sciendo

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **23**, No 1
Sofia • 2023 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2023-0003

APVNFC: Adaptive Placement of Virtual Network Function Chains

Brajesh Kumar Umrao, Dharmendar Kumar Yadav

Department of Ccomputer Science and Enginnering, Motilal Nehru National Institute of Technology Allahabad, Prayagraj, India
E-mails: brajeshumrao84@gmail.com dky@mnnit.ac.in

Abstract**:** *Designing efficient and flexible approaches for placement of Virtual Network Function* (*VNF*) *chains is the main success of Network Function Virtualization* (*NFV*). *However, most current work considers the constant bandwidth and flow processing requirements while deploying the VNFs in the network. The constant* (*immutable*) *flow processing and bandwidth requirements become critical limitations in an NFV-enabled network with highly dynamic traffic flow. Therefore, bandwidth requirements and available resources of the Point-of-Presence* (*PoP*) *in the network change constantly. We present an adaptive model for placing VNF chains to overcome this limitation. At the same time, the proposed model minimizes the number of changes* (*i.e., re-allocation of VNFs*) *in the network. The experimental evaluation shows that the adaptive model can deliver stable network services. Moreover, it reduces the significant number of changes in the network and ensures flow performance.*

Keywords**:** *Virtual Network Function, Network Function Virtualization, Point-of-Presence* (*PoP*), *Traffic Flow, Flow Processing.*

1. Introduction

Network Function Virtualization (NFV) is a promising technology that replaces hardware-based network devices with computer programs (software-based network functions). These computer programs can run on Commercial-Off-The-Shelf (COTS) servers using virtualization technology. A computer program running on a COTS server using virtualization technology is known as Virtual Network Function (VNF). The replacement of hardware-based network devices with computer programs brings several benefits: (i) it reduces the network operational expenses; (ii) it minimizes the maintenance cost by a cheaper update of the network functions than the expensive hardware upgrade; (iii) it provides flexible placement of VNF chains across the geographically distributed network [1].

NFV has made progress on several fronts, from the design and placement of VNFs [2-4] to the management and orchestration of network services (VNF chains) [5-8]. Despite the progress made on different facets of NFV, several research

opportunities still need to be explored. VNF placement and chaining are one of them. In this problem, we study how to determine the places for locating the VNFs in a given set of PoPs and how to route the network traffic among them according to the given description in Service Function Chaining (SFC) [9]. Software-Defined Networking (SDN) can realize traffic routing [10], enabling the flexible placement of VNFs and traffic routing. The problem complexity depends on the constraints and requirements that must be satisfied during the placement and chaining, such as the computing power of the PoPs (the place where the network functions will be deployed), end-to-end delay, and bandwidth between the PoPs. It has been proven that the VNF placement and chaining problem is NP-hard [11, 12]. This problem has been studied widely with different objectives, such as maximization of the network resource utilization and minimization of the network operational cost [2-4, 12-15].

When deploying a traffic request set, the existing VNF placement and chaining approaches consider the fixed and immutable VNF operational cost and available resources on PoPs. However, the changes in the available resources of PoPs and the operating cost of VNFs depend on the number of network requests [16]. The network requests are highly dynamic with respect to time. Therefore, constant operational cost and available resources on PoPs is the main limitation of the VNF placement and chaining problem. According to SFC specifications, traffic processing requirements may be violated during peak hours. The existing approaches mitigate this issue by observing the behavior of individual VNFs and instantiating the more number of VNFs while increasing the network requests. The individual search in the solution space for local solutions may lead to a different solution to balance the demand and supply for the VNF placement and chaining problem. Moreover, it may increase the waste of resources due to the failure to explore the idle processing capacity in VNFs/PoPs.

To overcome the above-mentioned limitation, we propose an adaptive model for orchestrating the VNFs. This model allows the service provider to (re)arrange the previously allocated network functions to deal with the dynamic character of requests and fluctuating resources in the PoPs. For this, the adaptive model (re)chains the traffic requests through VNFs with available computing power and bandwidth. Moreover, this model (re)allocates the VNFs on the PoPs with more available resources. In this article, we extend the research work [12] by providing: (i) the exhaustive discussion on an Integer Linear Programming (ILP) formulation that ensures the best provisioning of the network requests consideration with the dynamic changes in the demand and available network resources (physical or virtual); (ii) detailed evaluation of the ILP model in terms of effectiveness.

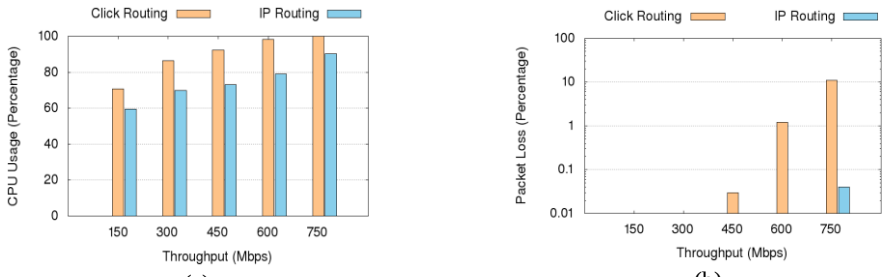
The remaining part of the article is organized in the following way: Section 2 comprises experimental evidence showing how VNFs perform due to the varying workload; Section 3 consists of an ILP formulation of adaptive provisioning of VNFs; Section 4 comprises an architecture for adaptive VNF chain placement; Section 5 includes the experimental results with evolution scenarios; Section 6 comprises the existing work related to our proposed work; Section 7 consists of the final consideration with some light on the future direction.

2. Effect of network traffic on VNF

In the context of the adaptive VNF provisioning model, it is to understand the VNF performance with the varying network loads. To illustrate the network function performance, we have experimented with different traffic patterns in the NFV-enabled network. Finally, we summarize the finding of the experiments in the form of CPU utilization, throughput, and packet loss.

