VNF placement for service chaining in a distributed cloud environment with multiple stakeholders

VNF Placement for service chaining in a distributed cloud environment with multiple stakeholders

Paola Cappanera[a], Federica Paganelli[b,*], Francesca Paradiso[a]

[a] *Department of Information Engineering, University of Florence, Italy*
[b] *CNIT, Research Unit at the University of Florence, Italy*

Abstract

The adoption of virtualization technologies in networking is promoting a radical innovation in the way network services are managed and delivered. Indeed, some network services may be provisioned to cope with complex and unpredictable traffic demands by dynamically creating a sequence of Virtual Network Functions (VNFs) and steering traffic flows through them. In this context, the optimized deployment of network services, composed of VNFs that may be instantiated in multiple Data Centers (DCs), is one of the most challenging orchestration target. VNF placement is the problem of choosing the set of optimal locations for a chain of VNFs according to the service request and the current characteristics of available computing resources and network links. With respect to the state of the art, our original contribution reflects a multi-stakeholder perspective (subscriber, service providers, infrastructure providers) in a multi-DC environment. We thus consider the problem of placing VNFs to maximize primarily the number of accepted requests from a set of incoming requests and secondarily the satisfaction of subscribers' preferences. Our model also allows to differentiate service requests in priority levels and guarantees that Quality of Service objectives for accepted service requests are fulfilled, including also a requirement on network service instantiation time. We provide an integer linear programming formulation of this problem that leverages a layered auxiliary graph built for each request in a set. Experimental evaluation is described in detail and

*Corresponding author

Email addresses: `paola.cappanera@unifi.it` (Paola Cappanera), `federica.paganelli@cnit.it` (Federica Paganelli), `francesca.paradiso@unifi.it` (Francesca Paradiso)

The final version of this article is available at https://doi.org/10.1016/j.comcom.2018.10.008

October 17, 2018

an assessment of the proposed placement approach is performed along three main directions: (*i*) service acceptance ratio in online and offline placement, (*ii*) preferences' satisfaction, and (*iii*) scalability expressed in terms of computational time. The performance of the approach is also compared to a greedy heuristic.

Keywords: Network Function Virtualization, Network Service, Service Chaining, VNF Placement, NFV Orchestration, Optimization Techniques

1. Introduction

Network Function Virtualization (NFV) is a paradigm proposed by the European Telecommunication Standardization Institute (ETSI)[1] to facilitate dynamic provisioning of network services through virtualization technologies. In this vision, network services can be implemented by chaining a set of functions, implemented either on dedicated hardware as Physical Network Functions (PNFs), or as software components on top of virtualized general-purpose hardware, i.e., Virtual Network Functions (VNFs). The adoption of virtualization allows flexible lifecycle management of network services as well as of their VNF components (e.g., creation, deletion, horizontal or vertical scaling operations). In this way, resource usage can be adapted to current demand and business targets, also avoiding the adoption of over-provisioning policies [2].

Software-Defined Networking (SDN) [3] complements NFV by offering programmatic access to abstracted network resources and full programmability of forwarding capabilities. Indeed, SDN control capabilities may be used to implement dynamic traffic steering policies so that flows are dynamically routed along a path traversing the VNF instances composing a given network service [4].

NFV and SDN technologies together introduce a level of flexibility in network service provisioning that is key for coping with requirements of complex and unpredictable traffic patterns in modern networking systems, such as Internet of Things, cloud networking and mobile data traffic toward new fifth generation (5G) networks [5, 6]. Indeed, NFV and SDN are jointly considered key technologies for supporting the degree of flexibility required by network slicing techniques in future 5G networks [7] as well as dynamic demand for low latency applications (Mobile Edge Computing [8]).

In this context, appropriate orchestration mechanisms are required to support such operational flexibility and make services more responsive to customer needs, while guaranteeing the achievement of target operating margins [9]. Therefore, orchestration mechanisms should account for both business value and customer experience, which can be represented as two conflicting goals, respectively: i) cost-effective resource utilization, to achieve the target range of operating margins (business performance); and ii) fulfillment of Quality of Service (QoS) objectives [10] specified in the Service Level Agreement (SLA) between a customer and a service provider and typically expressed as technical performance metrics.

In this scenario, the optimized deployment of network services, composed of VNFs that may be instantiated in multiple distributed Data Centers (DCs), is one of the most challenging orchestration target [11].

VNF placement is the problem of choosing the set of optimal locations for chained VNF instances according to the current characteristics of available computing resources and network links. Optimality has been defined in different ways in the literature (e.g., minimization of the overall delay or of deployments costs, maximization of remaining bandwidth, etc.).

However, a broader perspective on VNF placement in a distributed multi-DC environment, which also considers the needs of stakeholders, may help in eliciting novel criteria to be taken into account. Indeed, network operators are facing the problem of orchestrating resources so to profitably run VNFs, i.e., efficiently managing capital and operational expenditures (CAPEX and OPEX, respectively), while fulfilling SLAs agreed with subscribers [9]. The industrial research community [12] is also arguing whether there is a real benefit in minimizing SLA objectives, such as latency. Indeed, satisfiability seems to be more important than optimization in this context and admission control techniques are usually employed to determine whether latency targets can be met. As a consequence, the industrial community is looking for more pragmatic approaches, such as decision policies aiming at maintaining technical performance objectives within an acceptable range [9], while maximizing request acceptance rate [11, 13].

Hence, while most recent works focus on optimizing technical performance objectives (e.g., end-to-end delay and remaining bandwidth)[14–16] or cost minimization [17–22] in a joint VNF placement and routing problem, in this work we analyze the VNF placement problem and related orchestration scenario from a business perspective. The aim is to derive stakeholders' main requirements and specify the problem statement and optimization objectives accordingly.

We consider three types of stakeholders: *subscribers*, asking for the provision of network services, *service providers*, providing network services executed on top of virtual and physical infrastructure resources, and *infrastructure providers*, providing and managing virtual and physical infrastructures. We elaborate on their needs and their mutual interactions within a reference NFV/SDN architecture based on current standards [23, 24]. We then formulate our VNF Placement problem that reflects such multi-stakeholder perspective, including possible constraints on the extent to which detailed information about the infrastructure status is shared among stakeholders. We thus consider the problem of placing VNFs to maximize the number of accepted requests and subscribers' preferences, while delegating the possible optimization of technical performance objectives, such as latency or congestion minimization, to intra-domain orchestration mechanisms (e.g. online traffic engineering techniques implemented on top of SDN Controller North Bound interfaces [25]). We formulate the problem by means of a 0-1 Integer Linear Programming model. Our model allows to differentiate service requests in priority levels and guarantees that customized QoS objectives for accepted service requests are fulfilled, including also a requirement on network service instantiation time.

Subscribers can also express preferences and bans over infrastructure sites (i.e., DCs), so that placement decisions may take into account personal or organization values and concerns (e.g., sustainability, ethics, reputation, etc.). In order to cope with the elements discussed above, we model the infrastructural resource substrate by introducing features not considered in previous works, including available virtualization technology, such as Virtual Machines (VMs) vs containers, and DC's carbon footprint. A preprocessing phase is also provided that has a three-fold aim: (i) discard all those requests that cannot be accomplished by the system for infeasibility reasons, (ii) define the incompatibilities between a specific VNF of a given request and a DC (e.g., due to commercial or organization policies or DC's insufficient capacity), and (iii) process subscribers' preferences so that they are taken into account in the optimization model.

Summarizing, our work addresses the maximization of the request acceptance rate, while taking into account subscribers' preferences, priority levels and the fulfillment of QoS objectives. These issues have recently been identified in the literature [11, 13] as some of the more relevant criteria that need to be taken into account by novel VNF placement approaches.

The remainder of this paper is organized as follows. Section 2 discusses related work and highlights our contribution. In Section 3 we present the reference scenario and state the problem. Section 4 discusses the computational complexity of the problem addressed and presents the optimization model proposed. Section 5 describes the preprocessing phase in detail. Performance evaluation results are reported in Section 6. Finally, Section 7 concludes the paper with insights for future work.

2. Related Work

The problem of how effectively deploying and managing network services conceived as a chain of VNFs has raised a considerable interest in the research community. The rest of this section is organized as follows. First, we briefly analyze the literature on routing and placement for optimizing QoS metrics. Then, we analyze works targeting minimization of costs. Finally, we focus on stakeholders' perspectives, that is a crucial issue in our study. We then conclude by discussing our contribution with respect to the state of the art.

Many works jointly address VNF placement and routing problems to optimize specific QoS metrics, typically within a DC or in an operator's network. Liu et al. [14] consider two performance metrics, i.e., end-to-end delay and bandwidth consumption. They propose an integer linear program (ILP) and design two heuristic algorithms, i.e., a greedy algorithm and a simulated annealing approach. The work in [15] addresses both chain composition and placement. Specifically, it proposes an under-specified structure of a composed service that allows to dynamically modify the order of VNFs in a chain and a heuristic algorithm that places service components along the shortest paths. Bhamare et al. [16] formulate the problem of minimizing inter-cloud traffic and response time in a multi-cloud scenario as an ILP problem and propose an affinity-based allocation heuristic approach for solving it.

Several approaches have been proposed to minimize costs of running VNFs on virtual infrastructures, while fulfilling SLAs. Bari et al. [17] propose an exact approach for small networks and a heuristic for larger networks based on a multi-stage graph with the objective of minimizing total network operational cost and resource fragmentation.

Mechtri et al. [18] propose both an approach based on the eigendecomposition of adjacency matrices of the request and the infrastructure graphs, and a heuristic algorithm for finding the maximum weight matching. Leivadeas et al. [19] propose a set of algorithms that target minimization of provisioning costs as well as efficient resource usage. Gadre et al. [20] introduce a divide-and-conquer algorithm and a heuristic aiming to minimize an overall cost, assuming that the routes for the flows are a priori given and VNFs in the request have instance and service costs associated. The solution in Pham et al. [21] is based on a Markov approximation approach combined with matching theory. A stable and efficient matching is searched for that takes into account the service chain's preference over nodes (nodes with the greatest amount of available resources are preferred) as well as nodes' preferences over VNFs (based on the adopted consolidation policy). More recently, ASPER [22] is an automated approach for the joint scaling, placement and routing of network services, whose objective is to find a minimal number of constraint violations (i.e., CPU, memory and link capacity constraints) that is Pareto optimal with respect to a set of secondary objectives (e.g., total delay, total resource consumptions, etc.).

Recently, authors have begun explicitly contextualizing cost minimization and efficient resource usage problems in a multi-DC setting. Liberati et al. [26] propose a stochastic algorithm based on reinforcement learning (RL) that maximizes an expected mapping reward, which may be configured to target different objectives, such as costs minimization, load balancing or maximization of the acceptance rate. Implementation cost minimization as well as acceptance rate maximization are jointly addressed in [27] through two approximation algorithms. Luizelli et al. [28] propose a novel fix-and-optimize-based heuristic algorithm to minimize resource allocation, while meeting network flow requirements and constraints and addressing scalability. Wang et al. [29] address the cost-effective provision of VNF graphs in inter-DC optical networks in a multidomain environment (i.e., private and public domains). The problem is formulated as an ILP that models compute and network bandwidth constraints, and minimize the cost of compute resources and frequency slot usage on links. Gupta et al. [30] propose an approach that aims to reduce network resource consumption for a WAN interconnecting DCs by defining and placing multiple instances for each service chain. Gupta et al. [31] formulate an ILP to minimize usage of network resources, while evaluating four different deployment choices (e.g., hardware-based middleboxes, DCs, NFV-capable network nodes, etc.). Ayoubi et al. [32] consider both VNF placement and policy-aware traffic steering to maximize the number of served flows. The problem is decomposed into a master problem (placement) and a subproblem (policy-aware routing of every flow along the designated VNF instances). The model can be used to solve either an online or an offline problem. In the former case, the set of input requests is a batch of requests arrived within a time window, in the latter case it represents all flow requests, known in advance. Finally, in [33] the optimal placement of VNF chains is addressed and shown to be NP-complete but for very special cases. The authors also propose two polynomial time algorithms that can be used to determine a feasible solution to a simplified variant of the optimal VNF placement problem which occurs when the following two assumptions hold: (i) each VNF typology is hosted in one physical server; (ii) each traffic flow is splittable. Both the two approaches, referred to as the matrix-based algorithm and the multi-stage graph algorithm, use a maximum flow algorithm as a subtool, guarantee capacity constraints at the servers and bandwidth constraints on the links. Other kinds of constraints on the request, such as for example those concerning latency, are disregarded.

Focusing on stakeholders' perspective and business requirements, Altmann and Kashef [34] analyze cloud computing cost factors in federated hybrid clouds and propose a cloud cost model. They also propose a service placement optimization algorithm, which identifies the cost-minimizing service placement option through exhaustive search. Recently, Naudts et al. [35] consider the problem of service chain from an original perspective: indeed, they aim at increasing the infrastructure providers revenue by proposing a dynamic pricing algorithm where the requested substrate resources are priced on the basis of historical data, current infrastructure utilization levels and competitors' price.

While the main body of previous literature mainly addresses either the optimization of performance objectives [14–16] or takes into consideration the service providers' need of minimizing costs for service deployment and operation [17–22], in this work we develop a new concept of VNF Placement by moving from business requirements and considering the perspectives of three types of stakeholders (subscribers, service providers and infrastructure providers) to different extents. Similarly to our work, in [35] the problem statement originates from the analysis of roles stakeholders play in an NFV/SDN environment, but for a completely different problem. In addition, our work can be seen as a complement of [35] in that it allows to represent different pricing schemes for the requested substrate resources and manages DC preferences on behalf of price-sensitive consumers. Analogously to [21], we handle service chain's preferences over nodes, but in our case such preferences are configurable and their weight can be customized for each service request. Moreover, some works are explicitly contextualized in a multi-DC setting (e.g., [16, 26–32, 34], but they do not take into account possible limitations in information disclosure among different operators, as this work does.

