IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 4, 2016, accepted November 11, 2016, date of publication November 17, 2016,
date of current version November 28, 2016.

*Digital Object Identifier 10.1109/ACCESS.2016.2629278*

# Joint Optimization of Service Function Chaining and Resource Allocation in Network Function Virtualization

**LUHAN WANG[1,2], ZHAOMING LU[1,2], XIANGMING WEN[1], RAYMOND KNOPP[3], AND ROHIT GUPTA[3]**

[1]Beijing Advanced Innovation Center for Future Internet Technology, Beijing 100876, China
[2]Beijing Key Laboratory of Network System Architecture and Convergence, Beijing University of Posts and Telecommunications, Beijing 100876, China
[3]EURECOM, Biot 06410, France

Corresponding author: L. Wang (wluhan@bupt.edu.cn)

This work was supported in part by the Beijing Municipal Science and Technology Commission research under Project D151100000115002, and in part by China Ministry of Education-CMCC research under Project ''5G Core Network Prototype Research and Develop''.

**ABSTRACT** Network function virtualization (NFV) has already been a new paradigm for network architectures. By migrating NFs from dedicated hardware to virtualization platform, NFV can effectively improve the flexibility to deploy and manage service function chains (SFCs). However, resource allocation for requested SFC in NFV-based infrastructures is not trivial as it mainly consists of three phases: virtual network functions (VNFs) chain composition, VNFs forwarding graph embedding, and VNFs scheduling. The decision of these three phases can be mutually dependent, which also makes it a tough task. Therefore, a coordinated approach is studied in this paper to jointly optimize NFV resource allocation in these three phases. We apply a general cost model to consider both network costs and service performance. The coordinate NFV-RA is formulated as a mixed-integer linear programming, and a heuristic-based algorithm (JoraNFV) is proposed to get the near optimal solution. To make the coordinated NFV-RA more tractable, JoraNFV is divided into two sub-algorithms, one-hop optimal traffic scheduling and a multi-path greedy algorithm for VNF chain composition and VNF forwarding graph embedding. Last, extensive simulations are performed to evaluate the performance of JoraNFV, and results have shown that JoraNFV can get a solution within 1.25 times of the optimal solution with reasonable execution time, which indicates that JoraNFV can be used for online NFV planning.

**INDEX TERMS** NFV, resource allocation, service function chain, traffic scheduling, virtual function placement.

## I. INTRODUCTION

Service function chain (SFC) is defined as a sequence of network functions (NF) that should be traversed by a given service flow in a predefined order [1]. In order to offer various services and have more control on specific traffics, enterprise networks and telecom operators usually deploy different NFs to handle users' traffic. There are various policies to setup service function chains for different services. For example, an online video meeting service traffic may traverse transcoding server, flow distributing server in sequence, and an HTTP service may need to pass through proxies, DPI, firewalls. However, with the emergence of diverse categories of online services, current service chain management is facing great challenges. Traditionally, NFs are deployed on dedicated hardware, which are generally vendor specific, integrated and expensive. Besides, it is often difficult to dynamically scale their capabilities or add new functions to the existing devices.

Network Function Virtualization (NFV) is a new network architecture concept being standardized by ETSI in a joint effort [2]. By leveraging virtualization technologies and high-performance commodity hardware, NFV decouples the software implementation of NFs from dedicated hardware. Under the paradigm of NFV, service function chain is managed as a set of chained software instances that programmed to play a role of particular virtual network functions (VNF)

on virtualization platforms. This makes it possible to dynamically allocate virtualization resources, i.e., CPU, memory, storage, to each VNF according to the requested service traffic, and targeted service function chains can also be deployed flexibly based on geography or customer sets. Standardization organizations have already been working on introducing NFV into 5G networks. NGMN (Next Generation Mobile Networks) discusses the NFV solution in virtualized mobile core network [3]. 3GPP (The 3rd Generation Partnership Projec) SA5 has been committed to start work on NFV E2E management solutions in Release 14 [4], which is deemed as a release for pre-5G.

Despite the promising advantages, there are still many issues that need to be tackled in NFV enabled SFC orchestrating. One of the major challenges to deploy SFC is to achieve fast, efficient, and scalable composition and resource allocation for VNFs. This problem can be referred as NFV resource allocation (NFV-RA). Juliver G. Herrera et al. in [5] highlight three major phases in service function chaining: (1) VNFs Chain composition (VNF-CC), (2) VNF Forwarding Graph Embedding (VNF-FGE) and (3) VNFs Scheduling (VNFs-SCH). For more details, readers could refer to [5], we summarize the phases as followings: Firstly, VNF-CC mainly deals with problems that how many of each VNF to deploy and what's their order. Secondly, VNF-FGE deals with the problem that where to place the VNFs in the network infrastructure in a suitable way to both optimize the network cost and guarantee the services' requirement. Thirdly, VNF-SCH deals with scheduling of VNFs' execution time so as to minimize the total execution time of services. The above scenario of VNF-SCH is based on the assumption that virtualized server can only execute one VNF at the same time. However, with more light-weighted virtualization technologies, such as docker and LXC (Linux container) [6], it is possible to execute a large number of virtual network functions on the same node simultaneously.

Many works have already been done to study NFV resource allocation on each aspect, where [7]–[9] study the VNF embedding and placement problem, and [10], [11] study the scheduling problem in NFV. Another work in [12] studies the joint topology design and SFC mapping in NFV. Two major challenges can be summarized from current works. The first one is the coordination of NFV-RA phases. Though there are three phases, they are actually mutually depended, for example the VNF-FGE will always have an impact on the VNFT-SCH, and vice versa. Most of current works only focus on one or two phases in NFV-RA, and use others as input parameters. The second one is online planning. It can be noticed that evolution cycle of service is getting much shorter than before, it is even counted with weeks and days for now. NFV-RA should response timely to service traffic variation. MIVNF is a coordinate approach presented in [13]. However, it takes more than 12 hours to deploy VNFs on a 30-nodes topology [14].

To address the above challenges, a coordinated NFV-RA is studied in this work, to **J**ointly **O**ptimize the **R**esource **A**llocation in NFV (JoraNFV). The contributions of this work can be concluded as followings:

(1) A comprehensive cost model is incorporated to account for capital cost, operating cost and link cost. The coordinated NFV-RA is formulated as a mixed-integer linear programming (MILP). And we propose a heuristic based two-stage approach to get the near optimal solution.

(2) To make the coordinate NFV-RA more tractable, we propose to perform a one-hop scheduling (OneHop-SCH) while deploying VNFs, instead of scheduling after all VNFs are deployed. OneHop-SCH is formulated as a linear programming (LP) problem, which can be easily solved to get optimal traffic scheduling among adjacent VNF instances.

(3) A multi-path greedy algorithm (MPG) is proposed to solve VNF-CC and VNF-FGE phases in NFV-RA. In MPG, multiple candidate paths are searched and updated simultaneously to avoid trapped in local optima.

