Column Generation Based Service Function Chaining Embedding in Multi-domain Networks

Lin, Rongping; Yu, Song; Luo, Shan; Zhang, Xiaoning; Wang, Jingyu; Zukerman, Moshe

Published in:
IEEE Transactions on Cloud Computing

Published: 01/01/2023

Document Version:
Post-print, also known as Accepted Author Manuscript, Peer-reviewed or Author Final version

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.1109/TCC.2021.3084999

Publication details:

Citing this paper
Please note that where the full-text provided on CityU Scholars is the Post-print version (also known as Accepted Author Manuscript, Peer-reviewed or Author Final version), it may differ from the Final Published version. When citing, ensure that you check and use the publisher's definitive version for pagination and other details.

General rights
Copyright for the publications made accessible via the CityU Scholars portal is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights. Users may not further distribute the material or use it for any profit-making activity or commercial gain.

Publisher permission
Permission for previously published items are in accordance with publisher's copyright policies sourced from the SHERPA RoMEO database. Links to full text versions (either Published or Post-print) are only available if corresponding publishers allow open access.

Take down policy
Contact lbscholars@cityu.edu.hk if you believe that this document breaches copyright and provide us with details. We will remove access to the work immediately and investigate your claim.

Download date: 09/07/2024
Column Generation Based Service Function Chaining Embedding in Multi-domain Networks

Rongping Lin, Song Yu, Shan Luo, Xiaoning Zhang, Jingyu Wang, and Moshe Zukerman, Life Fellow, IEEE

Abstract—Network function virtualization (NFV) achieves cost-effective network service provisioning through exploitation of virtualization and automation by decoupling network functions (software) from dedicated hardware. The software of the various devices can then be hosted by low-cost general computation devices rather than by more expensive dedicated devices. To obtain a specific network service, the traffic flow is steered to go through a specific order of network functions that are hosted by cloud computing, and this network function sequence is known as a service function chaining (SFC). To allocate computation resources for network functions and bandwidth resources between network functions in a physical network is the SFC embedding problem. In this paper, we consider the SFC embedding problem in multi-domain networks, where no domain information, like domain topology and network resource, is disclosed among domains. We propose a new optimization algorithm based on column generation method to solve this problem, which is distributedly computed in each domain. To further improve the scalability, we also provide two heuristic algorithms. We selected two networks one large (158 nodes) and one small (18 nodes) to numerically validate the proposed algorithms and demonstrate that the acceptance ratio obtained by the heuristic algorithms is close (within 5.6%) to that of the optimal algorithm.

Index Terms—Network function virtualization, service function chaining embedding, multi-domain

I. INTRODUCTION

NETWORK function virtualization (NFV) technology [1], [2] facilitates the decoupling of hardware and software in traditional dedicated network devices, where the hardware can use off-the-shelf computational equipment, such as, X86 computers. Accordingly, multiple network functions can share computational resources to improve efficiency by increasing resource utilization, and the provision of a specific network function can be efficiently scaled up or down according to traffic variability [3]. With NFV technology, different network functions can be dispatched among the underlying network to achieve more flexible and efficient resource allocation than traditional networks [4]. Networks that implement NFV technology, called NFV networks, are enabled by the so-called service function chaining (SFC), where a traffic goes through a set of virtual network functions (VNFs) in a given order to obtain a network service [5]–[9]. For example, a video streaming service from a smart phone goes through network functions that provide firewall, proxy, and video transcoder in a sequence. In this case, we say that the video streaming goes through an SFC. Note that a network function (NF) can be shared by multiple SFCs, and these network functions can be optimally hosted by computational devices in cloud datacenters through network service orchestration.

The SFC embedding problem is to use limited computation and bandwidth resources of the substrate network and to provide efficient routing of the traffic through a fixed order of network functions [10]. Usually, the SFC embedding problem is investigated in a single-domain network, such as a datacenter network. Several publications, e.g., [11]–[15], provide optimal or approximate embedding solution given the network information, including network topology, computing resource of every physical node and bandwidth capacity of every link. However, cloud computing is to decentralize the location of network functions, and provides distributed cloud computing to end users, e.g. mobile users, or Internet of Things (IoT) devices. This new computing paradigm reduces the transport cost, and provides a better user experience when computation resources are near end users. This trend makes the distributed datacenters towards regional areas, and these distributed datacenters are multi-domain operated because of the heterogeneous transport among datacenters, different technology usages, different vendors or different network management [16]. Meanwhile, traffic flows require a range of network services for different functionalities, and these traffic flows may traverse different datacenters that belong to different domains.

In a multi-domain network, detailed domain information is unavailable to other domains, and existing SFC embedding methods that were developed for a single-domain network are not applicable. In particular, existing methods for single domain SFC embedding are based on the information of the topology and resources of the entire network. However, such information is not available in the case of a network with multiple domains because multiple domains usually do not share such information. Clearly, if the multiple domains belong to different companies such information will not be shared, but even if they do belong to the same company, they...
may not share it for scalability and security reasons. Note that improved scalability and security are two of the reasons for an enterprise to use multiple domains. Accordingly, new embedding methods are needed for multi-domain networks. In this paper, based on the column generation method, we will provide an optimal algorithm and scalable heuristic algorithms for multi-domain networks. The main contributions of this paper are as follows.

1) A column generation based mathematical model for the SFC embedding problem with embedding cost minimization is provided in multi-domain networks, where the information of the topology and resources of each domain is concealed, no information is exchanged among domains.

2) A branch-and-bound based algorithm is provided to optimally solve the problem to obtain the minimal embedding cost for the SFC request, which provides the benchmark to other algorithm designs for the same problem. In the algorithm, column generation related parameters (cannot infer topology and resource information of domain with these parameters) of each domain are collected by a high-level controller, which still satisfies the information isolation between domains.

3) Two heuristic algorithms extended from the optimal algorithm are provided, which reduce the branch-and-bound tree node solving and avoid the ILP solving in the algorithms.

4) Extensive numerical results evaluate the performance of the proposed algorithms and show that one heuristic algorithm has a close performance to the optimal algorithm and with a much lower complexity.

The remainder of this paper is organized as follows. Section II discusses existing work related to SFC embedding in single and multiple domains. In Section III, we provide a problem statement for SFC embedding in multi-domain networks. In Section IV, we provide and explain the relevant column generation formulation. In Section V, three algorithms based on the column generation are provided. Section VI evaluates and validates the three algorithms by simulations. Section VI concludes this paper.

II. RELATED WORK

The SFC embedding problem has been widely studied, and various related topics have been investigated [9], [11]–[15], [17]–[26]. Ye et al. [9] investigated end-to-end delays of traffic flows traversing an embedded VNF chain with a tandem queuing model, where both computing and transmission resources are considered. Huin et al. [11] focused on the energy-efficient SFC embedding problem, and proposed several algorithms to reduce the energy consumption. Liu et al. [12] solved the problem of SFC readjustment in response to dynamic user demands, and also used a column generation algorithm to reduce the running time. Kuo et al. [13] considered the number of flows is limited by the process overhead at the node, and provided a method of SFC embedding to maximize the total amount of flows with constraints on node computation capability and link bandwidth capacity. Fan et al. [14] considered the availability problem of network service provisioning, and provided an availability-aware algorithm to minimize the physical resources usage for the SFC embedding. Sallam et al. [15] considered scenarios where network functions are provided by a mixture of physical network functions and virtual network functions, and provided methods to solve the shortest path and the maximum flow problems satisfying the SFC embedding constraints. Zeng et al. [17] measured performance interferences of the different co-located VNFs on a hardware. Guo et al. [18] investigated online and offline algorithms for the SFC embedding problem in datacenters. Fei et al. [19] introduced a proactive approach to provide new VNF instances for the overloaded ones, which is based on the demand prediction of the service function chaining. Huang et al. [20] investigated the SFC embedding problem in hybrid networks where both physical devices and virtual functions provide network services collaboratively. Savi et al. [21] investigated the impact of computation resource sharing in SFC embedding where excessive consolidation of VNFs may cause long latencies. Gu et al. [22] investigated both deployment cost and communication cost minimization problem of SFC embedding in geo-distributed data centers. Zhang et al. [23] investigated the VNF placement problem in 5G network slices where both edge cloud and core cloud servers were considered. Xiao et al. [24] investigated the SFC embedding with the deep reinforcement learning method to automatically deploy SFCs under different QoS requirements. Zhang et al. [25] provided a solution for the joint optimization of SFC embedding and request scheduling. Fei et al. [26] investigated the SFC embedding problem for load balances in geo-distributed NFV Infrastructure. However, single-domain networks are assumed in these works, where the results obtained cannot be applied to multi-domain networks.

