Latency- and Capacity-aware Placement of Chained Virtual Network Functions in FMC Metro Networks[☆]

Ali Hmaity[a], Marco Savi[b,*], Leila Askari[a],
Francesco Musumeci[a], Massimo Tornatore[a], Achille Pattavina[a]

[a] *Politecnico di Milano, Via Ponzio 34/5, 20133 Milano, Italy*
[b] *Fondazione Bruno Kessler, CREATE-NET Research Center, 38123 Trento, Italy*

Abstract

Most of today's cloud applications are delivered by Cloud Service Providers (CSPs) on top of a physical network managed by one or multiple Infrastructure Providers (InPs). This new way of delivering services is impacting InPs' revenues, as InPs are only responsible for transporting data to users. Network Function Virtualization (NFV) was proposed to help InPs gain more flexibility in provisioning new services over their networks, hence achieving lower capital and operational costs, keeping stable revenue margins, and resisting the competition of CSPs (e.g., the "Over-The-Top"players). NFV aims at moving from the traditional approach of network functions running over dedicated hardware (e.g., firewall, NAT, etc.) into virtualized software modules running on top of Commercial Off The Shelf (COTS) equipment. However, deploying NFV in an operational network requires addressing two fundamental problems. The first consists on determining the locations where Virtual Network Functions (VNFs) will be hosted (i.e., VNF placement) and the second on how to properly steer network traffic to traverse the required VNFs in the right order (i.e, routing), thus provisioning network services in the form of Service Function Chains (SFCs). In this work we try to solve both problems focusing our analysis on a metro-regional scenario, where link bandwidth and COTS node processing capacity is inherently limited and where the current trend consists on moving towards a Fixed and Mobile Convergence (FMC) network infrastructure. We propose and compare different heuristic strategies for SFC provisioning, characterized by latency and/or capacity awareness (i.e., able to best exploit latency of links and/or processing capacity of COTS nodes for an effective placement of VNFs) and by the adoption of a load balancing policy for traffic routing, with

[☆] A preliminary version of this paper appeared in International Conference on Transparent Optical Networks (ICTON), 2016 [1]

[*] Corresponding author. *Tel:* +390461312478
Email addresses: ali.hmaity@polimi.it (Ali Hmaity), m.savi@fbk.eu (Marco Savi), leila.askari@polimi.it (Leila Askari), francesco.musumeci@polimi.it (Francesco Musumeci), massimo.tornatore@polimi.it (Massimo Tornatore), achille.pattavina@polimi.it (Achille Pattavina)

the aim of maximally consolidating VNFs. We assess the benefits of our strategies against a state-of-the-art algorithm, both in terms of number of required COTS nodes in the metro/access network and of SFC acceptance ratio. Our findings indicate that combining latency and capacity awareness in the VNF placement process with a load-balancing routing strategy brings high benefits in terms of VNF consolidation and SFC acceptance ratio.

Keywords: Network Function Virtualization, Service Function Chaining, VNF placement, FMC networks, metro/access networks

---

1. Introduction

Traditional telecom networks rely on proprietary hardware appliances (e.g., firewall, NAT, IDPS, etc.) to provide the network functions requested to support Internet services. Such appliances are statically configured making it difficult to modify/deploy existing/new services. Further, with the rise of cloud services, Infrastructure Providers (InPs) (e.g., telecom network operators) are mostly responsible only of transporting traffic from Cloud Service Providers (CSPs) to users, and hence are suffering squeezed profit margins. Therefore, telecom network infrastructures need to evolve towards a more agile and flexible service-deployment platform that can ensure a reduction of Capital Expenditures (CapEx) and Operational Expenditures (OpEx).

In this context, Network Function Virtualization (NFV) [2] was proposed as a solution for InPs. NFV allows to integrate hardware functionality into software modules called Virtual Network Functions (VNFs) and run them on Commercial-Off-The-Shelf (COTS) hardware disseminated in the network. Such paradigm is expected to introduce new revenue opportunities by enabling faster time to market and advanced automation of service deployment. In NFV, user traffic for a specific Internet service must traverse a specific set of VNFs. Such process is referred to as Service Function Chaining (SFC) [3] (or Service Chaining, SC) and involves addressing two important problems: the operator needs to *(i)* decide where to place the VNFs in the network and *(ii)* how to steer user traffic to traverse specific VNFs in an ordered manner, based on the type of service.

Especially, deciding where to locate VNFs in the network (i.e., in which COTS nodes) while satisfying latency and bandwidth requirements for the provisioned SFCs is a non-trivial task. From a cost perspective, the optimal solution for a network operator would be placing the VNFs in a datacenter (DC) with high computational power, to provide all Internet services from a centralized and cheap location. As side effect, this solution may degrade the performance of latency-sensitive SFCs, due to the excessive distance of the DCs that, in some cases, can be even thousands of kilometers far from the users. Hence, to avoid such degradation, it is necessary to distribute VNFs closer to the users within the metro/access segments [4]. Passive Optical Networks (PONs) or Active Optical Networks (AONs) [5] are more and more adopted for the aggregation of

fixed or mobile users' traffic in the metro/access segments, but they still offer bandwidth capacity that is typically much scarcer than the available bandwidth in the core segment. This means that the metro/access networks can become a bottleneck if too much traffic is kept within the metro/access domain (i.e., by hosting too many VNFs), potentially causing disruptive effects [6]. This is true also for computational resources, which are in general much more constrained at the edge of the network than in remote DCs [7]. Hence, the trade-off between centralization and distribution of VNFs needs to be best exploited.

From a VNF distribution perspective, the main candidate nodes to host VNFs in the metro/access segments are the Central Offices (COs) at different hierarchical levels of the fixed and mobile aggregation networks. However, the effectiveness of VNF distribution across the edge of the network is hindered, aside from bandwidth limitations, by the fact that fixed and mobile access and aggregation networks have evolved independently, meaning that VNFs placed in fixed network COs cannot be easily accessed by mobile users (and vice versa). Recent studies (e.g. [8]) have targeted the definition of novel architectures for Fixed and Mobile Convergent (FMC) networks, where the fixed and mobile networks are jointly designed and optimized both from a functional (i.e., by unifying network functionalities) and structural (i.e., by sharing equipment and infrastructures) perspective. In this study, we argue that the adoption of a FMC access and aggregation network can help network operators consolidate VNFs in shared locations (i.e., over the same COTS hardware) for fixed and mobile users, drastically reducing CapEx and OpEx. To explore the benefits of Fixed and Mobile Convergence, we evaluate the impact of *FMC* on VNF consolidation when different SFCs must be embedded in the network, and we compare its performance with a state-of-the-art (*No FMC*) solution.

For VNF embedding and SFC provisioning in the metro/access segments, we initially adopted the heuristic algorithm proposed in [9]. However, such algorithm is designed for SFC provisioning in the core network segment, where available bandwidth can be considered unconstrained and available computational power (in remote DCs) is very high. Indeed, this assumption is not realistic in a metro-regional scenario, as already explained above. We thus improved the existing heuristic algorithm to make it more suitable for SFC deployment in the metro/access segments: we improved it to be *capacity-aware* (in terms of computational power), in order to better make use of the constrained computational resources available at the edge of the network. We also realized that the algorithm proposed in [9] can be further improved to reduce, in average, end-to-end latency for embedded SFCs: we refer to this performance improvement asserting that the new proposed heuristic algorithm is *latency-aware*. Also this property leads to benefits in the metro/access network scenario considered in this paper, especially for latency-sensitive SFCs carrying local traffic, i.e., traffic remaining in the metro/access segments. Finally, we removed the unrealistic unconstrained bandwidth assumption of [9], and we improved the heuristic algorithm to achieve load balancing over the metro/access links of the network, thus ensuring that the limited available bandwidth in the metro/access network is best used to guarantee satisfactory Internet services.

To recap, the main contributions of this work are as follow:

- We consider heterogeneous sets of Internet services (i.e., SFCs) to be provisioned in a metro-regional network, and evaluate the benefits of Fixed and Mobile Convergence in terms of consolidation of *NFV-nodes*[1] (i.e., in reducing the number of NFV-nodes that host at least one VNF).
- We designed a new heuristic algorithm, tailored to a metro-regional scenario. It considers latency and capacity metrics to choose the most promising NFV-nodes for the placement of VNFs, with the objective of maximally consolidate them (i.e., place them in the minimum number of NFV-nodes). We evaluate its performance (exploiting capacity and latency awareness properties both individually and jointly) in terms of NFV-nodes consolidation and SFC acceptance ratio, showing the benefits with respect to a previous algorithm especially in a scenario of constrained bandwidth.

We run multiple experiments considering different DC locations and traffic scenarios. We show that for some latency-sensitive Internet services the involved VNFs must be distributed across the edge to avoid a degradation of the service quality, and that FMC solutions help better consolidate VNFs in a smaller number of NFV-nodes. Additionally, our findings show that latency- and capacity-aware VNF placement can improve consolidation up to a factor of 4 and increase acceptance ratio by more than 50% with respect to the state of the art [9].