We have conducted the experiments in an environment with two servers, A and B. Each server is equipped with Intel Xeon CPU E5-2623 (8 cores with 3.00 GHz), 16 GB RAM, 1 Gbps Network Interface Card (NIC), 1 TB hard disk, and Ubuntu 18.04 LTS. NIC of server A physically connected to the NIC of server B through the LAN wire. We have configured the Open vSwitch and KVM hypervisor on top of Ubuntu 18.04 on both servers. On top of the KVM hypervisor on server A, we have built a Virtual Machine (VM) with two vCPU, 1 GB RAM, and two logical interfaces. The virtual interfaces of the VM are associated with a virtual switch, which is logically associated with the physical NIC. Similarly, we have configured two VMs on top of the KVM hypervisor on server B. Each interface of the VM is associated with the virtual switch, and the virtual switch is logically associated with the physical NIC of server B.

The description of the experimental setup is as follows. The first VM of server B is configured as an Iperf client in UDP mode, whereas the second VM of server B is configured as an Iperf server in UDP mode. VM of server A receives data packets from one interface and forwards these data packets to another interface. At the time of the experiment, the Iperf client (running on server B) generates the traffic (data packets), and these data packets come back to the Iperf server (running on server B) via VM (running on server A). In this experiment, VM configured on server A worked as a traffic forwarder. We have chosen this arrangement so that the traffic generation process will not affect the performance of VMs because our objective is to measure the performance of the VMs. In addition to that, we have carried out two different experiments with the bellow mentioned configurations: (i) VM with static routing and (ii) VM with routing functionality executing on Click Router [17]. For each experiment, we have evaluated the CPU usage by VM, host operating system, and other processes which are associated with this investigation. The experimental outcomes are an average of 20 executions.



(a) (b)

Fig. 1. Metrics measurement and their relationship with service chain: throughput vs CPU usage (a); throughput vs packet loss (b)

Fig. 1 illustrates that packet loss occurs when CPU usage approaches 100%. Predicting the starting point when the network function degrades the performance is possible. For example, the packet loss rate is approximately 0.06% when CPU usage is 95%, and throughput is 500 Mbps – the packet loss rate increases to 10% for 100% CPU usage and 750 Mbps throughput. We have observed that the additional overhead occurs with routing as a network function on Click. As a result, packet loss increases exponentially as CPU usage increases by 100%. We have observed that no significant variation occurs in memory usage.

It is essential to mention that higher throughput can be obtained using packet processing optimization technologies such as Data Plane Development Kit (DPDK) [18] and Single Root I/O Virtualization (SR-IOV) [19]. These packet optimization techniques can shift the bottlenecks to points other than those depicted in the graph, but it would still happen. In brief, the above result and discussion strengthen the need for an adaptive approach proposed in this article.

3. Adaptive VNF provisioning model

To deal with the fluctuating nature of network traffic and rearrange the VNF allocation without degrading the network performance or wasting the physical resources of the network. It is necessary to review the available formulations and heuristics for the VNF placement problem. To this end, we have selected an adaptive version of the ILP formulation proposed by B a r i et al. [12]. They have formulated the static placement and chaining of VNFs by applying constraints in a linear system. We present an adaptive ILP formulation for the placement and chaining of virtual network functions.

3.1. Formal model

Input. The ILP model presented by B a r i et al. [12] considers a batch of traffic requests T and a geographically distributed network $G = (V, L)$ where V and L are a set of nodes and links, respectively. Assume that the x86 servers are located at the different locations within the network and the VNFs are placed on these servers. The positions of servers within a network are called the Point-of-Presences (PoPs). The number of available servers in a network can be represented by S. There is a binary variable $h_{sv} \in \{0, 1\}$ that defines a server $s \in S$ is associated to a switch (node) $v \in V$. The computing capacity of any arbitary server $s \in S$ is represented by c_s. The bandwidth of a physiacl link $(u, v) \in L$ is represented by B_{uv} and the propagation delay of a physical link $(u, v) \in L$ is represented by D_{uv}. There is a function $\eta(u)$ that return a set of adjacent node for the node u.

(1) $$\eta(u) = \{v|(u, v) \in L\}, \text{ where } u, v \in V.$$

A network service is formed with the various combinations of the different types of VNFs. There is a symbol P to denote the possible types of VNFs. Each type of VNF $p \in P$ has following constraints. The required computing resource of a VNF type $p \in P$ is represented by k_p. Moreover, processing capacity of a VNF type $p \in P$ is represented by C_p, whereas processing delay of a VNF type $p \in P$ is represented by D_p. The actual values of a VNF characteristic depend on many parameters. To

simplify the proposed model, we choose the constant values for the VNF characteristic mentioned above. Here, we consider that VNF is a piece of software that can be deployed on any server.

A network traffic $t \in T$ can be represented by 5-tuple $t = \langle u^t, v^t, \mu^t, B^t, D^t \rangle$, where $u^t \in V$ is a source and $v^t \in V$ is a destination. A sequential order of VNFs (i.e., proxy → firewall → IDS) that has to traverse by a traffic $t \in T$ is denoted by μ^t. The bandwidth demand of a traffic $t \in T$ is denoted by B^t whereas, the maximum acceptable delay of a traffic $t \in T$ is denoted by D^t. In this model, sequential order of VNF μ^t can be represented by a directed graph $G^t = (V^t, L^t)$, where V^t represents the traffic nodes and L^t denotes the links between the nodes. Here, we define a function $\eta^t(u_1)$ to obtain the adjacent node of $u_1 \in V^t$:

(2) $$\eta^t(u_1) = \{u_2 | (u_1, u_2) \in L^t\}, \text{ where } u_1, u_2 \in V^t.$$

We consider that $u_1 < u_2$ iff u_1 appears before u_2 in a directed acyclic graph. Also, we define a binary variable q_{up}^t to denote the type of node u is p in a network traffic t:

(3) $$q_{up}^t = \begin{cases} 1 \text{ if node } u \in V^t \text{ is of type } p \in P, \\ 0 \text{ otherwise.} \end{cases}$$

