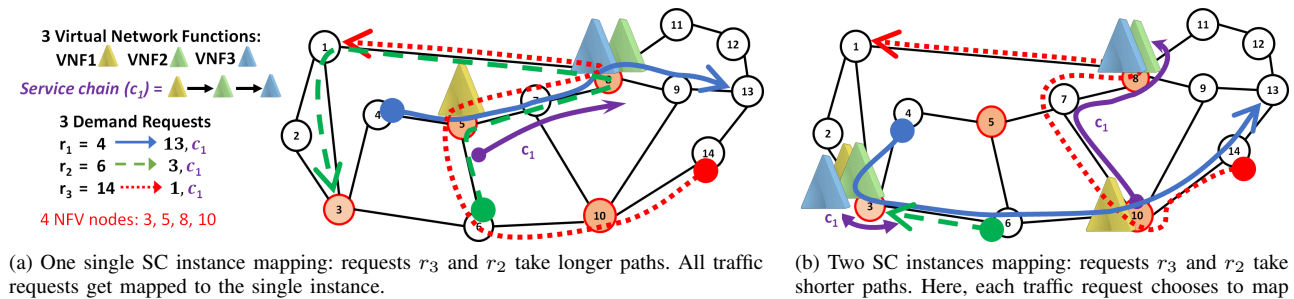
(a) One single SC instance mapping: requests r_3 and r_2 take longer paths. All traffic requests get mapped to the single instance.

(b) Two SC instances mapping: requests r_3 and r_2 take shorter paths. Here, each traffic request chooses to map to one of two given instances.

Fig. 1: Deploying more SC occurrence mappings reduces network resource consumption.

in the network[1]. However, this approach will increase the network orchestration overhead and increase capital expenditure, as there will be a large number of replicated VNF instances across nodes. To reduce excessive VNF replication, we bound the maximum number of nodes hosting VNFs.

Intuitively, the number of SC instances for a good solution will be a value between these two extremes. This solution will minimize the network resource utilization while not excessively increasing the number of nodes hosting VNFs.

A reasonable trade-off that leads to the optimal solution is difficult to calculate, as the problem of SC mapping with multiple SC instances results in quadratic constraints [3] that severely hamper the scalability of the solution. In this study, to answer the question above, we propose a two-phase solution, relying on a column-generation-based ILP model, which provides quasi-optimal solutions with reasonable computational time. Sub-optimality comes from the fact that we solve the problem in two phases: in the first phase, we group node pairs that will be forced to use the same SC instance; in the second phase, we run our scalable column-generation approach to find a solution starting from the grouping already performed in the first phase. Applying this approach over two realistic network topologies, we observe that an appropriate choice of only a small set of different SC mappings can lead to a solution very close to the minimum theoretical bandwidth consumption, even for a full-mesh traffic demand matrix.

The rest of this study is organized as follows. Section II overviews the existing literature on the SC mapping problem and remarks the novel contributions of this study. Section III formally describes the problem and its input parameters. Section IV describes the Integer Linear Program (ILP) formulation for the problem, while Section V describes the quadratic column-generation-based ILP model. Section VI introduces a heuristic to cluster groups of node pairs that will use the same SC instance; and then describes our column-generation-based ILP solution method. Section VIII provides some illustrative examples which demonstrate that a limited number of SC instances can lead to quasi-optimal solution of the problem. Section IX concludes the study.

---

[1]Using the shortest path also has the added effect of reducing latency for the service chain, but this aspect is out of scope for this study.

II. RELATED WORK

A number of studies exist on the VNF placement and routing problem. Ref. [3] was the first to formally define the problem of VNF placement and routing. However, they developed a Quadratic Constrained Program (QCP), making it unscalable beyond small problem instances. Ref. [4] studied a hybrid deployment scenario with hardware middleboxes using an ILP, but did not enforce VNF service chaining explicitly. Ref. [5] used an ILP to study trade-offs between legacy and NFV-based traffic engineering but did not have explicit VNF service chaining. Ref. [6] modeled the problem in a DC setting using an ILP to reduce the end-to-end delays and minimize resource over-provisioning while providing a heuristic to do the same. Here too VNF service chaining is not explicitly enforced by the model. Ref. [7] modeled the batch deployment of multiple chains using an ILP and developed heuristics to solve larger instances of the problem. However, it enforced that VNF instances of a function need to be on a single machine and restricts all chains to three VNFs. Our model does not impose such constraints, and we allow any VNF type to be placed at any node and any number of VNFs in a SC while service chaining VNFs for a SC explicitly. Ref. [8] accounted for the explicit service chaining of VNFs but focused on compute resource sharing among VNFs. Ref. [9] used a column-generation model to solve VNF placement and routing but considered dedicated SC instances per traffic pair, hence solving the second extreme case mentioned in the introduction, which is a particular case of our approach. Ref. [10] also used a column-generation model to solve the dynamic VNF placement and routing problem but considered a single SC instance per SC, which as mentioned earlier will lead to a sub-optimal solution.

Recently, there have been a few works on using multiple VNF instances for load balancing to reducing resource utilization and improve QoS. There are several differences between these and our work which we clarify below. Ref. [11] developed a load-balancing scheme for the Virtual Evolved Packet Core (vEPC) SC, given a set of pre-computed paths by replicating the instances of certain (not all) VNFs. Our approach deals with a SC in general without any prior computation of paths. Ref. [12] developed an online approach for scaling SCs by using VNF replicas and an approximate version of an offline scheme. It provided theoretical bounds for its

technique; however, the method does not provide a general mapping of an SC instance to network node and VNF replicas which is done by our approach. Ref. [13] looked at selection of NFV nodes and VNF assignment separately. Our approach does node selection and VNF assignment jointly while also holistically mapping SC instances to the allowed number of NFV nodes and VNF replicas for each SC.

Our previous work [14] and most existing works solve the problem for multiple SCs, but for each SC only a single instance of the SC is allowed. We remark again that, in the current work, we consider multiple SCs, but for each SC, multiple instances per SC are allowed; hence most existing works represent a particular case of our current work, where each node pair requesting an SC has its own instance. Further, we also consider multiple geographically-distributed node pairs which create heavily-populated (dense) traffic demands. As extending the model to multiple instances per SC results in quadratic constraints, we propose a novel decomposition model (column generation) for SC mapping with multiple SC instances, which, together with a traffic-grouping heuristic, provides a scalable solution to the problem (Section VI).

To the best of our knowledge, this is the first attempt to address the solution of the complete SC mapping problem (i.e., with multiple SC instances) over large network instances.

III. PROBLEM DESCRIPTION

An operator's network provides multiple services, and each service is realized by traversing a Service Chain (SC). To provide multiple services, the operator has to map corresponding SCs into network. We develop three solution approaches for this multiple SC mapping problem. First is the ILP described in Section IV. Second is a column-generation-based ILP (CG-ILP) detailed in Section V. Finally, we solve the problem using a two-phase approach described in Section VI.

A. Problem Statement

Given a network topology, capacity of links, a set of network nodes with NFV support (NFV nodes), compute resources at NFV nodes, maximum number of NFV nodes that can be used, traffic flows for source-destination pairs requiring a specific SC with a certain bandwidth demand, a set of VNFs, and a set of SCs, we determine the placement of VNFs and corresponding traffic routing to minimize network-resource (bandwidth) consumption. Note that VNFs can be shared among different SCs.

B. Input Parameters

- $G = (V, L)$: Physical topology of backbone network; V is set of nodes and L is set of links.
- $V^{\text{NFV}} \subseteq V$: Set of nodes that can host VNFs (NFV nodes).
- I_c: Number of instances for SC c.
- K: Maximum number of NFV nodes allowed to host VNFs.
- F, indexed by f: Set of VNFs.
- R_f: Maximum number of replicas of VNF f.

- n^{CORE}: Number of CPU cores present in a NFV node.
- n_f^{CORE}: Number of CPU cores per Gbps for function f.
- C: Set of chains, indexed by c.
- n_c: Number of VNFs in SC c.
- $\mathcal{SD}$: Set of source-destination (v_s, v_d) pairs.
- $\mathcal{SD}_c$: Set of source-destination (v_s, v_d) pairs for SC c.
- D_{sd}^c: Traffic demand between v_s and v_d for SC c.
- $\sigma_i(c)$: ID of ith VNF in SC c where $f_{\sigma_i(c)} \in F$.
- T_{fi}^c: Utility for translating the ith VNF in SC c to its VNF index f.

To facilitate model formulation and discussion, we propose the concept of configuration ($\hat{\gamma}$). We use the following notation for SC representation. Each SC, denoted by c, is characterized by an ordered set of n_c functions:

$$[\text{SC } c] \qquad f_{\sigma_1(c)} \prec f_{\sigma_2(c)} \prec \cdots \prec f_{\sigma_{n_c}(c)} \qquad (1)$$

Each deployment of SC c is defined by a set of VNF locations, a set of paths, from location of first VNF to location of last VNF, and set of traffoc flows traversing this deployment.

