

Network Service Embedding with Multiple Resource Dimensions

Angelos Pentelas, George Papathanail, Ioakeim Fotoglou, Panagiotis Papadimitriou

Department of Applied Informatics, University of Macedonia, Greece

{apentelas, papathanail, fotoglou, papadimitriou}@uom.edu.gr

Abstract—The wide adoption of cloud computing, along with the advent of Network Function Virtualization (NFV) and its auspicious applications, have generated a new class of combinatorial optimization problems, with network service embedding (NSE) being one of the most prominent. NSE methods aim at improved resource efficiency and increased revenues for cloud resource providers. However, these methods commonly handle resources types with a single dimension (*e.g.*, virtual nodes with only computing demands), thus limiting the scope of the generated solutions.

In an attempt to address NSE under a pragmatic scope, we investigate the potential gains of a heuristic algorithm, which takes into account both the CPU and the memory dimension of virtual nodes. To the best of our knowledge, the novelty of our work lies on the fact that the proposed heuristic exploits insights from research on multi-dimensional virtual machine allocation, *i.e.*, the computation and accounting of a suitability metric across multiple resource dimensions. Our simulation results demonstrate that the proposed NSE method outperforms both a mixed integer linear program (MILP) and a similar heuristic, which do not account for multiple resource dimensions.

I. INTRODUCTION

The advent of cloud computing is mainly marked by a substantial share of industry shifting from traditional, *on-premise*, server preservation, to *on-demand* resource utilization, with the latter being available through large-scale virtualized infrastructures. This transition is widely embraced since, maintaining an on-premise infrastructure entails substantial operational costs in terms of power consumption, hardware upgrade, and specialized personnel.

The aforementioned trend incited the deployment and operation of softwarized networks, commonly realized via Network Function Virtualization (NFV) [1], [2] and Software-Defined Networking (SDN) [3]. NFV and SDN are considered to be mutually beneficial [4]; to fully exploit their co-existence and interoperability, though, a considerable effort towards their evolution must be made, as implied in [5], [6]. NFV and SDN essentially enable the dynamic deployment of network services (NSes), *i.e.*, sequences of virtual network functions (VNFs), specified in the form of VNF-graphs.

At the advent of NFV, VNFs were seen as a form of software middleboxes, implementing packet processing appliances for enterprise networks, such as firewalls, intrusion detection systems, load balancers, proxies, caches, etc. Nevertheless, within the 5G and edge computing era, VNFs provide a multitude of

functionalities that range from cellular network components (*e.g.*, S-GW/PDN gateways, MME) to service elements, such as object identification [7], [8], [9]. This broader scope of NFV leads to diverse computing, memory and storage demands across the range of existing VNFs. Therefore, we consider VNFs as versatile components of a NS, with resource requirements potentially equivalent to those of a whole server. Embracing this broader perception of NSes, we consider VNFs with both CPU and memory resource demands. The latter complicates the NSE optimization, since an additional (in comparison to related studies) but essential parameter (*i.e.*, memory demand) is being introduced.

In order to address this problem, we exploit insights from multi-dimensional virtual machine (VM) allocation in the design of a new heuristic for NSE with multiple resource dimensions. More specifically, we consider virtual and physical nodes as rectangles, with side lengths determined by the requested and residual capacity on two resource dimensions (*i.e.*, CPU and memory). Thereby, we exploit the abstraction of each node to a geometric figure by devising an efficiency value for the mapping between a virtual to a physical one.

To the best of our knowledge, this approach to NSE has not been studied in related work [10], [11], [12], [13]. In essence, our study investigates the potential gains stemming from coupling NSE with multi-dimensional resource allocation. More precisely, we show that the proposed heuristic compares favorably with a mixed-integer linear program (MILP) and another heuristic, both of which do not account for multiple resource dimensions.

The remainder of the paper is organized as follows. In Section II, we elaborate on the NSE and the multi-dimensional VM allocation problems, and discuss directions for their coupling. In Section III, we discuss a MILP formulation and propose a heuristic for the NSE problem with multiple resource dimensions. In addition, we present a simplified version of this heuristic that only accounts for a single dimension. Section IV compares the efficiency of the three methods using simulations. Section V provides an overview of related work, whereas Section VI highlights our conclusions.

II. PROBLEM DESCRIPTION

In this section, we elaborate on the NSE and the multi-dimensional VM allocation problems. In particular, we aim at pointing out how methods from the latter can be incorporated

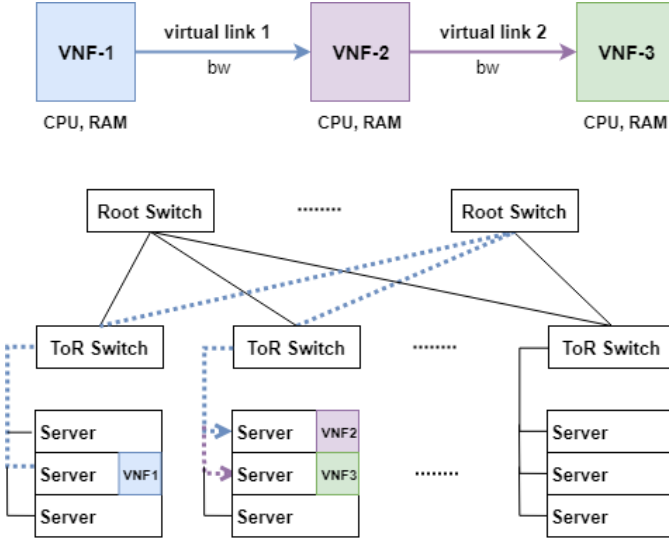


Fig. 1: NSE into a two-layer datacenter network topology.

into the former and, further explain the integration of multi-dimensional allocation principles into a heuristic algorithm that we later present and evaluate.