Enumeration of VNFs. To simplify the solution of the VNF placement problem, we enumerate all the VNFs in the network by computing the number of VNFs of each type that can be hosted on the servers. The number of hosted VNFs on a server can be computed by dividing the server resources to the required resources of VNF type. For example, a server consists of 16 CPU cores and the required CPU cores by proxy and IDS is 4 and 8 CPU cores, respectively. Therefore, 4 proxy and 2 IDS can be deployed on the server. The enumerated VNFs in the network a denoted by a symbol M. Each VNF $m \in M$ is associated to a perticular server $s \in S$. To express this relationship, we have a function $\sigma(m)$,

(4) $$\sigma(m) = s \text{ if VNF } m \text{ is associated to server } s.$$

We also have another function $\Omega(s)$ that represent the reverse association,

(5) $$\Omega(s) = \{m | \sigma(m) = s\}, \text{where } m \in M \text{ and } s \in S.$$

Now, we define a variable $d_{mp} \in \{0, 1\}$ that represent the type of a VNF:

(6) $$d_{mp} = \begin{cases} 1 \text{ if VNF } m \in M \text{ is type of } p \in P, \\ 0 \text{ otherwise.} \end{cases}$$

There is a function $\lambda(m)$ that returns the VNF type m,

(7) $$\lambda(m) = \{p | d_{mp} = 1\}, \text{ where } p \in P \text{ and } m \in M.$$

We have a binary variable x_m={0, 1} to indicate the VNF m is active or not,

(8) $$x_m = \begin{cases} 1 \text{ if VNF } m \text{ is active,} \\ 0 \text{ otherwise.} \end{cases}$$

Output. ILP model is formulated using a set of binary variables that are described in this section. The binary variable y_{um}^t represents the mapping of node node u of traffic t to the VNF m:

(9) $$y_{um}^t = \begin{cases} 1 \text{ if traffic node } u \in V^t \text{ is provisioned on a VNF } m \in M, \\ 0 \text{ otherwise.} \end{cases}$$

The binary variable $\hat{y}_{um}^t$ represents that the current placement of node u of traffic t has changed to its previous placement.

Now, we have another variable that determines the mapping of a traffic node to the node in the physical network:

(10) $z_{uv}^{t} = \begin{cases} 1 \text{ if traffic node } u \in V^t \text{ is provisioned on node } v \in V \text{ of the physical network,} \\ 0 \text{ otherwise,} \end{cases}$

z_{uv}^{t} is can be derived from the y_{um}^{t}: $z_{uv}^{t} = 1$ if $y_{um}^{t} = 1$ and $h_{\sigma(m)v} = 1$.

Similarly, the variable $\hat{z}_{uv}^{t}$ indicates that current allocation of a traffic node has changed to its previous allocation.

Finally, we define a decision variable w_{uv}^{tij} that represents the placement of network traffic link (i, j) on the physical network link (u, v):

(11) $w_{uv}^{tij} = \begin{cases} 1 \text{ if network traffic link } (i,j) \in L^t \text{ map on the physical network link } (u,v), \\ 0 \text{ otherwise.} \end{cases}$

The decision variable $\hat{w}_{uv}^{tij}$ indicates that placement of a network traffic link has changed to its previous placement.

3.2. Objective function and constraints

The proposed model comprises a multi-objective function that minimizes the following: (i) network resource consumption (server resources (PoPs), physical links, and VNFs), and (ii) modifications in the placement due to variation in the demand of allocated network requests (i.e., reassignment of network request, and provisioning of new network functions). The objective function consists of two components. The first component minimizes resource consumption. The resource consumption is reduced by minimizing the allocated VNFs (denoted by z_{uv}^{t}) and links (denoted by w_{uv}^{tij}). The second component of the objective function consists of two parts. The first part of the second component minimizes the changes made in the previously allocated VNFs, whereas second part of the second component minimizes the changes made in the previously allocated traffic links. Here, α and β are the weighted factors for setting the relative importance.

Objective:

$$\text{Minimize} \left(\sum_{t \in T} \sum_{u \in V^t} \sum_{v \in V} z_{uv}^{t} + \sum_{t \in T} \sum_{(i,j) \in L^t} \sum_{(u,v) \in L} w_{uv}^{tij} \right) - \left(\alpha \sum_{t \in T} \sum_{u \in V^t} \sum_{v \in V} \hat{z}_{uv}^{t} + \beta \sum_{t \in T} \sum_{(i,j) \in L^t} \sum_{(u,v) \in L} \hat{w}_{uv}^{tij} \right)$$

Subject to:

(12) $\sum_{m \in \Omega(s)} x_m \times k_m \leq c_s, \ \forall s \in S,$

(13) $\sum_{t \in T} \sum_{u \in V^t} y_{um}^{t} \times B^t \leq c_{\lambda(m)}, \forall m \in \{x | x \in M, \ a_x = 1\},$

(14) $y_{um}^{t} \times q_{up}^{t} = d_{mp}, \ \forall t \in T, \ u \in V^t, m \in M, p \in P,$

(15) $\sum_{t \in T} \sum_{u \in V^t} y_{um}^{t} = 1, \quad \forall m \in M,$

(16) $w_{uv}^{tij} + w_{vu}^{tij} \leq 1, \ \forall (i,j) \in \{(c,d) | c \in V^t, \ d \in \lambda^t(c), \ d > c\}, \ u, v \in V,$

(17) $\sum_{u \in V} \sum_{v \in V} (w_{uv}^{tij} + w_{vu}^{tij}) \times B^t \leq B_{uv}, \ \forall t \in T, \forall (i,j) \in \{(c,d) | c \in V^t, \ d \in \eta^t(c), d > c\},$

(18) $\sum_{u \in V} \sum_{v \in V} (w_{uv}^{tij} - w_{vu}^{tij}) = c_{iu}^{t} - c_{ju}^{t}, \ \forall t \in T, (i,j) \in \{(c,d) | c \in V^t, \ d \in \eta^t(c), d > c\},$

(19) $\sum_{u \in V} \sum_{v \in V} (w_{uv}^{tij} + w_{vu}^{tij}) \geq 1, \forall t \in T, \forall (i,j) \in \{(c,d) | c \in V^t, \ d \in \eta^t(c), d > c\}.$

The set of constraints mentioned above constructs the model described below. The first four constraints are related to the resource limitation of the network infrastructure. Equation (12) ensures that the total provisioned computational resources to all VNFs on a server will be, at most, the available resources in the server. Equations (13) provides that the amount of passing traffic through a VNF will stay within the functional capacity of the VNF. Equation (14) is responsible for mapping a traffic node to an appropriate type of VNF. Equation (15) ensures the mapping of each traffic node to exactly one VNF.