Summarizing, our work contributes to the literature in the following directions: (*i*) it aims at jointly maximizing service providers' profits in terms of accepted requests and satisfaction rate of subscribers' preferences, while fulfilling SLA requirements and considering an abstracted multi-DC network topology complying with possible information disclosure limitations among operators; (*ii*) it allows taking into account different priority levels and accommodate requests that need a fast deployment, as long as the substrate network may support them depending on the virtualization technology offered by nodes; (*iii*) it characterizes DC nodes in terms of their carbon footprint (we take Carbon Usage Effectiveness metric (CUE)[36] as reference metric) and pricing schemes. This allows users to express optional preferences on DCs that implement sustainable energy policies (i.e., those showing the lowest CUE values) and/or are more economically convenient. In regards to sustainability, as far as we know, Khosravi et al. [37] consider a similar parameter, i.e., Power Usage Effectiveness (PUE) but for a different purpose, i.e., energy- and carbon-efficient placement of VMs in distributed DCs.

3. Problem Statement

We consider a reference scenario for NFV orchestration characterized by the following three types of stakeholders: *Subscribers*, *Service providers*, *Infrastructure providers*. Hereafter, we introduce the main concepts of our reference scenario, in terms of stakeholders' perspectives and reference architectural guidelines, and then formulate the problem.

3.1. Subscriber's perspective

A Subscriber is an actor (also referred to as user or customer) that requests the provisioning of a network service. We model the subscriber needs in terms of both a set of QoS objectives that represent desired service performance, and preferences regarding possible VNF deployment options (i.e., preferences over available infrastructure sites).

In this work we consider the following QoS parameters: maximum tolerated latency, minimum guaranteed bandwidth, and network service instantiation time. The fulfillment of these objectives, when specified in the request, is mandatory, otherwise the request cannot be satisfied. While the first two objectives are quite common, the third objective concerns network service instantiation time and becomes effective when subscriber requires that network service is deployed and launched "as soon as possible". This requirement is taken into account through a policy enforcing that the network service is deployed on the appropriate infrastructure technology. In this work we take two alternative virtualization technologies as reference, VM vs. containers. Since container technologies may guarantee a shorter startup time with respect to VMs [38], when the subscriber requests a fast service setup, the orchestration maps such requirement into a specific constraint (i.e., deploying the network service components on containers). Although not yet widely considered in the literature, the specification of a requirement on instantiation time in VNF Placement is especially relevant if network service requests have to be satisfied as soon as they arrive (such as for online service requests [39]) to cope with dynamic user demands.

As regards preferences for VNF deployment, subscribers can specify preferences to be taken into account by service providers in the selection of the infrastructure site. Indeed, subscribers preferences typically regard pricing and technical performance metrics, but can also include additional attributes, such as provider reputation, ethicality and stability [40]. For instance, preferences can also require that environmental objectives are taken into account and services are provided with the smallest carbon footprint, as specified in emerging green or energy-aware SLAs [41, 42].

3.2. Service provider's perspective

The role of service providers consists in handling network service requests. They offer network services to subscribers and are therefore in charge of correct service provisioning and lifecycle management. Service providers can buy/lease service components and infrastructure from other providers (i.e., service providers and infrastructure providers). In this case, which is introduced by ETSI as NFVI as a Service (NFVIaaS) in [43], service providers have control on services, while infrastructure operators control the infrastructure. Typically, a service provider can choose the provider infrastructure domain and the site where VNFs should be placed.

Service providers aim at optimizing business value [9]. We mapped this requirement into the maximization of accepted network service requests, with respect to available infrastructure resources and a maximum accepted cost for service operation. In accordance with subscribers' perspective, service providers may also desire to minimize costs for service hosting on the physical substrate to improve target operating margins.

Service providers may also assign different levels of priority to incoming requests, depending on subscribers' profiles and application-based traffic differentiation (e.g., Service Classes defined in DiffServ specifications [44]). Requests which have a higher priority level will get preferential treatment with respect to lower priority requests.

3.3. Infrastructure provider's perspective

These actors offer virtual and physical resource infrastructures (e.g., DC providers and inter-DC Wide Area Network operators). Here, we consider an infrastructure provider that manages a multi-DC infrastructure, offering resources at a given price for capacity unit. Offered prices can vary from DC to DC. Resource offers by infrastructure providers can also be enhanced with information related to the carbon footprint of a DC. The infrastructure providers' perspective is modeled in this work as the requirement of efficiently using the infrastructure resources by balancing the load across multiple sites to avoid overhead conditions. Within a DC, an infrastructure provider may apply its own decision policies to orchestrate physical resources to optimize a given utility function (e.g., minimize power consumption, maximize server consolidation), but this problem is outside the scope of this work.

3.4. Reference architecture for network service provisioning

Hereafter, we briefly describe an NFV/SDN-based reference architecture for network service provisioning, elaborated by taking into account standard guidelines and architectural models promoted by the NFV ETSI Industry Specification Group [1, 23, 24]. ETSI specifications define a set of *Management and Orchestration (MANO) functions* , which include: i) a *Virtual Infrastructure Manager* (*VIM*) responsible for managing physical, virtual and software resources of related NFV Infrastructures (NFVI); ii) a *VNF Manager* handling the lifecycle of VNFs; and iii) a *VNF Orchestrator* (*VNFO*) managing the lifecycle of network services.

Fig. 1 shows a reference architecture for network service provisioning in a multiple stakeholder and multi-DC environment. This architecture integrates some ETSI functional blocks mentioned above with SDN network control capabilities within each DC domain and in the WAN segments interconnecting the DCs. For the sake of clarity only two DCs and one WAN segment that provides "on-demand connectivity services" are depicted in Fig. 1. The

WAN Infrastructure Manager (WIM) leverages the services provided by an SDN Controller and offers a North-Bound application interface [24]. We introduce two additional functional blocks: a *Service Portal* and a *Service Orchestrator*. The Service Portal offers a Graphical User Interface (GUI) to subscribers for selecting and requesting the provision of a network service with a given SLA to a service provider. The Service Orchestrator is responsible for the acceptance of service requests and for service deployment and management operations. For the scope of this article, we outline two main components of the Service Orchestrator: *a Service Request Manager* and an ETSI-compliant NFVO. The former handles incoming service requests, by mapping business-level service requests coming from the Service Portal into network service instantiation requests to the NFVO. For this purpose the Service Request Manager also performs decision making steps, including VNF placement, which is actually the target of our work. The NFVO manages such network service instantiation requests by handling the interaction with the affected VIMs and WIM. In accordance with the placement decision taken by the Service Request Manager, it generates appropriate requests for instantiating the VNFs (to the VIMs) and for enforcing the appropriate forwarding instructions (to VIMs and WIM) for steering traffic flows through the deployed chains.

Fig. 1 shows how different stakeholders are involved in network service provisioning. As also discussed in [11], VNF deployment and connectivity decisions could be taken at a single point (Service Orchestrator), which, to perform optimal decisions, needs to receive full NFVI information from NFVI control and management systems. However, since service and infrastructure providers can be different operators, this would require the full disclosure of internal details across different administrative domains. On the contrary, the NFVI provider could decide to expose only an abstracted view of resources and topologies [45] and hide internal details. We therefore consider a scenario where the responsibility of the Service Provider consists in deciding in which DCs VNFs should be placed considering an abstracted view of the NFVI, thus minimizing the type of monitoring and status information to be gathered from VIMs and WIMs (although leading to a suboptimal decision with respect to the previous case). This allows NFVI operators to hide internal implementation and status details, and finetune deployment decisions within their own organization domain boundaries.

In this work we consider a multi-domain NFV Infrastructure made by a set of geographically distributed infrastructure sites of different size (e.g.,

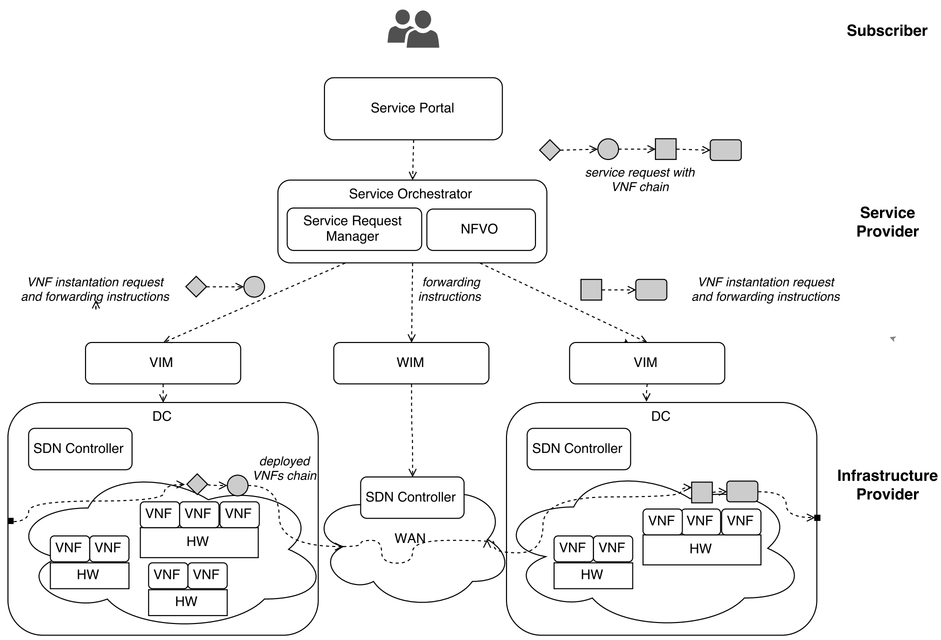


Figure 1: Reference NFV/SDN architecture

from micro to big DCs [46]). A DC is a container of physical hosts where one or more VNFs can be deployed. Each DC exposes its infrastructural resources at a given price per capacity unit and it is characterized by an energy efficiency and greenness metric (e.g., CUE) in order to promote sustainability assessments and comparisons among DCs. At a given instant in time, each DC is characterized by the amount of available resources (Capacity), such as CPU and memory. In this work, capacity is considered a multi-dimensional parameter in problem statement, while it is one-dimensional in the experimental testing as widely assumed in the literature ([47]). As discussed above, we also characterize DCs in terms of their technological infrastructure (e.g., availability of container technology).

Incoming network service requests include the specification of a service function chain as an ordered sequence of VNFs, or more precisely, types of VNFs (e.g., NAT, firewall, etc.). We assume that the whole chain has to be instantiated preserving the order of the sequence. For each VNF type, the amount of requested resources is provided. The request is further characterized by a *source node* (the source of the traffic flow) and a *destination node* (the destination of the traffic flow), priority levels, QoS parameters (maximum latency, minimum bandwidth and fast network service instantiation time) and maximum cost that can be afforded to deploy the service.

3.5. Problem formulation

Starting from a realistic network configuration where nodes correspond to forwarding elements or storage and compute elements in DCs, and links connect such nodes, we build an abstract network $G = (D, E)$ where nodes correspond to DCs and each arc $(i, j) \in E$ between DC i and DC j represents a path in the original network between nodes i and j. Specifically, arc $(i, j) \in E$ corresponds to the path with minimum latency among all the paths connecting nodes i and j in the original network. All arcs belonging to E are bidirectional. In addition, we define T (indexed by t) as the set of priority levels, R (indexed by r) as the set of service requests, and N (indexed by n) as the set of resources offered by the DCs service requests compete for. As an example, two types of requests can be considered: requests for premium services and requests coming for best effort services. In such a case, T would have cardinality two. In regards to the resources, typical resources considered in set N are CPU, RAM and storage, as an example. Sets used to state the problem formally are summarized in Table 1.

Table 1: Sets

D	set of nodes in the abstract network (each node corresponds to a DC)
E	set of arcs in the abstract network (arc (i, j) corresponds to a path from DC i to DC j)
T	set of priority levels
R	set of service requests
N	set of resources offered by DCs

In the following, a detailed description of network G in terms of nodes D and arcs E is given. We assume that each arc $(i, j) \in E$ is characterized by the parameters described in Table 2.

Table 2: Arc parameters

l_{ij}	latency of arc (i, j) expressed in ms
b_{ij}	available bandwidth of arc (i, j) expressed in Gbps

In this work latency refers to the propagation delay on the link which separates two nodes, thus it is directly dependent on the physical distance between them. Due to network abstraction, the bandwidth of an arc (i, j)

is the minimum bandwidth over all the links on the minimum latency path from i to j in the original network.

Each node i in D, i.e., each DC, is characterized by the parameters described in Table 3.

Table 3: Node parameters

u_i^n	capacity of i in terms of resource n
$\overline{p}_i^n$	upper percentage utilization of DC i relative to resource n
s_i	equal to 1 if i provides container, 0 otherwise
c_i	price of i per capacity unit
f_i	carbon footprint of i

For each DC i, the resource capacity parameter u_i^n represents its capacity in terms of resource n and $\overline{p}_i^n$ is a parameter which defines the maximum percentage utilization of i in terms of resource n. As mentioned above, the parameter s_i refers to the capability of DC i to instantiate VNFs in a container such as Docker [48], in order to allow a quicker service provision by avoiding setup time due to VM instantiation. Finally, c_i corresponds to the unitary price exposed by DC i and f_i refers to CUE as specified above.

