Fichera, S., Martínez, R., Martini, B., Gharbaoui, M., Casellas, R., Vilalta, R., Muñoz, R. y Castoldi, P. (2019). Latency-aware resource orchestration in SDN-based packet over optical flexi-grid transport networks. *IEEE/OSA Journal of Optical Communications and Networking*, 11 (4), pp. B83-B96.

DOI: https://doi.org/10.1364/JOCN.11.000B83

Latency-aware Resource Orchestration in SDN-based Packet over Optical Flexi-Grid Transport Networks

S. Fichera[1], R. Martínez[2], B. Martini[3], M. Gharbaoui[3], R. Casellas[2], R. Vilalta[2], R. Muñoz[2], and P. Castoldi[1]

[1]Scuola Superiore Sant'Anna, Pisa (Italy)
[2]Centre Tecnolgic de Telecomunicacions de Catalunya (CTTC/CERCA), Barcelona (Spain)
[3]National Inter-University Consortium for Telecommunications (CNIT), Pisa (Italy)

***Abstract*—In the upcoming 5G networks and following the emerging Software Defined Network/Network Function Virtualization (SDN/NFV) paradigm, demanded services will be composed of a number of virtual network functions that may be spread across the whole transport infrastructure and allocated in distributed Data Centers (DCs). These services will impose stringent requirements such as bandwidth and end-to-end latency that the transport network will need to fulfill. In this paper, we present an orchestration system devised to select and allocate virtual resources in distributed DCs connected through a multi-layer (Packet over flexi-grid optical) network. Three different on-line orchestration algorithms are conceived to accommodate the incoming requests by satisfying computing, bandwidth and end-to-end latency constraints, setting up multi-layer connections. We addressed end-to-end latency requirements by considering both network (due to propagation delay) and processing delay components. The proposed algorithms have been extensively evaluated and assessed (via a number of figures of merit) through experimental tests carried out in a Packet over Optical Flexi-Grid Network available in the ADRENALINE testbed with emulated DCs connected to it.**

***Index Terms*—5G; SDN; NFV; Orchestration; Virtual Network Function; VNF Forwarding Graph.**

I. Introduction

5G is not only a New Radio technology. It will bring new unique network and service capabilities by integrating networking, computing and storage resources into one programmable and unified infrastructure, leveraging both Network Function Virtualization (NFV) and Software-Defined Networking (SDN), i.e., *softwarization* [1]. NFV promotes a scenario in which network functions (i.e., from a switch/router to a software middle-box) are deployed in virtual machines (VMs) as Virtualized Network Functions (VNFs). VNFs can be deployed centralized in the cloud or distributed in clusters of small- and medium-DCs located at the edge of the network [2]. SDN provides programming network abstractions to enable the connectivity among the deployed VNFs [3]. On the other hand, the new radio capabilities will foster significantly higher throughput and lower latency enabling a new breed of applications in several domains (e.g., connected cars, Industry 4.0) that will require different network functions (e.g., security, deep packet inspection) along with various levels of QoS (e.g., bandwidth and end-to-end latency).

Thanks to network softwarization, a scenario can be envisioned where service providers may offer not only communication services, but also virtualized computing and storage capabilities by elastically slicing the (cloud and network) infrastructure into partitions (i.e., *network slices*) with customized VNFs for specific applications [4][5]. The computing and storage resources deployed in different DCs are interconnected through (virtual) links over a physical transport (e.g., optical) infrastructure. The topological terms of such a distributed VNF deployment are specified by the VNF Forwarding Graphs (VNF-FGs) [6].

The possibility of dynamically provisioning network slices is attracting a lot of interest from network and service operators and standardization organizations [6] eager to leverage its high flexibility, rapidity and cost-effectiveness when deploying network services [7][8]. However, the above softwarization capabilities are at the expenses of imposing a burden on the DCs and on the (metro and core) network interconnecting the DCs. Moreover, the VNFs may experience additional latency caused by the delocalization of the involved VNFs as well as by the limited DC resource availability, especially at the network edge. Thus, to exploit efficiently softwarization, it is crucial to select the DCs hosting the VNFs considering both DC and network resources availability while guaranteeing demanded capacity and delay performance of 5G applications [9][10][11].

We investigate resource orchestration strategies for the dynamic allocation of virtual resources deploy-

ing VNF-FGs over distributed DCs interconnected through a Multi-Layer Network (MLN) combining packet and optical flexi-grid transport technologies. VNF-FG allocation requests (*i.e.,VNF-FG_Reqs*) are handled by a Cloud and Network Resource Orchestrator that is in charge of (i) processing *VNF-FG_Reqs*, (ii) selecting the resources based on specified requirements and, accordingly, (iii) triggering the allocations in the underlying infrastructure. In particular, the Orchestrator relies on a transport SDN Controller (T-SDN Controller) to compute and establish MLN connections for the inter-DC connectivity satisfying the requested QoS needs (i.e., bandwidth and end-to-end latency) for Virtual Links (VLs) connecting VNFs.

The topic of allocating both cloud and networking resources for end-to-end services (e.g.,VNF-FG) encompassing multiple and remote DCs over transport network infrastructures (e.g., packet over optical) is nowadays getting notable attention [12][13][14]. Most of these works focus on the deploying mechanisms to allocate cloud and network resources over multiple DCs mainly satisfying compute and bandwidth service demands. Herein, besides such requirements we also address the increasingly stringent demands of latency-sensitive network services. The proposed resource orchestration algorithms aim at not only attaining an efficient use of the cloud and network resources, minimizing the resulting end-to-end latency, considering both propagation and DC processing delays.

The cloud and network resource orchestrator architecture and the devised resource orchestration algorithms are experimentally evaluated within the CTTC ADRENALINE testbed [15]. For this, different types of DCs are considered, supporting heterogeneous capabilities in terms of available compute resources (i.e., CPU, RAM and Disk) and offered processing delay of VNF data flows. The performance evaluation is carried out assuming dynamic network service (i.e.,VNF-FGs) generation, each request imposing different bandwidth and latency requirements. A number of figures of merit (e.g., acceptance ratio, average setup delay, blocked bandwidth ratio, resulting propagation delay and end-to-end latency, etc.) are used for the comparative analysis.

The rest of this paper is organized as follows: in Section II, we overview the related works to highlight the specific contribution of the proposed orchestration system. In Section III we describe the reference scenario and the deployed cloud and network orchestration system on top of distributed DC and MLN infrastructures. Section IV presents the resource orchestration algorithms. The experimental performance evaluation is presented in Section V. Finally, Section VI summarizes this work.

II. Related Work

This section discusses the research works in three main related areas while highlighting the contribution brought by this work.

A. Resource management in multi-layer networks

For both metro and core networks, it is widely agreed that the integration of packet switching over flexi-grid optical networks with sliceable bandwidth variable transponders (SBVTs) will allow leveraging the best from both worlds. On the one hand, the well-known *electrical grooming* which leverages the statistical multiplexing provided by packet switching that enables that low-rate data flows from different sources can be steered to different destinations being grouped and transported over a set of established optical connections with sufficient spare bandwidth; on the other hand, the tailored allocation of just enough optical spectrum according to the traffic demands exploiting the flexibility of SBVTs (e.g., adapting configuration parameter such modulation format, symbol rate, etc.) [16][17][15]. In addition to the electrical grooming, flexi-grid optical networks allow exploit the so-called *optical grooming* which does offload the electronic processing burden towards the optical layer [18]. By doing so, the goal is to apply strategies fostering both electrical and optical grooming decisions to achieve the most efficient use of all the involved MLN resources: packet ports, SBVTs and optical spectrum.

Upon receiving a new packet connection request, constrained shortest path computation (CSPF) algorithms are triggered. CSPF algorithms in the considered MLN rely on modified routing and spectrum assignment (RSA) algorithms. Such algorithms not only deal with the optical resource computation and their technology constraints (e.g., spectrum continuity and contiguity) but also consider the packet network topology. Established flexi-grid optical connections derive on virtual (packet) links which inherit attributes (e.g., available bandwidth, accumulated delay, etc.) from their underlying (optical) connections. Therefore, the output of the algorithms is composed of a set of virtual packet links (reusing spare available bandwidth, i.e. electrical grooming) combined with new flexi-grid optical path segments enabled by SBVT capabilities. Examples of these CSPFs algorithms addressing a number of objective functions such as energy-efficiency, network cost reduction, impairment-awareness, etc. can be found in [15], [17], [18], [19].

B. Resource allocation in multi-DC infrastructures

The problem of allocating resources to deploy VNFs in NFV environments is equivalent to the problem of assigning computational resources to application tasks in distributed DCs [20]. This problem is referred to as a *placement* problem. To achieve more efficient resource allocations across different DCs, the availability of network resources (e.g., link bandwidth) should be also considered while assigning these resources to transport traffic flows among VNFs [21]. This facet

is considered in [22] where an ILP formulation is proposed to optimize the selection of DCs while taking into account the available resources both in terms of CPU cycles and network bandwidth. In [23], the optimal placement and selection of computing and network resources for supporting cloud tasks is addressed statically, while accounting for protection against failures.