(4) Lastly, extensive simulations are carried out to evaluate the performance of JoraNFV. The results show that JoraNFV can always find solutions within 1.25 times of the optimal in reasonable execution time, which is acceptable in on-line planning.

The rest of this paper is organized as follows: Some related works are presented in section II. Scenario overview and system modeling are introduced in section II, the algorithms are presented in details in section IV. The proposed solution is evaluated and discussed in section V, and lastly in section VI, we give a brief conclusion of our work.

## II. RELATED WORKS

Some works in NFV resource allocation emerge in very recent years. As comprehensive surveys are already given in [5] and [15], we only give a short summary of related works in this section.

VNFs chaining composing is deemed as the first stage of NFV-RA. S. Mehraghdam *et al*. in [13] firstly provide a model to formalizing the chaining of network functions, and formulate the network functions mapping as a mixed integer quadratically constraint programming. Lastly, a Pareto analysis on a small size network with 12 nodes is given to show that it is possible to find a placement that optimizes all metrics. In [16], it's assumed that the service function chain is requested implicitly, and an approach is presented to create service plane via network service headers.

The second stage is VNF forwarding graph embedding, which has been widely studied in a rich of works [7], [17]–[21]. VNF-FGE can also be referred as middlebox/network function placement. The VNF-FGE is proved to be an NP-hard problem. Many of existing works formulate it as a mixed integer linear programming and apply a heuristic method to solve this kind of problems. In [7], the authors study two heuristic algorithms to optimize middlebox placement problem, one is a Greedy algorithm and the other

one is a Simulated Annealing algorithm. Their simulation and testbed results show that the Simulated Annealing algorithm outperforms the Greedy algorithm. References [19]–[21] use heuristic algorithms to solve the placement problem in different scenarios, such as SDN and data center networks. Some context-aware placement problems are also studied. T. Taleb *et al*. in [18] study VNF placement algorithms in virtual 5G network, with the goal of minimizing the path length and optimize the sessions mobility.

VNF scheduling can refer to execution scheduling and traffic scheduling among VNF instances. R. Mijumbi *et al*. propose greedy and metaheuristic approaches to tackle the VNF execution time scheduling in online VNF-FGE and VNF-SCH [11]. M. Mechtri *et al*. propose an eigendecomposition based approach to perform VNF embedding and traffic steering for requested VNF forwarding graph [22]. In CoordVNF [14], the authors study the coordinated allocation of service function chains and the traffic scheduling problem, such as traffic splitting and merging. In their work, the VNF chain composition is not predetermined, the algorithm will dynamically determine how to compose the SFC and how many instances should a VNF be deployed. The scenario in our work is quite similar with the one in CoordVNF, so in this work, we mainly use CoordVNF as a comparison algorithm.

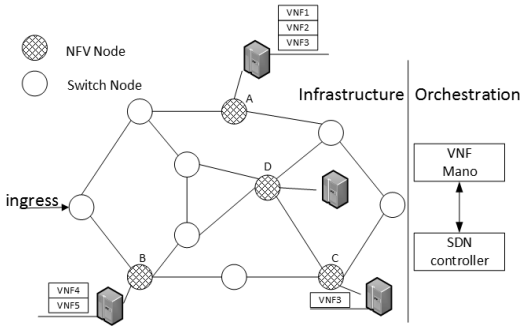


**FIGURE 1.** Basic network scenario in NFV resource allocation.

## III. SYSTEM MODELING

### A. SCENARIO OVERVIEW

Fig. 1 depicts the basic network scenario in NFV resource allocation. The left part is the network infrastructures. They mainly consist of switch nodes and NFV nodes, among which VNFs can be hosted on NFV nodes. By isolation, multiple VNFs can run on the same NFV node, e.g. three VNFs on node *A*. And due to the limited resources on each node, one VNF may also need to distribute on different nodes, e.g. VNF3 instances are located on node *A* and *C*. On the right part of Fig. 1, there is the orchestration part, as it is a simplified model, we only consider the SDN controller and VNF MANO. SDN controller can manage traffic according to bandwidth consumption, delay, etc. And the controller can also report network status information to VNF MANO for resource allocation. VNF MANO is broken up into three functional blocks, NFV orchestrator, VNF manager and virtualized infrastructure manager. With these three blocks, NFV MANO is responsible for VNF chaining, global resource management, and VNF lifecycle management. Thus NFV resource allocation can be performed on NFV MANO according to resource information and service request, the main goal of NFV MANO is to guarantee the service level agreement (SLA) while maximizing the global network performance, e.g. reducing CAPEX and OPEX, promoting network throughput.

### B. NETWORK FUNCTION

Network Functions refer to the network entities that perform specific functions. There can be many types of NFs in the operators' or enterprise network. We use $f_x$ to represent the VNF $x$, and let $F$ denote the set of network functions, $F = \{f_x | x = 1, 2 \ldots, X\}$. Different from dedicated hardware, VNF can be deployed in virtual machine running on the virtualization platform. It would require a certain amount of resources to run a VNF, such as CPU, memory, disk, etc., and the amount of required resources are often relevant with the traffic volume that passing through it. In this work, we assume it is a linear relationship between the traffic volume and required resources, and use $\overrightarrow{\beta}_x$ to present the coefficients for different NFs. If the traffic data rate handled by a VNF is $t$, and the resources required by the VNF are $\overrightarrow{R}_x = \overrightarrow{\beta}_x \cdot t$. $\overrightarrow{R}_x$ is a vector that presents the CPU, memory, disk, bandwidth required by one VNF.

VNFs can modify the traversing network flows in different ways [13]. For example, a load balancer can split the incoming flows into different branches. A video transcoder can change the encoding of the video, which might result in a higher or lower data rate. In our work, we consider that the load balancing can be performed by assigning different flow tables from SDN controller. So in the proposed model, we only take the later behavior into consideration, which is VNF can modify data rate of traffic flow. The data rate scaling ratio of each VNF is defined as $\eta_{f_x} = \frac{t_{output}}{t_{input}}$, $t_{input}$ and $t_{output}$ represent the input data rate and output data rate of VNF.

### C. SERVICE FUNCTION CHAIN

Service function chain is a sequence of network functions that a given service flow needs to traverse. We suppose there are K service functions chains totally, and SFC set can be represented as $S = \{s_k | k = 1, 2 \ldots, K\}$. Each service chain consists of several types of NFs, $s_k = \{f_1^k, f_2^k \ldots, f_M^k | f_m^k \in F\}$. SFC description language has been studied in [13], the authors define a context-free language for formalizing the chaining requests, which could provide the information about service chain start/end point, the order of NFs, the original data rate request, etc. We apply this description language in our work.