Multi-domain networks introduce challenges to the SFC embedding problem, where domain information, such as, topology, architecture, and computation resources, is concealed by each domain [27], [28]. Dietrich et al. [28] provided a method that the central controller partitions an SFC request and distributes NFs to different domains for embedding, where the controller collects information of domains. Vaishnavi et al. [29] proposed a hierarchical approach to solve the SFC embedding problem, in which a higher level controller maintains the hierarchical topology of all domains. Iovanna et al. [16] proposed an architecture for end-to-end harmonization connections in multi-domain networks, which is similar to the hierarchical approach of [29]. In [16], the central harmonizer combines the connectivity information provided by the domains to build a virtual network topology for an efficient inter-domain path handling, and each domain has a local controller for the intra-domain path computation. Toumi et al. [30] investigated SFC embedding in multi-domain networks, where a centralized framework with a limited visibility over the global network was proposed to allow SFC partitioning over domains. Wang et al. [31] investigated virtual network function graph (VNFG) embedding that is a mesh topology of VNFs in multi-domain elastic optical networks, where the private domain contains private datacenters and the public domain contains public datacenters. Based on the knowledge of the connectivity
information among datacenters, and the computation resource information of every datacenter, the central service provider makes the embedding decision in the entire multi-domain network in a similar way it is done in single domain networks. The embedding solution indicates that the mapping between datacenters and VNFs, but the exact physical nodes that host the VNFs are not given. All these publications are based on a central controller that collects the information of all domains, but the optimality of the embedding solution is compromised because the domain information is concealed in each domain.

Another way to embed an SFC request in multi-domain networks is to distribute embed SFC among multiple domains, where domains exchange messages directly to achieve the division of the SFC for each domain and the exact embedding of each NFs is then determined inside each domain [32], [33]. Zhang et al. [32] proposed a vertex-centric distributed framework to embed an SFC request, where each domain maintains its information locally and communicates with others through an orchestrator using messages. Abujoda et al. [33] introduced a bidding mechanism to embed an SFC request, where the SFC is divided into sub-chains according to the bidding price of each domain. However, these distributed methods may also compromise the optimality of the entire embedding solution and the convergence of the message exchange is still an open problem if network status changes. The state-of-the-art approaches with and without a central controller assume certain information exchanges between domains, which may result in sensitive information such as network topology and resource status being leaked. By comparison, in this paper, we assume no information exchange among domains except the exchange of columns provided by domains which completely masks sensitive information of domains. Accordingly, no other state-of-the-art approach addresses the problem that we do (of avoiding any exchange of sensitive information).

In this paper, the dynamic traffic scenario where SFC requests arrive one-by-one is considered, and we investigate the SFC embedding problem for each arrival SFC request with the available network resources in a multi-domain network. We apply the column generation method to coordinate multiple domains without information exchange of domain topology and network resource. There may be cases that domains share some but not all resource information with each other, and this limited information may increase the embedding efficiency in some algorithms that collect information from all domains and decide embedding centrally. However, the problem investigated in this paper considers no resource information is shared among domains, and the optimal SFC embedding solution is still derived.

III. COLUMN GENERATION DESIGN
A. Column Generation Method

Column generation is an efficient method for solving a large-scale linear programming (LP) problem. Column generation leverages the fact that for many large-scale LP problems, most of the variables have zero value in the final optimal solution. Then, only a small subset of variables needs to be considered when solving the problem. The column generation method only tries to generate the variables that can improve the objective function value, i.e., to find variables with negative reduced cost for a minimization problem [34]. The commonly used LP term reduced cost with respect to an additional variable is the net gain per unit in terms of the objective function value by using that variable. With the column generation method, a restricted master problem (RMP) and a pricing problem (PP) are formulated. Fig. 1 illustrates the framework of the column generation method. The RMP is the original minimization problem with only a subset of the decision variables, so it can be solved efficiently. After the RMP is solved, dual parameters for each of the constraints in the RMP are obtained. These dual parameters are then used in the objective function of the PP. The objective function of the PP is the reduced cost of the potential new variables, and the PP identifies a new decision variable for RMP, which is obtained from the optimal solution of PP, then this variable and its related parameters are added to RMP as a new column. Next, the RMP is solved again with improved optimal solution. The RMP and the PP continue to be solved alternately to approach the final optimal solution. The iteration stops until no positive improvement is achieved. This implies that the RMP is optimal having all the useful variables.

In a multi-domain network, an individual domain can be controlled by a controller. The domain controller manages the network, allocates resources of the computation and bandwidth of the domain. As mentioned above, information from other domains is unavailable to it. We also assume that a higher-level controller is deployed to coordinate domain controllers and allocates resources among domains, where the inter-domain bandwidth is managed by the higher-level controller. In Fig. 2, we present that the controllers in a multi-domain network embed SFC requests with the column generation method design, where three network domains, S1, S2, and S3, are controlled by three domain controllers, Controller 1, Controller 2, and Controller 3, respectively, and a higher-level controller coordinates three domain controllers by sending column generation parameters to make each domain obtains the optimal decision of the SFC embedding in each domain. To embed an SFC request, the higher-level controller and the domain controllers work cooperatively. Specifically, each domain controller optimizes the embedding for its relevant part of the SFC request, and the higher-level controller combines the embedding results of the domain controllers to be the embedding solution for the entire SFC request. Then, the SFC is embedded in the multi-domain network, and each domain embeds the relevant part of the SFC request by itself and no
resource information is exchanged among domains. The RMP solving is executed at the higher-level controller which realizes the coordination among domains to guide the domain controllers deriving the optimal decision of the SFC embedding in each domain. Each domain controller solves a PP with the column generation parameters from RMP to obtain the optimal decision of the SFC embedding in each domain. The execution of the RMP in the high-level controller and the executions of the PP in domain controllers serve as a network management software that requires limited computation resources at controllers. Because the RMP has a limited number of decision variables, the RMP solving in the central controller is not computationally intensive, which makes the central controller lightly loaded and suitable for practical application. Then, each domain distributedly solves its own PP problem and provides columns that represent the SFC embedding solutions inside the domain, to the central controller, which efficiently solves the entire SFC embedding problem for even large-scale problems. It is noted that since the information of each domain is restricted within the domain, only the values of the column generation parameters (dual parameters that denote the satisfaction degree of each constraint [34]) are exchanged between the higher-level and the domain controllers, which means that domain information (including topology, computation, and bandwidth) is still isolated within individual domains. Note that the information content of dual parameter values is limited and cannot retrieve any of the more comprehensive information of individual domains.