A. Network Service Embedding

According to [4], the deployment of NFV in virtualized network infrastructures entails a challenge known as the NFV resource allocation (NFV-RA) problem. A classification, presented in the same work, subdivides proposed NFV-RA solutions into VNF chain composition (VNFs-CC), VNF forwarding graph embedding (VNF-FGE) and VNF scheduling (VNFs-SCH). Hereby, we focus on the VNF-FGE sub-problem, which we refer as NSE.

NSE exhibits certain similarities with the Virtual Network Embedding (VNE) problem. VNE consists the main resource allocation challenge in network virtualization [14], since both virtual nodes and virtual links have to be mapped to the respective substrate components, subject to capacity constraints. Nonetheless, VNE was initially decoupled into two sub-problems; namely, the *virtual node mapping* and the *virtual link mapping* [15]. This decomposition, though, may lead to sub-optimal embeddings. This problem was rectified by subsequent VNE methods, which coordinate node and link assignment (e.g., [16]).

NSE differentiates itself from VNE in that the virtual networks are expressed as undirected graphs composed of vertices and edges, whereas NSes are specified as directed graphs [13], [17]. According to [10], a service chain represents the exact sequence of VNFs traversed by one or multiple flows. That is, NSes are defined as directed graphs composed of vertices and edges, which represent VNFs and virtual links (for VNF communication), respectively, as shown on the top of Figure 1.

The essence of the problem lies in the efficient assignment of both VNFs and virtual links to the respective counterparts of the substrate topology, while adhering to capacity constraints. Schematically, according to Figure 1, each VNF must be

placed at exactly one server, while each virtual link must be mapped to a sequence of physical links that, ultimately, forms a path that connects servers hosting adjacent VNFs.

The range of different allocation possibilities, both for VNFs and virtual links, corroborates the combinatorial nature of NSE. More specifically, NSE belongs in the class of *NP-hard* combinatorial optimization problems, being, as already mentioned, a particular instance of the *NP-hard* VNE. Furthermore, NSE can be tackled from a variety of optimization objectives, such as footprint minimization, revenue maximization, or load balancing.

B. Multi-dimensional VM Allocation

Research on single-resource VM allocation has led to several algorithms that exhibit a high degree of optimality, e.g., [18]. However, by considering multi-dimensional VMs and servers, the complexity of the problem increases significantly. That is, the notion of a VM *best fitting* a server can not be inherently extended to many dimensions. Therefore, the multi-dimensional VM allocation problem is approached as a special application of the Vector Bin Packing (VBP) problem and, currently, algorithms implemented for the former draw upon research on the latter.

The VBP problem considers the allocation of n -dimensional items to n -dimensional bins by using as few bins as possible. It has been proved to be an *NP-hard* combinatorial optimization problem (e.g., [19]); hence, a variety of approximate heuristic and meta-heuristic algorithms have been developed as more appropriate means of tackling it (e.g., [18], [20], [21], [22]). Assuming that items and bins correspond to VNFs and servers, respectively, VBP algorithms can be directly utilized to handle the *virtual node placement* phase discussed in Section II-A, although without guaranteeing success on our particular problem, since they cannot incorporate the bandwidth demands of a NS.

The limitations of a non-sophisticated allocation policy, with CPU and memory demands for VNFs, are merely illustrated in Figure 2. In particular, if the VNF is placed in *Server 1*, a significant portion of its CPU remains untapped, since the entire server's memory is utilized and, therefore, no additional VNFs can be accommodated. Instead, the placement of the VNF in *Server 2* seems more suitable for the opposite reason, i.e., the more efficient utilization of the servers' available resources. Minimizing resource wastage constitutes a key aspect of effective algorithms for the problem concerned.

Nonetheless, NSE deals with NSes comprising of more than one VNF. As a consequence, typical VBP algorithms that are driven by holistic criteria for the allocation of items to bins, cannot be directly applied to the problem that we investigate. In this respect, our work aims at exploring potential ways of leveraging VBP insights within the NSE problem space. This is anticipated to yield more efficient resource utilization in servers, compared to other relevant methods that do not exercise any notion of multi-dimensional resource mapping.

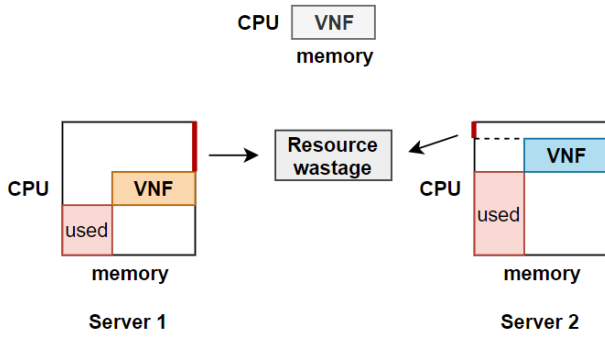


Fig. 2: The importance of efficient VNF-to-server mapping for 2-dimensional physical and virtual nodes.

TABLE I: Notations in the network model and the MILP/heuristic.

Symbol	Description
V	the set of physical nodes within the substrate topology
V_F	the set of virtual nodes comprising a network service
E	the set of physical edges within the substrate topology
E_F	the set of virtual edges between virtual nodes
d_i^k	demand of virtual node i on resource k
d^{ij}	bandwidth demand of edge (i, j) in $Mbps$
r_u^k	residual capacity of physical node u on resource k
r_{uv}	residual capacity of physical edge (u, v) in $Mbps$
x_u^i	assignment of virtual node i to physical node u
f_{uv}^{ij}	amount of bandwidth assigned to link (u, v) for the virtual edge (i, j) in $Mbps$
z_u	usage indicator of server u
s_u^i	suitability value of mapping VNF i to server u

III. NETWORK SERVICE EMBEDDING

In this section, we initially introduce models for the NSE Request (NSR) and the substrate network, *i.e.*, micro-datacenter (μDC). Subsequently, we describe in detail the NSE methods used in our evaluation.