The Orchestrator has to manage a set R of service requests characterized by different typologies. Specifically, for each priority level $t \in T$, R_t is the set of requests of typology t. Sets R_t, $\forall t$ define a partition of set R, i.e., $\cup_t R_t = R$, $R_{t^i} \cap R_{t^"} = \emptyset \, \forall t^i, t^" \in T$. Each request r in R, is characterized by the parameters described in Table 4.

The proposed model can be used to solve either the online or offline VNF placement problem. In the offline case, R represents the whole set of service requests, to be known in advance, whereas in an online problem, R represents a batch of requests arrived within a time window.

In regards to instantiation time, we point out that when a certain request r requires the instantiation time to be as short as possible (i.e., $s^r = 1$), all of the VNFs of its chain H^r must be placed on DCs equipped with container technology (if available). Preferences and incompatibilities between the VNFs in a request and DCs are computed in a pre-processing phase as detailed in Section 5.

We conclude the section by recalling all those features that characterize the problem studied both in terms of objective function and constraints. The problem is then mathematically formulated in Section 4.2.

Table 4: Request parameters

t	priority level
o^r	origin node of traffic request r - ingress node
d^r	destination node of traffic request r - egress node
l^r	maximum end-to-end delay tolerated by r
b^r_{hh+1}	minimum data rate capacity (bandwidth) accepted by r from the h-th VNF to the $(h+1)$-th VNF
c^r	maximum cost r is willing to pay to get service
s^r	equal to 1 when r requires short service instantiation time; 0 otherwise
f^r	equal to 1 if r is interested in environmental impact; 0 otherwise
$H^r = \{V^r_1, V^r_2, .., V^r_{\|H^r\|}\}$	ordered sequence of VNFs composing r ($\|H^r\|$ is the length of the chain)
$u^n_{V^r_h} \quad \forall h \in \{1, .., \|H^r\|\}$	quantity of resource n required by the h-th VNF of r
$p^r_{V^r_h i} \quad \forall h \in \{1, .., \|H^r\|\}$	preference expressed by request r to place its h-th VNF on DC i

In regards to the objective function, it is defined so as to reflect stakeholders' perspectives hierarchically: service provider perspective, first, and subscriber perspective, second. The service provider is interested in maximizing its profit which is given by the weighted sum of the served requests. Specifically, the weight associated with the accomplishment of a high priority request is bigger than the one associated with a low priority request. According to the subscribers' perspective, the placement of VNFs should be done to maximize their preferences. The secondary objective then consists in maximizing the overall preferences coming from all the requests.

In regards to the constraints that feasible solutions have to satisfy, the following are considered: (*i*) compatibility constraints; (*ii*) QoS constraints; (*iii*) service cost constraints; (*iv*) energy efficiency constraints; and (*v*) bandwidth constraints. Specifically, compatibility constraints assure that each of the VNFs composing a certain request is assigned to a node which is able to satisfy its requirements in terms of resource capacity and presence of a container. In addition, the order in which VNFs of a certain request are performed must respect the order specified in the request. QoS constraints refer to the end-to-end delay and, for each request, they have to guarantee that the delay of traffic flows traversing the service, once deployed over a set of nodes, is not greater than the maximum tolerated end-to-end delay. Service cost constraints guarantee that the cost paid by a request, given by the sum of the costs spent for the deployment of its VNFs on nodes, does not

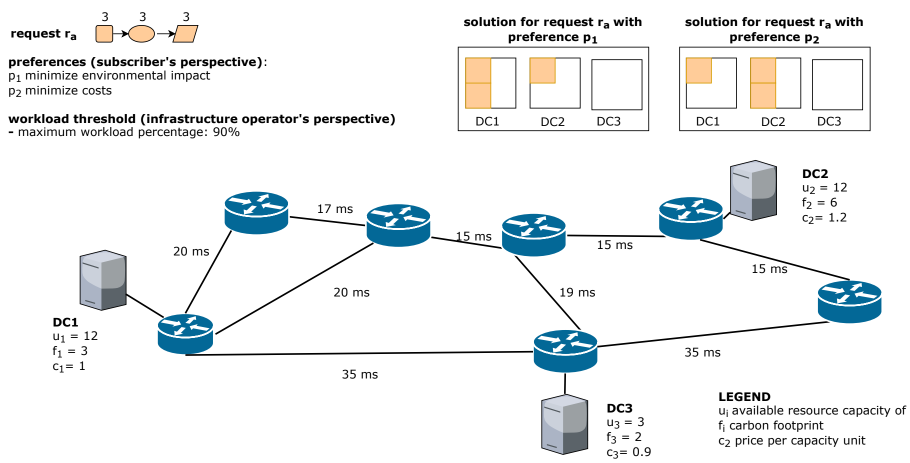


Figure 2: VNF Placement for service chaining problem: subscriber's and infrastructure provider's perspective

exceed the maximum cost request. A load distribution constraint guarantees that the workload assigned to a DC does not exceed a given threshold, therefore allowing the infrastructure provider to enforce a load distribution policy across managed sites. Finally, bandwidth constraints assure that, for each link in the network, the overall bandwidth consumed by all requests using that link does not exceed the bandwidth of the link.

Hereafter we provide two basic examples to clarify how stakeholder's perspectives are taken into account in the problem formulation. We consider three DCs (DC1, DC2, DC3) geographically distributed and interconnected via a WAN. The DCs offer a capacity of 20, 20 and 24 units, respectively.

Fig. 2 shows how subscribers' and infrastructure providers' perspectives are taken into account in the placement decision process for a request r_a made by three VNFs, each requiring 3 CPUs. The infrastructure provider may define a threshold proportional to available capacity u_i to avoid overload conditions (e.g., 90%). This implies that DC3 cannot be used. Subscribers may express preferences for cost and/or carbon footprint reduction. If only cost minimization is provided as preference, two VNFs will be placed in DC1 and one VNF in DC2. If carbon footprint is considered, one VNFs will be placed in DC1 and two in DC2.

Finally, Fig. 3 shows an example of the abstracted network view that the service provider has available for placement decisions. This view is built using monitoring data provided by infrastructure operators willing to hide

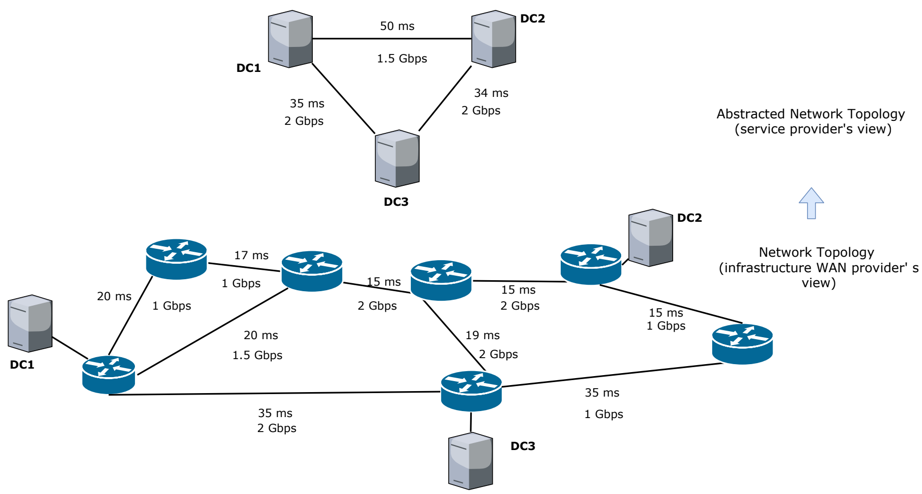


Figure 3: Example of abstracted network view: service provider's perspective

internal topology details. The example uses the bandwidth and latency definitions provided at the beginning of this section. An infrastructure operator can, of course, adopt different abstract latency and bandwidth definitions and is supposed to periodically provide the service provider with up-to-date monitoring data views.

4. Optimization Model

4.1. Computational complexity

This section analyzes the computational complexity of the problem addressed. We start showing that even the special case of the problem studied in which requests consist of only one VNF, each request is compatible with every DC, and bandwidth is disregarded is strongly NP-hard. Indeed, this fact is due to a reduction from a knapsack-like problem, as described in the following.

Theorem 1. *Consider the special case of optimally deploying VNFs to serve a set of requests, each of which consisting of only one VNF, in a multi-DC NFV infrastructure where bandwidth is assumed to be sufficient to manage all of the requests at the same time and the capacity of each DC is a one-dimensional parameter. Let P denote this problem. Then, P is strongly NP-hard.*

Proof 1. *Suppose that an instance of the* 0-1 Multiple knapsack problem (MKP) *is given. MKP is defined as follows. Given a set R of items with cardinality r, and a set D of knapsacks with cardinality d ($d \leq r$) with p_j equal to the profit of item j, w_j equal to the weight of item j, and c_i equal to the capacity of knapsack i, MKP consists in selecting d disjoint subsets of items so that the total profit of the selected items is a maximum, and each subset can be assigned to a different knapsack whose capacity is sufficient to contain the total weight of the items in the subset, computed as the sum of the weight of the items in the subset. More formally [49], MKP is:*

$$\max \sum_{i=1}^{d} \sum_{j=1}^{r} p_j x_{ij} \tag{1}$$

$$\sum_{j=1}^{r} w_j x_{ij} \leq c_i \qquad \forall i \in D \tag{2}$$

$$\sum_{i=1}^{d} x_{ij} \leq 1 \qquad \forall j \in R \tag{3}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in D, \forall j \in R \tag{4}$$

where x_{ij} is equal to one if item j is inserted in knapsack i and zero otherwise. As is usual in knapsack-related problems, it is assumed that (i) the coefficients w_j, p_j, and c_i are positive integers, (ii) $w_j \leq \max_{i \in D} c_i, \forall j \in R$, (iii) $c_i \geq \min_{j \in R} w_j \forall i \in D$, and (iv) $\sum_{j=1}^{r} w_j > c_i \forall i \in D$. Observe that non integer coefficients can be handled by multiplying them by a proper factor; all the items with a non positive profit or violating condition (ii) can be eliminated; all the knapsacks with a non positive capacity or violating condition (iii) can be eliminated. In addition, if there exists a knapsack with a capacity sufficient to contain all the items, i.e., a knapsack violating condition (iv), problem P admits the optimal trivial solution in which all the items are assigned to that knapsack. Finally, observe that if $d > r$ then the $(d - r)$ knapsacks with smallest capacity can be eliminated.

Now, suppose that an instance of MKP is given; we build an instance of P as follows. Each DC is associated with a knapsack, and each request is associated with an item. The priority level of a request is set to the profit of the corresponding item, the quantity of resource request j asks for is set to the weight of the corresponding item and the capacity of a DC is set to the capacity of the corresponding knapsack. From the optimal solution to P, we *can obtain the optimal solution to MKP.*

Observe that when the weight (and/or the profit) of an item depends on the knapsack in which it is inserted, MKP results in the *Generalized Assignment Problem (GAP)* that is NP-hard in the strong sense too [50]. In our case, the definition of a weight w_{ij} depending on the item j and on the knapsack i allows to manage the compatibility between requests and DCs. If request j is compatible with DC i, the weight reflects the quantity of resource required by the unique VNF in request; otherwise, when there is incompatibility between DC i and request j, the weight of item j is defined as greater than the capacity of the knapsack i thus interdicting the assignment of j to i. In our problem, a request j is compatible with DC i when (i) the latency of the path going from the origin of the request to DC i and from DC i to the destination of the request is not greater than the maximum end-to-end delay tolerated by j; (ii) the cost of assigning the VNF in the request to DC i is not smaller than the maximum cost request is willing to pay to get service, and (iii) DC j is able to satisfy the requirement of request i in terms of container virtualization technology. All these constraints can be managed by properly defining weight coefficients.

In summary, problem P is a special case of the problem addressed in this study and it is a MKP when each request can be accommodated by every DC or a GAP when incompatibility constraints between requests and DCs exist. Both MKP and GAP are NP-hard in the strong sense and, according to [51], this facts excludes the existence of a fully polynomial-time approximation scheme for them.

In the more general setting, the problem of optimally deploying VNFs on DCs to serve a set of requests in a multi-DC NFV infrastructure, consists in selecting the subsets of requests providing the maximum profit that can be accomplished by network resources. For each accepted request, the problem asks to find a (constrained) path connecting the origin node of the request with its destination node while satisfying global capacity constraints at DC nodes. The problem is thus a Maximum Integral k-multicommodity flow problem which is shown [52] to be APX-complete when the underlying network is a tree and paths are not constrained.

These results motivate us to formulate the problem as an ILP.