B. Initial Solution

In the column generation method, the final optimal solution is developed from an initial solution that is added to the RMP at the beginning. In this work, to build an initial solution, a dummy domain that contains physical nodes and links is added into the original multi-domain network. For example, in Fig. 3, there is a SFC request with ingress, egress and three NFs, and the network has four domains, S1, S2 and S3 of the original multi-domain network, and an added dummy domain DS indicated by the dotted figure. In domain DS, the same number of physical nodes as the number of NFs in the SFC request are added, so domain DS has three physical nodes in the line that can embed three NFs of the SFC request. The inter-domain links are added between the dummy domain and the ingress/egress nodes of the SFC request. In Fig. 3, the SFC request has the ingress node at a physical node of S1 and the egress node at a physical node of S3. Then, domain DS has inter-domain links connecting the ingress node in S1 and the egress node in S3. Finally, the initial solution is simply obtained by the embedding of the entire SFC request to the dummy domain as shown by the red line from the ingress to the egress in the figure.

In this example, the initial solution is formed by three columns (a column of variable values indicates that the SFC embedding solution in a domain) from domains DS, S1 and S3, where DS provides a column that indicates that NF1, NF2 and NF3 are embedded into the three physical nodes in the domain DS, respectively. Domains S1 and S3 provide two columns that indicate the embedding of the ingress and egress nodes in two domains, respectively. Finally, the initial solution represented by the red dashed lines together with the gray physical nodes in the figure is formed, and the embedding cost of the initial solution is assigned with a large value to ensure that the algorithm will seek a lower cost solution.

![Fig. 3. The initial solution for an SFC request.](image)

Improving the initial solution, the new embedding solutions are obtained. Note that each embedding solution must start from an ingress node and end at an egress node which are included in the SFC request. We build a column that indicates the ingress and egress node embeddings at domain DS, and this column will be used to form new embedding solutions. For example, in Fig. 4, an embedding solution is shown, and the ingress and egress nodes are embedded into domain DS, and domain DS has only one way out and in, which is to the specified physical node (fixed ingress node) in S1 and from the specified physical node (fixed egress node) in S3, respectively. In this case, after other NFs of the SFC request are embedded, the entire solution will be formed. It is noted that the final embedding solution will ignore the node and links of the dummy domain DS, and makes the SFC embedding start from the fixed ingress node in domain S1 and ends at the fixed egress node in domain S3.
C. Columns in Column Generation

To illustrate the evolution of the embedding solution from the initial one, Fig. 5 shows the procedure of columns adding and combinations to derive the final embedding solution for the SFC request in Fig. 4. The initial solution that contains three columns from dummy domain DS, domain S1 and domain S3 is added into the RMP. Accordingly, one column indicates that the ingress is embedded in domain S1. Another column indicates that NF1, NF2 and NF3 are embedded in domain DS, and the last column indicates that the egress is embedded in domain S3. It is noted that another column from domain DS that embeds the ingress and egress nodes of the SFC is also added which will be chosen in the final solution to satisfy locations of the ingress and egress nodes. In Iteration 0, with these four columns, the RMP derives an embedding solution (select columns from the existing four columns to form the solution) for the SFC request, and three columns are selected which is the initial solution. In Iteration 1, except the dummy domain DS, each domain derives the solution of PP in its domain, and submits a column that has a negative objective function value into RMP to improve the solution of the RMP. Domain S1 submits the column for embedding NF1 and NF2 in its domain. Domain S2 obtains a PP solution with a positive objective function value and no column is submitted. Domain S3 submits the column for embedding NF3 in its domain. Finally, The RMP selects three columns that with ingress and egress nodes embedded in domain DS, NF1 and NF2 embedded in domain S1, and NF3 embedded in S3, respectively, to form a solution with a lower embedding cost than the initial solution. In Iteration 1, except the dummy domain DS, each domain derives the solution of PP in its domain, and submits a column that has a negative objective function value into RMP to improve the solution of the RMP. Domain S1 submits the column for embedding NF1 and NF2 in its domain. Domain S2 obtains a PP solution with a positive objective function value and no column is submitted. Domain S3 submits the column for embedding NF3 in its domain. Finally, The RMP selects three columns that with ingress and egress nodes embedded in domain DS, NF1 and NF2 embedded in domain S1, and NF3 embedded in S3, respectively, to form a solution with a lower embedding cost than that in Iteration 1. There is one more iteration where all domains derive the PP solutions with positive objective function value, which means the objective value cannot be improved by adding columns. The column generation method stops and the final entire embedding solution is obtained which is shown in Fig. 4 as the red lines and gray physical nodes. After the final entire embedding solution is obtained, the physical nodes and links of the dummy domain DS are deleted from the embedding solution, then the embedding solution of the SFC request in the original multi-domain network is obtained.

IV. COLUMN GENERATION BASED FORMULATION

In this section, we provide the formulation of the SFC embedding problem in a multi-domain network for a given SFC request. We note that later in the sequel, we will consider a sequence of SFC requests that arrive one-by-one.

A. Problem Statement

Given:

1) A multi-domain network $MD(G, E)$ consists of $D$ domains, where $G = \{G_1, G_2, G_3, \ldots, G_D\}$ is the set of domains in the network, and $E$ is the set of inter-domain links. We assume there is one link between adjacent domains for simplicity. For a link from domain $s$ to domain $f$, the bandwidth capacity and the unit cost are denoted by $B_{s,f}$ and $C_{s,f}$, respectively, where $s \in G$, $f \in G$ and $f$ is an adjacent domain of $s$. We assume that the information of network resources and topology of a domain is not known to other domains. It is noted that this assumption conceals the changes within a domain, where variations in a domain are common, e.g. adding or removing computation devices or topology changes.

2) The domain $G_s(N^s, L^s) \in G$ is a directed graph, where $N^s$ is the set of physical nodes and $L^s$ is the set of physical
links. A physical node \( k \in N^* \) has available computation resources (CPUs) \( P_k \) with unit cost \( c_k \), and physical link \( mn \in L^* \) has bandwidth resources \( B_{mn} \) with unit cost \( c_{mn} \). All these information is concealed in each domain.

3) An arriving SFC request contains a sequence of network functions and an fixed ingress node and an fixed egress node. For example, a web browsing application usually requires services of firewall, deep packet inspection and proxy consecutively, and the ingress and egress nodes are the access router of the user and another router connected to the web server, respectively. The computation requirement of the \( i \)th NF in the SFC request is \( \phi_i \), \( i = 1, 2, ..., S \), where \( S \) is the number of NFs in the SFC request. The bandwidth requirement for packet transmissions between two consecutive NFs \( i \) and \( j = i + 1, \) is denoted as \( b_{ij} \), \( i = 1, 2, ..., S - 1 \).