Network Service Request (NSR) model. We use a directed graph $G_F = (V_F, E_F)$ to express a NSR. The set of nodes V_F includes all VNFs i that are associated with a demand value on resource k , denoted by d_i^k . Edges between nodes i and j , $i, j \in V_F$, are expressed as $(i, j) \in E_F$, whereas their respective bandwidth demands are denoted by d^{ij} .

Micro-datacenter (μDC) model. We model a μDC as an undirected graph $G = (V, E)$, where V is the set of all physical nodes within the datacenter, *i.e.*, servers and switches. Furthermore, r_u^k is assigned to every physical node u to express its residual capacity on resource k . Likewise, $(u, v) \in E$ denotes the link between nodes u and v , $(u, v \in V)$. Each link $(u, v) \in E$ is associated with its residual capacity, r_{uv} .

For the NSE problem that we study, we formulate, implement and evaluate three different methods. The first one is a mixed-integer linear program (MILP), which, however, exhibits the known drawback of exact methods, *i.e.*, high solver run-time. Therefore, we provide two approximate heuristics

in order to reduce solver run-time; the first one leverages on information about the geometry of virtual and physical nodes, as opposed to the second one which does not exhibit this feature.

A. Mixed Integer Linear Program (MILP)

The MILP that we employ in our study is essentially an extension of the MILP used in [10] for 2-dimensional nodes. Besides the notation for the description of the NSR and the μDC (Table I), we further use the binary variables x_u^i and z_u . The former indicates the placement of the virtual node $i \in V_F$ in the physical node $u \in V$, while the latter represents the allocation of a physical node $u \in V$ from the incumbent embedding. In addition, we utilize the variable f_{uv}^{ij} to denote the amount of flow of the virtual edge $(i, j) \in E_F$ that is mapped onto the physical edge $(u, v) \in E$. Thus, we define the objective function (1); for a given NSR, (1) prefers solutions that utilize fewer servers, while, at the same time, minimize the volume of network traffic. Our MILP formulation is as follows:

Minimize

$$\sum_{u \in V} z_u + \frac{1}{\sum_{(i,j) \in E_F} d^{ij}} \sum_{(u,v) \in E} \sum_{(i,j) \in E_F} f_{uv}^{ij} \quad (1)$$

subject to:

$$\sum_{u \in V} x_u^i = 1 \quad \forall i \in V_F \quad (2)$$

$$\sum_{\substack{v \in V_S \\ (u \neq v)}} (f_{uv}^{ij} - f_{vu}^{ij}) = d^{ij}(x_u^i - x_u^j) \quad i \neq j, \forall (i, j) \in E_F, \forall u \in V \quad (3)$$

$$\sum_{i \in V_F} d_i^k x_u^i \leq r_u^k \cdot z_u \quad \forall u \in V, \forall k \in \{1, 2\} \quad (4)$$

$$\sum_{(i,j) \in E_F} f_{uv}^{ij} \leq r_{uv} \quad \forall (u, v) \in E \quad (5)$$

$$x_u^i, z_u \in \{0, 1\} \quad \forall i \in V_F, \forall u \in V \quad (6)$$

$$f_{uv}^{ij} \geq 0 \quad \forall (u, v) \in E, \forall (i, j) \in E_F \quad (7)$$

With respect to expressions (2) to (7), constraint (2) ensures that each virtual node is mapped at exactly one substrate node (*i.e.*, server), while (3) enforces flow conservation. Constraints (4) and (5) are used to prevent infeasible placements, *i.e.*, placements that violate node and link residual capacities. Finally, constraints (6) and (7) enforce the variables' domains.

B. 2d Heuristic Algorithm

Following [10], we implement a heuristic that initially ranks the datacenter's racks in descending order, according to the average available capacity of their top-of-the-rack (ToR) to core switch links. The attempt of embedding a NS within the substrate topology initiates from the first ordered rack and ends on the last. In case that the whole NS cannot be placed within a single rack, the *min-cut* algorithm is applied to partition the request into segments, striving to minimize the amount of generated inter-rack traffic via the coordinated VNF and link assignment.

The aspect that we investigate appears during the placement of VNFs within a certain rack. For each VNF i of a given NSR, the value $s^i = \frac{d_i^1}{d_i^2}$ is calculated, with d_i^1 and d_i^2 denoting the CPU and memory requested by the VNF, respectively. Let $V_R \subset V$ be the set of servers within a single rack, then the same procedure is followed for each server $u \in V_R$, with $s_u = \frac{r_u^1}{r_u^2}$ and r_u^1, r_u^2 denoting the corresponding residual capacities of server u in terms of CPU and memory, respectively. Eventually, a suitability metric $s_u^i = |s_u - s^i|$ is computed, $\forall u \in V_R$, with the most suitable server for accommodating VNF i , expressed by $u = \argmin(s_u^i), u \in V_R$. Nevertheless, if the VNF can be placed in the same server as its predecessor, this placement is prioritized since, in this way, the generated traffic within the substrate network is reduced. These steps are shown in Algorithm 1.

Figure 3 exemplifies the logic behind the proposed algorithm, in an attempt to map a NS to a datacenter rack. In this example, we consider the embedding of a NS (composed of three VNFs) to a specific rack that consists of two servers. s -values denote the requested and available CPU-to-memory ratios for VNFs and servers, respectively. The heuristic seeks to map the VNFs sequentially, *i.e.*, it will commence with the mapping of VNF1 and terminate with VNF3.