The rest of the work is organized as follows: Section 2 reviews existing works related to the problem of VNF placement and SFC embedding. Section 3 recaps the system model and provides detailed description of all the parameters and assumptions considered in the modeling. Section 4 describes the proposed improved heuristic and shows an illustrative example. We present the case-study for our numerical analysis and discuss numerical results in Section 5. In Section 6 we draw the conclusion of our study.

2. Related work

The problem of Service Function Chaining has been widely investigated in recent years. To optimally solve the problem, most of the works provide an integer linear programming (ILP) model considering different objective functions. As in [10], authors propose a column-generation-based ILP model for VNF placement and traffic routing problem in a Wide Area Network (WAN). Their objective is to reduce network resource occupation in presence of large traffic demands, however, impact of latency on placement and routing is not considered in their model. The same problem is investigated in [11] considering a network interconnecting multiple DCs instead of a WAN. Their objective is to increase service acceptance ratio, satisfying QoS requirements of users (in terms

[1] With *NFV-node* we refer to a network node equipped with computing and storage capabilities to host VNFs

of latency and minimum guaranteed bandwidth, among others) and considering different priority levels of service requests. Although, they just consider link propagation latency in their model, while we take into consideration also node latency. Authors in [12] propose an ILP model for VNF placement and chaining in a Content Delivery Networks (CDNs) scenario, to provision added-value services. Their objective is to minimize the operational and communication costs while satisfying latency requirements of the provisioned services. To support VNF placement in large-scale CDNs with a large number of servers and end-users, they also propose a cost-efficient heuristic algorithm. Authors in [13] provide a model for VNF placement in a two-tier cloud architecture satisfying maximum tolerated latency for users. However, they do not consider any network resource limitation (in terms of bandwidth) in their model. Authors of [14] propose an MILP model and a heuristic algorithm based on genetic algorithm to schedule the activation of already-placed VNFs processing traffic of heterogeneous services in one or multiple DCs. Their objective is to minimize the scheduling delay. However, unlike our assumptions, in this work *(i)* traffic routing among VNFs and *(ii)* impact of processing-resource sharing delay (competing in node latency increase) is not considered. Ref. [15] also provides a model to solve the VNF scheduling problem jointly with VNF mapping and traffic steering problems, assuming that just one VNF can be placed on each NFV-node; in our work, we remove this unrealistic assumption. The model proposed in [16] performs instead VNF placement with the objective of minimizing utilization of links in the network. Ref. [17] provides an ILP model for reducing energy consumption of online SC across multiple clouds.

As solving ILP and MILP problems is not scalable, some of the existing studies have focused in the design of heuristic algorithms to solve the VNF placement problem. For example, in [18], authors present algorithms to perform a load-balanced Service Function Chaining in the network. Authors in [19] formulate the problem of VNF placement and propose an algorithm to minimize introduced network latency in cloud DCs considering both network and computational resource limitations. Ref. [20] proposes a heuristic approach to consolidates VNFs as much as possible and find shortest paths with less delay to map a service chain. Authors in [21] provide a heuristic algorithm to map a service chain in a multi-domain network.

Moreover, some works in literature investigate the case in which the traffic profile in the network is not always static. For example, in [7] the authors test some proposed approaches for jointly allocating bandwidth and computing resources in a realistic metro scenario by also considering users movements. Ref. [22] provides a column-generation-based ILP model to solve the problem of dynamic VNF placement with the objective of maximizing the profit for the service provider. An algorithm for dynamic VNF placement is proposed in [23] considering a network with multiple hierarchical levels and VNFs with different types, assuming that each VNF type is placed in a different network level.

To the best of our knowledge, no previous work has specifically investigated the impact of capacity- and latency-aware VNF placement on Service Function Chaining in FMC metro networks. In other words, in our work, the decision

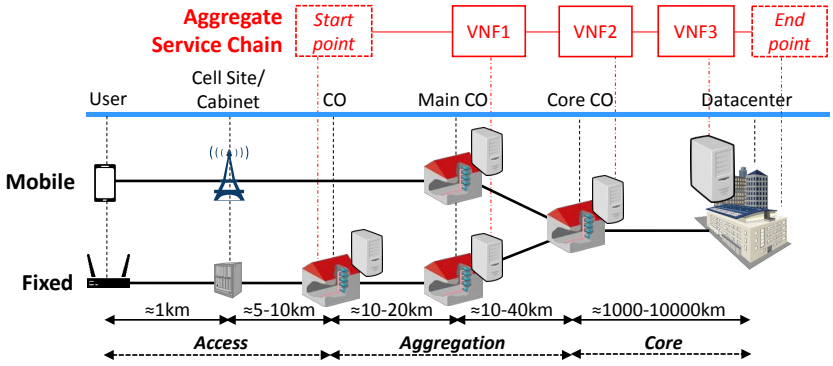


FIGURE 1: Example of NFV-enabled fixed and mobile metropolitan network

about where to place a VNF in the network is made considering residual computational capacity of NFV-nodes (in terms of CPU cores) and latency of the physical path to reach to each NFV-node. Moreover, very few works, among which [7], focus on a metropolitan network scenario, and none of them evaluates the benefits of Fixed and Mobile Convergence on VNF consolidation with respect to a scenario where FMC is not achieved.

3. System model

Our model for a NFV-enabled network architecture includes a representation of (see Fig. 1): *(i)* a physical fixed and mobile metro/access network composed of a set of physical nodes (including NFV-nodes) and bidirectional links, *(ii)* a set of VNFs that must be deployed in NFV-nodes and *(iii)* a set of Internet services implemented as chains of VNFs (i.e., SFCs) to be provisioned over the physical network. The next subsections report the detailed models of the network, the SFCs and the VNFs, including also aspects related to processing-resource sharing in NFV-nodes.

3.1. Fixed and mobile metro/access physical network

We model the physical network as a directed graph $G = (V, E)$ where V and E denote the set of physical nodes and physical links, respectively. Physical nodes $v \in V$ forward traffic, while a subset $n \in N_{nfv} \subseteq V$ includes the NFV-nodes equipped with computing capacity (i.e., with a certain number of CPU cores [9]) to run instances of VNFs. Such nodes introduce additional processing-resource sharing latencies, as we will explain later. Physical links $(v, v') \in E : v, v' \in V$ are equipped with a limited amount of bandwidth capacity, denoted with $\beta_{v,v'}$, and introduce a latency $\lambda_{v,v'}$. Such latency includes both propagation and network devices transmission delay (incurred e.g. in switches). Figure 1 shows an example of physical network, spanning from the users to a remote DC. The main candidate NFV-nodes to host VNFs in the metro/access segments are the COs at different hierarchical levels of the fixed and mobile aggregation

networks (i.e., COs, Main COs and Core COs in Fig. 1). Note that *(i)* in this example we model the metro/access network as a tree, but COs could also be connected by hierarchical access/metro rings and *(ii)* we model the core network (connecting the metro/access segments to the remote DC) as a link with very high (or infinite) bandwidth capacity β and non-negligible latency λ.

3.2. Service Function Chains

Similarly to [9] we assume a one-to-one correspondence between an Internet service and a SFC. For the sake of simplicity we assume that all the SFCs c can be represented through a chain graph $C^c = (X^c \cup U^c, G^c)$ where X^c represents the set of start-/end-points (that are mapped to physical nodes), $u \in U^c$ represents the set of VNF requests u and G^c denotes the set of virtual links (u, u') chaining consecutive VNF requests $u, u' \in U^c$. Note that as in [9] we decouple the type of VNF $f \in F$ from the VNF requests $u \in U^c$ and relate these two concept using an input parameter $\pi_u^c = f$ to indicate that VNF request u of SFC c requests VNF of type f. In our assumptions, the end-point for a SFC can be either the remote DC or a Core CO in the metro/access segment. Terminating a SFC in the Core CO implies that such node can provide all the needed DC-like functionalities to deploy the specific services offered by SFCs. This assumption is in line with current trends of re-architecting COs as datacenters [24].

Moreover, operators generally deploy SFCs to process aggregated traffic (i.e., from multiple users) to provide a specific Internet service. In our model of metropolitan network, we consider that each provisioned SFC c serves an aggregated number of users which increases from COs to the Core COs or to the remote DC, thus demanding more and more bandwidth resources. Note that, in general, SFCs are associated with two types of performance constraints:

- *Bandwidth resources* $\delta^c_{u,u'}$: refers to the amount of bandwidth requested for embedding of virtual link (u, u') that concatenates VNF requests u and u' of service function chain c. Note that $\delta^c_{u,u'}$ depends on the number of users requesting SFC c.

- *End-to-end tolerated latency* φ^c: each SFC c is associated with a maximum end-to-end tolerated latency that must be guaranteed to avoid service degradation between start-/end-points of the SFC c.

