Evaluating Impacts of Traffic Migration and Virtual Network Functions consolidation on Power Aware Resource Allocation Algorithms

Khaled Hejja*, Xavier Hesselbach

Dept. Network Engineering, Universitat Politecnica de Catalunya, C/ Jordi Girona, 1-3 - Edif.C3 - Campus Nord - 08034 Barcelona - Spain

Abstract

Power consumption minimization and speed of solving the resource allocation problem on cloud datacenters adopting network function virtualization architecture are among the hot topics for future Internet networks. Therefore, this paper proposes a new power aware resource allocation algorithm supporting physical servers' consolidations combined with virtual networks consolidation to minimize datacenters' total costs for offline scenario. In addition, the new algorithm is also integrated with an optional standalone traffic migration algorithm that can be triggered according to specific conditions and at anytime. Simulations and evaluations of the algorithm resulted on lower total costs by 30% compared to recent algorithms from [15], and when virtual network functions consolidations option was activated, total costs were 25% lower than when it was inactive. However, when migrations option was activated in the proposed allocation algorithm it did not provide any significant savings in the total power consumptions, mainly because of the allocation strategy used by the algorithm in the first place, which managed to help it to precisely allocate and efficiently utilize the least physical resources. Finally, the results showed that without migrations, allocation times where faster by 10 times than activating migrations, suggesting to apply the migration option for emergency or maintenance conditions, and use the algorithm without migrations for faster allocations and efficient power consumptions.

Keywords: NFV Resource Allocation; Virtual Machines Consolidations; Traffic Migration.

1. Introduction

In future Internet networks, cloud datacenters are the main leveraging nodes to handle big data streams from cloud users and Internet of things (IoT) applications. These datacenters are usually equipped with very powerful computing, storage and networking resources that consume huge amount of power, and require solid resource management techniques to allocate the demanded network services (NS) efficiently.

Consequently, ITU and ETSI stressed on the benefits of designing these datacenters based on network function virtualization (NFV) architecture [3]-[9], which decouples the control plan that makes decisions about where traffic is sent, from the data plan which forwards the traffic physically to the selected destination network according to control plane logic. This is done in NFV environment by separating the network functions, such as firewalls, intrusion detectors, and caching, to name a few, from proprietary physical hardware appliances so they can run in software [8]. Accordingly, these physical resources could be shared among vast number of virtual machines (VM), to run the virtual network functions (VNFs) that were chained into service function chains (SFC) representing the network services. This process is known as resource allocation in NFV architecture, and is denoted by (RA-NFV).

To preform a successful and efficient RA-NFV process, cloud providers need to incorporate into their NFV based networks a precise resource allocation and power aware algorithms, and consider the consequent trade offs between minimizing power consumptions versus preserving the required quality of services (QoS). This was the target of ETSI in [9] which stated that NFV frameworks shall significantly reduce power consumption of networks' infrastructure, so that they support workload consolidations by scaling traffic loads and concentrate them on smaller number of servers during off-peak times, and consequently allow to turn-off or put on power saving mode all other not loaded servers.

In accordance with these requirements, several researchers applied virtual machines consolidation technique to minimize total power consumption in the physical networks. Therefore, authors of this paper are extending their previous work from [1] and [2], and propose a new power aware RA-NFV algorithms for offline and migration applications, to allocate the demands of service function chains that are known in advance, on the most appropriate physical resources, targeting minimizing the costs of the power consumption in the physical network. Key feature in the proposed algorithms is that they adopted the path construction methodology developed by [1], called segmentation, which was applied to solve the virtual network embedding problems, and modified it to work on NFV architecture. Consequently, the algorithms will allocate virtual network functions and their edges together, and in full coordination, on a physical network path having enough hosting resources, while guaran-

*Corresponding author

Email address: Khaled.Hejja, Xavier.Hesselbach @ upc.edu (Khaled Hejja*, Xavier Hesselbach)

teeing least power consumption in the whole network.

Moreover, the new algorithms in this paper deffer from those in [2] by adding a new optional feature that, combines virtual network functions consolidation together with the virtual machines consolidations for offline condition to further reduce the total power consumptions. The new algorithm targets allocating the cores of the largest VNF in any SFC, instead of allocating the total sum of the demanded cores by all VNFs of that SFC. In this way the allocated cores can be used by other smaller VNFs, and consequently will allow for further consolidations of the virtual machines running the VNFs on the hosting physical servers.

To test the new algorithms, four experiments were designed to evaluate the allocation process including traffic migrations and VNFs consolidations for offline conditions. First experiment does not consider VNFs consolidations, but only virtual machines consolidations, and was based and conducted against the recent work from [15], shedding more insights about the capabilities of the authors' work from [2], and providing further analysis and evaluations for the impacts of varying idle to maximum power ratio of datacenters' servers on total costs and traffic migrations. Second, third and fourth experiments focused on analyzing the performance of the new modified algorithms using both VNFs and virtual machines consolidations together. The second experiment kept the migration algorithm always active to evaluate the total costs and servers' utilizations when VNFs consolidations were allowed, and compared the results to the case when VNFs consolidations were not allowed. The third experiment kept the VNFs consolidations always active, but analyzed the algorithms' speed of allocation with and without migrations. Finally, in the fourth experiment, migrations were turned off completely, and focused on the allocation times of the proposed algorithm with and without VNFs consolidations.

Main contributions:

1. This paper proposes a new offline power aware resource allocation algorithm to solve RA-NFV problem in fractions of a second.
2. The new algorithm supports virtual network functions consolidations together with physical servers consolidations to further reduce the operational and power consumption costs.
3. Traffic migrations algorithm is another proposed optional feature, which can be used at anytime to consolidate traffic loads on the least active servers, but also can be used during emergency and maintenance conditions.

Rest of the paper is organized as follows: Section 2 provides related work, section 3 discusses the modified segmentation methodology that will be used for paths' constructions. Then section 4 detailes in depth the specifics of the offline system model, and 5 presents the simulation settings and discusses the results of the conducted experiments to test the proposed algorithms. Lastly, conclusions and future works are presented in section 6.

2. Related Work

A comprehensive survey of NFV was conducted by [10], providing an overview of the key research topics, standardization efforts, early implementation, use cases, and relationship between NFV, SDN and cloud computing. Another survey study was done by [11], where the authors presented a classified and comprehensive review for the NFV resource allocation problem, including a classification of the resource allocation problem in NFV, considering the VNFs chain composition, embedding and scheduling, optimization objectives, solution strategies and application domains.

Regarding resource allocation modeling in NFV environments, an early work targeting the complex scheduling problem was presented by [12], in addition to that, the same authors proposed an analytical model for the NFV forwarding graph to optimize the execution time of the Network Services' deployment [13]. An optimal solution for orchestrating the VNFs in small scale networks was presented in [14], who proved the NP-hardness of the NFV orchestration problem, and suggested a heuristic to solve it faster for real-world synthetic topologies and traffic traces. Moreover, the work in [31] introduced a dynamic fault tolerance and an efficient resource utilization elastic scheduling algorithm for scheduling real-time tasks in the cloud considering the system performance volatility, and [33] developed a dynamic real-time fault-tolerant model for task allocation and message transmission to ensure faults can be tolerated during the work flow execution.

For related work covering power efficiency in NFV environment, a recent article by [15] investigated the consolidation, routing and placement problems in NFV architectures based on vertical scaling techniques. The authors proposed a modified algorithm to improve the blocking performance, in which bandwidth rather than processing capacity was proposed as the constrained resource. In addition, the authors proposed a new consolidation technique taking into account the reconfiguration costs within the migration strategy. In [16] the authors formulated and modeled a virtual network function placement problem for power and traffic-aware cost minimization. To solve it, a novel approach that combines the Markov approximation with matching theory was proposed to find an efficient solution for the original problem. Furthermore, in [17], the authors proposed a power efficient VNF placement and chaining algorithm that minimizes the power consumption of the servers and switches. They formulated the problem using a Decision Tree model, and solved it using Monte Carlo Tree Search strategy. Moreover, [29] proposed a joint server and network consolidation model that takes into account the power efficiency of both, the switches forwarding the traffic and the servers hosting the VMs, and it powers down switch ports and routes traffic along the most energy efficient path towards the least energy consuming server under QoS constraints. Similar work was conducted by [30] who proposed an energy aware and QoS aware multi objective Ant Colony Optimization approach for virtual machine placement and consolidation, which makes a trade-off between energy efficiency, system performance, and service level agreement compliance. In addition, [32] proposed a self-adaptive

network-aware virtual machine clustering and consolidation algorithm, which periodically checks whether consolidation is necessary in order to cluster and consolidate virtual machines.

Moreover, the authors in [18] proposed a VNF placement algorithm that allows users to meet their service latency requirements while minimizing the power consumption at the same time. In the proposed algorithm, a shortest path between a source and a destination was first computed using Dijkstra algorithm, based on the service latency requirements from the users. Then, the VNFs are allocated to the appropriate virtual machines representing the physical infrastructure component along the path that can minimize the power consumption. In [19], the authors tried to find the most suitable node for the placement of a VNF from the service chain in order to minimize the total power consumption cost with certain constraints. They designed a power saving model using queuing network for the placement of multiple service chains on the network. In addition to that, the authors in [23] developed performance-aware placement algorithm which tracks the network elements to consolidate, and places the SFCs compactly and optimizes the global packet transfer cost. Moreover, [24] proposed a dynamic virtual machine consolidation algorithm, which minimizes the energy consumption of cloud datacenters through exploiting cloud service users' information, and in [25] the authors proposed a resource management scheme for virtual machine consolidation and optimization, to perform virtual machine allocation as well as detection and consolidation of the overloaded or underloaded physical machine.

For migrations, [26] gave an overview of virtual machines migration and discussed both its benefits and challenges, and classified virtual machines migration schemes based on manner, distance, and granularity. Then they reviewed the non-live migration and comprehensively surveyed live migration strategy based on memory data migration, storage data migration, and network connection continuity. Moreover, they elaborated on a quantitative analysis about virtual machine migration performance, and summarized the studies that covered virtual machines migration to user mobility. At last, they listed the open issues of optimizations on live virtual machines migration. Additionally, [27] provided a detailed review of the live migration of virtual machines and its main approaches in cloud computing environments. The authors reviewed the state-of-the-art optimization techniques devoted to developing live virtual machines migration according to memory migration, in addition to that they highlighted the open research issues that necessitate further investigation to optimize the process of live migration for virtual machines. Moreover, the survey paper of [28] provided state-of-the-art for virtual machine placement and migration in the area of cloud computing. They detailed cloud computing background, reviewed several existing proposals, discussed problem formulations, summarized advantages and shortcomings of the reviewed works, and highlighted the challenges for new solutions.

3. Segmentation Technique for NFV networks

The segmentation technique developed by [1] is modified in this paper to work for an NFV based physical networks when constructing the candidate paths for allocating the service function chains. Therefore, the following paragraphs discuss in depth the segmentation technique for datacenters using NFV architecture, in addition to explaining how to construct the segments of a service function chain and a physical path.

3.1. Review for the segmentation concept

It provides a direct technique to represent the information in any path. Assume that the set R of SFCs are required to be allocated on a physical network of datacenters, the proposed segmentation methodology will start converting the structure of each SFC^r, where $r \in R$, listing all information about its nodes and edges together in a set format denoted by Seg^r, which represent the demands of SFC^r. Afterwards, to allocate SFC^r, the information about the resources of a specific physical network path, including its nodes and edges, will also be converted into a physical segment format as well, and will be denoted as Seg^P.