By computing the absolute values of the differences between the ratios for the mapping of VNF1 to server1 and server2, we determine that VNF1 will be assigned to server1. In more detail, this decision is made based on the following two conditions: (i) server1 has sufficient capacity to accommodate VNF1 and (ii) $s_1^1 = |s_1 - s^1| = 0.1 < 1.5 = |s_2 - s^1| = s_2^1$ (*i.e.*, assigning VNF1 to server1 is more efficient). Subsequently s_1 is recomputed, since its available resources have changed. Ideally, VNF2 would also be placed on server1; however, it is apparent that it does not fit due to lack of available CPU resources. Thereby, VNF2 is mapped onto server2. We note that although VNF3 fits *better* in server1 (*i.e.*, $s_1^3 < s_2^3$), it will be placed on server2, where its predecessor has been assigned in order to avoid the generation of additional intra-rack traffic.

Finally, we briefly discuss additional metrics that can quantify server utilization efficiency. In addition, some of these metrics can be easily extended beyond two dimensions, *e.g.*, *L2 norm-based greedy*, *dot-product* [18], [21] and *resource skewness* [23]. For the objectives of this study, though, the s value defined above is deemed sufficient for investigating

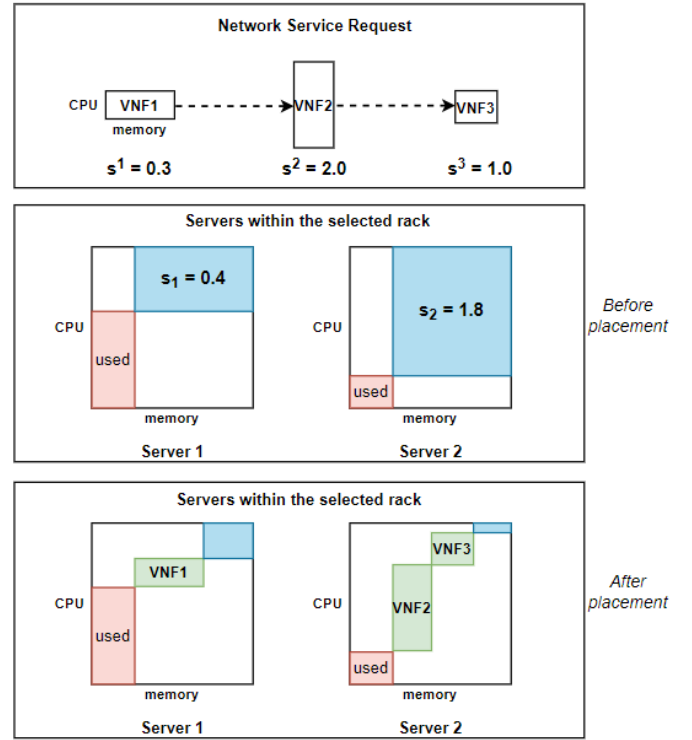


Fig. 3: NSE example with the proposed heuristic.

potential gains from coupling NSE with multi-dimensional resource allocation.

C. 1d Heuristic Algorithm

To ascertain the advantages or drawbacks of accounting for both dimensions on virtual and physical nodes, we implement another algorithm that we term as *1d heuristic*. This heuristic is similar to the one presented in Section III-B. However, instead of using the suitability metric defined in the same section, it ranks the servers within a rack according to their available CPU capacity, in descending order.

IV. EVALUATION

In this section, we initially present the parameter selection regarding the two network models described in Section III, as well as the specific settings used in our simulations (Section IV-A). Subsequently, we discuss the evaluation of the heuristics and the MILP (Section IV-B).

A. Evaluation Environment

For the substrate network topology, we simulate a μ DC with a two-layer hierarchical topology, consisting of five core switches and ten ToR switches. Each rack accommodates twenty servers. Furthermore, we consider the servers to be homogeneous and, since we associate CPU and memory values through ratios, it is essential that the respective initial capacities are normalized. In this way, we prevent the dominance of any of the two considered resource dimensions, hence, ensuring a fair comparison between the evaluated NSE methods. Finally, the maximum bandwidth of intra-rack links

Algorithm 1 2d Heuristic - VNF allocation

```

1: Input:  $V, racks, V_F, E_F$ 
2: Output: mapping, rejected_service
3:  $rejected\_service := True$ 
4:  $N := |V_F|$ 
5:  $s\_racks = sort(racks)$ 
6: for  $R \in s\_racks$  do
7:    $prevServer := "None"$ 
8:    $placed\_vnfs := 0$ 
9:   for  $i \in V_F$  do
10:     $placed\_vnf := False$ 
11:     $s_u^i = |s_u - s^i|, \forall u \in V_R$ 
12:     $sorted\_V_R = argsort(s)$ 
13:    if  $prevServer \in sorted\_V_R$  and  $i$  fits in  $prevServer$  then
14:      update mapping
15:       $placed\_vnf := True$ 
16:       $placed\_vnfs += 1$ 
17:      if  $placed\_vnfs == N$  then
18:         $rejected\_service := False$ 
19:    else
20:      for  $u \in sorted\_V_R$  do
21:        if  $i$  fits in  $u$  then
22:          update mapping,  $prevServer$ 
23:           $placed\_vnf := True$ 
24:           $placed\_vnfs += 1$ 
25:          if  $placed\_vnfs == N$  then
26:             $rejected\_service := False$ 
27:            break
28:        else
29:          continue
30:      if  $placed\_vnf$  and  $rejected\_service$  then
31:        continue
32:      else
33:        break
34:    if  $rejected\_service$  then
35:      continue
36:    else
37:      break
38:  if  $rejected\_service$  and  $|V_F| > 1$  then
39:     $V_{F_1}, V_{F_2} = minCut(V_F, E_F)$ 
40:     $mapping1, rejected1 = vnPlace(V, racks, V_{F_1}, E_F)$ 
41:     $mapping2, rejected2 = vnPlace(V, racks, V_{F_2}, E_F)$ 
42:    if  $rejected1$  or  $rejected2$  then
43:      return  $\{\}, True$ 
44:    else
45:      return  $mapping1 \cup mapping2, False$ 
46:  else if  $rejected\_service$  and  $|V_F| = 1$  then
47:    return  $\{\}, True$ 
48:  else
49:    return mapping,  $False$ 