3.3. Virtual Network Functions

We consider a VNF as an abstract object that performs some processing on ingress traffic. Each VNF f requires a given amount of processing capacity per user, denoted with c_f. The total amount of computation required for a VNF request u where $\pi_u^c = f$ is the product of c_f and the number of users whose traffic is aggregated in the SFC c. Similarly to [9], we model such capacity request as a fraction of the total amount of CPU cores.

3.4. Modeling of processing-resource sharing latencies

As mentioned earlier, we consider that both physical links and NFV-nodes introduce some latency in crossing an SFC from start- to end-point. In particular, the latency introduced by NFV-nodes is due to sharing of processing resources among multiple VNFs hosted by the same NFV-node. For each SFC, we define σ^c to account for both *context switching* and *upscaling* latencies introduced by processing-resource sharing in each NFV-node [9].

We identify *context switching*, i.e., the operation of saving/loading the state for parallel execution of the multiple VNFs sharing the NFV-node, as the primary source of latency in the traversal of NFV-nodes. We also consider an *upscaling* latency, deriving from the fact that a VNF placed in an NFV-node can require more processing than the one provided by a single CPU core. This happens when the VNF must process a high quantity of traffic (e.g., when a high number of users share such VNF). In this case, the network traffic handled by that VNF must be balanced among different CPU cores involved in the processing, and the new layer of load balancing is responsible for the upscaling latency. In fact, every VNF that is hosted by the NFV-node needs a dedicated load balancer that takes the decision on how the traffic is distributed among the CPU cores: the load balancer can be seen as an auxiliary VNF performing the specific task of balancing traffic among the CPU cores. Hence, it needs some time to take the decision about how to steer the traffic. Context and upscaling latencies could be not negligible and shall be explicitly accounted, especially in a consolidation scenario as the one considered in this paper. For more details on upscaling and context switching latencies the reader should refer to [9].

4. Latency- and capacity-aware SFC provisioning

In this section we propose our heuristic algorithm for VNF embedding and SFC provisioning in metro/access networks. As already discussed, the objective of the algorithm is to *maximally consolidate* the VNFs, i.e., placing them in the minimum number of NFV-nodes while meeting *(i)* physical network constraints (in terms of consumed bandwidth and consumed CPU cores capacity), *(ii)* SFCs constraints (in terms of requested bandwidth and maximum tolerated latency) and *(iii)* VNFs constraints (in terms of required processing capacity).

Note that presenting a mathematical formulation for the VNF embedding and SFC provisioning problem is out of the scope of this paper. Such a formulation can be found in [9]. Instead, in the following we explain why latency and capacity awareness are important criteria to best solve the considered problem. We also present our proposed heuristic algorithm and how it differs from the state-of-the-art solution it takes inspiration from, presented in [9].

4.1. Latency and capacity awareness criteria for VNF request embedding

When a SFC needs to be provisioned in the network, one of the most important tasks is determining where to embed its chained VNF requests. When

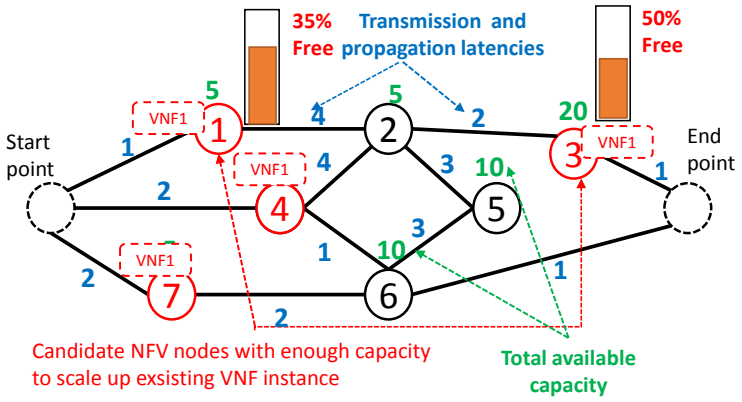


FIGURE 2: Latency Awareness (LA) and Capacity Awareness (CA)

deciding the best location to embed a VNF request, multiple instances of the desired VNF can be active simultaneously in the network and multiple NFV-nodes with enough processing capacity to host a new VNF instance can be available. Hence, a criterion to select the best location for the current VNF request is needed. We consider two different criteria, namely *Latency Awareness* (LA) and *Capacity Awareness* (CA), to select the best VNF instance to be scaled-up or, if no existing VNF instance is found, to select the best NFV-node where to activate a new instance.

In Fig. 2 we show an illustrative example to explain LA and CA. We consider a physical network composed of 9 nodes, two of which are the start- and end-point of the SFC to be embedded. Latency of each physical link is reported in blue close to the link. Such latency values are used as link weight in the shortest path computation used by our defined heuristic algorithm, described in detail in the next subsection.

Suppose that VNF1 is the first chained VNF of a SFC, and that four instances of VNF1 can be scaled up (i.e., are hosted in NFV-nodes with enough capacity for the scaling operation) and are already running in NFV-nodes 1, 3, 4, and 7 (called *candidate NFV-nodes*). To provision the SFC, we need to find a path to transport traffic from the start-point to the end-point that traverses one NFV-node hosting an instance of VNF1. One straightforward criterion is selecting the NFV-node with lowest latency distance from the start-point, which corresponds to NFV-node 1 in the example. We refer to such criteria as *source-based* selection: this criterion is adopted in the algorithm proposed in [9].

However, a fully *latency-aware* (LA) approach should take into account both the total latency cumulated from the source of the SFC to any candidate NFV-node, and from the candidate NFV-node to the destination of the SFC. This "end-to-end" *latency metric* should be calculated for all the candidate NFV-nodes, and the NFV-node associated to the minimum value should be selected.

From a routing perspective, this latency metric calculation is done by summing the latency introduced by the chosen path from the start-point (or previous chained VNF) to the considered candidate NFV-node and from the NFV-node to the end-point. By computing such metric for the example shown in Fig. 2, NFV-nodes 1, 4 and 7 return values equal to 8, 4 and 5, respectively. Therefore, a latency-aware placement would select NFV-node 4 as the best node to scale-up VNF1, instead of NFV-node 1 (as happens instead with the source-based approach). Latency awareness allows to improve the embedding performance, since it allows to take the best end-to-end embedding decision, instead of only considering information related to the source node (or previously-embedded VNF, if the considered VNF is not the first chained VNF).

We now introduce the concept of *capacity awareness* (CA), which can operate in conjunction with the latency awareness criterion. We assume in our example (Fig. 2) that each node is equipped with some CPU cores (denoted in bold green) and that NFV-nodes 1 and 3 host a running instance of VNF1. Note that both NFV-nodes 1 and 3 have the same value of the considered latency metric, i.e., they are equivalent from a latency awareness perspective. Hence, latency awareness criteria would randomly select one of the two nodes. To further improve our node selection, we introduce *capacity awareness*, which, in case of multiple NFV-nodes with equal latency metric value, selects the node with the highest overall capacity in terms of CPU cores. In the example presented in Fig. 2, NFV-nodes 1 and 3 are equipped with 5 and 20 CPU cores. Hence, the capacity awareness would select NFV-node 3. In this way, we ensure that nodes with less available processing capacity are not saturated and could be used by following SFCs that must be embedded and that really need the available processing resources available on that node (e.g. because they require a very low latency that can be guaranteed only by that node).

Note that a capacity-aware selection of NFV-nodes can occur also without considering any latency aspect, i.e., neither latency-aware nor source-based node selection approach. In this specific case, a *capacity awareness* (CA) criterion adopts an NFV-node selection strategy that chooses for VNF embedding the NFV-node with *highest residual computational capacity*. Additionally, unlike what has been shown until now, it would also be possible to apply capacity-aware criteria *before* latency-aware ones. However, in this paper we decided to prioritize latency awareness over capacity awareness because, for a network operator, it is easier to have control on node capacity (i.e., it can always add new processing in an NFV-node) than on latency of SFCs to be provisioned (i.e., if SFC end-to-end latency is not met, the operator may need to compute a new solution and could incur in loss of revenues).

4.2. Algorithm description

The algorithm we propose takes inspiration from the algorithm presented in [9], called in the remainder of this paper simply *heuristic* (H). The pseudo-code presented in Algorithm 1 refers to the case where both LA and CA, as described in the previous section, are considered (i.e., *H-LCA*). Note that, however, other heuristic algorithms can be designed by adopting only LA or CA (i.e., *H-LA*

Table 1: Different heuristic approaches and correspondent criteria for NFV-nodes selection

	Source-based	Latency-awareness (LA)	Capacity-awareness (CA)
H [9]	✓	✗	✗
H-LA	✗	✓	✗
H-CA	✗	✗	✓
H-LCA	✗	✓	✓

and *H-CA*). The complete set of heuristic approaches and the correspondent NFV-nodes sorting criteria are summarized in Table 1.