Our goal is to find the embedding solution for the SFC request at the minimal embedding cost in the multi-domain network. The domain topology and resource information are concealed by each domain, and we provide an SFC embedding algorithm that optimally integrates partial embedding solutions of individual domains to form the embedding solution for the multi-domain network. However, if the network cannot accommodate the request because of lack of resources, this request is blocked. Otherwise, the algorithm provides:

1) The embedding information of each NF, including the domain and the exact physical node where the NF is embedded.
2) The routes in the substrate network for pairs of adjacent NFs in the SFC. A route may stay within a domain or span multiple domains, which is decided by the places of the two adjacent NFs embedded. If two NFs are embedded at two different domains, the physical path traverses inter-domain links to connect the physical nodes that host the NFs.

\[ \sum_{r \in R^s} y_r^s \leq 1 \quad \forall s \in G \] (2)

Equation (2) ensures that at most one column will be selected for each domain. All selected columns are formed to be the optimal solution of the RMP.

\[ \sum_{s \in G} \sum_{r \in R^s} y_r^s A_{s,f}^{ij,r} = 1 \quad \forall i \in N^v \] (3)

Equation (3) ensures that every NF of the SFC must be embedded, and an NF must be embedded into a single domain.

\[ \sum_{r \in R^s} Out_{s,f}^{ij,r} y_r^s - \sum_{r \in R^f} In_{i,f}^{ij,r} y_r^f = 0 \quad \forall ij \in L^v, s \in G, f \in G \] (4)

Equation (4) ensures the flow conservation from domain \( s \) to \( f \). Specifically, for a virtual link embedding traversed the link between domain \( s \) and its adjacent domain \( f \), the number of outgoing flow from domain \( s \) to domain \( f \) and the number of incoming flow from domain \( s \) to domain \( f \) are equal.

\[ \sum_{ij \in L^v} \sum_{r \in R^s} Out_{s,f}^{ij,r} b_{ij} y_r^s \leq B_{sf} \quad \forall s \in G, f \in G \] (5)

Equation (5) ensures that an inter-domain link have enough bandwidth if this link is used in the SFC embedding.

### B. RMP Formulations

The objective of the RMP is to minimize the SFC embedding cost by choosing the best combination of the embedding results provided by different domains. Specifically, in the RMP, the solution is the combination of the selected columns provided by the PP solutions at different domains, where a column is the embedding result for the relevant part of the SFC request in that domain. Multiple columns are selected and combined to form an end-to-end SFC embedding solution for the multi-domain network by the RMP. The following notations are used in RMP. The notations used in RMP are defined in Table I.

\[ \text{Min } \sum_{r \in R^s} \sum_{s \in G} \left( \text{Cst}(y_r^s) y_r^s + \sum_{ij \in L^v} \sum_{f \in G} C_{s,f} Out_{s,f}^{ij,r} b_{ij} y_r^s \right) \] (1)

Equation (1) is the objective function of RMP, which is to minimize the cost of SFC embedding in the multi-domain network. The first term is the embedding cost in each domain including bandwidth and computation costs, and the second term is the bandwidth cost of the traversed inter-domain links.
C. PP Formulations

A solution of PP in a domain provides a partial embedding solution of the SFC, which is provided as a column in RMP. Because a domain keeps its domain information locally, the domain information is unknown to other domains. In this section, for a specific domain, a dummy node that represents all adjacent domains is added. For example, in Fig. 6, a dummy node $k^*$ takes places of domains S1 and S3, and $k^*$ is connected to the boundary nodes of the domain S2. With this method, the PP solving in domain S2 is isolated from other domains.

![Fig. 6. A domain with a dummy node](image)

The PP in the domain is to derive the embedding solution for the SFC request in the domain and the dummy node, where NFs can be embedded in physical nodes of the domain, or the dummy node $k^*$. If an NF is embedded into the dummy node $k^*$ according to a PP solution, the NF is not embedded in domain S2 in this solution. Some notations of RMP are borrowed and extended in this subsection to make the formulation consistent between RMP and PP. Because we use dummy nodes and links in our formulation, to avoid confusion, we will use the terms physical node or physical link to refer to an existing node or link in the substrate network and the terms dummy node or dummy link to refer to node $k^*$ and links connect $k^*$. The notations used in PP are defined in Table II

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G_s(N^s, L^s)$</td>
<td>A directed graph represents a domain $s$, where $N^s$ is the set of nodes including physical nodes in the domain $s$ and the dummy node $k^<em>$, and $L^s$ is the set of links including physical links in the domain $s$ and dummy links associated with $k^</em>$</td>
</tr>
<tr>
<td>$I^s_{mk^*}$</td>
<td>It denotes domain $f$ can be directly reached through a dummy link $mk^*$ from domain $s$ or not</td>
</tr>
<tr>
<td>$I^s_{k^*n}$</td>
<td>It denotes domain $s$ can be directly reached from domain $f$ through a dummy link $k^*n$ or not</td>
</tr>
<tr>
<td>$E^s$</td>
<td>Set of dummy links</td>
</tr>
<tr>
<td>$\phi_i$</td>
<td>Computational resource requirement of NF $i$</td>
</tr>
<tr>
<td>$P_k$</td>
<td>Available computational resources of physical node $k$</td>
</tr>
<tr>
<td>$B_{mn}$</td>
<td>Available bandwidth of physical link $mn$</td>
</tr>
<tr>
<td>$c_k$</td>
<td>Unit cost of computational resources of physical node $k$</td>
</tr>
<tr>
<td>$x^*_m$</td>
<td>Unit cost of physical link $mn$</td>
</tr>
<tr>
<td>$M$</td>
<td>Boolean variable. It indicates whether the $i$th NF is embedded in the physical node $m$ or not. Variable $x^*_m$ will not be submitted to RMP due to the disclose policy</td>
</tr>
<tr>
<td>$M^{ij}_{mn}$</td>
<td>Boolean variable. It indicates whether the virtual link $ij$ is embedded in this domain and traverses the physical link $mn$ or not. Variable $M^{ij}_{mn}$ will not be submitted to RMP</td>
</tr>
<tr>
<td>$A^i$</td>
<td>Boolean variable. It indicates whether the $i$th NF is embedded in this domain or not. The variable value $A^i$ will be submitted to RMP as $A^{i,\ast}$ in the $i$th column</td>
</tr>
<tr>
<td>$Out^i_f$</td>
<td>Boolean variable. It indicates whether the $i$th NF is embedded in this domain and the next NF $j$ is embedded in the adjacent domain $f$ or not. The variable value $Out^i_f$ will be submitted to RMP as $Out^{i,\ast}_{f}I^s$ in the $r$th column</td>
</tr>
<tr>
<td>$In^i_j$</td>
<td>Boolean variable. It indicates whether the $i$th NF is embedded in the adjacent domain $f$ and the next NF $j$ is embedded in this domain or not. The value of variable $In^i_j$ will be submitted to RMP as $In^{i,\ast}_{j}\ast$ in the $r$th column</td>
</tr>
</tbody>
</table>

\[
\sum_{i \in N^s, m \in N^s} c_m \phi_i x^i_m + \sum_{ij \in L^s, mn \in L^s - E^s} c_{mn} b_{ij} M^{ij}_{mn} + \sum_{ij \in L^s, f \in G} \left[ Out^{ij}_f (C_{s,f} b_{ij} - \beta^{ij}_{s,f} + \gamma_{s,f} b_{ij}) \right] + \sum_{ij, r \in L^s} \left[ In^{ij}_r (\beta^{ij}_{r,s} - \pi_s) \right] = \sum_{i \in N^s} \alpha_i A^i - \pi_s (6)
\]

Equation (6) is the PP objective function (reduced cost) derived by a Lagrangian function from RMP, where $\pi_s, \alpha_i, \beta^{ij}_{s,f},$ and $\gamma_{s,f}$ are the dual parameters of constraints (2)-(5) in the RMP, respectively. If a non-negative objective function value is obtained, the objective function value of RMP cannot be improved by the PP of this domain. Otherwise, the partial embedding solution in this domain will be added to the RMP as a new column, and the embedding cost $C^{st}(y^s)$ in this domain calculated as follows is also submitted to RMP in the column.

\[
\sum_{i \in N^s, m \in N^s} c_m \phi_i x^i_m + \sum_{ij \in L^s, mn \in L^s - E^s} c_{mn} b_{ij} M^{ij}_{mn} = \sum_{i \in N^s} x^i_m = 1 \quad \forall i \in N^s (7)
\]

If all of the PP objective function solutions in all domains are non-negative, the final optimal solution is obtained and the iteration stops.

\[
\sum_{m \in N^s} x^i_m \leq 1 \quad \forall m \in N^s - \{k^*\} (8)
\]