Equation (16) to (19) describes the chaining constraints of the network traffic requests. Each traffic link is mapped only to the single direction on the physical network link. Equation (16) satisfies the single-direction link mapping constraints to the link of the physical network. The traffic bandwidth demand is, at most, the physical link on which it is mapped. Equation (17) is responsible for satisfying the bandwidth constraints of the physical path. The flow conservation constraints must meet at all intermediate nodes. The incoming and outgoing traffic must be equal at each switch (node) except for the source and destination nodes of the mapped traffic. Equation (18) ensures the flow conservation constraint. Each traffic link is mapped on the physical network in the way they will build a network path. This model consists of Equation (19) to ensure that each traffic link is mapped on the path of the network.

4. Architecture for adaptive VNF chain placement

This section presents the architecture for the placement and orchestration of the VNF chain. This architecture relies on the proposed ILP model, which enables the dynamic allocation of VNFs in response to the changes in the processing requests. It also follows the basic building blocks and interface standards recommended by the ETSI MANO (MANagement and Orchestration).

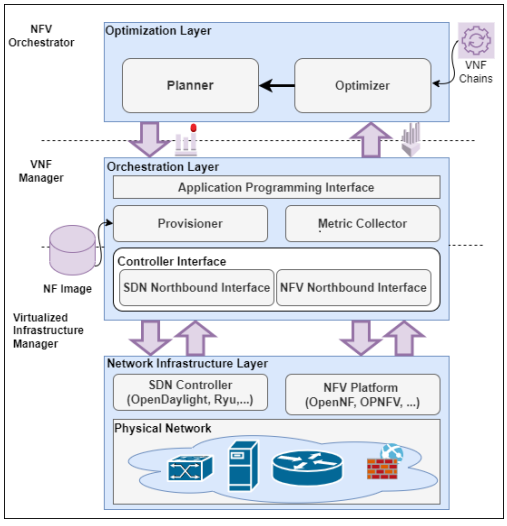


Fig. 2. Overview of the NFV architecture [1]

Fig. 2 demonstrates the architecture for the placement and orchestration of requested network requests. According to the functionalities of architecture, it can be grouped into three layers – Optimization Layer, Orchestration Layer, and Infrastructure Layer.

4.1. Optimization Layer

This layer comprises two modules – Optimizer and Planner. Optimizer and Planner modules work together to optimize and plan the instantiation of VNFs to satisfy the processing required in the network infrastructure. When (re)planning the VNF allocation in the network infrastructure, these modules are processed both deployed and to-be-deployed processing requests.

The optimizer module computes the best feasible locations for allocating the VNFs in the network infrastructure. It considers the deployed and to-be-deployed processing requests along with the current network state (available N-PoPs and link resources). For this work, the Optimizer module incorporates the ILP model discussed in Section 3. The outcome of the Optimizer module is forwarded to the Planner module. The Planner module is responsible for carrying out the necessary changes on VNF placement and their corresponding chaining in the network infrastructure. The objective of this module is to maintain the network state close to optimal with the minimum number of changes. Various strategies can be adopted to ensure the transition between states and the network infrastructure must avoid service disruption [20, 21].

4.2. Orchestration Layer

The Orchestration layer comprises several modules. All modules work together to provide the service chains in the network infrastructure. A Provisioner module is responsible for VNF chains placement according to the service chains mapping received from the Optimization Layer. The next module is the Metric Collector, which monitors the deployed VNFs in the network infrastructure and consolidates the operational statistics of deployed VNFs. Moreover, it gauges the operational states of VNF to determine required reallocations to deal with changes in traffic requests. The consolidated VNF performance measures are forwarded to the Optimization Layer.

Both modules communicate to the controller through Controller Interface to perform the orchestration and monitoring activities of VNFs on the network infrastructure. This interface can further categorize into two sub-modules: SDN northbound interface and NFV northbound interface. The SDN northbound interface translates chain installation requests and state queries to the protocol used by the SDN controller. The NFV northbound interface is used to transform the requests related to the VNFs into the protocol used by the NFV platform of the network infrastructure.

4.3. Infrastructure Layer

The Infrastructure Layer consists of all elements of the physical network. The physical network contains COTS servers, the SDN controller, the NFV platform, and VNFs that can execute on these COTS servers. SDN controller has complete

information about the physical network, including the location of each network component and its connectivity. Both the SDN controller and NFV platform can interact with the physical network using the southbound interface.

5. Evaluation

We have performed a couple of experiments to prove and validate the behavior of the proposed model. The experimental work is carried out in a machine with configuration Intel(R) Core(TM) i7-4790 3.60 GHz processors and 12 GB RAM. We have installed the Ubuntu18.04 operating system.

5.1. Experimental setup and workload

We adopt a similar strategy employed in the state-of-the-art [12] experiment. We use the Internet2 [22] topology as a network infrastructure with 12 nodes (switches) and 15 links. There are seven servers within the network; each server has 16 CPUs. These servers are associated with nodes 1, 3, 4, 5, 9, 10, and 11 of the network infrastructure. Each network link has a uniform bandwidth and delay of 10 Gbps and 10 ms, respectively.

We have four types of VNF images: firewall, proxy, nat, and IDS. These VNF images are chosen randomly to create a network request. Moreover, network requests follow a line topology, and endpoints are selected randomly in a traffic request. The 30 network requests are submitted in a batch to perform the evaluation. The length of each network request is uniform (i.e., three) in a batch.