```

TABLE II: μ DC Model Parameters

Number of Core Switches	5
Number of Racks	10
Number of Servers per Rack	20
ToR-to-Server link capacity	4 Gbps
Inter-rack link capacity	16 Gbps

TABLE III: NSR Model Parameters

CPU demand	uniform distr. [0.1, 0.5]
Memory demand	uniform distr. [0.1, 0.5]
Inbound NS traffic	uniform distr. [10, 100]
Inbound-to-outbound traffic ratio	uniform distr. [0.5, 1.5]
Splittable flow	Yes

TABLE IV: Evaluation Environment Parameters

Number of NSRs	200
Number of VNFs per NSR	uniform distr. [3, 12]

is set to be 4 Gbps while, concerning inter-rack links, the respective value is 16 Gbps, as shown in Table II.

The main parameters regarding the NS requests are depicted in Table III. We define a VNF's CPU and memory demand, as a uniformly random number in the range [0.1, 0.5]. These values essentially denote the percentage of the resources requested from a server. Furthermore, we determine the inbound traffic of a NS to be a random value within the range [10, 100] (in Mbps), whereas the inbound-to-outbound traffic ratio for each VNF in the service chain is set to a random value within [0.5, 1.5].

We conduct simulations with the embedding of 200 non-expiring NSRs, each one comprising three to twelve VNFs. NSRs are embedded in an online manner across three μ DCs with the topology specification discussed previously in this section. The simulations are executed with each one of the NSE methods presented in Section III, allowing their comparative study. All tests run on a server with 8 CPU cores at 2.1GHz and 8GB of main memory, using the Linux-based Ubuntu 16.04 (LTS) operating system. The simulation environment is implemented in Python, while for the MILP we rely on the Gurobi solver [24].

B. Evaluation Results

We initially compare the three NSE methods in terms of acceptance rate, which indicates the percentage of NSes that are successfully embedded in the substrate network, out of all submitted NSRs. As Figure 4 illustrates, the acceptance rate of the 2d heuristic is steadily higher than both of the other two methods, since the proposed algorithm manages to embed more NSes than the MILP and the 1d heuristic. In particular, the gain of the 2d heuristic over the 1d heuristic is notable, with a maximum of $\sim 6\%$ difference at 75 requests. In contrast, the gain of the 2d heuristic compared to the MILP is only marginal. This stems from a limitation of the 2d heuristic in conjunction with an advantage of the MILP, as an exact method. More precisely, the former strongly depends on the deterministic *min-cut* algorithm, whereas the MILP performs

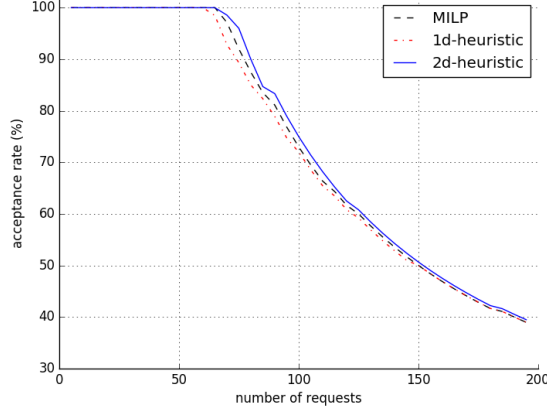


Fig. 4: Request acceptance rate.

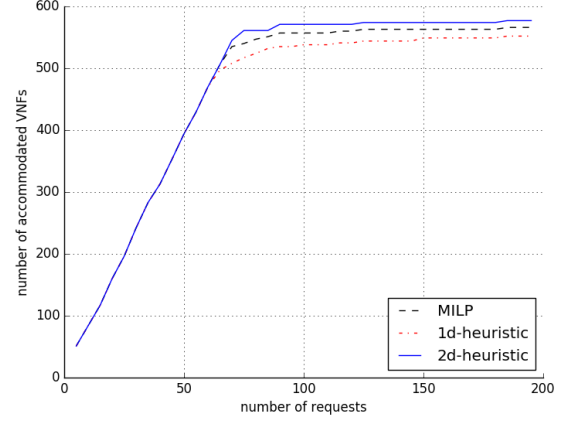


Fig. 5: Number of accommodated VNFs.

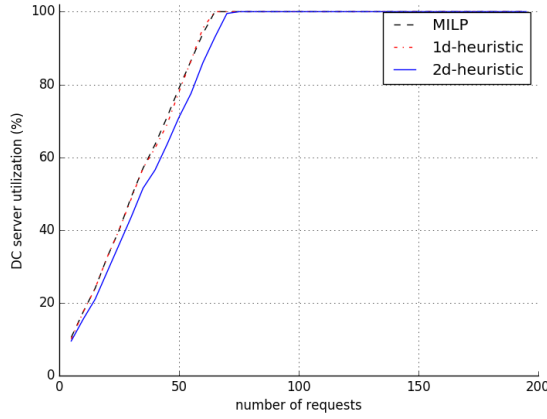


Fig. 6: DC server utilization.