To understand the segmentation technique more clearly, the following sections will use the physical network shown in Fig.1b to explain the segmentation technique. It includes two datacenters (Datacenter-1 and Datacenter-2) of fate-tree topology, composed of four access nodes, a, b, c, d, two routers assigned as nodes 1 and 2, two switch nodes 3 and 4, and five server nodes 5, 6, 7, 8, and 9. The physical server nodes are identified by their processing cores, cpu_p, and the physical edges by their bandwidth capacities, bw_{ij}.

3.2. Formulating service function chain segment (Seg^r)

Assume SFC$^1_{ad}$ demanding forward throughput capacity of bw^1 and end-to-end delay d^1 between access node a (at Datacenter-1) and access node d (at Datacenter-2), and requests allocating core processing power of cpu^1_1 for VNF^1, cpu^1_2 for VNF^2, and cpu^1_3 for VNF^3. To construct SFC$^1_{ad}$ segment, the concept of path segment from [1] can be modified to translate the information of SFC$^1_{ad}$ as shown in Fig.1a and construct its segment as in Eq.(1) bellow:

$$Seg^1_{ad} = \{a^1_{loc}, d^1_{loc}, cpu^1_1, cpu^1_2, cpu^1_3, bw^1, d^1\} \quad (1)$$

In this way, the demands of SFC$^1_{ad}$ were translated into a segment representing the demands of SFC$^1_{ad}$. Same procedure can be done for SFC$^2_{ab}$ as shown in Fig.1c.

3.3. Formulating physical path segment (Seg^P_{ad})

Assume that the NFV orchestrator already has the list of all pairs in the physical network shown in Fig.1b, and it can instruct the hypervisor layer to list all pairs that can be used to construct each physical path in the network in advance. Hence, the segmentation section in the proposed algorithm in this paper will select a path fulfilling the locations of the demanded access nodes and all the pairs in between.

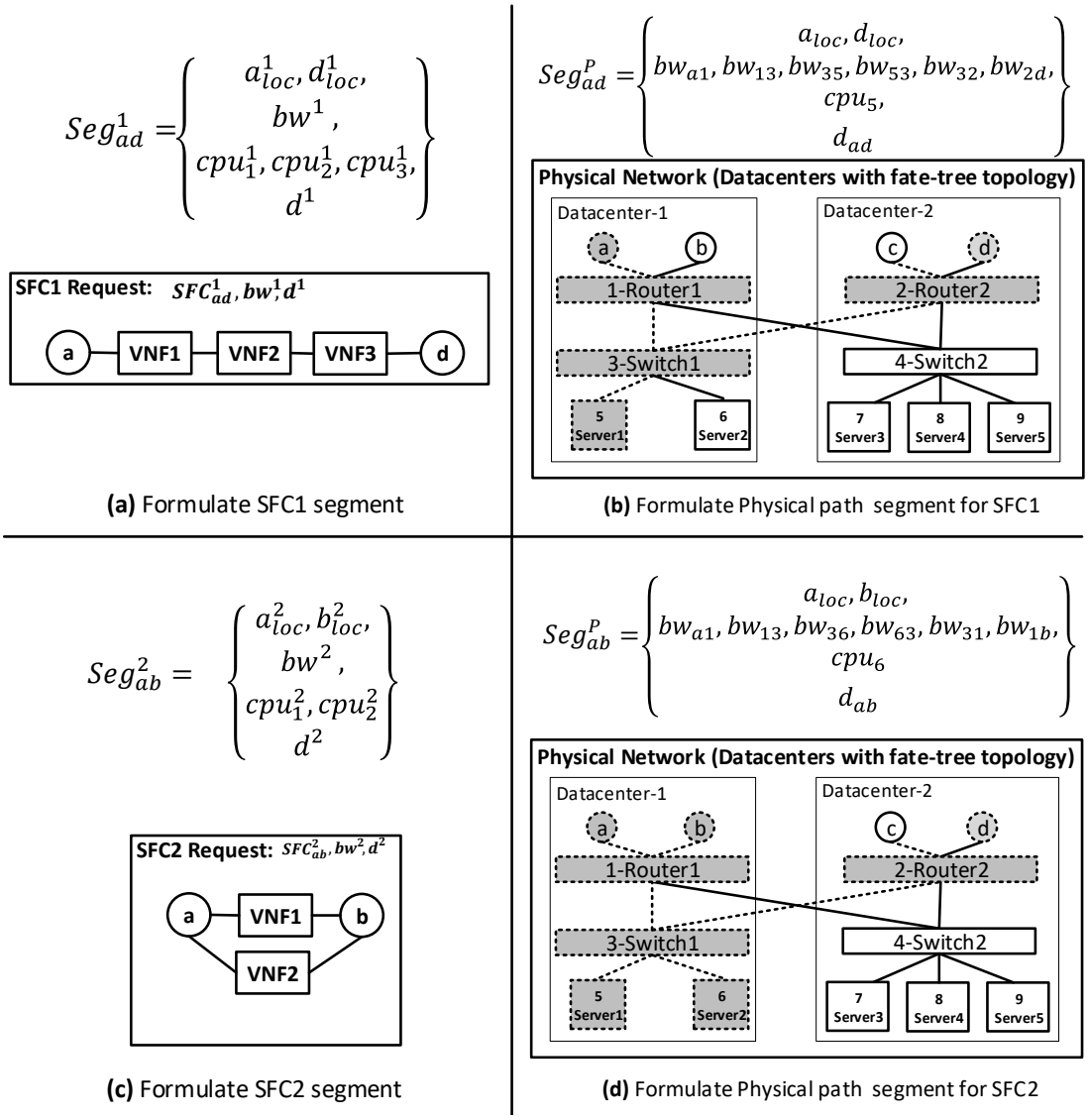

Figure 1: Segments' formulation demonstration: (a) shows the service function chain SFC_{ad}^{1} and its segment, (b) shows physical network and the segment of a candidate hosting path P_{ad}, (c) shows service function chain SFC_{ab}^{2} and its segment, (d) shows physical network and the segment of a candidate hosting path P_{ab}

To clarify that, refer to Fig.1b. Given that all physical paths and the list of their pairs are known in advance, and since SFC^1_{ad} demanding access between nodes a and d, the proposed allocation algorithm will select the physical paths that match the demanded source and destination access nodes.

For example, assume the physical path $P_{ad} = \{a, 1, 3, 5, 3, 2, d\}$ was the top path in the list, starting from the demanded source node a passing through router-1 (defined as node 1), then switch-1 (as node 3), reaching server-1 (as node 5). This is the forward path from source access node a to server node 5 given as $\{a, 1, 3, 5\}$. The other forward path $\{5, 3, 2, d\}$ from server 5 to destination access node d, starts from server node 5, passing through switch-1 (node 3), then router-2 (as node 2), and finally reaches the demanded destination access node d. Accordingly, Seg^P_{ad} representing path P_{ad} can be constructed as shown in Eq.(2), which includes the locations of the required access nodes, the available throughputs passing through each edge between the assigned access nodes, processing power of the the candidate server that will host VNFs of SFC^1_{ad}, and sum of delays of all edges in the path P_{ad}:

$$Seg^P_{ac} = \{a_{loc}, d_{loc}, bw_{a1}, bw_{13}, bw_{35}, bw_{53}, bw_{32}, bw_{2d}, cpu_5, d_{ad}\} \tag{2}$$

In this way, the resources of P_{ad} were translated into a segment representing the physical path P_{ad}. Same procedure can be done for P_{ab} as shown in Fig.1d.

3.4. Segmentation technique for coordinated allocation

To allocate all demands of SFC^1_{ad} on the physical path P_{ad} in full coordination between nodes and edges allocations, each element in the two formulated set of segments Seg^1_{ad} and Seg^P_{ad} will be directly compared one-to-one, which means for each parameter value in Seg^1_{ad} representing a virtual function in SFC^1_{ad}, check if its counterpart in Seg^P_{ad} can fulfill its demands, and for the demanded throughput capacity and delay by Seg^1_{ad}, check if each edge in Seg^P_{ad} has at least enough residual bandwidth capacity to host bw^1_{ad}, and the sum of delays from each of these edges does not exceed the demanded end-to-end delay given by d^1_{ad}.

This process can be achieved ***if and only if*** each comparison statement in the (**if-AND**) conditions shown in Eq.(3) is true, which guarantees that all the demands of SFC^1_{ad}, including access nodes locations, processing powers of its virtual functions, and bandwidth and delays of its virtual edges, will be hosted together as one set by their counterparts in the physical path P_{ad}.

$$\begin{gathered}
\textit{if } a_{loc} = a^1_{loc} \textbf{ AND} \\
\textit{if } d_{loc} = d^1_{loc} \textbf{ AND} \\
\textit{if } bw_{a1} - bw^1 \geq 0 \textbf{ AND} \\
\textit{if } bw_{13} - bw^1 \geq 0 \textbf{ AND} \\
\textit{if } bw_{35} - bw^1 \geq 0 \textbf{ AND} \\
\textit{if } bw_{53} - bw^1 \geq 0 \textbf{ AND} \\
\textit{if } bw_{32} - bw^1 \geq 0 \textbf{ AND} \\
\textit{if } bw_{2d} - bw^1 \geq 0 \textbf{ AND} \\
\textit{if } cpu_5 - (cpu^1_1 + cpu^1_2 + cpu^1_3) \geq 0 \textbf{ AND} \\
\textit{if } d_1 \leq d_{ad}
\end{gathered} \tag{3}$$

The same procedure can be done to guarantee all the demands of SFC^2_{ab}, will be hosted together as one set by their counterparts in the physical path P_{ab}.

Based on this process, segments formulations and the check-up comparisons made the allocations process of the SFCs' virtual functions, fully coordinated with the allocations of their virtual edges, together and on the same physical path.

4. Offline modeling, algorithm, and simulation

The following subsections will introduce and explain the overall design model for the RA-NFV system in offline scenario, starting by defining the physical network model and the service function chain model, then introduce the offline problem formulation including the objective function and its constraints. Moreover, the offline resource allocation and migration algorithms will be explained and discussed. Notations and their description are summarized in Table 1.

4.1. Offline Scenario

Offline scenario is designed to evaluate the overall performance of the proposed algorithms under fully controlled conditions when allocating the service function chains that are known in advance. In this paper, two algorithms were proposed for the offline scenario, the first one is an offline power aware allocation algorithm, PaNFV, coordinating allocations of the physical servers and edges' resources as demanded by the virtual nodes and edges of the service function chains. The second algorithm is the migration extension of PaNFV, denoted by mPaNFV, which is triggered by the allocation algorithm, PaNFV, to check if migrating some or all of the already allocated SFCs would result on minimizing the total costs of the physical infrastructure network.

4.2. Physical Network Model

The physical network is assumed following the network function virtualization architecture as proposed by [8]. It is modeled as a weighted directed graph $G^P = (N^P, E^P)$, where N^P and E^P are the sets of physical nodes and edges respectively, given that $i \in N^P$ represents a physical node from the set N^P, and $(i, j) \in E^P$ represents a physical edge from the set E^P. The set of physical nodes N^P is composed of two sets, $i^a \in N^P$ referring to the physical access nodes, which has no computing resources, but only serves as an originating or terminating interface points for the service function chains SFCs, and $i^p \in N^P$ referring to the set of physical server nodes, which will host the virtual functions of the SFCs. Each $i^p \in N^P$ is characterized by the following parameters: maximum processing power capacity CPU^p_i in terms of the number of cores in i^p, cpu_{i^p}

representing the current available cores, cpu^{min} the minimum consumed cores to trigger traffic migration, and P^{Busy} and P^{idle} are the maximum and idle power consumptions of the physical server. Each physical edge $(i, j) \in E^P$ transferring the traffic between a pair of physical nodes i and j is associated with maximum bandwidth capacity given by, BW_{ij} in bps, and the current available bandwidth bw_{ij}, and delay d_{ij}. In addition to that, the network is also characterized by the set of all directed paths given by $P^P = \{P_{sd}\}$, where $P_{sd} = \{(i, j)\}$ represents a directed path connecting any physical source node s to destination node d and is constructed of more than one physical edge (i, j).