We generate a set of *SC configurations* where each configuration ($\hat{\gamma}$) is associated with a potential provisioning of a SC c, i.e., with a potential node placement of its functions and a potential subset of traffic flows from $\mathcal{SD}_c$. Let $\hat{\Gamma}$ be the set of configurations, and $\hat{\Gamma}_c$ be the subset of configurations associated with service chain $c \in C$: $\hat{\Gamma} = \bigcup_{c \in C} \hat{\Gamma}_c$.

Potential set of configurations for a SC c is given by:

$$\hat{\Gamma}_c = \sum_{sd=1}^{N_{SD_c}} \binom{N_{SD_c}}{sd} \times \{N_{V^{NFV}}\}^{n_c} \times P_{paths}^{n_c - 1}$$

where sd is the number of number of source-destination (v_s, v_d) pairs using a configuration, N_{SD_c} gives the number of source-destination (v_s, v_d) pairs for SC c, $N_{V^{NFV}}$ gives the number of NFV nodes and P_{paths} refers to the number of paths from the location of $f_{\sigma_i(c)}$ to the location of $f_{\sigma_{i+1}(c)}$.

A chain configuration ($\hat{\gamma}$) is characterized by the following parameters:

- Traffic flows: $\delta_{sd}^{\hat{\gamma}} = 1$ if (v_s, v_d) uses configuration $\hat{\gamma}$; 0 otherwise.
- Location of functions: $a_{vi}^{\hat{\gamma}} = 1$ if ith function $f_i \in c$ is located in v in configuration $\hat{\gamma}$; 0 otherwise.
- Connectivity of locations: path from location of current VNF to next VNF in SC c. If link ℓ is used in the path from location of $f_{\sigma_i(c)}$ to location of $f_{\sigma_{i+1}(c)}$, then $b_{i\ell}^{\hat{\gamma}} = 1$; 0 otherwise.

IV. INTEGER LINEAR PROGRAM

We precompute $\hat{\Gamma}$, which is an input for our ILP model. ILP selects the best configuration ($\hat{\gamma}$) based on other input parameters and constraints, and computes the route from v_s (source) to first VNF of c and from last VNF of c to v_d (destination) for each source-destination (v_s, v_d) pair.

Variables:

- $z_{\hat{\gamma}} = 1$ if configuration $\hat{\gamma}$ is selected; 0 otherwise.
- $x_v^{ci} = 1$ if ith function of c is located in v; 0 otherwise.
- $y_\ell^{f_1(c), sd} = 1$ if ℓ is on path from v_s to location of first VNF in c; 0 otherwise.

- $y_\ell^{f_{n_c}(c),sd} = 1$ if ℓ is on path from location of last VNF in c to v_d; 0 otherwise.
- $h_v = 1$ if v is used as a location for a VNF; 0 otherwise.

Objective: Minimize bandwidth consumed:

$$\min \sum_{c \in C} \sum_{\hat{\gamma} \in \hat{\Gamma}_c} \overbrace{\left(\sum_{(s,d) \in \mathcal{SD}} D_{sd}^c \right)}^{\text{Overall traffic using } c} \overbrace{\left(\sum_{\ell \in L} \sum_{i \in I} \delta_{sd}^{\hat{\gamma}} b_{i\ell}^{\hat{\gamma}} \right)}^{\substack{\text{Number of links} \\ \text{in the route of } c}} z_{\hat{\gamma}} +$$
$$\sum_{c \in C} \sum_{\ell \in L} \sum_{(s,d) \in \mathcal{SD}} D_{sd}^c \left(y_\ell^{f_1(c),sd} + y_\ell^{f_{n_c}(c),sd} \right) \quad (2)$$

Total bandwidth consumed in placing multiple SCs depends on configurations ($\hat{\gamma}$'s) selected for each SC c. Each $\hat{\gamma}$ for c locates VNFs of c and gives the route to traverse these VNF locations. So, bandwidth consumed when going from v_s to v_d and traversing the SC depends on selected $\hat{\gamma}$.

Constraints:

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} z_{\hat{\gamma}} \leq I_c \qquad c \in C \quad (3)$$

$$\sum_{c \in C} \sum_{\hat{\gamma} \in \hat{\Gamma}_c} \sum_{i=1}^{n_c} \mathrm{T}_{fi}^c a_{vi}^{\hat{\gamma}} z_{\hat{\gamma}} \leq M x_{vf} \qquad f \in F, v \in V^{\mathrm{NFV}} \quad (4)$$

$$\sum_{c \in C} \sum_{\hat{\gamma} \in \hat{\Gamma}_c} \sum_{i=1}^{n_c} \mathrm{T}_{fi}^c a_{vi}^{\hat{\gamma}} z_{\hat{\gamma}} \geq x_{vf} \qquad f \in F, v \in V^{\mathrm{NFV}} \quad (5)$$

$$\sum_{v \in V^{\mathrm{NFV}}} x_{vf} \leq R_f \qquad f \in F \quad (6)$$

$$M h_v \geq \sum_{f \in F} x_{vf} \geq h_v \qquad v \in V^{\mathrm{NFV}} \quad (7)$$

$$\sum_{v \in V^{\mathrm{NFV}}} h_v \leq K \quad (8)$$

$$\sum_{c \in C} \sum_{\hat{\gamma} \in \hat{\Gamma}_c} \sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c \delta_{sd}^{\hat{\gamma}} *$$
$$\left(\sum_{f \in F} \sum_{i=1}^{n_c} \mathrm{T}_{fi}^c n_f^{\mathrm{CORE}} a_{vi}^{\hat{\gamma}} \right) z_{\hat{\gamma}} \leq \mathrm{N}^{\mathrm{CORE}} \qquad v \in V^{\mathrm{NFV}} \quad (9)$$

$$\sum_{c \in C} \sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c *$$
$$\left(y_\ell^{f_1(c),sd} + y_\ell^{f_{n_c}(c),sd} + \sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} z_{\hat{\gamma}} \sum_{i=1}^{n_c-1} b_{i\ell}^{\hat{\gamma}} \right)$$
$$\leq \mathrm{CAP}_\ell \qquad \ell \in L \quad (10)$$

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} z_{\hat{\gamma}} = 1 \qquad c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0 \quad (11)$$

Constraints (3) guarantee that we select exactly I_c configurations for SC c and force c to have I_c instances. Each $\hat{\gamma}$ is associated with a set of $a_{vi}^{\hat{\gamma}}$ required to be consistent with x_{vf}, which is resolved by Eqs. (4), (5) where T_{fi}^c is to find the VNF f at sequence i in SC c. Eq. (6) is used to limit the number of VNF replicas. Eq. (7) is used to keep track of NFV nodes used for hosting VNFs while Eq. (8) limits the number of NFV nodes allowed to host VNFs. Constraints (9) ensure that each NFV node has a sufficient number of CPU cores for

hosting f. Eq. (10) constrains link capacity. Eq. (11) enforces that, for each source-destination pair (v_s, v_d) requesting SC c, there is exactly one configuration $\hat{\gamma}$.

Route from v_s to first function location:

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} a_{v_s,1}^{\hat{\gamma}} z_{\hat{\gamma}} + \sum_{\ell \in \omega^+(v_s)} y_\ell^{f_1(c),sd} = 1$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0 \quad (12)$$

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} a_{v1}^{\hat{\gamma}} z_{\hat{\gamma}} - \sum_{\ell \in \omega^-(v)} y_\ell^{f_1(c),sd} \leq 0$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V^{\mathrm{NFV}} \setminus \{v_s\} \quad (13)$$

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} a_{v1}^{\hat{\gamma}} z_{\hat{\gamma}} + \sum_{\ell \in \omega^+(v)} y_\ell^{f_1(c),sd} - \sum_{\ell \in \omega^-(v)} y_\ell^{f_1(c),sd} = 0$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V^{\mathrm{NFV}} \setminus \{v_s\} \quad (14)$$

$$\sum_{\ell \in \omega^+(v)} y_\ell^{f_1(c),sd} - \sum_{\ell \in \omega^-(v)} y_\ell^{f_1(c),sd} = 0$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V \setminus (V^{\mathrm{NFV}} \cup \{v_s\}) \quad (15)$$

We assume that an unique route exists from v_s to first VNF location. This is imposed by selecting exactly one outgoing link from v_s unless first VNF is located at v_s. We account for these scenarios using Eq. (12). To find the route from v_s to first VNF, flow conservation needs to be enforced at the intermediate nodes which may or may not have NFV support. Eqs. (14) and (15) enforce flow-conservation constraints at nodes with and without NFV support, respectively.