For example, Table 1 gives a simple example of a service request. For simplicity, we assume service traffic comes from one single *ingress*, and ends at the *egress*. The *ingress* and *egress* are located at the switch nodes. With requested data

**TABLE 1. An example of service function chain request.**

| | |
|---|---|
| Ingress | := node1 |
| Orders | := Firewall, VideoTranscoder, Proxy |
| Request Data Rate | := 10Gbps |
| Egress | :=node109 |

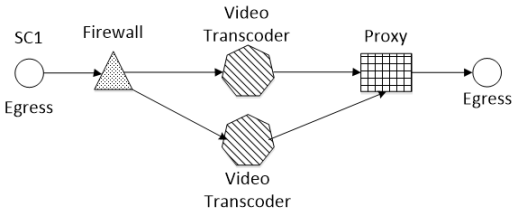


**FIGURE 2. An example of service function chaining.**

rate $t_k$ and the data rate scaling ratio $\eta$, the data rate on each VNF can be obtained. Let $t_m^k$ represent traffic data rate of each VNF in service chain $s_k$. However, in actual deployment, multiple instances of one VNF can be deployed on different VNF nodes, as shown in Fig. 2. We introduce a decision variable $p_m^k$ to present the amount of VNF $f_m^k$ instances in SFC $s_k$. And use $f_{m,i}^k$ to denote the $i^{th}$ instance of VNF $f_m^k$. Another decision variable $\tau_{m,i,j}^k$ is introduced to present the data rate of traffic flow scheduled from $f_{m,i}^k$ to $f_{m+1,j}^k$.

### D. PHYSICAL NETWORKS

The physical networks are usually a set of node and links between them, among which there are some locations that can be used to deploy the VNFs. We represent the nodes and the links between them as an undirected graph $\overline{G} = (\overline{E}, \overline{V})$, where $\overline{E}$ and $\overline{V}$ denote the set of links and nodes, respectively. And we also define an abstract graph $G = (E, V)$, where $V = \{v_n | n = 1, 2 \ldots, N\}$, a subset of $\overline{V}$, denotes the NFV nodes. Actually, there might be multiple paths between two NFV nodes, so we use $E$ to denote the shortest path set between NFV nodes. Each node has a certain amount of resources to host VNFs. The resource capacity of the NFV node $v_n$ is defined as $\overrightarrow{R}_{v_n}^{phy}$. $y_{m,(i,n)}^k$ is a binary value, denoting whether $f_{m,i}^k$ is located on node $v_n$.

$$y_{m,(i,n)}^k = \begin{cases} 1 & if\ f_{m,i}^k\ is\ located\ on\ node v_n, \\ 0 & otherwise. \end{cases} \tag{1}$$

### E. COST MODELING

The factors that need to be considered in NFV resource allocation are mainly costs and the service performances. Generally, deploying more VNF instances, usually bring better service performance, however, it would also bring more capital and operating costs. On the contrary, deploying fewer VNF instances might depreciate the services performance, which will potentially decrease its future revenue. So in order to evaluate the total cost, we divide the cost into three parts: capital expenditures, operating expenditures, and link costs.

#### 1) CAPITAL EXPENDITURES

It will need a certain amount of cost to deploy one VNF instance, considering the standby energy cost and license cost [23]. We take this this kind of costs as the capital expenditure. The cost for each kind of VNF is $\gamma_{f_x}$. So the capital expenditures of each service chain can be denoted as:

$$C_{capex} = \sum_{f_m \in s_k} p_m^k \gamma_{f_m} \tag{2}$$

#### 2) OPERATING EXPENDITURES

Similar to the assumption in Section II.A, we also assume there is a linear relationship between energy consumption and traffic volume, and the energy consumption coefficient for each type VNF is $\varsigma_{f_x}$. So the operating expenditures can be denoted as:

$$C_{opex} = \sum_{f_m \in s_k} \varsigma_{f_m} t_m^k \tag{3}$$

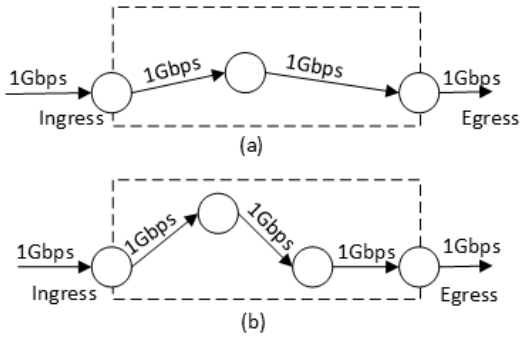


**FIGURE 3. An example of different bandwidth costs.**

#### 3) LINK COST

Link costs are divided into bandwidth cost and delay cost. It is intuitive that bandwidth is kind of link cost. The bandwidth consumption is determined by the forwarding times of a given traffic flow. For example, in Fig. 3(a), the traffic is forwarded one time, and link bandwidth consumption is 2Gbps, while in Fig. 3(b), the traffic is forwarded twice, and the link bandwidth consumption is 3Gbps. We can compute the cost of link bandwidth consumption as:

$$C_{bandwidth} = \sum_{f_m^k \in s_k} \sum_{i=1}^{p_m^k} \sum_{j=1}^{p_{m+1}^k} \tau_{m,i,j}^k h_{f_{m,i}^k, f_{m+1,j}^k} \delta \tag{4}$$

Where $h_{f_{m,i}^k f_{m,j}^k}$ is the number of hops between these two NFs, and $\delta$ is the cost of transmitting one unit traffic through one link in the network.

End-to-end delay is a kind service performance metric, bad service performance usually leads to low service revenue. We can treat delay as a penalty to service revenue, so it is reasonable to deem the delay as a kind of cost. Though precise

relationship between end-to-end delay and penalty should be modeled by investigating historic marketing statistics, we can also assume that its a linear relationship without loss of generality. The penalty factor is denoted as $\xi$, and thus the service delay penalty can be calculated:

$$C_{delay} = \sum_{f_m^k \in s_k} \sum_{i=1}^{p_m^k} \sum_{j=1}^{p_{m+1}^k} \tau_{m,i,j}^k d_{f_{m,i}^k, f_{m+1,j}^k} \xi \tag{5}$$

Where $d_{f_{m,i}^k, f_{m+1,j}^k}$ is the delay between VNF instance $f_{m,i}^k$ and $f_{m+1,j}^k$.

So the link cost can be obtained:

$$C_{link} = C_{bandwidth} + C_{delay} \tag{6}$$

The goal of NFV resource allocation is to minimize the total cost:

$$C_{total} = C_{link} + C_{capex} + C_{opex} \tag{7}$$

So the problem of NFV resource allocation can be formulated as:

$$\min \ C_{link} + C_{capex} + C_{opex}$$
$$s.t. \begin{cases} C1: \sum_{n=1}^{N} \sum_{i=1}^{p_m^k} y_{m,(i,n)}^k \geq 1, \quad f_m \in s_k \\ C2: \sum_{n=1}^{N} \sum_{i=1}^{p_m^k} y_{m,(i,n)}^k \leq N, \quad f_m \in s_k \\ C3: \sum_{f_m \in s_k} \sum_{i=1}^{p_m^k} y_{m,(i,n)}^k t_{m,i}^k \beta_m \leq R_{v_n}^{phy}, \quad v_n \in V \\ C4: \sum_{i=1}^{p_m^k} t_{m,i}^k \geq t_m^k, \quad f_m \in s_k \\ C5: t_{m+1,j}^k \geq \sum_{i=1}^{p_m^k} \tau_{m,i,j}^k, \quad f_m, f_{m+1} \in s_k \\ C6: \sum_{i=1}^{p_{m+1}^k} t_{m+1,i}^k \geq \eta_m \sum_{i=1}^{p_m^k} t_{m,i}^k, \quad f_m, f_{m+1} \in s_k \end{cases} \tag{8}$$