The problem of embedding virtual networks in a substrate network is another related research area and represents the main resource allocation challenge in network virtualization [24]. This is usually referred to as the Virtual Network Embedding (VNE) problem. A wide range of algorithms have been proposed in the literature to optimally map virtual networks into substrate infrastructures while assigning resources to provide customized end-to-end guaranteed services to end users [25]. This optimality can be accomplished from different perspectives: ranging from QoS [26][27], economical targets [28] to survivability over energy-efficiency [29][30][31][32][33]. I n t he s pecific context of flexi-grid o ptical n etworks, t he a uthors i n [34] addressed the problem of deploying virtual networks (i.e., tenants) on top of an elastic optical network to be used for inter-connecting a myriad of remote DCs. Moreover in [35], the authors demonstrated a solution for orchestrating both cloud and network resources for inter-connecting multiple DCs over a packet and optical transport infrastructure.

All these approaches focus on optimizing the resource consumption and do not consider the accumulated end-to-end latency which is becoming an important requirement when deploying network services [11]. Moreover, they do not consider heterogeneous DC sizes depending on their network location (e.g., smaller DC at the network edge) which notably leads to experience different computing capacity and, thus, processing delays performance. In this context, in [36], the network and processing latency requirements are considered while allocating VNFs at the edge of a fixed mobile convergent network. The paper highlights that the mobile network functions present strict latency requirements that can be satisfied b y p lacing them in the metro/access network. However, the placement optimization is carried out in a static way from the service provider's perspective, i.e., in terms of resulting consolidation of VNFs, without assessing the service request admittance and latency.

C. Resource orchestration in virtualized environments

A number of works have been proposed in the literature on the orchestration of virtual resources in both cloud and network domains while embedding virtual topologies for multiple VNFs (i.e., VNF-FGs).

In [37] authors formulated an optimization problem for placing VNF-FGs within a NFV-enabled operators infrastructure, while addressing both tenant requirements related to the maximum tolerable end-to-end latency, and operator needs with respect to the maximum number of VNF instances to be deployed. Another related work is [38], which faces the problem of placing a set of VNFs on a network of physical nodes to serve a set of service chain requests while minimizing the number of used servers. However, these works do not address the problem of assigning paths to the incoming traffic flow requests to connect nodes running virtual functions, distinguishing different types of DCs. In [39] authors provide an ILP formulation and corresponding heuristics for the VNF Orchestration Problem (VNF-OP) to provision VNFs while respecting both the capacity and delay constraints and the placement order of the VNFs. However, with respect to our work, they mainly focus on the overall cost minimization of VNFs placement and do not consider network constraints such as the available links bandwidth nor the end-to-end latency.

[40] presents the UNIFY project framework which proposes a functional architecture that supports automated and dynamic service (VNF-FG) creation leveraging NFV, SDN, and cloud virtualization techniques through unifying cloud and carrier network resources in a common orchestration framework. The output of this project is ESCAPE [41], an orchestrator prototype enabling coordinated deployment of distributed DC resources (i.e., VMs) inter-connected by an SDN-controlled network. It was enhanced with new features within the 5GEx framework [42] to coordinate resources and/or service orchestration at the multi-technology and/or multi-operator level. However, in both cases, the orchestration framework does not consider peculiarities of MLNs such as packet over optical transport network.

A number of orchestration platforms have been developed to coordinately allocate virtual resources in cloud DCs: OpenSource MANO (OSM) [43], Cloudify [44], Open Network Automation Platform (ONAP) [45]. Such orchestration platforms manage descriptors where the network services are specified in terms of VNF components, instantiation parameters and forwarding graph, and where to allocate the set of resources in the underlying infrastructure is statically specified, i.e., decided in advance by the OSS. In this work we address an orchestration process the goes beyond the above state of the art orchestrators by autonomously making some deployment decisions, i.e., based on specified user requirements and dynamical context information (e.g., current availability of resource capacity, resource topology). Indeed, the proposed functionality performs the selection of underlying resources based on load status information and on offered latency performance from multiple cloud DCs. Moreover, the proposed orchestration process considers a multi-DC environment thereby handling also the dynamic selection of inter-DC links across WAN and their automatic set-up (this feature has been just very recently added in the OSM v5). Indeed, the work described in this paper contributed to the software

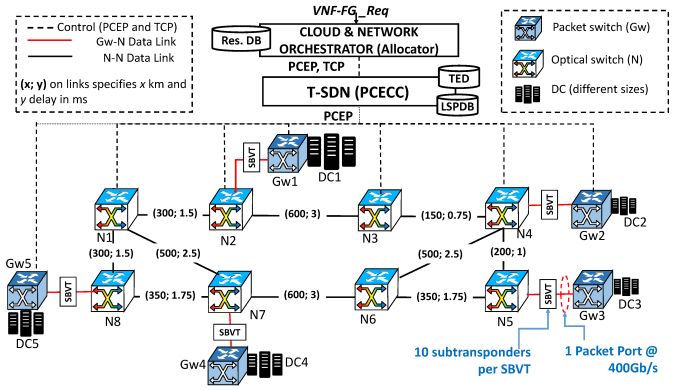


Fig. 1: Network Scenario and Deployment Set-up

development of extensions to the OSM and Cloudify orchestrators to dynamically provide the resource placement decisions and feed the above orchestrators with the descriptors complete of all information to perform their automated operations to deploy the network services [46][47].

III. Reference Scenario and Cloud/Network Orchestration system set-up

Fig. 1 shows the reference scenario and the component building blocks enabling the orchestration process. It has been set-up to be compliant with the ETSI NFV MANagement and Orchestration (MANO) framework [6] also considering the case of network setvice set-up in multi-site environments [48]. More specifically, it is composed of a transport MLN interconnecting several DCs. Each DC hosts computing and storage systems where it is assumed to deploy software virtualization technologies (i.e., Hypervisors) to offer partitions of computing and storage capabilities (e.g., VMs in the servers) to different slices. Each DC has transport network connectivity towards other DCs via a packet MPLS node (Gateway, Gw) connected to a flexi-grid optical network via SBVT elements equipped with a set of subtransponders. Thus, the MLN combines packet and flexi-grid optical technologies to establish inter-DC connectivity aiming at leveraging both packet and optical grooming opportunities while computing multi-layer network paths, i.e., virtual links connecting pairs of Gws.

The proposed Cloud and Network Orchestrator (also referred to as Allocator) dynamically processes *VNF-FG_Req* by allocating (i.e., placing) a set of virtual resources in the underlying infrastructure (i.e., VMs in DCs and virtual links in MLN) while addressing specified requirements (i.e., computing and storage capabilities, and bandwidth demands, maximum latency). This component has been presented for the first time in [49]. The Allocator interacts with the T-SDN Controller to request path computation, setting up and configuring the network resources within the MLN. The Allocator uses a Resource DB (Res. DB in Fig. 1) to store the abstract view of both network and DC resources. As for the network, the Res. DB contains the available capacity and the resulting delay of the virtual links derived from previously established flexi-grid optical flows. As for the DCs, the resource view includes aggregated view of the available CPU, RAM and Disk for each DC. With respect to [49], the Allocator has been extended with placement algorithms to select the destination DC according different policies (i.e., *minimum distance, minimum latency*). In fact, using the Res. DB, the Allocator runs a placement algorithm to decide the proper set of DCs able to host the required VNFs as well as the VLs to use to connect those DCs in order to satisfy an incoming request. We devised a set of placement (i.e., orchestration) algorithms to this purpose (described in the next sections). Finally, the output of the placement algorithm is implemented by reserving the selected DC resources (i.e., VMs running the VNFs) and requesting to the T-SDN Controller the setting up of the connectivity between the selected DCs.

In particular, once the Allocator has selected the DCs to host the VNFs, it checks whether a packet path using the existing VLs is feasible for a VNF interconnection. If a path is feasible, then it is instantiated; otherwise (i.e., in case of lack of connectivity in the packet VLs), the Allocator must rely on the T-SDN Controller for computing and providing a new MLN connection, i.e., the flexi-grid optical flows to support a new VL needed to address the request.

The T-SDN Controller (based on a PCE - Path Computation Element Central Controller implementation) supports the following functions: i) processing packet connection requests arriving from Allocator (via a NBI API that uses the PCEP - PCE protocol); ii) performing MLN path computations using an updated view provided by the TED (Traffic Engineering Database) repository; iii) record of the active packet and optical connections within the Label Switched Path Database (LSPDB), and iv) configuring the computed packet and optical network elements (i.e., switches and SBVTs). Such a configuration (e.g., indicating the input/output ports and optical spectrum to be allocated in the optical switches, SBVT selected parameters, selected MPLS label, etc.) is performed by the T-SDN controller using a SouthBound Interface (SBI). The SBI is based on PCEP and enables point-to-point control communication between the T-SDN and every node agent governing each particular packet and optical node.

The path computation at the T-SDN Controller relies on a modified Yen algorithm providing K Constrained Shortest Path First (K-CSPF) calculations. The objective function is to exploit as much as possible the opportunities to attain both electrical and optical grooming decisions along with ensuring the demands of the incoming requests (i.e., bandwidth and maximum latency). Specific details of the MLN routing algorithm used as baseline for the K-CSPF algorithm

are described in [15]. In brief, the K-CSPF algorithm sorts the computed kth MLN paths with respect to their total cost. Such costs are associated to the use of allocating resources in both the virtual packet and optical links, ports and SBVTs. Herein, the cost of allocating available bandwidth in a virtual packet link is related to the number of underlying optical hops which induced such a VL. Reusing VLs with spare available bandwidth allows leveraging the benefits o f electrical grooming. On the other hand, the cost associated to occupy a frequency slot in a physical optical link is set to 1. To exploit optical grooming opportunities, when a required optical path segment is being computed, it is considered that SBVTs with subtranponders being used (by existing optical flows) a re f avoured (i.e., assigning a lower cost) with respect to SBVTs without any occupied subtransponder. By doing so, it is fostered that the new optical path segment could be optically groomed with other existing optical flows in the same SBVT (but using different subtransponders). This, as mentioned above, provides notable benefits with respect to the optical spectrum utilization as well as energy savings. This cost model leads to jointly attain both electrical and optical grooming decisions on every computed kth MLN path. Therefore every kth MLN path will have a total computed path cost. Paths having the same cost are sorted by the lowest latency. The first r esulting M LN p ath s atisfying t he latency restriction of the request being served is then chosen.