4.2. The mathematical model

This section describes the mathematical model used to formulate the problem of optimally deploying VNFs on DCs to serve a set of requests in a

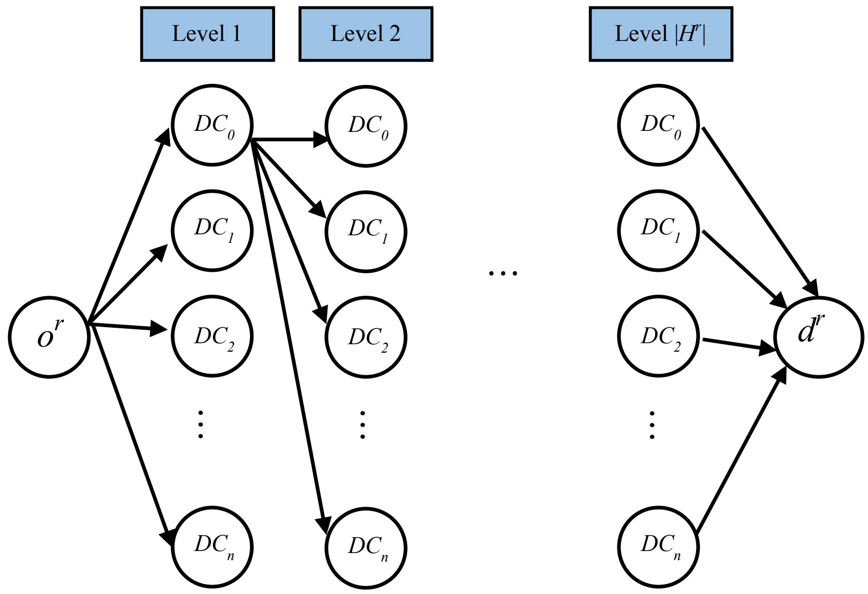


Figure 4: Auxiliary multi-layer graph for a request r

multi-DC NFV infrastructure. As already done in [53] for the optimal VNFs selection problem for service chaining, we make use of an auxiliary graph $G^r = (N^r, A^r)$ for each request $r \in R$. Specifically, G^r is a layered graph with a level for each of the $|H^r|$ VNFs appearing in request r (numbered from 1 to $|H^r|$), in addition to two extra levels: the first level, namely level 0, containing the origin node o^r of the request and the last level, namely level $|H^r|+1$ containing the destination node d^r of the request. Each intermediate level $h \in H^r$ is composed by all the DCs. The arc set A^r is organized in three groups: (i) arcs connecting the source node in level 0 to each node in level 1; (ii) arcs connecting each DC in level $|H^r|$ to the destination node in last level; and (iii) arcs linking each DC i in level h with each DC j in level $h+1$ for each intermediate level $(h \in \{1, .., |H^r| - 1\})$. In this latter group, arc from DC i to DC j is characterized by the propagation latency l_{ij} and the bandwidth b_{ij}. DCs and arcs between any couple of DCs are shared among requests. A graphical representation of the auxiliary graph G^r is given in Figure 4.

Servicing request r corresponds to determine a path in G^r from o^r to d^r.

By construction, such a path visits exactly a node in each level. Specifically, the node visited in intermediate level h corresponds to the DC where the h-VNF of the request r, namely V_h^r, is deployed. The layered structure of the graph thus ensures that the order of VNFs specified in the request is preserved.

This work does not lose generality if, for the sake of clarity, it is assumed that all the requests are characterized by VNF chains of the same length $|H| = \max_r\{|H^r|\}$. In that case, for each request, last level corresponds to level $|H| + 1$ and it contains the destination of each request. Indeed, every time a request is characterized by a chain shorter than $|H|$, then all the nodes in level $|H^r|$ are connected directly to the destination node in $|H| + 1$ and all the levels comprised between $|H^r| + 1$ and $|H|$ are consequently neglected (hop across levels). In other words, when $|H^r| < |H|$, with a short abuse of notation, level $|H^r| + 1$ identifies the last level, i.e., $|H| + 1$.

In order to model the problem, two groups of decision variables are considered corresponding respectively to path design variables (allocation of VNFs to DCs), and requests' satisfaction (maximal covering). Specifically, they are defined as follows:

$$x^r_{ihjh+1} = \begin{cases} 1 & \text{if the arc linking node } i \text{ in level } h \text{ and node } j \text{ in level} \\ & \quad (h+1) \text{ belongs to the path relative to } r \in R \\ 0 & \text{otherwise} \end{cases}$$

$$\forall\, r \in R,\ i \in D \cup \{o^r\},\ j \in D \cup \{d^r\},\ h \in H^r \cup \{0, |H| + 1\}$$

$$z^r = \begin{cases} 1 & \text{if request } r \text{ is served} \\ 0 & \text{otherwise} \end{cases} \qquad r \in R.$$

Besides the notation introduced in Tables 1, 2, 3 and 4, the model makes use of the additional parameters defined in Table 5. All the sets and the parameters contained in these tables define the input of the optimization model.

Table 5: Additional model parameters

w_t	weight associated with a request of priority level t
W	weight used in the hierarchical objective function
L_h	nodes belonging to level h in the auxiliary graph
I	incompatibilities set

By using the above-defined variables and notation, the problem can be stated formally, as follows.

$$\max \quad W \sum_{t \in T} \sum_{r \in R_t} w_t \cdot z^r + \sum_{r \in R} \sum_{h=0}^{|H^r|-1} \sum_{i \in L_h} \sum_{j \in L_{h+1}} p^r_{V^r_{h+1} i} \cdot x_{jhih+1} \tag{5}$$

$$\sum_{j \in L_1} x^r_{o^r 0 j 1} = z^r, \quad \forall r \in R \tag{6}$$

$$\sum_{j \in L_{|H^r|}} x^r_{j|H^r|d^r|H|+1} = z^r, \quad \forall r \in R \tag{7}$$

$$\sum_{j \in L_{h-1}} x^r_{jh-1ih} - \sum_{j \in L_{h+1}} x^r_{ihjh+1} = 0, \quad \forall r \in R, \forall h \in \{1, \ldots, |H^r|\}, \forall i \in L_h \tag{8}$$

$$\sum_{r \in R} \sum_{h=0}^{|H^r|-1} \sum_{j \in L_h} u^n_{V^r_{h+1}} x^r_{jhih+1} \leq \overline{p}^n_i \cdot u^n_i, \quad \forall i \in D, \forall n \in N \tag{9}$$

$$\sum_{h=0}^{|H^r|-1} \sum_{i \in L_h} \sum_{j \in L_{h+1}} c_j \sum_{n \in N} u^n_{V^r_{h+1}} x^r_{ihjh+1} \leq c^r, \quad \forall r \in R \tag{10}$$

$$\sum_{h=0}^{|H^r|} \sum_{i \in L_h} \sum_{j \in L_{h+1}} l_{ij} x^r_{ihjh+1} \leq l^r, \quad \forall r \in R \tag{11}$$

$$\sum_{r \in R} \sum_{h=0}^{|H^r|} b^r_{hh+1} x^r_{ihjh+1} \leq b_{ij}, \quad \forall (i,j) \in E \tag{12}$$

$$\sum_{j \in L_{h-1}} x^r_{jh-1ih} = 0, \quad \forall r \in R, \forall (V^r_h, i) \in I \tag{13}$$

$$x^r_{ihjh+1} \in \{0,1\}, \quad \forall r \in R, \forall i \in D \cup \{o^r\}, \forall j \in D \cup \{d^r\}, \forall h \in \{0, \ldots, |H^r|\} \cup \{0, |H|+1\} \tag{14}$$

$$z^r \in \{0,1\}, \quad \forall r \in R \tag{15}$$

The hierarchical objective function is defined in (5) and it consists in the maximization of the weighted sum of the two criteria introduced in Section 3, namely provider utility and user utility. Weight W is used to give more relevance to the first criterion. In addition, weights w_t are set so as to give more privileges to requests with higher priority level. The second criterion accounts for preferences satisfaction: $p^r_{V^r_h i}$ expresses the preference grade of request r for placing the h-th VNF on DC i. Constraints (6), (7) and (8) are, for each $r \in R$, the flow conservation constraints defining the path from o^r to d^r. Specifically, for each r, constraint (6) assures that exactly one unit of flow leaves the source node o^r when request r is accepted ($z^r = 1$); in that case, since by definition, the path design decision variables are 0-1 variables, exactly one of the arcs outgoing from o^r will be selected. The ending node of such an arc belongs to level L_1 and it identifies the DC that hosts the first virtual function in request r. Conversely, when request r is not served ($z^r = 0$), no unit of flow will leave the source node. Symmetrically, for each accepted request $r \in R$, exactly one unit of flow enters the destination node d^r as imposed by constraint (7). Constraints (8) assure that for each request $r \in R$, for each intermediate level h and for each node $i \in L_h$, the quantity of flow entering node i is exactly the same as the one leaving node i. Constraints (9) are the workload constraints and they are defined for each DC and for each resource n. Specifically, they guarantee that the actual workload of each DC, which is given by the the sum of resources of a given typology required to execute VNFs deployed on it, must not exceed a given threshold which is proportional to its capacity u^n_i. Percentage $\overline{p}^n_i$ is used to define the maximum ($\overline{p}^n_i \cdot u^n_i$) workload of DC i relevant to resource n, thus avoiding overhead. Constraints (10) guarantee that for each request, the total cost spent for all the resources and the VNFs of its chain does not exceed the cost c^r request r is willing to pay to get the service. Constraints (11), for each request r, assure that the end-to-end delay experienced to accomplish the service must not exceed the maximum tolerated latency l^r. Constraints (12), for each arc (i, j) in the abstract network, guarantee that the total bandwidth required to accomplish all of the service requests using (i, j) must not exceed the maximum available bandwidth b_{ij}. Observe that, the inner summation in constraints (12) considers all the copies of arc (i, j) between any two consecutive layers. Constraints (13) are the incompatibility constraints and they guarantee that if the h-VNF of request r is incompatible with DC i, then none of the arcs ingoing node i in level h can be used by request r or equivalently, the corresponding path design variable is set to 0. Finally constraints (14) and (15) define variable domain.

5. Pre-Processing phase

The system designed to solve the VNF Placement problem is equipped with a pre-processing phase which has a three-fold aim: (*i*) discard all those requests that cannot be accomplished by the system for infeasibility reasons, (*ii*) define the potential incompatibility between a specific virtual function of a given request and a DC, and (*iii*) define user preferences that are then used in the secondary objective function of the optimization model. In the following three sections the three features of the pre-processing phase are described in detail.

5.1. Infeasibility check

In regards to infeasibility check, three conditions are controlled concerning respectively latency, bandwidth and cost. Specifically,

1. *Latency check*: for each request $r \in R$, the maximum tolerated end-to-end delay l^r is compared with the minimum possible achievable propagation latency from o^r to d^r, i.e., $l_{o^r d^r}$. If, for a given r,

$$l^r < l_{o^r d^r}, \tag{16}$$

   then request r is rejected.
2. *Bandwidth check*: for each request $r \in R$, the maximum bandwidth consumption of r is compared with the maximum possible achievable bandwidth for all the paths connecting o^r to d^r in the abstract network, namely P^r. If, for a given r,

$$\max_{p \in P^r}\{\min_{(i,j) \in p} b_{ij}\} < \max_{h=1,\dots,|H^r|-1} b^r_{hh+1}, \tag{17}$$

   then request r is rejected.
3. *Cost check*: for each request $r \in R$, the maximum cost r is willing to pay for the service, i.e., c^r is compared with the minimum cost achievable on the network which occurs when the total capacity required by the request, namely u^r is provided by the DC with minimum cost per capacity unit. If, for a given r,

$$u^r \cdot \min_{i \in D}\{c_i\} > c^r \tag{18}$$

   where

$$u^r = \sum_{h=1}^{|H^r|} \sum_{n \in N} u^n_{V^r_h} \tag{19}$$

then request r is rejected. In equation (19), we assume that the total quantity of resources required by a request is given by the sum of the quantities required by all the resources and all the VNFs in the chain. We also assume that the cost of a request depends on the aggregated use of resources. However, in order to define the cost of a request, other linear combinations of the parameters involved can be managed as well; as an example, in [54], the price exposed by a DC depends on the DC itself and on the resource considered.

These three controls must all be satisfied to allow the request be given in input to the optimization solver; this does not guarantee that it will definitely be served, but only that it is compatible with the system supply.

5.2. Incompatibility definition

In regards to the definition of incompatibility between a specific VNF of a request and a DC, two conditions are controlled concerning respectively capacity and setup time. Specifically,

1. *Capacity check*: if a resource $n \in N$ exists for which the corresponding resource capacity of the i_{th} DC, namely u_i^n is smaller than the capacity required by the h-th virtual function of request r in terms of resource n, namely $u^n_{V_h^r}$, then the assignment between V_h^r and i is forbidden and the couple (V_h^r, i) is inserted in the incompatibility set I, i.e.,

$$\text{if there exists } n \text{ s.t. } u_i^n < u^n_{V_h^r} \text{ then } (V_h^r, i) \in I. \tag{20}$$

2. *Instantiation time check*: the availability of a container-based virtualization technology at the i_{th} DC, namely the binary parameter s_i, is compared with the instantiation time requirement of request r, namely the binary parameter s^r. Specifically, request r can be served by DC i when $s_i \geq s^r$ that means that if request r needs to be deployed on a container to minimize the instantiation time ($s^r = 1$), then DC i has to provide a container ($s_i = 1$). If the condition does not hold, then none of the virtual functions of r can be deployed on DC i, i.e.,

$$\text{if } s_i < s^r \text{ then } (V_h^r, i) \in I \quad \forall h \in H^r. \tag{21}$$

This preprocessing phase can be easily extended by managing further conditions, such as commercial alliances and conflicts of interest (e.g., DCs managed by competitors are banned).

5.3. Definition of user preferences

User preferences are built upon a ranking algorithm that provides, for each request r, an ordered preference list of DCs to be used in the placement. Specifically, a set M of preference criteria are considered to define the global vote q_i^r request r attributes to DC i, i.e.,

$$q_i^r = \sum_{m \in M} w_m^r q_{mi} \quad \text{with} \sum_{m \in M} w_m^r = 1 \quad \forall r \in R, \forall i \in D, \tag{22}$$

where weight w_m^r reflects the importance request r gives to criterion m and q_{mi} expresses the vote to DC i with respect to criterion m. Indeed, users assign weights to the criteria according to their business and/or private goals. Votes q_{mi} assume values in the range [0,1], thus also the global vote q_i^r is in the range [0,1]. Then, for each request r, DCs are ranked according to decreasing values of q_i^r.