In the following, we describe in detail only *H-LCA* (see Algorithm 1). In fact, *H-LA* and *H-CA* logic is very similar to the logic of *H-LCA*, and we believe there is no need for a detailed description of such heuristic algorithms, since they only differ in the NFV-nodes selection procedure for VNF placement, as discussed in the previous subsection. The *H-LCA* heuristic algorithm takes into account end-to-end latency requirements of SFCs to be provisioned and the processing requirement of VNFs composing the SFCs. The main idea is building an embedding solution that tries to exploit already-placed VNFs instances first, by scaling up their resources, before activating new instances. From a high-level perspective, the algorithm works in three distinct phases. For the placement of each VNF, the algorithm checks in *Phase 1* whether an instance of such VNF already exists in the network. In case such instance is found and the NFV-node hosting it has enough capacity, such VNF instance is scaled up to accommodate more traffic. Otherwise, if there is no existing VNF instance to scale up, the algorithm places a new instance (*Phase 2*). This strategy is in line with the objective of provisioning new SFC requests to maximally consolidate VNFs. If the algorithm fails to provision the SFC, then it releases the resources reserved previously and consolidates all the VNFs on the latency shortest path, choosing the NFV-node with highest betweenness centrality value[2] (*Phase 3*). Going more in depth, as first step (line 1 of Algorithm 1), *H-LCA* algorithm sorts in an increasing order the SFCs based on their latency requirements (φ^c) to prioritize the placement of latency-sensitive SFCs. After that, in line 3, the next SFC to provision is chosen. At the beginning of *Phase 1* (line 5), the next VNF request belonging to the chosen SFC is selected and NFV-nodes with an already-placed VNF instance for that VNF request are found in the network (line 6). When more than one VNF instances are available in the network, the algorithm uses the LA and CA criteria presented in the previous subsection to choose among one of them. After choosing the most promising NFV-node, the processing resources of the selected VNF instance f on the NFV-node v (p_v^f), should be scaled up by a value of π_f (line 14). As context switching and upscaling costs both depend on p_v^f (as mentioned in Section 3), their corresponding values in

[2]In graph theory, betweenness centrality is a measure of centrality in a graph. In our case, we consider the betweenness centrality with respect to shortest paths, i.e., its value, for each vertex, is the number of shortest paths that pass through the vertex.

NFV-node v could be increased. Hence, the new NFV-node latency may affect the end-to-end latency of already-embedded SFCs that have one or more VNF

Algorithm 1 Latency- and Capacity-Aware Placement of VNFs (H-LCA)

```
Sort the SFCs by increasing value of latency requirement
repeat
    Pick the next SFC
     \* Phase 1 *\
    repeat
        Pick the next VNF request in the SFC
        if ∃ instance of VNF already placed then
            Calculate latency shortest path between current node and VNF instances
            if there is a physical path then
                Insert the VNF instance in the list of valid VNF instances
            end if
            Sort the valid VNF instances by increasing latency metric (i.e., latency awareness)
            if multiple instances have the same value of latency metric then
                Select the one with highest processing capacity (i.e., capacity awareness)
                Try to scale up the VNF instances until success or all VNF instances tried
                if success then
                    continue
                end if
            end if
        end if
        if no valid VNF instance available then
        \* Phase 2 *\
            Calculate latency shortest path between current node and NFV-nodes
            if there is a physical path then
                Insert the NFV-nodes in the list of valid NFV-nodes
            end if
            Sort the valid NFV-nodes by increasing latency metric (i.e.,latency awareness)
            if multiple NFV-nodes have the same value of latency metric then
                Select the one with highest processing capacity (i.e., capacity awareness)
                Try to add VNF instance on an NFV-node until success or all NFV-nodes tried
            end if
            if failed then
                return (infeasible)
            end if
        end if
    until all the VNF requests are chained
     \* Phase 3 *\
    Check end-to-end latency of the embedded SFC against requirement
    if success then
        continue
    end if
     \* Failed *\
    Release resources allocated in Phase 1 and Phase 2
    Place the VNFs on the shortest path in the NFV-node with highest betweenness
    Check end-to-end latency of the embedded SFC against requirement
    if failed then
        return (infeasible)
    end if
until all SFCs are embedded
return(success)
```

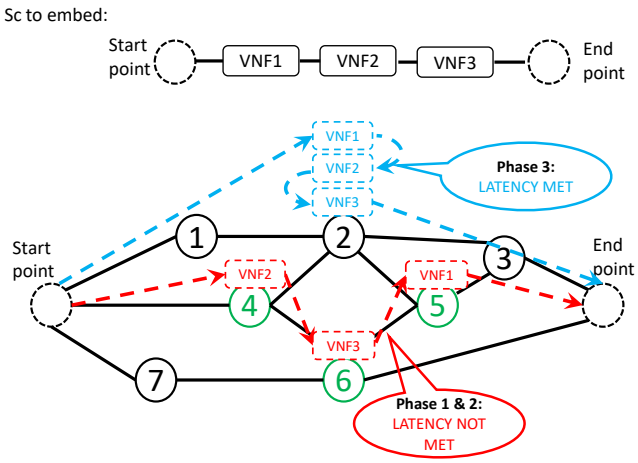


FIGURE 3: Pictorial example of the proposed heuristic algorithm (*Phase 1*, *Phase 2* and *Phase 3* execution)

requests mapped to this NFV-node v. If this is the case, this VNF instance cannot be used and next VNF instance in the sorted list of VNF instances is checked (line 14). If the scale up succeeds, the mapping of the VNF request to the VNF instance in NFV-node v is done and the latency shortest path between *current node* (which is either the start-point, if the VNF requests is the first traversed in the SFC, or the NFV-node where the precedent VNF requests has been mapped) and the selected NFV-node v is used to route the SFC aggregated traffic. If this process returns failure for all the NFV-nodes with already active VNF instances, the algorithm starts *Phase 2* and sorts the remaining NFV-nodes based on the criteria shown in Table 1.

Once NFV-nodes are sorted, the algorithm tries to place a new VNF instance with processing requirement π_f on the first NFV-node v on the list (line 28). As mentioned before, scaling up of processing resources on an NFV-node leads to an increase in the context switching/up-scaling latencies, so also at this step, the algorithm checks whether the end-to-end latency for already-embedded SFCs that use one or more VNF requests mapped to v is still satisfied or not. The algorithm also checks if the residual processing capacity of the NFV-node v is enough to host the new VNF instance. Then, if the scale-up operation succeeds, the VNF request is mapped to the newly-activated VNF instance and the latency shortest path between *current node* and the selected NFV-node v is used to route the SFC aggregated traffic. Note that the operation reported in line 28 implies that, if residual processing capacity of already-active NFV-nodes is not enough, the algorithm can also activate a new NFV-node.

After embedding all the VNF requests of a SFC c in NFV-nodes (line 34), the NFV-node where the last VNF request has been mapped is connected to the end-node through the latency shortest path and SFC is provisioned. At this point, *Phase 3* starts. The algorithm checks end-to-end latency requirement of

the embedded SFC c. If the total latency accumulated on physical links and on all the traversed NFV-nodes exceeds the latency requirement of the SFC c, then the resources previously reserved are released and all VNF requests of the SFC are consolidated and activated on the NFV-node v with highest betweenness centrality along the latency shortest path between start- and end-points. Otherwise, if no NFV-node is available for VNFs placement, the request is infeasible and the SFC cannot be provisioned. *Phase 3*, aside from ensuring that SFC end-to-end latency is met, also helps escape from local optima that could be reached when greedily embedding SFCs by means of *Phase 1* and *Phase 2*.

Note that, in case of limited bandwidth, the latency shortest path, whether computed in *Phase 1* or *Phase 2*, must have enough bandwidth resources to carry the processed traffic to the selected VNF instance. Hence, in both lines 11 and 25, the VNF instances are considered valid (i.e., can be used) if and only if there exists a physical path with enough resources to carry the traffic from the *current node* to the NFV-node where the selected VNF instance is running. To better clarify the embedding procedure, Fig. 3 shows a pictorial view of the execution of the three different phases of the algorithm.

4.2.1. Computational complexity analysis

Our proposed algorithm leverages the Dijkstra's algorithm for the computation of the latency shortest path between NFV-nodes, whose worst-case complexity in its simplest implementation is $\mathrm{O}(|N_{nfv}|^2)$ [25]. For each SFC $c \in C$, the latency shortest path is computed at most $|U^c|+2$ times: $|U^c|+1$ in *Phase 1/Phase 2* and 1 in *Phase 3*. Since heterogeneous SFCs with different numbers of chained VNFs (i.e., $|U^c|$) can be embedded, in our computational complexity evaluation we consider the worst case, i.e., where all the SFCs $c \in C$ chain the maximum number of allowed VNFs, i.e., $|U_{max}|$. Given all the above considerations, the computational complexity of our approach is $\mathrm{O}(|C| \cdot |U_{max}| \cdot |N_{nfv}|^2)$. Note that computational complexity could be reduced by using a more efficient implementation of the Dijkstra's algorithm [25], which has as worst-case computational complexity $\mathrm{O}(|E|+|N_{nfv}| \log |N_{nfv}|)$ when $N_{nfv} = V$, i.e., when all the physical nodes V are NFV-nodes. In this case, the computational complexity of our approach would be $\mathrm{O}(|C| \cdot |U_{max}| \cdot |E| + |C| \cdot |U_{max}| \cdot |N_{nfv}| \log |N_{nfv}|))$.