4.3. Service Function Chain Request Model

Given the set F representing the types of all virtual network functions that can be used to compose logical graphs representing each SFC, where $F = f_1, f_2, ..., f_F$ and f_u being the u^{th} VNF type (typically a VNF could be: Deep Packet Inspector (DPI), Policy Control and Charging Rules Function (PCRF), Load Balancer (LB), Firewall (FW),...), and each has a specific packet processing time given by t_u^{proc}. Assume it is required to allocate a total of R service function chain requests, where SFC^r is composed of one or more f_u nodes and can be modeled as a weighted and directed graph $G^r = (N^r, E^r)$, where N^r and E^r are the sets of the logical f_u nodes and their connecting logical edges respectively. $u \in N^r$ is the VNF node u in the set of VNFs N^r, and $(u, v) \in E^r$ is a virtual logical edge belonging to the set of edges E^r connecting VNFs u and v. The set of VNFs N^r are composed of two sets, N_A^r representing the set of fictitious logical access nodes used for originating and terminating SFC^r, and N_S^r representing the set of logical VNF server nodes belonging to SFC^r, and $u \in N_S^r$ is a logical virtual server node associated with cpu_u representing the demanded cores for consumption by VNF node u, and T^u is the sum of throughputs incoming to VNF node u. Each SFC^r is associated with L^r representing the traffic flows' packet length of SFC^r, demanded bandwidth capacity denoted by bw^r, and demanded end-to-end delay denoted by d^r.

4.4. Problem Formulation

In this paper the RA-NFV problem is modeled as an integer linear programming (ILP) problem of optimization objective function, having positive integer and linear variables, and solved based on satisfying certain number of constraints. The following paragraphs introduce and formulate the RA-NFV ILP objective function and its constraints as follows:

4.4.1. RA-NFV objective function and formulation

For the offline scenario, the objective function aims to minimize the total cost C^{tot}, which sums the total power consumption costs, C^{pc}, and the total migration costs, C^{mig}, in the whole physical network when S_R successful SFCs' allocations were performed [15]. The power consumption costs, C^{pc}, sums up the cost of the consumed power by the servers when they were idle, and the cost of the traffic handled by the servers due to the allocated SFCs. The migration costs, C^{mig}, includes the impacts due to migrating all or some of the S_R successfully allocated SFCs. It is important to notice that the migration costs reflect the allocations' efficiency, giving that the lower the migration costs the more optimal and efficient were the allocations in the first place.

To make sure that a specific server node is hosting at least one VNF, variable $x_{i^p}^u$ is used in the ILP objective function formulation, which takes a binary value of 1 if server node i^p is assigned to host VNF u from SFC^r. Variable λ_{i^p} is 1 if physical node i^p is active, or 0 if it is turned-off, variable $\mu_{i^p}^u$ is 1 if all VNF nodes hosted by i^p are migrating, or 0 if they remain, and variable $x_{ij}^{uv^r}$ is set to 1 if physical edge (i, j) is hosting virtual edge (u, v).

Mathematical formulation of the objective function C^{tot}, C^{pc}, and C^{mig} were based on [15] and are represented in Eq.(4), Eq.(5), and Eq.(6) as follows:

$\forall i^p \in N^P,\ \forall r \in R$

$$min\ C^{tot} = C^{pc} + C^{mig} \tag{4}$$

$$C^{pc} = \beta_e \Delta t (P^{idle} \sum_{i^p \in N^P} \lambda_{i^p} + \sum_{i^p \in N^P} \frac{\sum_{u \in N^r} x_{i^p}^u t_u^{proc} T^u / L^u}{CPU_{i^p}}) \tag{5}$$

$$C^{mig} = \beta_d \sum_{r \in S_R, u \in N^r} T_{down}^r T^u \mu_{i^p}^u \tag{6}$$

β_e is the unit cost of a consumed 1 Watt of power, β_d is the revenue loss due to migrating 1 Gbit of traffic, Δt is the allocation duration interval, T_{down}^r is the downtime of all VNFs in SFC^r during the migration process, and T^u is set equal to the demanded bandwidth by the SFC, since each VNF has one incoming edge only [15].

Accordingly, to minimize the overall costs in the physical network, the objective function works in two directions: first it attempts to minimize the total power consumption costs in the whole physical network due to all allocated SFC^r. Second, to further minimize the costs of the physical network, the objective function considers the migration costs coming from migrating the hosted traffic by the under utilized physical servers to other more utilized physical servers. This in turn should lead to freeing and turning off the old servers if any.

4.4.2. Constraints formulation

The solution of the objective function Eq.(4) will be controlled by capacity and domain constraints as shown bellow.

1. *Constraints on the physical server nodes*

   To ensure that the maximum CPU capacity in physical server node i^p is greater than or equal, to the demanded capacities by all allocated VNFs on this server node, Eq.(7) is formulated as follows:

   $$\forall\ i^p \in N^P \sum_{r \in R, u \in N^r} cpu_u x_{i^p}^u \leq CPU_{i^p} \tag{7}$$

   To ensure that a server is switched-on when it hosts at least one virtual function node Eq.(8) is formulated as follows:

   $$\forall\ i^p \in N^P, \sum_{u \in N^r} x_{i^p}^u \geq \lambda_{i^p} \tag{8}$$

Table 1: Notation

Notation	Description.	Notation	Description.
G^P	Physical network directed graph.	N^P	Set of all physical nodes.
E^P	Physical network edges.	CPU_{i^p}	Maximum CPU capacity at server i.
cpu_{i^p}	Current available CPU at server i.	cpu^{min}	Minimum CPU to trigger migration.
BW_{ij}	Maximum bandwidth of edge (i, j).	bw_{ij}	Current available bandwidth in edge (i, j).
P^P	All paths in the physical network.	P_{sd}	Physical path between s and d.
F	All virtual network functions' types.	SFC^r	Service function chain number r.
G^r	SFC^r weighted directed graph.	N^r	All VNFs in SFC^r.
N^r_A	Set of all logical Access nodes.	N^r_S	Set of all logical server nodes.
E^r	All virtual edges in SFC^r.	T^r_{down}	Average down time of migrating SFC^r.
R	Set of SFCs to be allocated.	Δt	Cyclic stationary intervals' duration.
T^u	Sum of throughputs incoming to VNF u.	L^u	Packet length of SFC's traffic flows in Bytes.
cpu^u	Demanded CPU capacity by VNF u.	t^{proc}_u	Packet processing time of u.
bw^r	Demanded bandwidth by SFC^r.	S_R	Set of successfully allocated SFCs.
Seg^P	P_{sd} segment listing all its parameters.	Seg^r	SFC^r segment listing all its parameters.
C^{tot}	Total cost of the physical network.	C^{pc}	Cost of power consumption.
C^{mig}	Cost of migration.	a	Idle to maximum power ratio.
P^{idle}	$a * P^{Busy}$.	P^{Busy}	Maximum power consumption of the server node.

2. *Migration constraint*

To ensure that a virtual function node u is migrating, if the processing capacity of the physical server node i^p hosting it is below the threshold value cpu^{min}, Eq.(9) is formulated as follows:

$$\forall\ i^p \in N^P \sum_{r \in R, u \in N^r} cpu_u x^u_{i^p} \leq cpu^{min} \mu^u_{i^p} \tag{9}$$

3. *Constraints on the physical edges*

To ensure that the total consumed bandwidth on physical edge (i, j) is less than or equal to the maximum bandwidth capacity at that edge, Eq.(10) is formulated as follows:

$$\forall\ ij \in E^P\ \forall\ uv \in E^r \sum_{r \in R} bw^r x^{uv^r}_{ij} \leq BW_{ij} \tag{10}$$

4. *Domain constraints*

To solve the problem as an ILP, Eq.(11)-Eq.(14) are defined as follows:

$$\forall\ i^p \in N^P\ x^u_{i^p} \in \{0, 1\} \tag{11}$$

$$\forall\ i^p \in N^P\ \mu^u_{i^p} \in \{0, 1\} \tag{12}$$

$$\forall\ i^p \in N^P\ \lambda_{i^p} \in \{0, 1\} \tag{13}$$

$$\forall\ ij \in E^P\ \forall\ uv \in E^r\ x^{uv^r}_{ij} \in \{0, 1\} \tag{14}$$

To ensure that all virtual functions $u \in N^r$ of a single SFC^r are mapped only to one physical server node i^p, Eq.(15) is defined as follows:

$$\forall u \in N^r \sum_{\forall i^p \in N^P} x^u_{i^p} = 1, \tag{15}$$

4.5. PaNFV to solve RA-NFV in full coordination

Optimal solution to solve the RA-NFV objective function in Eq.(4) under the constraints in Eq.(7)-Eq.(15) implies allocating the virtual functions on the physical server nodes that are capable of meeting the demanded computing resources. Theoretically, this is used to be performed through introducing binary constraints to allocate all VNFs belonging to SFC^r on one physical server node, which is similar to the multi-dimensional Bin Packing problem [15]. Moreover, the optimal solution for Eq.(4) implies also connecting one edge only for each server node, and this is usually treated as a commodity between pairs of nodes, which is similar to finding an optimal flow for the commodity in any network model, and that was proved to be an NP-hard problem and not solvable in polynomial times [22].

Consequently, the majority of RA-NFV approaches followed heuristic or meta-heuristic algorithms to solve the optimization problem in a reasonable polynomial time [10][11]. Therefore, this paper proposes a new RA-NFV power aware algorithm, PaNFV, to solve the offline objective function by fully coordinating allocating the VNF nodes and edges together, while minimizing the power consumption in the network as much as possible. PaNFV applies the segmentation approach introduced in 3 for constructing the best path that will be used to allocate virtual functions and edges, as will be explained in the following paragraphs.

4.6. PaNFV code explained:

Pseudo-code for PaNFV heuristic is shown in Algorithm 1, which is explained as follows:

The PaNFV is designed to work on NFV environments, giving that the physical network is constructed of an interconnected datacenters. The algorithm starts listing all physical paths connecting source-to-server nodes, as well as the paths from server-to-destination nodes. Its assumed that the physical network is fixed, therefore, the main elements formulating

any path, such as number and connectivity of the nodes and edges are also fixed and do not change, but only their capacities varies due to the consumption. This is performed in advance and ahead of handling any SFC. Accordingly, the PaNFV algorithm will always have a full list of all the paths in the network.

In the segmentation and ranking steps 2, 3, and 4 in Algorithm 1, PaNFV formulates SFC^r segment Seg^r, then formulates physical segment Seg^P of a candidate physical path P_{sd}, recalling all physical paths that start and terminate with the Access nodes as requested by the SFC^r. Next, it ranks these paths according to the utilization of their servers, and selects the top ranked physical path P_{sd} and formulates its segment.

The most critical step in PaNFV is to guarantee full coordinated allocations for all VNFs and virtual edges in SFC^r. This is done in step 5, where PaNFV compares each element in the physical segment Seg^P to its counterpart in Seg^r, one-to-one, at the same time, and in one comparison step as shown in the comparison inequalities given by Eq.(16).