Starting from q_i^r, user preference grades $p^r_{V_h^r i}$ can be assigned according to different policies and range of values which contribute to design a flexible tool capable of copying with general preference schemes. In this study, we consider two preference criteria: i) cost minimization (C) and ii) environmental impact minimization (F) (e.g., carbon dioxide emissions).

In regards to costs, we assume that each request competes with the others to place its constituent VNFs in the DCs that are more economically convenient. For each DC i, the vote with respect to cost minimization ($m = C$) is given by

$$q_{Ci} = \frac{c_{min}}{c_i} \tag{23}$$

where

$$c_{min} = \min_{i \in D} \{c_i\} \tag{24}$$

is the minimum service price exposed over the whole set of DCs.

In regards to environmental impact, we assume that the users who have expressed their interest in reducing the environmental impact ($f^r = 1$), favor DCs characterized by the lowest possible CUE. Thus, for each DC i, the vote with respect to environmental impact minimization ($m = F$) is given by:

$$q_{Fi} = \frac{f_{min}}{f_i} \tag{25}$$

where

$$f_{min} = \min_{i \in D} \{f_i\} \tag{26}$$

is the minimum CUE value over the whole set of DCs.

Different strategies can be adopted to exploit the DC ranking based on the above described global vote calculation procedure to assign appropriate values to preferences in the user utility part of the hierarchical objective function defined in (5) in Section 4.2. In practice, strategies can differ on what is considered full or partial satisfaction, taking into account that, due to resource capacity constraints, not all VNFs can be placed on the respective first ranked DCs. In this work we consider a strategy considering that only the assignment to the first and second positioned DCs can be respectively considered as full and partial satisfaction, while the remaining options are considered dissatisfaction. This strategy (called 2LevelSat strategy) is implemented as follows. The global vote formula in (22) is used for creating a rank of DCs for each VNF in a request, then the first positioned DC is assigned a preference value equal to 1, the second positioned DC a value equal to 0.5 and 0 otherwise. We also formulate an alternative strategy that use more granular preferences respect to the previous strategy. More specifically, preferences assume exactly the same value of the global vote, in the range [0,1], as defined in (22). This means that VNFs assigned to DCs that are not in the first two positions howsoever contribute to the global satisfaction level and are consequently considered as partially satisfied. The 2LevSat strategy adopts a more restrictive definition of partial satisfaction with respect to GradSat. In Section 6.4 (Performance Evaluation) we evaluate how far preferences are satisfied by these two strategies.

6. Performance Evaluation

In this section we describe the activities carried out to evaluate the proposed VNF placement solution. First, we briefly describe the experimental settings and the adopted metrics, then we describe the tests and discuss obtained results.

To evaluate the proposed solution, we have developed a testing tool based on CPLEX and MATLAB. The preprocessing steps are performed by Matlab scripts while the VNF Placement problem is solved using CPLEX 12.8.

6.1. Benchmark instances

We considered three different network topologies, namely a hypothetical German backbone network (17 nodes), a Pan-European network (28 nodes) and a US Network (14 nodes). Topological parameters have been gathered from the literature [55].

We adopted the betweenness centrality metric to select the nodes that can host VNFs (the so called DC nodes). Betweenness centrality of a node is calculated as the number of shortest all-to-all paths that pass through that node and is thus a good indicator of the importance of a node in the network [56]. The sum of the resources available in all DC nodes is called overall capacity and assumed to be equal to 100 units.

We generated request data sets by mirroring realistic traffic using the traffic distribution used in [56] and elaborated from the global IP Traffic Forecast by Cisco [57]. We considered three types of service requests, similarly to the settings in [30, 56, 58, 59]. Each service request type contains a sequence of VNFs and requires a specific amount of bandwidth and a maximum end-to-end latency (see Table 6). Within each service request set, service request types are distributed according to percentages derived from realistic traffic distribution [57].

Table 6: Service chains that have been considered to compose each request set [58]

Service	**Chain**	**Latency**	**Bandwidth**	percentage
Web Service (WS)	NAT-FW-TM-WOC-IDPS	500 ms	100 kbit/s	18.2 %
VoIP	NAT-FW-TM-FW-NAT	100 ms	64 kbit/s	11.8 %
Video Streaming (VC)	NAT-FW-TM-VOC-IDPS	80 ms	4 Mbit/s	70.0 %

At each iteration, a set of requests is generated that stresses the network with a given overall request load, defined as the ratio between the total amount of resources required by the requests in the set and the overall capacity offered by the multi-DC network. For instance, given a target request load of 80%, the amount of required resources by all requests in the set is calculated as a percentage of the actual overall capacity (i.e., 80 over 100 units), and is equally distributed among all requests in the request set. We consider two priority levels, premium and best effort, where premium's priority level is higher than best effort's one. Since each type of chain contains 5 VNFs and we assume that all VNFs require the same amount of resources (1 unit), the target request load is thus achieved by varying the number of requests in the set.

Each request of the set is generated by varying its characteristics at each iteration. Source and destination nodes are randomly selected among DC nodes. Configuration of further attributes (e.g., priority level, service cost, setup time, and carbon footprint preference) is described hereafter for each test case. Finally, the weights of the hierarchical objective function have been defined in order to give more relevance to the acceptance rate criterion (weight W=1000) with respect to preference satisfaction and to preferably accept premium requests than best effort ones (w_p=3 and w_b=1).

6.2. Evaluation metrics

We define a test case for each of the following metrics:

- *Acceptance Rate*: the ratio between the number of accepted requests (i.e., requests that have been deployed in the optimal solution), also differentiated per priority level, and the total number of requests in a request set. The request set is generated so that all requests pass the feasibility check.

- *Preference satisfaction*: it provides a measure of how much the preferences expressed in a request have been satisfied.

- *Execution time*: time required by the solver to process a set of requests and return the optimal solution.

- *DC utilization factor*: percentage of used resources against maximum resource capacity for each DC.

- *Request Load spread across DCs*: percentage of the overall resource demand of a request set assigned to each DC.

- *Request latency vs maximum tolerated latency*: it is the ratio between the computed latency of an accepted request vs its corresponding maximum tolerated latency.

6.3. Acceptance Rate

This test case has the goal of assessing to which extent the service provider profit is maximized in terms of acceptance rate. Tests have been run over the three network topologies where 60% of nodes have been modeled as DC nodes. We consider three different combinations of premium (P) and best effort (BE) priority levels in the request set, as follows:

1. P=70% and BE=30%;
2. P=50% and BE=50%;

3. P=30% and BE=70%.

We vary the request load from 70% to 120% with an increment step of 10% in order to increasingly stress the network.

We run 50 test iterations for each combination of priority level distribution and request load. In each iteration we slightly vary some parameters characterizing the request set and the substrate. As regards the request set, source and destination nodes of the service request are randomly mapped to the subset of compute nodes in the network and the maximum cost allowed for each request is calculated by multiplying the amount of resources required by the request with a maximum cost for unit capacity that randomly varies in the range [0.9,1.1]. A 25% of requests (randomly selected) requires a fast instantiation time (i.e., s^r set to 1). As regards preference criteria, 75% of requests in each set has cost reduction as unique preference criterion and the remaining 25% of requests has both cost and carbon footprint preference criteria (see Section 5.3). As regards topology settings, the price offered by each node per capacity unit randomly varies in the range [0.7,1.2], while the CUE randomly varies in a discretized range [1,7] and 50% randomly selected nodes offer a container virtualization technology, i.e., they can satisfy requests with s^r set to 1.

Fig. 5 shows the average percentage of accepted requests for each combination of P and BE requests, without differentiating results per classes, for the Pan-European topology. For request loads lower than 100%, almost all requests are accepted, with negligible difference with respect to the three combinations of P and BE requests. When the request load is more challenging (i.e., greater than 100%), the overall acceptance rate slightly decreases, but such decrease is mainly caused by the reduction in the number of accepted BE requests in favor of premium ones, as more clearly shown in Fig. 6. This was expected, since in our tests premium and best effort requests require the same amount of resources and when resources offered by the substrate are getting scarce for high request loads, preference is given to premium requests. Tests conducted with the German and US network topologies show analogous trends thus confirming the expected behavior of the algorithm.

6.4. Preference satisfaction

This test case aims at evaluating how far preferences are satisfied in the placement decision, considering both 2LevelSat and GradSat preference assignment strategies.

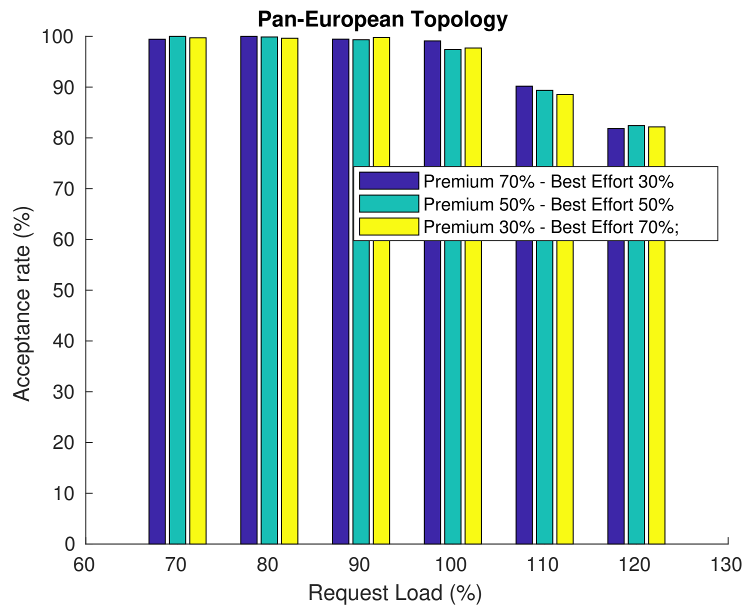


Figure 5: Overall Acceptance Rate vs Request load for different combinations of premium and best effort requests - Pan-European topology

Based on the global votes calculated in the preprocessing phase (Section 5.3), an ordered list of DCs is created for each VNF in the request set, expressing a descending order of preference for placement.

In order to measure how far preferences are satisfied, we count how many preferences expressed in the request set have been satisfied. More specifically, we count how many VNFs of the request set have been placed in the first-ranked DCs and how many VNFs in the second-ranked DCs.

Tests have been carried out on the German and Pan-European topologies with the same settings of the substrate network as in the previous test. As regards the request set, we considered three different combinations of premium (P) and best effort (BE) priority levels as in the previous test (i.e., P=70% and BE=30%, P=50% and BE=50%, P=30% and BE=70%). We considered increasing load values (70%, 80%, 90%, 100%), maximum cost in the range [0.9,1.1] and two different preference settings, described hereafter.

First, we evaluate results obtained with the adoption of 2LevelSat strategy. Table 7 shows the results obtained with the first preference settings (called Settings A) where 25% of requests equally take into account cost and environmental impact as guiding criteria ($w_C^r = w_F^r = 0.5$ in equa-

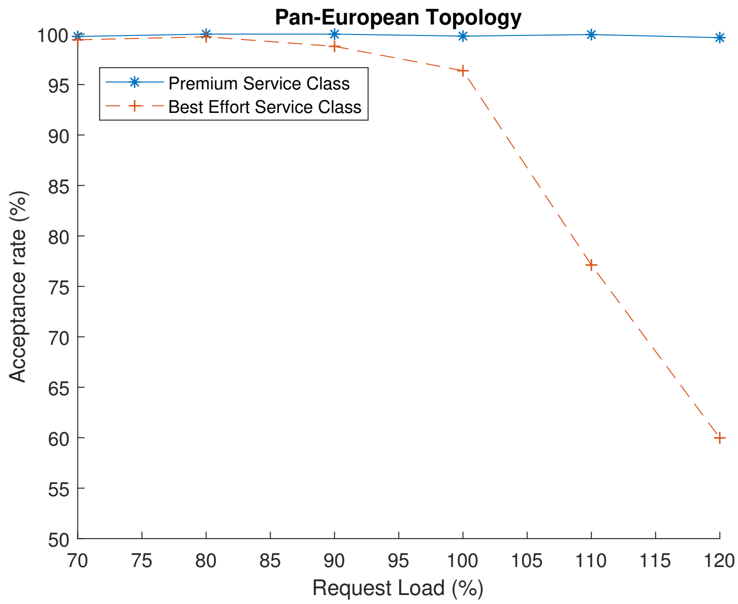


Figure 6: Acceptance Rate per priority level vs Request load - Pan-European topology

tion (22)), while 75% of requests take into account only the cost criterion ($w_C^r = 1, w_F^r = 0$).

As the request load (and thus the overall number of VNFs to be placed) increases, the percentage of VNFs placed in DCs ranked in the 1st and 2nd position clearly decreases. This is due to the fact that more VNFs compete to be placed in the preferred DCs and this effect is exacerbated by the fact that with these preference settings a large percentage (75%) of requests compete to be placed in the most economically convenient nodes. As the percentage of DC nodes increases, preference satisfaction decreases since, as explained before, the overall resource capacity is fixed to 100 and the resource quota assigned to each node diminishes as the number of DCs increases. Therefore, as the number of DC nodes increases, first and second positioned DCs can accommodate fewer requests. However, Table 7 shows that, even with high request loads and number of DCs, the percentage of VNFs placed in first or second position is quite high.

Table 8 shows the results obtained with the second preference settings (called Settings B) where 50% of requests consider only the cost criterion ($w_C^r = 1, w_F^r = 0$) and the remaining 50% considers only the environmental impact ($w^r_C = 0, w^r_F = 1$). As shown in Table 8, preference satisfaction improves with respect to the previous configuration. This is due to the fact that we divided the request set in two disjoint subsets (one targeting cost effective DCs, and the other one targeting DCs minimizing the environmental impact) and thus the competition on the resource substrate decreases.