5. Illustrative numerical results

We have implemented and simulated the proposed heuristic algorithms in Matlab. This section aims at showing a performance overview of the proposed strategies. The section is structured as follows. After recalling the simulation settings, we present numerical results for the considered case study (i.e., when solving the problem of VNF consolidation in a metro/access network) both when Fixed and Mobile Convergence has been achieved and when it has not. In this analysis, we adopt H to solve the embedding problem. Then, we show evaluation results showing the goodness of our novel heuristic approaches (i.e., *H-LA*, *H-CA* and *H-LCA*) with respect to H, highlighting the improvement introduced

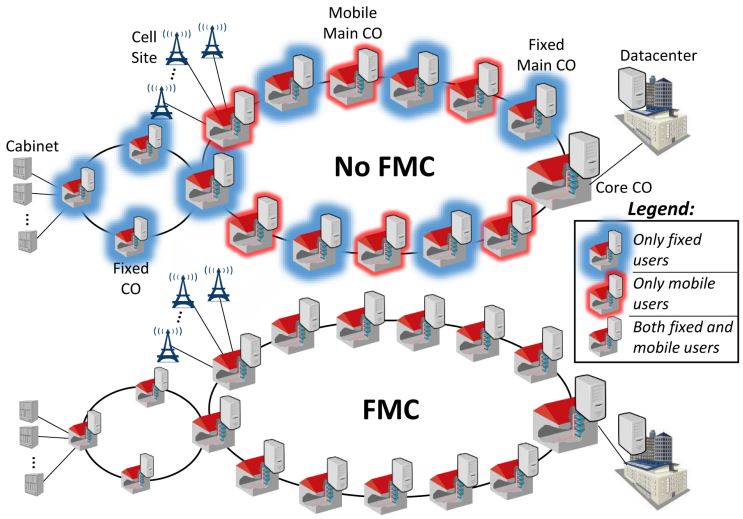


FIGURE 4: NFV-node accessibility in FMC and No FMC architectures

by adopting more sophisticated metrics in the selection of NFV-nodes. For each of the simulation settings presented in the following subsections, we simulate 1000 runs, and we show the obtained average result for each considered metric.

5.1. Simulation settings

5.1.1. Metro/access network topology

We consider the metro/access network topology shown in Fig. 4. Such network topology is based on the urban geotype proposed in [26]. In particular, it consists of 1 Core CO, 6 fixed Main COs and 6 mobile Main COs, each covering an area of 15 km^2. Each fixed Main CO aggregates the traffic of 3 fixed COs, all connected in a ring. Each fixed CO in turn aggregates the traffic of 95 cabinets, and a cabinet is able to aggregate the traffic of different users, i.e., homes. For the mobile network, each mobile Main CO aggregates the traffic of 23 cell sites, each one aggregating the traffic of 3000 mobile users, unless otherwise specified. The total coverage area of this network is in the order of the surface of a large Italian metropolitan city. The Core CO and fixed/mobile Main COs are connected in a ring network. We model the core network as a single link whose latency reproduces the total latency experienced to reach the DC location. The Core CO and Main COs can host COTS hardware with limited processing capacity (i.e., they can be considered as NFV-nodes), while the DC has infinite processing capacity. Unless otherwise specified, the Core CO is assumed to have twice the processing capacity of the fixed/mobile Main COs, while the COs have 30% of the capacity of the Main COs due to physical space limitations.

We consider the two network architectures introduced above: *FMC* and *No FMC*. In the *No FMC* architecture, fixed users can access only VNFs that are

placed into the fixed network infrastructure, i.e., in fixed Main COs and fixed COs (vice versa for mobile users). In addition, the Core CO and the DC can be accessed by both fixed and mobile users. On the other hand, in the *FMC* architecture, fixed and mobile users can share the network infrastructure and thus also the NFV-nodes. We consider 3 different DC location configurations: *Close DC*, *Mid-range DC* and *Very Far DC* with user-to-DC latencies equal to 15, 75 and 150 ms, corresponding to a national, continental and intercontinental DC location. These latencies are based on measurement performed using [27], a tool to evaluate the ping distance from Amazon Web Services around the globe, and express the one-way latency to the DC location from the Core CO. Note that, given the small geographical coverage of the metro/access network (see Fig. 1), the latency of physical links is negligible with respect to the latency introduced by the core network. However, in our analysis we consider also the *context switching* latency introduced by an NFV-node, which increases linearly with respect to the number of VNFs sharing such NFV-node and is equal to 400 μs per VNF [28], while we consider the *up-scaling* latencies negligible, unless otherwise specified.

5.1.2. Service Function Chains and Virtual Network Functions

We consider a set of 5 different SFCs, as shown in Table 2. Each SFC type is associated to a different end-to-end *latency requirement* φ^c. The Web Service (WS) is recognized to have a loose latency requirement, while novel 5G Services (5GS or 5G, e.g. Augmented Reality) require very strict end-to-end latency [29]. We consider also other 3 SFC types, i.e., VoIP, Video Conferencing (VC) and Cloud Gaming (CG). Each VNF f is associated to a *processing requirement* π_f per user, obtained by some middleboxes datasheet (see Table 3).

A SFC serves a set of aggregate users and the start-/end-points for each fixed or mobile SFC are fixed in the network. In case of a non-convergent metro/access network (*No FMC*) the start-points for fixed (mobile) SFCs are fixed Main CO and CO (mobile Main CO), for an overall number, according to the topology of Section 5.1.1, of 30 SFCs. Each SFC aggregates the traffic of the users connected to the fixed/mobile Main CO or fixed CO it starts from. Whereas, in case of FMC, there is no distinction between fixed and/or mobile COs/Main COs, hence we make no distinction between fixed and mobile users. Concerning the end point, we compare three different settings with a different percentage of local traffic terminating in the metro network: *0%*, *50%*, *100%*. The first setting (*0%*) represents the case where all the SFCs have as destination point the DC location. In the second setting (*50%*), half of the SFCs have as destination the Core CO in the metro network and the remaining half terminate at the DC location. Finally, in the last setting (*100%*) all the SFCs terminate at the Core CO (i.e., at the edge of the metro network).

Note that considering a fraction of SFCs that terminate in the metro network follows the current trend of telecom operators, which tend to push the content towards the users. For example, a Video Content Provider, may place Video Servers in the metro/access network (i.e., in our case, in a *micro DC* located in the Core CO).

Table 2: Performance requirements for the SFCs per user [9]

SFCs	Chained VNFs	Latency req. (ms)	Bandwidth req. (Kb/s)
WS	NAT-FW-TM-WOC-IDPS	500	500
VoIP	NAT-FW-TM-FW-NAT	100	100
VC	NAT-FW-TM-VOC-IDPS	80	4000
CG	NAT-FW-VOC-WOC-IDPS	60	500
5GS	NAT-FW-TM-WOC-VOC	20	1000

Table 3: Processing requirement in terms of fraction of CPU core (per user) for the VNFs [9]

Virtual Network Function	π^f
Network Address Translator (NAT)	0.00092
Firewall (FW)	0.0009
Traffic Monitor (TM)	0.0133
WAN Optimization Controller (WOC)	0.0054
Intrusion Detection Prevention System (IDPS)	0.0107
Video Optimization Controller (VOC)	0.0054

5.2. Effect of FMC on VNF consolidation

We compare the results obtained by considering homogeneous scenarios, i.e., only one specific type of SFC among WS, VoIP, VC, CG and 5GS is provisioned in the network. This way, we can independently evaluate the impact of different Internet services on VNF consolidation. In the first simulation scenario, we investigate the benefit of FMC on VNF consolidation by comparing the results obtained in case of *FMC* against *No FMC* architecture. Such comparison has been carried out adopting the state-of-the-art heuristic (H), while taking into consideration the three different local traffic percentages and the three DC locations scenarios introduced in Section 5.1.2, and by assuming *unlimited bandwidth* resources in the physical links.

In Fig. 5 we present the number of active NFV-nodes when adopting the H algorithm for SFC provisioning in *FMC* and *No FMC* architectures. For the *Close DC* configuration, we observe that the most convenient solution in terms of VNF consolidation is to host all the VNFs in the DC. This results hold for every homogeneous scenario, every architecture and every percentage of local traffic, except for the 5GS SFCs for *50%* and *100%* of local traffic. In these cases, it is required the activation of some fixed/mobile Main COs and of the Core CO. In fact, for all the SFC types except the 5GS, consolidating the VNFs in the DC, even though part or all of the traffic is kept local, is a possible solution because the Round-Trip Time to the DC (30 ms) does not affect the latency requirement of the SFCs. This is not true for the 5GS homogeneous scenario due to the very strict latency requirement of its SFCs (20 ms). In this case and in conditions of local traffic, placing all the VNFs in the DC would degrade the performance. For this reason, it is necessary to distribute the VNFs in the metro network to meet latency requirements for the SFCs terminating in the Core CO.

