Protection Strategies for Dynamic VNF Placement and Service Chaining

Leila Askari, Mohammadhassan Tamizi, Omran Ayoub, Massimo Tornatore
Department of Electronics, Information and Bioengineering, Politecnico di Milano
E-mail: firstname.lastname@polimi.it

***Abstract*—Network Function Virtualization (NFV) provides a major shift in the provisioning of telecommunication services by decoupling network functions from dedicated hardware devices. Such decoupling enables operational expenditure (OpEx) and capital expenditure (CapEx) reduction and allows to increase service agility. NFV relies on Virtualized Network Functions (VNFs) and, by placing VNFs on NFV-capable network nodes, and by chaining them in a specific order while guaranteeing a given end to end latency, Service Chains (SCs) are formed to provide a specific service. To achieve great flexibility in resource assignment in the network and decrease further OpEx, it is important to consider provisioning of SCs in a dynamic scenario in which traffic evolves in the network. In this study we observe that, when deploying a SC in a situation where SC requests arrive dynamically in the network, it is important to consider protection techniques to withstand failures of the network components supporting the SC. Different protection approaches can be followed to protect the SC against failures. We consider three different protection strategies, namely, Virtual-Node protection, Virtual- Link protection and End-to-End protection, which provide protection against single virtual node (hosting a VNF), single virtual link (connecting two consequent VNF of SC together) and single virtual node/virtual link failure for dynamic VNF placement. For each of them, we provide a heuristic approach for dynamic provisioning of the SC with protection. In our simulative numerical results over realistic network and SC settings, we compare the three strategies and show that End-to-End protection and Virtual-Node protection have both high blocking, however, End-to-End is able to satisfy the latency requirement of more SCs with respect to virtual node protection. Of the three protection strategies, Virtual-Link protection requires less network and computational resources and achieves lower SC latency violation.**

***Index Terms*—NFV, Protection for Dynamic Service Chaining**

I. Introduction

Thanks to recent advances in Network Function Virtualization (NFV), new emerging 5G services (e.g., Smart Factory and Augmented Reality) can now be realized by concatenating software instances, called Virtual Network Functions (VNFs), in a specific order [1]. These services are referred to as Service Chains (SCs). Considering that the next deployment phase of 5G targets Ultra Reliable Low Latency Communication (URLLC) [2], it is important to provision SCs while ensuring both the strict latency and reliability requirement of SCs [3]. However, traditional protection strategies must now be evolved in the context of NFV architectures, where SC resources to be guaranteed against failures are both computational (VNFs) and transmissive (communication links). A SC is considered available only if all of its components are available, and a reliable SC mapping into physical network is not trivial.

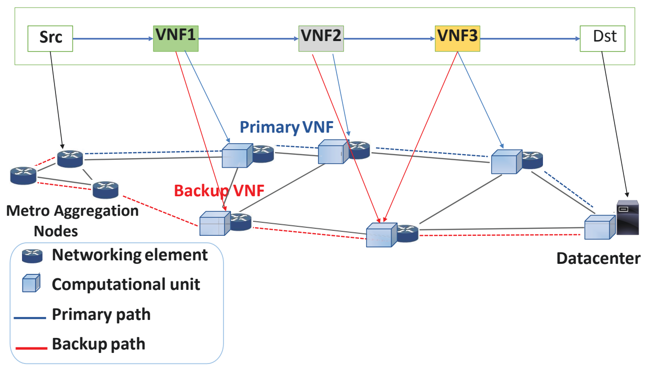


Fig. 1: Network model

Fig. 1 shows an example of protection of a SC by provisioning a working chain together with a backup chain. A SC is represented by a set of VNFs (virtual nodes) and a set of links connecting them (virtual links). To provision a SC in the network, we need to map its VNFs (virtual nodes) to physical NFV-nodes (i.e., physical nodes that are capable of hosting VNFs, in the figure they are indicated by the presence of computational units), and map virtual links (links connecting the VNFs) to physical links in the network. In our example, to provide protection against virtual node and virtual link failure, each VNFs of SC is mapped to two different physical nodes, as shown in the figure. Note that this protection strategy is only a possible protection approach, but other protection approaches can be devised. For example, to provide protection against virtual link failure, i.e., the protection against the failure of links connecting two VNFs of a SC, a backup physical path is calculated for the SC, traversing physical links that differ from those of primary physical path. Another protection strategy only considers protecting single virtual node against failure. In other words, for each NFV-node of SC there will be a backup NFV-node hosting the same VNF instance.

Therefore, in all the protection strategies, a primary SC is provisioned to deliver the related service in normal conditions and is protected through a backup SC which has its VNFs embedded in different physical locations. A more detailed description of the three protection strategies will be provided later in the paper.

It is worth mentioning that NFV enables network operators to easily duplicate the VNFs deployed in the network by simply duplicating software on different servers. Therefore, is easier to fulfill reliability requirements of deployed SCs in comparison to the case where services are provisioned using middleboxes deployed in the network. However, to efficiently protect SCs in the network, one of the important aspects to consider is the utilization of limited network resources (i.e., physical links and computational units) to fulfill the reliability requirement of each request. In fact, network operators aim to increase service acceptance ratio (decrease blocking probability) by serving as many SC requests as possible. In addition, it is desired by network operators to decrease as much as possible the Operational Expenditure (OpEx), e.g., power consumption of provisioning the SCs. Furthermore, the new URLLC services have stringent latency requirement and therefore, it is important to consider the end-to-end latency requirement of SCs while providing protection. Hence, it is essential to apply an effective protection mechanism and an efficient SC mapping strategy to meet the Service Level Agreement (SLA) (e.g., the requirement for reliability) and Quality of Service (e.g., latency requirements) of customers while consuming a smaller amount of physical resources such as to maintain high service acceptance ratio and reducing OpEx. While some studies have already investigated various SC protection techniques for static traffic, in our work, we investigate the problem of reliable SC mapping under dynamic settings. We devise three heuristic algorithms each corresponding to a protection strategy (Virtual-Link protection, Virtual-Node protection and End- to-End protection), and compare the performance of three different protection strategies. To do so, we use dynamic simulations over realistic network topology.

The remainder of this paper is as follows. In Section II we discuss related works on protection strategies for SC provisioning. Section III explain in details the protection strategies considered in this paper. Section IV formally states the problem addressed in this paper. Section V describes the algorithms developed in this study. In Section VI we discuss some illustrative numerical results obtained by simulation. Section VII concludes the paper.