Decision matrix for the allocation process is given as follows:

$$\forall\, i^p \in P_{sd} \textbf{ if } cpu_{i^p} - \sum_{r\in R, u\in N^r} cpu_u \geq 0 \textbf{ AND}$$

$$\forall\, ij \in P_{sd} \textit{ and } \forall r \in R \textbf{ if } bw_{ij} - bw^r \geq 0 \textbf{ AND}$$

$$\forall P_{sd} \in P^P \textit{ and } \forall r \in R \textbf{ if } d_{sd} - d^r \leq 0 \tag{16}$$

Moreover, when allocating the virtual nodes, PaNFV strictly allocates all VNFs in SFC^r on the same server only if there is enough capacity as in step 6, otherwise if no other servers can host that SFC it will be rescheduled or dropped. In this paper the SFCs will be dropped in order to be fair when comparing to [15]. This will make sure that all the virtual nodes of the SFC are hosted in one server to speed up the allocation and migration processes, in addition to enhancing the utilization of the server nodes and to use the least physical resources as much as possible. Important to point out that, PaNFV allocates all demanded cores by each VNF on the same server, one after another, regardless the order of the VNFs. The expense of using this mode is going to be consuming more resources from the servers. However, in case VNFs are strictly ordered in the SFC, PaNFV is also capable of consolidating the VNFs and allocate them on the same server, but with less number of cores. This is possible, since PaNFV can determine in advance the required cores for the largest VNFs, and reserve them on the candidate server, so they can be used by other smaller VNFs as well. Regarding allocating the virtual edges, PaNFV checks if the available bandwidth in each edge of the physical segment Seg^P is at least equal to the demanded bandwidth element in Seg^r, bw^r in Eq.(16), and checks if the total delay in the physical candidate path P_{sd}, d_{sd} is less than or equal to the demanded SFC delay d_r.

Finally PaNFV will turn off all non utilized servers as in step 7, then checks if traffic loads are at low profile to trigger the migration algorithm in step 8.

4.7. mPaNFV code explained

The PaNFV triggers the migration phase and calls mPaNFV at two conditions, 1) if the traffic level in the network is very low (i.e, off peak-times), or 2) at any time it detects the utilization of any active server node is below or equal to the migration threshold defined by parameter cpu^{min}. The migration's pseudo-code is shown in Algorithm 2, where mPaNFV lists all allocated SFC^{r}s in a list of candidates for migration and at the same time it lists their hosting physical paths, then it sorts and ranks all these physical paths according to the utilization of their server nodes.

Algorithm 1, PaNFV Pseudo-Code

1. Input: G^P and G^r.
2. **For** each SFC^r formulate SFC^r parameters into segment Seg^r generalizing Eq.(1).
3. **For** the set of all saved physical paths P^P **Do**
   - List all physical paths matching the sources and destination Access nodes as requested by SFC^r.
   - Rank them in descending order based on the consumed cores of the server nodes in these paths.
4. **For** the top ranked path P_{sd}, formulate its segment Seg^P similar to Eq.(2).
5. **Compare** each element in Seg^r against its counterpart in Seg^P
   - Check for CPU and bw and d constraints according to Eq.(16).
6. **If** satisfied, allocate SFC^r on P_{sd}.
   - For P_{sd} server nodes and edges, update CPU and BW resources.
   - Else go to next ranked physical path, step-3.
7. **For** all idle server nodes, turn them off to save power.
8. **For** all active server nodes **Do**
   - If traffic at low profile, or consumed cores in the server are less than or equal to the cpu^{min} **Do**
   - Call the migration code mPaNFV
   - Else Continue
9. Calculate the concerned metrics.
10. **If** SFC^r list is not empty **- Go** to next SFC^r step-2.
11. **End**

Next, it performs three (if) conditional checks, starting by the top ranked physical path and the first SFC^r in the candidates for migration list, if (first condition) the top ranked physical path is currently hosting this candidate SFC^r no migration occurs, and it jumps to the next SFC^r in the list, otherwise, mPaNFV goes to check if (second condition) the server in the top ranked physical path is different from the server currently hosting SFC^r, and if the result of that condition was not true, it jumps to the next ranked path, otherwise it checks (third condition) if the utilization of the server in the top ranked physical path is higher than the utilization of the current server hosting SFC^r, then it decides that migration can take place now.

Next, mPaNFV compares the segment of the selected top ranked physical path Seg^P to the segment of the candidate migrating SFC^r, Seg^r. If the new physical path has enough residual resources to host the migrating SFC^r, mPaNFV

performs the allocation, updates the CPU and BW of the old and new physical paths, then it turns off the old server node to minimize the power consumption. However, if the new physical path does not have enough resources to host the migrating SFC^r, mPaNFV jumps to the next ranked physical path. On the other hand, if there was no physical path to migrate the SFC^r to it, mPaNFV jumps to the next SFC^r in the list for migration. Once there are no more SFC^r to migrate, mPaNFV returns again to the allocation algorithm PaNFV to continue allocating other SFC^r.

Algorithm 2, mPaNFV Pseudo-Code

1. **Input:**
   - List all allocated SFCs.
   - List all physical paths that are currently hosting SFCs.
2. **For** each allocated SFC^r, reformulate its Seg^r Eq.(1).
3. **- List** all physical paths already in use, matching the source and destination access nodes as requested by SFC^r.
   - Rank them in descending order based on the consumed cores of the paths' server nodes.
   - Reformulate Seg^P for the top ranked physical path similar to Eq.(2).
4. **Check if** all of the following conditions are satisfied:
   - If The top ranked physical path does not currently host SFC^r, *AND*
   - If The candidate server in the top ranked physical path is different from the current server hosting SFC^r.
   – If not true Go to the next SFC^r, Step-2.
   - AND If Utilization of the server in the top ranked physical path is higher than that of the server hosting SFC^r.
   – if not true Go to the next physical path, Step-3.
5. **- Compare** Seg^r against Seg^P using Eq.(16).
   - If all inequalities are true, then, migrate SFC^r to the top ranked physical path of Seg^P.
   - Update CPU and BW resources, then turn off old server node
   - Else Go to the next physical path from step-3.
6. **Go** to migrate the next SFC^r, step-2.
7. **If** no more SFC^r to migrate, **Go** back to PaNFV.

4.8. PaNFV Computational Time Complexity

Based on the size of the physical network, PaNFV constructs all types of paths in $O(|N^P| + |E^P|)$ processing time, where the total number of nodes is N^P and edges are given by E^P [22]. This step is performed and saved only once before the arrival of any SFC, and it has no impact on the real computational time complexity of the RA-NFV process. However, to evaluate the computational time complexity of PaNFV, the focal computational component of the heuristic is determined based on the time consumed while sorting all the listed paths in the physical network. Accordingly, for each SFC^r, PaNFV adopted the "Bubble Sort" algorithm to sort and rank all physical network paths in descending order [22], which means that PaNFV algorithm will have a quadratic computational time complexity in the order of $O(n^2)$, where n is number of potential physical paths for allocating or migrating the SFCs.

PaNFV and mPaNFV were developed using Eclipse IDE for Java Developers, version Mars.2 Release (4.5.2), using a computing machine of an i7 processor, 5820K, 12 cores, 3.30 GHz, 16 GB RAM, and the operating system was Ubuntu 16.04.3 LTS.

4.9. Evaluation Metrics

The following evaluation metrics were used to evaluate the performance of the proposed algorithms in this paper.

•Total cost, C^{tot} due to the successfully allocated SFCs :

$$C^{tot} = \sum_{\forall r \in R} (C^{pc} + C^{mig}) \tag{17}$$

•Total Cost of servers' power consumption, $C^{pc^{tot}}$ after all allocated SFCs :

$$C^{pc^{tot}} = \sum_{\forall r \in R} C^{pc} \tag{18}$$

C^{pc} is given as in Eq.(5)

•Total Cost of migrations, $C^{mig^{tot}}$:

$$C^{mig^{tot}} = \sum_{\forall r \in R} C^{mig} \tag{19}$$

C^{mig} is given as in Eq.(6)

•Fraction of dropped SFC bandwidth: is a ratio to represent how PaNFV algorithm is performing in terms of blocking rate due to limited bandwidth resources, and is calculated by dividing the blocked bandwidth demands by the SFCs over the total demanded bandwidth by all SFCs [15]:

$$\forall\, ij \in E^P \;\forall\, uv \in E^r Blocking = \frac{\sum_{r \in R} bw^r - \sum_{r \in R} bw^r x_{ij}^{uv^r}}{\sum_{r \in R} bw^r} \tag{20}$$

•servers utilization represents the scalability trend after all simulation iterations. Its defined as ratio between consumed CPU cpu_i, and maximum CPU resources.

$$CPU_{util} = \sum_{\forall i \in N^P} \frac{(CPU_i - cpu_{i^p})}{CPU_i} * 100) \tag{21}$$

5. Simulation settings, Results and Discussions

In this section the settings and results of four experiments are shown to evaluate the performance of the proposed algorithms in this paper. The first experiment studies impact of varying idle to maximum power ratio on the total costs of the allocation algorithm PaNVF with migrations. It compares the original PaVNF developed by [2] against the allocation and migration

algorithms RVPP/RLARCDP from [15]. The second experiment focuses on the modified PaNFV and keeps its migration strategy always active, and studies the impact of activating and not activating VNFs consolidations on the total costs, server's utilizations, and allocation times. Moreover, the third experiment keeps VNFs consolidation always active in all allocation algorithms, and studies the impacts on the allocation times when migrations were turned-off, as in PaNFV, and compares that to PaNFV/mPaNFV. Finally, the fourth experiment focuses on speed of allocation times using the original PaNFV, which does not include the migration option, but uses VNFs consolidation and virtual machines consolidations together. It evaluates the allocation times, and compares total costs and servers' utilizations performance when VNFs consolidations were activated and not activated.

5.1. Settings

The simulations of PaNFV without migrations and PaNFV including the migration extension mPaNFV, were conducted against the best performing allocation and migration algorithms provided by [15]. The same settings used to test RVPP/RLARCDP were also used for PaNFV and mPaNFV, as well as using the same SFCs and physical network topologies. The physical network used in the simulation is shown in Fig.2, which includes 4 interconnected datacenters, each of them has 16 servers, formed in fat-tree topology, and each server has 48 cores. To evaluate the impacts of varying the idle to maximum power ratio, the cost results of PaNFV/mPaNFV were compared to those of the global policy results in [15] when the fraction of server's idle to maximum powers defined by the variable a was set equal to $(0.3, 0.5, 0.7, 1)$, giving that P^{idle} is calculated using $a * P^{Busy}$, and $P^{Busy} = 1000$ Watts.

5.2. Traffic Model

To characterize semi-daily traffic profile and to be fair when comparing to [15], it is assumed that the bandwidth demands of the SFCs were scaled according to the traffic patterns of a cyclic-stationary profile, where the changes on traffic are predictable and reoccur on periodical and regular basis. Eq.(22) suggested by [15] is also used in this paper for fair comparisons, and it provides the mathematical formula for the scale factor (α_h) controlling the proposed daily traffic profiles counted by traffic intervals h for $(Q = 24)$ daily periods. The values of α_h varies over $(h = 0, 1, ..., Q-1)$, where $\alpha_0 = 1$, denotes the scale factor at peak traffic conditions, and $\alpha_{\frac{Q}{2}} = \alpha_{min}$, for the least traffic condition that triggers the migrations. Accordingly, PaNFV will evaluate each of the mentioned constraints above at each interval h and calculates the evaluation metrics.