Route from last function location to v_d:

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} a_{v_d,n_c}^{\gamma} z_{\hat{\gamma}} + \sum_{\ell \in \omega^-(v_d)} y_\ell^{f_{n_c}(c),sd} = 1$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0 \quad (16)$$

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} a_{v,n_c}^{\hat{\gamma}} z_{\hat{\gamma}} - \sum_{\ell \in \omega^+(v)} y_\ell^{f_{n_c}(c),sd} \leq 0$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V^{\mathrm{NFV}} \setminus \{v_d\} \quad (17)$$

$$\sum_{\hat{\gamma} \in \hat{\Gamma}_c} \delta_{sd}^{\hat{\gamma}} a_{v,n_c}^{\hat{\gamma}} z_{\hat{\gamma}} - \sum_{\ell \in \omega^+(v)} y_\ell^{f_{n_c}(c),sd} + \sum_{\ell \in \omega^-(v)} y_\ell^{f_{n_c}(c),sd} = 0$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V^{\mathrm{NFV}} \setminus \{v_d\} \quad (18)$$

$$\sum_{\ell \in \omega^+(v)} y_\ell^{f_{n_c}(c),sd} - \sum_{\ell \in \omega^-(v)} y_\ell^{f_{n_c}(c),sd} = 0$$
$$c \in C, (v_s,v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V \setminus (V^{\mathrm{NFV}} \cup \{v_d\}) \quad (19)$$

Eq. (16) selects one incoming link to v_d to ensure a route to v_d. For cases where last VNF is placed at destination node, we use Eq. (17). Eqs. (18) and (19) enforce flow conservation at nodes with and without NFV support, respectively.

V. COLUMN GENERATION - ILP

Pre-computing all configurations becomes computationally expensive for large networks. As the number of configurations grows with network size, the problem fits naturally in the column-generation framework [15].

Column generation (**CG**) is a decomposition technique, where the problem (called Master Problem-**MP**) to be solved is divided into two sub-problems: restricted master problem (**RMP**) (selection of the best configurations) and pricing problems (**PP_SC**$(c))_{c \in C}$ (configuration generators for each chain). CG process involves solving **RMP**, querying the dual values of **RMP** constraints, and using them for **PP_SC**(c) objective. Each improving solution (i.e., with a negative reduced cost) of **PP_SC**(c) is added to **RMP**, and previous step is repeated until optimality condition is reached ([15], [16]), with **PP_SC**(c) explored in a round-robin fashion.

The advantage here is that we do not have to precompute configurations. CG generates a column (here, a configuration) by itself, adds them to **RMP** and solves **RMP**. This set of steps is repeated until reduced cost becomes non-negative (RED_COST ≥ 0). We convert the final **RMP** to an ILP and solve to get integer solution. **RMP** selects the best $\gamma \in \Gamma_c$ for each SC c. Also it finds a route from v_s (source) to first VNF of c and from last VNF of c to v_d (destination).

An illustration of the constraint splitting between **RMP** and **PP_SC**(c) is depicted in Fig. 2. Nodes circled in purple are NFV nodes, yellow nodes do not host VNFs at present but have NFV support, and orange nodes currently host VNFs. Figure 2(a) has f_1 located at v_1. When a different configuration is selected in Fig. 2(b) and f_1 is located at v_2, then **RMP** finds the path from v_s to location of f_1. Similarly, **RMP** finds the path from last VNF to v_d, i.e., f_5 to v_d here.

A. Reduced Master Problem (RMP)

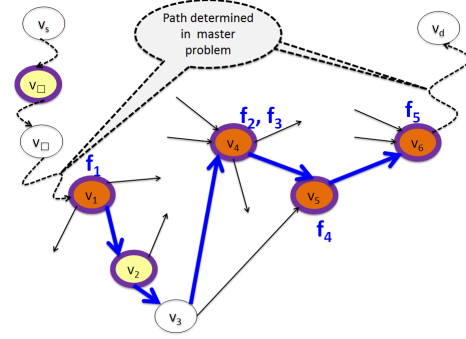**Objective:** Minimize bandwidth consumed:

$$\min \sum_{c \in C} \sum_{\hat{\gamma} \in \hat{\Gamma}_c} \underbrace{\left(\sum_{(s,d) \in \mathcal{SD}} D_{sd}^c \right) \left(\sum_{\ell \in L} \sum_{i \in I} \underbrace{\delta_{sd}^{\hat{\gamma}} b_{i\ell}^{\hat{\gamma}}}_{y_{i\ell}^{\hat{\gamma},sd}} \right)}_{\text{COST}_{\hat{\gamma}}} z_{\hat{\gamma}} +$$
$$\sum_{c \in C} \sum_{\ell \in L} \sum_{(s,d) \in \mathcal{SD}} D_{sd}^c \left(y_{\ell}^{f_1(c),sd} + y_{\ell}^{f_{n_c}(c),sd} \right) \quad (20)$$
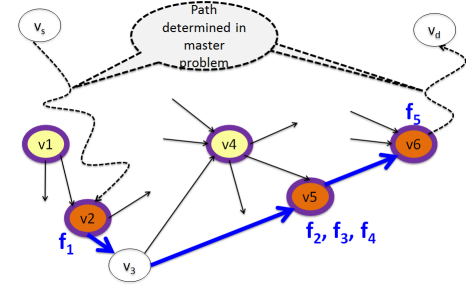
The formulation for the reduced master problem is same as the ILP in Section IV. However, the **RMP** is solved as a Linear Program (LP) for the duration of the CG. After the CG solves the RMP optimally (when RED_COST ≥ 0), we solve the final RMP as an ILP to get integer solution.

B. Pricing Problem: PP(c)

PP_SC(c) generates configurations. Here, we discover that the configuration $(\hat{\gamma})$ structure results in **quadratic** constraints since we have to also determine which traffic flows will make up the configuration. **Quadratic** expressions can be seen in Eqs. (21), (22), and (23). We **linearize** these constraints using

(a) A first configuration (γ_1) for c

(b) A second configuration (γ_2) for c

Fig. 2: Two configuration examples for chain $c = (f_1 \prec f_2 \prec f_3 \prec f_4 \prec f_5)$.

Eqs. (25) and (26), however, the performance of the CG is still affected.

Objective: Minimize reduced cost of variable $z_{\hat{\gamma}}$ (after linearization):

$$[\textbf{PP\_SC}(c)] \qquad \text{RED\_COST}_{\hat{\gamma}} = \text{COST}_{\hat{\gamma}} + u^{(3)}$$
$$+ \sum_{v \in V^{\text{NFV}}} \sum_{f \in F} \sum_{i=1}^{n_c} u_{fv}^{(4)} \text{T}_{fi}^c a_{vi} - \sum_{v \in V^{\text{NFV}}} \sum_{f \in F} \sum_{i=1}^{n_c} u_{fv}^{(5)} \text{T}_{fi}^c a_{vi}$$
$$+ \sum_{v \in V^{\text{NFV}}} u_v^{(9)} \sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c \sum_{i=1}^{n_c} \text{T}_{fi}^c n_f^{\text{CORE}} p_{v,i}^{sd}$$
$$+ \sum_{\ell \in L} \sum_{(v_s,v_d) \in \mathcal{SD}} u_\ell^{(10)} D_{sd}^c \sum_{i=1}^{n_c-1} q_{i\ell}^{sd} - \sum_{(v_s,v_d) \in \mathcal{SD}} u_{sd}^{(11)} \delta_{sd}$$
$$- \sum_{(v_s,v_d) \in \mathcal{SD}} u_{sd}^{(12)} p_{v_s,1}^{sd} + \sum_{(v_s,v_d) \in \mathcal{SD}} \sum_{v \in V^{\text{NFV}} \setminus \{v_s\}} u_{sd,v}^{(13)} p_{v,1}^{sd}$$
$$- \sum_{(v_s,v_d) \in \mathcal{SD}} \sum_{v \in V^{\text{NFV}} \setminus \{v_s\}} u_{sd,v}^{(14)} p_{v,1}^{sd}$$
$$- \sum_{(v_s,v_d) \in \mathcal{SD}} u_{sd}^{(16)} p_{v_d,n_c}^{sd} + \sum_{(v_s,v_d) \in \mathcal{SD}} \sum_{v \in V^{\text{NFV}} \setminus \{v_s\}} u_{sd,v}^{(17)} p_{v,n_c}^{sd}$$
$$- \sum_{(v_s,v_d) \in \mathcal{SD}} \sum_{v \in V^{\text{NFV}} \setminus \{v_s\}} u_{sd,v}^{(18)} p_{v,n_c}^{sd} \quad (21)$$

where $u^{(3)}$, $u_{fv}^{(4)}$, $u_{fv}^{(5)}$, $u_v^{(9)}$, $u_\ell^{(10)}$, $u_\ell^{(11)}$, $u_{sd}^{(12)}$, $u_{sd,v}^{(13)}$, $u_{sd,v}^{(14)}$, $u_{sd}^{(16)}$, $u_{sd,v}^{(17)}$ and $u_{sd,v}^{(18)}$ are dual variables associated with Eqs. (3), (4), (5), (9), (10), (11), (12), (13), (14), (16), (17) and (18) respectively.