Equation (7) ensures that each NF of the SFC request is embedded. Equation (8) ensures that each physical node hosts at most one NF of the SFC request (a physical node can host multiple NFs with the same type from different SFC requests). This is because a specific type of NF usually requires specific hardware (like I/O, storage, or computation), and to reduce the cost of deployed hardware, each physical node is equipped with specific hardware and provides only one type of NF. In this paper, we assume that each physical node can provide one type of NF, which also means for an SFC request, each physical node hosts at most one NF of the SFC request. However, the dummy node $k^*$ denotes all other domains and is not constrained by this equation.
\[ \sum_{m \in N^s} M_{mk}^{ij} - \sum_{n \in N^s} M_{kn}^{ij} = x_k^i - x_k^j \quad \forall ij \in L^v, k \in N^s \quad (9) \]

Equation (9) constrains the embedding of SFC link \( ij \) between NFs \( i \) and \( j \). There are three cases according to values of the right-hand side of the equation: 1) The right-hand side is equal to 0, which means the \( x_k^i \) and \( x_k^j \) are of the same value. In this case, this equation ensures that the number of incoming links is equal to that of outgoing links at physical node \( k \), so that the left-hand side of the equation is equal to 0; 2) The right-hand side is equal to 1 (\( x_k^i \) is equal to 1 and \( x_k^j \) is equal to 0), which means the NF \( j \) is embedded in node \( k \) and physical path that embeds virtual link \( ij \) ends at node \( k \). Then, this equation ensures that for node \( k \), the number of incoming links is larger than the number of outgoing links by 1; 3) The right-hand side is equal to -1 (\( x_k^i \) is equal to 0 and \( x_k^j \) is equal to 1), which means the NF \( i \) is embedded in node \( k \). Then, this equation ensures that for node \( k \), the number of outgoing links is larger than the number of incoming links by 1.

\[ \sum_{m \in N^s - \{k\}} x_m^i = A^i \quad \forall i \in N^v \quad (10) \]

In Equation (10), if NF \( i \) is embedded on a physical node of the domain, \( A^i \) is set to 1, otherwise 0.

\[ \sum_{f \in G} I_{k_n}^{ij} f = M_{kn}^{ij} \quad \forall n \in N^s, ij \in L^v \quad (11) \]

\[ \sum_{f \in G} O_{mk}^{ij} f = M_{mk}^{ij} \quad \forall m \in N^s, ij \in L^v \quad (12) \]

Equation (11) ensures that if the embedding of virtual link \( ij \) traverses dummy link \( k^*n \), there must be an adjacent domain \( f \) embeds NF \( i \), which makes variable \( I_{k_n}^{ij} \) to be 1. Similarly, Equation (12) ensures if the embedding of virtual link \( ij \) traverses dummy link \( mk^* \), there must be an adjacent domain \( f \) embeds NF \( j \), which make variable \( O_{mk}^{ij} \) to be 1.

\[ \sum_{f \in G} (I_{ij}^{ij} - O_{ij}^{ij}) = A^j - A^i \quad \forall ij \in L^v \quad (13) \]

There are three cases according to values of the right-hand side of Equation (13): 1) The right-hand side is equal to 0. This implies that \( A^i \) and \( A^j \) are of the same value. In this case, NFs \( i \) and \( j \) are both or neither embedded in this domain, thus the numbers of traversed inter-domain links going into and going out from this domain are equal, so that the left-hand side of the equation will be equal to 0; 2) The right-hand side is equal to 1 (\( A^j \) is equal to 1 and \( A^i \) is equal to 0). This implies that NF \( j \) is embedded in this domain and NF \( i \) is not. In this situation, the number of traversed inter-domain links going into the domain is larger than that of going out from the domain by 1; 3) The right-hand side is equal to -1 (\( A^j \) is equal to 0 and \( A^i \) is equal to 1), which implies that NF \( i \) is embedded in this domain and NF \( j \) is not. In this situation, the number of traversed inter-domain links going out from the domain is larger than those going into the domain by 1.

\[ \sum_{i \in N^v} \phi_i x_k^i \leq P_k \quad \forall k \in N^s \quad (14) \]

\[ \sum_{ij \in L^v} b_{ij} M_{mn}^{ij} \leq B_{mn} \quad \forall mn \in L^s \quad (15) \]

Equation (14) ensures that NFs of the SFC request are embedded into the physical nodes that have sufficient computational resources. Equation (15) ensures that the bandwidth resource used by the SFC request is less than the available bandwidth.

V. ALGORITHMS DESIGN

In this section, we present an optimal algorithm using the branch-and-bound method to derive integer solutions, and two heuristic algorithms are also proposed for time efficiency.

A. Optimal Solution with Branch-And-Bound

To solve the column generation model above, the RMP is first linearly relaxed. We refer here to a common Integer Linear Programming (ILP) or Mixed ILP (MILP) heuristic algorithm called linear relaxation [36], where the integer constraints are relaxed and the optimal solution of the LP problem is used to obtain integer solutions by the branch-and-bound method. A branch-and-bound tree graph will be generated during the problem solving, and each node of the tree graph represents a problem for which the solution is derived by iterations between RMP and PP. The CG-branch&bound algorithm (CG stands for column generation) is designated as Algorithm 1 in the paper. This algorithm is the general framework of column generation method with branch-and-bound which is similar to our previous work [37]. The CG-branch&bound algorithm derives the optimal solution for the SFC embedding problem, which provides the benchmark for other non-optimal algorithms. It is noted that the optimal ILP is non-polynomial, and one usual way to solve ILP problems is to use linear relaxation and the branch-and-bound method to derive the integer solution. Similarly, in this paper, we relax the original ILP problem (RMP) into the linear problem and apply the branch-and-bound method to derive the integer solution. The solving of linear problem in the CG-branch&bound algorithm is optimal, and the branch-and-bound method maintains the optimality of the integer solution. Then, the algorithm is optimal.

In the algorithm, \( TNodeList \) is a first-in-first-out(FIFO) list, and is used to store nodes of the branch-and-bound tree. \( UB \) denotes the best solution already obtained. In line 2, the initial solution is added to the RMP, and the RMP is linearly relaxed in line 3. In line 4, the root of the tree is added to \( TNodeList \). In the while loop, from line 6 to line 12, a node of the branch-and-bound tree is solved. In line 6, a node is popped from \( TNodeList \), then, the node is solved by iterations between RMP and PP from line 7 to line 9 until the reduced cost is not negative. When iteration stops, three cases are checked: 1) if \( Z \) is larger than \( UB \), this node will be discarded; 2) if \( Z \) is smaller than \( UB \) and \( Y \) is an integer solution, the current best solution \( Y^* \) and \( UB \) are updated; 3)
Algorithm 1: CG-branch&bound algorithm

Input: An arriving SFC request \( G^*(N^*, L^*) \), and existing network resources.
Output: The optimal embedding solution \( Y^* \), and updated network resources.
BEGIN:
1. \( TNodeList \leftarrow \emptyset, UB \leftarrow \infty, Y^* \leftarrow \emptyset \).
2. Add the initial solution to the RMP.
3. Relax the RMP to be a continuous linear programming.
4. Add the root of branch-and-bound tree to \( TNodeList \).
5. while \( TNodeList \neq \emptyset \) do
   6. Pop a node from \( TNodeList \).
   7. Solve the relaxed RMP to obtain solution \( Y \) and objective function value \( Z \).
   8. Solve the PPs at domains.
   9. If a column with a negative reduced cost has been found, add the column to the RMP, go to line 7.
   10. If \( Z \geq UB \), go to line 5.
   11. If \( Y \) is an integer solution, \( Y^* \leftarrow Y \) and \( UB \leftarrow Z \).
   12. If \( Y \) is a fraction solution, branch and add two new tree nodes to \( TNodeList \).
13. If \( Y^* \) is not the initial solution, allocate resources for the SFC request according to \( Y^* \).
END

if \( Z \) is smaller than \( UB \) and \( Y \) is a fractional solution, two new nodes are generated by branching and added to \( TNodeList \). After all nodes in \( TNodeList \) have been processed, \( Y^* \) is the optimal SFC embedding solution. Finally, in line 13, the network resources are allocated according to \( Y^* \), otherwise, the SFC request is blocked.