II. Related work

Few recent studies have started the investigation of different mechanisms to provide protection against failures for SCs (please see [4] for detailed study of techniques for fault management and how virtualization affects fault control). A number of works proposed Integer Linear Programming (ILP) models for the problem. For example, authors in [5] propose an ILP model and a heuristic algorithm to provide protection against single-link failure for a VNF placement and scheduling using multiple link-disjoint paths in an elastic optical network. Their approach is able to enhance the efficiency of both the spectrum and computing resource compared to the conventional single- path scheme while reducing the probability of blockage. In [6] authors formulate the SC protection against single- link failure problem as an ILP model and compare three different protection schemes in terms of both computational and optical resources. Authors in [7] model the problem of VNF protection as a Mixed ILP (MILP) model and propose two greedy heuristic algorithms to map VNFs to servers and protect each VNF by providing a replica for each. However, all these works do not address SC protection in dynamic settings. Considering dynamic traffic, Ref. [8] proposes a genetic algorithms to provide end-to-end protection for a deployed SC by providing back ups for all VNFs and links connecting them. They evaluate the performance of their algorithm in a Wavelength Division Multiplexing (WDM) network in terms of service blocking ratio comparing it with those of unprotected approach and the approach in which only VNFs of a SC are protected (VNF protection). In [9], authors devise a strategy to ensure availability of a SC using both physical network protection and virtual layer VNF replicas. By assessing how to spread VNF replicas between the primary path and the backup path to achieve maximum availability of the SC, they decide the number of replicas in the SC for each VNF, and distribute the replicas to physical nodes while preserving sequentially among VNFs. In [10], authors propose a VNF forwarding graph structure that is used to choose the eligible backups by using the cost-aware Importance Measure (CIM). With CIM, the VNF placement is taken through extremely accurate mapping of these backups to the physical nodes. The proposed strategy reduces backup costs relative to the existing algorithms and retains high cost-efficiency. Ref. [11] provides two protection schemes, namely protection for all nodes and protection for only one node, and formulates the problem as an ILP for each protection scheme. Ref. [12] formulates the problem of reliable VNF placement in mobile edge-cloud networks with the objective of maximizing the profit of network service providers. Ref. [13] provides a model to design a service provider's network that balances the trade-off between profit maximization and providing a specific service availability. However, in all the above-mentioned works the impact of protection strategies on the consolidation of VNF in few NFV-nodes (an important aspect, considering its impact on OpEx as energy consumption) and end-to-end latency of SCs is neglected. Moreover, to the best of our knowledge, none of the existing studies investigates and compares three different protection strategies for dynamic SC provisioning considering latency requirement of SCs.

III. Protection Strategies

The three SC protection strategies considered in our study, which are depicted in Fig. 2 are described in the following.

A. Virtual-Node protection

The virtual-node protection scheme provides resiliency against single-node failure. Each VNF composing the SC is instantiated in two disjoint physical locations, whereas the physical paths used to concatenate the primary and backup VNFs might share physical links. This protection scheme is ideal when the probability of a failure in nodes is higher with respect to link failure probability. An example of this scenario is shown in Fig. 2(a). As illustrated, the same physical links are used in the primary (shown in red) and backup paths (shown in blue) while the same physical nodes are not used to map VNFs for primary and backup SCs.

B. Virtual-Link protection

The Virtual-Link protection strategy provides resiliency against failure of the physical links over which a virtual link connecting the VNFs of a specific SC are mapped. Each virtual link of the SCs is embedded through two disjoint physical paths, one primary path and one backup path. The primary and backup paths cannot share any physical link, while they can share physical nodes. An example of virtual-link protection is shown in Fig. 2(b). Primary and backup paths share the same physical nodes on which VNFs of both primary and backup SCs are mapped, however, the physical links belonging to primary and backup SCs are different.

C. End-to-End protection

In End-to-End protection the primary and backup path must be both node and link disjoint. Consequently, End-to-End protection provides resiliency against both single-link and single-node failures for each SC. The physical paths used to chain primary and backup VNFs must also be node disjoint. Fig. 2(c) shows an example of end-to-end protection where six VNF instances and nine links are occupied to protect a SC composed of three VNFs.

IV. Problem Statement

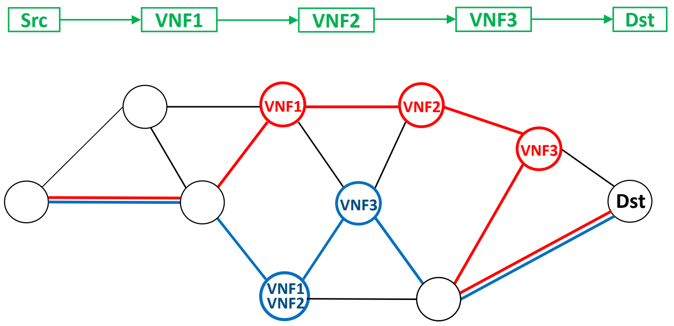
In this Section we first describe our network and SC model, then we formally state the problem of dynamic SC provisioning with protection.

A. SC modelling

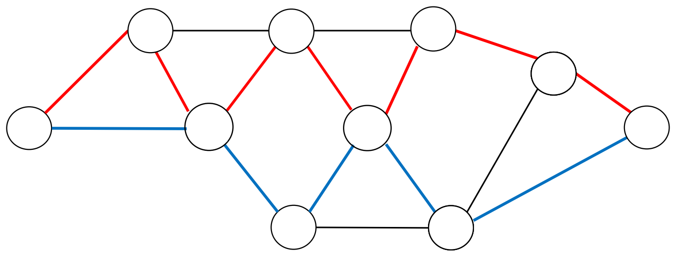
We consider that SC requests in the network are initiated by users directly connected to metro aggregation nodes that are nodes without computational unit, (source of the SC) and, depending on the latency and computational requirements of the SC, a destination node of the SC[1] is defined. Note that, in addition to a source, destination and an ordered set of VNFs, a SC request is defined by the number of users requesting that particular SC, a specific amount of bandwidth and the maximum tolerable end-to-end latency. Moreover, each VNF requires a specific amount of computational capacity defined as the fraction of the CPU cores (VCPU) usage per user.