Variables:

- $\delta_{sd} = 1$ if configuration $\hat{\gamma}$ to be generated contains node pair (v_s, v_d) requiring c; 0 otherwise.
- $p_{v,i}^{sd} = \delta_{sd}\, a_{vi} = 1$ if node pair (v_s, v_d) is provisioned using the provisioning of c / placement function (ith function of SFC c in location v) of the configuration under construction; 0 otherwise
- $q_{i\ell}^{sd} = \delta_{sd}\, b_{i\ell} = 1$ if node pair (v_s, v_d) is provisioned using the provisioning of c (with link ℓ being used in the path from the location of the ith function to the location of the $(i+1)$th function) / placement function of the configuration under construction; 0 otherwise

Constraints:

$$\sum_{(v_s,v_d)\in\mathcal{SD}} D_{sd}^c \sum_{i=1}^{n_c} n_f^{\text{CORE}} \mathbf{T}_{fi}^c \underbrace{\delta_{sd}\, a_{vi}}_{p_{v,i}^{sd}} \leq \mathbf{N}^{\text{CORE}}$$
$$v \in V^{\text{NFV}} \qquad (22)$$

$$\sum_{(v_s,v_d)\in\mathcal{SD}} D_{sd}^c \sum_{i=1}^{n_c-1} \underbrace{\delta_{sd} b_{i\ell}}_{q_{i\ell}^{sd}} \leq \text{CAP}_\ell \qquad \ell \in L \qquad (23)$$

Eq. (22) enforces a capacity constraints in CPU cores on all NFV nodes while Eq. (23) imposes link capacity.

$$\sum_{v\in V^{\text{NFV}}} a_{vi} = 1 \qquad\qquad i = 1, 2, \ldots, n_c \qquad (24)$$

$$p_{v,i}^{sd} = a_{vi} \wedge \delta_{sd} \qquad (v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$v \in V^{\text{NFV}}, i = 1, 2, \ldots, n_c \qquad (25)$$

$$q_{i\ell}^{sd} = b_{i\ell} \wedge \delta_{sd} \qquad (v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0,$$
$$\ell \in L, i = 1, 2, \ldots, n_c - 1 \qquad (26)$$

$$\sum_{\ell\in\omega^-(v)} b_{1,\ell} \leq 1 - a_{v,1} \qquad v \in V^{\text{NFV}} \qquad (27)$$

$$\sum_{\ell\in\omega^+(v)} b_{n_c-1,\ell} \leq 1 - a_{v,n_c} \qquad v \in V^{\text{NFV}} \qquad (28)$$

$$\sum_{\ell\in\omega^+(v)} b_{i\ell} - \sum_{\ell\in\omega^-(v)} b_{i\ell} = a_{vi} - a_{v,i+1}$$
$$v \in V^{\text{NFV}}, i = 1, 2, \ldots, n_c - 1 \qquad (29)$$

$$\sum_{\ell\in\omega^+(v)} b_{i\ell} - \sum_{\ell\in\omega^-(v)} b_{i\ell} = 0$$
$$v \in V \setminus V^{\text{NFV}}, i = 1, 2, \ldots, n_c - 1 \qquad (30)$$

Eq. (24) ensures that each VNF in SC c is placed exactly once. Eqs. (25)[2] and (26) introduce the variables to linearize the model. Eq. (27) ensures that, if $f_1(c)$ is located in v, there is no flow b that is incoming to v. Eqs. (29) and (30) are flow-conservation constraints: Eq. (29) for nodes with NFV support

---

[2] Linearization details:

$$\forall (v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0, \quad \forall v \in V^{\text{NFV}}, \quad i = 1, 2, \ldots, n_c$$

Eq. (25) can be linearly represented as below.

$$p_{v,i}^{sd} \leq a_{vi} \qquad (31)$$

$$p_{v,i}^{sd} \leq \delta_{sd} \qquad (32)$$

$$a_{vi} + \delta_{sd} - 1 \leq p_{v,i}^{sd} \qquad (33)$$

Eq. (26) can also be similarly represented.

and Eq. (30) for other nodes. Eq. (28) ensures that, if $f_{n_c}(c)$ is located in v, there is no flow b that is outgoing v.

C. Solution Scheme

The **PP_SC**(c) are solved in a round-robin fashion, and the final **RMP** is solved as an ILP, as in [15], [16].

VI. TWO-PHASE MODEL

As already mentioned, we are solving this problem considering that each SC request chooses to map to one of multiple instances, which leads the model discussed in Section V to have quadratic constraints, reducing the scalability of the model. So, to avoid quadratic constraints, we develop a new solution approach consisting of two phases:

- Phase 1: We fix the number N_c of instances accepted per SC (N_c can go from 1 up to the number of demands for that SC), and then we group the traffic requests in N_c groups of requests. All the requests in a group are forced to use the same SC instance (Section VI-A). Then we pass the N_c instances as distinct SCs to the next phase.
- Phase 2: We solve the SC mapping problem with one single instance per SC based on the inputs of Phase 1. The solution of this simplified (linear, yet still very complex) problem is based on a column-generation-based decomposition model (Section VI-B).

As a result of Phase 1, we no longer have to account for traffic flows as part of a configuration. This happens because we partition the traffic flows in Phase 1, and so it becomes much easier to find the best possible configuration for each partition in the second phase. For the two-phase-model, the configuration is γ. A chain configuration γ in the two-phase model is characterized by the following parameters:

- Location of functions: $a_{vi}^\gamma = 1$ if ith function $f_i \in c$ is located in v in configuration; 0 otherwise.
- Connectivity of locations: path from location of current VNF to next VNF in SC c. If link ℓ is used in the path from location of $f_{\sigma_i(c)}$ to location of $f_{\sigma_{i+1}(c)}$, then $b_{i\ell}^\gamma = 1$; 0 otherwise.

A. Phase 1: Shortest-Path Traffic Grouping (SPTG) Heuristic

Now, we propose a Shortest-Path Traffic Grouping (SPTG) heuristic, which forms N_c groups of node pairs for each SC (given by SD_c), to be given as input to the decomposition model in Section V that will treat them as distinct SC and decide the best SC mapping for each of the N_c node-pair groups. As a result, we will have a solution mapping multiple SC instances per SC.

The logic of the SPTG algorithm is that groups are formed among node pairs that share links along their shortest path(s). SPTG is designed to make the largest flows take shortest paths, the intuition being that, if largest flows take shortest paths, network resource consumption will be reduced. Details of SPTG approach can be found in Algorithm 1.

If Algorithm 1 terminates with $SD_c^{\text{LEFT}} = \emptyset$ and a number of groups that is $< N_c$, partition some of the groups in order to reach N_c groups.

Algorithm 1 SPTG(c)

Require: G, SD_c, N_c

Ensure: PARTITION $\leftarrow$ partition of node pairs (v_s, v_d) into groups

1: PARTITION $\leftarrow \emptyset$
2: $numberOfGroups \leftarrow 0$
3: $SD_c^{\text{LEFT}} \leftarrow SD_c$ ▷ list of (v_s, v_d) for c
4: $bigFlow \leftarrow largestFlow(SD_c^{\text{LEFT}})$ ▷ selects largest flow in SD_c^{LEFT}
5: **while** $numberOfGroups < N_c$ & $SD_c^{\text{LEFT}} \neq \emptyset$ **do**
6:      **for** (v_s, v_d) in G **do**
7:          CLUSTER$_{sd}$ $\leftarrow$ set of traffic pairs whose shortest path passes through (v_s, v_d)
8:      **end for**
9:      $largestCluster \leftarrow \max\limits_{(v_s,v_d):D_{sd}^c>0}$ CLUSTER$_{sd}$ & $bigFlow \in$ CLUSTER$_{sd}$
10:      $SD_c^{\text{LEFT}} \leftarrow SD_c^{\text{LEFT}} \setminus largestCluster$ ▷ remove traffic pairs of $largestCluster$ from SD_c^{LEFT}
11:      Add $largestCluster$ to PARTITION
12:      $numberOfGroups \leftarrow numberOfGroups + 1$
13:      $bigFlow \leftarrow largestFLow(SD_c^{\text{LEFT}})$
14: **end while**
15: **if** $SD_c^{\text{LEFT}} \neq \emptyset$ **then**
16:      **for** $trafficPair \in SD_c^{\text{LEFT}}$ **do**
17:          add $trafficPair$ to GROUP $\in$ PARTITION, such that the (v_s, v_d) associated with GROUP provides the shortest path for provisioning $trafficPair$
18:      **end for**
19: **end if**

B. Phase 2: Column-Generation Approach

Since our definition of configurations (γ) has been simplified, CG becomes linear and faster.

*1) Restricted Master Problem (**RMP**) :* **Variables:**

- $z_\gamma = 1$ if configuration γ is selected; 0 otherwise.
- $x_v^{ci} = 1$ if ith function of c is located in v; 0 otherwise.
- $y_\ell^{\text{first}(c),sd} = 1$ if ℓ is on path from v_s to location of first VNF in c; 0 otherwise.
- $y_\ell^{\text{last}(c),sd} = 1$ if ℓ is on path from location of last VNF in c to v_d; 0 otherwise.
- $h_v = 1$ if v is used as a location for a VNF; 0 otherwise.

Objective: Minimize bandwidth consumed:

$$\min \sum_{\gamma \in \Gamma} \underbrace{\overbrace{\left(\sum_{(s,d) \in \mathcal{SD}} D_{sd}^c\right)}^{\text{Overall traffic using } c} \overbrace{\left(\sum_{\ell \in L} \sum_{i \in I} b_{i\ell}^\gamma\right)}^{\substack{\text{Number of links} \\ \text{in the route of } c}}}_{\text{COST}_\gamma} z_\gamma +$$

$$\sum_{c \in C} \sum_{\ell \in L} \sum_{(s,d) \in \mathcal{SD}} D_{sd}^c \left(y_\ell^{f_1(c),sd} + y_\ell^{f_{n_c}(c),sd}\right) \quad (34)$$

Total bandwidth consumed in placing multiple SCs depends on configuration γ selected for each SC c. Each γ for c locates VNFs of c and gives the route to traverse these VNF

locations. So, bandwidth consumed when going from v_s to v_d and traversing the SC depends on selected γ.