IV. Latency-aware Orchestration Algorithms

This section focuses on describing the latency-aware algorithms being executed by the Allocator for the placement of VNF-FGs over the MLN. In the following subsections besides presenting the formal problem to be addressed when dynamically processing incoming *VNF-FG_Req*s, a description (in pseudocode) of each algorithm executed at the Allocator is provided. The resulting allocation computed by a particular algorithm is represented with the corresponding workflow. This allows detailing the control and orchestration interactions between the Allocator and the T-SDN Controller.

A. System Model and Problem Formulation

The cloud/network infrastructure is modeled as a direct graph *G(V, E)*, where *V* is the set of DCs connected via a set of (virtual) packet links *E* derived over the physical flexi-grid optical connections.

Each DC ($v \in V$) is described as a tuple {*Id, Gw, availCPU, availRAM, availDISK, p*} where *Id* is the DC identifier, *Gw* determines the IP address of the MPLS switch attached to the DC providing inter-DC packet connectivity, *availCPU*, *availRAM* and *availDISK* define the available amount of DCs CPU, RAM and DISK, respectively. Finally, *p* describes the average processing time (in ms) incurred by a DC *v* according to its size (i.e., amount of cloud resources)[50].

Virtual (unidirectional) packet links ($e \in E$), derived from existing underlying flexi-grid optical flows, interconnect pairs of remote DCs. Each link is described as a tuple $\{u,\ v,\ ilid_{u,v},\ elid_{u,v},\ availBw,\ c_{u,v},\ d_{u,v},\ delay_{u,v}\}$ where *u, v* $\in V$ are the respective interconnected DCs (i.e., their Gws), $ilid_{u,v}$ and $elid_{u,v}$ are the ingress and egress packet link identifiers used to unambiguously determine a link *e* since multiple virtual packet links can be created for a given *u, v* DC pair; *availBw* reflects the unused bandwidth in b/s; $c_{u,v}$ determines the cost of using such a link; $d_{u,v}$ is the distance expressed in km whilst $delay_{u,v}$ determines the overall (propagation) delay of the packet link.

The *VNF-FG_Req* conveys a graph with vertices (i.e., VNFs) featured by the amount of required computing resources (i.e., number of VMs, CPU, RAM and Disk per VM), and with edges (i.e., VLs connecting VNFs) featured by the packet inter-DC bandwidth demand (Bw, b/s). Latency requirements are also specified (i.e., l, ms).

In this work we aim at mainly investigating the implications of doing resource orchestration in the SDN-controlled MLN considering twofold latency performance components (i.e., propagation and DC processing delays) while making the placement decision. Without lack of generality, we consider that every request has two vertices (i.e., VNFs at two remote DCs). The *srcDC* is imposed under the assumption of location constraints being specified in advance, e.g., the access connection of the users to be served. The generalization to the case of arbitrary *VNF-FG_Req* is left as future work. Each incoming *VNF-FG_Req* is determined by eq. (1):

$$\textit{VNF-FG\_Req} = ([\textit{srcDC}, srcVM], \#VM@dstDC], cpu, ram, disk, bw, l) \quad (1)$$

where,

- *srcDC* identifies the source DC.
- *srcVM* specifies the required number of VMs required at the *srcDC*.
- *#VM@dstDC* specifies the required number of VMs at the *dstDC*.
- *cpu* specifies the amount of CPU resource to be allocated to each VM (to be placed either in *srcDC* or *dstDC*).
- *ram* specifies the amount of RAM resource to be allocated to each VM (to be placed either in *srcDC* or *dstDC*).
- *disk* specifies the amount of DISK resource to be allocated to each VM (to be placed either in *srcDC* or *dstDC*).
- *bw* specifies the required packet inter-DC connectivity bandwidth (in b/s).
- l specifies the maximum tolerated latency (in ms) for the inter-DC link connecting *srcDC* and *dstDC*.

The output of the resource orchestration algorithm includes the resources to be instantiated at both the

srcDC and *dstDC*, along with the request for computing and/or allocating virtual packet link resources to the T-SDN Controller for the inter-DC packet connectivity. Regardless of the algorithm, if the Allocator is unable to compute a feasible packet connectivity between the selected DCs (e.g., due to disconnected graph, insufficient available bandwidth on the existing VLs, unsatisfied l atency r equirement, e tc.), t he path and network resource computation is delegated the T-SDN Controller. If the required cloud resources in the DCs or the networking requirements cannot be met, the *VNF-FG_Req* is blocked. Observe that the actual allocation of cloud and networking resources is carried out into two separated and sequential steps: first, t he A llocator a lways s elects a nd a llocates the cloud resources at the DCs. Depending on the adopted orchestration algorithm (described in the following sections) the Allocator's output may also contain a packet path describing the inter-DC connectivity to be deployed. On the other hand, the T-SDN controller performs the computation (if it is not provided by the Allocator) as well as conducts the allocation of the selected networking resources (done by either the Allocator or the T-SDN controller by itself). In both cases, if the T-SDN controller cannot allocate the computed path, the *VNF-FG_Req* is blocked. This entails that the previously allocation of the cloud resources made by the Allocator needs to be conveniently released.

B. Latency-aware resource orchestration algorithms

Three different algorithms are proposed and compared differing not only on the targeted optimization objective (e.g., minimizing the end-to-end latency, attaining a more efficient use of both cloud and networking resources, etc.) but also on the (abstracted) network-related information passed from the network controller towards the Allocator.

1) No Network Information (NNI) Algorithm: In the NNI algorithm, the Allocator only handles information about the DCs resources (i.e., *G(V, null)*), and no network information, i.e., set of established virtual packet links, is passed from the T-SDN Controller. Thus, the Allocator only performs the DC selection while the selection and configuration of the VLs is delegated to the T-SDN Controller.

As shown in Algorithm 1, in the NNI algorithm it is checked whether the total requested cloud resources at both *srcDC* and *dstDC* (referred to as *src/dstTCPU*, *src/dstTRAM* and *src/dstTDISK*) can be actually allocated (lines 12-23). If this works, the requested cloud resources are assumed to be immediately allocated at the DCs modifying its status on the Resource DB (lines 25-26). The output of the NNI algorithm is a request to the T-SDN Controller for deriving a new VL enabling the inter-DC packet connectivity (lines 30-31). Otherwise, if the cloud resources allocation at DCs fails, resources are then released in the Resource DB, and the *VNF-FG_Req* is blocked.

Algorithm 1 NNI Algorithm

1: **Input**: $G(V, NULL)$, *VNF-FG_Req*
2: **Output**: PCInitiate$(Gw_{src}, Gw_{dst}, bw, l)$ OR BlockVNF-FG_Req
3: **Initialization**:
4: $found\_srcDC = False$
5: $found\_dstDC = False$
6: $srcTCPU = srcvm * CPU$ ▷ Total CPU at srcDC
7: $srcTRAM = srcvm * RAM$ ▷ Total RAM at srcDC
8: $srcTDISK = srcvm * DISK$ ▷ Total DISK at srcDC
9: $dstTCPU = dstvm * CPU$ ▷ Total CPU at dstDC
10: $dstTRAM = dstvm * RAM$ ▷ Total RAM at dstDC
11: $dstTDISK = dstvm * DISK$ ▷ Total DISK at dstDC
12: **function** CHECKSRCAVAILABILITY($V, srcDC, srcTCPU, srcTRAM, srcTDisk$)
13: **for** v **in** V **do**
14: **if** $srcDC == v$ **then**
15: **if** $(availCPU_v \geq srcTCPU)$ AND $(availRAM_v \geq srcTRAM)$ AND $(availDISK_v \geq srcTDISK)$ **then**
16: $found\_srcDC = True$
17: **break**
18: **else**
19: **return** BlockVNF-FG_Req
20: **end if**
21: **end if**
22: **end for**
23: **end function**
24: $dstDC = random(v$ **in** $(V - srcDC))$ ▷ Select ramdomly a dstDC
25: **if** $(availCPU_v \geq dstTCPU)$ AND $(availRAM_v \geq dstTRAM)$ AND $(availDISK_v \geq dstTDISK)$ **then**
26: $found\_dstDC = True$
27: **else**
28: **return** BlockVNF-FG_Req
29: **end if**
30: **if** $(found\_srcDC)$ AND $(found_dstDC)$ **then**
31: **return** PCInitiate$(Gw_{src}, Gw_{dst}, bw, l)$
32: **else**
33: **return** BlockVNF-FG_Req
34: **end if**

Inter-DC connectivity requests are handled by sending a PCEP PCInitiate message from the Allocator to the T-SDN Controller (see Fig. 2). This message includes the IP addresses of the DCs endpoints (i.e., Gw_{src}, Gw_{dst}) along with the bw and l requirements. PCInitiate message is processed by the frontend T-SDN module (referred to as PCECC - PCE Central Controller). PCECC asks the Path Computation function to compute a MLN path (via PCEP PCReq). Observe that this message carries the same requirements parameters as the received PCInitiate. The executed algorithm at the Path Computation function aims at computing, selecting and allocating MLN resources (i.e., packet port and virtual packet link's bandwidth, SBVT subtransponders and optical spectrum) dealing with the *VNF-FG_Req* requirements. The MLN path computation is based on the previous described K-CSPF algorithm. It is worth mentioning that whenever the latency-aware orchestration mechanisms executed at the Allocator are unable to find the targeted inter-DC packet connectivity, the T-SDN Controller is requested to trigger the K-CSPF algorithm as

detailed in section IV-B2 and section IV-B3.