Table 7: Preference satisfaction with 2LevelSat preference assignment strategy - percentage of VNFs placed in 1st and 2nd ranked DCs for sets with 75% requests with $w^r_C = 1$ and $w^r_F = 0$ and 25% requests with $w^r_C = w^r_F = 0.5$ (Settings A)

	German Topology				**Pan-European Topology**			
	11 DC nodes (∼60%)		**14 DC nodes (∼80%)**		**16 DC nodes (∼60%)**		**21 DC nodes (∼80%)**	
Request Load	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked
70%	27.0	19.7	22.7	17.3	19.3	17	18.3	15
80%	25.3	18.3	21.0	16.0	18.3	15.0	15.7	13.3
90%	22.7	16.7	19.7	14	17.3	13.0	14.0	11.7
100%	21.3	16.3	17.7	14.0	15.7	13.0	12.7	11.0

Table 8: Preference satisfaction with 2LevelSat preference assignment strategy - percentage of VNFs placed in 1st and 2nd ranked DCs, 50% requests with $w^r_C = 1$ and $w^r_F = 0$ and 50% requests with $w^r_C = 0$ and $w^r_F = 1$ (Settings B)

	German Topology				**Pan-European Topology**			
	8 DC nodes (50%)		**14 DC nodes (80%)**		**17 DC nodes (60%)**		**22 DC nodes (80%)**	
Request Load	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked
70%	29.7	22.3	24.3	20.3	22.3	18.3	17.7	16.3
80%	27.0	21.3	22.3	18.7	20.7	17.0	16.7	14.3
90%	24.0	19.0	21.3	16.0	18.0	15	14.7	13.3
100%	23.0	18.0	18.7	16.0	17.0	13.7	14.0	12.3

The remaining part of this section is dedicated to show the results obtained with the alternative GradSat strategy to assign preferences. We repeated the same tests (i.e., reusing the same request sets, preference and topology configurations) and report the results in Tables 9 and 10.

The resulting behaviour is quite similar to the one obtained with the previous preference assignment strategy, i.e., the percentage of VNFs placed

Table 9: Preference satisfaction with GradSat preference assignment strategy - percentage of VNFs placed in 1st and 2nd ranked DCs for sets with 75% requests with $w^r_C = 1$ and $w^r_F = 0$ and 25% requests with $w^r_C = w^r_F = 0.5$ (Settings A)

	German Topology				**Pan-European Topology**			
	11 DC nodes (∼60%)		**14 DC nodes (∼80%)**		**16 DC nodes (∼60%)**		**21 DC nodes (∼80%)**	
Request Load	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked
70%	24.3	20.0	20.0	17.0	17.3	14.7	14.7	14.0
80%	22.0	17.3	18.0	15.0	16.7	14.0	13.3	11.7
90%	20.0	15.0	17.0	12.3	14.3	12.0	12.3	10.3
100%	19.0	15.0	15.3	13.0	13.7	11.3	11.0	9.7

Table 10: Preference satisfaction with GradSat preference assignment strategy - percentage of VNFs placed in 1st and 2nd ranked DCs for sets with 50% requests with $w^r_C = 1$ and $w^r_F = 0$ and 50% requests with $w^r_C = 0$ and $w^r_F = 1$ (Settings B)

	German Topology				**Pan-European Topology**			
	8 DC nodes (50%)		**14 DC nodes (80%)**		**17 DC nodes (60%)**		**22 DC nodes (80%)**	
Request Load	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked	1st ranked	2nd ranked
70%	26.7	23.3	21.0	20.3	19.3	17.7	15.0	15.7
80%	24.0	21.3	19.0	18.3	17.3	15.7	13.7	13.7
90%	20.0	19.7	18.0	14.7	14.7	13.7	12.3	12.0
100%	18.7	17.7	15.3	14.3	14.0	12.7	11.7	10.3

in the DCs ranked in the 1st and 2nd position decreases with the request load both in Table 9 and Table 10. Also in this case preference satisfaction in Table 10 is higher than in Table 9.

Comparing these two strategies, it is evident that the first strategy (2LevelSat) succeeds in allocating a greater percentage of VNFs in the first and second ranked DCs. In addition, different preference assignment strategies may also impact the computational time required to solve the optimization problem. Although the evaluation on computational time is discussed in the following section, it is worth highlighting here how the first strategy leads to generally shorter execution time than the second one does, with an average computational time over all iterations of 1636 ms versus 4821 ms, respectively. However, the comparison of the user utility objective value achieved shown in Fig. 7 shows that the GradSat strategy obtains higher objective

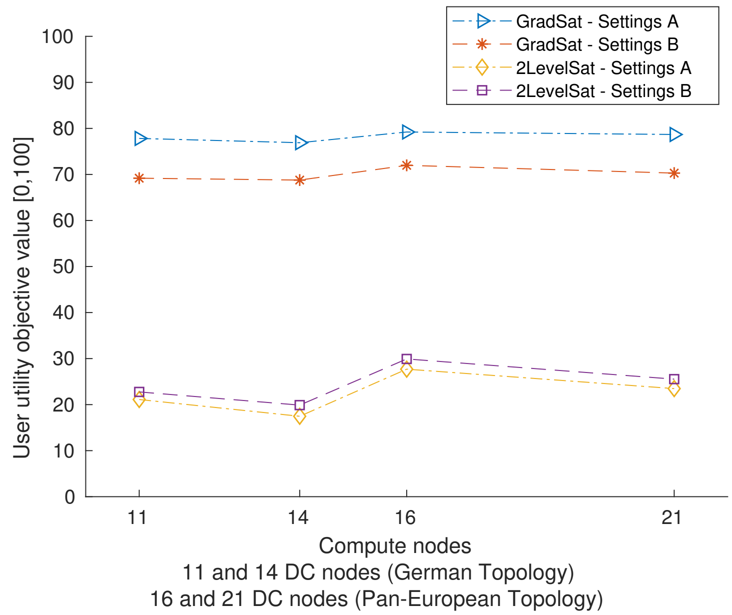


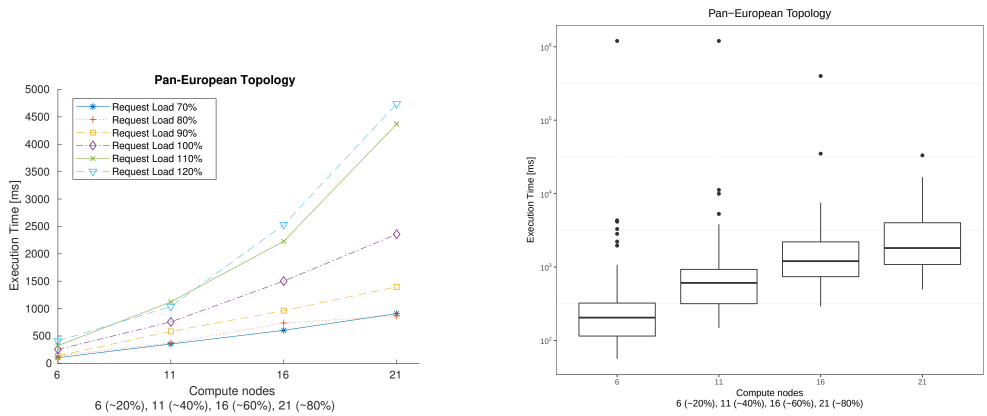
Figure 7: Comparison of Preference assignment strategies in terms of user utility objective value (Request Load=100%)

function values than 2LevelSat's ones in both types of tests (i.e., Settings A and B). Fig.7 shows the average user utility objective value obtained with Request Load equal to 100%.

6.5. Execution Time

This test aims at evaluating the computational time required by the optimization algorithm to solve the VNF placement problem. Tests have been performed on all topologies by varying the percentage of nodes selected as DCs (approximately 20%, 40%, 60%, 80%). We have varied the number of requests in the input request set (from 14 to 24 requests) to correspondingly vary the overall request load (from 70% to 120% of the overall capacity with an increasing step of 10%).

For each combination of request load and DC nodes percentage, we run 50 iterations, varying some parameters' settings. Analogously to previous test settings, at each iteration we vary the following parameters: source and destination nodes of the service request are randomly mapped to the subset of compute nodes in the network and the maximum cost allowed for each request is determined by multiplying the amount of resource required by the



(a) Execution time for different request loads - trimmed average values

(b) Execution time distribution - logarithmic scale

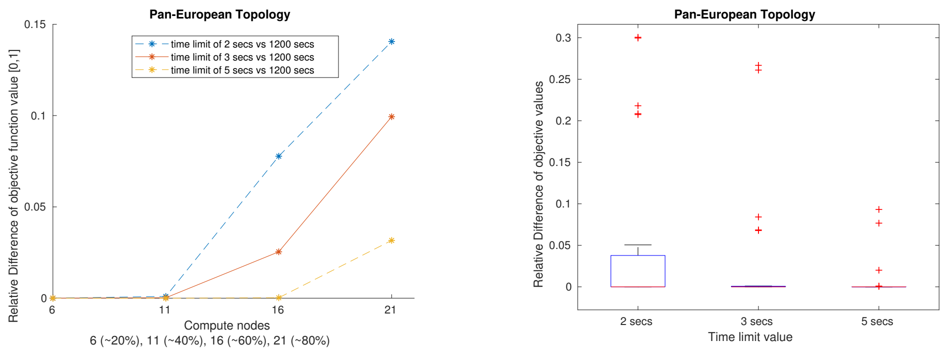
Figure 8: Execution time vs % of compute nodes - Pan-European topology

request with a maximum cost for unit capacity that is made randomly vary in the range [0.9,1.2]. A 25% percentage of requests (randomly selected) require a fast instantiation time (i.e., s^r set to 1). Moreover, 75% of requests in each set have cost containment as unique preference criterion and the remaining 25% of requests express both cost and carbon footprint preference criteria. The settings of the topology substrate is the same as in previous tests.

Figure 8 shows results obtained for the Pan-European topology. Graphic (a) on the left, shows how the ten percent trimmed value of execution time varies against the percentage of nodes considered as possible VNF locations for different request loads. Conversely, the distribution of the execution time, including also outliers, is shown in graphic (b) on the right. Analogous results have been obtained for the German and US topology, which are not reported here for the sake of conciseness, thus corroborating the validity of the approach. As expected, the results confirm that the time needed to find the optimal solution is influenced by the request load more than by the number of DCs. Specifically, we observe that the number of nodes has an almost linear impact on the computational time.

It is worth noticing that all tests have been run with a time limit for the solver set to 1200 seconds. As shown in Fig. 8b, in most cases the computational time stays well under this limit, while some outliers are highlighted with values well above 3 secs.

In order to evaluate the tradeoff between solution quality and efficiency



(a) Relative difference averaged over different request loads

(b) Relative difference distribution

Figure 9: Relative difference of the obtained objective function values with respect to the algorithm configured with a time limit of 1200 seconds

(computational time), we repeated the same test case by imposing a time limit of 5, 3 and 2 seconds, respectively. This further experiment is done only for the Pan-European topology which is the one with the highest computational times.

Figure 9a shows the relative difference of the obtained objective function values with respect to the algorithm configured with a time limit of 1200 seconds. Results shows that the relative difference (averaged over different request loads) is almost zero in most cases and increases with the number of DC nodes, but it is lower than 14%. Fig 9b shows the distribution of the objective function relative difference, highlighting outliers and median values (close to zero).

6.6. DC Utilization factor and request load spread across DCs

We analyzed results of the tests conducted on a 11 DC network in the Pan-European topology to evaluate how DC resources are used for request sets demanding 70% of overall resources (i.e. request load). Fig. 10 shows the percentage of resource usage for each DC. Considering the above mentioned request load and the fact that no upper thresholds on DC resource usage have been set, it is worth noticing that the average utilization factor of each DC is above 60%, thus demonstrating a good balance of resource usage across DCs.

Figure 11 shows how the request load is spread across DCs, highlighting a quite fair distribution of request loads across DCs.