Constraints:

$$\sum_{\gamma \in \Gamma_c} z_\gamma = 1 \qquad\qquad c \in C \quad (35)$$

$$\sum_{c \in C} \sum_{\gamma \in \Gamma_c} \sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c \left(\sum_{i=1}^{n_c} a_{vi}^\gamma \delta_{fi}^c n_f^{\text{CORE}}\right) z_\gamma \leq n^{\text{CORE}}$$
$$v \in V^{\text{NFV}} \quad (36)$$

$$\sum_{c \in C} \sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c$$
$$\left(y_\ell^{f_1(c),sd} + y_\ell^{f_{n_c}(c),sd} + \sum_{\gamma \in \Gamma_c} \sum_{i=1}^{n_c-1} b_{i\ell}^\gamma z_\gamma\right)$$
$$\leq \text{CAP}_\ell \qquad\qquad \ell \in L \quad (37)$$

$$\sum_{\gamma \in \Gamma_c} a_{vi}^\gamma z_\gamma = x_v^{ci} \qquad f_i \in F(c), c \in C, v \in V^{\text{NFV}} \quad (38)$$

$$M x_{vf} \geq \sum_{c \in C: f \in c} \sum_{i \in \{1,2,...,n_c\}: f_i=f} x_v^{ci} \geq x_{vf}$$
$$v \in V^{\text{NFV}}, f_i \in F \quad (39)$$

$$M h_v \geq \sum_{f \in F} x_{vf} \geq h_v \qquad v \in V^{\text{NFV}} \quad (40)$$

$$\sum_{v \in V^{\text{NFV}}} h_v \leq K \quad (41)$$

Constraints (35) guarantee that we select exactly one γ for SC c and force c to have a single instance. Each γ is associated with a set of a_{vi}^γ (from **PP_SC**(c)) required to be consistent with x_v^{ci} in **RMP**, which is resolved by Eqs. (38).

Constraints (36) ensure that each NFV node has a sufficient number of CPU cores for hosting f. Eq. (37) enforces link-capacity constraints for the complete route for SC c from v_s to v_d for all $(v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0$).

Eq. (39) keeps track of VNF replicas. Eq. (40) keeps track of NFV nodes used for hosting VNFs while Eq. (41) enforces the number of NFV nodes allowed to host VNFs.

Route from v_s to first function location:

$$\sum_{\ell \in \omega^+(v_s)} y_\ell^{f_1(c),sd} = 1 - x_{v_s}^{c,1} \qquad c \in C,$$
$$(v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0 \quad (42)$$

$$\sum_{\ell \in \omega^-(v)} y_\ell^{f_1(c),sd} \geq x_v^{c,1} \qquad c \in C,$$
$$(v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0, v \in V^{\text{NFV}} \setminus \{v_s\} \quad (43)$$

$$\sum_{\ell \in \omega^+(v)} y_\ell^{f_1(c),sd} - \sum_{\ell \in \omega^-(v)} y_\ell^{f_1(c),sd} = -x_v^{c,1}$$
$$c \in C, (v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0, v \in V^{\text{NFV}} \setminus \{v_s\} \quad (44)$$

$$\sum_{\ell \in \omega^+(v)} y_\ell^{f_1(c),sd} - \sum_{\ell \in \omega^-(v)} y_\ell^{f_1(c),sd} = 0$$
$$c \in C, (v_s, v_d) \in \mathcal{SD} : D_{sd}^c > 0, v \in V \setminus (V^{\text{NFV}} \cup \{v_s\}) \quad (45)$$

We assume that an unique route exists from v_s to first VNF location. This is imposed by selecting exactly one outgoing

link from v_s unless first VNF is located at v_s. We account for these scenarios using Eq. (42). To find the route from v_s to first VNF, flow conservation needs to be enforced at the intermediate nodes which may or may not have NFV support. Eqs. (44) and (45) enforces flow-conservation constraints at nodes with and without NFV support, respectively.

We can enforce same functionality as Eqs. (42), (44), (45), and (43), on route from location of last VNF to v_d. For the interested reader, similar details are provided in [17].

2) Pricing Problem: Mapping configurations for each SC c ($c \in C$) corresponds to the solution of pricing problems. The number of pricing problems to be solved equals the sum of the number of SC instances to be deployed.

Pricing problem **PP_SC**(c) generates: *(i)* A set of locations for VNFs of c; and *(ii)* a sequence of paths from the location of VNF f_i to the location of VNF f_{i+1}, for $i = 1, 2, \ldots, n_c-1$ for chain c. Each solution that is generated by **PP_SC**(c) with a negative reduced cost leads to a new potential γ for c of interest. Please see [17] for further details.

Let $u_c^{(35)} \lesseqgtr 0, u_v^{(36)} \geq 0$, and, $u_{vf}^{(38)} \geq 0$ be values of dual variables associated with constraints (35), (36), (38), respectively.

Variables:

- $a_{vi} = 1$ if ith function f_i of c is located in $v \in V^{\text{NFV}}$; 0 otherwise.
- $b_{i\ell} = 1$ if ℓ is on the path from location of f_i to location of f_{i+1}; 0 otherwise.

Objective: Minimize reduced cost of variable z_γ:

$$
\begin{aligned}
\text{[\textbf{PP\_SC}}(c)\text{]} \quad &\text{RED\_COST}_\gamma = \text{COST}_\gamma - u^{(35)} \\
&+ \sum_{v \in V^{\text{NFV}}} u_v^{(36)} \sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c \sum_{i=1}^{n_c} n_{f_i}^{\text{CORE}} a_{vi} \\
&+ \sum_{\ell \in L} \sum_{(v_s,v_d) \in \mathcal{SD}} u_\ell^{(37)} D_{sd}^c \sum_{i=1}^{n_c-1} b_{i\ell} - \sum_{i=1}^{n_c} \sum_{v \in V^{\text{NFV}}} u_{vci}^{(38)} a_{vi}.
\end{aligned}
$$
(46)

where RED_COST value indicates whether an optimal γ for c has been found. A non-negative value of RED_COST indicates optimality for our model.

Constraints:

$$
\sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c \sum_{i=1}^{n_c} n_f^{\text{CORE}} \delta_{f_i}^c a_{vi} \leq n^{\text{CORE}}
$$
$$
v \in V^{\text{NFV}} \quad (47)
$$

$$
\sum_{(v_s,v_d) \in \mathcal{SD}} D_{sd}^c \sum_{i=1}^{n_c-1} b_{i\ell} \leq \text{CAP}_\ell \qquad \ell \in L \quad (48)
$$

Eqs. (47) and (48) are compute resource and capacity constraints, similar to those in **RMP** and are linear. The rest of the equations are the same as Eqs. (24) to (30) in Section V-B and perform the same function.

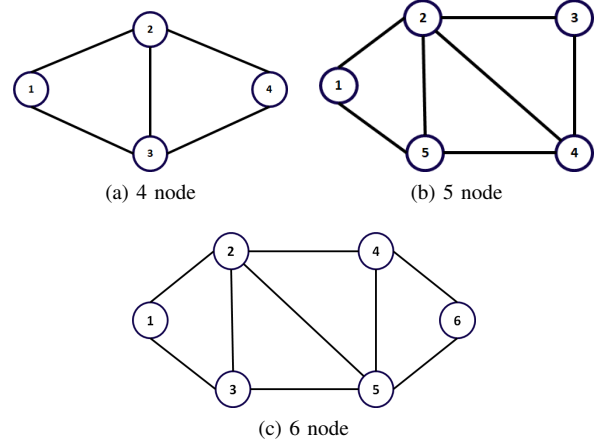*3) Solution Scheme:* Same as in Section V-C.

(a) 4 node   (b) 5 node

(c) 6 node

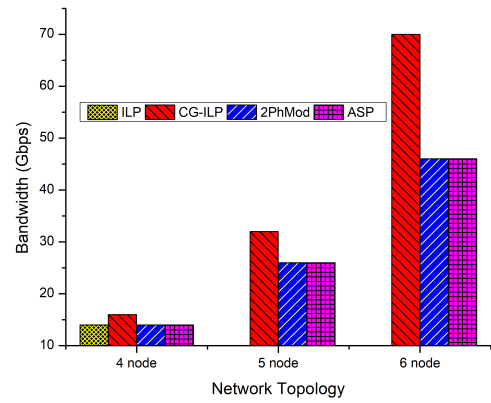Fig. 3: Network topologies.

Fig. 4: Comparison of bandwidth used.