B. Protection for dynamic service chaining problem

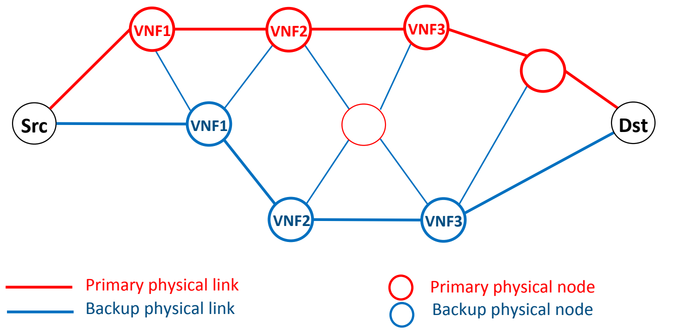
Given: i) a WDM optical metro network hosting different NFV-capable nodes (i.e., Central Offices (COs) which are equipped with computational and storage resources and can host VNFs), ii) dynamically arriving SC requests, **decide** the provisioning of the primary SC (i.e., placement of VNFs in NFV-nodes and the mapping of virtual links onto physical links) and the provisioning of the backup SC depending on considered protection strategy (virtual node virtual link, or end-to-end protection) with the **objective** of minimizing the number of NFV-nodes activated in the network (with at least one running instance of VNF), number of SCs latency violations, and the number of SC requests that are not provisioned in the network, **under the constraints** of limited link and NFV-nodes computational capacity .



(a) Virtual-Node Protection Scheme



(b) Virtual-Link Protection Scheme



(c) End-to-End Protection Scheme

Fig. 2: Protection strategies

V. Protection Algorithms

The proposed algorithms for dynamic service chaining with protection are 1) Virtual-Node Protection (*P-VN*), 2) Virtual-Link Protection (*P-VL*) and 3) End-to-End Protection (*P-E2E*) corresponding to the protection strategies discussed in subsections III-A, III-B and III-C, respectively.

All the three proposed algorithms consist of two phases: 1) Re-using a VNF instance: The algorithm tries to reuse an already activated VNF instance in the network taking into account the requirements of protection strategy and 2) Instantiating a new VNF instance: The algorithm selects the NFV-node to activate considering locality[2] and betweenness

[1]If the SC has stringent latency requirement, a destination node close to the source of SC is chosen. If the SC needs high amount of computational resources, a destination node close to the Core CO is chosen.

[2]Length of shortest path between the source node of the SC and the considered node plus the length of the shortest path between that node to the destination node of SC.

centrality[3]. *P-VN*, *P-VL* and *P-E2E* take as input the following parameters:

- *G(N,E)*: Current status of the network modeled as a layered graph
- F: Set of NFV-nodes in the network ($F \subseteq N$)
- V: Set of VNFs to be mapped in the network
- S: Types of SCs to be deployed
- W: Number of wavelengths for each link in the network
- r: SC request specified by:
  - S_r: Source node of the SC request
  - D_r: Destination node of the SC request
  - $N_{vnf,r}$: Number of VNFs forming the SC
  - V_r: VNFs forming this SC ($V_r \in V$)
  - L_r: Latency tolerated by users requesting SC
  - B_r: Bandwidth requirements of SC
  - H_r: Holding time of SC request

A. P-VN algorithm

As the first phase, to place the first VNF belonging to SC, *P-VN* searches for two NFV-nodes (for primary and backup SC) running VNF instances in the network with enough computational capacity. In case this phase is successful, algorithm stores both $F_{primary}$ and F_{backup}, and a flag is set indicating that they belong to primary and backup SCs respectively, enforcing NFV-node-disjointness. This step is repeated whenever a node is chosen either for $F_{primary}$ or F_{backup}. Note that, the node with lower locality is chosen as $F_{primary}$ while the other one is chosen as F_{backup} to guarantee having shortest path for primary SC. It is worth mentioning that before choosing a node as $F_{primary}$, *P-VN* makes sure the node does not have a flag set for backup SC. Similarly, for choosing an NFV-node for F_{backup}, *P-VN* makes sure the node does not have a flag set for primary SC.

If such NFV-nodes are not found, *P-VN* will start the second phase. In this phase, *P-VN* at first calculates the shortest path between S_r and D_r ($SP_{S,D}$) then it tries to find two NFV-nodes with enough computational capacity on this path. If the two NFV-nodes are successfully found, algorithm will keep these nodes as $F_{primary}$ and F_{backup} and sets the corresponding flags for them. However, in case only one such NFV-node is available on $SP_{S,D}$, this node is chosen as $F_{primary}$ and, as explained earlier, this node will be invalidated to avoid using it for the backup SC. After that, *P-VN* will start the procedure to find F_{backup}. Note that in case no NFV-node with enough computational capacity is found on $SP_{S,D}$, *P-VN* selects, out of all the NFV-nodes with enough computational capacity in the network, two nodes (or one node in case $F_{primary}$ is already chosen) with lowest values of locality are chosen as $F_{primary}$ and F_{backup}. After that, the flags are set for $F_{primary}$ and F_{backup} as mentioned earlier. These steps are repeated until either all the VNFs of a SC are placed on NFV-nodes and bandwidth is allocated to both primary and backup SCs (SC is provisioned), or one of the above-mentioned steps is failed and SC is blocked.

[3]This metric is a measure of centrality in a graph based on the shortest path. It indicates the number of all feasible shortest paths between any node pair that are passing through each considered node.