B. Rounding off Algorithm

In the CG-branch&bound algorithm, the number of nodes in the branch-and-bound tree may be large and the total running time can be quite long. Considering that the solution of the root of the branch-and-bound tree provides a lower bound of the optimal integer solution, we propose a heuristic algorithm, named CG-rounding off algorithm hereafter, that is based on the solution of the tree root and uses the rounding off method to convert fractional variables one-by-one into integers. However, both the feasibility and optimality are not guaranteed.

The CG-rounding off algorithm is designated as Algorithm 2 in the paper. This algorithm is similar to the CG-branch&bound algorithm. Specifically, after the solution of the root in the CG-rounding off algorithm is obtained in the same way as in the CG-branch&bound algorithm, the CG-rounding off algorithm identifies variables \( y_r^s \) that have fractional values in line 12, and chooses the one \( y_r^s \) that has the largest fractional value to round to 1. This means the domain \( s \) will stop providing columns to RMP in the further calculations, and RMP will stick to the \( r \)th column of domain \( s \) in the column selection. A new tree node will be generated, in which the \( r \)th column will be explicitly selected. The variable rounding off and new node adding will continue until all variables \( y_r^s \) become integer. It is noted that because one domain stops providing columns to RMP after each rounding off operation, the number of node solving in the algorithm is no larger than the number of domains, which makes the running time of CG-rounding off algorithm is shorter than that of the CG-branch&bound algorithm. In line 13, the feasibility and optimality of the rounding off solution are checked, which can be applied as the final embedding solution.

C. Approximate PP Algorithm

The branch-and-bound tree node solving in the CG-branch&bound and the CG-rounding off algorithms is based on iterations between the relaxed RMP and PP, where PP is an ILP problem which is not scalable. We provide an CG-approximate PP algorithm that is the same as the CG-rounding off algorithm except that the PP is approximately solved at line 8 in the CG-rounding off algorithm.

In the CG-approximate PP algorithm, an auxiliary graph is built, and the SFC embedding in a domain (PP problem) has been transformed to be the shortest path problem. The objective of the SFC embedding in a domain is to minimize the reduced cost as shown in (6), which is the objective function of the PP problem. To design the shortest path routing that has the same objective function as the PP problem, different
Each NF is embedded into the corresponding layer (NF1 is the beginning of the algorithm (shown at Iteration 0 in Fig. 5). The original links \( L \) are the copies of \( * \) belongs to). It is noted that if a NF is embedded in the domain DS by selecting the column (represent the ingress and egress nodes are embedded in the dummy domain) that was added at the beginning of the algorithm (shown at Iteration 0 in Fig. 5). Each NF is embedded into the corresponding layer (NF1 is embedded into the layer where dummy node \( k^* - 1 \) belongs to, NF2 is embedded into the layer where dummy node \( k^* - 2 \) belongs to, and NF3 is embedded into the layer where dummy node \( k^* - 3 \) belongs to). It is noted that if a NF is embedded in the dummy node of the corresponding vertical layer, it means this NF is not embedded in this domain.

The costs of these links are assigned as Equation (16), where there are six cases of link cost assignments, and the computation cost is contained only in the cost of links between vertical layers. 1) The first is to set zero cost to the link between dummy nodes, because it indicates that two nodes of the SFC request and the link between them are embedded in other domains which is not considered by this domain; 2) The second is the physical link cost in layer \( i \). The link cost \( c_{mn} b_{ij} \) considers the bandwidth requirement \( b_{ij} \) between NFs \( i \) and \( j \); 3) The third is the cost of solid line that is from physical nodes to the dummy node in the same vertical layer. The usage of this link indicates that the inter-domain links have been traversed; 4) The fourth is the cost of the dashed line that is from physical nodes to the next layer dummy node. The usage of this link indicates that a NF is not embedded in the domain but the next NF is; 5) The fifth is the cost of solid line between physical layers, which contains the computation cost and bandwidth cost if two adjacent NFs are embedded into this domain (the computation cost of a NF is calculated only one time from the SFC perspective); 6) The sixth is the cost of dashed line that is from physical nodes to the next layer dummy node. This cost contains both computation and bandwidth cost, and the usage of this link indicates that a NF is embedded in the physical node of the domain but the next NF is not.

\[
 c'_{mn} = \begin{cases} 
 1) 0, & m = k^* - i, n = k^* - j \\
 2) c_{mn} b_{ij}, & m \in N^s_i - \{k^* - i\}, n \in N^s_i - \{k^* - i\} \\
 3) C_s f_{ij} - \beta^j f_{ij} + \gamma_{s,f} b_{ij}, & m \in N^s_i - \{k^* - i\}, n = k^* - i \\
 4) \beta^j f_{ij} + c_{mn} b_{ij}, & m = k^* - i, n \in N^s_j - \{k^* - j\} \\
 5) c_m \phi_i - \alpha_i + c_{mn} b_{ij}, & m \in N^s_i - \{k^* - i\}, n \in N^s_j - \{k^* - j\} \\
 6) c_m \phi_i - \alpha_i + C_s f_{bij} - \beta^j f_{ij} + \gamma_{s,f} b_{ij}, & m \in N^s_i - \{k^* - i\}, n = k^* - j. 
\end{cases} 
\]

(16)

where \( N^s_i \) is the node set of vertical layer \( i \), and \( i, j \in \{I, 1, 2, 3, E\} \).

D. Complexity Analysis

The CG-branch&bound algorithm provides the optimal solution for the SFC embedding with the minimal embedding cost. However, the complexity of the algorithm is not polynomial, because the branch-and-bound method is used to derive the integer solution, which causes the worse case of the number of tree node solving is exponential. The CG-rounding off algorithms reduces the number of tree node solving to be \( D \) (the number of domains), but at each tree node solving, the solving of PP is optimally as ILP solving, where the ILP solving is non-polynomial. Then the CG-rounding off algorithm is non-polynomial. The CG-approximate PP algorithm has the same number of tree node solving \( (D) \) as the CG-rounding off algorithms, and at each tree node solving, the PP is approximately solved based on the shortest path algorithm which is a polynomial algorithm. Then the
complexity of a tree node solving is polynomial. There are total $D$ tree node solving in the algorithm which makes the complexity of the CG-approximate PP algorithm polynomial.

The CG-branch&bound algorithm derives the optimal solution for the investigated embedding problem at the cost of a very high complexity, and two heuristic algorithms have lower complexities than the CG-branch&bound algorithm, but the optimality cannot be guaranteed.

VI. Numerical Results

In this section, we compare the CG-branch&bound algorithm, the CG-rounding off algorithm and the CG-approximate PP algorithm in terms of acceptance ratio, number of domains traversed, total embedding cost, virtual link embedding cost and NF embedding cost. We note that another mathematical method using ILP to formulate the same SFC embedding problem has been verified, where all information is shared among domains. This is the same as the SFC embedding problem in single-domain networks, and the simulation result is the same as the CG-branch&bound algorithm because both methods provide the optimal embedding solution for the arriving SFC request. The result is not shown for brevity.