Table 2 lists all simulation settings used to analyze the performance of PaNFV with and without mPaNFV, against the algorithms from [15], and the results will be discussed in some details in the following paragraphs.

$$\alpha_h = \begin{cases} 1, & \text{if } h = 0 \\ 1 - 2\dfrac{h}{Q}(1 - \alpha_{min}), & \text{if } h = 1, ..., \dfrac{Q}{1} \\ 1 - 2\dfrac{Q-h}{Q}(1 - \alpha_{min}), & \text{if } h = \dfrac{Q}{2} + 1, ..., Q - 1 \end{cases} \tag{22}$$

5.3. Experiment-1: Impacts of varying idle to maximum power ratio on total costs

The main objective of this experiment is to compare PaNFV with migration algorithm mPaNFV against the original PaNFV with no migration strategy, and against the routing and VNF placement problem algorithm (RVPP), including the revenue loss aware resource consolidation/de-consolidation algorithm (RLARCDP) for migrations as suggested by [15].

As an overall overview, the difference between PaNFV/mPaNFV and RVPP/RLARCDP is that PaNFV allocates the SFCs one after another on the most utilized servers, and mPaNFV checks to migrate the allocated SFCs from the least utilized servers to other highly utilized ones, then PaNFV turns off all non utilized servers afterwards. However, in [15] RVPP allocates the SFCs based on their demanded bandwidths in decreasing order and on the least stressed servers. While in RLARCDP, it selects some of the already allocated SFCs and checks the ones that, if migrated, would minimize the network operation costs. In RLARCDP, the migrations are chosen by global policy in RVPP, which relies on the knowledge of the entire daily traffic profile in advance. The global policy uses cyclic-stationary traffic pattern and considers a priori chosen offline allocations.

To understand the impacts of different idle to maximum power ratios, Fig.3 shows that when idle to maximum power ratio increases, the only obvious impact on the overall performance of PaNFV and PaNFV/mPaNFV was the increase in the levels of total costs as the a values go higher, which takes values of $a = 0.3, 0.5, 0.7, 1$. However, performance of RVPP/RLARCDP experienced very different behavior as the a ratio increases as observed from both, power consumption costs and migration costs too. The following paragraphs will discuss impacts in more details.

5.3.1. Total Costs

Total costs sums up the costs of the servers' power consumptions and the costs of migrations while allocating the arriving SFCs, in which each of them has only one VNF randomly picked from the defined set of VNFs. This should reflect the overall efficiency of each of the compared algorithms when the idle to maximum power ratio changes from 0.3 to 1, and accordingly rule out the best performing algorithm in pure terms of costs units. Fig.3:a,d,g,j shows the total costs of PaNFV without migrations, PaNFV with migrations using mPaNFV, and RVPP/RLARCDP from [15], which resulted on PaNFV without migration has outperformed the PaNFV with migrations by 35.9% points on average across all a values, and at the same

Table 2: Simulation settings to evaluate PaNFV and mPaNFV for different idle to maximum power ratios.

Scenario	Offline.	Network	Shown in Fig.2.
servers' cores	48 cores/server.	Edges' bandwidth	10 Gbps server-Switch, 40 Gbps else.
SFCs topology	SFCs of one VNF.	Number of SFCs	500, or 1500 for blocking simulations.
SFCs bandwidth	Random (100, 150, 200, 250, 300) Mbps.	VNFs types	Random, FW, or IDS, or EV.
VNFs cores	Random, $FW = 4$, or $IDS = 8$, or $EV = 4$.	t_u^{proc} per VNF type	120, 160, 82.67 μsec.
Δt	1 hour.	T_{down}	2 sec.
β_d	$1.8E^{-7} - 2.7E^{-6}$	β_e	1
Simulation Runs	10	P^{idle}	$a * P^{Busy}$.
L^u	1500 Bytes.	a	(0.3, 0.5, 0.7, 1)
T^u	Equal to SFC bandwidth.	P^{Busy}	1000 Watts.

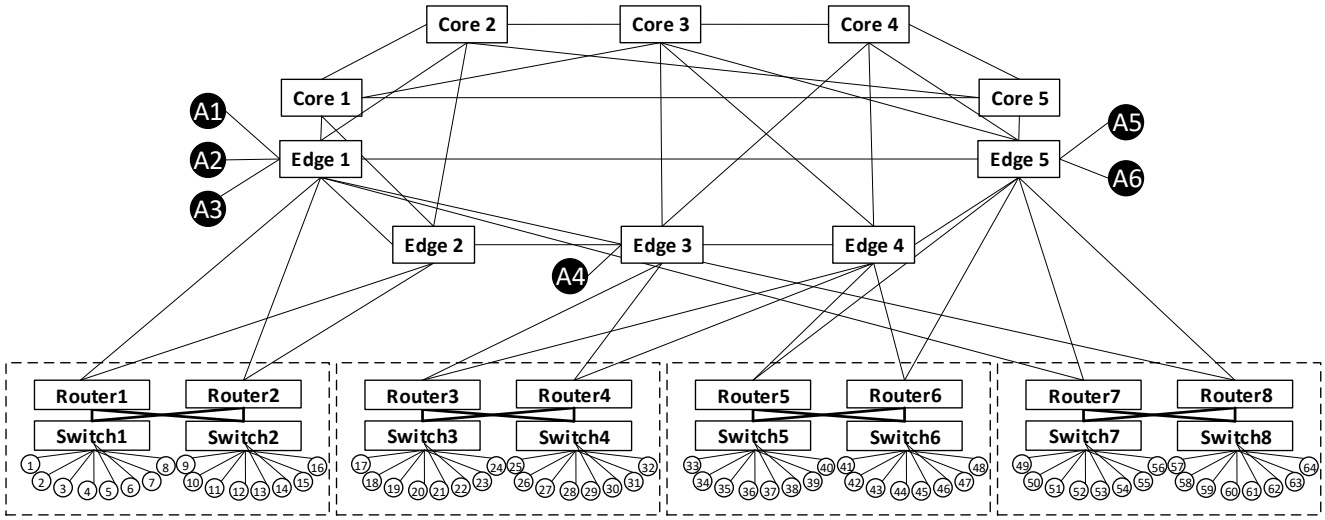

Figure 2: As in [15], Physical network of four datacenters located in separate geographical areas.

time outperformed [15] for the same a values by 32.1% points. For example refer to the results from Fig.3:a,d,g,j, where on average the total costs of PaNFV/mPaNFV were lower than those of RVPP/RLARCDP by by 54% for $a = 0.3$, 39% for $a = 0.5$, 27% for $a = 0.7$, and 9% for $a = 1$. However when PaNFV without migrations was compared to RVPP/RLARCDP, it outperformed it by 58% for $a = 0.3$, 42% for $a = 0.5$, 30% for $a = 0.7$, 13% for $a = 1$. Noting that in Fig.3j, both PaNFV without migrations and PaNFV/mPaNFV gave higher costs up until the cost per Gbit lost given by β_d reached $7.2E^{-7}$, where the migration costs of RLARCDP started to increase dramatically most likely due to its tendency to activate much more servers as a values increase, causing the total costs of [15] to become higher than PaNFV and PaNFV/mPaNFV.

Therefore, the main outcome from the results of total costs indicate that the superior performance of PaNFV over RVPP is because it did not include migration costs at all, yet it had almost very near results as when the PaNFV and migration algorithm was activated.

5.3.2. Power consumption Costs

Regarding the results of the power consumption costs as shown in Fig.3:b,e,h, they indicate that PaNFV and PaNFV/mPaNFV have nearly flat costs as a function of the cost of Gbits lost, and resulted on less costs than RVPP by 55% for $a = 0.3$, 36% for $a = 0.5$, and 17% for $a = 0.7$, thanks to the use of the segmentation technique which guaranteed precise allocations and less activated servers. However, in Fig.3k when $a = 1$, both PaNFV with no migrations, and PaNFV/mPaNFV resulted on more power consumption costs than RVPP by 16.4%, mainly because PaNFV/mPaNFV does not consider future migrations in advance as the case used by RVPP, which places the virtual functions and edges of the SFCs on the least stressed servers, or directly on the servers that have available resources. Notice that the savings by RVPP will be translated into much more migration costs as will be seen in next section, and that will increase its overall costs compared to PaNFV.

Nevertheless, the advantage of PaNFV with and without migrations is based on its precise allocation strategy in the first place, which will help in avoiding excessive migrations in the future. To clarify more, PaNFV carefully allocates the virtual functions and edges together using the segmentation technique,

on the path of the most active (utilized) server, before activating any new servers, and keeps all other non-active ones turned off. In this way it activates the physical resources one-by-one based on the loads, and does not allow to use other resources unless the current active ones were fully utilized.

5.3.3. Migration Costs

Referring to the migration costs shown in Fig.3:c,f,i,l, mPaNFV had almost negligible migration costs, mainly because of the precise allocations by PaNFV, which used the segmentation technique to select the most fit physical resources that guarantee consuming as least resources as possible in the first place. Notice that, the PaNFV using the segmentation technique ranks the physical paths based on the utilization of their physical servers, therefore, it ensures to consume the full capacities of the active servers before activating new ones. In this way, PaNFV utilizes the physical resources as efficient as possible, and when mPaNFV is triggered, it may find some servers that were not fully utilized, accordingly, it can identify some migration attempts that mostly can fit small SFCs. This is shown in Fig.3:c,f,i,l, were the number of identified migrations were very limited, which resulted on very minor costs.

However, focusing on the results of RLARCDP migration algorithm as extracted from [15], it had much higher migration costs compared to those of mPaNFV, most probably because of the relaxed allocation strategy used by the RVPP algorithm, causing the use of more physical resources, thus the more migrations and costs. It is important to notice that, since RVPP allocates the virtual functions first on any server that has enough residual resources to host the VNFs, then allocates the virtual edges in another separate phase using the shortest path algorithm, that may result on lack of coordination between the two phases, therefore, raising the possibilities of using more longer paths than needed, and consequently keeping more active servers. That what most likely forced the migration algorithm, RLARCDP to increase the number of migrations to minimize the number of active servers, resulting on additional migration costs, and therefore, increased the total overall costs.

Overall, in light of the different allocation and migration strategies used by PaNFV and PaNFV/mPaNFV than those from [15], the results of PaNFV costs were much lower than those from the global policy used by RVPP on average, and the migration algorithm mPaNFV will always result on much lower costs than RLARCDP.

5.3.4. Understanding Blocking Results

To evaluate the blocking probability of PaNFV, it is performance was compared against the allocation algorithm RVPP by [15], and the VNF placement algorithm developed by [14] as extracted from [15]. Important to clarify that, in real world networks, SFCs should not be blocked, but can be rescheduled for future allocation attempt. However, since RVPP and VNF placement algorithm allow SFC blocking, and for the sake of comparison with them, blocking is also allowed by the algorithms in this paper.

The conducted simulations to test PaNFV blocking probability used the same network topology as in [15], injecting 1500 SFCs, each of a single VNF demanding 4 cores if it is a FW or EV, or 8 cores if its an IDS, and varied the demanded bandwidths randomly between $(100, 150, 200, 250, 300)$ Mbps. The results of PaNFV against RVPP and Bari's placement algorithm are shown in Fig.4, presenting the fraction of dropped bandwidth demanded by the rejected SFCs as a function of the total demanded bandwidth by all SFCs. All algorithms, PaNFV, RVPP and Bari placement heuristic, allocate the core resources for any VNF of any SFC on a single server, giving that PaNFV and Bari's algorithm will try to utilize the most loaded servers first before activating the new ones, but RVPP distributes the loads on the physical servers uniformly.

PaNFV uses the segmentation technique to construct the path that will host the SFC, coordinating the allocation of the virtual functions and the virtual edges together, while RVPP allocates the VNFs on the physical servers first, then uses the shortest path to allocate the virtual edges on the physical counterparts connecting the physical servers hosting the virtual functions. Similarly, VNF placement algorithm by [14] works in two phases when allocating the demands of the SFCs, it allocates the virtual functions on the servers first using Viterbi algorithm according to specific VNF deployment and power costs, then uses the shortest path in terms of delay to connect these servers with edges that fulfill certain values, such as costs of forwarding the allocated VNF's traffic, and penalty in case of that edge is violating service level agreement conditions.