Algorithm 1: P-VN Algorithm

```
1: Given: Network state G(N,E), Service Chain request $r(S_r, D_r, N_{vnf,r}, V_r, L_r, B_r, H_r)$
2: repeat
3:     $V'_r \leftarrow$ Select the next VNF $\in V_r$
4:     if $\exists$ NFV-nodes $f \in F$ with enough capacity hosting $V'_r$ then
5:         $F_{vnf} \leftarrow$ all$f$
6:         Sort all $f_v \in F_{vnf}$ by increasing value of $loc(f_v)$
7:         if Flag( $f_v$[0])=Null or !=backup then
8:             $F_{primary} \leftarrow f_v[0]$
9:         end if
10:        if Flag( $f_v$[1])=Null or !=primary then
11:            $F_{backup} \leftarrow f_v[1]$
12:        end if
13:        if success then
14:            Scale up $F_{primary}$ and $F_{backup}$ & update $N_{state}$
15:            Flag( $F_{primary}$) $\leftarrow$ primary
16:            Flag( $F_{backup}$) $\leftarrow$ backup
17:            goto 3
18:        end if
19:    else \* Activate a new $f$ with enough capacity and place $V'_r$ *\
20:        $SP_{S,D} \leftarrow shortestpath(S_r, D_r)$
21:        Select the $f \in SP_{S,D}$ with the lowest loc(f) & Flag(f)=Null or !=backup as $F_{primary}$
22:        Flag( $F_{primary}$) $\leftarrow$ primary
23:        Select next $f \in SP_{S,D}$ with the lowest loc(f) & Flag(f)=Null or !=primary as $F_{backup}$
24:        Flag( $F_{backup}$) $\leftarrow$ backup
25:        Update $N_{state}$
26:        if failed then
27:            Try to map $V'_r$ on a $F_{primary}$ and a $F_{backup}$ until all $f \in F$ tried and update the flags
28:        end if
29:        if failed then
30:            return r blocked
31:        end if
32:    end if
33: until All $V'_r \in Vr$ are placed and chained
```

B. P-VL algorithm

At first, *P-VL* tries to place the first VNF belonging to V_r reusing an already active VNF instances in the network selecting an NFV-node with higher value of betweenness centrality and enough computational capacity. Note that, since the NFV-nodes are shared between primary and backup SC, the available computational capacity of a candidate NFV-node should be higher than twice the required computational capacity for the VNF to be possibly placed. If two disjoint paths from S_r to this node are available, the node will be chosen as NFV-node for primary and backup SC, and two disjoint paths are stored as $Path_{primary}$ and $Path_{backup}$. Note that, when we are deciding about the placement of the first VNF $\in V_r$, the shortest path is calculated between S_r and candidate NFV-node. Otherwise, the shortest path is calculated between the last NFV-node chosen for r and this node. Then, flags are set to ensure that primary and backup SCs are link-disjoint. Consequently, $Path_{primary}$ and $Path_{backup}$ are calculated considering their respective links.

If the first phase is not successful, *P-VL* starts the second phase to instantiate a new instance of VNF to select on an

NFV-node. To this end, *P-VL* chooses an NFV-node with enough computational capacity and higher value of betweeness centrality. If two disjoint paths with proper flags are available to connect this node either to the S_r (in case first VNF in V_r needs to be placed) or the last NFV-node chosen for SC, these two paths are stored as $Path_{primary}$ and $Path_{backup}$ and the flags for links belonging to these paths are updated. It is worth mentioning that if two disjoint paths are not available for the chosen NFV-node, another NFV-node with highest value of betweenness centrality among rest of NFV-nodes is chosen. The same steps are followed until all the VNFs in V_r are mapped to an NFV-node or one of the steps is not successfully followed hence blocking the SC.

Algorithm 2: P-VL Algorithm

```
1: Given: Network state G(N,E), Service Chain request $r(S_r, D_r, N_{vnf,r}, V_r, L_r, B_r, H_r)$
2: repeat
3:     $V'_r \leftarrow$ Select the next VNF $\in V_r$
4:     if $\exists$ NFV-nodes $f \in F$ with enough capacity hosting $V'_r$ then
5:         $F_{vnf} \leftarrow$ all $f$
6:         Sort all $f_v \in F_{vnf}$ by decreasing value of $Betweenness(f_v)$
7:         if $F_r$ is empty then
8:             SP $\leftarrow shortestpath(S_r, f_v[0])$
9:         else
10:            SP $\leftarrow shortestpath(F_r[last], f_v[0])$
11:        end if
12:        if $\exists$ 2 paths in SP and links have Null or correct flags then
13:            $F_r \leftarrow f_v[0]$
14:            Scale up $F_r$ & update $N_{state}$
15:            $Path_{primary} \leftarrow SP[0]$
16:            Flag (all $links \in Path_{primary}) \leftarrow primary$
17:            $Path_{backup} \leftarrow SP[1]$
18:            Flag (all $links \in Path_{backup}) \leftarrow backup$
19:        else
20:            goto 22
21:        end if
22:    else \* Activate a new f with enough capacity and place $V'_r$ *\
23:        Select the $f \in F$ with the highest $Betweenness(f)$ & enough computational resources
24:        if $F_r$ is empty then
25:            SP $\leftarrow shortestpath(S_r, f)$
26:        else
27:            SP $\leftarrow shortestpath(F_r[last], f)$
28:        end if
29:        if $\exists$ 2 paths in SP and links have Null or correct flags then
30:            $F_r \leftarrow f$
31:            Scale up $F_r$ & update $N_{state}$
32:            $Path_{primary} \leftarrow SP[0]$
33:            Flag (all $links \in Path_{primary}) \leftarrow primary$
34:            $Path_{backup} \leftarrow SP[1]$
35:            Flag (all $links \in Path_{backup}) \leftarrow backup$
36:        else
37:            goto 23
38:        end if
39:        if failed then
40:            return r blocked
41:        end if
42:    end if
43: until All $V'_r \in Vr$ are placed and chained
```

C. P-E2E algorithm

This algorithm, whose pseudocode is shown in *Algorithm* 3, at the first phase tries to find an already active NFV-node with enough computational capacity that has one running instance of first VNF in V_r. In case more than two nodes are available, the *P-E2E* will choose the two with lower value of locality and higher value of betweenness centrality considering that the shortest paths between S_r and each of the nodes are not sharing any link. If it is the case, the flags for $F_{primary}$, F_{backup} and links belonging to the path between S_r and $F_{primary}$ ($Path_{primary}$) and links belonging to the path between S_r and F_{backup} ($Path_{backup}$) are updated accordingly. Note that each time an NFV-node and path for primary or backup SC is defined, the flags of NFV-nodes and links belonging to the paths need to be updated accordingly. However, if only one such node is available, it will be chosen as $F_{primary}$ and after the $Path_{primary}$ is calculated the flags will be updated, indicating $F_{primary}$ and links belonging to $Path_{primary}$ belong to primary SC. Therefore, in the case that no already active NFV-node is available either for F_{backup} or for both $F_{primary}$ and F_{backup}, *P-E2E* will start the second phase. In the second phase, initially, *P-E2E* chooses two NFV-nodes with enough computational capacity and lower locality value and higher betweenness centrality value in a way that the shortest path between S_r and these nodes are link-disjoint. Assuming that such NFV-nodes are found, the flags for NFV-nodes and links belonging to paths are updated and the same procedure will be followed to place all VNFs in V_r. Note that for the next VNFs in V_r, the primary and backup path are calculated between last NFV-node in the SC and the selected NFV-node instead of S_r. *P-E2E* stops when either all the VNFs in V_r for both primary and backup SCs are successfully provisioned, or when one of the steps fails.