The CG-branch&bound algorithm provides an optimal solution for SFC embedding in a multi-domain network, which provides a benchmark for other algorithms including the heuristic algorithms proposed by this paper. As demonstrated by the simulation results shown below, these heuristic algorithms can provide near optimal performance evidenced by the comparison with the CG-branch&bound benchmark.

In the simulation, experiments were carried out on two arbitrary selected networks, one small and one large. In the small network (18 physical nodes), there are 3 domains, one domain has 7 physical nodes and 20 physical links, another one has 6 physical nodes and 16 physical links, the last one has 5 physical nodes and 14 physical links, and there are 6 inter-domain links in the network. In the large network (158 physical nodes), there are 5 domains, one domain has 24 physical nodes and 86 physical links, one has 29 physical nodes and 82 physical links, one has 32 physical nodes and 82 physical links, one has 35 physical nodes and 100 physical links, the last one has 38 physical nodes and 140 physical links, and there are 12 inter-domain links in the network. For each of the two networks, SFC requests arrived sequentially and were assumed to follow Poisson processes with rate $\lambda$. The holding time of the requests were assumed to be exponentially distributed with mean $1/\mu$. The value of $\mu$ was set to 1 in all the experiments, and $\lambda$ was varied to consider different traffic loads. Then, the network load equals to $\lambda/\mu$ (Erlang).

The number of NFs in an SFC request were assumed to be uniformly distributed (we use $\text{unif}(m, n)$ to denote a discrete uniform distribution in the range $m, m + 1, \ldots, n$), and the ingress and egress nodes of each SFC request are randomly chosen among the physical nodes of the entire network, then the ingress and egress nodes can be in the same domain or in two different domains. The computation and bandwidth requirements were also assumed to be uniformly distributed, assuming that a physical node can provide any VNF with the virtualized technology. A physical node can host any NF if it has sufficiently available computation resources, so in the experiments, an NF of the SFC request is embedded to a physical node satisfying the computation requirement. Table III provides the parameter settings used in the simulations of the small and large networks. We conducted one set of independent 11 experiments for each network all with the same given traffic load. Then, we repeated the experiments on the same network with different traffic loads. Using the results of the 11 experiments for each traffic load level, error bars of 95% confidence intervals based on Student’s t-distribution were provided.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Small network</th>
<th>Large network</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$ (arrival rate)</td>
<td>$25 \sim 55$</td>
<td>$300 \sim 360$</td>
</tr>
<tr>
<td>$1/\mu$ (mean of service time)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Domains</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Nodes in a domain</td>
<td>unif(5, 8)</td>
<td>unif(24, 40)</td>
</tr>
<tr>
<td>Links in a domain</td>
<td>unif(8, 20)</td>
<td>unif(80, 140)</td>
</tr>
<tr>
<td>Node computation</td>
<td>50</td>
<td>200</td>
</tr>
<tr>
<td>Link bandwidth</td>
<td>30</td>
<td>100</td>
</tr>
<tr>
<td>Inter-domain link bandwidth</td>
<td>60</td>
<td>400</td>
</tr>
<tr>
<td>Computation cost</td>
<td>unif(1, 3) cost-unit</td>
<td>unif(1, 4) cost-unit</td>
</tr>
<tr>
<td>Bandwidth cost</td>
<td>unif(1, 3) cost-unit</td>
<td>unif(1, 4) cost-unit</td>
</tr>
<tr>
<td>Inter-domain bandwidth cost</td>
<td>unif(2, 6) cost-unit</td>
<td>unif(2, 8) cost-unit</td>
</tr>
<tr>
<td>Number of NFs in an SFC</td>
<td>unif(2, 4)</td>
<td>unif(2, 6)</td>
</tr>
<tr>
<td>Computation requirement</td>
<td>unif(1, 10)</td>
<td>unif(1, 40)</td>
</tr>
<tr>
<td>Bandwidth requirement</td>
<td>unif(1, 3)</td>
<td>unif(1, 10)</td>
</tr>
<tr>
<td>Number of requests</td>
<td>10000</td>
<td>100000</td>
</tr>
<tr>
<td>Number of experiments</td>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>
Figs. 8–12 present numerical results that have been obtained for the small size network where different network traffic loads are considered. Fig. 8 presents the results of the three algorithms on the acceptance ratio. In the figure, we can see that as expected, for all three algorithms the acceptance ratio decreases as the network load increases. The CG-branch&bound algorithm has the highest acceptance ratio, which provides the optimal embedding solving for each SFC request, followed by the CG-rounding off algorithm and the CG-approximate PP algorithm. The CG-rounding off algorithm has almost the same acceptance ratio as the CG-branch&bound algorithm. It validates our proposed rounding off method which reduces computations of the CG-branch&bound algorithm while the high performance is maintained. The CG-rounding off algorithm obtains an integer solution with rounding off variables one-by-one, and uses limited and fine steps to approach a good quality integer solution. In addition, the CG-approximate PP algorithm achieves a lower average acceptance ratio than the CG-rounding off algorithm by about 0.4%. Also, the average value difference between the CG-branch&bound algorithm and the CG-approximate PP algorithm is no more than 0.5%. This is explained by the fact that in addition to applying the rounding off technology to reduce the number of branch-and-bound tree node solving, the CG-approximate PP algorithm uses the auxiliary graph to approximately solve the PP problem, which provides lower quality non-optimal embedding solutions as compared to the CG-rounding off algorithm.

Fig. 9 shows the average embedding cost of SFC requests. When network load increases, all the three algorithms exhibit a similar increasing trend. This is because a higher network load leads to more SFC requests competing for limited network resources where costly physical nodes and long physical paths are used in the embedding solutions. In Fig. 9, the average embedding cost of CG-approximate PP algorithm is the highest, followed by the CG-rounding off algorithm and the CG-branch&bound algorithm. A higher embedding cost implies more resource consumption, which leads to lower acceptance ratios of an algorithm (shown in Fig. 8).

From our experiments, we observe lower average acceptance ratios by about 0.5%, and higher average embedding cost by about 2%, for the CG-approximate PP algorithm as compared to the CG-branch&bound algorithm. These results demonstrate that the CG-approximate PP algorithm is an efficient algorithm and can be applied in large scale networks.

Figs. 10 and 11 show the link embedding cost caused by inter-domain links and links in the same domain (called domain links in this section). In Fig. 10, the CG-approximate PP algorithm has the highest inter-domain link embedding cost value, followed by the CG-rounding off algorithm and the CG-branch&bound algorithm. It is noted that the unit cost of an inter-domain link is usually higher than that of a domain link. In Figs. 11 and 12, domain link embedding cost and NF embedding cost of three algorithms are compared, where the same order of the three algorithms as in Fig. 10 can be also observed.

Figs. 8–12 present numerical results that have been obtained for the small size network where different network traffic loads are considered. Fig. 8 presents the results of the three algorithms on the acceptance ratio. In the figure, we can see that as expected, for all three algorithms the acceptance ratio decreases as the network load increases. The CG-branch&bound algorithm has the highest acceptance ratio, which provides the optimal embedding solving for each SFC request, followed by the CG-rounding off algorithm and the CG-approximate PP algorithm. The CG-rounding off algorithm has almost the same acceptance ratio as the CG-branch&bound algorithm. It validates our proposed rounding off method which reduces computations of the CG-branch&bound algorithm while the high performance is maintained. The CG-rounding off algorithm obtains an integer solution with rounding off variables one-by-one, and uses limited and fine steps to approach a good quality integer solution. In addition, the CG-approximate PP algorithm achieves a lower average acceptance ratio than the CG-rounding off algorithm by about 0.4%. Also, the average value difference between the CG-branch&bound algorithm and the CG-approximate PP algorithm is no more than 0.5%. This is explained by the fact that in addition to applying the rounding off technology to reduce the number of branch-and-bound tree node solving, the CG-approximate PP algorithm uses the auxiliary graph to approximately solve the PP problem, which provides lower quality non-optimal embedding solutions as compared to the CG-rounding off algorithm.