From Fig.4, it can be seen that, on average PaNFV performance was better than RVPP by 7% and 37% than the VNF Placement algorithm of [14], thanks to PaNFV's precise allocations, accepting all bandwidths until nearly 800 SFCs, then PaNFV slightly lagged behind RVPP when $800 - 1050$ SFCs were allocated, but was better than RVPP and VNF Placement algorithm for SFCs between $1050 - 1500$. However, the high blocking results of Bari's VNF placement algorithm would most likely be due to Bari's strategy on choosing the most loaded servers in separate phase than the edges, which may have caused some SFCs to be rejected and resulting on worse performance than PaNFV and RVPP up until it handled 1000 SFCs, where it had similar performance as PaNFV and RVPP did.

Overall, PaNFV allocation strategy using the segmentation technique on the most utilized paths, provided solid and stable performance close to that from RVPP, which applies vertical scaling technique to uniformly utilize the physical servers' processing and edges' bandwidth resources.

5.4. Experiment-2: Impacts of VNFs consolidations on allocation times of PaNFV with migrations

The main objective of this experiment is to evaluate the performance of PaNFV/mPaNFV when migrations are always allowed, but with and without the option of VNFs consolidation during the allocation process.

5.4.1. VNFs consolidation

in this paper, PaNFV was designed to allocate all the VNFs from an SFC^r on the same server only if there is enough capacity, otherwise if no other servers can host that SFC it will be

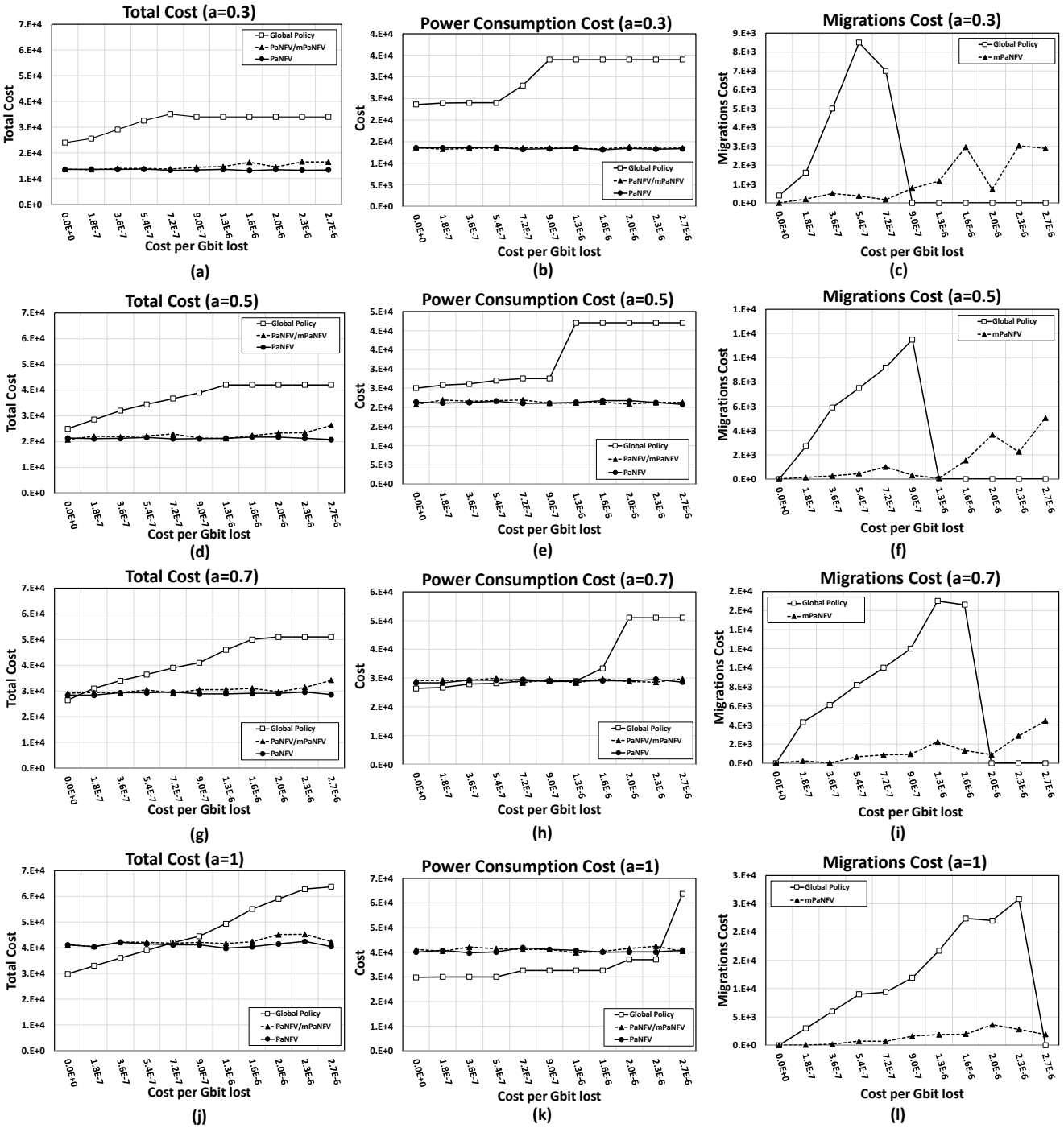

Figure 3: PaNFV/mPaNFV compared to RVPP/RLARCDP using Global Policy for different idle to maximum power values as set by a. In (a,d,g,j), the average total costs of PaNFV and PaNFV/mPaNFV were lower than Global Policy, while their power consumption costs were higher than RVPP on average as shown in (b,e,h,k). And the migration algorithm mPaNFV in (c,f,i,l) resulted on lower values than those of RLARCDP.

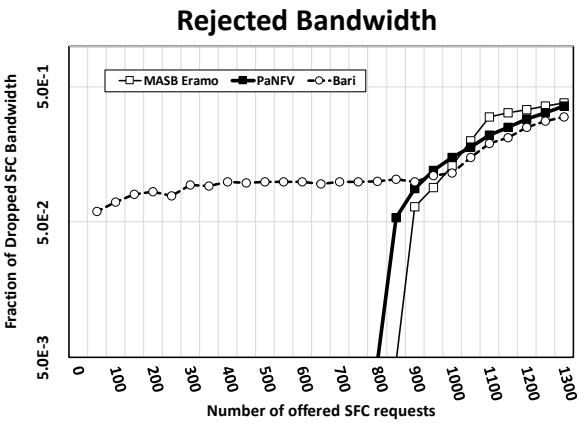

Figure 4: Fraction of Blocked bandwidth for PaNFV, RVPP from [15], and the VNF Placement algorithm of [14].

rescheduled or dropped. This will make sure that all the virtual nodes of the SFC are hosted in one server to speed up the allocation and migration processes, in addition to enhancing the utilization of the server nodes, minimizing impacts of delays, and guaranteeing to use the least physical resources as much as possible.

In this experiment, when allocating the VNFs on a server, PaNFV can work in two modes,

a) The non-consolidated VNFs mode, where PaNFV allocates all demanded cores for each VNF from the SFC on the same server at the same time, and keeps them allocated as long as the SFC demands. For example, if an SFC has two VNFs, "FW" demands 4 cores, and "IDS" demands 10 cores, then PaNFV will find a server that has 14 free cores to host that SFC, and keeps them reserved as long as requested by the SFC. This is the generic and classic way of allocating the VNFs, and therefore is applicable if the VNFs strictly demand to be allocated according to specific order, or if some of them need to work on parallel with other VNFs, or if no allocation order is required. The expense of using this mode will be translated into slower speed of allocations and increased utilizations of the physical resources, but it offers more guarantees to the required quality of service for the allocated SFC.

b) The consolidation VNFs mode, where PaNFV scans the VNFs of an SFC and identifies the one that has the largest demanded cores to be allocated. Accordingly, PaNFV selects a physical server that has enough capacity to host the demanded cores by that VNF and reserves them for that SFC as a whole. For example, in the above SFC of the "FW" and "IDS" VNFs, the IDS VNF demands the largest core resources, therefore, PaNFV will find a server that can host 10 cores only, which will be used for both the FW and the IDS. This is used in case the VNFs are strictly ordered, that is, given the 10 cores are already reserved, and FW is ordered to be allocated a head of the IDS, then, PaNFV will allocate the FW of the 4 cores first, and once it expiries, PaNFV will allocate the IDS using all the 10 cores afterwards. The benefits of this mode are to increase the efficiency of the servers' utilizations by allocating more SFCs, and to minimize the allocation time as much as possible. The drawbacks are in case the VNFs are not strictly ordered, which means the operation of some VNFs need to go in parallel, or if the operation of the VNFs is dependent on each other, and therefore keeping them active together. In that case non-consolidated mode is triggered again.

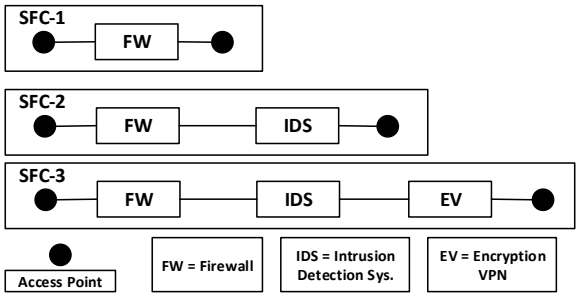

Figure 5: SFCs topologies for experiments 2 and 3

Accordingly, if non-consolidated mode is allowed, cpu_u in Eq.(16) will be the sum of the demanded cores by all VNFs in the SFC, otherwise cpu_u will be just the demanded cores by the largest VNF in the SFC if VNFs consolidated mode is activated.

Simulation settings for experiment-2 are the same as those in the first experiment, but using the SFCs as shown in Fig.5, and for idle to maximum servers' power ratios of 0.3 and 1.

5.4.2. Discussions for the results of experiment-2

Referring to Fig.6:a,d, the average total costs of PaNFV/mPaNFV for the non-consolidated mode are higher by 27.88% than the consolidated mode for all a values, which reflects the tendency of PaNFV/mPaNFV on activating more servers than the consolidated mode. On server's utilizations, Fig.6:b,e shows that PaNFV/mPaNFV on non-consolidated mode utilizes the physical servers faster than the consolidated mode, which had negative consequences on the amount of accepted SFCs by 38.48% lower than the consolidated mode. However, when considering the allocation times of PaNFV/mPaNFV, Fig.6:c,f shows that, on average the consolidated mode consumes more time to allocate a single SFC by a ratio of 1.54 than the non-consolidated mode, mainly because of the availability of more free resources in the consolidated mode to accept more SFCs, which forces the PaNFV/mPaNFV to invest more time when searching for the right allocations.