Constraint C1, C2 ensure that each VNF in $s_k$ should be deployed at least on one node, and no more than $N$ nodes. C3 ensures that the total resource consumption of VNFs on one node should not exceed its resource capacity. C4 ensures that total traffic processing capacity should be equal or larger than requested traffic volume. Actually, for sake of reducing cost, $\sum_{i=1}^{p_m^k} t_{m,i}^k$ is equal to $t_m^k$ in SFC. C5, C6 ensures that the traffic processing capacity of VNF $f_{m+1}^k$ instances should be able to deal with the traffic from upstream VNF instances. In the above formulation, $p_m^k$, $y_{m,(i,n)}^k$, $t_{m,i}^k$, $\tau_{m,i,j}^k$ are decision variables, which corresponding to the three stages in NFV-RA, accordingly. Many literatures have proven that NFV resource allocation is an NP-hard problem [7], [24], it is difficult to get the optimal solution within polynomial time, considering with the network size.

## IV. COORDINATED ALGORITHMS

In this section, the coordinated approach JoraNFV is studied to get the near optimal solution of NFV-RA. We firstly present the one-hop optimal scheduling (OneHop-SCH) in VNFT-SCH stage, and then present the multi-path greedy algorithm(MPG).

### A. OneHop-SCH

Scheduling traffic on a pre-deployed NFV network is also a very important issue in NFV-RA. As studied in [11], traffic scheduling is also an NP-hard problem, it is very time-consuming to get the near optimal scheduling scheme even with some heuristic algorithms, such Tabu search, etc., let alone it is just a sub-problem of NFV-RA.

In this part, we focus on one-hop optimal traffic scheduling between adjacent VNF instances. The one-hop traffic scheduling is formulated as a linear programming problem. Though it cannot guarantee the optimal solution for global traffic scheduling, its meaningful due to two reasons, one is that one-hop scheduling performance can be used as a metric to indicate the candidate VNF deployment is good or not, more solutions will be searched in MPG phase. The other reason is that one-hop based scheduling can be formulated as a linear programming problem, which can be solved with many low complexity algorithms and get a deterministic solution with only a few steps.

**Algorithm 1** OneHop-SCH

**Input:** $N_{src}$, $T_{src}$, $N_{dst}$, $C_{link}$, $R_{left}$
**Output:** Cost $C_{total}$, Scheduling scheme $\Theta$, Resource $R_{left}$

1: Let
$$t_i^{left} = t_i,\ r_j \in R_{left},\ \tau_{ij} = 0 \in \Theta,\ C_{total} = 0$$
2: **if** $\beta \sum_{i=1}^{I} t_i^{left} > \sum_{j=1}^{J} r_j$ **then**
3: $C_{total} = \inf,\ \Theta = \emptyset$
4: **else**
5: Formulated as a LP problem according to (9).
6: Solve with Simplex algorithm
7: Update $\Theta$ with each $\tau_{ij}$
8: Update $R_{left}$: $r_j = r_j - \beta_{m+1} \sum_{i=1}^{I} \eta_m \tau_{ij}$
9: **end if**
10: **return** $C_{total}$, $\Theta$, $R_{left}$

The basic steps of OneHop-SCH are given in Algorithm 1. The input parameters include nodes set hosting source VNF instances $N_{src} = \{n_i | i = 1, 2 \ldots, I\}$, traffic on each source VNF instance $T_{src}$=$\{t_i | i = 1, 2 \ldots, I\}$, nodes set hosting destination VNF instances $N_{dst}\{n_j | j = 1, 2 \ldots, J\}$, link cost between source and destination VNF instance $C_{link}\{c_{ij} | i = 1, 2 \ldots, I; j = 1, 2 \ldots, J\}$, resources left on destination nodes $R_{left} = \{r_j | j = 1, 2 \ldots, J\}$. Firstly, we need to check whether the resources on destination nodes can cope with the traffic from upstream VNF instances. If the answer is no,

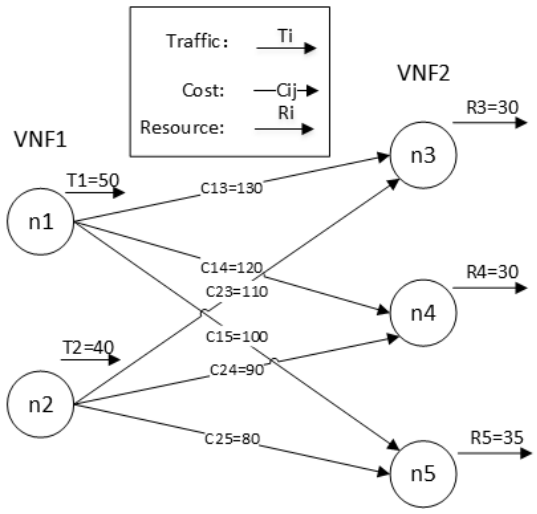


**FIGURE 4.** One Hop Traffic Scheduling.

then the total cost is set to inf, indicating that the scheduling cannot be satisfied. Otherwise one-hop traffic scheduling is formulated as a linear programming problem. A simple example is give in Fig. 4. VNF1 instances are deployed on node $n_1$ and $n_2$, with 50 units and 40 units traffic on each instances accordingly, these traffic need to be scheduled to 3 VNF2 instances. The transmission delay and forwarding times vary on each link, and yield different link costs. The target of OneHop-SCH is to minimize the total link costs of transmitting traffic between adjacent VNF instances, $\tau_{ij}$ is the decision variable indicating the traffic allocating between source and destination VNF instances, so the problem can be formulated as:

$$\min \sum_{i=1}^{I}\sum_{j=1}^{J} \tau_{ij} c_{ij}$$
$$s.t. \begin{cases} C1: \sum_{j=1}^{J} \tau_{ij} \geq \eta_m t_i, & i \in \{1, 2 \ldots I\} \\ C2: \beta_{m+1} \sum_{i=1}^{I} \eta_m \tau_{ij} \leq r_j, & j \in \{1, 2 \ldots, J\} \end{cases} \quad (9)$$

Where constraint C1 ensures that all traffic from each source VNF instance is allocated, and C2 ensures that the resources on destination node $n_j$ can handle all allocated traffic. Due to the fact that number of variables and constraints are always not very large in one hop traffic scheduling, many algorithms can be applied to get the optimal solution efficiently, for example, simplex algorithm is used in our experiments [25].