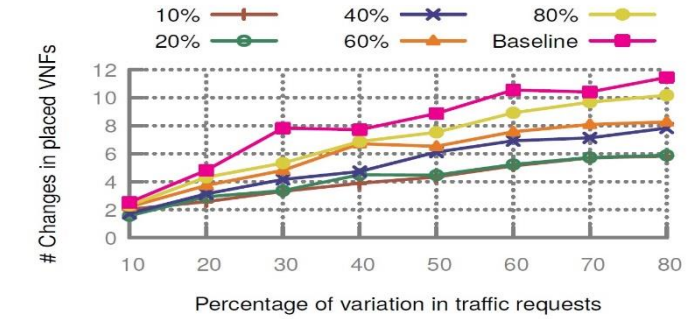
Our analysis focuses on the solution that the optimization model generates. To evaluate the model's capability for re-designing the network with minimum disordering, we have provisioned the number of network requests between the regular and peak-hour workloads. Therefore, re-planning for placing the VNFs is necessary to maintain the stability and performance of the network. We have compared the proposed model with B a r i et al. [12]. If re-planning is required, all network requests are submitted again on the optimization model. The optimization model provides the location for instantiating the VNFs in the network.

5.2. Required changes in the network

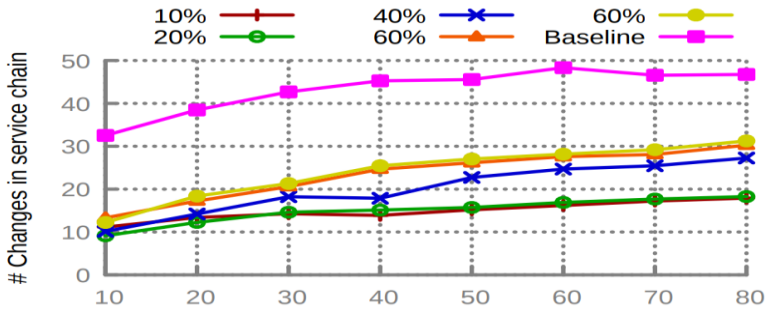
Fig. 3 a shows the number of changes in the VNF locations for some variations in the traffic requests, and Fig. 3b shows the number of changes in the mapping of service chains. The number of network requests varies from 10 to 80% of the total allocated services in the network. Moreover, the demand for network requests surpasses the provisioning capacity from 10 to 80%. Different curves are used to illustrate various scenarios. The values of the relative weighting factors (i.e., α and β) are chosen to be 1 for this experiment. Observe that a constraint does not bind all possible values of these weighting factors since the network characteristics (i.e., load and size) may influence the effectiveness of any setting for them. We leave the analysis of the relationship among the weighting factors for future research.

We have perceived that the count of modifications required (y-axis) to fine-tune the network for the new traffic demand is proportional to the following: (i) the

percentage of service chains with the increased demand, and (ii) the outreached demand values (*x*-axis). Moreover, the number of modifications in the positions of VNFs is significantly lesser than the reassignment of the service chains. This shows that the proposed model is feasible for real-world applications because the time required for programming a routing device is lower than that of the instantiation or migration of VNF. From Fig. 3a, one can observe that the proposed model reduces up to 25% of changes in the VNF placement compared to the baseline. Moreover, the proposed model can reduce more than two times in the VNF chaining compared to the baseline.



(a)



(b)

Fig. 3. Reassignment of VNFs for fluctuating traffic requests (a), reassignment of service chain for fluctuating traffic requests (b)

5.3. Over-commitment effect on the network

We initially have deployed the 40 service chains over the network under the expected workload, increasing the flow processing demand by 40%. The over-commitment value of the VNFs varies from 10% up to 40%. The higher value of over-commitment increases the possibility of performance degradation of some VNFs. Fig. 4 shows the required number of modifications in the network to adjust the service chain in response to the varying network traffic demand. Fig. 4 illustrates that the excessive over-commitment value decreases the network's number of modifications. The 20% over-commitment reduces the 20% overall changes in the network infrastructure compared to the 10% over-commitment.

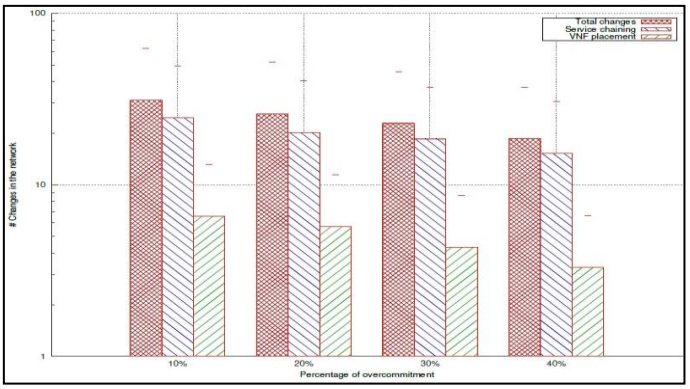


Fig. 4. Effect of over-commitment on the network

5.4. Efficiency in re-planning the service chains

Fig. 5 illustrates the average time required to find the best solution for the service chain re-planning problem. The proposed model can get the result within 5 s for all cases. We have observed that our model solves the problem in lesser time than the case in which all service chains are re-assigned.

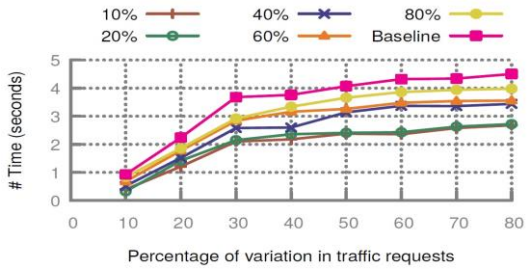


Fig. 5. Average times to re-allocate the service chains

We have observed that the increased number of service chains in the usual case requires more time to discover the solution. Though VNF placement and chaining is an NP-hard problem, these results propound that attaining an exact solution is possible for medium and small-scale problems. But, more research is required to evaluate the computing time limits for the large-scale problem.

6. Related work

The VNF placement and traffic routing problem has attracted many researchers to develop various methods with different objectives. Many researchers have considered the VNF placement problem extension of the Virtual Network Embedding [23, 24]. The network services form a chain of VNFs. The VNF chain can be represented as a directed graph. This graph is mapped over a substrate network to accomplish a network service. More than one VNF can be mapped on a server (PoP), and each graph link is mapped on a physical connection. However, VNF and VNF placement and chaining problems have different objectives and mapping constraints.