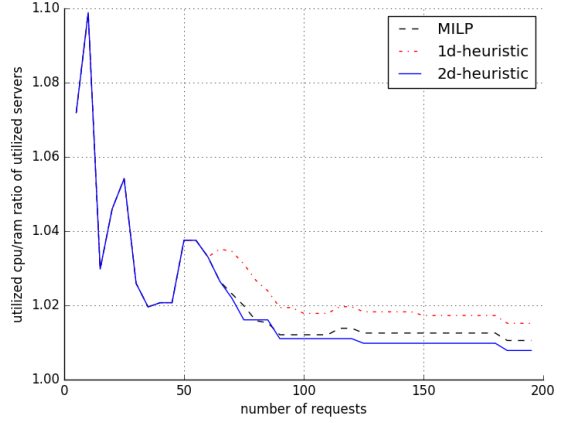


Fig. 7: CPU to memory allocation ratio in servers.

a more intensive search of the solution space to obtain the optimal mapping. However, this comes at a severe cost in terms of computation time, *i.e.*, an order of magnitude higher than the heuristic. We note that since NSes are non-expiring, the acceptance rate will eventually reach zero as the number of incoming NSRs increases, *i.e.*, the substrate network will be saturated.

The acceptance rate per se, however, does not provide sufficient information about the number of accommodated VNFs. For instance, embedding four NSes with two VNFs each and rejecting a single NS consisting of 12 VNFs yields an acceptance rate of 80%. Instead, embedding the long NS and two short ones leads to 60% acceptance rate, which may be preferred by an Infrastructure Provider (InP), especially when the number of assigned VNFs is proportional to the assigned resources. Based on Figures 4 and 5, it becomes apparent that the higher acceptance rate of the *2d heuristic* is translated into a higher number of assigned VNFs. This indicates that our algorithm computes efficient mappings for all lengths of NSes and does not skew in favor of short ones. More importantly, the proposed heuristic achieves this efficiency by utilizing in most

cases a smaller number of servers, as shown in Figure 6. As such, the *2d heuristic* yields an enhanced level of consolidation during NS mapping. For instance, according to Figure 6, the *2d heuristic* embeds 50 NSes using $\sim 5\%$ less servers than the other two methods.

In order to reason about the gains brought by the *2d heuristic* compared to the other two embedding methods, we take into account Figure 7. The three plots in this figure essentially illustrate how balanced the consumed resources of the servers are. In particular, this balance is calculated as the ratio of the average utilized CPU over the average utilized memory across all allocated servers. Evidently, the closer this ratio approaches one (since CPU and memory demands are random values in the same range), the more balanced the consumption of the different resource types in the servers. We note that the *2d heuristic* does not follow a holistic VBP-driven criterion in order to assign VNFs into servers, as explained in Section II. Instead, it exploits insights from multi-dimensional VM allocation, comprising a more efficient NSE method, suitable for *2-dimensional* nodes. In this respect, the *2d heuristic* achieves a more balanced resource allocation

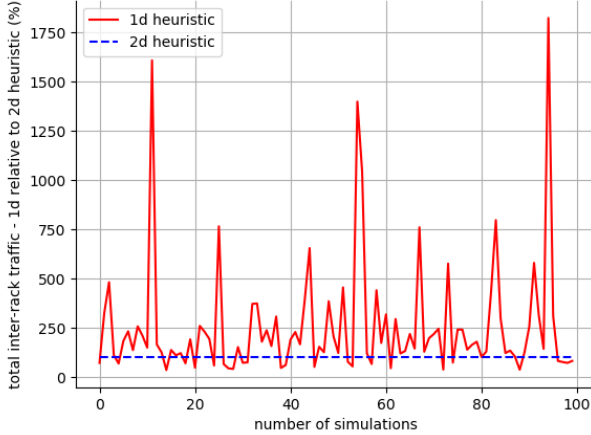


Fig. 8: Inter-rack traffic ratio (1d relative to 2d heuristic).

(across resource dimensions), compared to the MILP and the *1d heuristic*.

Furthermore, we examine the ratio of generated inter-rack traffic with both heuristics. After conducting 100 simulations, the inter-rack traffic generated from the *1d heuristic* (Figure 8) is considerably higher (percentage-wise) than the respective traffic generated by the *2d heuristic*. This stems from the incapacity of the *1d heuristic* to achieve a balanced consumption of the CPU and memory in servers, which increases the frequency of NS partitioning among the racks, *i.e.*, *min-cut* triggering. Hence, in a case of high inter-rack traffic, employing the *1d heuristic* is expected to lead to lower acceptance rates than those shown in Figure 4.

Essentially, our simulation results corroborate the viability of leveraging multi-dimensional allocation insights into an NSE algorithm, as well, as the benefits in terms of embedding efficiency. Furthermore, the proposed approach is also grounded to reality given that resource demands in cloud environments typically encompass more than one dimension.

V. RELATED WORK

Throughout this section, we briefly discuss related work on the following two areas: (i) NSE and (ii) multi-dimensional VM allocation.

A. Network Service Embedding

Methods designed to tackle the NSE problem do not necessarily share the same objective. In [25], for instance, *Ma et al.* consider profit maximization as their primary objective, while an energy-efficient embedding proposal is delivered through [26]. Another approach that has received significant research interest is the minimization of inter-rack traffic along the substrate topology. Such studies include, among others, STRATOS [13], CloudNaaS [27], Nestor [10] and DistNSE [11]. In particular, STRATOS [13] comprises a framework for a multitude of NFV orchestration functions, such as service chaining and resource provisioning. The resource provisioning aspect of STRATOS, which is the most

relevant to our work, mainly deals with flow distribution, *i.e.*, the assignment of flows to the various VNF instances. This problem is formulated as a linear program. Nestor [10] and DistNSE [11] tackle the multi-provider NSE problem, taking into account the privacy restrictions imposed by the different participating providers. Both Nestor and DistNSE decompose multi-provider NSE into two sub-problems: (i) service chain partitioning (among providers), and (ii) mapping of each chain segment onto the respective provider's domain. While Nestor relies on a centralized broker for service chain partitioning, DistNSE handles this in a distributed manner, using an embedding protocol (which essentially obviates the need for any centralized orchestrator).