For the *Midrange DC* configuration, only the VNFs for the WS scenario can be all consolidated into the DC. The VoIP and VC homogeneous scenarios

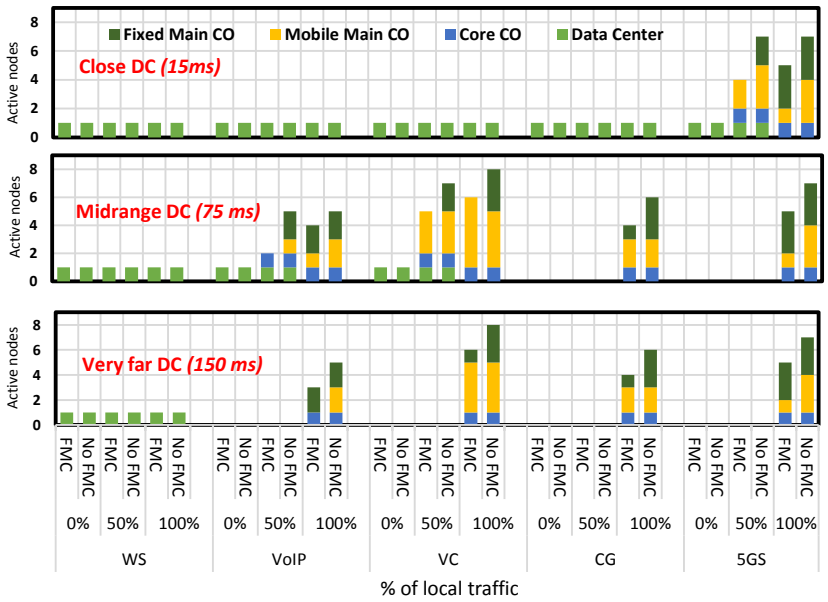


FIGURE 5: Number of active NFV-nodes in the *FMC* and *No FMC* architectures under three different DC location configurations and under three different local traffic percentages

require the activation of NFV-nodes in the metro/access network to meet the latency requirement when part of the traffic is kept local (*50%* and *100%*). Moreover, in the CG and 5GS homogeneous scenarios, no feasible solution (i.e., no active NFV-nodes in Fig. 5) exists when part of the traffic is *not* local (*0%* and *50%*), since the latency to reach the DC (75 ms) is alone higher than the latency requirement for CG (60 ms) and 5GS (20 ms). In such case, the only feasible solution is keeping all the traffic local (*100%*) and distributing the VNFs in the metro/access NFV-nodes.

Finally, for the *Very Far DC* configuration, only the WS homogeneous scenario can still be guaranteed by placing the VNFs in the DC for all the percentage of local traffic conditions. For the other scenarios, the latency requirement forces all the VNFs to be placed in the metro/access network and to keep all the traffic always local (*100%*), since all the other settings are infeasible. In general, from Fig. 5 we can see that the impact of latency requirement on VNF consolidation is similar for the *FMC* and *No FMC* architectures. However, when the VNFs are distributed in the metro/access network, the *FMC* architecture requires 30% to 60% less NFV active nodes than the *No FMC* one. This happens because in the *FMC* architecture the NFV-nodes as well as the VNFs placed on those nodes are shared between fixed and mobile users. Hence, adoption of a FMC metro/access network can consistently improve consolidation of VNFs.

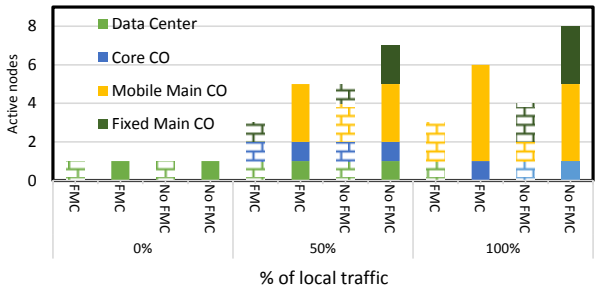


FIGURE 6: Number of active nodes for VC with increased processing capacity for the NFV-nodes by 50% (standard bars) with respect to the settings of Fig. 5 (brick shaded bars)

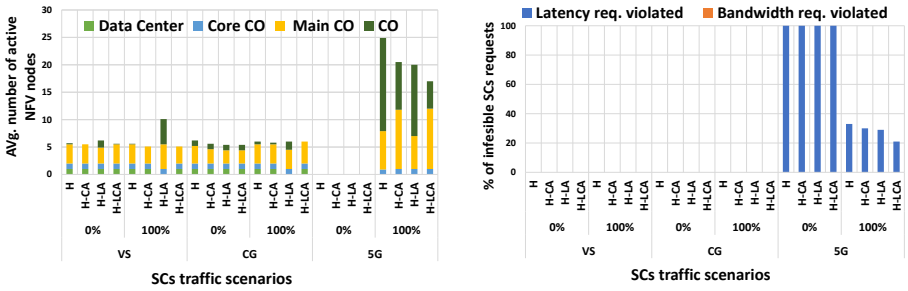
5.2.1. Impact of increased available computational capacity

We now focus on some *processing* aspects. Considering Fig. 5 for the *Midrange DC* case, we can notice that the VC homogeneous scenario requires the activation of more NFV-nodes than the other scenarios. This happens because, in average, the VNFs chained by VC have a higher processing requirement than the other types of SFCs. Moreover, the placement of VNFs is slightly different when NFV-nodes have more processing capacity. In Fig. 6 we compare the results for the most processing-hungry service (i.e., the VC), obtained with the previous simulation settings (i.e., brick shaded bars), with the case where the processing capacity of the NFV-nodes is increased by 50% (i.e., standard bars). The increase of processing capacity allows to place the VNFs in less NFV-nodes (from 30% to 40%) for both the *FMC* and *No FMC* architectures. This means that increasing the processing capacity of NFV-nodes, when possible, is beneficial for VNF consolidation.

5.3. Effect of latency and capacity awareness on VNF consolidation

After having showed the benefits of FMC on VNF consolidation, in this subsection we compare the performance of the proposed heuristic algorithms (see Table 1) in terms of average number of active NFV-nodes and percentage (%) of infeasible runs, including the constraint that leads to infeasibility (i.e., either *latency* or *bandwidth violation*). For this set of experiments we consider one DC configuration (short-range, i.e., *Close DC*) and two local traffic percentages (*0%* and *100%*). We show numerical results for the most latency-sensitive SFCs, i.e., VS, CG and 5G. We consider the metro/access network topology shown in Fig. 4, focusing on the FMC architecture. We compare the performance of the heuristic algorithms in Table 1 under three different *network scenarios*:

1. (Network scenario 1) It considers *unlimited bandwidth* resources and routing to concatenate two consecutive VNFs is performed by selecting the latency *shortest path* (SP).
2. (Network scenario 2) It considers *limited bandwidth* resources and routing is performed by selecting the latency *shortest path* (SP) with enough resources.



(a) Average number of active NFV-nodes (b) Percentage of infeasible SFCs requests

FIGURE 7: *Unlimited bandwidth* resources and *shortest path* routing (network scenario 1)

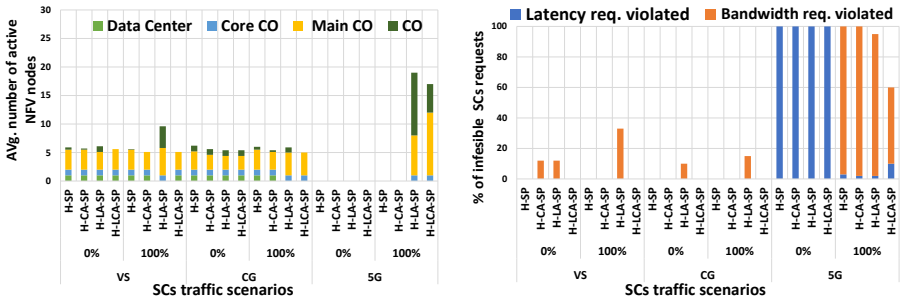
3. (Network scenario 3) It considers *limited bandwidth* resources and the *least-loaded path* (LLP) is selected for routing traffic among VNFs. The least-loaded path is the path with less bottleneck capacity, i.e., the path whose bottleneck link (i.e., link with minimum residual bandwidth of the path) has the maximal residual bandwidth with respect to the bottleneck links of other paths. This way, the load on physical links is balanced.

We extend the heuristic algorithm acronyms with *SP* and *LLP* to refer to network scenarios 2 and 3. Note instead that network scenario 1 is the standard scenario, i.e., the one considered in all the other subsections of the paper, and thus does not require any acronym extension.