The abovementioned problem has been investigated as NP-hard [12, 13, 25-27]. The existing techniques for solving the VNF placement and chaining problem can be categorized as exact, heuristic, and meta-heuristic. The same approach finds all possible solutions for the NP-hard problem. The search space increases exponentially as increase the problem size [12, 13, 27]. Therefore, the exact approach obtains the optimal solution for small and medium-scale problems. Heuristics are problem-dependent, so the heuristic approach is formulated to solve the situation. The heuristic approach determines the solution quickly but compromises the solution optimality [12, 27]. The meta-heuristic techniques are the extension of the heuristic. These approaches offer a problem-independent framework that finds the near-optimal solution by enhancing the solutions in each step concerning quality measures.

We have reviewed the existing exact, heuristic, and meta-heuristic approaches. Most researchers have used Integer Linear Programming (ILP) to solve the VNF placement and chaining problem. Riggio et al. [26] presented an ILP model for jointly solving VNF placement and scheduling problems in the radio access network. Additionally, they design a heuristic algorithm, Wireless Network Embedding (WiNE), to solve the VNF placement and scheduling problem. Bhamare et al. [27] present service chain placement approaches for geographically distributed clouds. The authors have designed an affinity-based allocation algorithm to solve the problem. The authors have presented an ILP model that minimizes inter-cloud traffic and response time in multi-cloud scenarios. Sun et al. [28] formulate an ILP model to solve the VNF placement problem and reduce deployment costs. The authors have designed an affiliation-aware VNF placement heuristic that is time-efficient. Li, Hong and Xue [29] formulate an ILP model for placing the VNFs over cloud data centers. ILP formulation aims to reduce the total resource consumption by minimizing the number of active servers. Moreover, the authors have designed a multi-stage heuristic based on a greedy search and adjustment approach for mapping the service chain requests. Yi, Wang and Huang [30] address the scalable issue in provisioning the requested service chain and present an ILP model. Additionally, they give a backtrack proactive and reactive heuristic for the defined problem.

There are some solutions based on Mixed Integer Linear Programming. Marotta et al. [31] formulate a MILP model for placing VNFs and routing problem considerations with the demand uncertainties and latency constraints. Additionally, they propose a time-efficient heuristic that minimizes the solution complexity. Sang et al. [32] formulate the MILP model and have presented a greedy heuristic to solve the proposed VNF allocation problem. Qu et al. [33] have obtained the optimal solution for the VNF placement and traffic routing problem using MILP formulation. They have developed a heuristic algorithm that uses a greedy approach to get the k-shortest path. The problem objective maximizes the service acceptance rate and minimizes the end-to-end delay. Zhu et al. [34] also have developed the MILP model for the service chain mapping problem. The problem objective maximizes the service acceptance ratio and minimizes resource fragmentation. Mechtri, Ghribi and Zeghlache [35] present an eigendecomposition-based heuristic for jointly solving the distributed cloud's VNF placement and chaining problem.

Some researchers have selected single-objective meta-heuristic approaches to place the VNFs in an NFV-enabled network. Kim et al. [36] present a Genetic Algorithm (GA) based Algorithm for deploying the VNFs in the cloud environment. The problem objective is to minimize the total power consumption and meet the required service latency. Xing et al. [37] propose a Gray Wolf Optimizer (GWO) for placing the VNFs in the NFV-enabled network. The objective of the optimizer is to minimize the end-to-end delay of requested services. Chantre and Fonseca [38] identify the redundant placement of VNFs in NFV-enabled networks. They have selected Particle Swarm Optimization (PSO) to mitigate the problem of redundant VNF placement and minimize the end-to-end delay of requested services. Farshin and Sharifian [39] mix the features of Ant Colony Optimization (ACO) and GWO to solve the VNF placement and traffic routing problem. They have used ACO to create the best routing path, and then VNFs are deployed on servers that fall on the selected route. The service chains are distributed over the different cloudlets. These cloudlets are connected to each network router, so each service chain somewhat utilizes the cloudlet resources. GWO tunes the controllable components of ACO. The mixed meta-heuristic performs better load balancing but suffers from high time complexity.

Lazarov [40] presents a mathematical model for analyzing the effect of malware attacks in computer networks. It suggests applying the different stochastic distributions of malware attacks to generalize the results. Petrosyan and Astsatryan [41] have developed an architecture that enables HPC workloads to be serverless on the cloud. Shoc (Serverless HPC over Cloud) architecture integrates scaling and scheduling policies to deal with Cloud platforms like Azure, Alibaba Cloud, Salesforce, OpenStack, and Amazon Web Services. Bhargavi and Shiva [42] use the Dyna-Q-Learning task scheduling technique to manage behavioral and primary uncertainty task and resource parameters.

Limited research uses multi-objective meta-heuristic approaches for mapping the service chains in the NFV-enabled network. Khebbache, Makhlouf and Zeghlache [43] have presented a Non-dominated Sorting Genetic Algorithm (NSGA) to solve the multi-objective VNF placement problem. The objective of the proposed meta-heuristic is to reduce the total mapping expenses and bandwidth usage. The experimental results demonstrate that the proposed NSGA functions better than some heuristic approaches. Bezerra et al. [44] have formulated virtual network function placement across the distributed Micro-Data Centers (MDCs) as a multi-objective optimization problem. They propose NSGA-II and Generalized Differential Evolution 3 (GDE3) to solve the optimization problem for the cellular network scenarios. The Servicing Gateway (SGW) and Packet Data Gateway (PGW) are the responsible network functions for the data plane in the Long-Term Evolution (LTE) architecture. The experimental results demonstrate that GDE3 can attend to two conflicting objectives (minimize failure and maximize energy consumption), whereas NSGA-II prioritizes energy consumption.