The output of the MLN path computation (K-CSPF algorithm) is returned to the PCECC via a PCEP PCRep message encoding a set of N ($N >= 1$) Explicit Route Objects (EROs). The ERO describes the nodes, links and resources (including SBVT' subtransponders and their configuration c apabilities a long w ith the selected optical spectrum) to be allocated [15]. As a MLN path, it combines virtual packet links (VLs with spare available bandwidth) with new flexi-grid optical path segments to be established. The latter are typically set up when for instance no entire virtual packet connectivity from the targeted endpoints (i.e., DC's Gws) is feasible. It is worth noting that setting up an optical path segment will eventually induce creating a VL between the optical path segment endpoints (e.g., DCs Gws). As said, the VL inherits attributes derived from its underlying flexi-grid optical connection. These attributes are the metric (based on the number of traversed optical switches), the available bandwidth derived from the SBVT configuration and the accumulated delay resulting from the traversed optical fiber links. Note that the available bandwidth of a new induced VL may be larger than the requested bandwidth (i.e., *bw*) in the *VNF-FG_Req*. This in turn will foster upcoming computation exploiting the electrical grooming [15].

The computed ERO related to the optical path segment/s is passed to the Provisioning Manager function of the T-SDN Controller to configure fi rst th e optical resources as well as triggering the operations to create and notify a new VL. This is done, as reflected in Fig. 2, via a PCEP PCInitiate message containing the so-called *Server Layer ERO* which lists the optical nodes and links to be traversed as well as the selected optical spectrum (i.e., frequency slot) and SBVT capabilities to be configured. T he P rovisioning M anager v ia the PCEP-based SBI [15] communicates with each optical node agent to perform the required configurations. For the sake of completeness, this is done into two sequentially steps: i) a PCEP PCInitiate message is sent to the agent governing the ingress optical node (i.e., $N_{ingress}$) of the optical path segment. Such a node agent responds to the T-SDN Provisioning Manager with an PCEP PCRpt containing a PCEP-specific LSP identifier (PLSP-ID) t hat i dentities t he o ptical path (see Fig. 2); ii) using the PLSP-ID, the provisioning manager sends PCEP PCLabelUpd messages to every involved optical network nodes' agents to complete the configuration. T his P CLabelUpd m essage carries the input and output ports of every optical switch along with the selected frequency slot. Additionally, for both the ingress and egress nodes (i.e., $N_{ingress}$ and N_{egress}) the PCEP PCLabeUpd message also provides the SBVT parameters to be programmed.

Once the optical path segment is set up, a new VL is induced inheriting the attributes (i.e., metric, available bandwidth and delay) related to the underlying established optical path segment. This allows the Provisioning Manager function allocating packet network resources (i.e., bandwidth) over such new VLs as well as previously existing VLs (with sufficient available bandwidth). Likewise the optical path segment establishment, the allocation of the packet resources between packet nodes (i.e., Gw_{src} and Gw_{dst}) is performed through the PCEP-SBI exchanging the PCEP PCInitiate with the Gw_{src} agent and, afterwards PCEP PCLabelUpd messages with both Gw_{src} and Gw_{dst} agents. FOr the packet connection, the PCLabelUpd messages carry the input and output packet ports, the bandwidth (i.e., bw) and the MPLS label to to be allocated and used in the packet connection being established.

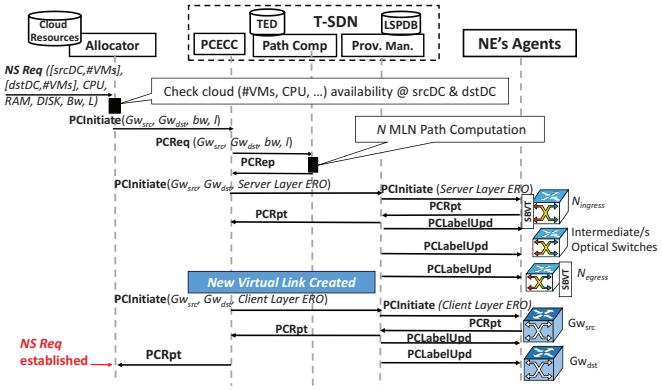


Fig. 2: No Network Information (NNI) Workflow

The successful establishment of the requested inter-DC packet connection is notified to the PCECC (using the PCRpt), which in turn relies on the same PCRpt to inform the Allocator. At this point, the *VNF-FG_Req* is considered as served. If errors occur when allocating network resources, PCEP PCError messages are returned to the PCECC. This will cause the Allocator releasing occupied cloud resources and blocking the *VNF-FG_Req*.

2) Minimum Distance (MD) Algorithm: In the MD algorithm, the Allocator handles both information about the DCs resources and abstracted network resource information on the set of existing VLs passed from the T-SDN Controller. The objective of the MD strategy is to minimize the distance between *srcDC* and *dstDC*. By doing so, the end-to-end propagation delay is minimized. Thus, the MD algorithm seeks for a *dstDC* attaining the shortest distance (in km) from the *srcDC* fulfilling the *VNF-FG_Req* requirements from both cloud and networking perspectives.

As in the NNI algorithm and as shown in Algorithm 2, upon receiving a *VNF-FG_Req*, it is checked that at the *srcDC* there are sufficient available cloud resources to accommodate the demand (line 4). Otherwise, the *VNF-FG_Req* is blocked. Next, the MD algorithm seeks for candidate DCs (different than the *srcDC*) with sufficient available cloud resources satis-

Algorithm 2 MD Algorithm

1: **Input**: $G(V, E)$, *VNF-FG_Req*
2: **Output**: PCInitiate($Gw_{src}, Gw_{dst}, bw, l$) OR (PCInitiate($Gw_{src}, Gw_{dst}, ERO$)) OR BlockVNF-FG_Req
3: **Initialization**: ▷ likewise in NNI
4: **Function CHECKSRCAVAILABILITY**: ▷ likewise in NNI
5: ▷ Find a candidate $dstDC$ with minimum distance
6: $minDist \leftarrow \infty$
7: $dstDC = \emptyset$
8: **for** v **in** V **do**
9: $Dist_v = \infty$
10: **if** $srcDC == v$ **then**
11: continue
12: **end if**
13: **if** $(availCPU_v < dstTCPU)$ OR $(availRAM_v < dstTRAM)$ OR $(availDISK_v < dstTDISK)$ **then**
14: continue
15: **end if**
16: $Dist_v \leftarrow$ PCReq(Gw_{src}, Gw_v, bw)
17: **if** $Dist_v < minDist$ **then**
18: $minDist = Dist_v$, $dstDC = v$
19: **end if**
20: **end for**
21: **if** $dstDC == \emptyset$ **then**
22: **return** BlockVNF-FG_Req
23: **end if**
24: Collect *Client Layer Network*
25: ERO $= \emptyset$ ▷ at packet layer
26: Compute ERO using $Gw_{src}, Gw_{dst}, bw, l$
27: **if** ERO $== \emptyset$ **then** ▷ no feasible path
28: **return** PCInitiate($Gw_{src}, Gw_{dst}, bw, l$)
29: **else**
30: **return** PCInitiate(Gw_{src}, Gw_{dst}, ERO)
31: **end if**

fying the *VNF-FG_Req* at the *dstDC* (lines 8-15). For each candidate *dstDC*, the Allocator requests to the T-SDN Controller computing a feasible path from the *srcDC* to the under-considered candidate *dstDC* (fulfilling the *VNF-FG_Req*'s bw) (lines 16-19). If a feasible path is found, the total distance (in km) for that path is returned. The Allocator selects the candidate *dstDC* with lowest distance. If no feasible path is found for all the candidate *dstDC*s, the connection is blocked, releasing any previous cloud resource allocation.

For the selected *dstDC*, the Allocator requests to the T-SDN Controller an abstracted view of the existing VLs (lines 24-26). This forms a virtual packet network, which is used by the Allocator to trigger a shortest path computation between the *srcDC* and *dstDC* (i.e., Gw_{src} and Gw_{dst}) with the requirements of bw and l. If a feasible path is computed (i.e., a complete ERO at the packet layer), then the Allocator sends to the T-SDN Controller a PCInitiate message carrying the computed ERO to configure the selected networking resources (line 30). Observe that this allows reusing spare available capacity of VLs, favoring electrical grooming strategies. The computed ERO is passed towards the T-SDN controller. The T-SDN controller triggers its Provisioning Manager function to actually perform the packet connection establishment based on the received ERO (i.e., *Client Layer ERO*). Via the PCEP-based SBI (i.e., PCEP PCInitiate and PCLabelUpd messages) the Provisioning Manager coordinates the allocation of the required bw and selected MPLS label over the VLs specified in the ERO as shown in Fig. 3.