D. Complexity of heuristic algorithms

In this section we analyse the complexity of the proposed heuristic algorithms. To calculate the shortest path for *Algorithm* 1 and *Algorithm* 3, we use Dijkstra's algorithm that is implemented using binary heaps. While for *Algorithm* 2 we use Suurballe's algorithm which is implemented by using Dijkstra's algorithm twice. Therefore, the computational complexity for shortest path calculation is in the order of $O(E + N \log N)$ for all the algorithms. In addition, since the shortest path calculation needs to do be done for each VNF, therefore, the complexity for all the algorithms is in the order of $O(|V_r|[E + N \log N])$.

VI. Illustrative Simulative Results

To compare the performance of our algorithms, we developed a discrete dynamic event-driven simulator in C++. In our simulations, SC requests, chosen randomly from the SCs listed in Table I with diverse requirements in terms of bandwidth and latency, dynamically arrive based on Poisson distribution with negative exponential holding time. The VNFs forming these SCs are Firewall (FW), Traffic Monitor (TM), Intrusion Detection System (IDS), Network Address Translation (NAT), Video Optimizer (VO). The CPU core requirements of each of the VNFs are tabulated in Table II.

Algorithm 3: P-E2E Algorithm

```
1:  Given: Network state G(N,E), Service Chain request $r(S_r, D_r, N_{vnf,r}, V_r, L_r, B_r, H_r)$
2:  repeat
3:      $V_r' \leftarrow$ Select the next VNF $\in V_r$
4:      if $\exists$ NFV-nodes $f \in F$ with enough capacity hosting $V_r'$ then
5:          $F_{vnf} \leftarrow$ all $f$
6:          Sort all nodes $\in F_{vnf}$ by decreasing value of $Betweenness()$
7:          $f_v \leftarrow$two nodes $\in F_{vnf}$ with lowest value of $loc(f_v)$
8:          if Flag( $f_v$[0])=Null or !=backup then
9:              $F_{primary} \leftarrow f_v[0]$
10:             if $F_{r,primary}$ is empty then
11:                 $Path_{primary} \leftarrow SP(S_r, f_v[0])$
12:             else
13:                 $Path_{primary} \leftarrow SP(F_{r,primary}[last], f_v[0])$
14:             end if
15:             Flag( $F_{primary}$) $\leftarrow$ primary
16:             Add $F_{primary}$ to list $F_{r,primary}$
17:             Flag (all $links \in Path_{primary}$) $\leftarrow primary$
18:             Scale up $F_{primary}$ & update $N_{state}$
19:         end if
20:         if Flag( $f_v$[1])=Null or !=primary then
21:             $F_{backup} \leftarrow f_v[1]$
22:             if $F_{r,backup}$ is empty then
23:                 $Path_{backup} \leftarrow SP(S_r, f_v[0])$
24:             else
25:                 $Path_{backup} \leftarrow SP(F_{r,backup}[last], f_v[0])$
26:             end if
27:             if $\exists$ $Path_{backup}$ with correct flags then
28:                 Flag( $F_{backup}$) $\leftarrow$ backup
29:                 Add $F_{backup}$ to list $F_{r,backup}$
30:                 Flag (all $links \in Path_{backup}$) $\leftarrow backup$
31:                 Scale up $F_{backup}$ & update $N_{state}$
32:                 goto 3
33:             end if
34:         end if
35:     else \* Activate new NFV-nodes with enough capacity and place $V_r'$ *\
36:         Sort nodes $\in F$ with the highest $Betweenness()$ & enough computational resources
37:         $f_v \leftarrow$two nodes $\in F$ with lowest value of $loc(f_v)$
38:         goto 8
39:         if failed then
40:             return $r$ blocked
41:         end if
42:     end if
43: until All $V_r' \in Vr$ are placed and chained for primary and backup SCs
```

We considered a realistic optical metro topology (shown in Fig. 3) of 52 nodes out of which two are Metro Core Backbone (MCB), 6 are Metro Core (MC) nodes and the rest are Metro Aggregation (MA) nodes in which SC requests are generated. We consider that there are 21 NFV-nodes (all the MCBs and MCs and 13 MAs) each equipped with 64 vCPU except for MCBs that have unlimited computational capacity. In this network there are 156 bidirectional WDM links each supporting 16 wavelengths with 10 Gbps capacity.

In addition to the proposed algorithms, we consider a baseline heuristic approach with no protection *No-P*. We compare the performances of these approaches in terms of: i) *Bandwidth Blocking Probability (*BBP*)*: calculated considering the percentage of bandwidth associated with the blocked SC requests out of total bandwidth related to all SC requests; ii) *average number of active NFV-nodes*: represents the average number of NFV-nodes with at least one running instance of a VNF throughout the simulation which gives an estimation of the power consumption; iii) *latency violation ratio*: represents the number of provisioned SC requests that did not satisfy the end-to-end latency requirement of the SC, out of the total number of SC requests generated in the network. All the results are attained and plotted with a confidence level of 95% until a 5% confidence interval on blocking probability.

Fig. 4(a) shows BBP with respect to traffic load for all protection strategies. First, results show that the protection strategies, compared to *No-P*, show significantly higher BBP (more than two orders of magnitudes higher), due to the relatively larger use of physical networks resources with respect to *No-P*. This shows that ensuring protection of SCs requires a significant additional investment in network physical resources. Among the protection strategies, for all values of traffic load, *P-E2E*, as expected, experiences the highest BBP, as, for this strategy, primary and backup SCs are completely disjoint, i.e., they cannot share the same physical resources, hence, *P-E2E* utilizes significantly more network resources than other strategies, leading to higher BBP. *P-VN*, although capable of sharing some link resource to provision working and backup SCs, has to duplicate VNF instances to provide virtual node protection, leading to long paths and ending up in utilizing more network resources than *P-VL*, which, yields the lowest BBP among the protection strategies as the same VNFs can be exploited for both the primary and backup SCs while only links need to be disjoint.