VII. COMPARISON OF SOLUTION APPROACHES

To benchmark our solution approaches ILP, CG-ILP, and Two-Phase model, we use the All Shortest Path (ASP) calculation. ASP assumes that, in the best possible scenario, all traffic flows requiring a SC c will have a SC instance deployed on their shortest path. Thus, total bandwidth used will be equal to all traffic flows taking a shortest path.

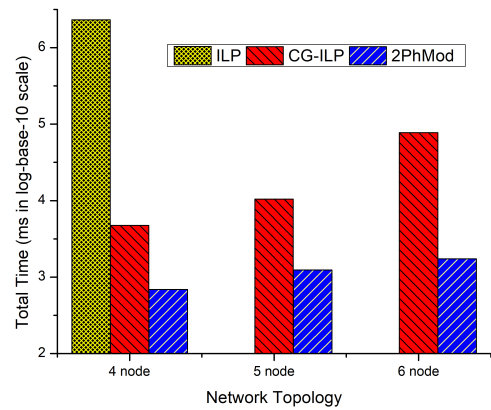Fig. 4 compares bandwidth consumption of our three ap-
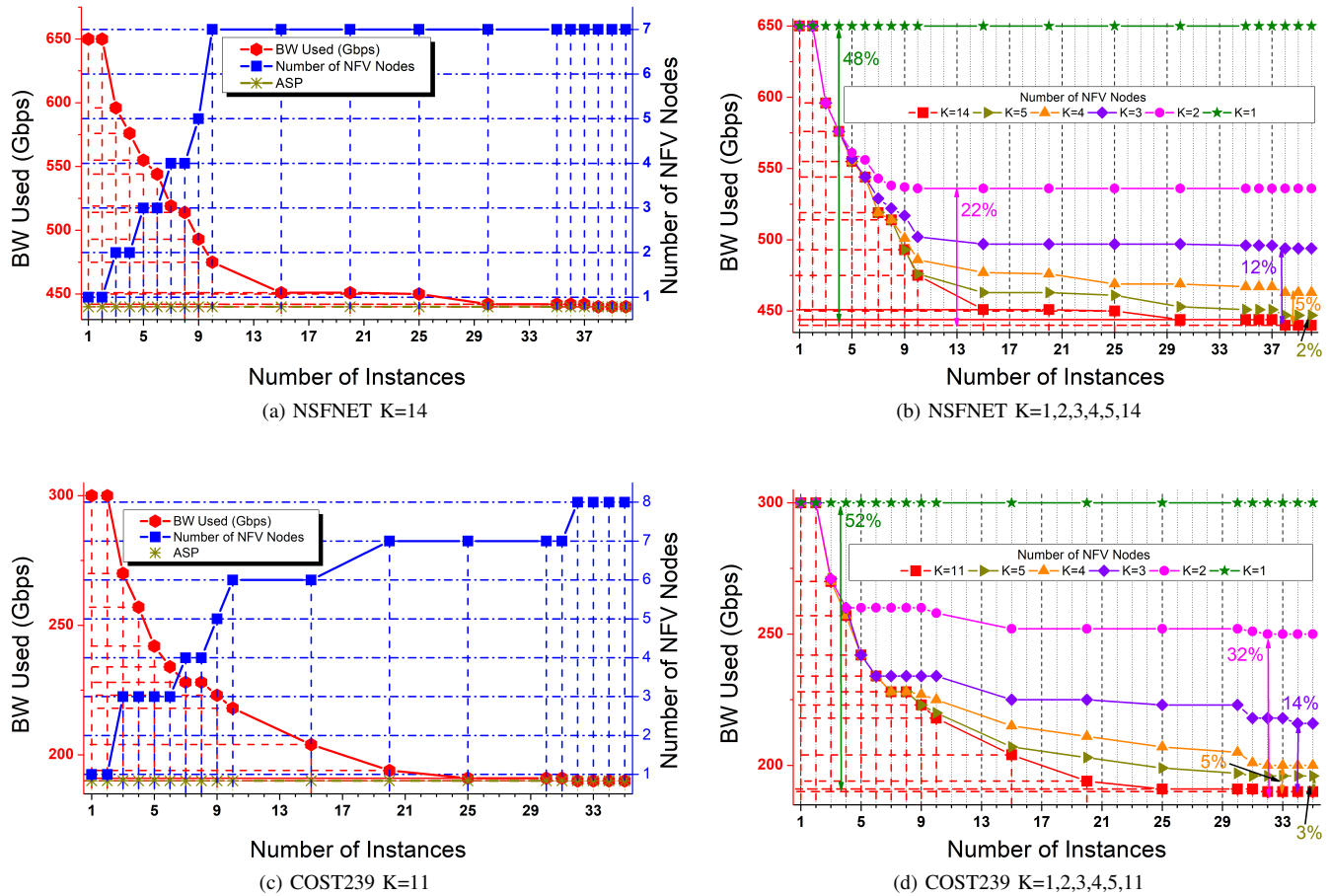

Fig. 5: Comparison of total time.

(a) NSFNET K=14

(b) NSFNET K=1,2,3,4,5,14

(c) COST239 K=11

(d) COST239 K=1,2,3,4,5,11

Fig. 6: Bandwidth vs. number of NFV nodes in NSFNET and COST239 networks.

Service Chain	Chained VNFs	%traffic
Web Service	NAT-FW-TM-WOC-IDPS	18.2%
VoIP	NAT-FW-TM-FW-NAT	11.8%
Video Streaming	NAT-FW-TM-VOC-IDPS	69.8%
Online Gaming	NAT-FW-VOC-WOC-IDPS	0.2%

TABLE I: Service Chain Requirements [8]; Network Address Translator (NAT), Firewall (FW), Traffic Shaper (TM), WAN Optimization Controller (WOC), Intrusion Detection and Prevention System (IDPS), Video Optimization Controller (VOC).

proaches across three network topologies shown in Fig. 3 for a single SC c deployment. We consider all-to-all traffic flows in each network and allow I_c instances for all solution approaches. ILP is shown to be as good as ASP for four-node networks; however, it does not scale for larger networks because of pre-computation of all possible configurations. CG-ILP does not provide optimal solutions because of ε-optimality gap (difference between ILP and LP values). Two-Phase model performs as well as ASP for all topologies.

Fig. 5 shows total time taken by various approaches. Note that Two-Phase Model scales best across all topologies.

VIII. ILLUSTRATIVE NUMERICAL EXAMPLES

A. Single Service Chain Scenario

We first tested our two-phase optimization process on a 14-node NSFNET WAN topology [17] with a complete traffic matrix, i.e., with traffic flows between all node pairs, assuming all nodes can be made NFV nodes. The link capacities are sufficient to support all flows. Each traffic flow is 1 Gbps and demands the same 5 VNF service chain (SC) for video streaming, as shown in Table I. Compute resource (CPU) at each node is sufficient to support traffic demand, which helps in determining the optimal location to deploy CPU cores and number of CPU cores at each location. The second run of the model is on an 11-node COST239 WAN topology [18] under the same specifications as above.

Figure 6(a) shows the bandwidth consumption as the number of SC instances increases. Here, we allow all nodes (K=14) to host VNFs. We find that, as number of deployed SC instances increases, bandwidth consumption decreases. With a higher number of instances, more groups of traffic node pairs are able to take short paths. We see that, at 38 instances, we achieve minimum possible bandwidth consumption, meaning all traffic flows are taking the shortest path. Note that number of traffic node pairs in the network is 182, requiring apriori upto 182 different instances (solving the problem for 182 instances would be equivalent to obtaining a solution with

existing models as in [5][6][7]). Instead, our approach, with only 38 instances, achieves minimum bandwidth consumption. This is important as an operator may deploy multiple SCs and manage multiple instances per SC including routing flows to a particular SC instance. So, a lower number of instances will lower the orchestration overhead for the operator.
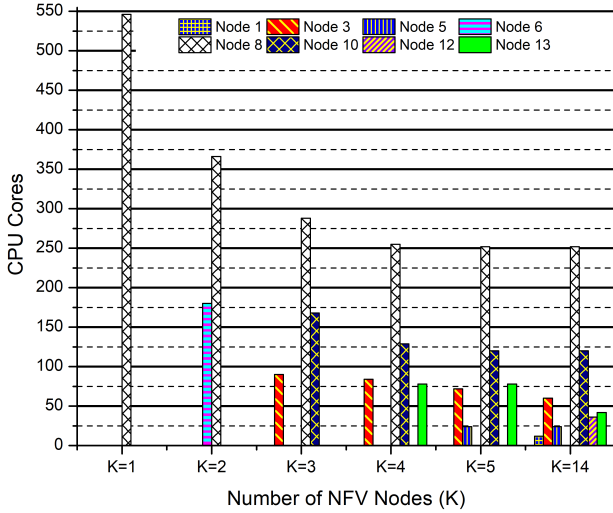

Fig. 7: CPU core distribution across K (NSFNET).