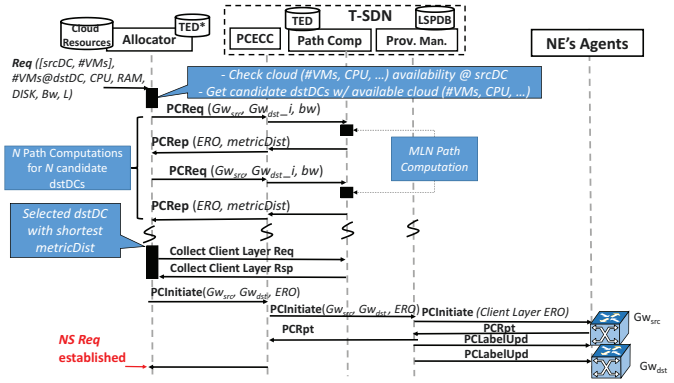


Fig. 3: Minimum Distance (MD) Workflow

If the Allocator cannot find a feasible path connecting both *srcDC* and *dstDC* due to disconnected packet network graph or existing VLs do not have sufficient available bandwidth, the packet path computation and allocation, as in the NNI approach, is entirely delegated to the T-SDN Controller (lines 27-28). In other words, since the T-SDN controller does not receive from the Allocator a computed packet ERO, the MLN path computation (K-CSPF algorithm) needs to executed. The output of this MLN path computation may combine new optical path segments to be set up (inducing new VLs) as well as reusing existing VLs. The procedures and interactions for setting up new optical path segments and allocating bw on the derived VLs follow the same description as represented in Fig. 2.

3) Minimum Latency (ML) Algorithm: As in the MD algorithm, in the ML algorithm, the Allocator handles both information about the DCs resources and network resources encompassing the set of established VLs passed from the T-SDN Controller. In the ML strategy, it is considered that at the time of dealing with the *VNF-FG_Req*s maximum end-to-end latency (i.e., l) in addition to the propagation delay of the inter-DC connectivity, the processing time incurred at the selected DCs needs to be also taken into account. Thus, ML algorithm enhances MD strategy seeking for a *dstDC* that attains the lowest contribution of both the distance to the *srcDC* (i.e., propagation delay) and the DCs processing delay. Recall that we consider heterogeneous DC sizes [50] (i.e., amount of cloud equipment) which does entail different processing times at each DC.

As shown in algorithm 3, the key difference with respect to the MD algorithm is that at the time of considering a candidate *dstDC*, from the maximum demanded latency (i.e., l) by the *VNF-FG_Req*, it is extracted the processing time consumed by both the *srcDC* and the under-consideration *dstDC*. The

Algorithm 3 ML Algorithm

1: **Input**: $G(V, E)$, *VNF-FG_Req*
2: **Output**: PCInitiate(Gw_{src}, Gw_{dst}, bw) OR
3: PCInitiate(Gw_{src}, Gw_{dst}, ERO) OR BlockVNF-FG_Req
4: **Initialization**: ▷ likewise in NNI
5: **Function CHECKSRCAVAILABILITY**: ▷ likewise in NNI
6: ▷ Find a candidate $dstDC$ with minimum end-to-end latency
7: $minLatency \leftarrow \infty$
8: $dstDC = \emptyset$
9: **for** v **in** V **do**
10: $Latency_v = \infty$
11: $maxPropDelay_v = l - p_{srcDC} - p_v$
12: $PropDelay_v = 0$
13: **if** $srcDC == v$ **then**
14: continue
15: **end if**
16: **if** ($availCPU_v < dstTCPU$) OR ($availRAM_v < dstTRAM$) OR ($availDISK_v < dstTDISK$) **then**
17: continue
18: **end if**
19: $PropDelay_v \leftarrow$ PCReq($Gw_{src}, Gw_v, bw, maxPropDelay_v$)
20: $Latency_v = PropDelay_v + p_{srcDC} + p_v$
21: **if** $Latency_v < minLatency$ **then**
22: $minLatency = Latency_v$, $dstDC = v$
23: **end if**
24: **end for**
25: **if** $dstDC == \emptyset$ **then**
26: **return** BlockVNF-FG_Req
27: **end if**
28: Collect *Client Layer Network*
29: ERO $= \emptyset$ ▷ at packet layer
30: Compute ERO using $Gw_{src}, Gw_{dst}, bw, l$
31: **if** ERO $== \emptyset$ **then** ▷ no feasible path
32: **return** PCInitiate($Gw_{src}, Gw_{dst}, bw, l$)
33: **else**
34: **return** PCInitiate(Gw_{src}, Gw_{dst}, ERO)
35: **end if**

resulting is the maximum propagation delay budget available to accommodate the inter-DC connectivity between the *srcDC* and the candidate *dstDC*. This budget (expressed as *maxPropDelay* in Algorithm 3) is then used to constrain a request sent to the T-SDN Controller to find a feasible route between the Gw_{src} and the candidate Gw_{dst}. If the T-SDN Controller succeeds in the path computation, the output is made up of a computed ERO along with the actual propagation delay (i.e., *PropDelay* in algorithm 3). Then, for the considered candidate set of *dstDCs*, the end-to-end latency through the selected path is computed adding to the *PropDelay* (line 19), the processing delays related to both *srcDC* and *dstDC* (i.e., p_{srcDC} and p_{dstDC}) (line 20). Finally, the ML algorithm selects the *dstDC* attaining the lowest end-to-end computed latency (lines 21-22). Next, ML algorithm triggers a shortest path computation between the *srcDC* and *dstDC* (i.e., Gw_{src} and Gw_{dst}) with the requirements of bw and l. If a feasible path is computed, then the Allocator sends to the T-SDN Controller a PCInitiate message with the ERO to configure the selected networking resources (line 34). Otherwise, the Allocator requests to the T-SDN Controller the path computation and allocation (line 32) of the packet connection which may entail setting up new optical path segments and the consequent creation of new VLs required to accommodating

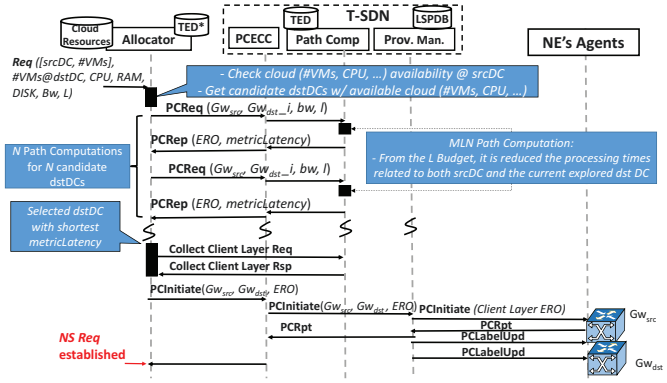


Fig. 4: Minimum Latency (ML) Workflow

the targeted connectivity between the selected DCs' endpoint Gw_{src} and Gw_{dst}. The workflow reflecting the ML strategy is represented in Fig. 4.

V. Experimental Performance Evaluation

The experimental performance evaluation is performed using the CTTC ADRENALINE testbed encompassing a cloud/network infrastructure as shown in Fig. 1. This is formed by 5 (emulated) DCs attached to 5 packet Gw nodes (i.e., MPLS switches). Each Gw has a single port (operating at 400 Gb/s) connected to the optical flexi-grid transport network via an SBVT supporting 10 subtransponders. Each subtransponder can use 3 different modulation formats - MFs- (namely, DP-16QAM, DP-8QAM and DP-QPSK) enabling 3 respective bit rates (i.e., 200, 150 and 100 Gb/s, respectively) for 3 different maximum distances (650, 1000 and 3000 km). Optical flexi-grid physical links support 128 Nominal Central Frequencies spaced 6.25GHz. The fiber link distances as well as its associated delay are explicitly indicated on each link (see Fig. 1) using the notation *(x; y)* where *x* determines the link distance in km and *y* specifies the delay in ms. For the sake of completeness, we have assumed that the link delay is obtained applying the propagation delay of 5 μs/km. Such link distance and delay are used for not only selecting a feasible MF for an optical flow during the K-CSPF execution, but also computing the accumulated propagation delay when dealing with *VNF-FG_Req*'s latency restriction.

Every experimental point is realized with 1000 requests following a Poisson process *VNF-FG_Req* whose mean inter-arrival time is set to 25s and the duration (holding time, HT) is exponentially modelled varying its mean to 100, 150, 200, 250, 300 and 350 s. This provides different offered traffic loads (expressed in Er): 4, 6, 8, 10, 12 and 14. The requirements of each *VNF-FG_Req* are generated as follows: the number of VMs per DC is uniformly distributed between [1,5]; the IT resources (i.e., CPU, RAM and Disk) are randomly chosen in the ranges of [1,4] cores, [1,6] GB and [4,10,20,40] GB, respectively; the demanded

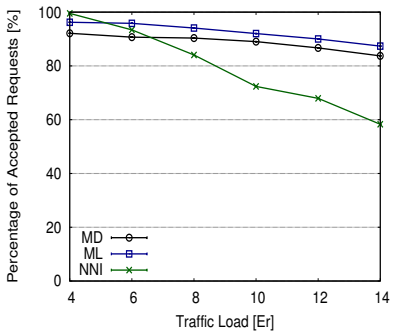


Fig. 5: Percentage of Accepted Requests (%) vs. Traffic Load (Er).