### B. MULTI-PATH GREEDY

Generally, the basic idea of greedy algorithms is to iteratively choose the best solution for each step. Some greedy based algorithms are studied in [7] and [23], it can be noticed that greedy approaches can usually get a good solution, but compared with other heuristic algorithms greedy tends to be trapped in a local optimal solution. However, greedy approaches are usually very fast, and can be used in on-line planning. Considering this merit, we develop a multi-path greedy algorithm for NFV-RA. Different from the general greedy approach, in the initialization stage of MPG, multiple available paths are generated, and paths are updated according to greedy manner while deploying each VNF. Instead of determining VNF-CC and VNF-FGE in two independent steps, we evaluate the determination of VNF-CC and VNF-FGE in one shot for each VNF in service chain. Algorithm 2 gives the details of the proposed MPG algorithm.

**Algorithm 2** Multi-Path Greedy

**Input:** $s_k, t, \overline{G}, R, n_i, n_e, MB$
**Output:** $\psi_i, \theta_i, C^i_{total}$
**Initial:** Deploying first VNF $f^k_1$
Generate placement schemes $\phi^i_1$ for first VNF $f^k_1$, make each path $\psi_i$ start with $\phi^i_1$.
Let $C^i_{total} = 0$
OneHop-SH between $n_i$ and $f^k_1$ on each path $\psi_i$
**for** each $\phi^i_1$ **do**
  $(C^i_1, \theta^i_1, R^i_{left})$
    $= OneHop\_SCH(n_i, t, \phi^i_1, C_{link}, R)$
**end for**
**for** NF $f^k_m \in s_k, m \geq 2$ **do**
  **for** $iter = 1 : MB$ **do**
    **for** Path $\psi^i \in \Psi$ **do**
  $(C^i_m, \phi^i_m, \theta^i_m, R^i_{left})$
    $= SelectPlacement(\phi^i_{m-1}, \theta^i_{m-1}, \overline{G}, R^i_{left}, f^k_m, iter)$
    **end for**
    **if** $\min(C^i_m) <$ inf, $i = 1, 2, 3 \ldots$ **then**
      $C^i_{total} = C^i_{total} + C^i_m$
      Remove $\psi^i$ with $C_total^i = inf$ from $\Psi$
      Update $\psi^i$ with $\phi^i_m$, $\psi^i \in \Psi$
      Update $\theta^i$ with $\theta^i_m$, $\psi^i \in \Psi$
      **Break**
    **else**
      **Continue**
    **end if**
  **end for**
**end for**
**if** $\min(C^i_{total}) \neq$ inf **and** $\Psi \neq \emptyset$ **then**
  **return** $\psi^i, \theta^i, C^i_{total}$
**else**
  Allocation Failed
**end if**
**End**

The input parameters include: requested service function chain $s_k$, which contains a set of ordered VNFs, the request traffic data rate $t$ on the first VNF, network topology $\overline{G}$, node resources set $R$, ingress node $n_i$ and egress node $n_e$, and also the maximum balancing number $MB$, which indicates the maximum number of instances that each VNF can be deployed. Other symbols used in the algorithm are given in Table 2.

**TABLE 2. Parameters used in MPG.**

| | |
|---|---|
| $\psi^i$ | The $i^{th}$ candidate path |
| $\theta^i$ | Traffic scheduling scheme on $\psi^i$ |
| $\phi_m^i$ | The possible placement scheme set of $f_m$ on path $\psi^i$ |
| $\psi_m^i$ | The accepted placement scheme of VNF $f_m$ on path $\psi^i$, $\psi_m^i \in \phi_m^i$ |
| $\theta_m^i$ | The accepted scheduling scheme of VNF $f_m$ on path $\psi^i$ |
| $\theta_m^{i,j}$ | The traffic scheduling scheme on $j^{th}$ candidate scheme of $f_m$ on path $\psi^i$ |
| $\phi_m^{i,j}$ | The generated $j^{th}$ scheme to deploy $f_m$ on path $\psi^i$, $m \neq 1$ |
| $\phi_1^i$ | The generated $i^{th}$ placement of $f_1$, and make it the start scheme of each $\psi^i$ |
| $\theta_1^i$ | The traffic scheduling scheme for $i^{th}$ placement of $f_1$ |

In the initial stage, the first VNF $f_1$ is deployed, and multiple paths are generated with different deployment schemes of VNF $f_1$. $\psi^i$, $\theta^i$ is set to $\emptyset$, the possible of placement scheme $\phi_1^i$ of $f_1$ is searched in just-enough manner, which means that if we can find solutions to deploy $x$ instances to process the requested traffic on $f_1$, we don't try to deploy $x+1$ instances. And the amount of candidate paths is set to number of $f_1^k$ placement schemes. Each $f_1$ placement scheme is added to each path $\psi^i$. And one hop scheduling can be performed by OneHop-SCH, then $\psi^i$,$\theta^i$ are updated. In this stage, every candidate path is kept.

In next stage, each VNF is iteratively deployed. A sub-algorithm named *SelectPlacement* is introduced to deploy each VNF on different paths. The main role of *SelectPlacement* is to select the best placement scheme of VNF $f_m^k$ instances on path $\psi_i$, and return the traffic scheduling for this placement. In *SelectPlacement*, the candidate placement schemes for *iter* instances of $f_m^k$ are generated. Then OneHop-SCH is applied to compute the cost and traffic scheduling of each candidate scheme. The placement with least cost will be selected and returned, along with the traffic scheduling scheme $\theta$ and cost $C$.

While deploying each VNF, the multi-path greedy algorithm will try to deploy VNF from 1 instance to *MB* instances iteratively on each path. After each iteration, if the minimum cost of candidate paths is less than inf, which means at least one path succeeds to deploy current VNF instances, the candidate path with $C = \text{inf}$ will be removed from $\Psi$, and then break out of this loop. Otherwise, it means this VNF can't be deployed with only *iter* instances, and then algorithm will try to deploy $iter + 1$ in next iteration.

When all VNF are deployed, the path with least cost will be selected, and meanwhile, the scheduling is also determined. If the set $\Psi$ is empty, it indicates the requested service function chain can not be deployed on current network.

### C. DISCUSSION ABOUT THE IMPLEMENTATION ISSUES

In the previous section, we present the details of JoraNFV, as we deploy VNF instances in a just-enough manner, from the definition in equation (2) and (3), the capital cost and operating cost will not vary among different deployment and scheduling schemes. It only matters when considering

**Algorithm 3** Select Placement

**Input:** $\psi^i$, $\theta^i$, $\overline{G}$, $R_{left}^i$, $f_m^k$, $iter$
**Output:** Cost $C$, Placement $\phi$, Scheduling $\theta$, Resource $R_{left}$

```
Generate J placement schemes φ_m^{i,j} on Ḡ for f_m^k
φ_m^i = nchoosek(G, iter), φ_m^{i,j} ∈ φ_m^i
J = size(φ_m^i)
for φ_m^{i,j} ∈ φ_m^i do
    Calculating C_link according to (6)
    Perform OneHop-SCH
    (C_m^{i,j}, θ_m^{i,j}, R_left^j)
        = OneHop_SCH(θ_{m-1}^i, ψ_{m-1}^i, φ_m^{i,j}, C_link, R_left)
end for
if min(C_m^{i,j}(j)) ≠ inf then
    ĵ = arg min_j(C_m^{i,j})
    C = C_m^{i,ĵ}, φ = φ_m^{i,ĵ}, θ = θ_m^{i,ĵ}, R_left = R_left^ĵ
else
    C = inf, φ = ∅, θ = ∅, R_left = R_left^i
end if
return C, φ, θ
```

deploy multiple service chains. So in practical deployment of the proposed algorithms, the deployment manager should check whether the requested VNF is deployed or not when a service chain is requested. If there have already existed such VNF instances, the NFV MANO should firstly scale up the instances capacity according to the traffic request, which would not cause extra capital costs.