TABLE I: Service Chains with their corresponding VNFs, bandwidth and latency requirements

Service Chain	Service Chain VNFs	Bandwidth	Latency
Augmented Reality	NAT-FW-TM-VO-IDS	100 Mbps	1 ms
MIoT	NAT-FW-IDS	100 Mbps	5 ms
Smart Factory	NAT-FW	100 Mbps	1 ms

TABLE II: CPU Core Usage for VNFs

VNF Name	NAT	FW	VO	TM	IDS
CPU Core	0.0184	0.018	0.108	0.266	0.214

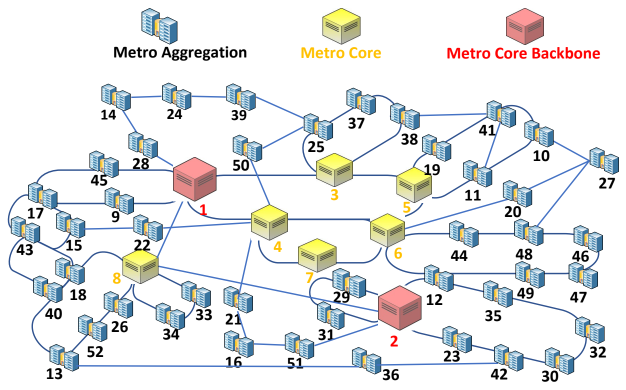
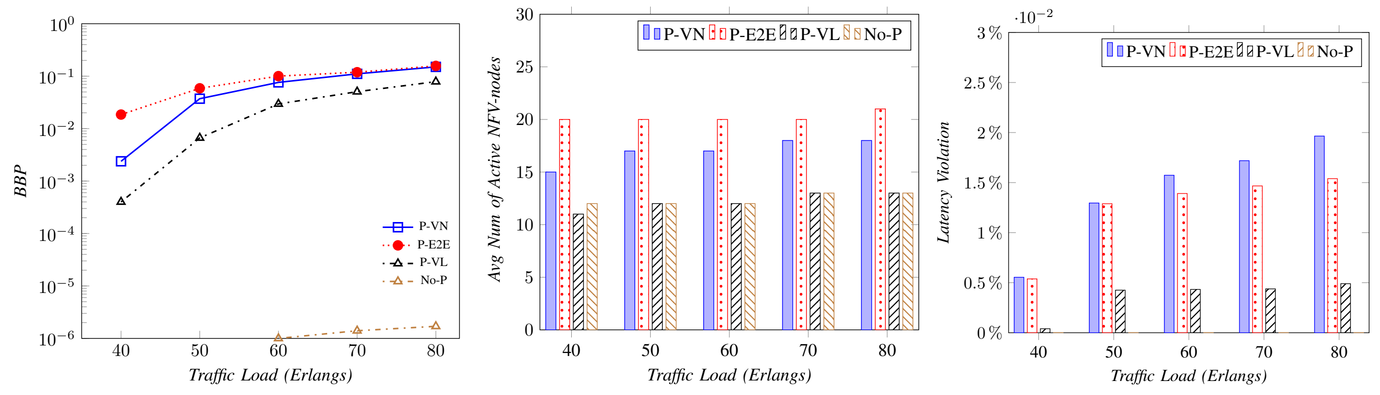


Fig. 3: Network topology



(a) BBP (b) Average number of active NFV-nodes (c) Latency violation

Fig. 4: Simulation results for different protection strategies

Fig. 4(b) shows the average number of active NFV-nodes. Results show that *P-E2E* activates more NFV-nodes with respect to other strategies as both links and nodes are required to be disjoint, and, hence, if an NFV-node on $Path_{primary}$ has enough resources and is not yet hosting any VNF for primary SC, it cannot be utilized for backup SC, as otherwise the links will not be disjoint (links reaching this NFV-node are already used for primary SC). Note that, for lower traffic loads, *P-VL* activates almost 50% less NFV-nodes in comparison to *P-E2E* as links are less congested and so it is more likely to find two disjoint shortest paths between two already active NFV-nodes. Finally, for *P-VN*, the average number of active NFV-nodes lies between those of *P-E2E* and *P-VL* due to the same reason as in the case of BBP, i.e., because it has to duplicate VNF instances to provide virtual node protection.

We now compare the latency violation ratio which is depicted in Fig. 4(c). First, we note that *P-VL* has the lowest latency violation among protection strategies. This is because *P-VL* approach tries to find the two shortest path between any virtual node pairs of SC, and shares VNFs between primary and backup SC (the placement is already done in a way to minimize end-to-end latency). On the contrary, *P-VN* has the highest latency violation, as the NFV-nodes chosen for the SC are not always the closest ones since *P-VN* is not allowed to share any NFV-node between primary and backup SC. Hence, *P-VN* tries to reuse the already activated NFV-node in the network even with longer path. However, in *P-E2E*, paths are disjoint, and so, NFV-nodes can be chosen along disjoint shortest path between source and destination of SC, allowing to avoid latency violation more than in *P-VN*. We further note that, for all approaches, around 60% of the latency violation relates to the backup path while the rest related to the primary path, which is due to the fact that the length of the backup path is always longer than the primary path.

A. Impact of NFV-node capacity on the performance of the heuristic algorithms

In this section we compare the performance of three protection strategies in terms of blocking probability for the case in which the computational capacity of all NFV-nodes (except the Core CO) is considered to be equal to 16 vCPU instead of 64 vCPU. Fig. 5 shows the BBP with respect to traffic load for all protection strategies in this case. Results show that by decreasing node capacity, the difference between blocking probability in *P-VL* protection and other approaches decreases (becomes smaller with respect to the previous case depicted in Fig. 4(a)). This is due to the fact that, for *P-VL* protection approach, we share the NFV-nodes between primary and backup SC. Therefore, when nodes have lower computational capacity, as we are placing the VNFs of both primary and backup SC on a single node, the probability of having a node with enough computational capacity for VNFs of both primary and backup SC decreases. Hence, *P-VL* protection approach achieves higher blocking probability with respect to the scenario in which nodes have more computational resources, thus approaching the BBP of *P-VN* and *P-E2E*. Moreover, as expected, the BBP of all three approaches increases with respect to the previous scenario in which NFV-nodes have higher capacity. In particular, BBP increases by up to 20% for *P-VN*, by up to 37% for *P-E2E* and by up to 50% for *P-VL*. We also observe that, blocking probability due to node capacity in *P-VL* protection approach is about 30%. As for *P-VN* protection approach, around 30% of blocking probability is related to link capacity. Note that, by increasing traffic load, the blocking happens mostly due to the lack of computational resources on NFV-nodes.