On the other hand, number of NFV nodes increases as number of SC instances increases. Indeed, as SC mappings become more distributed, more nodes are being used for hosting virtual functions. In Fig. 6(a), 11 nodes are NFV enabled for 38 different SC mappings. For a network operator, capital expenditure in making 11 out of 14 nodes capable of hosting VNFs is very high. So, operators may want to minimize the number of NFV nodes while also trying to reduce bandwidth consumption by using multiple SC mapping instances. This led us to explore how the bandwidth consumption varies when the numbers of NFV nodes are limited.
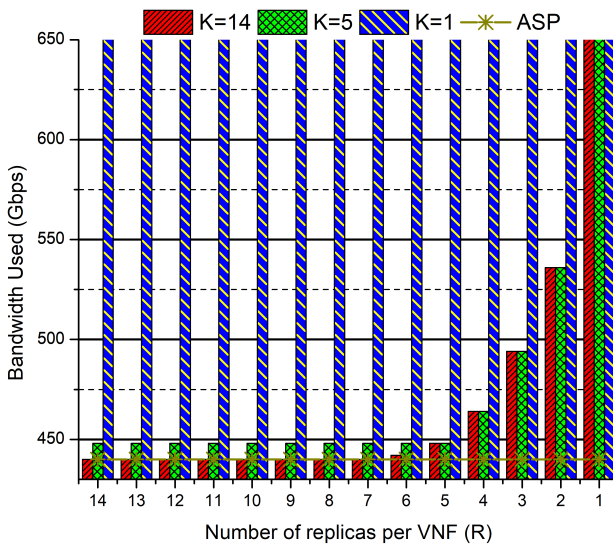

Fig. 8: Bandwidth used across R (NSFNET).

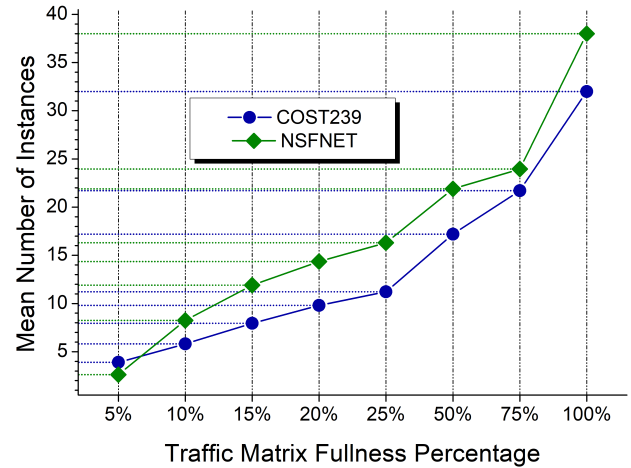Figure 6(b) shows bandwidth consumption for SC mapping

Fig. 9: Mean number of SC instances across varying number of traffic flows.

instances for various K values. When $K = 1$, all traffic flows have to traverse the one node in the network; hence, number of instances does not affect bandwidth consumption. At $K = 2$, deploying more than 10 instances does not improve bandwidth utilization. For $K = 3$ and 35 instances, we are able to achieve close to 10% of the minimum bandwidth utilization. Similarly, at $K = 4$, we reach within 5% of the optimal bandwidth consumption. Bandwidth consumption comes to within 2% of the optimal when K=5 and 38 instances. Thus, we can achieve near-optimal bandwidth consumption by a using a small number of instances and nodes.

Figures 6(c) and 6(d) corroborate our findings in Figs. 6(a) and 6(b) over COST239 network.
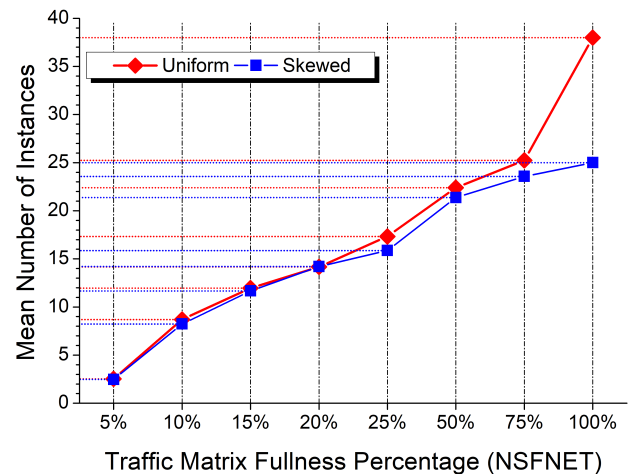

Fig. 10: Mean number of SC instances for uniform and skewed traffic.

Figure 7 shows amount of compute resources (CPU cores) and their location for different values of K, given that we know optimal number of SC instances required. CPU cores used by each VNF depends on VNF type and throughput required as shown in Table II. At $K = 1$, node 8 will be selected for deploying CPU cores. When $K = 2$, the best location for deploying CPU resources are nodes 8 and 6. Note that the
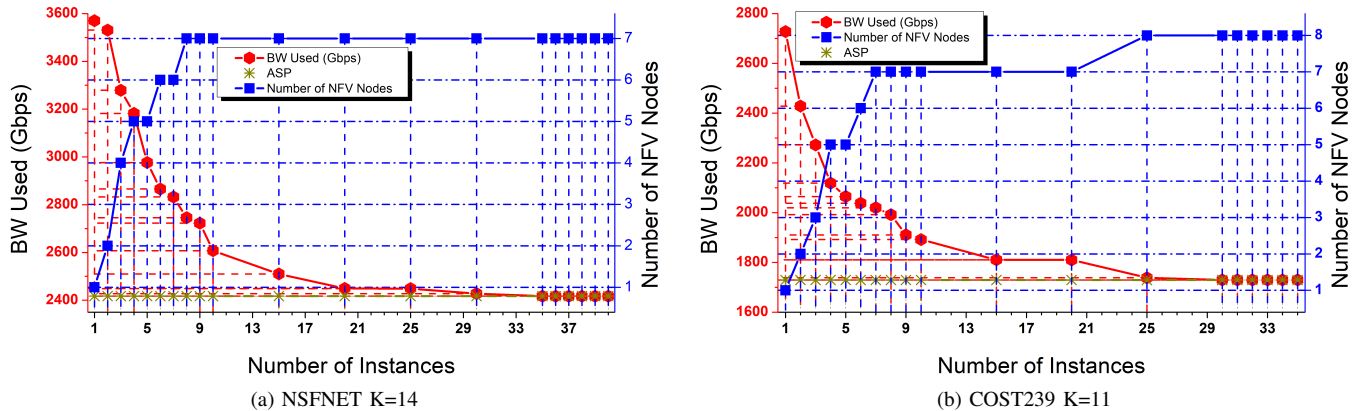
(a) NSFNET K=14

(b) COST239 K=11

Fig. 11: Bandwidth vs. number of NFV Nodes in NSFNET and COST239 when deploying all service chains in Table I.

Application	Throughput		
	1 Gbps	5 Gbps	10 Gbps
NAT	1 CPU	1 CPU	2 CPUs
IPsec VPN	1 CPU	2 CPUs	4 CPUs
Traffic Shaper	1 CPU	8 CPUs	16 CPUs

TABLE II: VNF requirements as per throughput [19].

traffic remains the same across $K = 1$ and $K = 2$, so the total amount of CPU cores used remain the same. However, when $K = 2$, two nodes get selected as it reduces the total bandwidth consumption, and the distribution of CPU cores happens across nodes. We find that more nodes are selected for $K = 3, 4, 5, 14$, and the compute resources become more distributed. At K=14, we find that only 7 nodes are used to host CPU cores, which means we need to have at max 7 NFV nodes to achieve ASP bandwidth consumption.

In the above results, we determine the number of SC instances required for each K to get minimum bandwidth consumption. We define this count of SC instances to be optimal. Now, given this optimal number of SC instances, we want to observe the effect different number of replicas of VNFs (R) has on different K values. Here, $R = 14$ means all VNFs in the SC are allowed 14 replicas. Fig. 8 compares bandwidth used in NSFNET (when $K = 1, 5, 14$) when different R are allowed. We find that when $K = 5, R = 5$ our bandwidth consumption is close to ASP, implying that we require a small number of K and R.

Figure 6(a) shows the number of SC instances required to achieve ASP bandwidth consumption when there are traffic flows between all traffic pairs. We call this a 100% traffic matrix fullness, i.e., all entries in the traffic matrix have been filled. Fig. 9 shows the mean number of SC instances required to reach ASP bandwidth consumption across different traffic matrix fullness percentages (percentage of entries that are filled in the traffic matrix) under the same traffic load. We find that the mean number of SC instances required to reach ASP bandwidth consumption increases as the traffic matrix

fullness percentage increases for both COST239 and NSFNET.

All results until now assumed an uniform traffic distribution. However, traffic can be skewed. So, we skew the traffic load based on [20] (skewed based on population size of the nodes) for varying number of traffic flows (traffic matrix fullness percentage) and display the number of SC instances required to achieve ASP bandwidth consumption. We compare $SPTG$ performance for uniform and skewed traffic in Fig. 10. We find that $SPTG$ can achieve ASP bandwidth consumption for skewed traffic distribution for lower number of SC instances, especially as number of traffic flows increase.