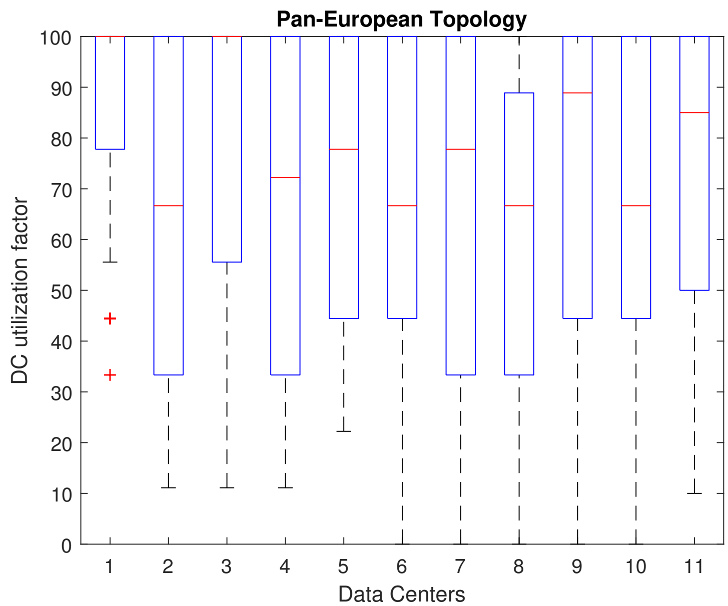


Figure 10: DC utilization factor - Pan-European topology, 11 DCs, 70% request load

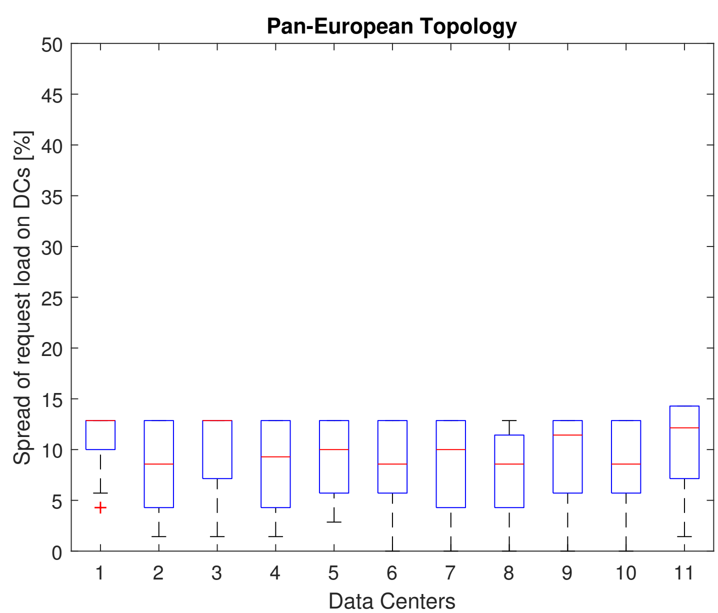


Figure 11: Spread of request load across DCs - Pan-European topology, 11 DCs, 70% request load

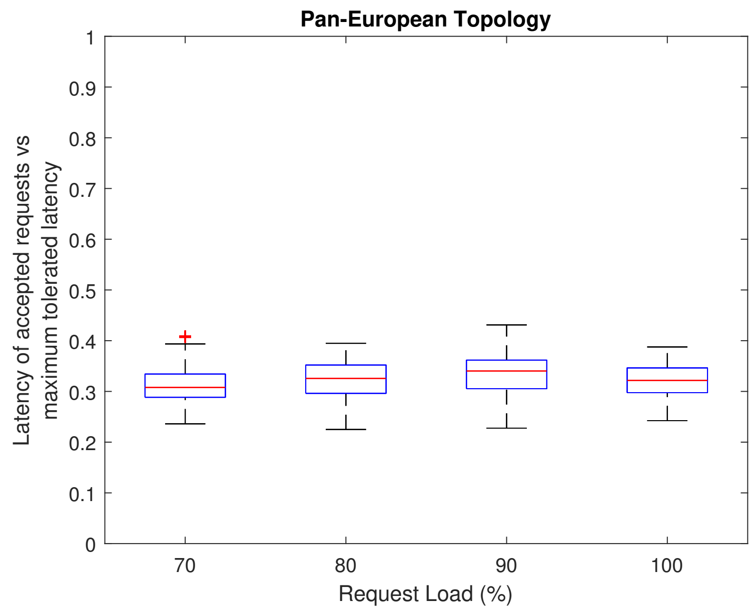


Figure 12: Latency variability across requests - Pan-European topology, 11 DCs

6.7. Request latency vs maximum tolerated latency

We also evaluated the ratio between the latency of accepted requests and the corresponding maximum tolerated latency at increasing request loads. Fig. 12 shows that for all request loads the latency of accepted requests is well below the maximum tolerated one.

6.8. Greedy heuristic

In this section we present the results obtained with a simple greedy heuristic which works as follows. Requests are considered according to their priority level so as to manage first the premium ones. Then, for each request, VNFs are considered in the order they appear in the chain and placed on the first DC in the ordered list of DCs if the assignment is feasible. DCs are ordered according to the price they offer so as to consider first the most convenient DCs. More specifically, for a given request, VNFs are considered one by one and the placement of a VNF on a DC is feasible only if (i) the DC and the VNF are compatible, (ii) the DC has enough capacity, (iii) the cost and the latency of the VNFs currently placed do not exceed their maximum allowed values, (iv) bandwidth on links is not exceeded. If the assignment is feasible, network resources are updated accordingly; otherwise, the next DC is

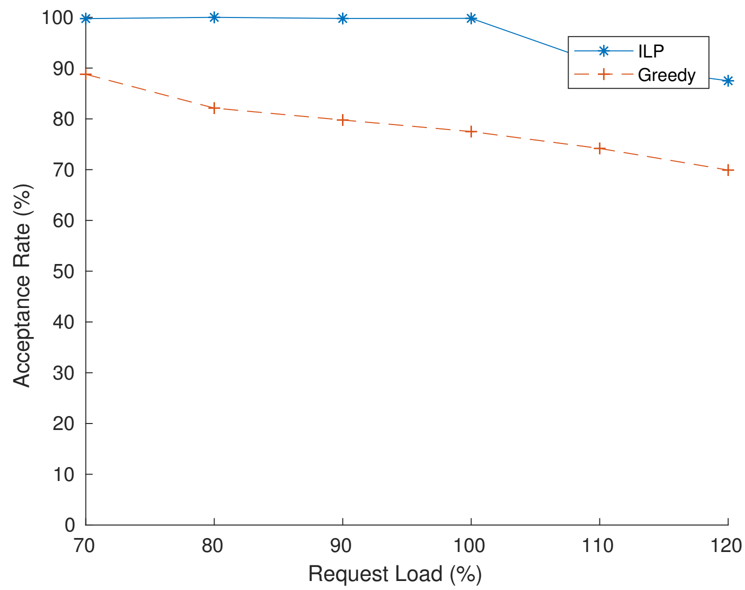


Figure 13: Overall acceptance rate - Pan-European topology, 11 DCs

considered until a DC is found or the list of DCs is exhausted. When, for the considered request, the assignment of a VNF to any DC is infeasible, the request is discarded and the resources potentially allocated to the previous VNFs in the chain are restored. We compare our ILP approach with the greedy heuristic in terms of acceptance rate at increasing request loads. Tests are performed on a Pan-European network topology of 11 nodes, requests are generated to vary the request load from 70% to 120% with each VNF in a chain requiring an amount of resources in the set $\{0.5, 1, 1.5, 2\}$. Our approach outperforms the greedy one in the acceptance rate of both classes of requests. Since the heuristic prioritizes placement of premium requests, the gap between the two approaches (ILP vs greedy) in the acceptance rate for premium requests is lower than for best effort ones. The execution time of the greedy heuristic is around 20 ms and remains almost stable, as opposed to the performance of the ILP approach, characterized by an execution time that increases with the request load as discussed in Section 6.5 and with an average value of 900 ms.

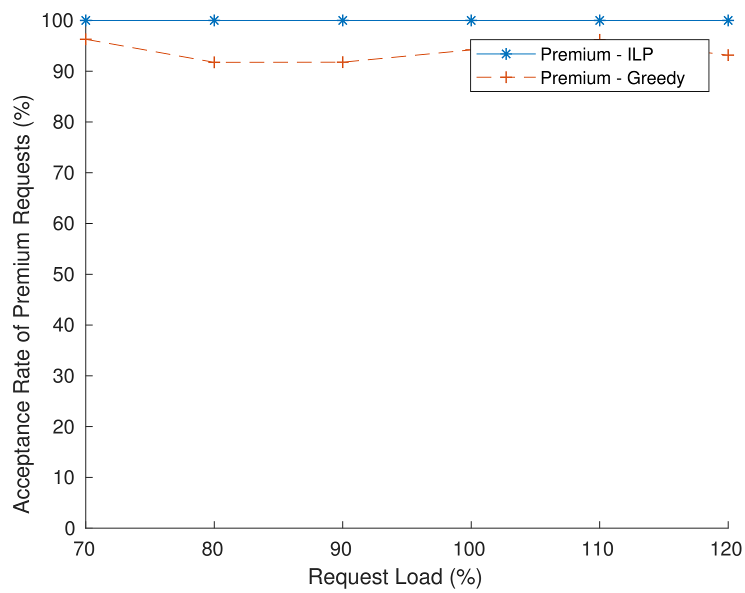


Figure 14: Acceptance rate for Premium requests - Pan-European topology, 11 DCs

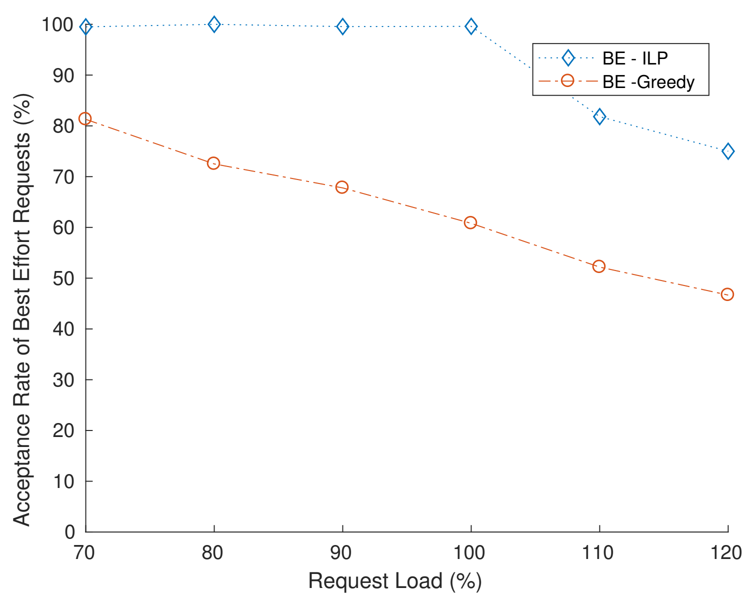


Figure 15: Acceptance rate for Best Effort requests - Pan-European topology, 11 DCs

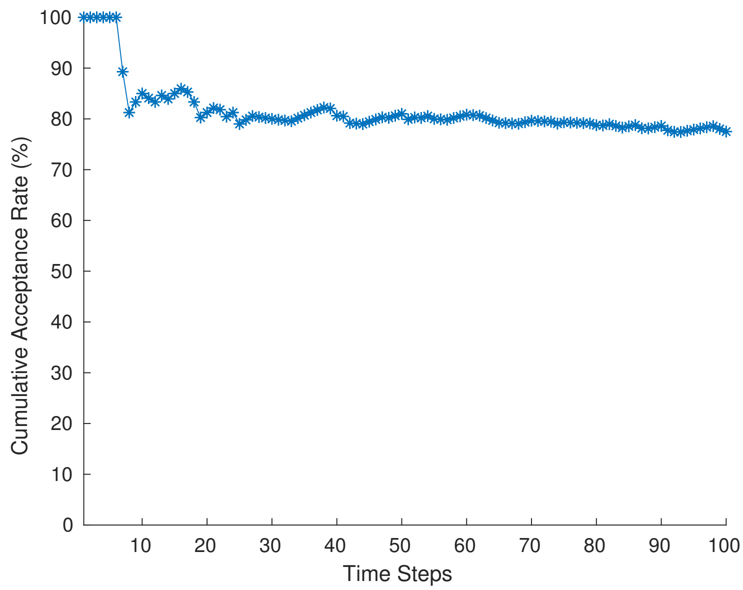


Figure 16: Cumulative acceptance rate - Pan-European topology, 11 DCs

6.9. Evaluation in an online placement scenario

In the online placement scenario, at each time step, the algorithm evaluates a batch of b incoming requests. Each VNF in a chain may request an amount of resource capacity in the set $\{0.5, 1, 1.5, 2\}$. For each request in the set, the service duration time is randomly set in the range of [1,10] timesteps. At the end of each time step, the status of the network is updated according to deployment choices and the amount of resources of terminating services to be released. The tests have been performed considering a Pan-European network topology with 11 DCs. The batch size b is set to 4 and requests are generated so that the overall request load of the batch is 20 units. Simulations are run for 100 time steps. The curve of cumulative acceptance rate in Fig. 16 shows a trend that, after a few iterations, becomes stable around 80%. Fig. 17 shows the execution time at each time step, corresponding to an average execution time of 61 ms. We consider this value acceptable in comparison with network service deployment time (e.g. 40-50 secs ca. [13]).

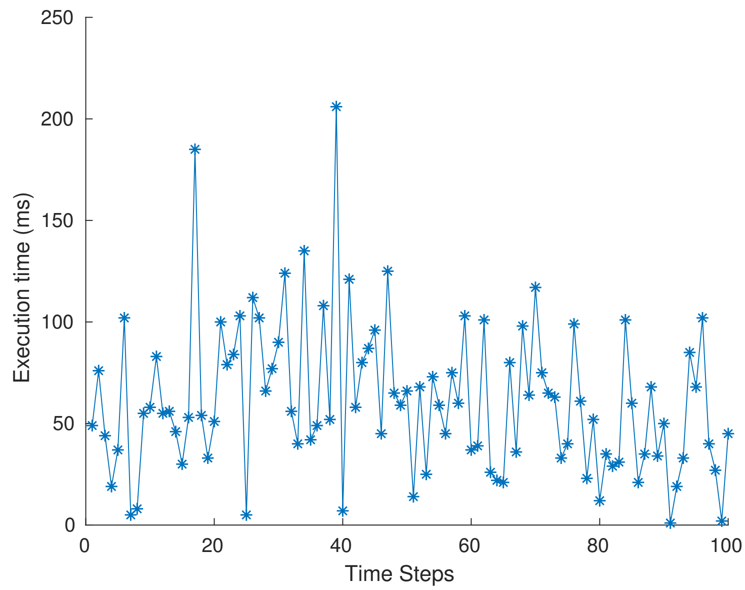


Figure 17: Execution time - Pan-European topology, 11 DCs

7. Conclusions

In this paper we presented a novel VNF placement algorithm for embedding a set of network service requests in a multi-DC physical substrate that accounts for multiple stakeholders' perspective. More specifically, we formulate an ILP-based optimization problem aiming at maximizing primarily service acceptance rate and, secondarily, satisfaction of subscribers preferences, while handling different priority levels and guaranteeing QoS objectives' fulfillment. The problem formulation leverages a layered auxiliary graph built considering the characteristics of the physical substrate topology. The layered structure of the graph ensures that the order of virtual functions specified in the request is preserved. Additional constraints (e.g., maximum allowed network latency on the whole path, minimum bandwidth) are taken into account during the graph construction phase. Our optimization algorithm solves the placement of a batch of requests assumed to be arrived within a given time window, however it allows to differentiate services that need a fast setup from standard ones.