For each of these scenarios, we compare the average number of active NFV-nodes and the percentage of infeasible runs, for each of the four heuristic algorithms listed in Table 1. For these experiments, beside considering context switching latencies, we set upscaling latencies to 0.15 ms [9]; we also assume that the Core CO, Main COs and COs are equipped with 20, 10 and 3 CPU cores, respectively.

5.3.1. Unlimited bandwidth and shortest path routing (network scenario 1)

Figures 7a and 7b show a comparison of the proposed heuristic algorithms. The first observation is that all heuristics return feasible solutions for VS and CG SFCs, independently from the percentage of local traffic. This means that such SFCs can be served in both cases with no performance degradation, both when the end-point (e.g., Game Server) is located in the DC or in the Core CO. Moreover, the latency-aware heuristic (*H-LA*) activates a number of NFV-nodes greater with respect to the other approaches. This effect is very strong for the VS SFC and 100% local traffic scenario, where *H-LA* activates twice the number of NFV-nodes, with respect to the other approaches. On one side, this is due to the fact that VS is a computationally-intensive SFC. In addition, given that LA selects the best NFV-node on latency shortest paths, and those SFCs requests start from the COs, the NFV-nodes that are selected are mainly COs in the same ring, which do not have high amount of processing capacity. This process



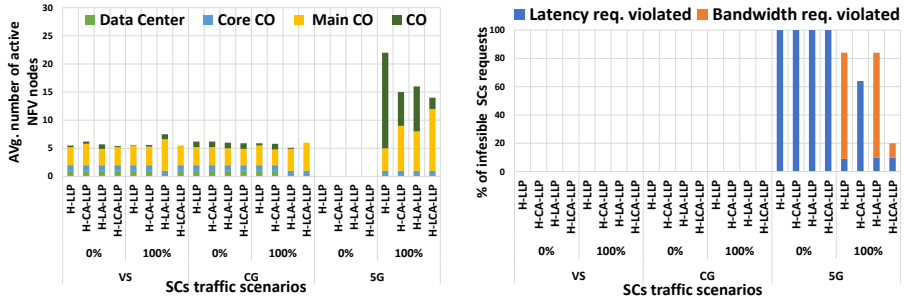
(a) Average number of active NFV-nodes (b) Percentage of infeasible SFCs requests

FIGURE 8: *Limited bandwidth* resources and *shortest path* routing (network scenario 2)

leads to *H-LA* activating a high number of COs, with respect to other proposed approaches, especially for SFCs processing a high amount of traffic.

We now focus on the 5G SFC. First, note that none of the proposed heuristics can tolerate VNFs to be consolidated within the DC when all the traffic is directed to the DC (*0%* of local traffic). All heuristic algorithms return *100%* infeasibility due to unsatisfiable latency requirement. We found the same results in case of *Midrange DC* and *Very Far DC* configurations, but we do not report the results for the sake of conciseness. Whereas in case of *100%* of local traffic, we observe that the source-based heuristic approach (*H*) activates the highest number of NFV-nodes, while *H-CA*, *H-LA* and *H-LCA* activate 17%, 19% and 31% less NFV-nodes. This is due to the fact that *H-CA* privileges more the Main COs than the COs and when coupled with latency awareness (*H-LCA*) achieves the highest consolidation while decreasing infeasibility percentage by 36% with respect to *H*. Note that infeasibility is always due to latency violations, since no bandwidth limitation is considered in this scenario.

5.3.2. Limited bandwidth and shortest path routing (network scenario 2)



(a) Average number of active NFV-nodes (b) Percentage of infeasible SFCs requests

FIGURE 9: *Limited bandwidth* resources and *least-loaded path* routing (network scenario 3)

In this subsection, we limit bandwidth resources on physical links assuming

that the links constituting the external rings, internal ring (connecting Main COs) and the link connecting Core CO to DC in the core network have capacities of 1 Gb/s, 2.5 Gb/s and 10 Gb/s, respectively. We also assume that SFC requests start at COs and aggregate the traffic of 2500 users.

Numerical results, shown in Figs. 8a and 8b, confirm a similar trend for VS and GC SFCs, in terms of average number of active NFV-nodes, with respect to the unlimited bandwidth network scenario. In terms of percentage of infeasible runs, we observe that *H-LA-SP* increases from 12 up to 33 %, whereas all requests are successfully provisioned when the placement of VNFs is both latency- and capacity-aware (*H-LCA-SP*).

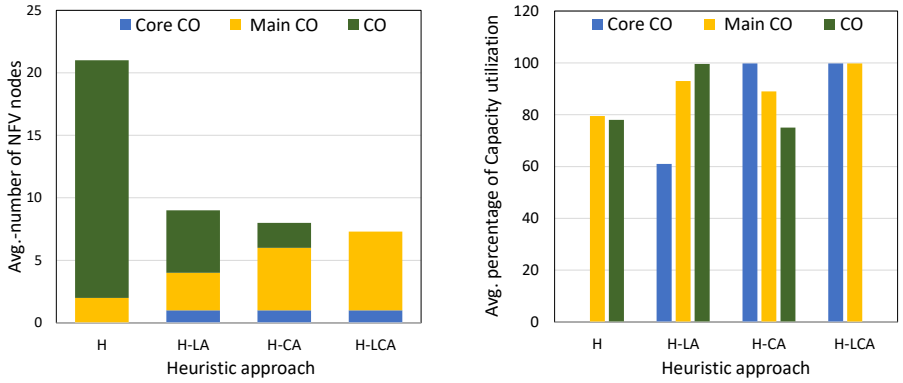
In case of 5G SFC, none of the proposed heuristics provides feasible solutions in the *0%* local traffic scenario, as usual. Conversely, in case of *100%* local traffic, *H-SP* and *H-CA-SP* block 100% of the SFC requests, while *H-LA-SP* and *H-LCA-SP* show an infeasibility percentage equal to 95% and 60% respectively, mainly due to violation of the bandwidth requirement. These results are a further confirmation that VNF placement taking into consideration latency and capacity awareness significantly improves consolidation and acceptance ratio, especially for latency-sensitive SFCs.

5.3.3. Limited bandwidth and least-loaded path routing (network scenario 3)

In this subsection we show the results obtained from the comparison of the proposed approaches, when the least-loaded path (LLP) is selected to concatenate consecutive VNFs in the SFC and bandwidth is limited as in the previous subsection. For VS and CG we observe that, in terms of average number of active NFV-nodes, LLP routing does not have a significant impact, except for VS SFCs with *100%* of local traffic, where the number of NFV-nodes decreases by 22% with respect to the case where routing is performed through the latency shortest paths (shown in Fig. 8a). Moreover, in terms of acceptance ratio, LLP returns 0% infeasibility instead of 33% obtained in case of limited bandwidth with routing on the latency shortest paths (see Fig. 8b). Therefore, LLP is beneficial when targeting for VNF consolidation. In case of latency-sensitive SFCs, LLP allows to increase the acceptance ratio, with respect to the latency shortest path routing strategy, for all the proposed heuristics approaches. However, the highest improvement is obtained in case of *H-LCA-LLP*, where the number of infeasible SFC requests is reduced by 40%.

5.3.4. Effect of latency and capacity awareness on NFV-nodes utilization

Finally, we perform further analysis to confirm the importance of latency and capacity awareness in the process of VNF placement and SFC provisioning. In the previous set of experiments, we observed that latency and capacity awareness help improve VNF consolidation and that this improvement allows to accommodate more SFCs requests, particularly when LLP routing is adopted. Moreover, we found that LA is beneficial when placing VNFs of latency-sensitive SFCs, while CA allows to improve consolidation for bandwidth-intensive SFCs. To confirm these statements, we performed additional experiments, taking into



(a) Average number of active NFV-nodes (b) NFV-nodes capacity utilization

FIGURE 10: Comparison of NFV-nodes capacity utilization

consideration the 5G SFC in the case where *100%* of traffic is local and the DC configuration is *Very Far DC*.

Figures 10a and 10b show the obtained results, for all the proposed heuristics, in terms of average number of active NFV-nodes and NFV-nodes capacity utilization, respectively. Generally, we observe that the source-based approach (*H*) tends to place more VNFs across the edge of the network by relying more on COs than on Main COs, with levels of capacity utilization not exceeding 80%. In *H-LA*, we notice that the Core CO is activated and its capacity used by more than 60%. In addition, *H-LA* activates a greater number of Main COs and less COs, with respect to *H*, with values of capacity utilization equal to 90 and 100%, respectively. The *H-CA* approach tends to privilege more the NFV-nodes with higher computational capacity: for instance, the Core CO is fully loaded. Finally, the combination of both latency and capacity awareness provided by the *H-LCA* approach activates the Core CO and a higher number of Main COs to compensate the fact that COs are not used. This approach achieves the best consolidation in terms of active NFV-nodes and in terms of capacity utilization, since it only uses Core CO and Main COs, leaving COs unloaded and available for future latency-sensitive SFCs, if needed.