B. Multiple Service Chain Scenario

In the previous subsection, we performed simulations where all traffic flows require the same service chain, i.e., all traffic requires the same VNFs. However, when traffic requires different service chains, not all VNFs are required by all traffic requests and the conclusions on a single SC may not hold. Hence, it becomes important to analyze the effect of varying the number of allowed VNF replicas (R) to focus on the role of each VNF on bandwidth consumption separately. In this subsection, we jointly deploy the four service chains in Table I for a total traffic load of 1 Tbps. The distribution of traffic across service chains follows realistic relative popularity of the four services (see last column in Table I). All four service chains are requested by all traffic pairs in the network, i.e., all four service chains have 100% traffic matrix fullness.

Figure 11(a) shows bandwidth consumption as SC instances increase for all SCs deployed in NSFNET. We find that 35 instances for each SC deployed is sufficient to achieve ASP bandwidth consumption. Number of NFV nodes used also does not vary much from previous result of Fig. 6(a).

We then analyze the effect of varying number of VNF replicas (R). Figure 12 shows the bandwidth consumed when $K = 5$ and $R = 1, 2, 3, 4, 5$. We reduce the number of replicas for a specified VNF to R while the remaining VNFs have R=K, i.e., R=5 here, for all the unspecified VNFs. So here, when $R = 1$ for FW, that means the number of replicas allowed for FW is 1 while the other VNFs have replicas equal to K (here, $K = 5$). We always see $R \leq K$ since a VNF can only be
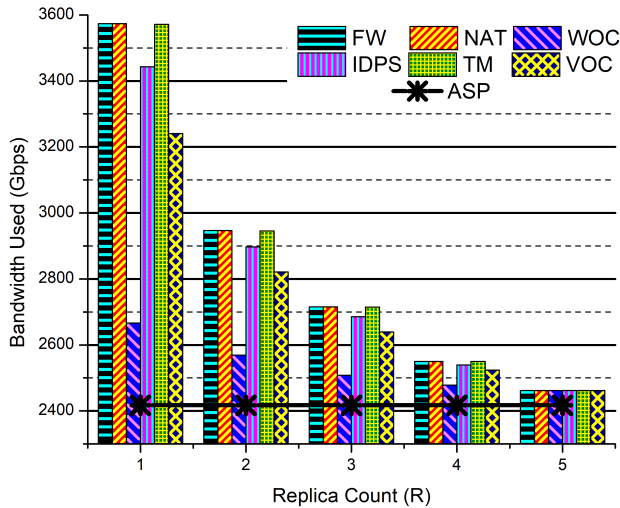
Fig. 12: Bandwidth consumed for varying VNF replica (R) for $K = 5$ (NSFNET).

Network	Nodes	Links	Mean Time (s)
COST239	11	44	12.1
NSFNET	14	40	14.2
EON[21]	16	46	25.5
JAPAN	19	62	225.3
US24[22]	24	86	755.9
GERMANY[9]	50	176	108000

TABLE III: Mean run time across networks (in seconds).

allowed to replicated in K locations at maximum. Note that some VNFs like FW and NAT are shared across all service chains, while others like WOC and VOC are only shared across two service chains. When $R = 1$ for FW and NAT, highest bandwidth consumption is experienced as these VNFs are shared across all four service chains. Conversely, when $R = 1$ for WOC, least bandwidth consumption is experienced since WOC is required by only 18.4% of total traffic. When $R = 2, 3, 4$, bandwidth consumption reduces as R increases, and decrease in bandwidth consumption across VNFs for each R is seen to be dependent on amount of traffic requiring the VNF. This relative deference in bandwidth consumption between VNFs reduces as R increases. This happens as R becomes a less salient parameter for bandwidth consumption as R approaches K. At $R = 5$, bandwidth consumption is the same for all VNFs. This happens as when $R = 5$ for FW, R values for the unspecified VNFs are also 5. Each column when $R = 5$ represents the same situation where all VNFs used are allowed 5 replicas. We find that when $K = 5$, $R = 5$ (when all VNFs have 5 replicas) is sufficient for achieving close to ASP bandwidth consumption.

C. Scalability

Scalability of a solution determines its applicability in real scenarios. So, we show mean run times of our Two-Phase model for networks of different sizes in Table III. Run time is the second phase ($CG + ILP$) execution time. First phase ($SPTG$) execution times were excluded as they were found to be negligible compared to second phase. Note that the Two-Phase model scales well for all networks.

IX. CONCLUSION

We introduce the problem of multiple service chain (SC) mapping with multiple SC instances in presence of highly-populated traffic demands. We developed a Two-Phase model, based on a column-generation model along with a Shortest-Path Traffic Grouping (SPTG) heuristic which results in a scalable linear model, thereby solving this complex problem in a relatively small amount of time. Further, we demonstrate that a near-optimal network resource consumption can be achieved with a relatively small number of SC instances, NFV nodes, and VNF replicas for a 100% populated traffic matrix. This is critical to reduce the network operator's orchestration overhead and capital expenditures.

REFERENCES

[1] ETSI, "Network functions virtualisation: Introductory white paper," portal.etsi.org/NFV/NFV_White_Paper.pdf, 2012.
[2] IETF, "Network service chaining problem statement," https://tools.ietf.org/html/draft-quinn-nsc-problem-statement-00, 2013.
[3] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *IEEE 3rd International Conference on Cloud Networking (CloudNet)*, Oct 2014, pp. 7–13.
[4] H. Moens and F. De Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *10th International Conference on Network and Service Management (CNSM)*, Nov. 2014, pp. 418–423.
[5] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," *https://hal.inria.fr/hal-01170042/*, 2015.
[6] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," in *IFIP/IEEE Intl. Symp. on Int. Netw. Mgmt (IM)*, May 2015, pp. 98–106.
[7] M. Bari, S. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions in NFV," *Computing Research Repository*, vol. abs/1503.06377, 2015. [Online]. Available: http://arxiv.org/abs/1503.06377
[8] M. Savi, M. Tornatore, and G. Verticale, "Impact of processing costs on service chain placement in network functions virtualization," in *IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, 2015, pp. 191–197.
[9] N. Huin, B. Jaumard, and F. Giroire, "Optimization of Network Service Chain Provisioning." [Online]. Available: https://hal.inria.fr/hal-01476018
[10] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On Dynamic Service Function Chain Deployment and Readjustment," *IEEE Transactions on Network and Service Management*, vol. PP, no. 99, 2017.
[11] F. Carpio, W. Bziuk, and A. Jukan, "Replication of Virtual Network Functions: Optimizing Link Utilization and Resource Costs," *Computing Research Repository (CoRR)*, vol. abs/1702.07151, 2017. [Online]. Available: http://arxiv.org/abs/1702.07151
[12] Y. Jia, C. Wu, Z. Li, F. Le, and A. X. Liu, "Online Scaling of NFV Service Chains across Geo-distributed Datacenters," *Computing Research Repository (CoRR)*, vol. abs/1611.08086, 2016. [Online]. Available: http://arxiv.org/abs/1611.08086
[13] X. Fei, F. Liu, H. Xu, and H. Jin, "Towards load-balanced VNF assignment in geo-distributed NFV Infrastructure," in *IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, June 2017.
[14] A. Gupta, M. Habib, P. Chowdhury, M. Tornatore, and B. Mukherjee, "Joint Virtual Network Function Placement and Routing of Traffic in Operator Networks," *Technical Report, UC Davis*, 2015.
[15] B. Jaumard, C. Meyer, and B. Thiongane, "On column generation formulations for the RWA problem," *Discrete Applied Mathematics*, vol. 157, pp. 1291–1308, 2009.
[16] B. Jaumard and M. Daryalal, "Efficient spectrum utilization in large scale RWA problems," *IEEE/ACM Transactions on Networking*, 2017.

[17] A. Gupta, B. Jaumard, M. Tornatore, and B. Mukherjee, "Multiple Service Chain Placement and Routing in a Network-enabled Cloud," *Computing Research Repository (CoRR)*, vol. abs/1611.03197, 2016. [Online]. Available: http://arxiv.org/abs/1611.03197

[18] M. F. Habib, M. Tornatore, M. De Leenheer, F. Dikbiyik, and B. Mukherjee, "Design of disaster-resilient optical datacenter networks," *Journal of Lightwave Technology*, vol. 30, no. 16, pp. 2563–2573, 2012.

[19] Cisco, "Cisco Cloud Services Router 1000V 3.14 Series Data Sheet," http://www.cisco.com/c/en/us/products/collateral/routers/cloud-services-router-1000v-series/datasheet-c78-733443.pdf, 2015.

[20] R. Hulsermann, A. Betker, M. Jager, S. Bodamer, M. Barry, J. Spath, C. Gauger, and M. Kohn, "A set of typical transport network scenarios for network modelling," *ITG FACHBERICHT*, vol. 182, pp. 65–72, 2004.

[21] N. M. Garcia, P. P. Monteiro, M. M. Freire, J. R. Santos, and P. Lenkiewicz, "A new architectural approach for optical burst switching networks based on a common control channel," *Optical Switching and Networking*, vol. 4, no. 3, pp. 173–188, 2007.

[22] S. Ferdousi, F. Dikbiyik, M. F. Habib, M. Tornatore, and B. Mukherjee, "Disaster-aware datacenter placement and dynamic content management in cloud networks," *Journal of Optical Communications and Networking*, vol. 7, no. 7, pp. 681–694, 2015.