Experimental evaluation has been carried out through extensive testings. We showed that the proposed algorithm is effective in maximizing the service acceptance rate for offline and online placement problems and we compared two different subscribers' preference assignment strategies. In regards to efficiency, we evaluated how the computational time varies with the request load and topology size, demonstrating that computational time limits of 2, 3 and 5 seconds lead to solutions that are very close to the optimal one. Test results also show that the proposed approach fairly distributes the overall request load across available DCs. Finally, we compared our ILP-based approach with a greedy heuristic, which shows a faster execution time but penalizes best effort requests.

We plan to extend this work in a number of ways. We plan to further study the layered graph building step on top of the physical network topology to more robustly handle the dynamic change of topology characteristics (e.g., available bandwidth). We also plan to improve the formulation of a request's expected latency by extending the model to consider link transmission delays as well as delay introduced by VNFs (i.e., VNF processing delay).We also plan to evaluate our placement approach in a multi-DC testbed. To this purpose, we are developing a Service Request Manager component that manages the deployment of network services on top of a multi-DC environment leveraging the proposed placement algorithm. The placement decision is used to appropriately compose a Network Service Description file that is sent to a NFV Orchestrator for actual network service deployment, in compliance with ETSI standard specifications. In order to accomplish service deployment in the physical infrastructure, the Service Request Manager will interface with some existing implementations of NFV Orchestrator (e.g., OpenBaton [60]) and Virtual Infrastructure Management components (e.g., OpenStack [61]).

8. References

[1] ETSI, Network Function Virtualization - Introductory White Paper, `http://portal.etsi.org/NFV/NFV_White_Paper.pdf`, 2012.

[2] B. Martini, F. Paganelli, A Service-Oriented Approach for Dynamic Chaining of Virtual Network Functions over Multi-Provider Software-Defined Networks, Future Internet 8 (2016).

[3] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, S. Uhlig, Software-defined networking: A comprehensive survey, Proceedings of the IEEE 103 (2015) 14–76.

[4] F. Paganelli, M. Ulema, B. Martini, Context-aware service composition and delivery in NGSONs over SDN, IEEE Communications Magazine 52 (2014) 97–105.

[5] W. Stallings, Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud, Pearson Education, 2015.

[6] M. K. Weldon, The future X network: a Bell Labs perspective, CRC press, 2016.

[7] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, G. S. Paschos, The algorithmic aspects of network slicing, IEEE Communications Magazine 55 (2017) 112–119.

[8] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young, Mobile edge computing A key technology towards 5G, ETSI white paper, http://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf, 2015.

[9] Catalyst project: Maximizing profitability with network functions virtualization, TeleManagement Forum White Paper, https://www.viavisolutions.com/fr-fr/node/50121, 2014.

[10] International Telecommunication Union, Quality of telecommunication services: concepts, models, objectives and dependability planning Use of quality of service objectives for planning of telecommunication networks, E.860, https://www.itu.int/rec/T-REC-E.860-200206-I/en, 06 2002.

[11] A. Veitch, Use Cases and Analysis on Integrated NFV and Network Optimization, IETF Internet-Draft, https://tools.ietf.org/html/draft-veitch-nfvrg-nfv-nw-optimization-00, 2017.

[12] J. Evans, A. Afrakteh, R. Xiu, Demand Engineering: An new approach to SDN-based traffic management for IP and MPLS networks, CoRR abs/1606.04720 (2016).

[13] M. Mechtri, C. Ghribi, O. Soualah, D. Zeghlache, NFV Orchestration Framework addressing SFC Challenges, IEEE Communications Magazine 55 (2017) 16–23.

[14] J. Liu, Y. Li, Y. Zhang, L. Su, D. Jin, Improve Service Chaining Performance with Optimized Middlebox Placement, IEEE Transactions on Services Computing 10 (2017) 560–573.

[15] S. Draxler, H. Karl, Specification, composition, and placement of network services with flexible structures, International Journal of Network Management 27 (2017).

[16] D. Bhamare, M. Samaka, A. Erbad, R. Jain, L. Gupta, H. A. Chan, Optimal virtual network function placement in multi-cloud service function chaining architecture, Computer Communications 102 (2017) 1 – 16.

[17] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, O. C. M. B. Duarte, Orchestrating virtualized network functions, IEEE Transactions on Network and Service Management 13 (2016) 725–739.

[18] M. Mechtri, C. Ghribi, D. Zeghlache, A scalable algorithm for the placement of service function chains, IEEE Transactions on Network and Service Management 13 (2016) 533–546.

[19] A. Leivadeas, M. Falkner, I. Lambadaris, G. Kesidis, Optimal virtualized network function allocation for an SDN enabled cloud, Computer Standards & Interfaces 54 (2017) 266–278.

[20] A. Gadre, A. Anbiah, K. M. Sivalingam, A customizable agile approach to Network Function Placement, in: 2017 European Conference on Networks and Communications, EuCNC 2017, pp. 1–6.

[21] C. Pham, N. H. Tran, S. Ren, W. Saad, C. S. Hong, Traffic-aware and Energy-efficient vNF Placement for Service Chaining: Joint Sampling and Matching Approach, IEEE Transactions on Services Computing (2017).

[22] S. Dräxler, H. Karl, Z. . Mann, Jasper: Joint optimization of scaling, placement, and routing of virtual network services, IEEE Transactions on Network and Service Management (2018) 1–1.

[23] ETSI, Network Function Virtualization Management and Orchestration, GS NFV-MAN 001 V1.1.1, `http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf`, 2014.

[24] ETSI, Network function virtualization Ecosystem - Report on SDN Usage in NFV Architectural Framework, GS NFV-EVE 005 V1.1.1, http://www.etsi.org/deliver/etsi_gs/NFV-EVE/001_099/005/01.01.01_60/gs_nfv-eve005v010101p.pdf, 2015.

[25] B. Martini, F. Paganelli, A. A. Mohammed, M. Gharbaoui, A. Sgambelluri, P. Castoldi, SDN controller for context-aware data delivery in dynamic service chaining, in: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), pp. 1–5.

[26] F. Liberati, A. Giuseppi, A. Pietrabissa, V. Suraci, A. Di Giorgio, M. Trubian, D. Dietrich, P. Papadimitriou, F. Delli Priscoli, Stochastic and exact methods for service mapping in virtualized network infrastructures, International Journal of Network Management 27 (2017).

[27] Z. Xu, W. Liang, A. Galis, Y. Ma, Throughput maximization and resource optimization in NFV-enabled networks, in: 2017 IEEE International Conference on Communications (ICC), pp. 1–7.

[28] M. C. Luizelli, W. L. da Costa Cordeiro, L. S. Buriol, L. P. Gaspary, A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining, Computer Communications 102 (2017) 67 – 77.

[29] Y. Wang, P. Lu, W. Lu, Z. Zhu, Cost-efficient virtual network function graph (vnfg) provisioning in multidomain elastic optical networks, Journal of Lightwave Technology 35 (2017) 2712–2723.

[30] A. Gupta, B. Jaumard, M. Tornatore, B. Mukherjee, A Scalable Approach for Service Chain Mapping With Multiple SC Instances in a Wide-Area Network, IEEE Journal on Selected Areas in Communications 36 (2018) 529–541.

[31] A. Gupta, M. F. Habib, U. Mandal, P. Chowdhury, M. Tornatore, B. Mukherjee, On service-chaining strategies using virtual network functions in operator networks, Computer Networks 133 (2018) 1 – 16.

[32] S. Ayoubi, S. Sebbah, C. Assi, A cut-and-solve based approach for the vnf assignment problem, IEEE Transactions on Cloud Computing (2017) 1–1.

[33] S. Khebbache, M. Hadji, D. Zeghlache, Virtualized network functions chaining and routing algorithms, Computer Networks 114 (2017) 95 – 110.

[34] J. Altmann, M. M. Kashef, Cost model based service placement in federated hybrid clouds, Future Gener. Comput. Syst. 41 (2014) 79–90.

[35] B. Naudts, M. Flores, R. Mijumbi, S. Verbrugge, J. Serrat, D. Colle, A dynamic pricing algorithm for a network of virtual resources, International Journal of Network Management 27 (2017).

[36] D. Azevedo, M. Patterson, J. Pouchet, R. Tipley, Carbon usage effectiveness (CUE): a green grid data center sustainability metric, The green grid, White Paper, http://www.thegreengrid.org/en/library-andtools.aspx, 2010.

[37] A. Khosravi, S. K. Garg, R. Buyya, Energy and carbon-efficient placement of virtual machines in distributed cloud data centers, in: European Conference on Parallel Processing, Springer, pp. 317–328.

[38] K. Kaur, T. Dhand, N. Kumar, S. Zeadally, Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers, IEEE Wireless Communications 24 (2017) 48–56.

[39] B. Zhang, J. Hwang, T. Wood, Toward online virtual network function placement in software defined networks, in: 2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS), IEEE, pp. 1–6.

[40] I. Patiniotakis, Y. Verginadis, G. Mentzas, Pulsar: preference-based cloud service selection for cloud service brokers, Journal of Internet Services and Applications 6 (2015) 26.

[41] A. Amokrane, M. F. Zhani, Q. Zhang, R. Langar, R. Boutaba, G. Pujolle, On satisfying green slas in distributed clouds, in: 10th International Conference on Network and Service Management (CNSM) and Workshop, pp. 64–72.

[42] N. Joy, K. Chandrasekaran, A. Binu, Energy aware sla and green cloud federations, in: 2016 IEEE Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), pp. 7–11.

[43] ETSI, Network Functions Virtualisation (NFV); Use Cases, ETSI GS NFV 001 (V1.1.1),http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf, 2013.

[44] J. Babiarz, K. Chan, F. Baker, Configuration guidelines for diffserv service classes, Internet Requests for Comments, RFC 4594, 2006.

[45] ETSI, Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Report on architecture options to support multiple administrative domains, ETSI GR NFV-IFA 028 V3.1.1, http://www.etsi.org/deliver/etsi_gr/NFV-IFA/001_099/028/03.01.01_60/gr_NFV-IFA028v030101p.pdf, 2018.

[46] S. Clayman, E. Maini, A. Galis, A. Manzalini, N. Mazzocca, The dynamic placement of virtual network functions, in: 2014 IEEE Network Operations and Management Symposium (NOMS), pp. 1–9.

[47] S. Zhang, Z. Qian, Z. Luo, J. Wu, S. Lu, Burstiness-aware resource reservation for server consolidation in computing clouds, IEEE Transactions on Parallel and Distributed Systems 27 (2016) 964–977.

[48] Docker web site, https://www.docker.com, 2018. Accessed: 2018-03-20.

[49] S. Martello, P. Toth, Knapsack Problems, Wiley & Sons, Chichester, 1990.

[50] M. R. Garey, D. S. Johnson, Computers and intractability: a Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Francisco, 1979.

[51] M. R. Garey, D. S. Johnson, “ strong ” np-completeness results: Motivation, examples, and implications, J. ACM 25 (1978) 499–508.

[52] N. Garg, V. V. Vazirani, M. Yannakakis, Primal-dual approximation algorithms for integral flow and multicut in trees, Algorithmica 18 (1997) 3–20.

[53] B. Martini, F. Paganelli, P. Cappanera, S. Turchi, P. Castoldi, Latency-aware composition of Virtual Functions in 5G, in: 2015 1st IEEE Conference on Network Softwarization (NetSoft), IEEE, pp. 1–6.

[54] C. Li, L. Y. Li, Optimal resource provisioning for cloud computing environment, The Journal of Supercomputing 62 (2012) 989–1022.

[55] A. Betker, C. Gerlach, R. Hülsermann, M. Jäger, M. Barry, S. Bodamer, J. Späth, C. Gauger, M. Köhn, Reference transport network scenarios, MultiTeraNet Report (2003).

[56] N. Huin, B. Jaumard, F. Giroire, Optimization of Network Service Chain Provisioning, in: IEEE International Conference on Communications 2017, Paris, France.

[57] CISCO, Cisco visual networking index: Forecast and methodology, 2015.

[58] M. Savi, M. Tornatore, G. Verticale, Impact of processing costs on service chain placement in network functions virtualization, in: Network Function Virtualization and Software Defined Network (NFV-SDN), 2015 IEEE Conference on, IEEE, pp. 191–197.

[59] L. Askari, A. Hmaity, F. Musumeci, M. Tornatore, Virtual-network-function placement for dynamic service chaining in metro-area networks, in: 2018 International Conference on Optical Network Design and Modeling (ONDM), IEEE.

[60] G. A. Carella, M. Pauls, T. Magedanz, M. Cilloni, P. Bellavista, L. Foschini, Prototyping nfv-based multi-access edge computing in 5G ready networks with Open Baton, in: 2017 IEEE Conference on Network Softwarization (NetSoft), pp. 1–4.

[61] Openstack: Open source cloud computing software, `https://www.openstack.org/`, 2018. Accessed: 2018-03-20.