Fig. 9 shows the average embedding cost of SFC requests. When network load increases, all the three algorithms exhibit a similar increasing trend. This is because a higher network load leads to more SFC requests competing for limited network resources where costly physical nodes and long physical paths are used in the embedding solutions. In Fig. 9, the average embedding cost of CG-approximate PP algorithm is the highest, followed by the CG-rounding off algorithm and the CG-branch&bound algorithm. A higher embedding cost implies more resource consumption, which leads to lower acceptance ratios of an algorithm (shown in Fig. 8).

From our experiments, we observe lower average acceptance ratios by about 0.5%, and higher average embedding cost by about 2%, for the CG-approximate PP algorithm as compared to the CG-branch&bound algorithm. These results demonstrate that the CG-approximate PP algorithm is an efficient algorithm and can be applied in large scale networks.

Figs. 10 and 11 show the link embedding cost caused by inter-domain links and links in the same domain (called domain links in this section). In Fig. 10, the CG-approximate PP algorithm has the highest inter-domain link embedding cost value, followed by the CG-rounding off algorithm and the CG-branch&bound algorithm. It is noted that the unit cost of an inter-domain link is usually higher than that of a domain link. In Figs. 11 and 12, domain link embedding cost and NF embedding cost of three algorithms are compared, where the same order of the three algorithms as in Fig. 10 can be also observed.

We also provide numerical results for the large size network shown in Figs. 13–17. In Fig. 13, the order of the three algorithms is the same as that in Fig. 8 for the small size net-
work. The average difference between the CG-branch&bound algorithm and the CG-rounding off algorithm is about 0.4%, and the average difference between the CG-branch&bound algorithm and the CG-approximate PP algorithm is about 5.6%. The performance of the CG-approximate PP algorithm is degraded as compared to the CG-branch&bound algorithm for this large size network, and the reason is that for the large size network, the PP solving with the auxiliary graph approximates the SFC embedding in domains, and the inaccuracy of this approximation increases when the network size increases.

In Fig. 14, the CG-approximate PP algorithm has the lowest average embedding cost, this is because the acceptance ratio of the algorithm is quite low (shown in Fig. 13), which leads that the accepted SFC requests can find low cost solutions with plenty of resources. In Fig. 15, the order of the three algorithms is the same as that in Fig. 10. In Figs. 16 and 17, domain link embedding cost and NF embedding cost of three algorithms are compared, and the CG-rounding off algorithm has higher values than the CG-branch&bound algorithm in both figures. Because of the low acceptance ratio, the CG-approximate PP algorithm has the lowest domain link embedding cost when the network load is lower than 330 Erlang as shown in Fig. 16, and has the lowest NF embedding cost as shown in Fig. 17. We also carry out experiments in a series of randomly generated networks following the parameter settings in Table III, the same trends and conclusions have been obtained.

To further demonstrate the performance of our algorithms when the number of domains varies, we have carried out more simulations in multi-domain networks, where the number of domains varies from 5 to 9 and the domains are small randomly generated networks with parameter settings as provided in Table III. The arrival rate is fixed at 70 Erlangs. The acceptance ratios of three algorithms in multi-domain networks that contain different numbers of domains are provided in Table IV. It is shown that the three algorithms have the same acceptance ratio order as obtained by the other simulations described above, where the CG-branch&bound algorithm has the highest value, followed by the CG-rounding off algorithm and the CG-approximate PP algorithm. When the number of domains increases, the acceptance ratio of each algorithm increases, because there are more network resources that can accommodate more SFC requests.

### Table IV

<table>
<thead>
<tr>
<th>Number of domains</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG-branch&amp;bound</td>
<td>0.735</td>
<td>0.88</td>
<td>0.974</td>
<td>0.999</td>
<td>0.999</td>
</tr>
<tr>
<td>CG-rounding off</td>
<td>0.726</td>
<td>0.866</td>
<td>0.944</td>
<td>0.949</td>
<td>0.958</td>
</tr>
<tr>
<td>CG-approximate PP</td>
<td>0.721</td>
<td>0.849</td>
<td>0.898</td>
<td>0.914</td>
<td>0.938</td>
</tr>
</tbody>
</table>

### VII. Conclusion

We have considered the SFC embedding problem in a multi-domain network where topology and resource information of each domain are concealed from each other. The column
generation method has been used to derive embedding solutions in the proposed two-level controllers, where the higher-level central controller effectively solves the RMP problem and each low-level controller of individual domain solves a PP distributedly. Accordingly, three algorithms, the CG-branch&bound algorithm, the CG-rounding off algorithm and the CG-approximate PP algorithm, have been introduced based on column generation. Specifically, the CG-branch&bound algorithm can provide an optimal solution, the CG-rounding off algorithm reduces the problem computation complexity by rounding off the LP solutions, and the CG-approximate PP algorithm solves the PP approximately to reduce the computation complexity. Numerical results have demonstrated very close (within 0.5% difference) acceptance ratio between the CG-branch&bound and the CG-approximate PP algorithms for the small size network. For the large size network, this difference increases to about 5.6%. The CG-approximate PP algorithm has the lowest computation complexity at the cost of small inaccuracy in acceptance ratio and still provides efficient resource allocations in multi-domain networks.

REFERENCES


Rongping Lin received the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, 2013. He is currently an Associate Professor with the School of Information and Communication Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, P.R. China. From 2013 to 2014, he was a Senior Research Assistant in City University of Hong Kong. His research interests include optimization, machine learning, and their applications in wire/wireless networks.

Song Yu is pursuing the M.S. degree with the School of Information and Communication Engineering, UESTC, P.R. China. His research interest includes optimization and network function virtualization.

Shan Luo received the Ph.D. degree in information engineering from Nanyang Technological University in 2014. She is currently an Associate Professor with the School of Aeronautics and Astronautics, UESTC, P.R. China. Her research interests include wireless communications and optimization.

Xiaoning Zhang is a Professor with the School of Information and Communication Engineering, UESTC, P.R. China. His research interests include network modeling and network management in communication networks.

Jingyu Wang is a Professor with the State Key Laboratory of Networking and Switching Technology at Beijing University of Posts and Telecommunications. His main research interests include broad aspects of Intelligent Network, Edge Computing, Machine Learning, Knowledge-Defined Network, Internet of Vehicles and Intent-Based Network.

Moshe Zukerman (M’87-SM’91-F’07-LF’20) received the B.Sc. degree in industrial engineering and management, the M.Sc. degree in operations research from the Technion-Israel Institute of Technology, Haifa, Israel, and the Ph.D. degree in engineering from University of California, Los Angeles, in 1985. He was an independent consultant with the IRI Corporation and a Postdoctoral Fellow with the University of California, Los Angeles, in 1985–1986. In 1986-1997, he was with Telstra Research Laboratories (TRL), first as a Research Engineer and, in 1988-1997, as a Project Leader. He also taught and supervised graduate students at Monash University in 1990-2001. During 1997-2008, he was with The University of Melbourne, Victoria, Australia. In 2008 he joined City University of Hong Kong as a Chair Professor of Information Engineering, and a team leader. He has over 300 publications in scientific journals and conference proceedings. He has served on various editorial boards such as Computer Networks, IEEE Communications Magazine, IEEE Journal of Selected Areas in Communications, IEEE/ACM Transactions on Networking and Computer Communications.