Conclusions form experiment-2 are two folds, first given this is an offline scenario and migrations were kept active in both modes, the only obvious outcome of activating migrations in both cases was on the high total allocation times, which were around 604 ms in non-consolidated modes, and 988 ms in consolidated mode. These are very high times in both cases and exceed the requirements for 5G networks [4]-[6]. Therefore, the use of migrations should only be considered at off-peak traffic conditions, and if online scenario is considered, then carefully use it and better trigger it for emergency conditions. Second,

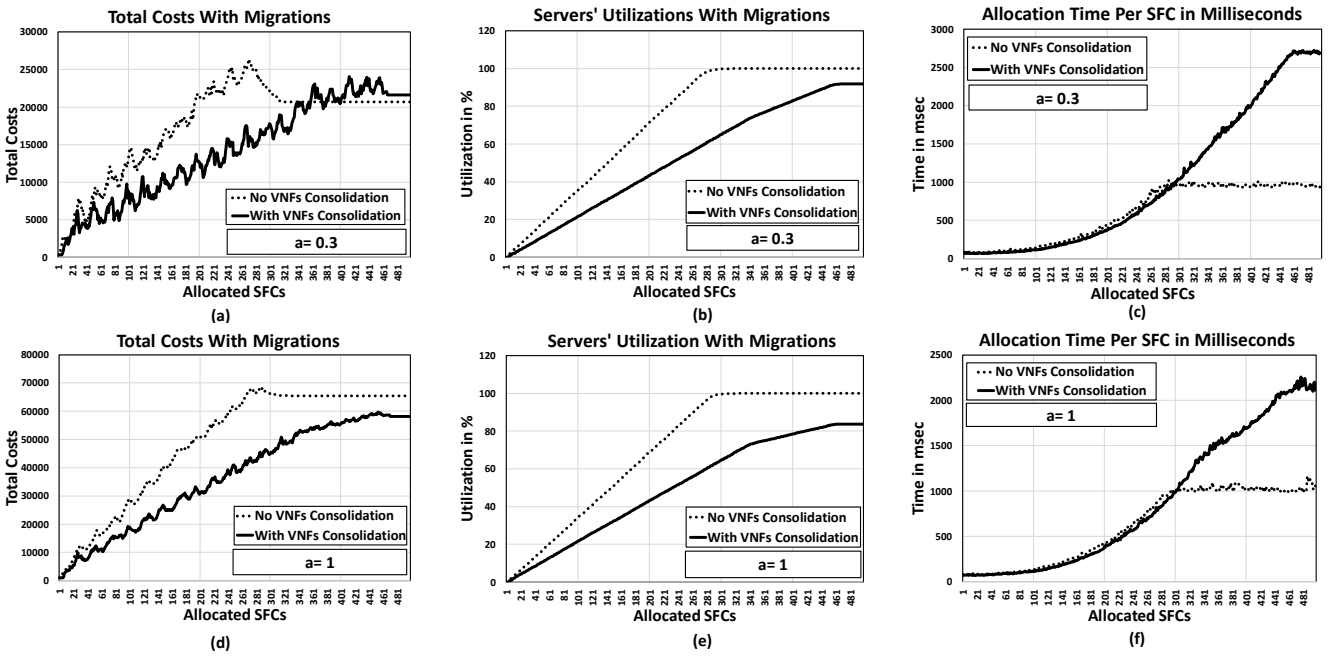

Figure 6: Impacts of activating VNFs consolidations when migrations are always used

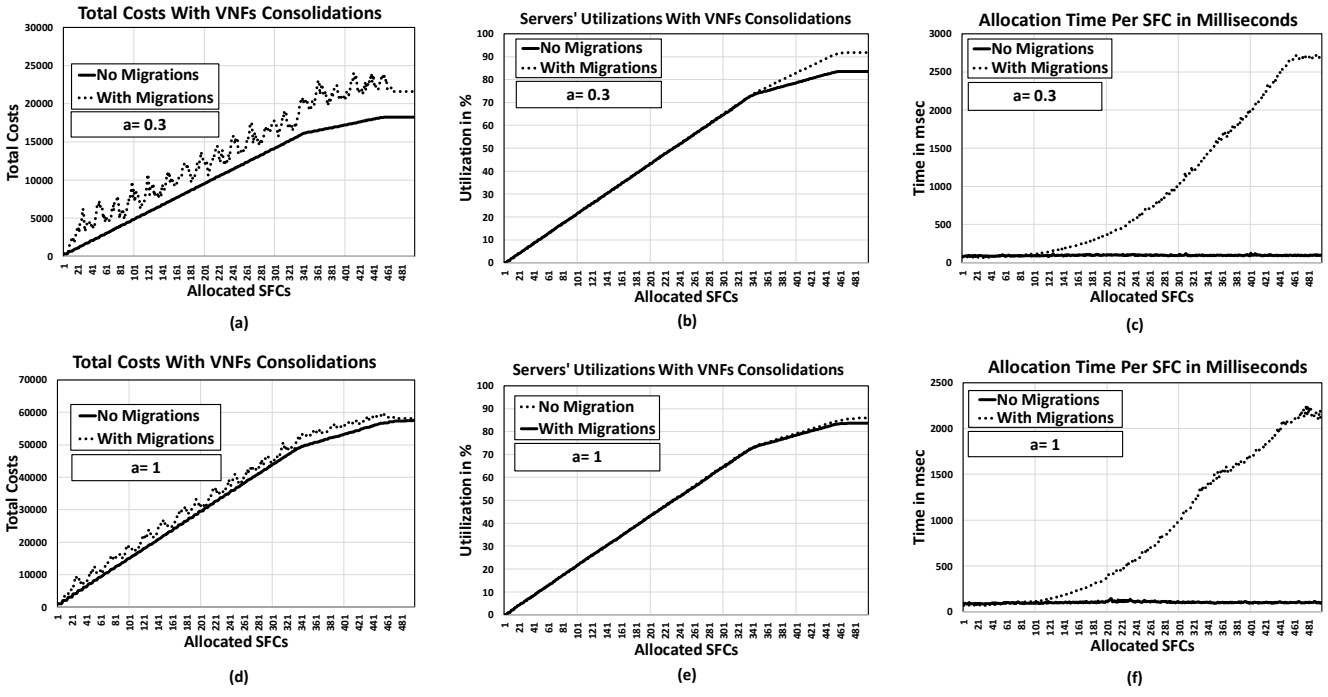

Figure 7: Impacts of activating Migrations when VNFs consolidations are always used

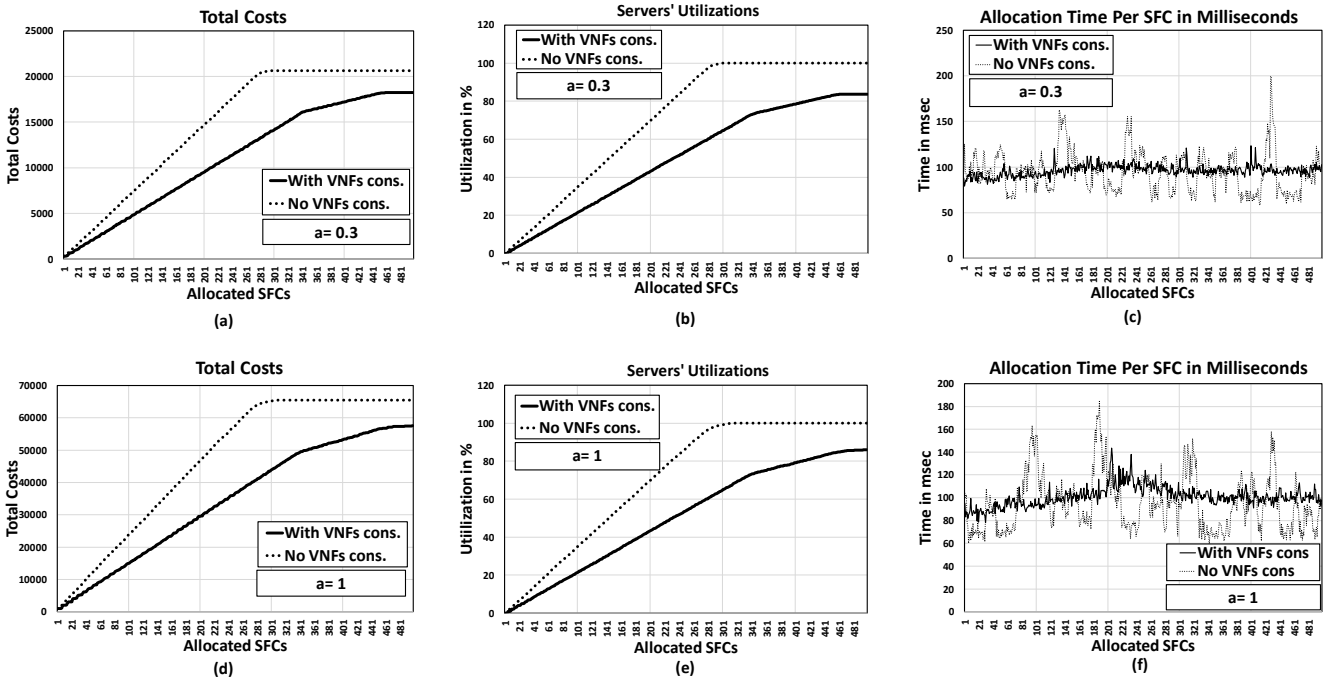

Figure 8: Impacts of VNFs consolidations on the original PaNFV with no migrations at all

the experiment shows that, in terms of accepting more SFCs and generating more revenues, the consolidated mode is much preferred than the non-consolidated mode, conditional that, PaNFV/mPaNFV strictly deals with SFCs of ordered VNFs.

5.5. Experiment-3: Impacts of Migrations and VNFs consolidations on allocation times

In this experiment the main objective is to evaluate the overall performance of PaNFV in terms of allocation times, but with and without the migration option. In both cases the option of VNFs consolidations was kept active, as well as all simulation settings of experiment-2.

Regarding allocation times, Fig.7:c,f shows that PaNFV without migrations clearly outperformed PaNFV/mPaNFV, and was on average faster by 9.59 times when allocating any single SFC across all a values. For example, in Fig.7:c, PaNFV managed to allocate the SFCs on averages of 96 ms, compared to 988 ms in case of migrations were allowed, and the main reason for that is that when migrations are not activated, PaNFV focused more on direct allocations of the SFCs without waisting anytime in tracking, searching, and migrating any allocated SFC again. Notice that the allocation times for PaNFV/mPaNFV increases with each allocated SFC, since the algorithm spends more time on evaluating the benefits of migrations, if any, and that takes time when the network has more allocated SFCs.

Moreover, the total costs as shown in Fig.7:a,d indicate that PaNFV without migrations had lower values by 14.06% than PaNFV/mPaNFV, mainly because it did not include any migration costs. However, with reference to server's utilizations shown in Fig.7:b,e, PaNFV/mPaNFV had more utilizations than PaNFV without migrations by 1.71% on average, which indicate that including migrations did manage to slightly free more physical resources to be used for allocating other SFCs, and that means an increase in the acceptance ratio and accordingly the revenues as well.

In conclusion for experiment-3, PaNFV alone without migration proved huge and significant advantageous in terms of allocations times, and less total costs, and that should be an attractive feature for future 5G networks, where speed of allocations is at its core center stage. In addition, in some conditions when the allocation times could be tolerated, then PaNFV with its migration strategy could be beneficial, especially for emergency or maintenance needs.

5.6. Experiment-4: Impacts of VNFs consolidations on allocation times of the PaNFV without migrations

In this experiment the main objective is to evaluate impacts of activating VNFs consolidation on the allocation times of original PaNFV which does not include migration option. The same simulation settings of experiment-2 are also used in this experiment.

Regarding allocation times, Fig.8:c,f shows that the original PaNFV without migrations and without VNFs consolidations was on average faster than PaNFV without migrations but with VNFs consolidations. More specifically, results of PaNFV with no VNFs consolidations shown in Fig.8:c for $a = 0.3$ were slightly lower by 1.7% on average than the case when VNFs consolidations were allowed, scoring an allocation time of 94.43 ms compared to 96.12 ms. Similar trends are also shown in Fig.8:f for $a = 1$, which indicate that allocation times of PaNFV with no VNFs consolidations were lower by 6.2%

on average than the case of VNFs consolidations, scoring an allocation time of 94.67 ms compared to 100.96 ms.