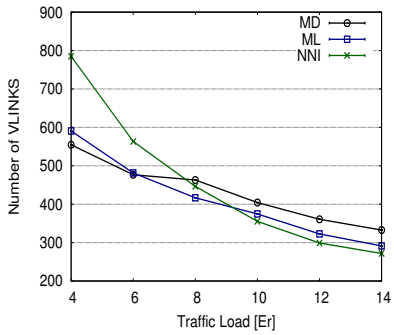


Fig. 6: Number of induced virtual links vs. Traffic Load (Er).

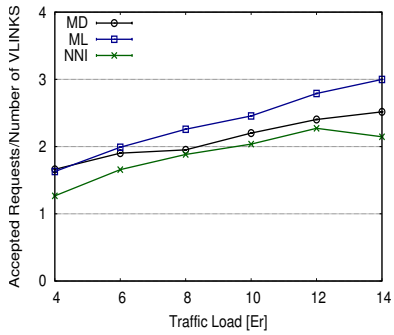


Fig. 7: Accepted requests/virtual links ratio vs. Traffic Load (Er).

bandwidth (bw) is randomly selected among [10, 40, 100] Gb/s, and the latency (l) is in the range of [8,15] ms.

The 5 DCs have different sizes and cloud resource characteristics. We assign 40 CPU Cores, 160GB of RAM and 7TB of Disk (i.e., Storage) for the so-called *small* DCs which are connected to Gw2 and Gw3 in Fig. 1; 80 CPU cores, 320GB of RAM and 10TB of Storage for the *medium* DCs attached to Gw4 and Gw5; and 500 CPU cores, 2400GB of RAM and 135TB of Storage for the $large$ DCs connected to Gw1. Additionally, taking into account the DC size, we assigned a different processing capacity to each DC as follows: 7.5 GBps for small DCs, 15 GBps for medium DCs and 30 GBps for the largest one [50]. Then, based on our previous works where different data traffic amounts were considered in order to properly resize the multi-DC infrastructure [11][51], we fixed the data request to 250 Mb. As a result, using the eq. (2), the average processing time for each DC type equals: 4.16 ms for small DCs, 2.08 ms for medium DCs and 1.04 ms for the largest one.

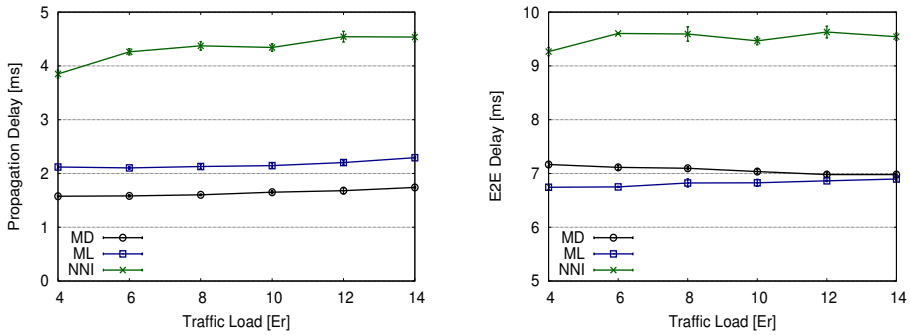
$$Processingtime[s] = Data[GB]/ProcessingCapacity[GB/s] \quad (2)$$

A. Experimental Results

The obtained results focus on comparing the performance attained by the considered allocation approaches: MD, ML and NNI. Such a comparison is done from a set of figures of merit, namely, *VNF-FG_Req* acceptance ratio, number of induced VLs, average setup time, attained propagation delay (i.e., latency), average SBVT utilization, bandwidth blocked ratio (BBR) and principal causes blocking *VNF-FG_Req*. In Fig. 5 the *VNF-FG_Req*'s acceptance ratio is illustrated as a function of the traffic Load. At the lowest traffic Load, the NNI algorithm presents comparable acceptance ratio values with respect to both ML and MD algorithms. We recall that NNI performs separated cloud and networking resource selections and does not address the minimization of the distance (in km) and the latency (in ms) as done in MD and ML. This results in fewer constraints to be fulfilled thereby obscuring the less efficient resource selection due to the separation. This explains the high values of acceptance rates for NNI, even higher than MD and ML for low traffic Load values(4 and 6 Er)where NNI achieves more than 99% of accepted requests. However, as traffic load increases, the performance of NNI significantly degrades (e.g., at traffic load set to 14 Er the acceptance rate is around 55%), whist MD and ML algorithms present acceptance ratio values smoothly decreasing, being stabilized around 85%. The rationale behind this is at higher offered traffic load, cloud and networking resources become more occupied and the separated cloud and networking selections made by NNI encounters more difficulties to be satisfied (especially for the cloud) despite fewer constraints need to be fulfilled. On the other hand, the advantage of joint selections becomes evident for ML and MD where where they allow attaining a more efficient use of the cloud and network resources thanks to the fact that both are in somehow jointly selected at the Allocator.

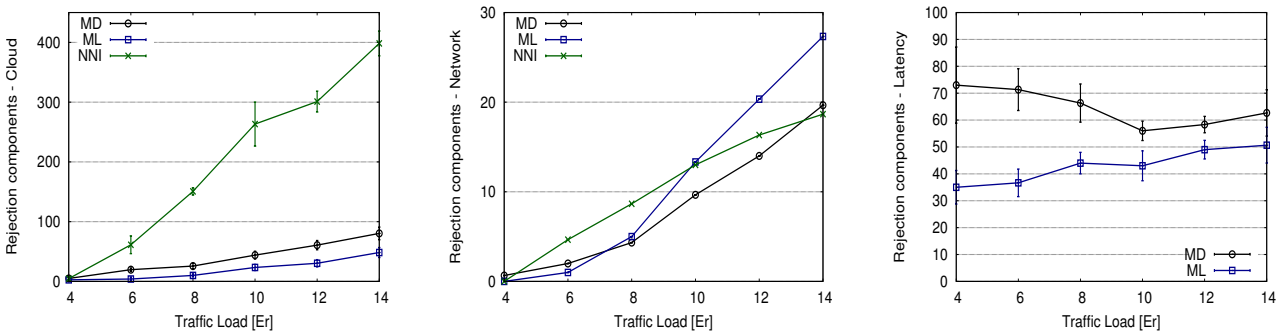
In the adopted MLN, packet VLs are supported by underlying and existing flexi-grid optical flows. Fig. 6 shows the total number of derived VLs for the three orchestration strategies when varying the traffic load. At low traffic load values, both MD and ML solutions lead to create a lower number of VLs compared to the NNI approach. This figure of merit reflects how each approach performs on the objective of attaining efficient use of the network resources, i.e. exploiting electrical grooming decisions. Creating less VLs means allocating less optical resources and saving packet ports' bandwidth which does increase the likelihood of successfully accommodating new *VNF-FG_Req*s. Increasing the traffic load, as expected, networking resources are more occupied and less *VNF-FG_Req*s are served which decreases the number of derived VLs.

In order to put in relation the number of created VLs with the ratio of accepted requests and to show how well an orchestration strategy performs on reusing spare available bandwidth on existing VLs, we derive the curve plotted in Fig. 7. We can observe that re-



(a) Propagation delay (b) End-to-end accumulated delay

Fig. 8: Obtained propagation and end-to-end delays vs. Traffic Load (Er)



(a) Cloud rejection occurrences (b) Network rejection occurrences (c) Latency rejection occurrences

Fig. 9: Main rejection causes.

gardless of the traffic load, both ML and MD strategies perform better than NNI and, in a similar way, demonstrate higher efficient selection of resources with, in particular, an enhanced reuse of the spare available bandwidth in VLs. Among them, the ML outperforms MD because it allows better tackling of the end-to-end latency requirement of *VNF-FG_Req*. This behaviour will be clarified in the following discussions.

Fig. 8 depicts the propagation delay (due to fiber links) and the end-to-end delay (i.e., propagation delay plus DC processing latency) for the orchestration approaches. As shown in Fig. 8(a), MD approach, aiming at minimizing distance between source and destination DCs, attains the lowest propagation delay. Conversely, as shown in Fig. 8(b), ML approach does minimize the end-to-end experienced latency (i.e. sum of the propagation and processing delays). As expected, NNI achieves the worst performance since it focuses on simply satisfying the *VNF-FG_Req*'s latency requirement without conducting an optimization of any delay source. In Fig. 9, we show the amount of occurrences for each of the three identified main reasons blocking a *VNF-FG_Req*: *i)* lack of cloud resources (when either source DC or destination DC does not have enough available cloud resources); *ii)* the network unavailability (when no feasible inter-DC path can be computed, e.g. no available bw, SBVTs' subtransponders, Gws' packet ports); and *iii)* unaccomplished *VNF-FG_Req* latency requirement. The rejection cause due to the DC lack of resources is depicted in Fig. 9(a). This numerously happens when adopting the NNI approach. The rationale behind that is that in the NNI approach the *VNF-FG_Req* imposes both source and the destination DCs. Therefore, the NNI orchestration strategy cannot make a selection of a (destination) DC checking whether enough compute resources are indeed available in it. However, in ML and MD strategies, the cloud and network resource orchestrator makes an explicit selection of the destination DC from a candidate set which ensures that enough available cloud resources are available to serve the *VNF-FG_Req*. Thus, this rejection reason becomes less impacting. Fig. 9(b) illustrates the network rejection occurrences. In general, for all the approaches, this rejection cause grows (almost linearly) as the traffic load increases mainly due to the occupation of the network resources. However, at high Traffic Load, ML is more impacted. The reason is that aiming at minimizing the end-to-end latency, ML tends to select the large DCs as destination ones. This makes that specific set of links towards those DCs become more saturated which does increase the networking rejection for the inter-DC paths. Finally, the latency rejection reason is represented in Fig. 9(c). Such a rejection

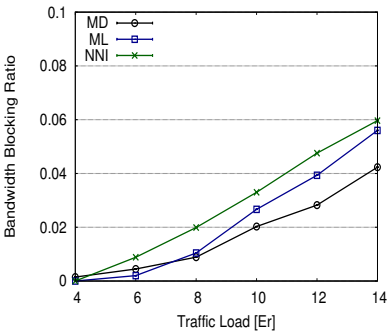


Fig. 10: Bandwidth Blocking Ratio vs. Traffic Load (Er).