The other implementation issue is the convergence before last VNF is deployed, which means that all the candidate paths may converge to one path before last VNF is deployed. For example, if there only exists one placement scheme $\phi_m^{x,1}$ for VNF $f_m^k$ on each path and the placement scheme on each path is exactly the same, then all paths will converge to one, even though there might be different deployment schemes for the remaining VNFs. And this could lead to bad performance. We propose that an on-path selection can be applied while deploying service chain. When path convergence on VNF $f_k$ is detected, the SFC deployment should be paused, select the best path from the candidate paths up to now and save it as the *best-partial-path*. Then VNF $f_k$ is taken as the ingress of the left VNFs, and the deployment progress will repeat from the initial step. After all VNFs are deployed, all *best-partial-path* are connected together, and finally, the resource allocation scheme is obtained.

## V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of JoraNFV. As only a few coordinated NFV-RA can be found in current literatures. We compare our approach with CoordVNF [14] and a simulated annealing based approach.

In CoordVNF, the authors provide a heuristic method to coordinate the composition of VNF chains and their embedding into subnet network. The idea behind CoordVNF is that

deploy each VNF on the node with the lowest cost, if no node have enough resource to host this VNF, then split the traffic into multiple branches, and use branch with the lowest cost firstly. If in even worse case, the total resources on all remaining nodes are not enough for hosting the NF, then backtrack the deployment one VNF by one VNF. Actually, it is like a Water Filling approach that always use the best position at each step, and schedule the overflowed traffic to the second optimal position.

The simulated annealing (SA) based middle box placement is studied in [7]. In order to evaluate the performance of MPG, we replace MPG with SA in VNF-CC and VNF-FGE phases. As each VNF only needs to deploy one instance in [7], we make some modifications to adapt it to our scenario. (1) **Temperature**: Other than defining *Temperature* with end-to-end delay, we define *Temperature* with the total cost. (2) **Neighbour functions**: Given one SFC deployment, the neighbour function is defined as: randomly select one VNF, and switch to a new randomly placement scheme. Then update the traffic scheduling with OneHop-SCH for current VNF and following VNFs. Other definitions and conditions are aligned with [7].

**TABLE 3.** Network and service simulation parameters.

| Parameters | Distribution | Mean | Var |
|---|---|---|---|
| Link Delay(ms) | $Uniform$ | 1.5 | 0.34 |
| Node Resources(unit) | $Uniform$ | 30 | 31.8 |
| Request Traffic(unit) | $5, 10, 20$ | N/A | N/A |
| Num of NFs in SFC | $3, 4, 5$ | N/A | N/A |

**TABLE 4.** Virtual service function parameters.

| | VNF1 | VNF2 | VNF3 | VNF4 | VNF5 |
|---|---|---|---|---|---|
| Resource Requirement $\beta_m$ (unit/unit) | 1 | 1 | 2 | 1.5 | 1.8 |
| Traffic Scale $\eta_m$ | 1.1 | 1.2 | 1.5 | 1.3 | 2 |

### A. SIMULATION SETUP

We carry out our simulations on some public available network topologies from Topology-zoo, generated random networks, and Fat-tree networks. Topology zoo provides a set internet topologies collected from all over the world [26]. The random networks are generated with principle proposed in [27], which is that the edge probability between a pair of nodes depends on the distance between them. First two types of topologies can represent typical topology of internet and ISP networks. And FatTree usually represents the network with layered structure and it is usually used in data centers [28]. Other simulation parameters are randomly generated according to Table 3. The delay on each link and resources on each node are generated following a uniform distribution. For the resources on NFV nodes, we don't explicitly differentiate the types of resources (CPU, Memory, Disk), just use one single value to represent the resource capacity on each node. 5 kinds of VNF are used in this simulation, the parameters are given in Table 4. We assume the number of VNFs in each SC can be 3, 4, 5. Simulations are carried out on a Laptop with four 2.4GHz CPU cores and 12GB memories using Matlab 2015b.

**TABLE 5.** Cost factors.

| Factors | NF $f_x$ static cost | NF $f_x$ operating cost | Bandwidth cost | Delay Penalty |
|---|---|---|---|---|
| Value | 10 | 1 | 1 | 1 |

### B. COST PERFORMANCE

According to equation (7), the overall cost consists of capital expenditure, operating expenditure, and link cost. In order to make these costs in a comparable value range, we set cost coefficients as in Table 5. To compare the performance of ratio to optimal, we use CPLEX to get the optimal value only on small size networks, as it takes too much time to get an optimal value on a larger size network [24]. We choose four topologies from Topology Zoo, AboveNet (23 nodes), AGIS (25 nodes), ChinaTelecom (25 nodes), Cernet (41 nodes); four random networks with 20, 30, 40, 50 nodes, and three Fat-tree networks with 20, 40, 60 (FatTree-4 is applied). It is assumed that all nodes can be used to deploy VNFs. Multiple service chains are generated according to Table 3. The NFV resource allocation is performed 100 times on each topology category.

The results are plotted in Fig. 5. We can observe that SA-based approach slightly outperforms JoraNFV, but we can say these two algorithms are comparable. Both JoraNFV and SA-based approach outperform CoordVNF in three types of topologies. This is mainly due to two reasons. Firstly, both JoraNFV and SA-based approach are more likely to recover from local optima. In JoraNFV, multiple paths are searched from the first VNF and on-path selection can also make it easier to recover from local optima. In SA-based approach, it will try to search even more combinations of different VNF placement schemes. Second reason is the optimal traffic scheduling between two adjacent NFs. In the CoordVNF, traffic always goes to the VNF instance with least cost, however, this might lead to bad performance. And we can also notice that in the FatTree topology networks, the CoordVNF performs a bit better than the other two scenarios, we argue that this is because that FatTree is a more structured and layered topology with few shortcuts and random connections, the local optimal solution usually generate a rather good global performance. However, the topology characters have few impacts on the result of the other two approaches. Among three scenarios, JoraNFV can provide the solutions within 1.25 times of the optimal solution.