The NSE problem has also been studied in the context of cellular network slicing. The aim here is to optimize the placement of VNFs that implement certain elements of a virtualized cellular network, such as the eNodeB (eNB), the Serving Gateway (S-GW), the Packet Data Network Gateway (P-GW), and the Mobility Management Entity (MME). In particular, authors in [7] propose an exact method for the embedding of such VNF-graphs onto a virtualized cellular core. Besides capacity constraints, the proposed method optimizes the placement of VNFs, taking into account delay budgets between VNFs, as mandated by 3GPP. *Papagianni et al.* follow a different approach to this NSE problem, as they promote the sharing of VNFs among multiple service chains in a network slice, as means to reduce the deployed VNF instances and, consequently, the provisioning and management cost of the virtualized cellular network [8]. To this end, the authors propose a MILP formulation for NSE with shared VNFs.

Even though all of the above-mentioned studies shed light on critical aspects of NSE, their consideration of single-dimensional nodes (typically defined by computing resources) precludes their applicability to NSEs with diverse requirements on multiple resources. Hence, we address this limitation by providing an efficient NSE algorithm that accounts for multiple resource dimensions.

B. Multi-dimensional VM Allocation

Another research domain, tightly related with our work, is multi-dimensional VM allocation. In essence, this problem is a specific application of the more generic VBP problem, as discussed in Section II-B. Within the context of mapping VMs to physical hosts, recent studies (*e.g.*, [23], [28]) primarily focus on VM consolidation, *i.e.*, the assignment of VMs into the minimum number of servers. This consists a crucial energy-saving factor in datacenter operations. Consolidation, relevant to green cloud computing, is preferred from InPs compared to other allocation objectives, such as load balancing.

Ghodsi et al. is considered to be one of the most influential studies on the discussed topic [29]. This study is among the first that approaches the fair allocation of multiple resources types as pertaining to users with heterogeneous requirements. To this end, this study proposes Dominant Resource Fairness (DRF), a multi-dimensional allocation scheme that meets a

multitude of preferable properties. However, the applicability of DRF can be seen as more suitable for large computer clusters, where all users have equal rights of utilizing their resources, as opposed to commercial cloud infrastructures, at which profit maximization is commonly sought.

An additional, quite insightful, research paper on multi-dimensional VM allocation is [23]. In particular, *Xiao et al.* propose an algorithm that can dynamically optimize the resource consumption of a server infrastructure, with consolidation being a key factor to this. In order to deal with the implications stemming from the multi-dimensional aspect, authors propose the concept of resource skewness. Essentially, skewness acts as means to quantify the efficiency of server resource utilization and triggers VM migrations that improve the overall server utilization.

Motivated from VM placement problems, the studies in [18], [21] propose new heuristics for multi-dimensional VM consolidation and the VBP problem in general. In both studies, authors underline the difficulty of extending the objective of *best fitting* VMs to servers, when considering both of them as multi-dimensional vectors. To circumvent this difficulty, they offer two heuristics; namely, the *Dot Product* and the *Norm-based Greedy*. The former allows to identify the suitability of VM requirements with residual server resources, while the latter utilizes the euclidean metric to quantify the distance between the aforementioned vectors.

The objective of the present study is aligned with related work [10], [13], [27], which targets the minimization of the generated inter-rack traffic and also ensures correctness for the embedded service chains. Since we deal with *2-dimensional* virtual and substrate nodes, we further incorporate the *s* metric defined in Section III-B. A similar approach, *i.e.*, the utilization of resource ratios, is also used in [30]. To the best of our knowledge, our approach of coupling NSE with multi-dimensional resource allocation has not been investigated in previous work.

VI. CONCLUSIONS

The wide adoption of the NFV paradigm brought about the NSE problem and raised the need for efficient algorithms to tackle it. The embedding of NSEs within substrate networks is classified as a *NP-hard* combinatorial optimization problem; consequently, most studies focus on the development of approximate algorithms to effectively obtain acceptable solutions. Yet, the vast majority of these algorithms assume *single-dimensional* nodes, *i.e.*, servers and VNFs with CPU capacities and demands, respectively. To overcome this limitation, we have presented a new NSE method that accounts for *2-dimensional* physical and virtual nodes (*i.e.*, CPU and memory).

Concerning our algorithmic approach, we have developed a heuristic that quantifies the suitability of mapping a virtual to a physical node. This is achieved through a metric that, essentially, is implied within [30]. We have further compared the efficiency of the proposed heuristic against an

exact method (MILP). Our simulation results show marginally higher performance of the heuristic against the MILP in terms of acceptance rate, number of accommodated VNFs and server utilization. To further illustrate the potential benefits of our heuristic, we compare its efficiency with a variant of itself, which only accounts for a single resource dimension, *i.e.*, CPU. This comparison uncovers significant gains for the proposed heuristic, which are reflected by a higher acceptance rate and substantially lower inter-rack traffic. These gains essentially stem from more efficient server utilization and a higher degree of VNF consolidation.

As a concluding remark, we acknowledge our algorithm's limitation of not being easily extendable beyond two resource dimensions. An evaluation and selection of the most appropriate metric that quantifies the efficiency of VNF-to-server assignment will be further investigated in future work. Despite this limitation, we consider that the proposed heuristic fulfils the main objective of this study, *i.e.*, the investigation of intertwining NSE with multi-dimensional resource allocation.