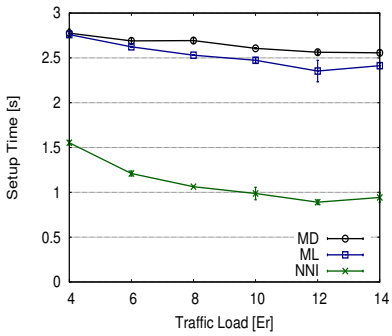


Fig. 11: Setup Time (s) vs. Traffic Load (Er).

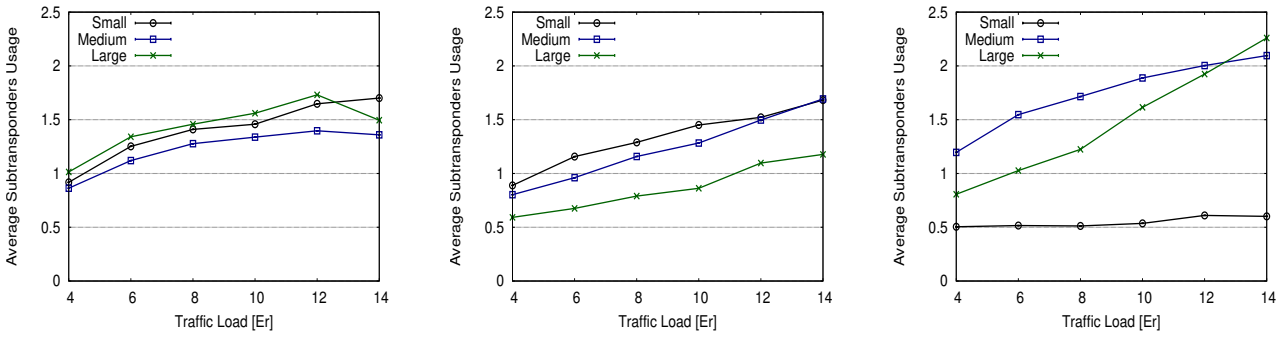
is significantly less happening in the ML strategy since it targets minimum end-to-end latency favoring satisfying *VNF-FG_Req*'s latency requirement. *VNF-FG_Req* demands different bandwidth requirements. The BBR figure of merit reflects the amount of bandwidth being blocked with respect to the total bandwidth being requested. Fig. 10 plots the attained BBR performance for each orchestration approach varying the traffic load. As traffic load is increased, the BBR, regardless of the orchestration approach, is degraded. Indeed, more networking resources (i.e., optical spectrum, SBVT's subtransponders and packet ports) are occupied to accommodate *VNF-FG_Req*. That said, NNI approach performs the worst. The rationale behind this is that ML and MD strategies foster more notably the reuse of the spare available bandwidth over existing VLs than NNI approach as also discussed in the acceptance ration figure of merit. Additionally, comparing both ML and MD approaches, the latter attains a slightly better BBR performance. As said above, ML aims at minimizing the end-to-end latency which in turn leads to more rapidly exhaust networking resources on specific links (those providing the connectivity towards larger DCs).

Fig. 11 depicts the average setup time (in s) with respect to the offered traffic load. The setup time is computed as the elapsed time between the reception of an incoming *VNF-FG_Req* and when it is successfully set up. This indicator is always higher in both MD and ML orchestration strategies than NNI because it results the price to pay for a truly joint cloud and network orchestration requiring further operations with the T-SDN Controller to acquire aggregated network resource (at packet layer) information. The setup time overhead introduced by ML and MD approaches with respect to the NNI strategy is slightly above 2 seconds which remains almost steady as traffic load grows. However, we observe increasing traffic load the average setup time performance for all the resource orchestration strategies tends to be reduced. Traffic load being increased entails that larger resources (both compute and networking such as Gw's packet port bandwidth) are more occupied. This makes also, from the networking perspective, to reduce the amount of created VLs as shown in Fig. 6. Indeed, the creation of VLs is one of the main contributions increasing the average setup delay in the T-SDN controller since it involves several control operations such as: *i*) configuration of the underlying flexi-grid optical connection associated to each VL; *ii*) creation of the VL requiring explicit control interactions with the network node endpoints (Gws); *iii*) and finally the dissemination using (BGP-LS protocol) of the created VLs. As a result, this requires significant time before a inter-DC packet connection is provisioned over a new virtual packet link. At high traffic load, since less VLs are created, the control operations handled by the T-SDN controller are reduced, which makes that successfully established inter-DC connections are attained requiring a lower average setup time.

Since different DC sizes are considered, focusing on how physical networking resources are occupied on each DC provides interesting insights to realize about the obtained performance. We take the average use of SBVT's subtransponders attached to the DC's Gws to this end. The idea is to correlate the usage of SBVT's subtransponders with the selection of specific DC type.

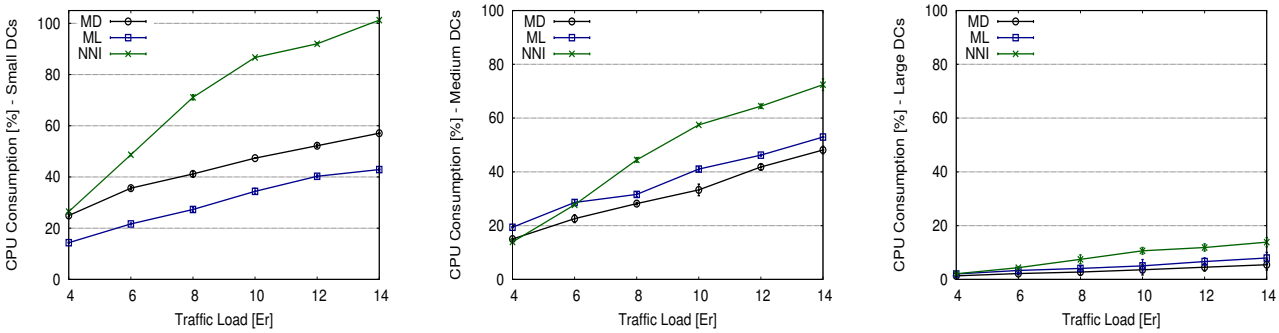
Fig. 12 depicts the average SBVT's subtransponders usage for DC size for each orchestration strategy. For the NNI approach, SBVT's subtransponder usage is distributed throughout all the DCs since both the source and destination DCs are imposed by the actual *VNF-FG_Req*. Thereby, the SBVT's subtransponder utilization is proportional to the amount of devices for each DC type within the considered topology. Adopting the MD approach (minimizing inter-DC distance), we observe that the SBVT's subtransponder at both small and medium size DCs are more occupied. In the considered MLN topology, minimum inter-DC distances are attained between pairs of small and medium DCs. However, in the ML approach (targeting minimum end-to-end inter-DC latency) medium and large DCs are preferred because of their shorter processing time.

Finally, in Fig. 13 the CPU consumption for each DC size is depicted. Observe that CPU consumption are very low for all the orchestration approaches in the



(a) NNI approach (b) MD Approach (c) ML approach

Fig. 12: Average SBVT's subtransponders Usage vs. Traffic Load (Er).



(a) Small DCs (b) Medium DCs (c) Large DC

Fig. 13: Average CPU Consumption [%].

large DC, Fig. 13(c), where resources are abundant. Small DCs (see Fig. 13(a)) are less selected in ML approach, as stated above, because they have a high processing time whilst, conversely, in MD are more used for minimizing the inter-DC distance.

VI. Conclusion

In this paper we presented an orchestrator to select and allocate network and cloud resources for network services (VNF-FGs) being deployed over distributed DCs. The interconnection of the DCs is provided by a MLN combining packet and flexi-grid optical (with SBVTs) technologies. We compared three different resource orchestration algorithms that select cloud and network resources while addressing specified requirements in terms of bandwidth and end-to-end latency. The algorithms differ in their abstraction and view of the resources on which they base their decisions (i.e., combined cloud and network resource view or separated) and in the objective function driving such a resource selection (i.e., minimization of end-to-end latency or the distance between DCs also impacting the latency performance of the service). A particular contribution of this work is the consideration of a MLN (packet over optical flexi-grid) set-up while performing resource orchestration. Moreover, we considered end-to-end delays; the former being dependent on the distance between DCs and the latter on the size of the DC. Indeed, DCs may present different computing capabilities based on their location in the network (e.g., small DC at the network edge closer to the users), affecting the offered processing capacity (e.g., smaller in edge DCs and larger in cloud DCs) and thus the offered processing delays. We experimentally evaluated the orchestration strategies using the CTTC ADRENALINE testbed. The presented experimental results show the effectiveness of the proposed algorithms in terms of higher resource utilization and acceptance rate of requests at the expense of higher (although almost steady) set-up time of services when the orchestrator handles more detailed information from the underlying infrastructure. In light of the obtained results, it can be stated that regardless of the considered network and cloud infrastructure if the latency demand of the VNF-FGs becomes a stringent requirement to be satisfied the application of the ML algorithm allows performing better (i.e., higher acceptance request ratio) when compared to the other proposed approaches (i.e., MD and NNI).