### C. EXECUTION TIME

To evaluate the execution time, we use JoraNFV, CoordVNF and SA-based approach to deploy service function chain on different size random networks with 3 level of traffic volumes. The requested service chain contains 5 VNFs, in the

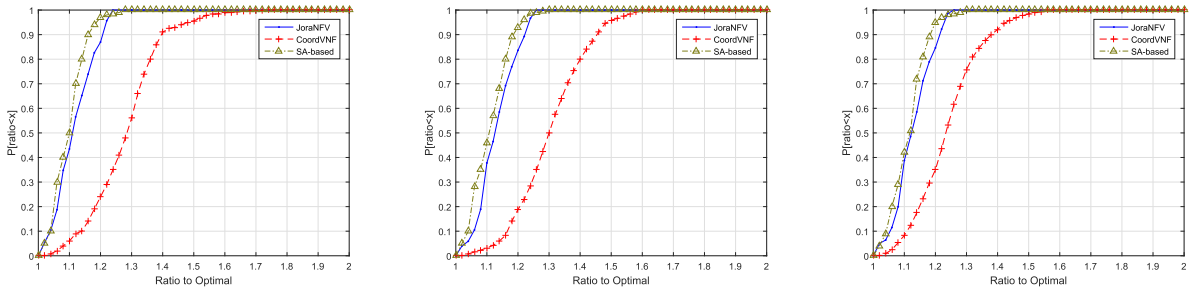


**FIGURE 5.** CDF of cost performance on different topologies. (a) Topologies from Topology-Zoo. (b) Random Topologies. FatTree-4 Topologies.

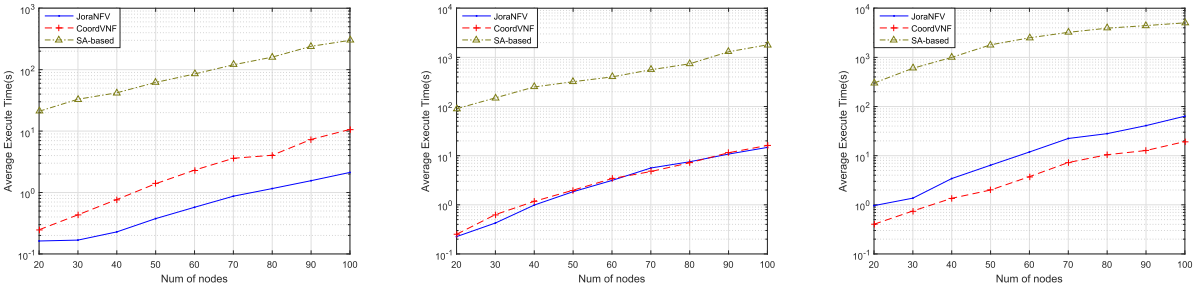


**FIGURE 6.** Execution time performance on different requested traffic. (a) Requested Traffic 5 units. (b) Requested Traffic 10 units. (c) Requested Traffic 10 units.

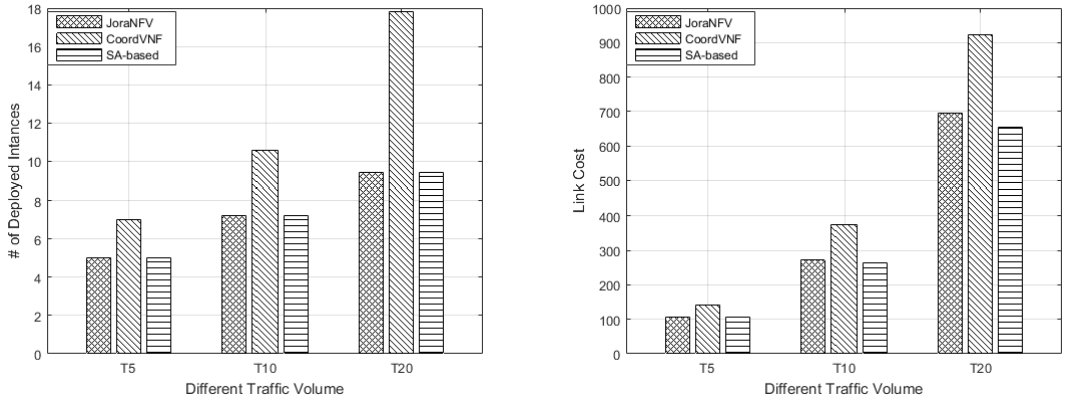


**FIGURE 7.** (a)Number of deployed instances on different requested traffic; (b) Link cost on different requested traffic.

order of $\{f_1, f_2, f_3, f_4, f_5\}$. The requested traffic data rate is set to 5, 10, 20 units, which are denoted as $T5, T10, T20$ accordingly. And the number of nodes in network is set from 20 to 100 with an interval of 10.

The results are shown in Fig. 6. The SA-based algorithm takes much more time than JoraNFV and CoordVNF. It can also be noticed that the requested traffic volume has different impacts on JoraNFV and CoordVNF. When requested traffic is $T5$, our approach outperforms CoordVNF. In this scenario, the resources needed by each VNF are quite few, nodes that have enough resource to deploy VNFs can always be found. Thus in JoraNFV, only one instance is deployed for each VNF. However, CoordVNF always deploys the VNF to node with lowest link cost, which might cause traffic splitting while deploying VNF. And this would cause extra time consumption. The traffic volume $T10$ and $T20$ represent the scenario that at least one VNF need to deploy two instances, and the scenario that at least one VNF need to deploy three instances, accordingly. The execution time of JoraNFV increases faster than CoordVNF, this is because that when VNF do need to deploy multiple instances, the search space of each VNF deployment becomes the main factor of execution time. For

JoraNFV, the search space for each VNF is $C_N^{p_m}$, which represents the number of combinations to choose nodes from nodes, so for each VNF the execution time is proportional to $N^{p_m}$. However, for CoordVNF the search space is always $N$ nodes for each instance, for each VNF the execution time is proportional to $p_m N$.

We also plot the number of deployed instances and the link cost in Fig. 7. It shows that, considering link cost and deployed instances, JoraNFV and SA-based approach perform consistently on different requested traffic volumes. The cost-performance of CoordVNF becomes worse when larger volume traffic is requested, and more instances are deployed, which also make resources more fragmentized.

## VI. CONCLUSION

In this work, we propose a coordinated approach JoraNFV to jointly optimization three steps in NFV resource allocation. In JoraNFV, one-hop traffic scheduling among adjacent VNF instances is performed instead of scheduling traffic after all VNF instances are deployed. OneHop-SCH could make it more tractable to perform coordinated NFV-RA. And besides, the OneHop-SCH can be formulated as a linear programming problem, and thus a variety of methods can be applied to solve the problem. Then an MPG algorithm is proposed to jointly determine the VNF-CC and VNF-FGE. Combining with OneHop-SCH, MPG search multiple candidate paths simultaneously with greedy manner. Lastly, JoraNFV is evaluated and compared with CoorVNF. The results show that JoraNFV can usually get the solution within 1.25 times of the optimal solution. JoraNFV outperforms CoordVNF considering the cost-performance. When it comes to execution time, JoraNFV is also comparable with CoordVNF, however, we also discuss that this is because the search space for each VNF placement becomes much larger when the requested traffic increases. So in our future work, we would like to investigate some heuristic approaches to reduce the execution time for finding optimal solution of each VNF deployment when large volume traffic service function chain is requested.