Despite the various advances in VNF placement and chaining, existing solutions do not consider the bottleneck scenario and localized fluctuation due to varying traffic volumes in the network. An ad hoc approach deals with these fluctuations by

executing the VNF placement algorithm again and rearranging the previous placement according to obtained results. Although this ad hoc strategy is effective, it is computationally expensive and needs to allow it to respond effectively to dynamic traffic volumes. The research works in articles [45, 46] are close to an effective solution to this problem. In these papers, the authors combine fuzzy logic [47] and genetic algorithms [48] to solve the virtual network function placement and routing problem with scalable computing capability. These solutions isolate network functions without considering global optimizations, such as traffic routing for high-capacity VNFs with similar requirements.

Considering the limitations mentioned above, this article presents an adaptive model to re-adjust the network infrastructure against the varying network traffic demand by identifying bottlenecks in the flow processing, rearranging the previous placement and chaining of network functions, and aiming to reduce the disruption in the flow processing.

7. Conclusion

In this article, we have studied the VNF chain placement problem. The existing approaches for the VNF placement problem have considered the constant flow processing and bandwidth requirement, which becomes a critical limitation. We present a formal adaptive model for re-adjusting the VNFs and logical links for varying network traffic demands to deal with this limitation. Meantime, the adaptive model reduces the number of changes in the network. The experimental results demonstrate that the proposed adaptive model reduces 25% in re-positioning the VNFs and two times in the VNF chaining.

In the recent future, we will develop a method for traffic prediction and integrate this method into the formal model.

References

1. Yi, B., et al. A Comprehensive Survey of Network Function Virtualization. – Computer Networks, Vol. **133**, 2018, pp. 212-262.
2. Zhang, X., et al. Near-Optimal Energy-Efficient Algorithm for Virtual Network Function Placement. – IEEE Transactions on Cloud Computing, 2019.
3. Yue, Y., et al. Resource Optimization and Delay Guarantee Virtual Network Function Placement for Mapping SFC Requests in Cloud Networks. – IEEE Transactions on Network and Service Management, Vol. **18**, 2021, No 2, pp. 1508-1523.
4. Zahedi, S. R., S. Jamali, P. Bayat. A Power-Efficient and Performance-Aware Online Virtual Network Function Placement in SDN/NFV-Enabled Networks. – Computer Networks, Vol. **205**, 2022, 108753.
5. Tavares, T. N., et al. NIEP: NFV Infrastructure Emulation Platform. – In: Proc. of 32nd IEEE International Conference on Advanced Information Networking and Applications (AINA'18), IEEE, 2018.
6. Castillo-Lema, J., et al. Mininet-NFV: Evolving Mininet with OASIS TOSCA NVF Profiles towards Reproducible NFV Prototyping. – In: Proc. of IEEE Conference on Network Softwarization (NetSoft'19), IEEE, 2019.
7. Chowdhury, S. R., et al. μNF: A Disaggregated Packet Processing Architecture. – In: Proc. of IEEE Conference on Network Softwarization (NetSoft'19), IEEE, 2019.

8. K a u r, K., V. M a n g a t, K. K u m a r. A Review on Virtualized Infrastructure Managers with Management and Orchestration Features in NFV Architecture. – Computer Networks, 2022, 109281.
9. H a l p e r n, J., C. P i g n a t a r o. Service Function Chaining (SFC) Architecture. – RFC, Vol. **7665**, 2015, pp. 1-32.
10. M c K e o w n, N., et al. OpenFlow: Enabling Innovation in Campus Networks. – ACM SIGCOMM Computer Communication Review, Vol. **38**, 2008, No 2, pp. 69-74.
11. S u n, J., et al. A Survey on the Placement of Virtual Network Functions. – Journal of Network and Computer Applications, 2022, 103361.
12. B a r i, F., et al. Orchestrating Virtualized Network Functions. – IEEE Transactions on Network and Service Management, Vol. **13**, 2016, No 4, pp. 725-739.
13. P h a m, C., et al. Traffic-Aware and Energy-Efficient vNF Placement for Service Chaining: Joint Sampling and Matching Approach. – IEEE Transactions on Services Computing, Vol. **13**, 2017, No 1, pp. 172-185.
14. K u o, T.-W., et al. Deploying Chains of Virtual Network Functions: On the Relation between Link and Server Usage. – IEEE/ACM Transactions on Networking, Vol. **26**, 2018, No 4, pp. 1562-1576.
15. A b d e l a a l, M. A., G. A. E b r a h i m, W. R. A n i s. Efficient Placement of Service Function Chains in Cloud Computing Environments. – Electronics, Vol. **10**, 2021, No 3, 323.
16. L u i z e l l i, M. C., et al. The Actual Cost of Software Switching for NFV Chaining. – In: Proc. of IFIP/IEEE Symposium on Integrated Network and Service Management (IM'17), IEEE, 2017.
17. K o h l e r, E., R. M o r r i s, B. C h e n, J. J a n n o t t i, M. F. K a a s h o e k. The Click Modular Router. – ACM Transactions On Computer Systems (TOCS), Vol. **18**, 2000, No 3, pp. 263-297.
18. L i, Z. HPSRouter: A High Performance Software Router Based on DPDK. – In: Proc. of 20th International Conference on Advanced Communication Technology (ICACT'18), IEEE, 2018.
19. P i t a e v, N., et al. Characterizing the Performance of Concurrent Virtualized Network Functions with OVS DPDK, FD. IO VPP and SR-IOV. – In: Proc. of ACM/SPEC International Conference on Performance Engineering, 2018.
20. K a u r, K., V. M a n g a t, K. K u m a r. Towards an Open-Source NFV Management and Orchestration Framework. – In: Proc. of 14th International Conference on COMmunication Systems & NETworkS (COMSNETS'22), IEEE, 2022.
21. C h i u, Y.-S., et al. A Cloud Native Management and Orchestration Framework for 5G End-to-End Network Slicing. – In: Proc. of IEEE International Conference on Service-Oriented System Engineering (SOSE'22), IEEE, 2022.
22. Internet2. Research Network Topology and Traffic Matrix (Online, accessed 26-September-2021). **https://www.cs.utexas.edu/~yzhang/ research /AbileneTM/**
23. S a h h a f, S., et al. Network Service Chaining with Optimized Network Function Embedding Supporting Service Decompositions. – Computer Networks, Vol. **93**, 2015, pp. 492-505.
24. S h a h r i a r, N., et al. Virtual Network Embedding with Guaranteed Connectivity under Multiple Substrates Link Failures. – IEEE Transactions on Communications, Vol. **68**, 2019, No 2, pp. 1025-1043.
25. U m r a o, B. K., D. K. Y a d a v. Algorithms for Functionalities of Virtual Network: A Survey. – The Journal of Supercomputing, Vol. **77**, 2021, No 7, pp. 7368-7439.
26. R i g g i o, R., et al. Virtual Network Functions Orchestration in Wireless Networks. – In: Proc. of 11th International Conference on Network and Service Management (CNSM'15), IEEE, 2015.
27. B h a m a r e, D., et al. Optimal Virtual Network Function Placement in Multi-Cloud Service Function Chaining Architecture. – Computer Communications, Vol. **102**, 2017, pp. 1-16.
28. S u n, Q., et al. Forecast-Assisted NFV Service Chain Deployment Based on Affiliation-Aware vNF Placement. – In: Proc. of IEEE Global Communications Conference (GLOBECOM'16), IEEE, 2016.
29. L i, D., P. H o n g, K. X u e. Virtual Network Function Placement Considering Resource Optimization and SFC Requests in Cloud Datacenter. – IEEE Transactions on Parallel and Distributed Systems, Vol. **29**, 2018, No 7, pp. 1664-1677.