6. Conclusion

In this paper, we focused on the impact of latency, bandwidth and processing requirements on the distribution of VNFs in metro/access networks. We considered heterogeneous Internet services and proposed a heuristic algorithm to solve the problem of VNF placement and traffic routing for an effective SFC provisioning, with the objective of maximizing VNF consolidation (thus leading to CapEx and OpEx reduction). We showed that Fixed and Mobile Convergence helps achieve good VNF consolidation and that our proposed heuristic

algorithm best works in a constrained environment in terms of *(i)* NFV-nodes computational capacity and *(ii)* link bandwidth, as a metro/access network is.

To evaluate the benefits of Fixed and Mobile Convergence on VNF consolidation, we compared the results obtained under different remote DC locations and different traffic distributions with respect to a network infrastructure where fixed and mobile networks are functionally and structurally separated. Results showed that a convergence of fixed and mobile networks leads to an improvement in VNF consolidation, requiring up to 60% less active NFV-nodes. Moreover, both in FMC and non FMC metro/access networks, latency-sensitive Internet services (e.g. 5G services) can only be successfully provisioned by pushing the VNFs at the edge of the network, while Internet services with loose latency requirement (e.g. Web Service) can be guaranteed by consolidating all the VNFs in remote DCs, even very far from the users, ensuring benefits to network operators from a cost perspective.

Furthermore, our proposed algorithm was designed to be latency- and capacity -aware in the selection of NFV-nodes for VNF placement, and we proved that latency and capacity awareness are very beneficial in terms of both VNF consolidation and SFC acceptance ratio, especially when jointly adopted, with respect to a state-of-the-art solution that assumes link bandwidth being never a bottleneck and that is neither latency-aware nor capacity-aware. In this case, our proposed approach allows to activate up to 31% less NFV-nodes and to improve SFC acceptance ratio by up to 36%. Our strategy is even more beneficial in a bandwidth-constrained environment, particularly in terms of acceptance ratio: infeasibility of solutions can be reduced, in such constrained scenario, by up to 95%. Moreover, in the same scenario, we also showed that when traffic between VNFs is routed exploiting a load balancing policy instead of a simple latency-shortest-path policy, infeasibility of solutions can be further reduced by 40%, with no (or very reduced) impact on the number of active NFV-nodes. Finally, we showed that latency and capacity awareness allow to ensure the best processing capacity utilization of active NFV-nodes. In fact such criteria, when jointly adopted, avoid computational resource wastage by activating only larger NFV-nodes in the metro/access segment (i.e., Main CO and CO) and by fully utilizing them in terms of processing capacity.

Acknowledgment

This article leading to these results has been supported by the European Community under grant agreement no. 761727 Metro-Haul project funding.

References

[1] M. Savi, A. Hmaity, G. Verticale, S. Höst, M. Tornatore, To distribute or not to distribute? Impact of latency on Virtual Network Function distribution at the edge of FMC networks, in: 18th IEEE International Conference on Transparent Optical Networks (ICTON), 2016.

[2] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, R. Boutaba, Network Function Virtualization: state-of-the-art and research challenges, in: IEEE Communications Surveys Tutorials, Vol. 18, No. 1, pp. 236–262, 2016.

[3] A. M. Medhat, T. Taleb, A. Elmangoush, G. A. Carella, S. Covaci, T. Magedanz, Service Function Chaining in Next Generation Networks: state of the art and research challenges, in: IEEE Communications Magazine, Vol. 55, No. 2, pp. 216–223, 2017.

[4] R. V. Rosa, M. A. S. Santos, C. E. Rothenberg, MD2-NFV: The case for multi-domain distributed Network Functions Virtualization, in: International Conference and Workshops on Networked Systems (NetSys), 2015.

[5] M. Mahloo, J. Chen, L. Wosinska, PON versus AON: which is the best solution to offload core network by peer-to-peer traffic localization, in: Elsevier Optical Switching and Networking, Vol. 15, pp. 1–9, 2015.

[6] M. Savi, R. Fratini, G. Verticale, M. Tornatore, Performance evaluation of video server replication in metro/access networks, in: Elsevier Computer Networks, Vol. 93, pp. 96–110, 2015.

[7] J. Pedreno-Manresa, P. S. Khodashenas, M. S. Siddiqui, P. Pavon-Marino, On the need of joint bandwidth and NFV resource orchestration: a realistic 5G access network use case, in: IEEE Communications Letters, Vol. 22, No. 1, pp. 145–148, 2018.

[8] C. Behrens, S. Krauß, E. Weis, D. Breuer, Technologies for convergence of fixed and mobile access: an operator's perspective, in: IEEE/OSA Journal of Optical Communications and Networking, Vol. 10, No. 1, pp. A37–A42, 2018.

[9] M. Savi, M. Tornatore, G. Verticale, Impact of processing-resource sharing on the placement of chained virtual network functions, in: IEEE Transactions on Cloud Computing, 2019.

[10] A. Gupta, B. Jaumard, M. Tornatore, B. Mukherjee, A scalable approach for Service Chain mapping with multiple SC instances in a Wide-Area Network, in: IEEE Journal on Selected Areas in Communications, Vol. 36, No. 3, pp. 529–541, 2018.

[11] P. Cappanera, F. Paganelli, F. Paradiso, VNF placement for Service Chaining in a distributed Cloud environment with multiple stakeholders, in: Elsevier Computer Communications, Vol. 133, pp. 24–40, 2019.

[12] M. Dieye, S. Ahvar, J. Sahoo, E. Ahvar, R. Glitho, H. Elbiaze, N. Crespi, CPVNF: cost-efficient proactive VNF placement and chaining for value-added services in Content Delivery Networks, in: IEEE Transactions on Network and Service Management, Vol. 15, No. 2, pp. 774–786, 2018.

[13] F. B. Jemaa, G. Pujolle, M. Pariente, QoS-aware VNF placement optimization in edge-central carrier cloud architecture, in: IEEE Global Communications Conference (GLOBECOM), 2016.

[14] L. Qu, C. Assi, K. Shaban, Delay-aware scheduling and resource optimization with Network Function Virtualization, in: IEEE Transactions on Communications, Vol. 64, pp. 3746–3758, 2016.

[15] H. A. Alameddine, L. Qu, C. Assi, Scheduling Service Function Chains for ultra-low latency network services, in: IEEE International Conference on Network and Service Management (CNSM), 2017.

[16] F. Carpio, S. Dhahri, A. Jukan, VNF placement with replication for load balancing in NFV networks, in: IEEE International Conference on Communications (ICC), 2017.

[17] G. Sun, Y. Li, H. Yu, A. V. Vasilakos, X. Du, M. Guizani, Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks, in: Future Generation Computer Systems, Vol. 91, pp. 347–360, 2019.

[18] J. Cao, Y. Zhang, W. An, X. Chen, Y. Han, J. Sun, VNF placement in hybrid NFV environment: modeling and genetic algorithms, in: IEEE International Conference on Parallel and Distributed Systems (ICPADS), 2016.

[19] D. Cho, J. Taheri, A. Y. Zomaya, L. Wang, Virtual Network Function placement: towards minimizing network latency and lead time, in: IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 2017.

[20] G. Sun, G. Zhu, D. Liao, H. Yu, X. Du, M. Guizani, Cost-efficient service function chain orchestration for low-latency applications in NFV networks, in: IEEE Systems Journal, pp. 1-13, 2018.

[21] G. Sun, Y. Li, D. Liao, V. Chang, Service function chain orchestration across multiple domains: A full mesh aggregation approach, in: IEEE Transactions on Network and Service Management, Vol. 15, No. 3, pp. 175–1191, 2018.

[22] J. Liu, W. Lu, F. Zhou, P. Lu, Z. Zhu, On dynamic Service Function Chain deployment and readjustment, in: IEEE Transactions on Network and Service Management, Vol. 14, No. 3, pp. 543–553, 2017.

[23] F. Slim, F. Guillemin, A. Gravey, Y. Hadjadj-Aoul, Towards a dynamic adaptive placement of virtual Network Functions under ONAP, in: IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017.

[24] L. Peterson, A. Al-Shabibi, T. Anshutz, S. Baker, A. Bavier, S. Das, J. Hart, G. Palukar, W. Snow, Central office re-architected as a data center, in: IEEE Communications Magazine, Vol. 54, No. 10, pp. 96–101, 2016.

[25] M. Barbehenn, A note on the complexity of dijkstra's algorithm for graphs with weighted vertices, in: IEEE Transactions on Computers, Vol. 47, No. 2, 1998.

[26] FP7 COMBO project, Analysis of transport network architectures for structural convergence, Deliverable D3.3.

[27] http://www.cloudping.info, Cloudping.

[28] I. Cerrato, M. Annarumma, F. Risso, Supporting fine-grained Network Functions through Intel DPDK, in: IEEE European Workshop on Software Defined Networks (EWSDN), 2014.

[29] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, et al., Mobile-edge computing introductory technical white paper, in: Mobile-edge Computing (MEC) industry initiative, 2014.