## ACKNOWLEDGMENT

The authors would like to thank EURECOM for their support.

## REFERENCES

[1] P. Quinn and T. Nadeau, "Problem statement for service function chaining," IETF, Fremont, CA, USA Tech. Rep. RFC 7498, 2015.

[2] ETSI. *Network Functions Irtualisation Introductory White Paper*, accessed on May 4, 2016. [Online]. Available: https://portal.etsi.org/NFV/NFV_White_Paper.pdf/

[3] NGMN. *5G White Paper*, accessed on Sep. 14, 2016. [Online]. Available: http://www.ngmn.org/5g-white-paper.html

[4] 3GPP. *Management of Mobile Networks That Include Virtualized Network Function*, accessed on Oct. 15, 2015. [Online]. Available: http://www.3gpp.org/news-events/3gpp-news/1738-sa5_nfv_study

[5] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[6] R. Cziva, S. Jouet, K. J. S. White, and D. P. Pezaros, "Container-based network function virtualization for software-defined networks," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2015, pp. 415–420.

[7] J. Liu, Y. Li, Y. Zhang, L. Su, and D. Jin, "Improve service chaining performance with optimized middlebox placement," *IEEE Trans. Serv. Comput.*, to be published.

[8] S. Herker, X. An, W. Kiess, S. Beker, and A. Kirstaedter, "Data-center architecture impacts on virtualized network functions service chain embedding with high availability requirements," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2015, pp. 1–7.

[9] 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 *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 98–106.

[10] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Trans. Commun.*, vol. 64, no. 9, pp. 3746–3758, Sep. 2016.

[11] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *Proc. IEEE Conf. Netw. Softw. (NetSoft)*, Apr. 2015, pp. 1–9.

[12] Z. Ye, X. Cao, J. Wang, H. Yu, and C. Qiao, "Joint topology design and mapping of service function chains for efficient, scalable, and reliable network functions virtualization," *IEEE Netw.*, vol. 30, no. 3, pp. 81–87, May 2016.

[13] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proc. IEEE 3rd Int. Conf. Cloud Netw. (CloudNet)*, Oct. 2014, pp. 7–13.

[14] M. T. Beck and J. F. Botero, "Coordinated allocation of service function chains," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2015, pp. 1–6.

[15] J. Garay, J. Matias, J. Unzilla, and E. Jacob, "Service description in the NFV revolution: Trends, challenges and a way forward," *IEEE Commun. Mag.*, vol. 54, no. 3, pp. 68–74, Mar. 2016.

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

[17] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 1346–1354.

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

[19] S. Lange *et al.*, "Heuristic approaches to the controller placement problem in large scale SDN networks," *IEEE Trans. Netw. Service Manage.*, vol. 12, no. 1, pp. 4–17, Mar. 2015.

[20] A. Mohammadkhan, S. Ghapani, G. Liu, W. Zhang, K. K. Ramakrishnan, and T. Wood, "Virtual function placement and traffic steering in flexible and dynamic software defined networks," in *Proc. 21st IEEE Int. Workshop Local Metropolitan Area Netw.*, Apr. 2015, pp. 1–6.

[21] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for NFV chaining in packet/optical datacenters," *J. Lightw. Technol.*, vol. 33, no. 8, pp. 1565–1570, Apr. 15, 2015.

[22] M. Mechtri, C. Ghribi, and D. Zeghlache, "A scalable algorithm for the placement of service function chains," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 3, pp. 533–546, Sep. 2016.

[23] M. Bouet, J. Leguay, and V. Conan, "Cost-based placement of vDPI functions in NFV infrastructures," in *Proc. 1st IEEE Conf. Netw. Softw. (NetSoft)*, Apr. 2015, pp. 1–9.

[24] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Trans. Netw. Service Manage.*, to be published.

[25] T. G. Robertazzi, *Mathematical Programming for Planning*. Hoboken, NJ, USA: Wiley, 1999, pp. 13–41. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5263909

[26] University of Adelaide. *The Internet Topology Zoo*, accessed on Mar. 4, 2016. [Online]. Available: http://topology-zoo.org//

[27] B. M. Waxman, "Routing of multipoint connections," *IEEE J. Sel. Areas Commun.*, vol. 6, no. 9, pp. 1617–1622, Dec. 1988.

[28] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 63–74, 2008.

**LUHAN WANG** received the B.S. degree in communication engineering from Shandong University, Jinan, China. He is currently pursuing the Ph.D. degree in communications engineering with the Beijing University of Posts and Telecommunications, China. His current research interests include network architecture, network function virtualization, and soft-defined networks.

**ZHAOMING LU** received the Ph.D. degree from the Beijing University of Posts and Telecommunications in 2012. He joined the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, in 2012. His research includes open wireless networks, QoE management in wireless networks, software defined wireless networks, and cross-layer design for mobile video applications.

**XIANGMING WEN** received the B.E., M.S., and Ph.D. degrees in electrical engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China. He is currently the Vice-President of BUPT, where he is also a Professor with the Communication Network Center and the Director of the Beijing Key Laboratory of Network System Architecture and Convergence. He is also the Vice-Director of the Organization Committee of the China Telecommunication Association. He has authored over 100 papers. He is the Principle Investigator of over 18 projects, including the National Key Project of Hi-Tech Research and Development Program of China (863 Program) and the National Natural Science Foundation of China. His current research is focused on broadband mobile communication theory, multimedia communications, and information processing.

**RAYMOND KNOPP** received the B.Eng. (Hons.) and the M.Eng. degrees in electrical engineering from McGill University, Montreal, QC, Canada, in 1992 and 1993, respectively, and the Ph.D. degree in communication systems from the Swiss Federal Institute of Technology (EPFL), Lausanne, in 1997. He is currently a Professor with the Mobile Communications Department, EURECOM. From 1997 to 2000 he was a Research Associate with the Mobile Communications Laboratory, Communication Systems Department, EPFL. His current research and teaching interests are in the area of digital communications, software radio architectures, and implementation aspects of signal processing systems and real-time wireless networking protocols. He has a proven track record in managing both fundamental and experimental research projects at an international level and is also General Secretary of the OpenAirInterface.org open-source wireless radio platform initiative which aims to bridge the gap between cutting-edge theoretical advances in wireless communications and practical designs.

**ROHIT GUPTA** received the master's degree from Nanyang Technological University, Singapore, in 2003, and the Ph.D. degree in wireless communication from the University of Washington in 2009. From 2003 to 2005, he was a DSP System Design Engineer with STM Microelectronics. From 2010 to 2012, he was a Post-Doctoral Researcher with CEA, France. From 2012 to 2015, he was a Staff Engineer with National Instruments. He is currently a Development Manager of OpenAirInterface Software Alliance with EURECOM, where he is also responsible for managing Software Development Tools within the OpenAirInterface Software Alliance. His research interests include wireless communication, software radio architectures design and implementation, and wireless network virtualization.