30. Yi, B., X. Wang, M. Huang. Design and Evaluation of Schemes for Provisioning Service Function Chain with Function Scalability. – Journal of Network and Computer Applications, Vol. **93**, 2017, pp. 197-214.
31. Marotta, A., et al. A Fast Robust Optimization-Based Heuristic for the Deployment of Green Virtual Network Functions. – Journal of Network and Computer Applications, Vol. **95**, 2017, pp. 42-53.
32. Sang, Y., et al. Provably Efficient Algorithms for Joint Placement and Allocation of Virtual Network Functions. – In: Proc. of IEEE INFOCOM 2017-IEEE Conference on Computer Communications, IEEE, 2017.
33. Qu, L., et al. A Reliability-Aware Network Service Chain Provisioning with Delay Guarantees in NFV-Enabled Enterprise Datacenter Networks. – IEEE Transactions on Network and Service Management, Vol. **14**, 2017, No 3, pp. 554-568.
34. Zhu, Z., et al. Service Function Chain Mapping with Resource Fragmentation Avoidance. – In: Proc. of GLOBECOM 2017-2017 IEEE Global Communications Conference, IEEE, 2017.
35. Mechtri, M., C. Ghribi, D. Zeghlache. A Scalable Algorithm for the Placement of Service Function Chains. – IEEE Transactions on Network and Service Management, Vol. **13**, 2016, No 3, pp. 533-546.
36. Kim, S., et al. VNF-EQ: Dynamic Placement of Virtual Network Functions for Energy Efficiency and QoS Guarantee in NFV. – Cluster Computing, Vol. **20**, 2017, No 3, pp. 2107-2117.
37. Xing, H., et al. An Integer Encoding Grey Wolf Optimizer for Virtual Network Function Placement. – Applied Soft Computing, Vol. **76**, 2019, pp. 575-594.
38. Chantre, H. D., N. L. S. da Fonseca. Redundant Placement of Virtualized Network Functions for LTE Evolved Multimedia Broadcast Multicast Services. – In: Proc. of IEEE International Conference on Communications (ICC'17), IEEE, 2017.
39. Farshin, A., S. Sharifian. A Modified Knowledge-Based Ant Colony Algorithm for Virtual Machine Placement and Simultaneous Routing of NFV in Distributed Cloud Architecture. – The Journal of Supercomputing, Vol. **75**, 2019, No 8, pp. 5520-5550.
40. Lazarov, A. D. Mathematical Modelling of Malware Intrusion in Computer Networks. – Cybernetics and Information Technologies, Vol. **22**, 2022, No 3, pp. 29-47.
41. Petrosyan, D., H. Astsatryan. Serverless High-Performance Computing over Cloud. – Cybernetics and Information Technologies, Vol. **22**, 2022, No 3, pp. 82-92.
42. Bhargavi, K., S. G. Shiva. Uncertainty Aware T2SS Based Dyna-Q-Learning Framework for Task Scheduling in Grid Computing. – Cybernetics and Information Technologies, Vol. **22**, 2022, No 3, pp. 48-67.
43. Khebbache, S., H. Makhlouf, D. Zeghlache. A Multi-Objective Non-Dominated Sorting Genetic Algorithm for VNF Chains Placement. – In: Proc. of 15th IEEE Annual Consumer Communications & Networking Conference (CCNC'18), IEEE, 2018.
44. Diego de Freitas Bezerra, D., et al. Optimizing NFV Placement for Distributing Micro-Data Centers in Cellular Networks. – The Journal of Supercomputing. Vol. **77**, 2021, No 8, pp. 8995-9019.
45. Shokouhifar, M. FH-ACO: Fuzzy Heuristic-Based Ant Colony Optimization for Joint Virtual Network Function Placement and Routing. – Applied Soft Computing, Vol. **107**, 2021, 107401.
46. Zahedi, S. R., S. Jamali, P. Bayat. EmcFIS: Evolutionary Multi-Criteria Fuzzy Inference System for Virtual Network Function Placement and Routing. – Applied Soft Computing, Vol. **117**, 2022, 108427.
47. Bhargavi, K., S. G. Shiva. Fuzzy Neutrosophic Soft Set Based Transfer-Q-Learning Scheme for Load Balancing in Uncertain Grid Computing Environments. – Cybernetics and Information Technologies, Vol. **22**, 2022, No 4, pp. 35-55.
48. Madhumala, R. B., H. Tiwari, C. D. Verma. Virtual Machine Placement Using Energy-Efficient Particle Swarm Optimization in Cloud Datacenter. – Cybernetics and Information Technologies, Vol. **21**, 2021, No 1, pp. 62-72.

Received: 29.06.2022; Second Version: 22.11.2022; Accepted: 16.12.2022