However, the total costs and servers' utilizations shown in Fig.8:a,d and Fig.8:b,e indicate that PaNFV without VNFs consolidations had on average much higher values than PaNFV with VNFs consolidations. To clarify more, the total costs results of PaNFV with no VNFs consolidations shown in Fig.8:a for $a = 0.3$, were higher by 33.8% on average than the case when VNFs consolidations were allowed, and same trends are shown in Fig.8:d for $a = 1$, indicating that PaNFV with no VNFs consolidations had more costs by 36% than the case when VNFs consolidations was used.

Regarding results of servers' utilizations of PaNFV with no VNFs consolidations as shown in Fig.8:b for $a = 0.3$, they were higher by 41.57% on average than the case when VNFs consolidations were allowed, and in Fig.8:e for $a = 1$, PaNFV utilizations with no VNFs consolidations were higher by 40.3% than the case when VNFs consolidations was allowed.

These results reconfirm PaNFV results from experiment-2 in terms of allocation speeds without migrations when VNFs consolidations were not activated. But it increased total costs and servers' utilizations, since PaNFV without migrations and without VNFs consolidations tends to consume more physical resource faster, which on the other side may ultimately result on decreasing the acceptance ratio and revenues as well.

6. Conclusions

This paper proposed a modified offline power aware resource allocation algorithm for NFV based networks. The new allocation algorithm solves the resource allocation problem in fractions of a second, supports physical servers consolidations combined with virtual network functions consolidations, minimizes total costs in the datacenters, and comes with an optional migration strategy that can be triggered according to specific conditions and at anytime. The simulation results of the algorithm with migrations managed to outperform other algorithms by 32.1%, mainly because the proposed algorithm was designed to allocate the service function chains on the least number of physical resources in the first place.

In addition to that, the simulation results of using virtual network functions consolidations together with physical servers consolidations showed that, the total costs and servers' utilization when VNFs consolidations was allowed were 27.88% and 38.48% better than not consolidating the VNFs. Finally, to evaluate the resource allocation times, results from this paper showed that, the performance of the allocation algorithm without traffic migrations was faster by 9.59 times than when including migrations, and that was without any significant impacts on the total power consumptions or servers' utilizations.

Overall, the proposed power aware allocation algorithm without including migrations managed to significantly reduce total power consumptions, and proved to be a very reliable choice for allocating networks' services in fractions of a second. For the migration version, it did not show any significant impact on reducing the total power consumptions, mainly because the proposed offline algorithm managed to use the least resources during the allocation process, and therefore this paper concluded that migrations would be better used for emergency or maintenance purposes.

In future work, segmentation technique could be conducted on other types of networks using large and distributed cloud of core and edge datacenters for example, and more tests could be also done about the impacts of the last mile delays when using resource allocation process on a dynamic access network. Moreover, the segmentation technique can be modified by injecting some machine learning technologies considering load balancing, agile path classifications, and traffic prediction for more efficient resource allocation process.

7. Acknowledgment

This work has been partially supported by the Ministerio de Economy Competitividad of the Spanish Government under project TEC2016-76795-C6-1-R and AEI/FEDER, UE.

References

[1] Khaled Hejja, Xavier Hesselbach, "Power aware coordinated virtual network embedding with 5G delay constraint," Journal of Network and Computer Applications, Elsevier, 2018. Article reference: YJNCA2228.

[2] Khaled Hejja, Xavier Hesselbach, "Offline and online power aware resource allocation algorithms with migration and delay constraints," Computer Networks, Volume 158, 2019, Pages 17-34. DOI:10.1016/j.comnet.2019.04.030

[3] ITU-T Focus Group, "IMT-2020 Deliverables," 2017. www.itu.int.

[4] ITU, "Setting the Scene for 5G: Opportunities and Challenges," 2018. www.itu.int.

[5] 5G Americas, "5G Americas White Paper: Cellular V2X Communications Towards 5G," 2018. www.5gamericas.org

[6] ITU, "ITU-T Y.3001: SERIES Y: Global Information Infrastructure, Internet protocol Aspects and Next-Generation Networks. Future networks: Objectives and design goals," 2011. www.itu.int.

[7] 3GPP TR 28.801 (V15.0.0), "Study on management and orchestration of network slicing for next generation network," 2017. portal.3gpp.org

[8] ETSI GS NFV 002 v1.1.1, "Network Function Virtualisation (NFV); Architectural Framework," 2013. www.etsi.org

[9] ETSI GS NFV 004 v1.1.1, "Network Function Virtualisation (NFV); Virtualization Requirements," 2013. www.etsi.org

[10] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. De Turck and R. Boutaba, "Network Function Virtualization: State-of-the-Art and Research Challenges," in IEEE Communications Surveys and Tutorials, vol. 18, no. 1, pp. 236-262, 2016. https://ieeexplore.ieee.org/document/7243304/

[11] J. Gil Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," in IEEE Transactions on Network and Service Management, vol. 13, no. 3, pp. 518-532, 2016. DOI:10.1109/TNSM.2016.2598420

[12] J. F. Riera, X. Hesselbach, M. Zotkiewicz, M. Szostak and J. F. Botero, "Modeling the NFV forwarding graph for an optimal network service deployment," 2015 17th International Conference on Transparent Optical Networks (ICTON), Budapest, pp. 1-4, 2015. DOI:10.1109/ICTON.2015.7193483

[13] J. F. Riera, X. Hesselbach, E. Escalona, J. A. Garcia-Espin and E. Grasa, "On the complex scheduling formulation of virtual network functions over optical networks," 2014 16th International Conference on Transparent Optical Networks (ICTON), Graz, pp. 1-5, 2014. DOI:10.1109/ICTON.2014.6876564

[14] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba and O. C. M. B. Duarte, "Orchestrating Virtualized Network Functions," in IEEE Transactions on Network and Service Management, vol. 13, no. 4, pp. 725-739, 2016. DOI:10.1109/TNSM.2016.2569020

[15] V. Eramo, E. Miucci, M. Ammar and F. G. Lavacca, "An Approach for Service Function Chain Routing and Virtual Function Network Instance Migration in Network Function Virtualization Architectures," in IEEE-ACM Transactions on Networking, vol. 25, no. 4, pp. 2017. 2008-2025. DOI:10.1109/TNET.2017.2668470

[16] C. Pham, N. H. Tran, S. Ren, W. Saad and C. S. Hong, "Traffic-aware and Energy-efficient vNF Placement for Service Chaining: Joint Sampling and Matching Approach," in IEEE Transactions on Services Computing, 2017. DOI:10.1109/TSC.2017.2671867

[17] O. Soualah, M. Mechtri, C. Ghribi and D. Zeghlache, "Energy Efficient Algorithm for VNF Placement and Chaining," 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Madrid, pp. 579-588, 2017. https://ieeexplore.ieee.org/document/7973745/

[18] Kim, S., Park, S., Kim, Y., "VNF-EQ: dynamic placement of virtual network functions for energy efficiency and QoS guarantee in NFV," Cluster Computing, vol 20, issue 3, pp 2107-2117, 2017. DOI:10.1007/s10586-017-1004-3

[19] B. Kar, E. H. K. Wu and Y. D. Lin, "Energy Cost Optimization in Dynamic Placement of Virtualized Network Function Chains," in IEEE Transactions on Network and Service Management, vol. PP, no. 99, pp.372-386, 2018. DOI:10.1109/TNSM.2017.2782370

[20] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso, "Power provisioning for a warehouse-sized computer," In Proceedings of the 34th annual international symposium on Computer architecture (ISCA '07). ACM, New York, NY, USA, 2007. 13-23.DOI:10.1145/1250662.1250665

[21] M. Dayarathna, Y. Wen and R. Fan, "Data Center Energy Consumption Modeling: A Survey," in IEEE Communications Surveys and Tutorials, vol. 18, no. 1, pp. 732-794, 2016.. DOI:10.1109/COMST.2015.2481183

[22] J. Kleinberg and E. Tardos, "Algorithms Design," Addison-Wesley, 2009.

[23] Z. Meng, J. Bi, H. Wang, C. Sun and H. Hu, "CoCo: Compact and Optimized Consolidation of Modularized Service Function Chains in NFV," 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, pp. 1-7, 2018. DOI:10.1109/ICC.2018.8422641

[24] M. A. Khan, A. P. Paplinski, A. M. Khan, M. Murshed and R. Buyya, "Exploiting user provided information in dynamic consolidation of virtual machines to minimize energy consumption of cloud data centers," 2018 Third International Conference on Fog and Mobile Edge Computing (FMEC), Barcelona, pp. 105-114, 2018. DOI:10.1109/FMEC.2018.8364052

[25] Xu S., Wu C.Q., Hou A., Wang Y., Wang M. "Energy-Efficient Dynamic Consolidation of Virtual Machines in Big Data Centers," In: Au M., Castiglione A., Choo KK., Palmieri F., Li KC. (eds) Green, Pervasive, and Cloud Computing. Lecture Notes in Computer Science, vol 10232. Springer, 2017. DOI: 10.1007/978-3-319-57186-7

[26] F. Zhang, G. Liu, X. Fu and R. Yahyapour, "A Survey on Virtual Machine Migration: Challenges, Techniques, and Open Issues," in IEEE Communications Surveys and Tutorials, vol. 20, no. 2, pp. 1206-1243, 2018. DOI: 10.1109/COMST.2018.2794881

[27] M. Noshy, A. Ibrahim, and H. Arafat Ali, "Optimization of live virtual machine migration in cloud computing: A survey and future directions," Journal of Network and Computer Applications, Volume 110, pp. 1-10, 2018. DOI: 10.1016/j.jnca.2018.03.002

[28] M. C. Silva Filho, Claudio C. Monteiro, Pedro R.M. Inácio, Mário M. Freire, "Approaches for optimizing virtual machine placement and migration in cloud environments: A survey," Journal of Parallel and Distributed Computing, Volume 111, pp. 222-250, 2018. DOI: 10.1016/j.jpdc.2017.08.010

[29] Antonio Marotta, Stefano Avallone, Andreas Kassler, "A Joint Power Efficient server and Network Consolidation approach for virtualized data centers," Computer Networks, Volume 130, 2018, Pages 65-80. DOI:10.1016/j.comnet.2017.11.003.

[30] Mohammad-Hossein Malekloo, Nadjia Kara, May El Barachi, "An energy efficient and SLA compliant approach for resource allocation and consolidation in cloud computing environments," Sustainable Computing: Informatics and Systems, Volume 17, 2018, Pages 9-24. DOI:10.1016/j.suscom.2018.02.001.

[31] Hui Yan, Xiaomin Zhu, Huangke Chen, Hui Guo, Wen Zhou, Weidong Bao, "DEFT: Dynamic Fault-Tolerant Elastic scheduling for tasks with uncertain runtime in cloud," Information Sciences, Volume 477, 2019, Pages 30-46. DOI:10.1016/j.ins.2018.10.020

[32] Luo Gangyi, Qian Zhuzhong, Dong Mianxiong, Ota Kaoru, Lu Sanglu. "Improving performance by network-aware virtual machine clustering and consolidation," The Journal of Supercomputing, 2018, vol. 74, no. 11, pp 5846-5864. DOI:10.1007/s11227-017-2104-9

[33] X. Zhu, J. Wang, H. Guo, D. Zhu, L. T. Yang and L. Liu, "Fault-Tolerant Scheduling for Real-Time Scientific Workflows with Elastic Resource Provisioning in Virtualized Clouds," in IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 12, pp. 3501-3517, 2016. DOI:10.1109/TPDS.2016.2543731