VII. ACKNOWLEDGMENTS

This work is supported by the MESON (Optimized Edge Slice Orchestration) project, co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH CREATE - INNOVATE (project code: T1EDK-02947).

REFERENCES

- [1] "ETSI Network Function Virtualization," <http://www.etsi.org/technologies-clusters/technologies/nfv>.
- [2] "OPNFV," <https://www.opnfv.org/>.
- [3] S. Agarwal, M. Kodialam, and T. Lakshman, "Traffic engineering in software defined networks," in *2013 Proceedings IEEE INFOCOM*. IEEE, 2013, pp. 2211–2219.
- [4] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.
- [5] T. Wood, K. Ramakrishnan, J. Hwang, G. Liu, and W. Zhang, "Toward a software-based network: integrating software defined networking and network function virtualization," *IEEE Network*, vol. 29, no. 3, pp. 36–41, 2015.
- [6] S. Sun, M. Kadoch, L. Gong, and B. Rong, "Integrating network function virtualization with sdr and sdn for 4g/5g networks," *IEEE Network*, vol. 29, no. 3, pp. 54–59, 2015.
- [7] D. Dietrich, C. Papagianni, P. Papadimitriou, and J. S. Baras, "Network function placement on virtualized cellular cores," in *2017 9th International Conference on Communication Systems and Networks (COMSNETS)*. IEEE, 2017, pp. 259–266.
- [8] C. Papagianni, P. Papadimitriou, and J. S. Baras, "Rethinking service chain embedding for cellular network slicing," in *2018 IFIP Networking Conference (IFIP Networking) and Workshops*. IEEE, 2018, pp. 1–9.
- [9] K. Katsaros, V. Glykantzis, P. Papadimitriou, G. Papathanail, D. Dechouniotis, and S. Papavassiliou, "Meson: Facilitating cross-slice communications for enhanced service delivery at the edge," 2019.
- [10] D. Dietrich, A. Abujoda, A. Rizk, and P. Papadimitriou, "Multi-provider service chain embedding with nestor," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 91–105, 2017.
- [11] A. Abujoda and P. Papadimitriou, "Distnse: Distributed network service embedding across multiple providers," in *2016 8th International Conference on Communication Systems and Networks (COMSNETS)*. IEEE, 2016, pp. 1–8.
- [12] N. Torkzaban, C. Papagianni, and J. S. Baras, "Trust-aware service chain embedding," in *2019 Sixth International Conference on Software Defined Systems (SDS)*. IEEE, 2019, pp. 242–247.

- [13] A. Gember, A. Akella, A. Anand, T. Benson, and R. Grandl, "Stratos: Virtual middleboxes as first-class entities," 2012.
- [14] A. Haider, R. Potter, and A. Nakao, "Challenges in resource allocation in network virtualization," in *20th ITC Specialist Seminar*, vol. 18, no. 2009. ITC, 2009.
- [15] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.
- [16] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Transactions on networking*, vol. 20, no. 1, pp. 206–219, 2011.
- [17] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, "Simplifying middlebox policy enforcement using sdn," in *ACM SIGCOMM computer communication review*, vol. 43, no. 4. ACM, 2013, pp. 27–38.
- [18] R. Panigrahy, K. Talwar, L. Uyeda, and U. Wieder, "Heuristics for vector bin packing," *research.microsoft.com*, 2011.
- [19] V. Vazirani, "Approximation algorithms springer-verlag," *New York*, 2001.
- [20] F. Parreño, R. Alvarez-Valdés, J. F. Oliveira, and J. M. Tamarit, "A hybrid grasp/vnd algorithm for two-and three-dimensional bin packing," *Annals of Operations Research*, vol. 179, no. 1, pp. 203–220, 2010.
- [21] S. Lee, R. Panigrahy, V. Prabhakaran, V. Ramasubramanian, K. Talwar, L. Uyeda, and U. Wieder, "Validating heuristics for virtual machines consolidation," *Microsoft Research, MSR-TR-2011-9*, pp. 1–14, 2011.
- [22] M. H. Ferdous, M. Murshed, R. N. Calheiros, and R. Buyya, "Virtual machine consolidation in cloud data centers using aco metaheuristic," in *European conference on parallel processing*. Springer, 2014, pp. 306–317.
- [23] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 6, pp. 1107–1117, 2012.
- [24] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2019. [Online]. Available: <http://www.gurobi.com>
- [25] Y. Ma, W. Liang, Z. Xu, and S. Guo, "Profit maximization for admitting requests with network function services in distributed clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 5, pp. 1143–1157, 2018.
- [26] V. Eramo, M. Ammar, and F. G. Lavacca, "Migration energy aware re-configurations of virtual network function instances in nvf architectures," *IEEE Access*, vol. 5, pp. 4927–4938, 2017.
- [27] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "Cloudnaas: a cloud networking platform for enterprise applications," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*. ACM, 2011, p. 8.
- [28] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.
- [29] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types," in *Nsdi*, vol. 11, no. 2011, 2011, pp. 24–24.
- [30] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 331–340.



European Union
European Regional
Development Fund



HELLENIC REPUBLIC
MINISTRY OF
ECONOMY & DEVELOPMENT
SPECIAL SECRETARY FOR ERDF & CF
MANAGING AUTHORITY OF EPANAK



EPANAK 2014-2020
OPERATIONAL PROGRAMME
COMPETITIVENESS
ENTREPRENEURSHIP
INNOVATION



ΕΣΠΑ
2014-2020
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
Partnership Agreement
2014 - 2020

Co-financed by Greece and the European Union