As future work we plan to refine the process to estimate the network delay not based on the physical distance but based on based on real and continuous measurements of delay experienced by data while traversing the network.

ACKNOWLEDGMENT

This work has been partially funded by the EC H2020 5G-Transformer Project (grant no. 761536).

REFERENCES

[1] A. Manzalini *et al.*, "Software networks at the edge: A shift of paradigm," in *Proc. IEEE Workshop SDN4FNS*, 2013.
[2] H. Hawilo *et al.*, "NFV: State of the art, challenges and implementation in next generation mobile networks (vEPC)," *IEEE Network*, vol. 28, Nov-Dec 2014.
[3] A. Hakiri *et al.*, "Leveraging SDN for the 5G networks: Trends, prospects and challenges," in *preprint arXiv:1506.02876*, 2015.
[4] J. Ordonez-Lucena *et al.*, "Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges," *IEEE Comm. Mag.*, vol. 55, no. 5, May 2017.
[5] B. Martini and F. Paganelli, "A service-oriented approach for dynamic chaining of virtual network functions over multi-provider software-defined networks," *Future Internet*, vol. 8, no. 2, 2016. [Online]. Available: http://www.mdpi.com/1999-5903/8/2/24
[6] ETSI, "network functions virtualisation (NFV) management and orchestration," *ETSI, GS NFV-MAN 001*, 2014.
[7] Huawei *et al.*, "Mobile edge computing introductory technical white paper," *Tech. Rep.*, 2014.
[8] A. Rostami *et al.*, "An end-to-end programmable platform for dynamic service creation in 5G networks," in *Proc. of OFC*, 2017.
[9] A. A. Mohammed, M. Gharbaoui, B. Martini, F. Paganelli, and P. Castoldi, "Sdn controller for network-aware adaptive orchestration in dynamic service chaining," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, June 2016, pp. 126–130.
[10] M. Gharbaoui, B. Martini, D. Adami, G. Antichi, S. Giordano, and P. Castoldi, "On virtualization-aware traffic engineering in openflow data centers networks," in *2014 IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–8.
[11] B. Martini *et al.*, "Latency-aware composition of virtual functions in 5G," in *Proc. IEEE NetSoft*, 2015.
[12] P. Lu and Z. Zhu, "Data-oriented task scheduling in fixed- and flexible-grid multilayer inter-DC optical networks: A comparison study," *IEEE/OSA JLT*, vol. 35, Dec 2017.
[13] M. Zeng *et al.*, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *IEEE/OSA JLT*, 2016.
[14] "Cross-layer resource orchestration for cloud service delivery: A seamless sdn approach," *Computer Networks*, vol. 87, pp. 16 – 32, 2015.
[15] R. Martinez *et al.*, "Experimental evaluation of a PCE transport SDN controller for dynamic grooming in packet over flexi-grid optical networks," in *Proc. of 43rd ECOC*, April 2015.
[16] R. Martinez *et al.*, "Control plane solutions for sliceable bandwidth transceiver configuration in flexi-grid optical networks," *IEEE Commun. Mag.*, vol. 54, Aug 2016.
[17] S. Zhang *et al.*, "Evolving traffic grooming in multi-layer flexible-grid optical networks with software-defined elasticity," *IEEE/OSA JLT*, vol. 32, Aug 2014.
[18] J. Zhang *et al.*, "Energy-efficient traffic grooming in sliceable-transponder-equipped IP-over-elastic optical networks," *IEEE/OSA JOCN*, vol. 7, Jan 2015.
[19] S. Zhang *et al.*, "Dynamic traffic grooming in elastic optical networks," *IEEE JSAC*, vol. 31, Jan 2013.
[20] B. Jennings and R. Stadler, "Resource management in clouds: Survey and research challenges," *Journal of Network and Systems Management*, vol. 23, no. 3, 2015.
[21] M. Alicherry and T. V. Lakshman, "Network aware resource allocation in distributed clouds," in *Proc. of IEEE INFOCOM*, March 2012.
[22] W. Huang *et al.*, "Cooperative data center selection for optimal service performance: An ILP formulation," in *IEEE 10th ISPA*, 2012.
[23] C. Develder *et al.*, "Joint dimensioning of server and network infrastructure for resilient optical grids/clouds," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, Oct. 2014.
[24] A. Haider *et al.*, "Challenges in resource allocation in network virtualization," *20th ITC Specialist Seminar*, vol. 18, 2009.
[25] A. Fischer *et al.*, "Virtual network embedding: A survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, Fourth 2013.
[26] T. Trinh *et al.*, "Quality of service using careful overbooking for optimal virtual network resource allocation," in *8th International Conference on ECTI-CON*, 2011.
[27] X. Zhang *et al.*, "An overlay mapping model for achieving enhanced QoS and resilience performance," in *3rd ICUMT*, 2011.
[28] M. Rahman *et al.*, "Survivable virtual network embedding," *Networking*, 2010.
[29] G. Sun *et al.*, "The framework and algorithms for the survivable mapping of virtual network onto a substrate network," *IETE Technical Review*, vol. 28, 2011.
[30] J. F. Botero *et al.*, "Energy efcient virtual network embedding," *IEEE Commun. Lett.*, vol. 16, May 2012.
[31] J. Shamsi *et al.*, "Qosmap: Qos aware mapping of virtual networks for resiliency and efficiency," *IEEE Globecom Workshops*, November 2007.
[32] J. Shamsi *et al.*, "Efficient and dependable overlay networks," in *IPDPS 2008. IEEE International Symposium on*, April 2008.
[33] J. Shamsi *et al.*, "Qosmap: Achieving quality and resilience through overlay construction," in *4th ICIW*, 2009.
[34] Z. Zhu *et al.*, "Build to tenants' requirements: On-demand application-driven vsd-eon slicing," *IEEE/OSA JOCN*, vol. 10, Feb 2018.
[35] B. Kong *et al.*, "Demonstration of application-driven network slicing and orchestration in optical/packet domains: On-demand vdc expansion for hadoop mapreduce optimization," *Optics Express*, vol. 26, 2018.
[36] M. Savi *et al.*, "To distribute or not to distribute? impact of latency on virtual network function distribution at the edge of FMC networks," in *Proc. of 18th ICTON*, July 2016.
[37] S. Mehraghdam *et al.*, "Specifying and placing chains of virtual network functions," in *3rd IEEE CloudNet*, Oct 2014.
[38] H. Moens *et al.*, "VNF-P: A model for efficient placement of virtualized network functions," in *10th CNSM*, Nov 2014.
[39] F. Bari *et al.*, "On orchestrating virtual network functions," in *11th CNSM*, 2015.
[40] B. Sonkoly *et al.*, "UNIFYing cloud and carrier network resources: An architectural view," in *IEEE GLOBECOM*, 2015.
[41] A. Csoma *et al.*, "Escape: Extensible service chain prototyping environment using mininet, click, NETCONF and POX," in *ACM SIGCOMM*, 2014.
[42] B. Gero *et al.*, "The orchestration in 5G exchange-a multi-provider NFV framework for 5G services," in *IEEE NFV-SDN*, 2017.
[43] https://osm.etsi.org/.
[44] https://cloudify.co/.
[45] https://www.onap.org/.
[46] K. Antevski, J. Martn-Prez, N. Molner, C. F. Chiasserini, F. Malandrino, P. Frangoudis, A. Ksentini, X. Li, J. SalvatLozano, R. Martnez, I. Pascual, J. Mangues-Bafalluy, J. Baranda, B. Martini, and M. Gharbaoui, "Resource orchestration of 5g transport networks for vertical industries," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 158–163.
[47] L. Valcarenghi, B. Martini, K. Antevski, C. J. Bernardos, G. Landi, M. Capitani, J. Mangues-Bafalluy, R. Martinez, J. Baranda, I. Pascual, A. Ksentini, C. F. Chiasserini, F. Malandrino, X. Li, D. Andrushko, and K. Tomakh, "A framework for orchestration and federation of 5g services in a multi-domain scenario," in *Proceedings of the Workshop on Experimentation and Measurements in 5G*, ser. EM-5G'18, 2018, pp. 19–24.
[48] ETSI, "network functions virtualisation (nfv) release 3; management and orchestration; report on management and connectivity for multi-site services," *ETSI, GR NFV-IFA 022 V3.1.1*, 2018.
[49] S. Fichera *et al.*, "Experimental evaluation of orchestrating inter-DC quality enabled VNFFG services in packet/flexi-grid optical networks," in proc. *of ECOC*, Sept 2018.
[50] X. Wang *et al.*, "Five resource allocation strategies for the cloud infrastructure," in *Proc. IEEE ONDM*, 2013.
[51] M. Gharbaoui *et al.*, "An orchestrator of network and cloud resources for dynamic provisioning of mobile virtual network functions," in *Proc. of IEEE NetSoft*, 2016.