B. Impact of number of NFV-nodes in the network on performance of the heuristic algorithms

We perform further analysis considering different number of NFV-nodes in the network. That is, for each protection strategy, we considered different scenarios in which the number of NFV-nodes in the network differs and equals to 10, 21 and 30. Note that these numbers are chosen in a way to study the impact of having 20%, 40% and 60% of nodes in the network equipped with computational capacity to host VNFs, for each protection strategy. We used the blocking probability and average number of NFV-nodes metrics, introduced earlier in this section, to compare the performance of our heuristic algorithms in different scenarios. Fig. 6 illustrates the blocking

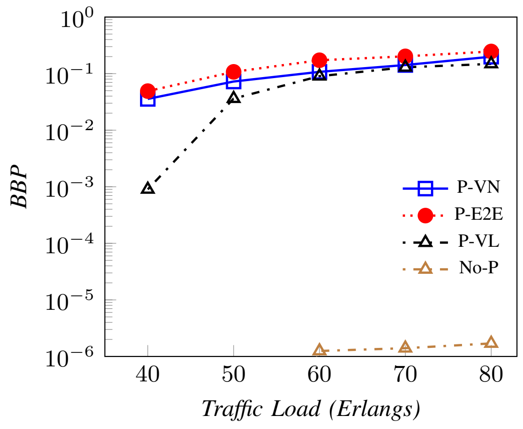


Fig. 5: BBP comparison for limited node capacity

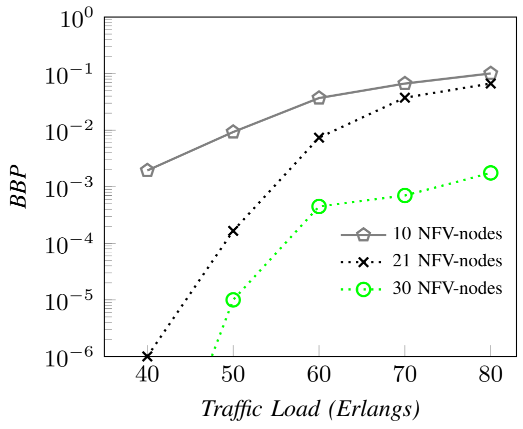


Fig. 7: BBP comparison for different number of NFV-nodes for P-VN

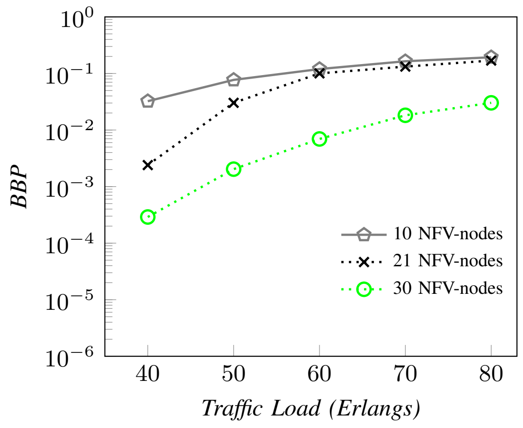


Fig. 6: BBP comparison for different number of NFV-nodes for P-E2E

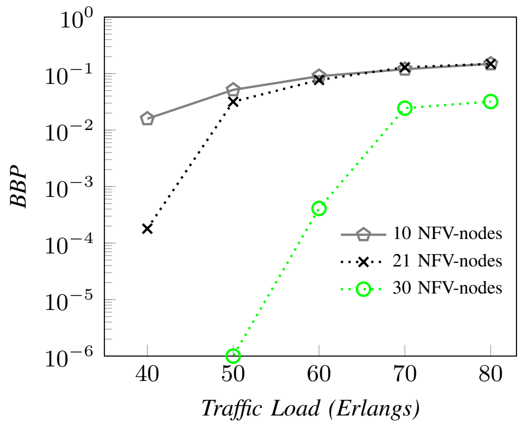


Fig. 8: BBP comparison for different number of NFV-nodes for P-VL

probability for *P-E2E* protection algorithm considering different number of NFV-nodes in the network. As it is shown, by tripling the number of NFV-nodes we are able to get better performance up to two orders of magnitude for *P-E2E* protection strategy. However, doubling the number of NFV-nodes only for lower traffic load improves algorithm performance and for higher traffic loads we observe that the scenarios with 10 and 21 NFV-nodes almost achieve the same performance in terms of connections provisioned. This is due to the fact that for higher traffic loads, links are more congested which results in having most of the NFV-nodes out of 21 NFV-nodes not accessible. As for the *P-VN* protection, in Fig. 7, we observe that increasing the number of NFV-nodes in the network has higher impact on the performance of algorithm with respect to *P-E2E* protection. This is due to the fact that in this approach since the primary and backup SCs only need to be node disjoint, adding more NFV-nodes in the network simply increases the number of SC requests that can be provisioned in the network. Finally, for *P-VL* protection algorithm, as depicted in Fig. 8 we notice that the impact of having more NFV-nodes in decreasing the blocking probability for lower traffic loads, is more significant than high traffic loads. This is due to the fact that for higher traffic loads links are more congested and therefore, the probability of having two disjoint path between node pairs decreases. Thus, having more NFV-nodes will not have high impact on increasing the number of SCs that can be provisioned in the network as there are no available links to connect these nodes together.

As for the average number of active NFV-nodes in the network, we observe that, for both *P-E2E* and *P-VN*, increasing number of NFV-nodes in the network leads to having more average active NFV-nodes. While for *P-VL*, on the contrary, increasing the number of NFV-nodes does not have high impact on the average number of active NFV-nodes. In fact, for the scenarios with 21 and 30 NFV-nodes, the average number of active NFV-nodes for *P-VL* remains almost the same for different traffic loads. This is due to the fact that in *P-VL* protection approach we are allowed to share NFV-nodes for primary and backup SC, therefore, in this approach even when we have more NFV-nodes available in the network the average number of active NFV-nodes will not differ significantly for different traffic loads.

VII. Conclusion

We investigated the problem of provisioning protected SCs in metro optical networks. We implemented three heuristic

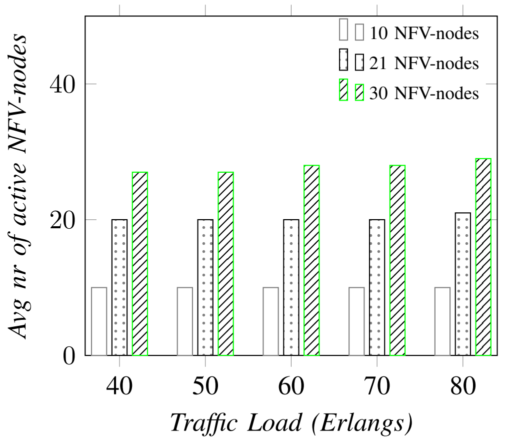


Fig. 9: Average number of NFV-nodes comparison for P-E2E

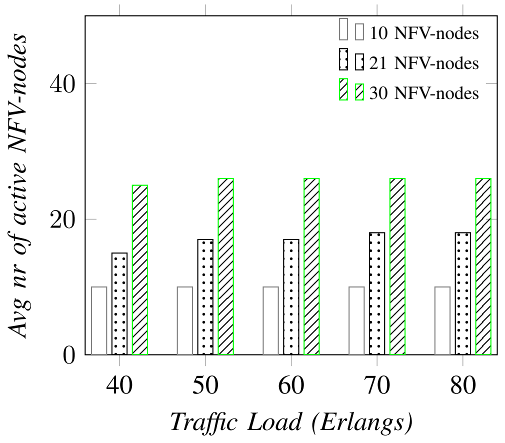


Fig. 10: Average number of NFV-nodes comparison for P-VN

algorithms for dynamic SC provisioning with protection considering *Virtual-Node*, *Virtual-Link* and *End-to-End* protection strategies against single link and/or single node failure. Using dynamic simulations we compared their performance to that of a baseline dynamic service chaining algorithm with no protection in terms of bandwidth blocking probability, average number of active NFV-nodes and latency violation in a realistic topology considering different types of SCs. Results show that, for *End-to-End* protection scheme, that provides the highest level of protection, for higher traffic loads the performance in terms of blocking probability gets very close to *Virtual-Node* protection strategy while for lower traffic loads the difference in performance between these two approaches is noticeable. In addition, *End-to-End* protection imposes up to 10% more OpEx with respect to *Virtual-Node*. However, *End-to-End* in terms of latency violation ratio achieves better performance with respect to *Virtual-Node* protection. In addition *Virtual-Link* is a protection strategy that requires least amount of network resources and achieves better performance in terms of satisfying latency requirement of SCs and decreasing OpEx.

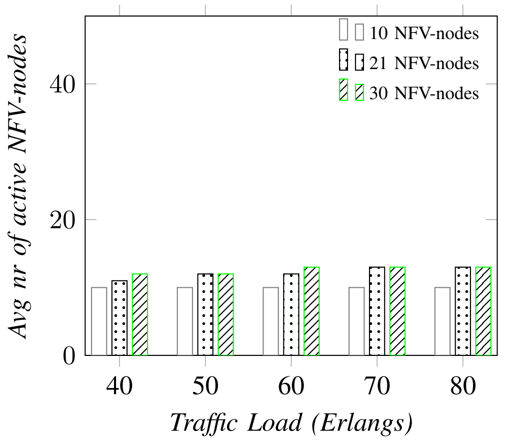


Fig. 11: Average number of NFV-nodes comparison for P-VL

References

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

[2] C. Zhang. (2018) "Optical Networking in the Cloud and 5G Era," Keynote talk at Optical Fiber Communications Conference and Exhibition (OFC). [Online]. Available: https://www.ofcconference.org/en-us/home/ news-and-press/ofc-video-library.

[3] J. Fan, Z. Ye, C. Guan, X. Gao, K. Ren, and C. Qiao, "GREP: Guaranteeing reliability with enhanced protection in NFV," in *Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, ser. HotMiddlebox '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 13–18.

[4] S. Cherrared, S. Imadali, E. Fabre, G. Gössler, and I. G. B. Yahia, "A survey of fault management in network virtualization environments: Challenges and solutions," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1537–1551, 2019.

[5] T. Gao, X. Li, W. Zou, and S. Huang, "Survivable VNF placement and scheduling with multipath protection in elastic optical datacenter networks," in *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, 2019, pp. 1–3.

[6] J. Pedro and A. Eira, "Hybrid backup resource optimization for VNF placement over optical transport networks," in *Optical Network Design and Modeling, ONDM 2019*.

[7] M. Casazza, P. Fouilhoux, M. Bouet, and S. Secci, "Securing virtual network function placement with high availability guarantees," in *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, 2017.

[8] L. Ruiz, R. J. Durán, I. de Miguel, N. Merayo, J. C. Aguado, P. Fernández, R. M. Lorenzo, and E. J. Abril, "Comparison of different protection schemes in the design of VNF-Mapping with VNF resiliency," in *2020 22nd International Conference on Transparent Optical Networks (ICTON)*.

[9] J. Kong, I. Kim, X. Wang, Q. Zhang, H. C. Cankaya, W. Xie, T. Ikeuchi, and J. P. Jue, "Guaranteed-availability network function virtualization with network protection and VNF replication," in *2017 IEEE Global Communications Conference*, 2017.

[10] W. Ding, H. Yu, and S. Luo, "Enhancing the reliability of services in NFV with the cost-efficient redundancy scheme," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.

[11] L. Fang, X. Zhang, K. Sood, Y. Wang, and S. Yu, "Reliability-aware virtual network function placement in carrier networks," *Journal of Network and Computer Applications*, vol. 154, p. 102536, 2020.

[12] J. Li, W. Liang, M. Huang, and X. Jia, "Reliability-aware network service provisioning in mobile edge-cloud networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1545–1558, 2020.

[13] S. Sharma, A. Engelmann, A. Jukan, and A. Gumaste, "VNF availability and SFC sizing model for service provider networks," *IEEE Access*, vol. 8, pp. 119 768–119 784, 2020.